Artificial Intelligence Review (2005) 24:277–299
DOI 10.1007/s10462-005-9001-y

Evolving General Term-Weighting Schemes for Information Retrieval: Tests on Larger Collections

RONAN CUMMINS* & COLM O'RIORDAN
Department of Information Technology, National University of Ireland, Galway, Ireland
*(*author for correspondence, e-mail: ronan.cummins@nuigalway.ie)*

Abstract. Term-weighting schemes are vital to the performance of Information Retrieval models that use term frequency characteristics to determine the relevance of a document. The vector space model is one such model in which the weights assigned to the document terms are of crucial importance to the accuracy of the retrieval system. This paper describes a genetic programming framework used to automatically determine term-weighting schemes that achieve a high average precision. These schemes are tested on standard test collections and are shown to perform as well as, and often better than, the modern BM25 weighting scheme. We present an analysis of the schemes evolved to explain the increase in performance. Furthermore, we show that the global (collection wide) part of the evolved weighting schemes also increases average precision over *idf* on larger TREC data. These global weighting schemes are shown to adhere to Luhn's resolving power as middle frequency terms are assigned the highest weight. However, the complete weighting schemes evolved on small collections do not perform as well on large collections. We conclude that in order to evolve improved local (within-document) weighting schemes it is necessary to evolve these on large collections.

Keywords: genetic programming, information retrieval, term-weighting schemes

1. Introduction

Information Retrieval (IR) deals with the retrieval of information according to specification by subject. IR stems from traditional data retrieval and deals with retrieving information from semi-structured or unstructured information sets using natural language queries. With the advent of the World Wide Web and the vast increase in the quantity of information available, the need to retrieve information based on a user's query has become increasingly important. IR systems attempt to return only documents that are relevant to a given information need. However, the use of natural language in both queries and document sets leads to difficulties in retrieving accurate information for the user. An IR system must make an effort to interpret the semantic content of a document and rank the documents in relation to the user's query.

The vector space model (Salton et al., 1975) is one of the most widely known and studied IR models. This is mainly due to its simplicity, its efficiency over large document collections and the fact that it is intuitively appealing. The effectiveness of vector based models depends crucially on the term-weighting applied to the terms of the document vectors (Salton and Buckley, 1988). These term-weights are typically calculated using measures of the terms in the documents and across the collection.

Founded in the early 1990s, the Genetic Programming (GP) area (Koza, 1992) has grown quickly and helped solve problems in a wide variety of areas including robotic control, pattern recognition, music and synthesis of artificial neural networks. GP is inspired by Darwinian theory of natural selection (1859), where individuals that have a higher fitness will survive and thus produce offspring. These offspring will inherit characteristics similar to those of their parents and, through successive generations, beneficial characteristics will survive. GP can be viewed as an artificial way of selective breeding.

This paper describes a Genetic Programming framework that artificially breeds term weighting schemes for the vector space model. This paper extends previous work (Cummins and O'Riordan, 2004a) where similar evolved schemes are compared against the traditional *tf-idf* scheme (Salton and Buckley, 1988). We compare evolved schemes against more modern weighting schemes and tests are conducted on larger TREC data. Section 2 introduces some background material in Information Retrieval. A background to Genetic Programming and some past approaches of evolutionary computation techniques applied to IR are reviewed in Section 3. Section 4 describes the system and experimental design. Results and analysis are discussed in detail in Section 5. Finally, our conclusions are discussed in Section 6.

2. Information Retrieval

2.1. *Early advances in IR*

Zipf (1949) showed that the frequency of terms in a collection when placed in rank order approximately follows a log curve. Luhn proposed that the resolving power of significant words, by which he meant the ability of words to discriminate content, reached a peak at a rank order position half way between the two cut-offs, and from the peak fell off in either direction reducing to almost zero at the cut-off points (Van Rijsbergen, 1979). In Figure 1, the bell-shaped curve of the graph relates the frequency of terms to their distinguishing (resolving) power (Luhn, 1958).

Salton et al. (1975) validate much of Luhn's work with empirical analysis by what they call the *discrimination value* of a term. They devise a scheme which weights terms on their ability to render the document space as dissimilar as possible. Thus, terms which decrease the similarity among documents in a collection (i.e. terms which push documents further apart in the vector space) receive a high weight. They come to similar conclusions to that of Luhn; that middle frequency terms are the most useful in terms of retrieval. Low frequency terms have, on average, a poor discrimination value while high frequency terms are the least useful (Salton and Yang, 1973; Salton et al., 1975).

2.2. *Vector space model*

The classic vector space model represents each document in the collection as a vector of terms with weights associated to each term. The weight of each term in a document vector is based on the frequency of the term within that document and across the document collection. The query is also modeled as a vector and a matching function is used to compare each document vector to the query vector. Once the documents are compared, they are sorted into a ranked list and returned to the user. One of the most common matching functions of

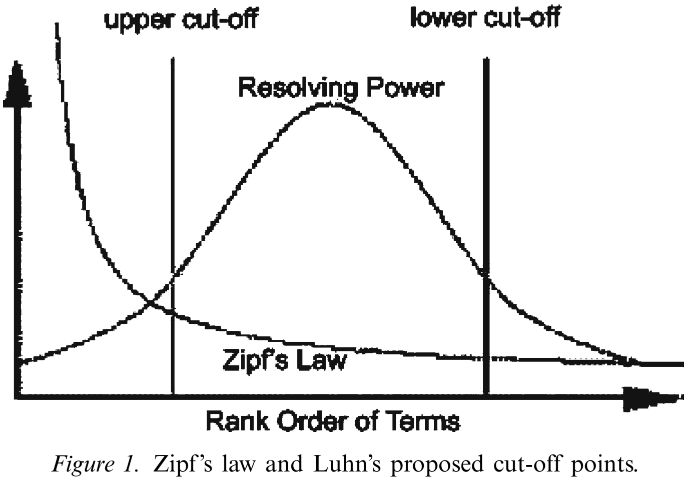

Figure 1. Zipf's law and Luhn's proposed cut-off points.

the vector space model is the inner-product measure which is calculated as follows:

$$\mathrm{sim}(d_i, q) = \sum_{k=1}^{t} (w_{ik} \times q_k) \tag{1}$$

where q is the query, d_i is the ith document in the document set, t is the number of terms in the document, w_{ik} is the weight of term k in the ith document and q_k is the weight of term k in the query.

