Iranian Journal of Science & Technology, Transaction B, Engineering, Vol. 30, No. B2
Printed in The Islamic Republic of Iran, 2006

A PROPOSED QUERY-SENSITIVE SIMILARITY MEASURE FOR INFORMATION RETRIEVAL*

M. ZOLGHADRI-JAHROMI[1**] AND M. R. VALIZADEH[2]

[1]Dept. of Computer Science and Engineering, Shiraz University, Shiraz, I. R. of Iran
Email: zjahromi@shirazu.ac.ir
[2]Azad University of Ilam, Ilam, I. R. of Iran

Abstract– Document clustering has been widely used in information retrieval systems in order to improve the efficiency and also the effectiveness of ranked output systems using clustering hypothesis. Based on this hypothesis, documents relevant to a query tend to be highly similar in the context defined by the query. In this way, a pair of documents has an overall similarity (ignoring the query) and a specific similarity (similarity of a pair of documents given a query). A Query-Sensitive Similarity Measure (QSSM) is a mechanism to measure the similarity of two documents given a query. In this paper, in the first step, we identify the sources of information that may be used for this purpose. In the second step, we propose a QSSM based on these information sources. Finally, we propose a parametric QSSM that simultaneously makes use of the product and weighted sum to fuse the information from the identified sources. A genetic algorithm is used to learn the optimal values of parameters in this measure for a specific collection. The leave-one-out method is used to evaluate the proposed learning scheme. Our motivation for this is to see whether the learning scheme can perform significantly better than the measure proposed in the second step. Using several document collections, the performance of each measure is evaluated and the results are compared with other QSSMs proposed in the past research.

Keywords– Query sensitive similarity measure, document clustering, genetic algorithm

1. INTRODUCTION

An information retrieval (IR) system is characterized by a set of documents and users who submit their queries to the system in order to find the information they require. Generally, the system returns both relevant and non-relevant documents to the user. Two most commonly used approaches to assist the users in finding the relevant information are ranked list and clustering of retrieved documents.

Clustering is a process by which an object space is partitioned into groups that are related in some context. It has been applied to the field of IR for improving the efficiency [1] and the effectiveness [2] of these systems.

Generally, a search engine presents the retrieved documents as a ranked list. The documents are ranked according to the probability of being relevant to a user's query. The highest ranked document is considered as the most likely relevant document. It is assumed that the user will examine the documents starting at the top of the list one by one.

It has been shown that the cluster hypothesis also holds in a subset of documents retrieved in response to a query [3]. Based on this hypothesis, documents relevant to a given query tend to be more similar to each other than non-relevant ones, and therefore tend to appear in the same cluster. The clustering can then be applied to the document subset retrieved by a conventional ranked list IR system before presenting

*Received by the editors June 29, 2005; final revised form March 1, 2006.
**Corresponding author

the results to the user. This can improve the effectiveness of a ranked list system by retrieving documents that have otherwise been ranked low through inter-document association (clustering) with other relevant documents.

The idea of QSSM was first introduced by Tombros and Van Rijsbergen [4]. According to this, a pair of documents has an overall similarity, which is fixed under all queries submitted to the IR system and a specific similarity, which is variable under different queries submitted to the IR system (i.e. a function of query). In IR, we are interested in the relationship between documents under the context defined by the query. The aim of QSSM is to provide a mechanism for calculating such a similarity.

In this paper, in the first step, we identify three sources of information that may be used in defining a QSSM and propose a QSSM based on these sources. In the next step, we generate new information sources using product as the fusion operator. The QSSM is then defined as the weighted sum of all possible sources of information, and search capability of the Genetic Algorithm (GA) is used to learn the best combination of weights in this measure (using the relevance feedback for a query subset of a specific document collection). We use the leave-one -out method to assess the performance of the learning scheme. The aim of the second step is to see if this adaptive scheme can develop QSSMs that perform significantly better than the measure that was proposed in the first step.

