Multi-Objective Optimisation

for

Information Access Tasks

Michelle J. Fisher Jonathan E. Fieldsend Richard M. Everson

University of Exeter*

Abstract

Information access tasks involve setting a large number of model parameters, for example choosing which terms from a document collection and which term weighting methods to use. Parameter values also have to be set, for example term weighting method parameters and thresholds. In addition, information access tasks frequently attempt to optimise multiple competing objectives, such as precision and recall. Here we introduce a multi-objective evolutionary algorithm framework for evaluating, analysing and improving performance in information access tasks by finding the sets of model parameters that simultaneously optimise both precision and recall. We present information access experiments using the TREC 7 and 8 collections for *ad hoc* query problems and the Cora collection for document classification. In particular, we compare the following term weighting functions: tf, idf, $tf \times idf$, $tf \times idf/ndl$, BM25 with standard parameter values, BM25 with parameters optimised for

*Authors' addresses: The School of Engineering, Computer Science and Mathematics, Harrison Building, North Park Road, Exeter, EX4 4QF. Email addresses: M.J.Fisher@ex.ac.uk, J.E.Fieldsend@ex.ac.uk and R.M.Everson@ex.ac.uk. A preliminary report on this work was given in (Fisher et al., 2004).

the document collection, and a novel neural network term weighting method. We find that the BM25 method with optimised parameter values outperforms all other methods including BM25 with standard values. The neural network method with optimised weights and biases also performs well. We also optimise the range of terms used for document classification and show how, for a particular information access task, to locate the optimum range of terms.

Categories and Subject Descriptors: G.1.6 [**Numerical Analysis**]: Optimization, H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval, H.3.4 [**Information Storage and Retrieval**]: Systems and Software - *Performance evaluation (efficiency and effectiveness).*

General Terms: Algorithms, Experimentation, Performance.

Additional Key Words and Phrases: Multi-objective optimisation, evolutionary algorithms, information retrieval, document classification, term feature selection, term weighting methods.

1 Introduction

In information access tasks, such as document classification (DC) or information retrieval (IR), there are many model parameters and, indeed, models to choose from. For example, when performing document classification we must choose the inputs to the model (which terms to choose, which stop words to remove, the term weighting method and values of associated parameters, etc.) together with the parameters of the model (classifier and values of associated parameters, etc.). In addition to the large number of parameters, in information access tasks, the performance of the model is evaluated according to multiple competing objectives, usually precision and recall. Clearly, it is usually impossible to locate a single combination of parameters which simultaneously optimises several objectives and the curve (for two objectives) or surface (for three or more objectives) that describes the trade-off between objectives is known as the Pareto front.

Unfortunately in information access, although it is known that carefully chosen model parameters result in improved precision and recall performance, many model parameters are chosen simply because the values have been reported to be successful in a previous application. At best a few different values for parameters are evaluated for the model and whichever performs best is used. Examples of this *ad hoc* parameter tuning procedure can be found in, for example, many of the papers published in the TREC proceedings (Harman et al., 2004; MacFarlane et al., 1999; Zhai and Lafferty, 2002). This has probably been the case because the only method to locate optimal parameters, namely exhaustive search, is infeasibly expensive. Also, the most popular search algorithms for complex problems with many parameters, for example simulated annealing (Kirkpatrick et al., 1983), genetic algorithms and evolution strategies (ES) (Fogel, 1994), have traditionally been formulated in terms of a single objective. However, recent advances in the field of evolutionary computation have led to a class of algorithms—multi-objective evolutionary algorithms (MOEAs)—capable of locating the Pareto front; see, for example, (Coello, 1999; Deb, 2001; Fonseca and Fleming, 1995; Veldhuizen and Lamont, 2000).

In this paper we present a methodology for evaluating, analysing and improving performance for information access tasks. We investigate the use of an MOEA to estimate optimal values for model parameters in information access tasks. In particular we show, for both information retrieval and document classification tasks, how to optimise the precision and recall for several popular term weighting methods with respect to their adjustable parameters and the index terms. This provides a methodology for selection of optimal parameters for particular tasks and elucidates the trade-offs between competing objectives. In addition, it provides insight into the nature of the models.

In Section 2, *information access tasks*, the similarities and differences between IR and DC are discussed and we develop a common framework in which precision and recall for either task can be optimised. In this section we also describe and discuss term weighting and selection methods, the parameters of which are subsequently optimised. In Section 3, *optimality and MOEAs*, we discuss the meaning of Pareto optimality, how to compare Pareto fronts and the algorithm used for the experiments. Section 4 describes experiments on document classification and information retrieval used to illustrate the methodology and gives the results. Conclusions are drawn and areas for further work are discussed in Section 5.