The *tf-idf* family of weighting schemes is the most popular form of weighting scheme used in modern retrieval systems. The *tf* (term-frequency) part of the weighting scheme is a normalised local (within-document) measure that assigns a higher weight to terms that occur more often within a document. The frequency is normalised as longer documents tend to have more terms and higher term frequencies. Due to the nature of IR algorithms, this would result in long documents being ranked as more relevant than short documents, when in fact this may not be true. The *idf* (inverse document frequency) part of the weighting scheme is a global (collection wide) measure that assigns a higher weight to terms which are rarer across the collection. The main heuristic behind the *idf* factor, is that a term that occurs infrequently is good at discriminating documents. However, it can be seen that *idf* is inconsistent with Luhn's theory of resolving power (1958) and Salton's discrimination value model (1975) at low frequency levels.

2.3. *Probabilistic models*

Probabilistic models weight the relevance of a term in a document on the probability that a term appears in a relevant document and the probability that it appears in a non-relevant document. The Binary Independence Model, developed by Robertson and Sparck Jones (1976), calculates the optimal weights for terms based on a set of relevant documents. There are two main assumptions behind this model. The term occurrences are assumed independent of one another and a binary weighting is assigned to the term frequency. However, at an initial retrieval stage there is little or no relevance information available. Thus, the effectiveness of this model depends on the availability of existing relevance information. This is usually supplied by relevance feedback techniques. The BM25 weighting scheme, developed by Robertson et al. (1995), is a weighting scheme based on the probabilistic model. The local (within-document) part of this scheme is called *Okapi-tf* and is calculated as follows:

$$Okapi\text{-}tf = \frac{rtf}{rtf + k_1\left((1-b) + b\frac{dl}{dl_{\mathrm{avg}}}\right)} \tag{2}$$

where rtf is the raw term frequency and dl and dl_{avg} are the length and average length of the documents, respectively. k_1 and b are tuning parameters which influence the term-frequency and document normalisation, respectively. The idf of a term as determined in the BM25 formula is as follows:

$$idf_t = \log\left(\frac{N - df_t + 0.5}{df_t + 0.5}\right) \tag{3}$$

where df_t is the document frequency of term t and N is the number of documents in the collection. The weight assigned to a term in the BM25 scheme is a product of *Okapi-tf* and *idf*. The ranking score given to a document can then be calculated as follows:

$$\sum_{t \in q \cap d} (Okapi\text{-}tf \times idf_t \times qrtf) \tag{4}$$

where $qrtf$ is the raw term-frequency of the term in the query. Although developed for the probabilistic model, the BM25 weights are often used in vector based IR systems. The BM25 scheme is currently one of the best performing weighting schemes in IR. However, it has been indicated that the probability of a term occurring in a relevant document tends to zero as the frequency of the term occurring in the collection tends to zero (Robertson and Walker, 1997). This would indicate probabilistically that terms with a low document frequency should initially be assigned a lower weight than that assigned by the *idf* measure. Greiff (1998) has also predicted that a flattening of the *idf* measure at low frequencies would lead to an increase in performance.

3. Genetic Programming

Inspired by the successes in traditional Genetic Algorithms, John Koza developed Genetic Programming (GP) in the early 1990s (1992). The GP approach has helped solve problems in a wide variety of areas. GP is inspired by Darwinian theory of natural selection (1859), where individuals that have a higher fitness will survive longer and thus produce more offspring. These offspring will inherit characteristics similar to those of their parents and through successive generations, the useful characteristics will survive. GP can be viewed as an artificial way of selective breeding. In GP, solutions are encoded as trees with operators (functions) on internal nodes and operands (terminals) on the leaf nodes. These nodes are often referred to as genes and their values as alleles. The coded version of a solution is called its genotype, as it can be thought of as the genome of the individual, while the solution in its environment is called its phenotype. The fitness is evaluated on the phenotype of a candidate solution while reproduction and crossover are performed on the genotype.

The basic flow of a GP is shown in Figure 2. Initially, a random population of solutions is created. These solutions are encoded as trees. Each solution is rated based on how it performs in its environment. This is achieved using a fitness function. Having assigned the fitness values, selection can occur. Goldberg (1989) uses the roulette wheel example where each solution is represented by a segment on a roulette wheel proportionately equal to the fitness of the solution to explain how selection occurs. Thus, solutions with a higher fitness will produce more offspring. Tournament selection is one of the most common selection method used. In tournament selection, a number of solutions are chosen at random and these solutions compete with each other. The fittest solution is then chosen as a parent. The number of solutions chosen to compete in the tournament is the tournament size and this can be increased or decreased to increase or decrease the speed of convergence.

Once selection has occurred, reproduction can start. Reproduction (recombination) can occur in a variety of ways. The most common form is sexual reproduction, where two different individuals (parents) are selected and two separate children are created by combining the genotypes of both parents. An example of crossover can be seen in Figure 3. Mutation (asexual reproduction) is the random change of the allele of a gene to create a new individual. Selection and recombination occurs until the population is replaced by newly created individuals. Usually the number of solutions from generation to generation remains constant. Once the recombination process is complete, each individual's fitness in the new generation is evaluated and the selection process starts again. The process usually ends after a predefined number of generations, or when convergence of the population is achieved or after an individual is found with an acceptable fitness.