This paper is organized as follows. In section 2, we give a brief survey of past research in the field of applying GA and Genetic Programming (GP) in IR. In section 3, we present the basic idea of QSSM along with past research in this subject. In section 4, the proposed QSSM is presented. In section 5, an adaptive scheme for QSSM (using GA) is discussed. In section 6, the detail of the experimental setup and simulation results are presented. Section 7 concludes this paper.

2. APPLICATION OF GA AND GP IN IR

GA and GP are problem-solving techniques based on the principles of evolution. These parallel global search algorithms are widely used to solve optimization and approximation problems [5].

Recently, considerable research has been carried out to apply GA and GP to the field of IR for improving the performance of these systems [6]. One early work in this area was carried out by Gordon [7]. He used GA to develop an adaptive method for document indexing. In this, keywords used to represent various documents are altered over time as the user interacts with the system. In this way, documents are represented by keywords which can be matched with keywords in user queries.

In a similar approach, Yang and Korfhage [8] applied GA to change the representation of documents using relevance judgments provided by the user. Instead of selecting keywords to represent a document, they used GA to alter the weight assigned to each keyword in representing the document. They reported improved performance of the system as the weights of keywords for representing documents improved in the process of evolution.

Pathak *et al*. [9] applied GA to adaptively change the matching function for a document collection. In their research, a linear combination of several well-known matching functions (i.e. Dice, Jaccard, cosine, etc.) was used as the overall matching function. They used GA to search for an optimal combination of these matching functions for a specific collection. A subset of documents of known relevance was used as input to GA to find the best combination of weights for the final matching function.

Fan *et al*. [10] continued their research in this field by applying GP to generate an optimal term weighting formula on a per query basis. Using a query and a subset of documents known to be relevant to that query, they used GP to evolve an optimal weighting formula for that specific query.

Cummins and O'Riordan [11] used GP to develop weighting formulas that operate well for all queries in a specific document collection rather than finding optimal weighting formulas on a per query basis.

The research reported in this paper is an attempt to use GA in developing a QSSM using relevant feedback. The QSSM is defined as a linear combination of multiple sources of information. GA is used to search for an optimal combination of weights in this measure for a specific collection.

3. QUERY SENSITIVE SIMILARITY MEASURES

In any clustering method, a similarity measure is needed to calculate inter-object similarities. In document clustering, traditionally, the similarity of each pair of documents is calculated off-line, in the form of a large matrix, before the submission of a query to the system. This form of clustering is sometimes called *static* because the similarity of two documents is independent of the query and remains constant under different queries submitted to the system. One of the most common measures of this type is the cosine similarity measure, which can be stated as:

$$\cos(D, X) = \frac{\sum_{k=1}^{n} d_k x_k}{\sqrt{\sum_{k=1}^{n} d_k \sum_{k=1}^{n} x^k}} \quad (1)$$

Where D and X are document vectors, d_k and x_k are the weights of the k-th index term in documents D and X, respectively and n is the number of index terms.

The idea of *query-specific clustering* was first introduced by Peerce [12]. In this approach, clustering is applied to the search results of a conventional ranked output IR system (i.e. top-n retrieved documents retrieved in response to the query). The difference in applying the clustering to a subset of documents and not to the whole collection is that the term weights of documents will change according to the distribution of terms in the subset (rather than distribution of terms in the full collection). Suppose that both documents D and X are retrieved in response to queries Q1 and Q2. The similarity of the two documents (D,X) will be different under the context of different queries (i.e. Q1,Q2), because the representations of both documents in each case depends on other documents in the subset (retrieved in response to query).

Hearst and Pedersen [3] examined the performance of this form of clustering in IR. The results of their experiments confirm that the relevant documents of a query tend to appear in the same clusters.

Tombros and van Rijsbergen [13] investigated the effectiveness of query-specific hierarchic clustering in IR. They applied hierarchical clustering to the search results of a conventional ranked output IR system. After performing a set of experiments on five document collections using four hierarchical clustering methods, they reported that query-specific clustering can increase the retrieval effectiveness compared both to that of static clustering and conventional ranked output IR systems.

