Applied Intelligence 18, 243–256, 2003

Genetic Mining of HTML Structures for Effective Web-Document Retrieval

SUN KIM AND BYOUNG-TAK ZHANG
Biointelligence Laboratory, School of Computer Science and Engineering, Seoul National University, Seoul 151-742, Korea
skim@bi.snu.ac.kr
btzhang@bi.snu.ac.kr

Abstract. Web-documents have a number of tags indicating the structure of texts. Text segments marked by HTML tags have specific meaning which can be utilized to improve the performance of document retrieval systems. In this paper, we present a machine learning approach to mine the structure of HTML documents for effective Web-document retrieval. A genetic algorithm is described that learns the importance factors of HTML tags which are used to re-rank the documents retrieved by standard weighting schemes. The proposed method has been evaluated on artificial text sets and a large-scale TREC document collection. Experimental evidence supports that the tag weights are well trained by the proposed algorithm in accordance with the importance factors for retrieval, and indicates that the proposed approach significantly improves the performance in retrieval accuracy. In particular, the use of the document-structure mining approach tends to move relevant documents to upper ranks, which is especially important in interactive Web-information retrieval environments.

Keywords: genetic algorithms, machine learning, web-documents, information retrieval

1. Introduction

The increasing amount of information on the Web raises new and challenging problems for information retrieval [1]. Web search engines have their ancestors in the information retrieval systems developed during the last fifty years. These engines use IR methods such as Boolean, vector space, probabilistic, and clustering models [2]. However, most of Web-documents are written in HTML (HyperText Markup Language) which consists of tags used for making structure. HTML documents exhibit two kinds of structures not present in plain text documents [3].

1. One is the internal structure consisting of typed text segments marked by HTML tags. HTML defines a set of roles to which text in a document can be assigned. Some of these roles are related to formatting, such as those defining bold and italic text. Others have richer semantic imports such as headlines and anchors, the text segments which serve as hyperlinks to other documents.
2. The other is the external structure. As a node in a hypertext, an HTML page is potentially related to a huge number of other pages, through both the hyperlinks it contains and the hyperlinks that point to it from other pages.

Recent works in information retrieval on the Web are mainly concerned with hyperlink structures (i.e. external structure) [4–7], rather than the tag information. They assume that citations signify deliberate judgment by the page author. For example, if there is a link from page a to page b, then they assume that the author of page a recommends page b and links often connect related pages. Spertus [8] observed that co-citation can indicate that two pages are related. That is, if page a points to both pages b and c, then b and c might be related. Chakrabarti et al. [9] use the links and their order to categorize Web pages. They show that the links that are near a given link in page order frequently point to pages on the same topic. An example site using hyperlink information is Google [10]. Google makes use of the link structure of the Web to calculate a quality

ranking (PageRank) for each Web page. A page can have a high PageRank if there are many pages that point to it, or if there are some pages that point to it and have a high PageRank. The basic idea using the internal structure of Web-documents is shown at the Lycos search engine [11]. The engine considers the word positions such as title, body, and header in Web-documents for determining a document-relevance score. Recent works using HTML structures consider the tag importance factors to improve retrieval performance [13, 12]. Boyan et al. implemented the LASER system, which offers a number of parameters that influence the ranking of retrieval results [3]. The parameters affect how the retrieval function responds to words in HTML fields, how hyperlinks are incorporated, how partial-word matches or query-term adjacency are adjusted and more. Given the parameters, they applied a simulated annealing to optimize the retrieval function.

In this paper, we propose a method to learn the internal structure of HTML documents using genetic algorithms for improving retrieval performance. This approach is based on the assumption that plain texts are semantically organized by HTML tags in Web-documents. Genetic algorithms are generally quite effective for rapid global search to find solutions in non-deterministic problems. We first present a similarity measure that incorporates tag weights in addition to the standard weighting schemes. After that, the importance factors for the HTML tags are learned on training data using the proposed algorithm. The tag selection for retrieval is performed on the tag weights over a threshold value.

In information retrieval, genetic algorithms have been used in several ways [13–16] but in different contexts than ours. An approach for document indexing is presented by Gordon [17]. Competing document descriptions (keywords) are associated with a document and altered by using genetic operations in the approach. A keyword represents a gene and a document's list of keywords represents an individual. A collection of documents initially judged relevant by a user represents the initial population. Based on a fitness measure, the initial population evolves through generations and eventually converges into an optimal population. Yang et al. [18, 19] have developed an adaptive method based on genetic algorithms to modify user queries automatically. They report the effect of adopting genetic algorithms in large databases, the impact of genetic operators, and GA's parallel searching capability. Horng and Yeh [20] propose an approach to automatically retrieve keywords in document retrieval. they apply genetic algorithms to tune the weights of retrieved keywords.

We evaluate our method on three data sets consisting of Web pages for conference CFP (Call For Papers) and TREC (Text REtrieval Conference) documents [21]. The CFP pages are obtained from the Web to evaluate whether the proposed algorithm learns the tag importance factors properly. The effect of information retrieval using the tag weights are tested on CFP pages and TREC documents. The experimental results indicate that the structure mining approach significantly improves the retrieval performance especially for the top-ranked documents.

The remainder of the paper is organized as follows. In Section 2, we describe the document retrieval system on which our research is based. Section 3 presents the genetic algorithm for learning tag weights in Web-document retrieval. Section 4 explains the data sets used for the experimental evaluation. Section 5 presents the experimental results obtained by using CFP pages and TREC documents. Section 6 draws conclusions.

2. The Document Retrieval System

The retrieval engine used in the work is SCAIR (SCAI Information Retrieval engine) which is built to participate in TREC [22]. SCAIR is based on the vector space model in which both documents and queries are represented as vectors [23]. The components of the vector are keywords extracted from documents or queries. The retrieval system ranks the documents according to the similarities between the documents and the query vector. The higher the value of the similarity measure is, the closer to the query vector the document is. In other words, it assumes that a document, which has a high value of similarity, is relevant to a query. The retrieval system returns a list of documents ranked by similarity in descending order. As mentioned in Section 1, we give more weights on the terms which appear in important tags. How the importance of tags is learned will be described in Section 3.

The overall steps for document retrieval in SCAIR using tag weights can be summarized as follows:

1. Adapt tag weights for training queries by genetic algorithms.
2. Select only the tags which have weights greater than a threshold value Θ.
3. Retrieve N documents per topic, without using tag weights.

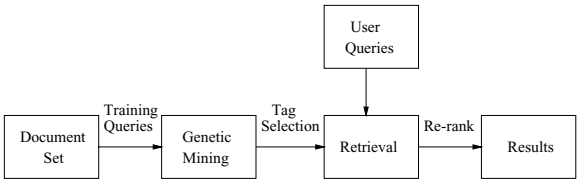

Figure 1. The document retrieval system studied in this paper.