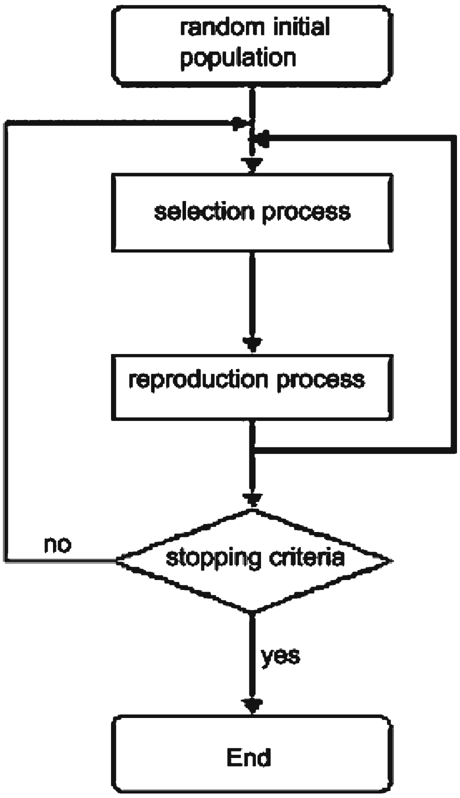

Figure 2. Flow of a basic GP.

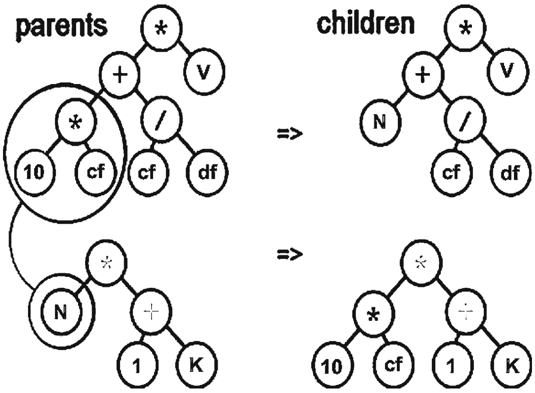

Figure 3. Example of crossover in GP.

3.1. *Existing evolutionary approaches to IR*

In recent years, there have been several approaches applying ideas from evolutionary computation to the domain of IR. Related work where a genetic programming technique is used to evolve weighting functions which outperform the traditional *tf-idf* weighting schemes has previously been conducted (Oren, 2002). Fan et al. (2004) also adopt a genetic programming framework to search for weighting functions and test them on large collections and against more modern weighting functions. Trotman (2004) also adopts this approach to evolve a general purpose ranking function and provides some analysis of the performance of individual queries. While most of these approaches have shown some increase in average precision over standard techniques, they lack a detailed analysis of any of the weighting schemes presented.

A genetic algorithm approach to modifying document representations (a set of keywords) based on user interaction has also been adopted (Gordon, 1988). By evolving sets of weights for the document (i.e. evolving the representation), better descriptions for the document in the collection can be found. Vrajitoru (2000) models the whole document collection as the genome of an individual and evolves a better representation for the whole collection.

Term-oriented models focus on finding the best discriminatory terms for improved document retrieval. For term-oriented approaches, a query is evolved so that the optimal document set is returned for a query. Horng and Yeh (2000) use this approach to extract keywords from a subset of relevant documents to construct this query and then adapt the weights to best suit the relevant documents. This approach is useful in building user profiles so that the system can learn a set of optimal terms that best describe the user's needs. Yang and Korfhage (1993) adopt a genetic algorithm approach to modify document representations by altering the weights associated with keywords.

Query expansion and adaptation techniques have been developed through evolutionary computation techniques. Vrajitoru (1998) uses genetic algorithms to evolve queries and introduces a modified crossover operator to create new queries. In information retrieval, potentially useful information is available from document tags in collections of semi-structured (e.g. html) documents. Several information retrieval systems exist which pay more attention to content associated with certain tags (e.g. title, author, keywords). Kim and Zhang (2001) attempt to learn the optimal set of tags, and their associated weights, using a genetic algorithm and demonstrate an improvement in retrieval performance.

4. Design and Experimental Setup

This section describes the experimental setup used. The terminal and function sets are defined. Benchmark weighting schemes and query weighting schemes are defined for use in the experiments. The GP approach adopted in this work evolves the weighting scheme over a number of generations. An initial population is created randomly by combining a set of primitive measures (e.g. df_t, rtf, N) using a set of operators (e.g. $+, -, \times, /$).

4.1. *Document test collections*

The three small document collections used in this research are the Medline, CISI and Cranfield collections.[1] These small collections are used for testing and training. The NPL collection is a medium sized collection available from the same source. The two larger document collections used are subsets of the TREC-9 filtering track (OHSUMED collection (Hersh et al., 1994).[2] The collection consists of abstracts from the Medline database from 1988 to 1991. The first subset we used consists of 70,825 documents from 1988 (OHSU88). The second collection consists of the documents from 1989 (OHSU89). Each collection consists of 63 queries although two queries have no relevant documents from 1988. We ignore queries that have no relevant documents associated with them. The relevance assessments for the OHSUMED collection are graded as *definitely* or *possibly* relevant. We make no distinction between *definitely* and *possibly* relevant documents in our tests and regard both grades as relevant. All documents and queries are pre-processed by removing standard stop-words from the Brown corpus[3] and are stemmed using Porter's stemming algorithm (1980). Table 1 shows some characteristics of the test collections used in this research.