Tombros and Van Rijsbergen introduced the idea of QSSM [4]. According to this, inter-document similarity is dynamic, and changes explicitly depending on the query. For a given query (Q), they proposed the following two formulas for calculating the similarity of two documents (D1, D2) based on the cosine similarity measure.

$$Sim(D_1, D_2 / Q) = \cos(C, Q) \quad (2)$$

$$Sim(D_1, D_2 / Q) = \cos(D_1, D_2)\cos(C, Q) \quad (3)$$

Where, C is a vector containing common terms of documents D_1 and D_2. The weight of a common term in C is the average of its weights in D_1 and D_2. The weights of terms not common between the documents are set to zero. Actually, in this research [4], another QSSM is presented which can be expressed as:

$$Sim(D_1, D_2 / Q) = \theta_1 \cos(D_1, D_2) + \theta_2 \cos(C, Q) \quad (4)$$

Where θ_1 and θ_2 are weighting factors in the unit interval and $\theta_1 + \theta_2 = 1$. The appropriate values of parameters θ_1 and $\theta 2$ (actually, it is sufficient to specify their ratio) are different across different collections. A learning scheme such as the one we propose in section 5 should be used to find the optimal values of parameters for a specific collection. As we shall see, our proposed parametric measure presented in section 5, covers this measure as a special case.

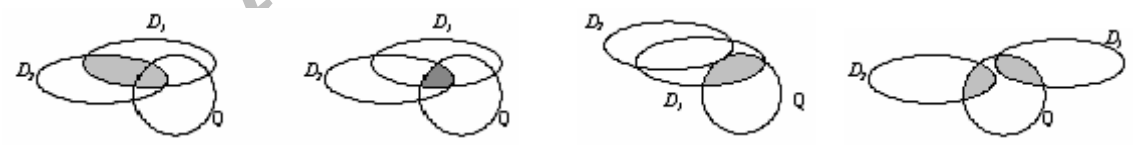
Using these QSSMs (Eqs. (2-4)), they performed several experiments to investigate the effectiveness of these measures using the N-Nearest Neighbor test [14]. In this test, for each relevant document of a specific query, the five nearest neighbors are found using equations (1)-(4). The average number of relevant documents in the neighborhood averaged over all relevant documents, and all queries in the collection were used as the effectiveness of the similarity measure. They used a SMART IR system [1] to perform the initial retrieval using tf-idf weighting for document and query terms. The top-n retrieved documents were used as the document subset. The documents in the subset were then represented (re-weighted) according to the distribution of terms in the subset. In this way, the QSSMs are, in fact, using both implicit and explicit information to calculate inter-document similarities and only in the case where n=full-collection size, the explicit form of QSSM is used. After a series of experiments on five document collections and across different values of n, they reported that:

- Measures (3) and (4) are significantly more effective than the cosine coefficient.
- Changing the ratio of parameters θ_1 and θ_2 in measure (4) can significantly affect the performance of the measure for a specific collection.
- None of the measures proves to be the best across different collections.

Notice that using product (Eq. (3)) and weighted sum (Eq. (4)) for combining the information from the two sources (X_1 and X_2) proves to be effective in defining a QSSM. Our parametric QSSM presented in section 5 makes simultaneous use of product and weighted sum to define the measure.

4. PROPOSED QSSM

The aim of a QSSM is to calculate the similarity of two documents (D_1, D_2) given a query (Q). In order to explain possible cases that can happen and find the sources of information that may be used for this purpose, we provide some examples in Fig. 1. The documents and queries are presented as sets of terms that constitute them. The overlap area between two sets corresponds to common terms between the sets, which is proportional to the number of common terms. In what follows, we try to justify the sources of information that can be used for calculating the similarity of a pair of documents under the context of query.



(a) (b) (c) (d)

Fig.1. Information sources for query sensitive similarity measure

The area highlighted in Fig.1a corresponds to common terms between the documents ignoring the query. Since the query can be expressed using different terms and the occurrence of query terms is not essential (although a good indication of relevance) for the documents to be relevant to the query, this is a source of information that we use in defining our QSSM. We call this source X_1 and calculate the overlap area using cosine coefficient ($X_1 = cos(D_1, D_2)$).

