National University of Ireland, Galway
Ollscoil na hÉireann, Gaillimh

DEPARTMENT OF INFORMATION TECHNOLOGY

technical report NUIG-IT-071204

Evolving, Analysing and Improving Global Term-Weighting Schemes in Information Retrieval

R. Cummins (NUI, Galway)
C. O'Riordan (NUI, Galway)

itreports@nuigalway.ie http://www.it.nuigalway.ie/

Evolving, Analysing and Improving Global Term-Weighting Schemes in Information Retrieval

Ronan Cummins and Colm O'Riordan
Dept. of Information Technology,
National University of Ireland,
Galway, Ireland.
ronan.cummins@nuigalway.ie
colmor@it.nuigalway.ie

January 24, 2005

Abstract

The ability of a term to distinguish documents, and ultimately topics, is crucial to the performance of many Information Retrieval models. We present and analyse global weighting schemes for the vector space model developed by means of evolutionary computation. The global schemes presented are shown to increase average precision over the *idf* measure on TREC data. The global schemes are also shown to be consistent with Luhn's theory of resolving power as certain middle frequency terms are assigned the highest weight. The use of the collection frequency measure of a term is seen as crucial to the performance of these schemes. We also show that the analysis of these evolved schemes is an important step to understanding and improving their performance.

1 Introduction

In recent years there has been more and more attempts applying machine learning techniques to the domain of Information Retrieval (IR). Machine learning techniques are proving a viable alternative to other standard analytical methods in many areas of IR. Genetic Algorithms and Genetic Programming (GP) [1] have been shown to be effective approaches to learning term-weights and term-weighting schemes in IR. Inspired by the Darwinian theory of Natural Selection [2], these approaches are stochastic in nature and efficient for searching large complex search spaces. Gordon [3] uses a genetic algorithm approach to modifying document representations (a set of keywords) based on user interaction. By evolving sets of weights for the document (i.e. evolving the representation), better descriptions for the document in the collection can be found. More recently, a genetic programming technique has been used to evolve weighting functions which outperform standard weighting schemes in a vector space framework [4, 5, 6]. However, in a number of these approaches, an analysis as to why the evolved solutions achieve a high average precision is not presented. It is important to critically analyse such solutions to gain an understanding of the solutions obtained from this stochastic search process.

This paper outlines a similar Genetic Programming process which evolves global term-weighting schemes for the vector space model [7] and explains why they achieve an increase in average precision higher than that of the *idf* measure. We evolve the global weighting separately for several reasons. For a term-weighting scheme, firstly, terms that aid the retrieval of documents should be promoted (traditionally this is achieved using *idf*). Then, the documents which contains these terms should be examined and weighted appropriately depending on the document specific characteristics. By evolving the global weighting separately, the analysis of the schemes evolved is made easier and the overall search space is reduced.

Section 2 introduces term discrimination models and specifically the vector space model of IR. Standard term-weighting approaches for the vector space model are also discussed. Section 3 introduces the Probabilistic model of IR and modern weighting approaches derived from such a model are introduced. The Genetic Programming approach is discussed in section 4. Section 5 details the experimental setup. Results and analysis are presented in section 6 and finally our conclusions are summarised in section 7.

2 Term-discrimination

This section presents background material for term discrimination models. Luhn's [8] and Zipf's [9] contribution to IR is summarised. The vector space model is outlined and traditional term-weighting schemes are introduced.

2.1 Early IR Advances

Zipf [9] showed that the frequency of terms in a collection when placed in rank order approximately follows a log curve. Luhn [8] proposed that terms that occur too frequently have little power to distinguish between documents and that terms that appear infrequently are also of little use in distinguishing between documents. Thus in Figure 1, the bell-shaped curve of the graph relates the frequency of terms to their distinguishing (resolving) power.

Salton et al. [10, 7] validate much of Luhn's work with empirical analysis. 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 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 are, on average, poor discriminators while high frequency terms the least useful [7].