4.2. *Terminal and function set*

Tables 2 and 3 show the terminal and function sets used in this research. The terminal set was chosen so that the GP could avail of

Table 1. Characteristics of document collections

Collection	Docs	Terms	Avg. len	Qrys	Terms	Avg. len
Medline	1033	10,975	56.8	30	249	11
Cranfield	1400	9014	59.6	225	639	8.8
CISI	1460	8342	47.8	76	1024	26.8
NPL	11,429	7759	18.78	93	331	6.78
OSHU88	70,825	175,021	49.40	61	195	5.05
OSHU89	74,869	185,304	50.45	63	197	4.97

Table 2. Terminal set

Terminal	Description	Domain
1	*The constant* 1	Both
rtf	Raw term frequency within a document	Local
l	Document length (number of unique words in a document)	Local
max_freq	Frequency of the most common term in a document	Local
dl	Total document length (number of words in a document)	Local
df	Number of documents a term appears in	Global
N	Number of documents in a collection	Global
V	Number of unique terms in the collection	Global
C	Collection size (number of words in the collection)	Global
cf	Collection frequency (number of times a term appears in the collection)	Global
max_c_freq	Frequency of the most common term in the collection	Global

Table 3. Function set

Function	Description
$+, \times, /, -$	Addition, multiplication, division and subtraction functions
log	The natural log
sin, tan	Trigonometric functions
$\sqrt{\ }$	Square-root function
sq	Square

as many primitive features of a term, document and collection which could possibly be related to determining the relevancy of a document. We kept the terminals primitive (atomic) so that domain specific information, and as a result bias, would be kept to a minimum (Kuscu, 2000). For each terminal in Table 2 we indicate whether the terminal is a local (within-document) measure or a global (collection wide) measure. This distinction is made by determining if the terminal value will change from document to document or remains constant throughout the collection. The constant *1* is an exception as it does not explicitly belong to either domain and so can be used in both a local and global context.

4.3. *GP parameters*

All experiments are run for 50 generations with an initial population of 1000. It was seen in our prior tests that when using the full terminal and function set, the population converges before 50 generations. Tournament selection is used and the tournament size is set to 10. The solutions are trained on an entire collection and tested for generality on the collections that are not included in training. The depth of the solution trees is limited to 6 (unless otherwise specified) to improve the generality of the solutions because shorter solutions are usually more general (Kuscu, 2000). This depth allows a large enough solution space to be searched in order to obtain high quality solutions. The creation type used is the standard ramped half and half creation method used by Koza (1992). No mutation is used in these experiments. Due to the stochastic nature of GP, a number of runs is often needed to show that the GP is converging to its best solution. We run each test 4 times and choose the best solution. We use an elitist strategy, i.e. the best individual is copied into next generation automatically.

4.4. *Training times*

Due to the nature of the GP approach, efficiency is of prime concern. For the GP, thousands of weighting schemes need to be evaluated over a document collection for many queries. For example, consider using the smallest document collection (Medline) as the training set, and choosing an initial population of 1000 solutions running for 50 generations. In this case, 50,000 weighting schemes need to be evaluated. Each evaluation requires the processing of 30 queries over 1033 separate documents. Thus, the system will process 1.5 million queries on the collection of 1033 documents. This requires searching and determining the relevance of over 1.5 billion documents. This test typically takes 6 h on the Medline collection using a standard desktop PC with a 2.0 GHz processor and 500 Mbs of RAM. Tree depth and query length also contribute to the training time as longer solutions and queries take longer to evaluate.

4.5. *Fitness function*

The average precision (AP), used as the fitness function, is calculated for each scheme by comparing the ranked list returned by the system against the human determined relevant documents for each query. Average precision is calculated using precision values for all points of recall. This is frequently used as a performance measure in IR systems.

4.6. *Benchmark term-weighting schemes*

We test our evolved scheme against the BM25 scheme introduced earlier (Equations (2) and (3)). We use the default tuning parameters of $b = 0.75$ and $k_1 = 1.2$. We also tested the BM25 with a value of 2.0 for k_1 as this is another commonly used value and found the value of 1.2 to be the best performing value on the collections used in this research.

4.7. *Matching function and query term weighting*

The matching function used in all experiments is the inner-product matching function. The weighting scheme applied to the query terms is a simple actual term frequency weighting scheme. This query weighting is applied to all weighting schemes used in this paper.

5. Results and Analysis

The first experiment aims to show that weighting schemes that achieve a high average precision can be evolved for the vector space model. We also show that these schemes are generalisable as they also achieve a high average precision on the collections which are not included in training. In the second experiment we present schemes evolved in a global (collection wide) context and show why they increase average precision over the *idf* measure. The third experiment shows some results from tests on larger collections.

5.1. *Evolving weighting schemes*

This first experiment uses all of the terminals and functions in Table 2 and Table 3. Solutions are evolved on each of the document sets to compare the generality of the solutions. The following is a typical fit solution found and it represents the best solution evolved on the Medline collection from 4 runs of the GP:

$$w_t = \frac{\frac{cf}{df} \times \left(\log(rtf) + \frac{cf}{df}\right)}{2df + l + rtf} \tag{5}$$

Table 4 shows the differences in average precision for the best solution when evolved on each document collection and tested on the other collections.

Firstly, it can be seen that the evolved weighting schemes show a significant improvement in average precision over the BM25 solution on the Medline collection. It can also be seen that the solutions achieve an average precision, on the collections on which they were not trained, which is within 2.2% of the best solution found on the collection on which they were trained (bold). This shows that the solutions evolved are quite general and exploit general natural language characteristics. The *cf/df* combination is seen to occur consistently in the fitter solutions evolved on all three collections as it does in the scheme presented (Equation (5)). It is interesting that the *cf* (collection frequency) measure is not used in traditional *tf-idf* type schemes or in the more modern BM25 scheme.