The second source of information is the number of common terms that are also the query terms (highlighted area in Fig.1b). We call this source X_2 and calculate the overlap area using cosine coefficient (X_2= *cos(C,Q)*). Where C is the vector representation of $D_1 \cap D_2$, which is used in Eqs. (2-4).

The third source of information corresponds to common terms between the query and each of the documents. Consider a situation in which the two documents have a large number of common terms (i.e. high overall similarity) and only one of them has a good number of query terms (Fig.1c). Now, if one of the documents is actually relevant to the query, we prefer that the QSSM give a large value for the situation just described (more likely both documents are relevant to the query). This is our third source of information which can be calculated using the cosine coefficient as: $X3= cos(D_1,Q)+ cos(D_2,Q)$. Other forms of calculating this source such as: Max($cos (D_1,Q),cos(D_2,Q)$) and Min($cos(D_1,Q), cos(D2,Q)$) were tried, but no significant differences were observed. Measure X_3 was used in all experiments reported in this paper.

One should notice that none of the above information sources on its own can provide good evidence that the two documents are similar in the context of a query. Consider the examples presented in parts (c) and (d) of Fig.1. Assuming that measure X_3 is the same for both of these cases, the documents in part (c) are more likely relevant (X_1 is large), while the documents in part (d) are more likely not related (X_1=0.). Combining the two information sources in the form of X_1X_3 or X_1+X_3 will separate the two mentioned cases. Our proposed QSSM is based on using $X_1+ X_3$ for this purpose, which can be represented as:

$$Sim(D_1, D_2 / Q) = 1/3[\cos(D_1, D_2) + \cos(D_1, Q) + \cos(D_2, Q)] \tag{5}$$

By treating the query exactly like a short document vector and considering D_1, D_2 and Q to form a cluster, Eq. (5) uses the average similarity between elements of the cluster to calculate the similarity of two documents under the context of query. As we are interested in relative values of similarities rather than their absolute values, we can equivalently use average pair-wise similarity between elements of the cluster (including self-similarity) instead of Eq. (5). Now, if documents and query are represented by unit length vectors and denoting the centroid of the cluster as $C=(D_1+D_2+Q)/3$, it can be easily shown [15] that the length of the centroid vector is the square root of average pair-wise similarity (including self-similarity) between elements of cluster. That is, instead of Eq. (5), the following measure can be used to reduce the computation time.

$$Sim(D_1, D_2 / Q) = \| C \|^2 \tag{6}$$

Where $\|C\|$ represents the length of the centroid vector.

5. GA FOR ADAPTIVE QSSM

Our aim in this section is to provide a mechanism to derive the most effective QSSM using the information sources identified in the last section. Since none of these sources on its own can provide good evidence that two documents are similar in the context of a specific query, some form of combining the information from these sources is required to define a QSSM. We propose using the following equation as the combiner function.

$$\begin{aligned} Sim(D_1, D_2 / Q) = {} & w_1 X_1 X_2 X_3 + w_2 X_1 X_2 + w_3 X_1 X_3 + w_4 X_2 X_3 + \\ & w_5 X_1 + w_6 X_2 + w_7 X_3 \end{aligned} \tag{7}$$

This equation provides a general scheme for combining the information from the identified sources that simultaneously make use of product and weighted sum to fuse the information from the mentioned

sources. This form of combining multiple sources of information has been used in the field of IR in the past. For example, Tombros and Van Rijsbergen [4] used product and weighted sum to combine the information from multiple sources (Eqs. (3) and (4)). Pathak *et al*. [9] used a linear combination of four popular similarity measures in IR to present an overall similarity measure.

Each of the weights w_1-w_7 *has a* range from 0.0 to 1.0. A higher weight indicates that the associated term is more important than a term with a lower weight. It is hypothesized that by a proper combination of weights, it should be possible to achieve a QSSM that is superior compared to that produced by individual sources of information in Eq. (7). The aim of GA is to search for an appropriate combination of weights. This is a multi-dimensional search for an optimum combination of weights.