4. Adjust the similarity of each retrieved document, using the selected tag weights.
5. Re-rank the retrieved documents in the order of revised similarities.

Figure 1 shows an abstract diagram of the retrieval system. The representation of queries and documents is based on the vector space model which is commonly used in information retrieval literature. A document or a query is regarded as a set of words called terms. A document collection is represented as a term-document matrix which is normally very sparse. For example, a query consists of terms, $Q = (q_1, q_2, \ldots, q_i, \ldots, q_T)$, where q_i denotes the ith term and T is the number of terms. A query vector is represented by the weight vector, $W = (w_1, w_2, \ldots, w_i, \ldots, w_T)$, where w_i is the weight of the ith term q_i.

2.1. Term Weighting

Term weights are real numbers indicating the significance of terms in identifying a document. If a term does not appear in a document, its corresponding weight in the document vector is zero. SCAIR supports three weighting schemes: $tf \cdot idf$ [24], Bayesian inference network model [25–27], and 2-poisson model [28, 29].

The weight of a term in a document is basically computed by the classical $tf \cdot idf$. tf (*term frequency*) is the number of times that a term t appears in a document. idf (*inverse document frequency*) is the inverse of document frequency in the collection that contain t. The $tf \cdot idf$ weighting scheme is defined as

$$w_k = tf_k \cdot \log\left(\frac{N}{df_k}\right), \tag{1}$$

where w_k is the weight of kth term in the document, tf_k is the frequency of the kth term in the document, N is the total number of documents in the collection, and df_k is the number of documents in which the kth term occurs.

Other weighting schemes, Bayesian inference network model and 2-Poisson model, are based on the probabilistic models in document retrieval. The weighting scheme of Bayesian inference network model is defined as follows:

$$w_k = \left(d_t \cdot H + (1 - d_t) \cdot \frac{\log(tf_k + 0.5)}{\log(\max tf_k + 1.0)}\right) \cdot \frac{\log(N/n)}{\log N}. \tag{2}$$

Here w_k is the weight of kth term in the document, d_t is the minimum frequency component when a term occurs in a document (set to 0.4), H is the constant depending on max tf (set to 1.0), tf_k is the frequency of the kth term in the document, max tf is the frequency of the most frequent term in the document, N is the total number of documents in the collection, and n is the number of documents in which term k occurs.

The weighting scheme of 2-poisson model is defined as follows:

$$w_k = \frac{tf_k}{K_k + tf_k} \cdot \log\frac{N - n + 0.5}{n + 0.5}, \tag{3}$$

where w_k is the weight of kth term in the document, tf_k is the frequency of the kth term in the document, $K = k_1((1 - b) + b\frac{dl}{avdl})$, k_1 is 2.0, b is 0.75, dl is the document length, $avdl$ is the average document length in the collection, N is the total number of documents in the collection, and n is the number of documents in which term k occurs.

Query terms are weighted by only tf in the inference network and 2-poisson models. But, in the case of $tf \cdot idf$ scheme, the query weighting is the same as the document weighting.

2.2. A Similarity Measure Using Tag Weights

The similarity between a document and a query text is measured by an inner product of document vector d and query vector q weighted by terms weights. Since there is a necessity of the process that applies HTML tag weights for term weighting when the documents are written in HTML, there is a potential of usefulness in using the additional weights on HTML tags. Two additional processes are included in our approach. One is using tags for each document separately, and the other is applying the weights according to the importance of the tags. The terms, which belong to the specific tags, are indicated during indexing. The classical similarity

measure used in the vector space model is modified by adding the tag constants, which is multiplied by the term weight in a document.

The new similarity function $sim(d, q)$ is defined as follows:

$$sim(d, q) = \sum_{k=1}^{T} \alpha_{dk} \cdot w_{dk} \cdot w_{qk}, \tag{4}$$

where w_{dk} is the weight of the kth term in the document d, w_{qk} is the weight of kth term in the query q, T is the number of terms, and α_{dk} is the weight of term k in the document d with respect to the tag weights.

For all the tags which are determined by term k, α_{dk} is the product of the tag weights. (When a term is not tagged in a document, α value is 1.0.) The weight of each tag is determined by the importance factor in document retrieval. Higher tag weights denote more importance in document retrieval. In this paper, the genetic mining approach is used to get the tag weights.

After determining the similarity between documents and a query, a sorted list of documents is produced.

3. Genetic Mining of HTML Structures

As described above, the characteristic of Web-documents is that they include tags for formatting and hyperlinks. In order to make use of the internal structure, we proposed in the previous section a similarity measure that is extended by tag weights. One remaining problem is to find a method which determines the significant tags and the optimal tag weights for document retrieval. We propose a genetic algorithm for this purpose.

Genetic algorithms are probabilistic search methods based on the mechanism of natural selection and genetics [30–32]. Genetic search is characterized by the fact that a number N of potential solutions of an optimization problem simultaneously sample the search space. The solutions are called *individuals* or *chromosomes* and denoted as $J_i \in \mathbf{J}$, where $\mathbf{J}$ represents the space of all possible individuals. The *population* $\vec{J} = \{J_1, J_2, \ldots, J_N\} \in \mathbf{J}^N$ is modified according to the natural evolutionary process. After the initial population is generated randomly, selection $\omega : \mathbf{J}^N \mapsto \mathbf{J}^N$ and variation $\Xi : \mathbf{J}^N \mapsto \mathbf{J}^N$ are executed in a loop until some termination criterion is reached. Each run of the loop is called a generation.

The selection operator is intended to improve the average quality of the population by giving individuals of higher quality a higher probability to be copied into the next generation. Selection thereby focuses the search on promising regions in the search space. The quality of an individual is measured by a fitness function $f : \mathbf{J} \mapsto \mathbf{R}$. The assumption is that better individuals are more likely to produce better offspring, i.e., there is a correlation between parental fitness and offspring fitness. One commonly used selection method is roulette-wheel selection [33], where each individual is represented by a space that proportionally corresponds to its fitness. By repeatedly spinning the roulette wheel, individuals are chosen by using stochastic sampling. In tournament selection [34, 35], a group of j individuals is randomly chosen from the population. This group takes part in a tournament, i.e., a winning individual is determined depending on its fitness value. For ranking selection [36–38] the individuals are sorted according to their fitness values, and rank N is assigned to the best individual and rank 1 to the worst. The selection probability is linearly assigned to the individuals according to their rank. In truncation selection [39] with threshold T only the fraction T best individuals are selected and they all have the same selection probability.