Table 4. AP for best solutions found on each collection

				Training set		
Collection	Docs	Qrys	BM25 (%)	CISI (%)	Medline (%)	Cranfield (%)
CISI	1460	76	22.67	**25.47**	24.86	24.03
Medline	1033	30	53.47	56.74	**58.85**	56.69
Cranfield	1400	225	42.08	41.33	41.85	**43.04**

Figure 4 shows the increase in average precision for the best individual and average of the population over the 50 generations for a population of 1000 for a typical run of the GP on the Medline collection. In all four tests conducted on the Medline collection the AP of the best solution from the randomly created population (0th generation) does not exceed 50% average precision and is more often well below this. Population sizes of 200 and 500 can often produce similar solutions. However, to maximise the quality of the best solution obtained from a single run it is often beneficial to use a large population. Due to the fact that our collection sizes are small we can often increase the population size and still evolve a solution in a reasonable time. However, for larger TREC style collections smaller population sizes will have to be used in order to evolve a solution in an reasonable time.

5.2. *Evolving global schemes*

In this experiment we only use the global measures from the terminal set shown in Table 2. The global part of the weighting scheme is evolved separately to investigate the properties of the *cf* measure as this appears consistently in our fittest evolved solutions and is

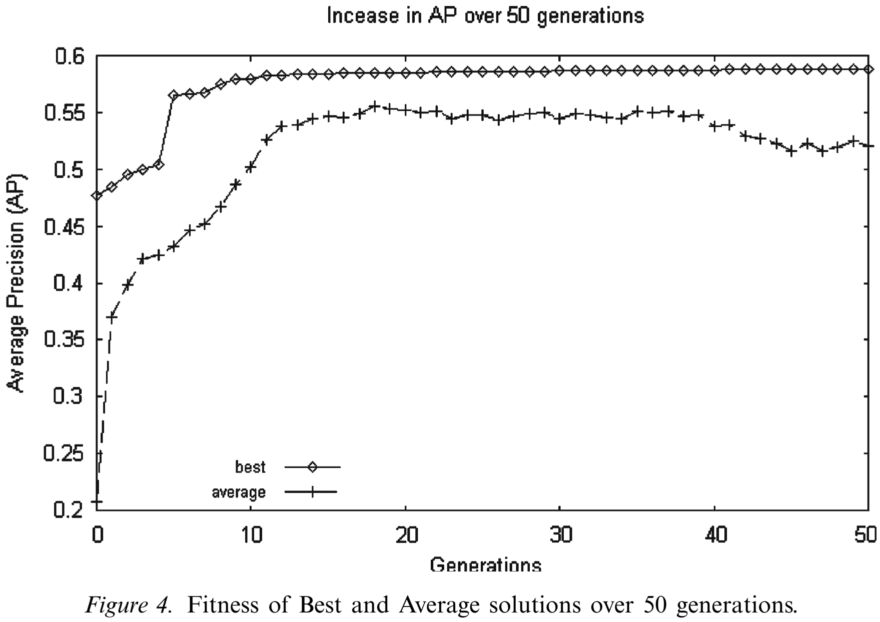

Figure 4. Fitness of Best and Average solutions over 50 generations.

not included in the terminal set in previous similar research by Oren (2002) or Fan et al. (2004) or in the BM25 scheme. The terminals used in this experiment are limited to *cf, df, 1* and *N*. We eliminated the terminals max_c_freq, C and V from the terminal set as they did not appear in any of the fitter schemes from the first experiment. A detailed analysis of the results arising from the experiment are detailed after the results are presented. The benchmark scheme used in this experiment is the idf measure (Equation (3)) as this is the global part of the BM25 scheme.

5.2.1. *Results*

The following is the best solution evolved on the CISI collection using only these global measures for 4 runs of the GP:

$$gw_t = \frac{\log(N/df)}{\sqrt{df}} \times \log\left(\frac{cf}{df}\right) \times \log(df) \qquad (6)$$

Table 5 shows the average precision for the CISI training collection and the collections that were not included in training. It is understandable that the average precision of these global schemes is lower than those in the previous experiment as we have no within-document knowledge of term-frequencies or document lengths. However, we see that the precision of the gw_t global scheme is consistently higher than that of the *idf* measure.

5.2.2. *Analysis*

Figure 5 shows the idf weight of the terms in the CISI collection when placed in rank order. The horizontal lines indicate terms of the same document frequency. The weight of these terms do not change for varying collection frequencies. When the terms in the collection are placed in rank order, as shown in Figure 6, the gw_t weight of these terms is similar to that which Luhn predicted would lead to identifying terms with a high resolving power. More recently, it has also been

Table 5. AP for idf and gw_t

Collection	Docs	Qrys	idf (%)	gw_t (%)
CISI	1460	76	18.85	22.25
Medline	1033	30	46.63	54.09
Cranfield	1400	225	33.41	37.06