Note that Eq. (7) covers all the QSSMs discussed in this paper as a special case. For example, by setting w_1, w_2, w_3, w_4 and w_6 equal to zero, Eq. (7) reduces to (4). Obviously, with (4) being a special case of (7), using GA to learn the parameters of (4) instead of (7) is not advised.

Since the numbers of queries in all IR text collections are relatively small, Savoy *et al*. [16] propose the leave-one-out evaluation scheme, which is a special case of the cross validation method. The scheme works as follows. For a test collection with N queries, the queries are divided into N subsets, each containing N-1 queries. The i-th set contains all queries except i-th query (which is used for test). Using these sets for training and testing, N experiments are performed to evaluate the learning algorithm. In this way, the learning algorithm uses nearly all the queries to adjust its parameter settings and all the queries will be used to measure the performance of the proposed learning scheme.

We use GA to learn the parameters of measure (7). That is, in the leave-one-out evaluation, for a collection with M queries, a QSSM is evolved using the relevance feedback from M-1 queries in the training set. The performance of the evolved measure on the single test query is calculated using the N-Nearest Neighbor test. In other words, M different QSSMs evolve to assess the performance of the scheme on the full collection.

6. EXPERIMENTAL RESULTS

In order to evaluate the performance of the different QSSMs discussed, we use CRAN, MED, LISA and CACM document collections. The statistics of these collections are given in Table 1.

Table 1. Statistics of document collections

Data set	CACM	LISA	MED	CRAN
Number of documents	3204	6004	1033	1400
Mean term per document	22.5	39.7	51.6	56
Number of queries	52	35	30	225
Mean term per query	13	19.4	9.9	9
Mean relevant docs per	15.3	10.8	23.2	8
Total relevant documents	796	379	694	924

For the purpose of initial retrieval, the default SMART stop list [17] was used to remove stop-words. Porter stemmer, which is the most commonly used stemmer in English, were used for stemming the remaining words. Assignment of term weights for documents and queries were then performed using tf-idf term weighting (Eq. (8)).

$$W_{ij} = \frac{tf_{ij} \ln(n / df_j)}{\sqrt{\sum_{k=1}^{t} (tf_{ik} \ln(n / df_k))^2}} \tag{8}$$

Where t is the number of index terms, tf_{ij} *is* the frequency of term j in document i, df_j is the document frequency of term j and n is the number of documents in the collection.

The initial retrieval was performed using the cosine similarity measure. The top-n ranked documents (we used the values of n=100, 200, 400, 800 and full-collection size in the experiments) were used as the document subset under investigation. Using the same weighting scheme as the initial retrieval, the vector representation of documents was modified according to the distribution of terms in the subset.

In the leave-one-out evaluation method, the performance of each QSSM (i.e. the evolved measure and other measures) on the single test query was measured by N-Nearest Neighbor test (we used 5-NN test in our experiments). That is, for each relevant document in the retrieved subset, we find the five nearest neighbors and count the number of relevant documents in that neighborhood. A single value corresponding to the number of relevant documents in the neighborhood, averaged over all the relevant documents present in the retrieved subset, gives the performance of QSSM for that query. The larger this number is, the higher the separation of relevant from non-relevant documents. The overall performance of the leave-one-out evaluation method can be represented by a single value when averaging over all queries in a collection. This value is calculated and displayed in the results reported in this section (Tables 2-5).

As the purpose of QSSM is to place the documents relevant to a query in the same cluster, the fitness function used to evaluate an individual in a GA population attempts to achieve this goal. Each individual in the GA population represents a candidate QSSM. To evaluate an individual, for each query in the training set and each document relevant to that query, N nearest neighbors of the document is found. The fitness of an individual is simply the number of relevant documents in the neighborhood averaged over all relevant documents of each query and all queries in the training set.

Regarding GA, various genetic operators used in the experiments are as follows:

- *Selection and reproduction*: All individuals in the previous generation were made available for reproduction in the next generation. The roulette-wheel reproduction process [18] was used to select individuals for reproduction.
- *Crossover*: A two-point crossover was used. A parameter 'cross-over rate' determined the number of individuals that actually mate.
- *Mutation*: Mutation was implemented by introducing Gaussian noise.
- *Process termination*: The genetic modification process was terminated after a preset number of generations or after convergence when no improvement in performance was observed for 10 generations.