2 Information Access Tasks

The vector space model underlies many information access endeavours (Baeza-Yates and Ribeiro-Neto, 1999; Salton, 1971) and we briefly review it to emphasise the similarities between *ad hoc* information retrieval and text classification. In common with the majority of information access methods, the vector space model ignores the order of terms within a document and describes a document d_j in a document collection of J documents $\mathcal{D} = \{d_1, ..., d_J\}$ as a bag of I words or terms, $t_i \in \mathcal{T} = \{t_1, ..., t_I\}$. Each document is represented by a vector of term weights $w_{i,j}$, thus $d_j = (w_{1,j}, \ldots, w_{i,j}, \ldots, w_{I,j})^T$. Using all terms within the collection to represent documents degrades performance by introducing noise, whereas discriminative power is lost if too few terms are used: Section 2.3 discusses current methods for choosing the index terms for a model. Many schemes have been proposed for assigning term weights and we defer discussion of these to Section 2.4. Both IR and DC commonly rely on the proximity of documents in the vector space representation, which we now review.

2.1 Information retrieval

In IR the goal is to retrieve those documents from $\mathcal{D}$ which are most relevant to a query. We represent the query by a document d_q, with weights $w_{i,q}$ which are non-zero only if the term t_i is in the query.

Documents close to d_q are judged more relevant than distant documents and we measure the distance between documents by the cosine similarity measure:

$$sim(d_q, d_j) = \frac{d_q \cdot d_j}{|d_q||d_j|} = \frac{\sum_{i=1}^{I} w_{i,q} w_{i,j}}{\sqrt{\sum_{i=1}^{I} w_{i,q}^2}\sqrt{\sum_{i=1}^{I} w_{i,j}^2}} \tag{1}$$

which measures the cosine of the angle between d_j and d_q.

The lengths $|d_q|$ and $|d_j|$ appearing in the denominator of (1) have the effect of, at least partially, normalising the distance between documents with respect to document length, and many term weighting schemes incorporate additional explicit document normalisation in the assignment of term weights. The popular BM25 weighting scheme (Robertson et al., 1994) uses more sophisticated document length normalisation in its assignment of term weights and hence when using the BM25 method we measure distances between documents by the unnormalised inner product:

$$sim(d_q, d_j) = d_q \cdot d_j = \sum_{i}^{I} w_{i,q} w_{i,j} \tag{2}$$

2.2 Classification

In document classification each document d_j in the document collection is associated with one of M classes $c_m \in \{c_1, ..., c_M\}$. The goal of a classifier is to determine the probability that a query or test document d_q in the document collection belongs to each class c_m. A straightforward and commonly used classification method, on which we focus here, is the K-nearest neighbours (K-nn) classifier, in which d_q is assigned to the class with the majority of neighbours in a sphere centred on d_q containing K neighbours. The K-nn classifier can be formulated to provide

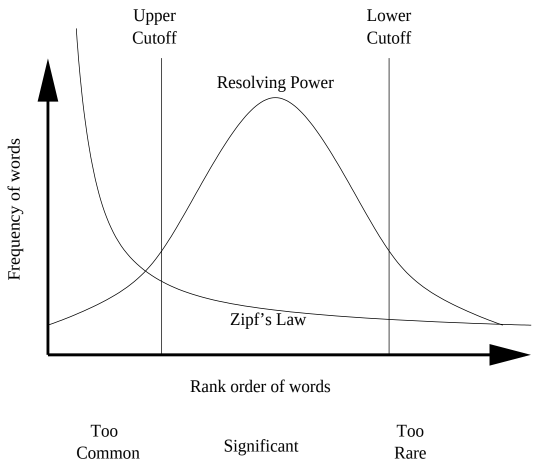

Figure 1: The resolving power of words (adapted from (Schultz, 1968)).

probabilities of class membership (Fukunaga, 1990): if K_m is the number of the K nearest neighbours of d_q in class c_m, then the probability of d_q belonging to class c_m is simply estimated as $P(c_m|d_q) = K_m/K$. K-nn classifiers can be shown to have an asymptotic error rate no worse than twice the optimal Bayes error (Fukunaga, 1990). In an identical way to IR the K-nearest neighbours are determined by distances derived from the cosine similarity (1) and (2), although more sophisticated metrics may be derived for alternative models; for example (Hofmann, 2000).

2.3 Choosing terms

In any information access task the terms or features to be used in the model must be selected. Until recently, due to a lack of computer processing power and memory, only a subset of the terms occurring in the document collection could be used. Fortunately, in accord with Luhn's predictions (Luhn, 1958), it has since been discovered that using all terms in a model decreases the performance, especially in tasks such as DC because the rare terms introduce noise and common terms are not discriminatory, while the middle range have greater resolving power, as shown in figure 1.