The operators for variation typically used are crossover and mutation. The idea forming the background of crossover is that useful segments of different parents should be combined in order to yield a new individual that benefits from advantageous bit combinations of both parents. Crossover is a sexual operator $r'_{\{p_c\}} : I^2 \to I$ that with probability p_c selects two parent individuals from the population, recombines them to form two new individuals, and discards one of the results at random. The one-point crossover [31] chooses one crossover position $\chi \in 1, \ldots, l-1$ within the bit-strings at random and exchanges the bits to the right of that position between both individuals. The multi-point crossover $r'_{\{p_c,z\}}$ allows $z \geq 1$ crossover positions and exchanges each second segment between subsequent crossover positions. For uniform crossover [40], the exchange segments reduce to single bits, since for each bit the decision whether to exchange it or not is done by means of independent coin tosses. The mutation operator, $m'_{\{p_m\}} : I \to I$, occasionally changes single bits of individuals with probability p_m. While randomized, genetic algorithms are no random walk. They efficiently exploit historical information for moving to new search points with expected improved performance.

We use a genetic algorithm to tune the weights of HTML tags, with the aim of producing an optimal tag weights. After that, the tags which are significant for

```
initialize chromosomes
for  g = 1 to gmax
        evaluate all chromosomes by fitness function
        for  i = 1 to M
                choose two chromosomes J1, J2
                offspring[i]  = crossover(J1,J2)
                offspring[i]  = mutation(offspring[i])
        end for
        replace  M chromosomes by offspring
end for
return optimal chromosome
```

Figure 2. General procedure for a genetic algorithm.

document retrieval are selected based on the optimal tag weights. The tag weights are encoded as a chromosome. The first step is to generate the initial population, which is made by randomly organized chromosomes and the next step is to modify the chromosomes according to the existing data set. The genetic algorithm for learning the tag weights is shown in Fig. 2.

3.1. Representation

A chromosome is defined as a list of tag weights which have real numbers. The definition of a chromosome is represented as

$$J = (j_1, j_2, \ldots, j_i, \ldots, j_L),$$

where j_i denotes the weight of the tag i and L is the number of tags to be considered. Each gene represents a tag weight. The genes of initial chromosomes are generated randomly and the range of weight values is from 0.0 to 4.0 for experiments.

3.2. The Fitness Function

The fitness function $f : \mathbf{J} \mapsto \mathbf{R}$ measures the performance of the retrieval results using tag weights. When binary scales are used for both relevance and retrieval, a table can be established showing how the document set is divided by these two measures (Table 1). Several measures such as precision and recall are used to evaluate the performance of document retrieval. Precision P is defined as the proportion of retrieved documents that are relevant, $P = \frac{w}{w+y}$. Recall is defined as the proportion of relevant documents that are retrieved, $R = \frac{w}{w+x}$.

Table 1. Evaluation contingency table. Recall of a document retrieval system is evaluated as $w/(w + x)$ and precision as $w/(w + y)$.

	Retrieved	Not retrieved
Relevant	w	x
Not relevant	y	z

The fitness function used to evaluate a chromosome is the non-interpolated average precision, which is one of the evaluation methods at TREC [41]. The non-interpolated average precision is the single-valued measure that reflects the performance over all relevant documents. The average precision for a single topic is the mean of the precision obtained after each relevant document is retrieved. If a relevant document is not retrieved at all, its precision is assumed to be 0. The non-interpolated average precision for a run consisting of multiple topics is the mean of the average precision scores of each of the individual topics in the run.

For example, consider a query that has four relevant documents which are retrieved at ranks 1, 2, 4, and 8. The actual precision obtained when each relevant document is retrieved is 1, 1, 0.75, and 0.5, respectively, the mean of which is 0.81. Thus, the non-interpolated average precision for this query is 0.81. Geometrically, non-interpolated average precision is the area underneath a non-interpolated recall-precision curve.

3.3. Genetic Operators

The genetic algorithm uses crossover and mutation operators to generate the offspring of the existing population. Before genetic operators are applied, parents should be selected for evolution to the next generation. We use truncation selection, where the parents are selected randomly from a half of the population in the decreasing order of quality. An individual is additionally selected to initialize offspring to be generated in the other half of the population, not in the part of the candidates as parents. As a result, we do not remove whole properties of an individual because it is possible that the optimal gene is hidden from a recessive chromosome. The quality of a chromosome is determined by the fitness function.

The selected parents produce offspring by crossover. The crossover used is the arithmetical crossover, which assigns the average of two parents for each location to

the corresponding location of the offspring [42]. Let J^x and J^y be two selected parent chromosomes, which are represented respectively as follows:

$$J^x = \left(j_1^x, j_2^x, j_3^x, \ldots, j_L^x\right),$$
$$J^y = \left(j_1^y, j_2^y, j_3^y, \ldots, j_L^y\right).$$

Let J^z be the offspring generated:

$$J^z = \left(j_1^z, j_2^z, j_3^z, \ldots, j_L^z\right).$$

For the initialized offspring, the weight vectors of the offspring are defined as follows:

$$j_i^z = \begin{cases} j_i^z & \text{if } \chi_i > p_c \\ \frac{(j_i^x + j_i^y)}{2.0} & \text{if } \chi_i \leq p_c, \end{cases} \tag{5}$$

where $\chi_i \in [0, 1]$ denotes a uniform random variable sampled anew for each location i and p_c is crossover probability.

Mutation is used for protect against possible loss of diversity. It changes the value of randomly selected position in a random chromosome with probability p_m:

$$j_i^{z'} = \begin{cases} j_i^z & \text{if } \chi_i > p_m \\ \text{random}(\alpha, \beta) & \text{if } \chi_i \leq p_m, \end{cases} \tag{6}$$

where $\chi_i \in [0, 1]$ denotes a uniform random variable sampled anew for each location and random(α, β) is a function to generate randomly real value with the range between α to β. For experiments, the mutation rate p_m is set 0.5 to 0.7 because the arithmetic crossover tends to converge quickly. After genetic operators are applied, a half of the population other than the selected parents are substituted by the produced offspring. Figures 3 and 4 describe crossover and mutation.

4. Data Sets

We now describe the data sets used for the experiments. We use three data sets: CFP A, CFP B, and TREC documents. CFP data sets are used to evaluate the capability of the genetic algorithm in finding the importance of tag weights. CFP B and TREC data sets are used to evaluate the effect of retrieval performance when the HTML internal structure is reflected.

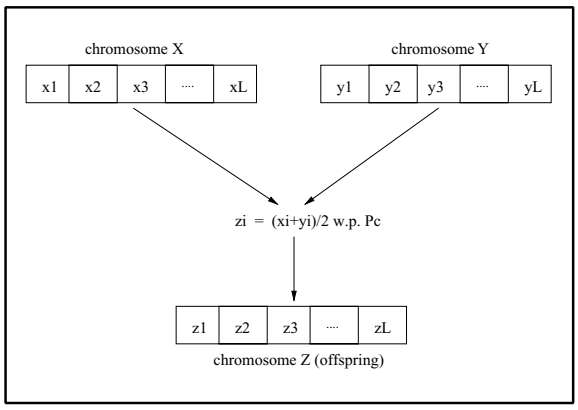

Figure 3. Crossover of two weight vectors (chromosomes).