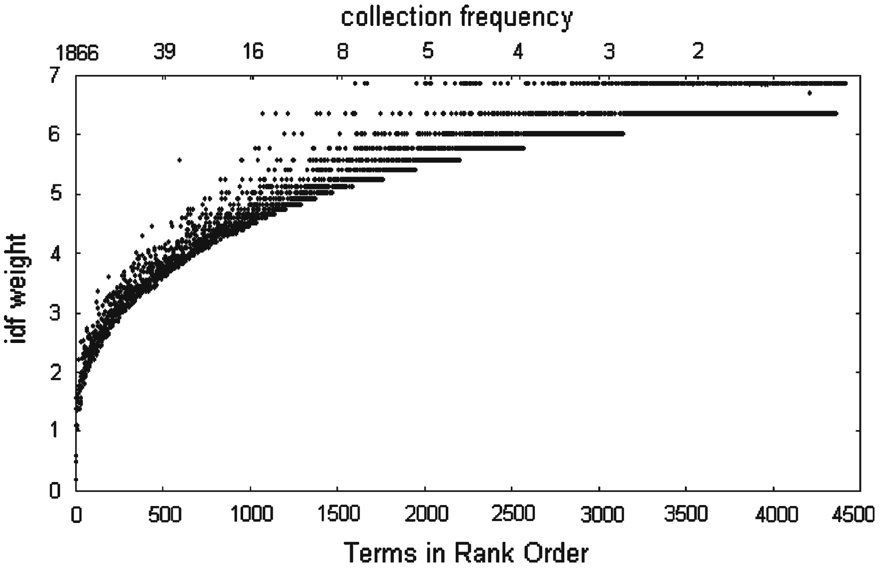

Figure 5. *idf* for terms placed in rank order for the CISI collection.

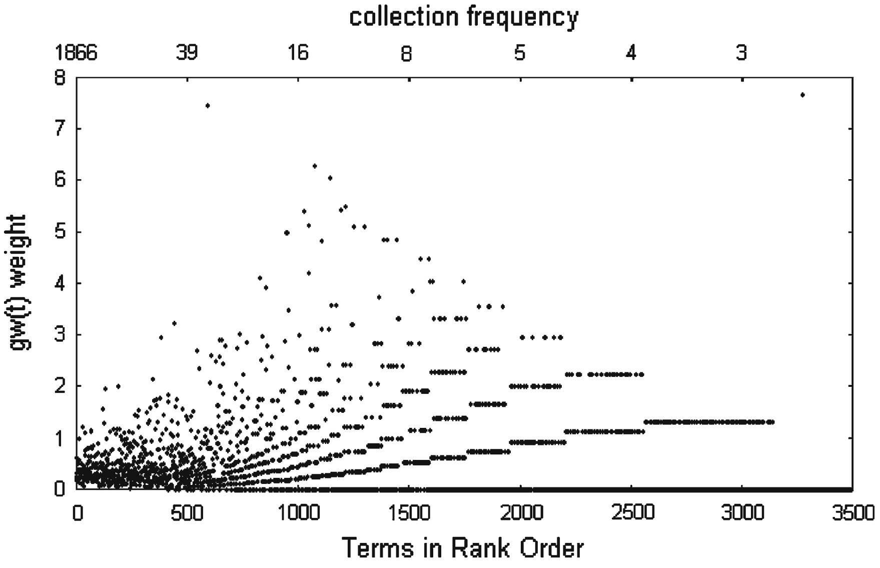

Figure 6. gw_t for terms placed in rank order for the CISI collection.

predicted that a flattening of the *idf* measure at low frequencies would result in increased precision (Greiff, 1998). This flattening typically takes place at a low collection frequency level as in Figure 6. Certain low document frequency terms are assigned a high weight while most are assigned a low weight. The reason for the flattening of the curve at low frequency levels is due to the presence of the *cf* measure. This weighting scheme completely eliminates terms of all frequencies that are totally concentrated in one document (i.e. have a document frequency of 1). It also eliminates terms that occur exactly once in every document and thus whose concentration is very low. This has the effect of considerably reducing the size of the vocabulary of the collection as words that appear once or twice represent about 65% of the vocabulary of a corpus. It is interesting that these characteristics are identified by evolutionary techniques as a variation of these are used in some feature extraction techniques (Yang and Pedersen, 1997). The traditional *idf* measure also appears in the evolved global weighting scheme, reinforcing its value in global weighting schemes. The idf weight performs well because it assigns a suitable weight for terms that appear in the majority of the documents in a collection. It can be seen that the general shape of the evolved scheme is idf in nature but in certain cases (particularly lower frequency terms) a term with a higher document frequency can be assigned a higher weight depending on its collection frequency. It should also be noted that the cf/df measure becomes a more accurate reflection of the actual term frequency at low document frequency levels. However, the cf measure contains weighting information that neither rtf nor df contain and can lead to re-ordering of documents. It has been shown that schemes that include the cf measure can achieve a higher average precision than those without this measure on certain document collections (Cummins and O'Riordan, 2004b). Of the 8342 terms in the CISI collection only 1782 (around 21%) are assigned a non-zero weight under the evolved scheme. The remaining 6560 are assigned a weight of zero and are effectively removed from the collection. Typically, these terms are low document frequency terms because the probability that they have a low concentration (i.e. $cf = df$) is higher at low frequencies. The fact that many low frequency terms are completely eliminated by our evolved global scheme is not advocated in particular and is rather an observation of the evolved schemes. However, it is again confirmed, by evolutionary techniques, that the usefulness of such terms is low for general queries. Table 6 shows the 20 highest weighted terms using the gw_t evolved scheme in the CISI collection and their frequency characteristics. It is worth noting that the document frequency is quite varied for many of the top 20 weighted terms.

Figure 7 shows the average weight assigned to terms of the same document frequency for the CISI collection. The diagram simply shows the terms in the CISI collection placed in bins according to their document frequency. The idf values are scaled in Figure 7 for viewing purposes. We show only the 10 lowest document frequencies as our aim is to show that on average the weights assigned to terms by the gw_t scheme is low for many low document frequency values. This is in contrast to the idf measure also shown for the 10 lowest document frequencies. The average weight assigned to terms using the gw_t scheme is simply the sum of the weights of terms for a particular document frequency divided by the number of terms in that document frequency bin. The 10 lowest document frequencies in the CISI represent over 86% (7,187) of the terms in the CISI collection.