2.2 Vector Space Model

The classic vector space model [7] represents each document and query as a vector of terms with weights associated to each term. A matching function is used to compare each document vector to the query vector. One common matching function is the inner-product measure and is calculated as follows:

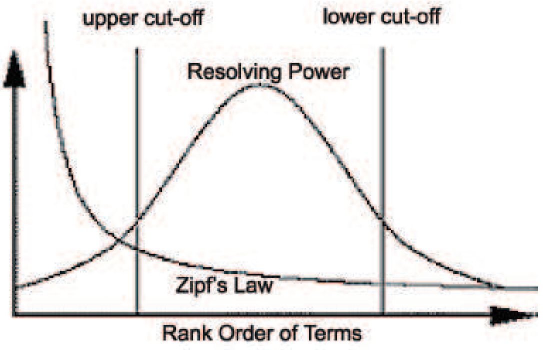

Figure 1: Zipf's law and Luhn's proposed cut-off points - adapted from [8]

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

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

2.3 Term-Weighting in the VSM

Yu and Salton [11] suggest that the best distinguishing terms are terms which occur with a high frequency in certain documents but whose overall frequency across a collection is low (low document frequency). They conclude from this that a term weighting function should vary directly with term frequency and inversely with document frequency. The *idf* scheme, first introduced by Sparck Jones [12], gives a higher weight to low frequency terms. Thus, the curve of the *idf* measure is an inverse of the Zipfian curve in Figure 1. The *idf* is calculated as follows:

$$idf_t = log(\frac{N}{df_t}) \qquad (2)$$

where N is the number of documents in the collection and df_t is the number of documents containing term t. It can be seen that the weight assigned to terms by *idf* is inconsistent with Luhn's resolving power at low frequency levels.

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.

3.1 Binary Independence Model

The Binary Independence Model [13] developed by Robertson and Sparck Jones calculates the optimal weights for terms based on a set of relevant documents. The effectiveness of this model depends on the availability of existing relevance information usually supplied by relevance feedback techniques. Croft and Harper [14] investigate the weighting of terms in the probabilistic model when no relevance information is available.

3.2 Modern weighting for BM25

A modern weighting scheme developed by Robertson et al. [15] is the Okapi-BM25 weighting scheme. The global part of this weighting scheme is a variation of the traditional *idf* measure.

$$idf_{bm} = log(\frac{N - df_t + 0.5}{df_t + 0.5}) \tag{3}$$

However, Robertson and Walker [16] have indicated that the probability of a term occurring in a relevant document goes to zero as the frequency of the term occurring in the collection goes to zero. Greiff [17] has predicted using probabilistic techniques that a flattening of the *idf* measure at both high and low frequencies would lead to an increase in performance.

4 Genetic Programming

John Koza [1] developed Genetic Programming (GP) in the early 1990's. The GP approach has helped solve problems in a wide variety of areas. GP is inspired by the Darwinian theory of natural selection [2] and can be thought of 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 basic flow of a GP is as follows: 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. Once this is done, selection can occur. Goldberg [18] 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 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 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 (figure 2). The other form of reproduction is mutation. Mutation (asexual reproduction) is the random change of 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 certain number of generations, or until convergence of the population is achieved or after an individual is found with an acceptable fitness.

5 Experimental Setup

The GP approach adopted evolves the global term-weighting scheme over a number of generations. A set of primitive terminals (eg. cf, *df* and N) are combined using a set of operators (e.g. $+$, $-$ and $\times$)

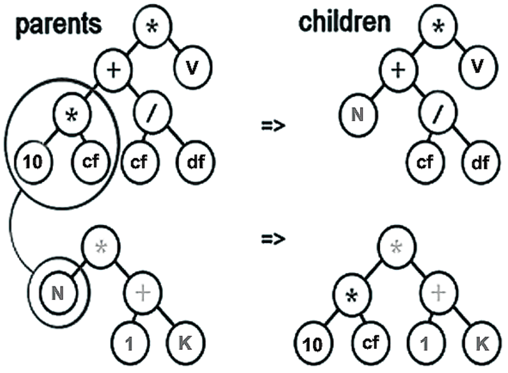

Figure 2: Example of crossover in GP

to produce weighting schemes that aim to maximise average precision.

5.1 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. The global weighting scheme is evolved using a binary weighting applied to the local (within document) weights.