Experiments were run for 50 generations with 50 individuals in each generation. Crossover and mutation rates were set to 0.6 and 0.1, respectively.

Tables 2-5 give the result of our experiments for the data sets used in this paper. For each data set, leave-one-out evaluation of our learning scheme is reported by the result of 5-NN test. The results for different values of top-n retrieved documents are reported in these tables. In each case, we also report on 5-NN test results for our proposed QSSM (5), other QSSMs proposed in the past research (Eqs. (2) and (3)) and the cosine similarity measure. As seen, the performance of the learning scheme is very close to our proposed measure in all data sets and across different values of top-n retrieved documents. This implies that of all seven information sources used in Eq. (7), only X_1 and X_3 are effective sources for defining a QSSM. In particular, when X_1 and X_3 are used to define the QSSM, X_2 is redundant.

In Tables 2-5, the values in parenthesis show the increase in performance when our proposed QSSM is compared with Eq. (3), which is reported to be more effective than (2). The improvement observed in the vast majority of cases is over 10%, which confirms the significance of the results [19]. It must be

noted that for CACM collection, 1587 documents without body text and 12 queries without any relevant documents were removed from the collection.

Table 2. Leave-one-out evaluation of CRAN collection (5-NN test results)

CRAN	Cosine Eq. (1)	QSSM Eq. (2)	QSSM Eq. (3)	Evolved QSSM Eq. (7)	Proposed QSSM Eq. (5)
top100	1.615	0.801	1.699	1.862	1.882 (10.73%)
top200	1.560	0.733	1.709	1.933	1.938 (13.38%)
top400	1.492	0.694	1.685	1.964	1.956 (16.13%)
top800	1.435	0.676	1.641	1.965	1.959 (19.40%)
full	1.390	0.660	1.600	1.956	1.960 (22.50%)

Table 3. Leave-one-out evaluation of MED collection (5-NN test results)

MED	Cosine Eq. (1)	QSSM Eq. (2)	QSSM Eq. (3)	Evolved QSSM Eq. (7)	Proposed QSSM Eq. (5)
top100	3.439	2.596	3.480	3.614	3.623 (4.09%)
top200	3.264	2.456	3.411	3.581	3.616 (6.00%)
top400	3.091	2.397	3.351	3.592	3.598 (7.39%)
top800	2.905	2.341	3.258	3.587	3.580 (9.88%)
full	2.850	2.230	3.140	3.593	3.580 (14.0%)

Table 4. Leave-one-out evaluation of LISA collection (5-NN test results)

LISA	Cosine Eq. (1)	QSSM Eq. (2)	QSSM Eq. (3)	Evolved QSSM Eq. (7)	Proposed QSSM Eq. (5)
top100	1.088	0.717	1.188	1.404	1.394 (17.37%)
top200	1.034	0.607	1.174	1.427	1.416 (20.66%)
top400	1.009	0.573	1.180	1.431	1.425 (20.68%)
top800	0.937	0.552	1.170	1.337	1.430 (22.31%)
full	0.760	0.520	1.140	1.406	1.410 (23.68%)

Table 5. Leave-one-out evaluation of CACM collection (5-NN test results)

CACM	Cosine Eq. (1)	QSSM Eq. (2)	QSSM Eq. (3)	Evolved QSSM Eq. (7)	Proposed QSSM Eq. (5)
top100	0.266	0.068	0.180	0.197	0.201 (11.72%)
top200	0.370	0.081	0.254	0.287	0.294 (15.59%)
top400	0.531	0.053	0.283	0.453	0.438 (54.91%)
top800	0.658	0.039	0.306	0.592	0.573 (87.02%)
full	0.830	0.530	1.130	1.824	1.800 (59.29%)

7. CONCLUSIONS

In this paper, we proposed a QSSM using the information sources that we identified for this purpose. Through a set of experiments that measures the degree at which QSSMs place relevant documents at close proximity to each other, we demonstrated that our proposed QSSM is significantly more effective than other measures proposed in the past.