5.3. *Testing schemes on larger collections*

This section introduces results on larger collections. Table 7 shows the average precision for the scheme found on the Medline collection (Equation (5)) tested on the 3 larger collections identified earlier. We see that this scheme performs considerably poorer than the BM25 scheme on all 3 larger collections.

Table 6. Top 20 terms in CISI

No.	cf	df	gw_t	No.	cf	df	gw_t
1	31	5	7.45	11	17	5	5.00
2	14	2	6.29	12	9	2	4.86
3	13	2	6.05	13	9	2	4.86
4	11	2	5.51	14	9	2	4.86
5	12	3	5.44	15	13	4	4.82
6	15	4	5.41	16	8	2	4.48
7	14	4	5.12	17	8	2	4.48
8	11	3	5.10	18	14	5	4.21
9	11	3	5.10	19	20	7	4.12
10	17	5	5.00	20	7	2	4.05

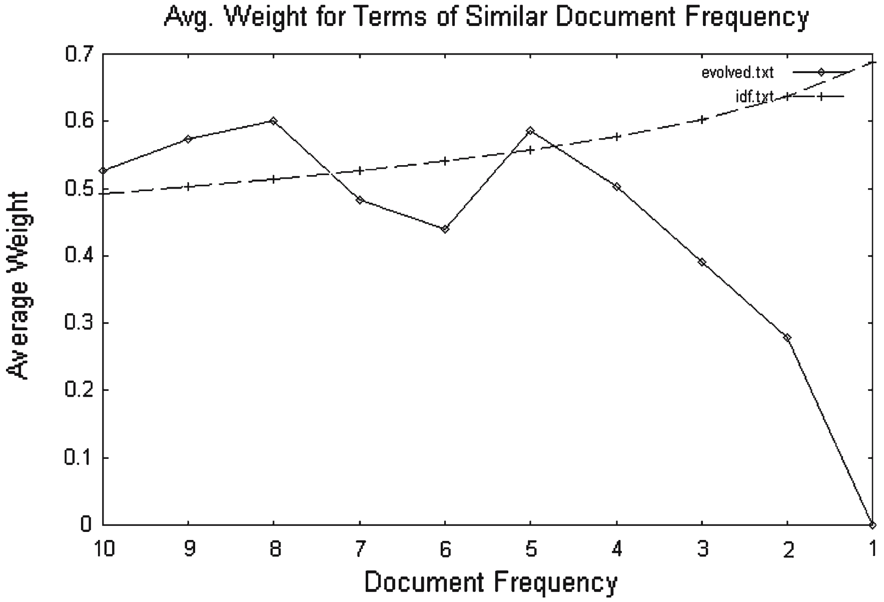

Figure 7. Average weight for terms with the 10 lowest document frequencies.

Table 7. AP for BM25 and w_t (5)

Collection	Docs	Qrys	BM25 (%)	$w_t(5)$ (%)
NPL	11,429	93	28.02	21.94
OHSU88	70,825	61	32.49	25.77
OHSU89	74,869	63	30.51	24.10

Table 8 shows the average precision of the gw_t scheme (Equation (6)) tested on the NPL, OHSU88 and OHSU89 collections. It is promising that the average precision of the gw_t scheme is higher than idf on all larger collections.

Table 8. AP for idf and gw_t

Collection	Docs	Qrys	idf (%)	gw_t (%)
NPL	11,429	93	25.66	28.89
OHSU88	70,825	61	25.75	27.83
OHSU89	74,869	63	26.06	27.70

As the performance of the gw_t scheme is higher than the w_t evolved weighting (Equation (5)) on all 3 larger collections, it is logical to assume that the local (within-document) part of the w_t solution (Equation (5)), which was evolved on the smaller Medline collection, cannot be applied directly to the larger collections. We can conclude that the global (collection-wide) characteristics that leads to an increase in average precision on small collections also leads to an increase in average precision on large collections. However, the local (within-document) relevance characteristics of the smaller collections are different than those of larger collections. Thus, in order to develop a complete weighting scheme which will increase average precision on large collections it will be neccessary to evolved the local (within-document) part of the weighting scheme on larger collections so that it can interact correctly with the gw_t scheme.

In the OHSU88 collection, only 32,774 of 175,021 terms recieve a non-zero weight using the gw_t weighting. Thus, less than 20% of the terms in the corpus are used after preprocessing and an increase in average precision is still observed. We can see that the gw_t scheme is particularly good at correctly weighting terms that have good retrieval properties. This characteristic may be of benefit in certain feature extraction techniques that use a reduced feature space. Table 9 shows the top 20 weighted terms in the OHSU88 collection and their frequency characteristics. It is interesting to see some middle document frequency terms (e.g. $df = 10$) being assigned a very high weight in this larger collection.

6. Conclusion

Weighting schemes can be evolved for small test collections that achieve a higher average precision than that of the BM25 scheme. The cf measure is an integal part of these weighting schemes. The global part of the evolved schemes have characteristsics similar to Luhn's resolving power. The global scheme evolved separately on the smaller collection is also shown to increase average precision over idf on large collections. This global scheme is also shown to use a limited number of indexing terms. The complete schemes evolved on the smaller collections (e.g. Equation (5)) do not increase the average precision over the BM25 scheme on larger collections. The local part of the evolved weighting is specific to the small collections. For future work, local schemes will be evolved on larger collections depen-

Table 9. Top 20 Terms in OHSU88

No.	cf	df	gw_t	No.	cf	df	gw_t
1	38	4	15.26	11	25	3	13.54
2	35	4	14.71	12	25	3	13.54
3	35	4	14.71	13	43	6	13.51
4	29	3	14.49	14	29	4	13.43
5	33	4	14.31	15	35	5	13.39
6	28	3	14.27	16	24	3	13.28
7	88	10	14.04	17	24	3	13.28
8	87	10	13.96	18	28	4	13.19
9	30	4	13.66	19	28	4	13.19
10	25	3	13.54	20	34	5	13.18

dent on the evolved global schemes that achieve a higher average precision than idf. In supplying more evidence about the actual documents (within-document measures) to the GP, it is likely that the average precision of these schemes will be increased on larger collections.