5.2 Document Collections

The document collections used in this research are subsets of the TREC-9 filtering track (OHSUMED collection [19])[1]. The collection consists of abstracts from the Medline database from 1988 to 1991. The OHSU88 subset we used consists of documents from 1988 (OHSU88). The OHSU89 collection consists of documents from 1990. Each collection consists of 63 queries although 2 queries have no relevant documents from 1988. We ignore queries that have no relevant documents associated with them. Our training data consists of the first half of the 1988 subset (TRAIN88). All experiments are conducted using the TRAIN88 as our training set. The relevance assessments for the OHSUMED collection are graded as *definitely* or *possibly* relevant. We make no distinction between *definitely* and *possibly* relevant and regard both grades as relevant. We also use the NPL and Cranfield collections[2] as these are of significantly different sizes. We wish our weighting scheme to be general across various size collections. The documents and queries are pre-processed by removing standard stop-words from the Brown corpus[3] and are stemmed using Porter's stemming algorithm [20]. Table 1 shows some characteristics of the test collections used in this research.

Table 1: Characteristics of document collections

Collection	*Docs*	*Qrys*	*Avg_doc_len*	*Avg_qry_len*
TRAIN88	35,412	61	48.02	5.05
OHSU88	70,825	61	49.40	5.05
OHSU89	74,869	63	50.45	4.97
NPL	11,429	93	18.78	6.78
Cranfield	1,400	225	59.60	8.8

5.3 GP Parameters

Tests are run for 50 generations with an initial population of 100. It is seen from prior tests that when using this terminal and function set, the population converges before 50 generations. Tournament selection is used and the tournament size is set to 4. The experiments were trained on the first half of the OHSU88 collection (TRAIN88) and tested for generality on the OHSU88 and OHSU89 collections. The depth of each solution is limited to 6 to improve the generality of the solutions. This depth allows a large enough solution space to be searched and does not compromise the quality of the solutions obtained significantly. The creation type used is the standard ramped half and half creation method used by Koza [1]. 4% mutation is used in our experiments. Due to the stochastic nature of GP a number of runs is often needed to allow the GP converge to a suitable solution.

[1] http://trec.nist.gov/data/t9_filtering.html

[2] ftp://ftp.cs.cornell.edu/pub/smart

[3] http://www.lextek.com/manuals/onix/stopwords1.html

5.4 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.

5.5 Terminal and Function Sets

Tables 2 and 3 show the terminal and function sets used in our experiments. It is worth noting that all the global measures included are in a primitive (unprocessed) form as our intention is to allow the GP to combine these primitve measures in an unbiased way.

Table 2: Terminal Set

Terminal	*Description*
1	*the constant* 1
10	*the constant* 10
N	no. of documents in the collection
df	document frequency of a term
cf	collection frequency of a term
V	vocabulary of collection
C	size of collection
0.5	*the constant* 0.5

Table 3: Function Set

Function	*Description*
+, ×, /, -	arithmetic functions
log	the natural log
$\sqrt{\ }$	square-root function
sq	square

6 Results and Analysis

In this section we present results from the experiments. Analysis of the results is also presented. The benchmark scheme used in the empirical comparison is the idf_{bm} measure as defined in the BM25 scheme (3). This measure is closely related to the traditional *idf* measure and its performance is similar.

6.1 Evolving a Global Weighting Scheme

The following formula is the best global weighting found after 4 separate runs of the GP with a population of 100 for 50 generations:

$$gw_t = log(\frac{cf_t}{df_t}) \times \sqrt{\frac{N}{df_t} \times (\frac{1}{df_t} + 1)} \qquad (4)$$

As we can see in table 4 the average precision of the evolved scheme is nearly 3% higher than that of idf_{bm} on the training collection. When tested on the full OHSU88 and OHSU89 collections the increase in average precision is about 2% in each case. There is also an increase in average precision seen on the NPL and Cranfield collections in the range of about 3%.