We also proposed an adaptive scheme to learn the most effective QSSM for a specific collection. The leave-one-out evaluation of the learning scheme failed to perform better than our proposed QSSM.

The collections we used for the evaluation of various QSSMs are very topic specific and very small for IR standards. It must be acknowledged that this can have an impact on the generality of the results. Further experiments on large heterogeneous collections such as trec are required to evaluate the proposed QSSM.

The next step in evaluating the performance of the proposed QSSM is to use this measure in practice for cluster-based retrieved. The effectiveness of the measure can be analyzed based on the quality of the emerged clusters.

REFERENCES

1. Salton, G. (1971). *The SMART retrieval system-experiments in automatic document retrieval.* Englewood Cliffs, New Jersey: Prentice Hall Inc.
2. Baeza-Yates, R. & Ribeiro-Neto, B. (1999). *Modern information retrieval.* Addison-Wesley.
3. Hearst, M. A. & Pedersen, J. O. (1996). Re-examining the cluster hypothesis: scatter/gather on retrieval results. In *Proceedings of the 19th Annual ACM SIGIR Conference*, 76-84, Zurich, Switzerland.
4. Tombros, A. & van Rijsbergen, *C. J. 2004.* Query-sensitive similarity measures for information retrieval. *Knowledge and Information Systems*, *6*(5), 617-642.
5. Back, T., Fogel, D. B. & Michalewicz, Z. (1997). *Handbook of evolutionary computation*. IOP Publishing and Oxford University Press.
6. Cordon, E., Herrera-Viedma, C., Lopez-Pujalte, M., Luque, P. & Zarco, C. (2003). A review on the application of evolutionary computation to information retrieval. *International Journal of Approximate Reasoning*, *34*(2-3):241-264.
7. Gordon, M. (1988). Probabilistic and genetic algorithms in document retrieval. *Communications of the ACM 31*(10), 1208–1218.
8. Yang, J. & Korfhage, R. R. (1993). Query improvement in information retrieval using genetic algorithms: A report on the experiments of the trec project. *Proceedings of the 1st Text Retrieval Conference.* 31–58, New York, USA.
9. Pathak, P., Gordon, M. & Fan, W. (2000). Effective information retrieval using genetic algorithms based matching functions adaptation. *Proceedings of the 33rd Hawaii International Conference on System Science.* 103-107, Maui, Hawaii.
10. Fan, W., Gordon, M. D. & Pathak, P. (2000). Personalization of search engine services for effective retrieval and knowledge management. *Proceedings of the 2000 International Conference on Information Systems.* 20–34, Brisbane, Queensland, Australia.
11. Cummins, R. & O'Riordan, C. (2004). Determining general term weighting schemes for the vector space model of information retrieval using genetic programming. *15th Artificial Intelligence and Cognitive Science Conference,* Galway-Mayo Institute of Technology, Castlebar, Ireland.
12. Peerce, S. E. (1973). Clustering as an output option. *Proceedings of American Society for Information Science*, *10*, 189-190, Dallas, Texas.
13. Tombros, A., Villa, R. & van Rijsbergen, C. J. (2002). The effectiveness of query-specific hierarchical clustering in information retrieval. *Information Processing & Management, 38*(4), 559-582.
14. Voorhees, E. M. (1985). The effectiveness and efficiency of agglomerative clustering in document retrieval. Ph.D. thesis, *Technical Report TR 85-705*, Department of Computing Science, Cornell University.
15. Shankar, S. & Karypis, G. (2000). Weight adjustment schemes for a centroid-based classifier. *TextMining Workshop, KDD 2000*, Boston, MA, USA.

16. Savoy, J. & Vrajitoru, D. (1996). Evaluation of learning schemes used in information retrieval. *Technical Report CR-I-95-02*, Faculty of Sciences, University of Neuch atel.
17. Salton, G. (1989). *Automatic text processing*. Addison-Wesley.
18. Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Reading M.A. Addison-Wesley.
19. Keen, E. M. (1992). Presenting results of experimental retrieval comparisons. *Information Processing & Management*, *28*(4), 491-502.