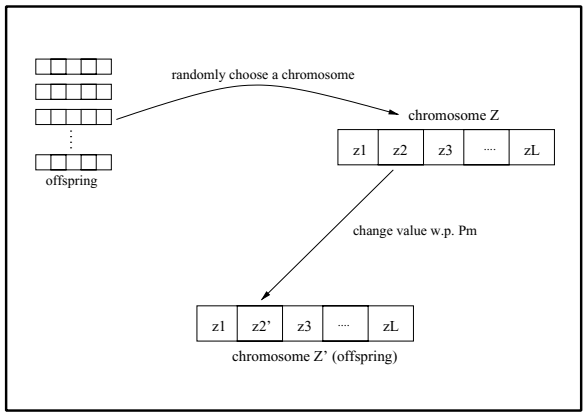

Figure 4. Mutation of a weight vector (chromosome).

4.1. CFP Data Sets

CFP data sets contain call-for-papers pages collected from the Web. They include the conference pages on SIGIR, IJCAI, EuroGP, and so on. Two data sets, CFP A and CFP B, are constructed for the experiments. Each set includes 2161 documents of same contents. The difference between CFP A and CFP B is the internal structure. Original documents obtained from the Web are included in CFP A. CFP B includes modified documents from CFP A. The keywords of each document in CFP B are emphasized by header tag (<Hx>).

Four sentences were generated for user queries. For example, one query consists of six words, 'conference', 'information', 'retrieval', 'section', 'genetic', and 'algorithms'. It assumes that a user wants to find the conferences related to information retrieval that include a session on genetic algorithms. Relevant documents of the queries were manually determined. While about 15

documents are judged as relevant for each query, others are irrelevant.

Intuitively, the header tag in CFP B should get higher weight than other tags after trained by the proposed algorithm. Also, CFP B should get more tag weight of the header than those of CFP A.

4.2. TREC Document Sets

The documents for TREC data is WT2g, which is used for Web Track of TREC sponsored by NIST (National Institute of Standards and Technology) [21]. It was collected by Internet Archive and includes all WWW pages [43]. There are 247,491 distinct pages of 2 Gigabytes.

A query is called a topic in TREC. A topic consists of title, description, and narrative. A sample topic is shown in Fig. 5. The title has been specially designed to allow experiments with very short queries. The title consists of up to three words that best describe the topic. The description field is one sentence description of the topic area. The description field contains all of the words in the title field. The narrative gives a concise description of what makes a document relevant.

The title and description fields of a topic are considered as queries in the experiments. 10 queries (Topic No. 401 to 410) are used for learning and another 10 queries (Topic No. 411 to 420) are used for retrieval.

We use the results of relevant documents for queries, which are published by NIST. Relevant documents are judged by using the pooling method [41, 44]. In this method, a pool of possible relevant documents is created by taking a sample of documents selected by the various participating systems of TREC. This pool is then shown to the human assessors. The sampling method used in TREC is to take the top 100 documents retrieved per a judged run for a given topic and merge them into the pool for assessment. This is a valid sampling technique since all the systems use the ranked retrieval methods, with those documents most likely to be relevant returned first. Each pool is sorted by document ID, so that assessors cannot tell if a document was highly ranked by some system or how many systems retrieved the document.

<title> foreign minorities, Germany
<desc> Description:
What language and cultural differences impede the integration of foreign minorities in Germany?
<narr> Narrative:
A relevant document will focus on the causes of the lack of integration in a significant way; that is, the mere mention of immigration difficulties is not relevant. Documents that discuss immigration problems unrelated to Germany are also not relevant.

Figure 5. A sample TREC topic.

5. Experimental Results

In this section, we present our experiments on learning tag weights and retrieval using the selected tags. The CFP A and CFP B data sets are used to judge whether the genetic learning determines the tag weights correctly, and the CFP B and TREC data sets are used to show the change when the tags learned by the proposed algorithm are applied for document retrieval.

Five HTML tags (<TITLE>, <H>, <B>, <I> and <A>) are used for the experiments. They denote Title, Header, Bold, Italic and Anchor, respectively. The title and header tags use only the words in a Web page marked as a part of the title or headings to classify that page. Thus, words marked with the tags are considered as representatives of the page. The bold and italic tags are taken because they are used to emphasize words. We assume that the hyperlinks of a document generally are connected to the related documents. Thus, the anchor tag is also used.

All weighting schemes supported by SCAIR are evaluated in our experiments. 'TFIDF', 'Bayesian', and '2-Poisson' respectively mean $tf \cdot idf$, Bayesian inference network, and 2-Poisson weighting schemes. For each experiment, the learning phase is repeated 20 times until the 30 generations with between 100 and 150 of population size.

5.1. Experiment 1: HTML Tag Learning

The method developed in Section 3 is tested on CFP A and CFP B data. The retrieval system returns one hundred documents for a topic ordered by the similarities between documents and a query. Two queries are used for learning HTML tag weights.

Experimental results in Fig. 6 show the average fitness for generations on Set A (CFP A) and Set B (CFP B). Here the weighting scheme used was the standard TFIDF. Set B is better structured than Set A by tagging the keywords in documents using header. It can be observed that Set B shows higher precision than Set A, and the difference between Set A and Set B become greater