Table 4: AP for idf_{bm} and gw_t

Collection	*Docs*	*Qrys*	idf_{bm}	gw_t
TRAIN88	35,412	61	19.22%	22.10%
OHSU88	70,825	61	25.74%	27.73%
OHSU89	74,869	63	26.06%	28.07%
NPL	11,429	93	25.66%	28.49%
Cranfield	1,400	225	33.64%	37.15%

Table 5 shows the 11 precision-recall points for the OHSU88 and OHSU89 collections. We see that on the OSHU88 collection the precision at low recall levels is greater than that of idf_{bm}. While on the OHSU89 collection the precision at low recall levels is slightly less than that of idf_{bm}. However, the precision at mid recall levels on the OHSU89 collection is greater than that of idf_{bm}.

Figures 3 and 4 show the increase in average precision of gw_t over idf_{bm} for each of the 63 queries in two collections. On the OHSU88 collection we can see that the evolved solution achieves a higher average precision on 40 of the queries. Five of the queries show no difference in average precision while 18 queries achieve a lower average precision under the evolved weighting. On the OHSU89 collection we can see that the evolved solution achieves

Table 5: Precision-Recall for idf_{bm} and gw_t

	$OHSU88$ $(\%AP)$		$OHSU89$ $(\%AP)$	
$Recall$	idf_{bm}	gw_t	idf_{bm}	gw_t
0%	57.50	61.93	63.02	62.08
10%	49.62	54.11	55.37	56.57
20%	43.68	44.02	40.60	45.92
30%	35.89	36.99	33.16	38.39
40%	30.85	31.98	26.94	31.04
50%	28.53	27.99	23.80	26.82
60%	20.66	22.36	21.38	23.54
70%	15.38	17.72	16.33	17.54
80%	12.51	14.75	12.73	14.06
90%	8.05	9.08	8.55	10.37
100%	5.18	5.67	6.37	5.94

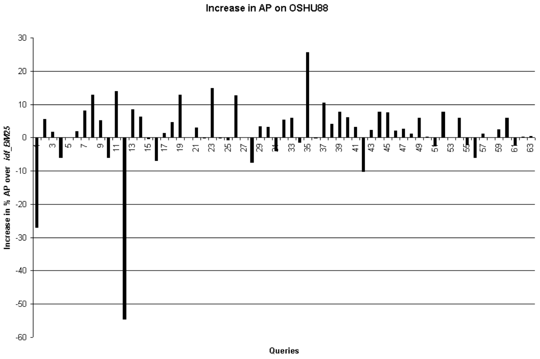

Figure 3: AP Histogram for gw_t vs idf_{bm} for queries

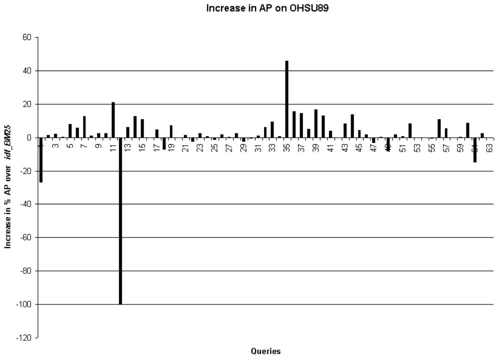

Figure 4: AP Histogram for gw_t vs idf_{bm} for queries

a higher average precision on 45 of the queries. Six queries show no difference in average precision while 12 queries have a lower average precision than the idf_{bm} scheme. Three queries (1, 12 and 61) do noticeably worse on both collections. Query 12 in particular needs to be investigated to see why it performs significantly worse under the evolved solution in terms of average precision than the idf_{bm} solution.