Eliminating certain terms in a collection may harm the recall of the system. By assigning at least some weight to these terms, the average precision of the scheme may be increased for certain queries. This issue will be investigated in future work.

Acknowledgements

This work is being carried out with the support of IRCSET (the Irish Research Council for Science, Engineering and Technology) under the Embark Initiative. The authors would also like to thank the reviewers for their useful comments.

Notes

1. ftp://ftp.cs.cornell.edu/pub/smart
2. http://trec.nist.gov/data/t9_filtering.html
3. http://www.lextek.com/manuals/onix/stopwords1.html

References

Cummins, R. & O'Riordan, C. (2004a). Determining General Term Weighting Schemes for the Vector Space Model of Information Retrieval Using Genetic Programming. In *15th Artificial Intelligence and Cognitive Science Conference (AICS 2004)*. Galway-Mayo Institute of Technology, Castlebar Campus, Ireland.

Cummins, R. & O'Riordan, C. (2004b). Using Genetic Programming to Evolve Weighting Schemes for the Vector Space Model of Information Retrieval. In Keijzer, M. (ed.) *Late Breaking Papers at the 2004 Genetic and Evolutionary Computation Conference*. Seattle: Washington, USA.

Darwin, C. (1859). *The Origin of the Species by means of Natural Selection, or The Preservation of Favoured Races in the Struggle for Life*. First edition.

Fan, W., Gordon, M. D. & Pathak, P. (2004). A Generic Ranking Function Discovery Framework by Genetic Programming For Information Retrieval. *Information Processing & Management*.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimisation and Machine learning*. Addison-Wesley.

Gordon, M. (1988). Probabilistic and Genetic Algorithms in Document Retrieval. *Communication of the ACM* 31(10), 1208–1218.

Greiff, W. (1998). A Theory of Term Weighting Based on Exploratory Data Analysis. In *Proceedings of the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98)*. Melbourne, Australia.

Hersh, W., Buckley, C. Leone, T. J. & Hickam, D. (1994). OHSUMED: An Interactive Retrieval Evaluation and New Large Test Collection for Research. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 192–201, Springer-Verlag: New York, Inc.

Horng, J. & Yeh, C. (2000). Applying Genetic Algorithms to Query Optimization in Document Retrieval. *Information Processing & Management* **36**(5): 737–759.

Kim, S. & Zhang, B. T. (2001). Evolutionary Learning of Web-Document Structure for Information Retrieval. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC2001)*.

Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA.

Kuscu, I. (2000). Generalisation and Domain Specific Functions in Genetic Programming. In *Proceedings of the 2000 Congress on Evolutionary Computation CEC00*. 1393–1400, IEEE Press.

Luhn, H. (1958). The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development* 159–165.

Oren, N. (2002). Re-examining tf.idf Based Information Retrieval with Genetic Programming. *Proceedings of SAICSIT*.

Porter, M. (1980). An algorithm for Suffix Stripping. *Program* **14**(3): 130–137.

Robertson, S. E. & Sparck Jones, K. (1976). Relevance Weighting of Search Terms. *Journal of American Society for Information Sciences* **27**(3): 129–146.

Robertson, S. E. & Walker, S. (1997). On Relevance Weights with Little Relevance Information. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 16–24, ACM Press.

Robertson, S. E., Walker, S., Hancock-Beaulieu, M., Gull, A. & Lau, M. (1995). Okapi at TREC-3. In Harman, D. K. (ed.) *The Third Text REtrieval Conference (TREC-3) NIST*.

Salton, G. & Buckley, C. (1988). Term-Weighting Approaches in Automatic Text Retrieval. *Information Processing & Management* **24**(5): 513–523.

Salton, G., Wong, A. & Yang, C. S. (1975). A Vector Space Model for Automatic Indexing. *Communications of the ACM* **18**(11), 613–620.

Salton, G. & Yang, C. S. (1973). On the Specification of Term Values in Automatic indexing. *Journal of Documentation* **29**: 351–372.

Trotman, A. (2004). An Artificial Intelligence Approach to Information Retrieval (Abstract Only). In *SIGIR*. p. 603.

Van Rijsbergen, C. J. (1979). *Information Retrieval*, 2nd ed. Department of Computer Science, University of Glasgow.

Vrajitoru, D. (1998). Crossover Improvement for the Genetic Algorithm in Information Retrieval. *Information Processing and Management* **34**(4): 405–415.

Vrajitoru, D. (2000). In Crestani, F. & Pasi, G. (eds.) *Soft Computing in Information Retrieval. Techniques and Applications*, 199–222. Physica-Verlag.

Yang, J. & Korfhage, R. (1993). Query Optimization in Information Retrieval Using Genetic Algorithms. In: *Proceedings of the Fifth International Conference on Genetic Algorithms*. 603–611.

Yang, Y. & Pedersen, J. O. (1997). A Comparative Study on Feature Selection in text Categorization. In Fisher, D. H. (ed.) *Proceedings of ICML-97, 14th Inter- national Conference on Machine Learning. Nashville*, US, 412–420, Morgan Kaufmann Publishers: San Francisco, US.

Zipf, G. (1949). *Human Behaviour and the Principle of Least Effort*. Addison-Wesley: Cambridge, MA.