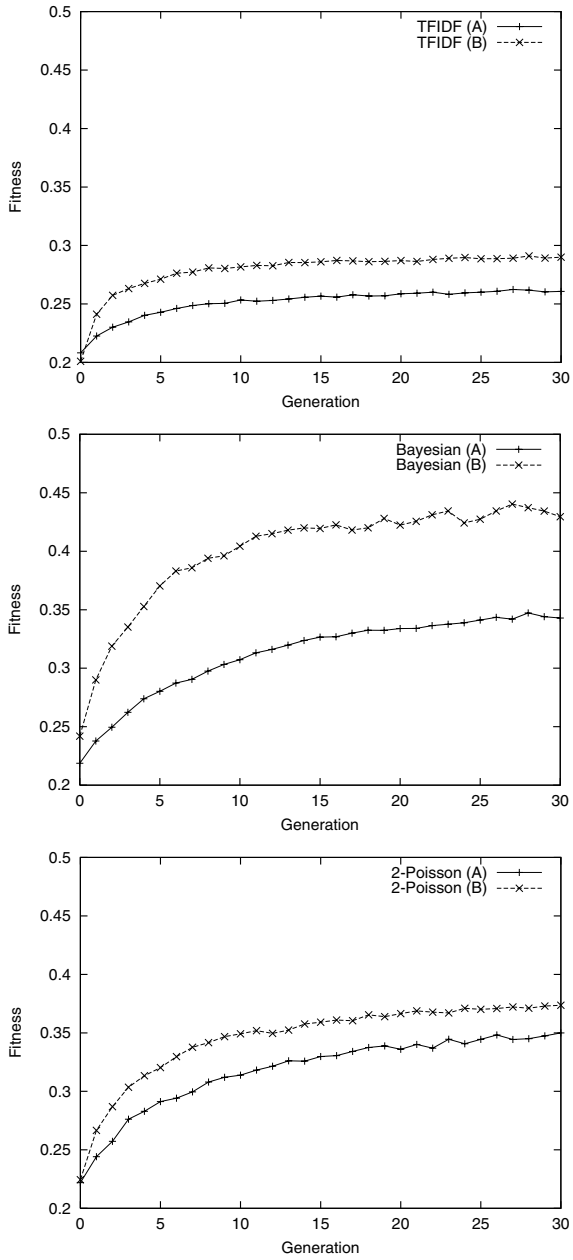

Figure 6. Average fitness for generations on CFP data. This figure shows that the difference of average fitness as generation continues.

as the generation continues. Similar experiments have been repeated using two different weighting schemes: Bayesian and 2-Poisson.

As the generation continues, further improvement is found in average population fitness as demonstrated in Fig. 6. The fitness rapidly increases until about the 8th generation. Then, the fitness increases rather slowly. It seems to be caused by the arithmetic crossover. The offspring are generated by the average of the parents which have higher fitness. In addition to it, the substituted chromosomes are the half of the population in one generation. Therefore, the early generations are converged to the chromosomes of the population which have high fitness even after one generation. As the generation progresses further, the increment of fitness falls down a little because most of the chromosomes are already converged to the high fitness.

The chromosome that has the highest fitness over generations is regarded as the optimal tag weights. The tag weights learned by the genetic algorithm are shown in Figs. 7–9 and Table 2. The tag weights except header, which has higher value in Set B than Set A, are similar in both Set A and Set B. As a result, it can be concluded that the proposed learning algorithm works efficiently as mentioned in Section 4, i.e. learns the weights according to the importance of the tags in document retrieval.

5.2. Experiment 2: Document Retrieval I

For the experiments on Web-document retrieval, we use CFP B in which the keywords of documents are emphasized by header tags. HTML tags in CFP B are already evaluated in Section 5.1. The HTML tag selection for retrieval is based on the weights learned by the genetic algorithm. First, we select the individual that has the best fitness value on CFP B over experiments, and then the HTML tags that have weights over 1.0 are selected for Web-document retrieval. The header tag is selected for all weighting schemes. The bold tag is selected for TFIDF and 2-Poisson models. The italic tag is selected for the 2-Poisson model. Table 3 is the weights of HTML tags learned by the proposed algorithm.

Other two queries not used for learning are selected for retrieval using HTML tags. The results returned by the system are limited to one hundred documents for a topic because users usually want to receive a small number of documents for a query, i.e. they want high precision and low recall.

Tables 4 and 5 show the retrieval results using HTML tags. They indicate that the retrieval using tag weights gives better performance than those not using tag weights. The precision-recall curves for using and not using the HTML tags are shown in Fig. 10. In case of TFIDF and Bayesian inference network, the retrieval accuracy using HTML tags is higher than that of normal retrieval at every recall point.

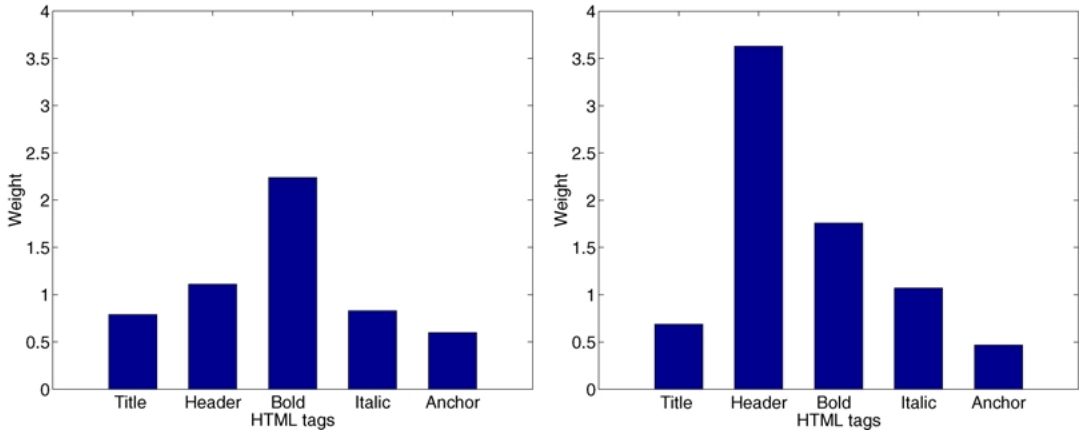

Figure 7. HTML tag weight averages of Set A (left) and Set B (right) for the TFIDF weighting scheme.

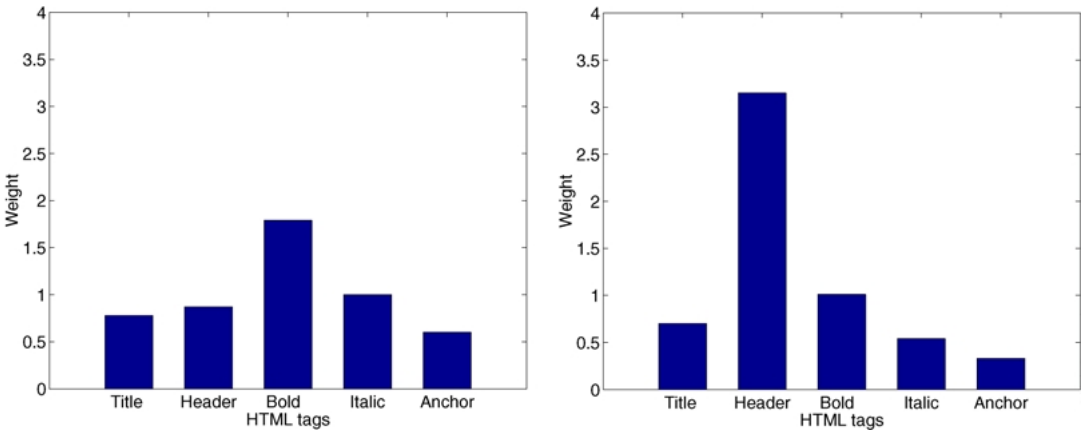

Figure 8. HTML tag weight averages of Set A (left) and Set B (right) for the Bayesian weighting scheme.