6.2 Analysis of Global Weighting

Figures 5 and 6 show the terms in the OHSU88 collection placed in rank order and the idf_{bm} and gw_t weight applied to each term in the collection. When the terms in the collection are placed in rank order and the gw_t weight is applied, certain middle frequency terms are assigned the highest weight. The gw_t weighting scheme gives many terms a weight which is quite different from that of the idf_{bm} measure identified in Figure 5. The reason for the flattening of the curve at lower frequency levels in Figure 6 is due the presence of the cf measure in our evolved solution. It has been shown that including the cf measure in term-weighting schemes can have a large increase in average precision on some small collections [21]. In Figure 6 it can be seen that a term with a lower document frequency does not always get a higher weight. A few low document frequency terms are assigned the highest weight while most low document frequency terms recieve a low weight (zero in many cases). Only the first 80,000 ranked terms are shown in Figures 5 and 6. The remaining terms have a collection frequency of 1. For the gw_t weight terms that occur once in the collections are assigned a weight of 0 because of the $log(cf/df)$ part of the weighting. In Figure 6, it is interesting to see the change in weights assigned to terms of various collection frequency. Each term of a specific collection frequency has cf number of weights it can be possibly be assigned. This is in contrast to idf_{bm}. This is shown in Figure 5 by the horizontal lines indicating weights assigned to terms of the same document frequency. Terms where $cf = df$ represent terms which have a low concentration in the documents in which they appear. The elimination of terms that occur once in the collection has the effect of con-

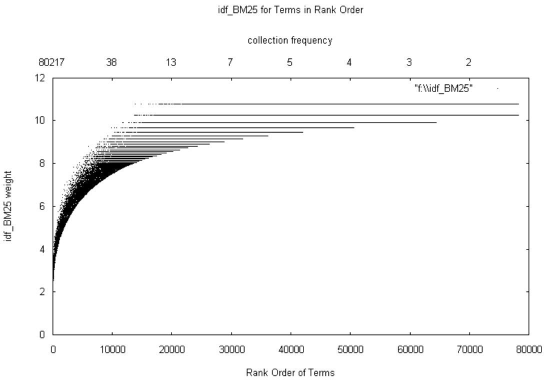

Figure 5: idf_{bm} for terms placed in rank order

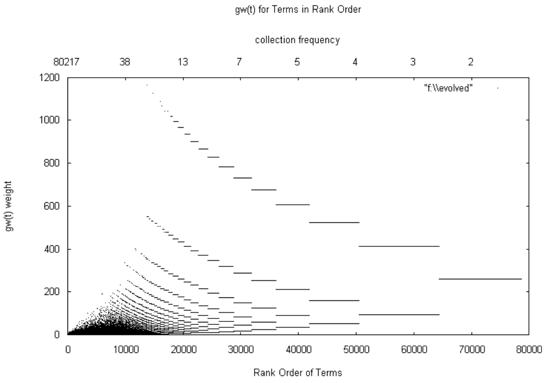

Figure 6: gw_t for terms placed in rank order

siderably reducing the size of the vocabulary of the collection as words that appear once typically represent at least 50% of the vocabulary of a corpus. It is interesting that these characteristics are identified by evolutionary techniques, since a variation of these can be seen in some feature extraction techniques like document-frequency thresholding [22]. However, the removal of terms with a collection frequency of 1, or terms where $cf = df$, may leave certain documents irretrievable. This may lead to a poor performance for queries which contain these types of terms. Thus, assigning a small weight to these terms may be beneficial.

Van Rijsbergen [23] summarizes the characteristics of useful terms as follows, "A term with high total frequency of occurrence is not very useful in retrieval irrespective of its distribution. Middle frequency terms are most useful particularly if the distribution is skewed. Rare terms with a skewed distribution are likely to be useful but less so than the middle frequency ones. Very rare terms are also quite useful but come bottom of the list except for the ones with a high total frequency". The evolved weighting depicted in Figure 6 has many of these characteristics. The cf/df measure, which consistently appears in the fitter solutions from the GP, has been used by Kwok [24] to improve the performace of short queries. Pirkola and Jarvelin [25] also use this measure to improve the resolution power of certain search keys.

6.3 Evolving Improvements

As stated in the previous section, certain documents may be left irretrievable due to the $log(cf/df)$ part of the evolved solution (4) which eliminates terms that have a low concentration in certain documents. As mentioned earlier, query 12 (5 terms) which contains two occurences of a term which has these characteristics ($df = cf = 5$ in OHSU88 and $df = cf = 2$ in OHSU89) performs significantly worse under the evolved scheme. This query has two relevant documents in both collections. It would seem that in eliminating this term the retrieval of this query's relevant documents is seriously affected. It would be logical to assume that in assigning at least a small weight to these terms that average precision would increase. It is also worth noting that this characteristic (i.e. $cf = df$) is most likely to occur at lower frequency levels. Thus, we propose the following modification to the formula outlined earlier:

$$gw_t = log(\frac{cf_t + k_1}{df_t}) \times \sqrt{\frac{N}{df_t} \times (\frac{1}{df_t} + 1)} \qquad (5)$$

where k_1 is some positive real number. It would seem logical to assume that k_1 is also a function of the collection frequency and/or document frequency of a term and that it is very small value for large frequency terms and higher value for lower frequency terms to fit in with the scheme depicted in Figure 6. Typically terms that have a higher frequency will not be eliminated from the collection. Thus, adding a constant to these terms would disrupt the relative weights between them and affect the retrieval of these terms which were found to be useful by the original evolutionary process. Thus, evolving the k_1 in this formula should increase the average precision of the queries with these types of terms while not harming other queries. We evolved k_1 using all the functions and terminals in tables 2 and 3 while keeping the original gw_t formula constant. After 3 runs of the GP the following is one of the best and simplest formulas found:

$$k_1 = \frac{0.5}{\sqrt{\sqrt{cf}}} = \frac{0.5}{cf^{\frac{1}{4}}} \tag{6}$$

Table 6 shows the average precision for the evolved gw_t scheme with the evolved k_1. It is understandable that the increase in average precision on the training set is not significant since if a significantly better solution existed for the training set the original GP approach should have been able to evolve such a solution. However, when tested on the OHSU88 and OHSU89 collections the increase in average precision is about 0.5% and 1.5% respectively as seen in table 5. This is because many of the relevant documents, which contain terms previously eliminated by the original evolved scheme, do not appear in the training set.

Table 6: AP for idf_{bm}, gw_t, gw_t with k_1

Collection	*Docs*	*Qrys*	idf_{bm}	gw_t	gw_t with k_1
TRAIN88	35,412	61	19.22%	22.10%	22.28%
OHSU88	70,825	61	25.74%	27.73%	28.24%
OHSU89	74,869	63	26.06%	28.07%	29.66%
NPL	11,429	93	25.66%	28.49%	28.31%
Cranfield	1,400	225	33.64%	37.15%	37.51%

Only five queries show a change in average precision after adding in the evolved k_1 factor on OHSU88. Figure 7 shows the increase in average precision for gw_t and gw_t with k_1 against the idf_{bm} weight. Importantly, all of these 5 queries see an increase in average precision. We see that query 28 is the only query that was performing worse than idf_{bm} and is now performing better. Query 12 however only shows a modest increase (5%) in average precision using the modifified scheme. However, it is worth noting that four of the five queries contained a term for which $cf = df$. Query 13 did not contain such a term and shows the smallest increase in average precision over the weighting presented in (4).

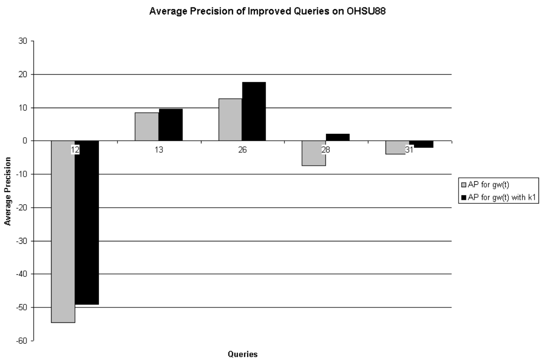

Figure 7: AP Histogram for gw_t and gw_t with k_1 against idf_{bm} for the 5 queries

On the OHSU89 collection (not depicted), only one query showed any difference in average precision under the modified evolved scheme. Query 12 which was performing poorly against idf_{bm} on OHSU89 has now the same average precision as idf_{bm} (100%) on this collection. This is solely responsible for the rise in average precision on the OHSU89 collection. In all cases on both large collections, the modified evolved scheme (5) performed as well as, or better than, the original evolved scheme (4). This seems to indicate that a small weight applied to such terms can increase the average precision for certain queries. The average precision on the two smaller collections is only slightly affected. On the NPL collection the average precision decreases slightly but increases slightly on the Cranfield collection. It is interesting that although the characteristics that lead to a larger increase in average precision on the OHSU89 collection are not present in the training set the modified solution significantly aids performance on the OHSU88 and OHSU89 collection. Thus, it is important to analyse these evolved schemes not just in order to show why they improve performance but also in order to develop better schemes.