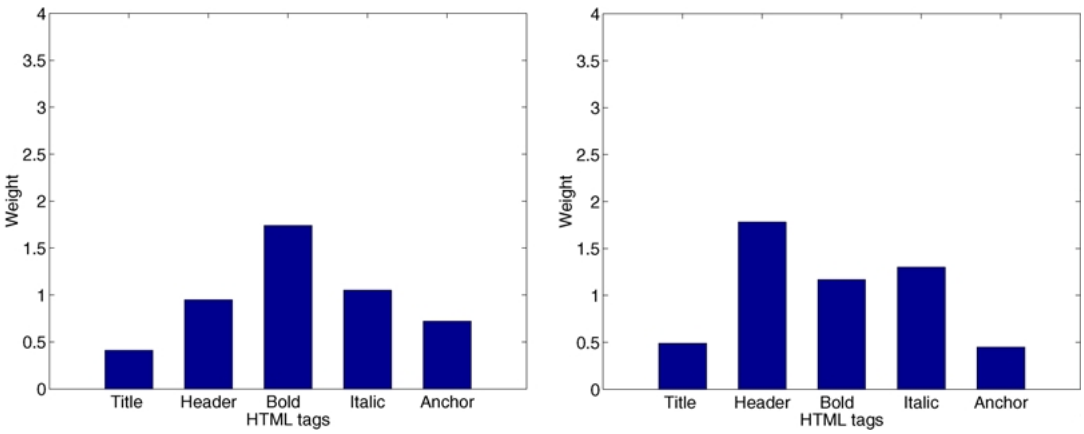

Figure 9. HTML tag weight averages of Set A (left) and Set B (right) for the 2-Poisson weighting scheme.

Table 2. HTML tag weights learned on CFP data. The numbers are average values over 20 runs.

	Set A			Set B		
HTML tags	TFIDF	Bayesian	2-Poisson	TFIDF	Bayesian	2-Poisson
Title	0.79 ± 0.56	0.78 ± 0.27	0.41 ± 0.25	0.69 ± 0.19	0.70 ± 0.18	0.49 ± 0.20
Header	1.11 ± 0.27	0.87 ± 0.14	0.95 ± 0.07	3.63 ± 0.35	3.15 ± 0.72	1.78 ± 0.13
Bold	2.24 ± 0.28	1.79 ± 0.07	1.74 ± 0.09	1.76 ± 0.52	1.01 ± 0.12	1.17 ± 0.12
Italic	8.33 ± 0.22	1.00 ± 0.11	1.05 ± 0.16	1.07 ± 0.57	0.54 ± 0.57	1.30 ± 0.09
Anchor	0.60 ± 0.35	0.60 ± 0.03	0.72 ± 0.06	0.47 ± 0.11	0.33 ± 0.07	0.45 ± 0.14

Table 3. HTML tag weights learned on CFP B.

HTML tag	TFIDF	Bayesian	2-Poisson
Title	0.7	0.6	0.4
Header	3.7	3.2	1.9
Bold	2.2	1.0	1.1
Italic	1.0	0.4	1.3
Anchor	0.5	0.2	0.5

Table 4. Comparison of non-interpolated average precision for CFP B.

	TFIDF	Bayesian	2-Poisson
Without HTML tags	0.1593	0.1502	0.2406
With HTML tags	0.2324	0.2956	0.2607

5.3. Experiment 3: Document Retrieval II

This experiment compares our genetic approach with normal retrieval not using HTML tags on TREC data set. TREC has a large document set (247,491 pages), contrary to the CFP data sets. It is used to examine the variance of results using HTML structure for a large document set. Ten queries of TREC-8 are selected for training and other ten queries for retrieval. The retrieval results returned by our system are limited

Table 6. HTML tag weights on TREC data.

HTML tag	TFIDF	Bayesian	2-Poisson
Title	0.5	1.1	0.7
Header	1.6	1.1	1.2
Bold	0.7	0.9	0.9
Italic	0.6	0.8	1.0
Anchor	1.6	1.3	1.2

Table 7. Comparison of non-interpolated average precision on the TREC data.

	TFIDF	Bayesian	2-Poisson
Without HTML tags	0.2383	0.2870	0.3113
With HTML tags	0.2503	0.3015	0.3522

Table 8. Comparison of precision at N retrieved documents for the TREC data.

	TFIDF		Bayesian		2-Poisson	
	P@10	P@20	P@10	P@20	P@10	P@20
Without HTML tags	0.3100	0.3750	0.4100	0.3700	0.5400	0.4200
With HTML tags	0.3400	0.4050	0.4500	0.3800	0.6300	0.4600

Table 5. Comparison of precision at N retrieved documents on CFP B. The symbol P@10 means precision at 10 retrieved documents and P@20 means precision at 20 retrieved documents.

	TFIDF		Bayesian		2-Poisson	
	P@10	P@20	P@10	P@20	P@10	P@20
Without HTML tags	0.2000	0.2000	0.2000	0.1500	0.3500	0.2750
With HTML tags	0.3500	0.2750	0.4500	0.3250	0.4000	0.3000

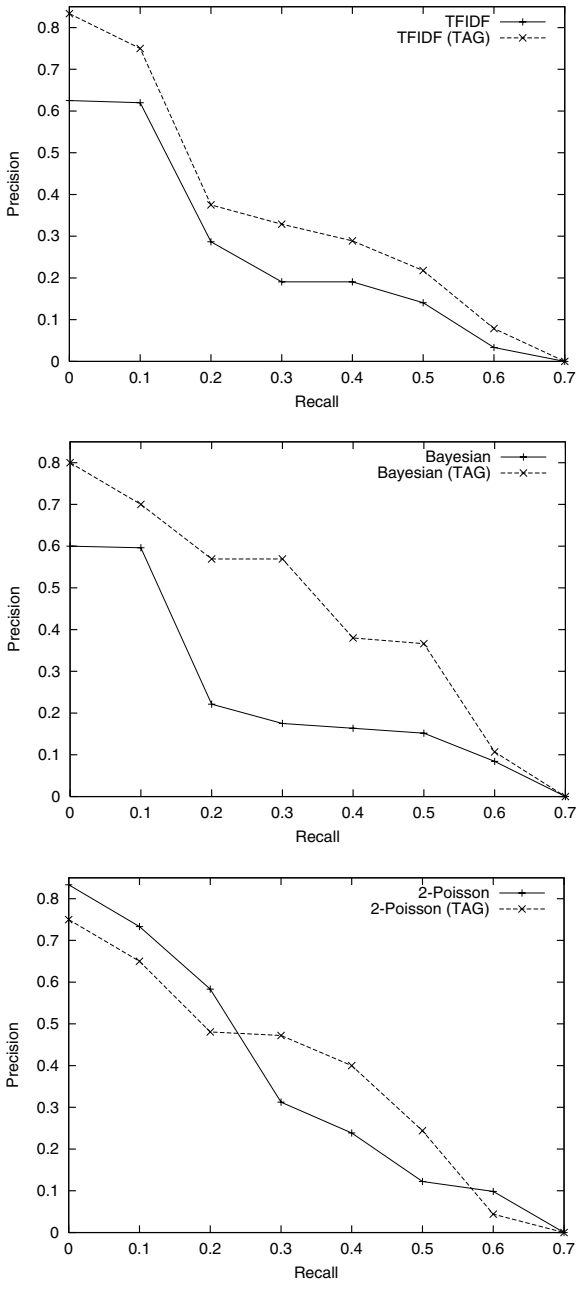

Figure 10. Interpolated recall-precision averages for CFP B.