7 Conclusion

We have shown that global weighting schemes can be found by by evolutionary techniques that outperform *idf* on general collections. We have shown that these evolved schemes achieve an increase in average precision without the use of within document measures. We have shown that the increase in performance is consistent across collections of various sizes. Reasons for the poor performance of some queries is investigated and a modification to the original evolved weighting is suggested. In particular, we have shown that it is beneficial and often neccessary to analyse weighting schemes produced from evolutionary techniques in order to further improve their performance. Future work includes evolving and analysing local (within document) term-weighting schemes that depend on global evolved weightings to produce full evolved general weighting schemes for the vector space model.

Acknowledgments

This work is being carried out with the support of IRCSET (the Irish Research Council for Science, Engineering and Technology) under the Embark Initiative.

References

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

[2] Darwin, C.: The Origin of the Species by means of Natural Selection, or The Preservation of Favoured Races in the Struggle for Life. First edn. (1859)

[3] Gordon, M.: Probabilistic and genetic algorithms in document retrieval. Commun. ACM **31** (1988) 1208–1218

[4] Oren, N.: Re-examining tf.idf based information retrieval with genetic programming. Proceedings of SAICSIT (2002)

[5] Fan, W., Gordon, M.D., Pathak, P.: A generic ranking function discovery framework by genetic programming for information retrieval. Information Processing & Management (2004)

[6] Trotman, A.: An artificial intelligence approach to information retrieval (abstract only). In: SIGIR. (2004) 603

[7] Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM **18** (1975) 613–620

[8] Luhn, H.: The automatic creation of literature abstracts. IBM Journal of Research and Development (1958) 159–165

[9] Zipf, G.: Human Behaviour and the Principle of Least Effort. Addison-Wesley, Cambridge, Massachusetts (1949)

[10] Salton, G., Yang, C.S.: On the specification of term values in automatic indexing. Journal of Documentation **29** (1973) 351–372

[11] Yu, C.T., Salton, G.: Precision weighting - an effective automatic indexing method. Journal of the ACM **23** (1976) 76–88

[12] Sparck Jones, K.: A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation **28** (1972) 11–21

[13] Robertson, S.E., Sparck Jones, K.: Relevance weighting of search terms. J. Am. Soc. for Information Sciences **27** (1976) 129–146

[14] Croft, W.B., Harper, D.J. In: Using probabilistic models of document retrieval without relevance information. Morgan Kaufmann Publishers Inc. (1997) 339–344

[15] Robertson, S.E., Walker, S., Hancock-Beaulieu, M., Gull, A., Lau, M.: Okapi at TREC-3. In: In D. K. Harman, editor, The Third Text REtrieval Conference (TREC-3) NIST. (1995)

[16] Robertson, S.E., Walker, S.: On relevance weights with little relevance information. In: Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press (1997) 16–24

[17] Greiff, W.: 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 (1998)

[18] Goldberg, D.E.: Genetic Algorithms in Search, Optimisation and Machine learning. Addison-Wesley (1989)

[19] Hersh, W., Buckley, C., Leone, T.J., Hickam, D.: 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, Springer-Verlag New York, Inc. (1994) 192–201

[20] Porter, M.: An algorithm for suffix stripping. Program **14** (1980) 130–137

[21] Cummins, R., O'Riordan, C.: 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 (2004)

[22] Lewis, D.: Feature selection and feature extraction for text categorization. Proceedings of Speech and Natural Language Workshop (1992) 212–217

[23] Van Rijsbergen, C.J.: Information Retrieval, 2nd edition. Dept. of Computer Science, University of Glasgow (1979)

[24] Kwok, K.L.: A new method of weighting query terms for ad-hoc retrieval. In: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press (1996) 187–195

[25] Pirkola, A., Jarvelin, K.: Employing the resolution power of search keys. J. Am. Soc. Inf. Sci. Technol. **52** (2001) 575–583