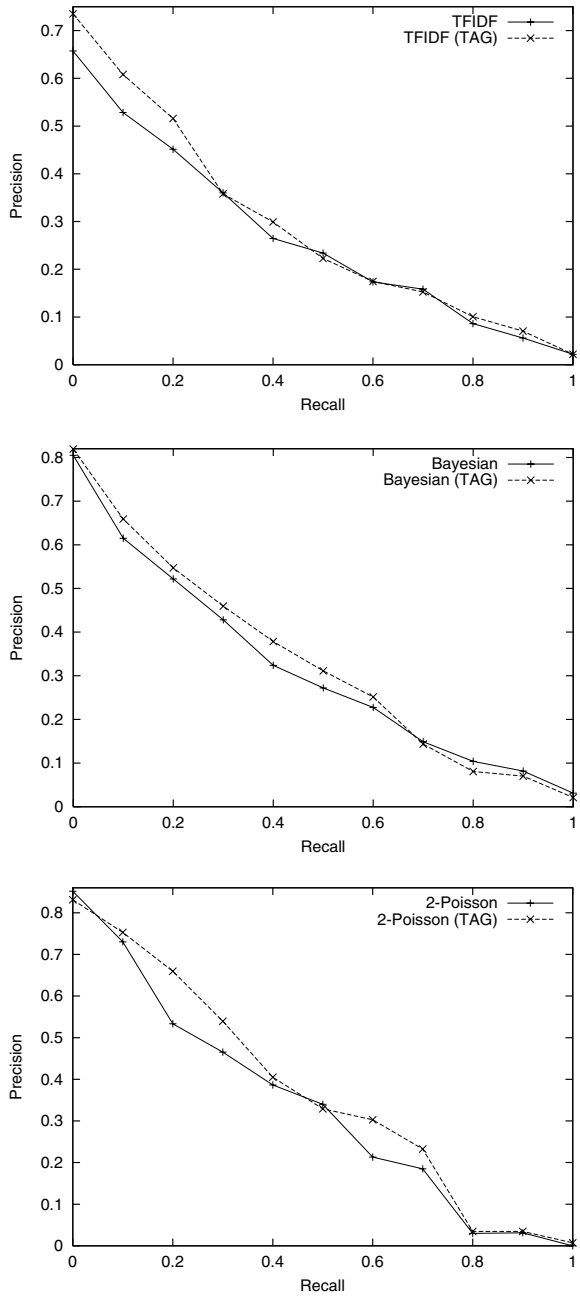

Figure 11. Interpolated recall-precision averages for the TREC data.

to two hundred documents for a topic. The weights of HTML tags are presented in Table 6 for each weighting schemes. The weights of the title tags turn out to be relatively small as shown in Tables 3 and 6. This seems due to the fact that the title of a document is too short to correctly describe the text. The weights of some tags, such as anchors, seem to vary over experiments since the genetic approach learns user's personal retrieval patterns and the characteristics of given document sets.

For retrieval using HTML tags, the header and the anchor tags are selected for all weighting schemes. The title tag is selected for only the Bayesian network model.

The comparison of our method with the method not using tag weights is shown in Tables 7 and 8. Even though the variance gets relatively smaller than the experiment on CFP data, the experimental results in the tables indicate that the retrieval using HTML tags have better performance results than normal retrieval in precision average. Figure 11 shows the precision-recall curves for using and not using the tag weights. Our approach has better performance in precision accuracy at any weighting schemes. High precision at low level of recall means that there are more relevant documents on top-ranked documents.

6. Conclusions

This paper proposes a text mining approach to Web-document retrieval that uses the tag information of HTML documents to improve retrieval performance. Genetic algorithms, which are generally quite effective for rapid global search in large search spaces, are applied to find the significant HTML tags and get the optimal weights. The approach is used to retrieve CFP and TREC documents according to the tag weights learned.

Our experimental results show that significant HTML tags of Web-documents can be properly selected and the document retrieval systems can take advantage of this information to outperform the traditional information retrieval approaches which are based on plain texts. We also find that the degree of performance improvement depends on how well the HTML documents are structured.

The genetic algorithm learns the retrieval pattern through a series of query and relevant documents. Thus, it can be used for retrieval in a specialized document set or for retrieval restricted to a user, i.e. a retrieval agent. In this study, we also find that improvement is limited to some extent because HTML documents are semi-structured as opposed to fully structured. Using techniques that make HTML documents more well-structured or that automatically convert Web-documents into well-structured documents would further improve the performance of HTML document retrieval. The presented structure-mining method can also be effectively used for XML-document retrieval.

Acknowledgments

This research was supported in part by AITrc, the Korea Ministry of Information and Telecommunications under Grant 00-034 through IITA, The Korean Ministry of Science and Technology under Brain Tech Program and the Ministry of Education under the BK21-IT Program.

References

1. M. Gordon and P. Pathak, "Finding information on the world wide web: The retrieval effectiveness of search engines," *Information Processing and Management*, vol. 35, no. 2, pp. 141–180, 1999.
2. N.J. Belkin and W.B. Croft, "Retrieval techniques," *Annual Review of Information Science and Technology*, vol. 22, pp. 109–145, 1987.
3. J. Boyan, D. Freitag, and T. Joachims, "A machine learning architecture for optimizing web search engines," in *Proceedings of the AAAI Workshop on Internet-Based Information Systems*, pp. 1–8, 1996.
4. K. Bharat and M.R. Henzinger, "Improved algorithms for topic distillation in a hyperlinked environment," in *Proceedings of the ACM SIGIR'98 Conference*, pp. 104–111, 1998.
5. S. Chakrabarti, "Data mining for hypertext: A tutorial survey," *ACM SIGKDD Explorations*, vol. 1, no. 2, pp. 1–11, 2000.
6. J. Kleinberg, "Authoriatative sources in a hyperlinked environment," in *Proceedings of the Ninth ACM-SIAM Symposium on Discrete Algorithms*, pp. 668–677, 1998.
7. J. Picard, "Modeling and combining evidence provided by document relationships using probabilistic argumentation systems," in *Proceedings of the ACM SIGIR'98 Conference*, pp. 182–189, 1998.
8. E. Spertus, "ParaSite: Mining structural information on the web," in *Proceedings of the Sixth International World Wide Web Conference (WWW6)*, pp. 1205–1215, 1997.
9. S. Chakrabarti et al., "Experiments in topic distillation," *ACM-SIGIR '98 Post-Conference Workshop on Hypertext Information Retrieval for the Web*, 1998.
10. S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," in *Proceedings of the Seventh International World Wide Web Conference (WWW7)*, pp. 107–117, 1998.
11. M.L. Mauldin, "Lycos: Design choices in an internet search service," *IEEE Expert*, vol. 12, no. 1, pp. 8–11, 1997.
12. S. Kim and B.-T. Zhang, "Web-Document retrieval by genetic learning of importance factors for HTML tags," in *Proceedings of the Sixth Pacific Rim International Conference on AI Workshop on Text and Web Mining*, pp. 13–23, 2000.
13. M. Cutler, H. Deng, S. Maniccam, and W. Meng, "A new study on using HTML structures to improve retrieval," in *Proceedings of the Eleventh IEEE Conference on Tools with AI*, pp. 406–409, 1999.
14. O. Frieder and H.T. Siegelmann, "On the allocation of documents in multiprocessor information retrieval systems," in *Proceedings of the ACM SIGIR'91 Conference*, pp. 230–239, 1991.
15. M.D. Gordon, "User-based document clustering by redescribing subject descriptions with a genetic algorithm," *Journal of the American Society for Information Science*, vol. 42, no. 5, pp. 311–322, 1991.

16. F. Petry, B. Buckles, D. Prabhu, and D. Kraft, "Fuzzy information retrieval using genetic algorithms and relevance feedback," in *Proceedings of the ASIS Annual Meeting*, pp. 122–125, 1993.
17. M. Gordon, "Probabilistic and genetic algorithms for document retrieval," *Communications of the ACM*, vol. 31, pp. 1208–1218, 1988.
18. J. Yang and R.R. Korfhage, "Effects of query term weights modification in document retrieval: A study base on a genetic algorithm," in *Proceedings of the Second Annual Symposium on Document Analysis and Information Retrieval*, pp. 271–185, 1993.
19. J. Yang, R.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 First Text Retrieval Conference (TREC-1)*, pp. 31–58, 1993.
20. J.T. Horng and C.C. Yeh, "Applying genetic algorithms to query optimization in document retrieval," *Information Processing and Management*, vol. 36, pp. 737–759, 2000.
21. NIST, *Text REtrieval Conference homepage*, http://trec.nist.gov.
22. D.-H. Shin and B.-T. Zhang, "A two-stage retrieval model for the TREC-7 ad hoc task," in *Proceedings of the Seventh Text Retrieval Conference (TREC-7)*, pp. 501–507, 1998.
23. G. Salton, A. Wong, and C.S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, pp. 613–620, 1975.
24. G. Salton, *Automatic Text Processing*, Addison-Wesley, pp. 279–281, 1989.
25. J. Broglio, J.P. Callan, W.B. Croft, and D.W. Nachbar, "Document retrieval and routing using the INQUERY system," in *Proceedings of the Third Text REtrieval Conference (TREC-3)*, pp. 29–38, 1995.
26. J.P. Callan, W.B. Croft, and S.M. Harding, "The INQUERY retrieval system," in *Proceedings of the Third International Conference on Database and Expert Systems Applications*, pp. 78–83, 1992.
27. H. Turtle and W.B. Croft, "Inference networks for document retrieval," in *Proceedings of the Thirteenth International Conference on Research and Development in Information Retrieval*, pp. 1–24, 1990.
28. S.E. Robertson and S. Walker, "Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval," in *Proceedings of the ACM SIGIR'94 Conference*, pp. 232–241, 1994.
29. S.E. Robertson et al., "Okapi at TREC-3," in *Proceedings of the Third Text Retrieval Conference (TREC-3)*, pp. 109–126, 1995.
30. T. Bäck, *Evolutionary Algorithms in Theory and Practice*, Oxford University Press, 1996.
31. J. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.
32. J.R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, 1992.
33. D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.
34. T. Blickle and L. Thiele, "A mathematical analysis of tournament selection," in *Proceedings of the Sixth International Conference on Genetic Algorithms*, pp. 9–16, 1995.
35. D.E. Goldberg and K. Deb, "A comparative analysis of selection schemes used in genetic algorithms," *Foundations of Genetic Algorithms*, pp. 69–93, Morgan Kaufmann, 1991.
36. J.E. Baker, "Adaptive selection methods for genetic algorithms," in *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*, pp. 101–111, 1985.
37. J.J. Grefenstette and J.E. Baker, "How genetic algorithms work: A critical look at implicit parallelism," in *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 20–27, 1989.
38. D. Whitley, "The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best," in *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 116–121, 1989.
39. H. Mühlenbein and D. Schlierkamp-Voosen, "Predictive models for the breeder genetic algorithm," *Evolutionary Computation*, vol. 1, no. 1, pp. 25–49, 1993.
40. G. Syswerda, "Uniform crossover in genetic algorithms," in *Proceedings of the Third International Conference on Genetic Algorithms and Their Applications*, pp. 2–9, 1989.
41. E.M. Voorhees and D. Harman, "Overview of the eighth text Retrieval conference," in *Proceedings of the Eighth Text Retrieval Conference (TREC-8)*, pp. 1–27, 1999.
42. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolutionary Programs*, Springer, pp. 104–105, 1992.
43. Internet Archive, *Building an Internet Library*, http://www.archive.org.
44. J. Zobel, "How Reliable are the Results of Large-Scale Information Retrieval Experiments?," in *Proceedings of the ACM SIGIR'98 Conference*, pp. 307–314, 1998.

Sun Kim is a Ph.D. student in the School of Computer Science and Engineering, Seoul National University. He received a BS degree in Computer Science from Soongsil University in 1999 and an MS degree in the School of Computer Science and Engineering from Seoul National University in 2001. His research interests include information retrieval, evolutionary computation, and bioinformatics.

Byoung-Tak Zhang is an Associate Professor of Computer Science and Engineering at Seoul National University (SNU). He

received his BS and MS degrees in computer engineering from SNU in 1986 and 1988, respectively, and a Ph.D. in Computer Science from University of Bonn, Germany in 1992. Prior to joining SNU, Dr. Zhang has been a research associate at German National Research Center for Information Technology (GMD) from 1992 to 1995. He serves as an Associate Editor of IEEE Transactions on Evolutionary Computation. His research interests are in learning and adaptive systems, evolutionary computation, probabilistic neural networks, and their application to real- world AI problems.