Unfortunately the cut-off points defining the useful range are not easy to determine for each document collection. Furthermore, the useful range will depend on the nature of typical queries. Normally, these upper and lower cut-off points are estimated by removing terms that only occur a few times throughout the collection (Baeza-Yates and Ribeiro-Neto, 1999) and also removing the most common terms in a document collection, called stopwords.

There are two standard stopword removal methods. The most popular method is to discard words occurring in a standard stopword list (see, for example, van Rijsbergen, 1975). Alternatively, statistical methods can be used to discover and remove those words that occur in more than approximately 80% of the documents, based on the empirical observation that words occurring more frequently do not generally improve IR performance.

An advantageous bi-product of stopword removal is that the size of the index is decreased so searching is faster and storage requirements reduced. However, there are also disadvantages. Consider the query 'to be or not to be': using most stopword lists, this query would become simply 'not' which clearly is unsatisfactory. In fact, some stopword lists include the word 'not', which results in all query terms being eliminated. However, the benefits of stopword removal, such as up to 40% reduction in index size and the removal of non-discriminatory terms have led to the widespread removal of stopwords.

There are, however, more principled methods than the 80% rule of thumb for selecting the most useful term features. Yang and Pedersen (1997) present a comparison of five of these methods for DC, which we briefly review here.

- The first and most simple method is document frequency thresholding for determining the lower cut-off point. Only the terms which occur in more than a predefined number of documents are used in the model; it is assumed that very rare terms are noise. This method is widely employed since it is simple and can be used in any information access task, but discriminating terms may be missed if they occur less frequently than many non-discriminating terms.

- The second method, information gain, measures the average number of bits of information obtained for class prediction by using the presence or absence of a term in a document. The information gain h_i of a term t_i is defined as:

$$h_i = -\sum_{c=1}^{C} p(c)\log p(c) + p(t_i)\sum_{c=1}^{C} p(c|t_i)\log p(c|t_i) + p(\neg t_i)\sum_{c=1}^{C} p(c|\neg t_i)\log p(c|\neg t_i) \quad (3)$$

  where there are C classes. The terms which have a information gain higher than a predefined threshold are used in the model.

- Mutual information provides a third method. The mutual information between term t_i and class c is $I(t_i, c) = \log \frac{p(t_i,c)}{p(t_i)p(c)}$. Clearly, when the occurrence of terms in documents is independent of the class $p(t_i, c) = p(t_i)p(c)$ so that the mutual information is zero. The information a term provides can be calculated in two ways: $I(t_i) = \sum_{c=1}^{C} p(c)I(t_i, c)$ and $I(t_i) = \mathtt{max}_{c=1}^{C} I(t_i, c)$. The terms providing the highest information are used in the model.

- The fourth method uses the well known χ^2 statistic which, like mutual information, measures the lack of independence between term t_i and class c; $\chi^2(t_i, c) = \frac{J(AD-CB)^2}{(A+C)(B+D)(A+B)(C+D)}$ where A is the number of times t_i and c co-occur, B is the number of times t_i occurs without c, C is the number of times c occurs without t_i and D is the number of times neither t_i nor c occurs. As with the mutual information method, an overall term score can be calculated in two ways: by summing over the probability of classes multiplied by the χ^2 score for the term and classes and by taking the maximum χ^2 score for any class. The terms with the highest scores are used in the model.

- The final method is called the term strength method. Documents in a collection are clustered and the cosine similarity measure is used to discover related documents, namely those with a similarity score exceeding a threshold. The term strength is then based on how commonly a term occurs in related documents. As usual only those terms with the highest term strengths are used in the model.

Only the document frequency and term strength methods can be used in all information access tasks since they do not require class information, although the term strength method is constructed from pseudo-classes derived using the clusters. Yang and Pedersen (1997) found that when selecting an equal number of terms to use in the model the document frequency, χ^2 and information gain methods outperformed the mutual information and term strength methods. They also found that the vocabulary size could be reduced by 98% without reducing the classification accuracy.

Here, rather than selecting individual terms, we adopt Luhn's point of view and seek to discard terms that occur more frequently than some upper cut-off, u, and less frequently than a lower cut-off, l. However, unlike previous work we regard the upper and lower cut-offs as parameters to be automatically optimised on a training set to produce the best precision and recall performance.

2.4 Term weighting methods

Many term weighting methods have been introduced, see (Harman, 1992; Robertson and Sparck Jones, 1997) for examples, but here we study just five commonly used term weighting methods and a novel neural network method as follows.

Term frequency, $tf_{i,j}$: Luhn (1961) observed that: *repetition is an indication of emphasis.* Frequently used terms in a document are important for characterising that document. Term frequency, the simplest term weighting method other than binary weighting, assigns weights equal to the number of times, $tf_{i,j}$, that term t_i occurs in document d_j:

$$w_{i,j} = tf_{i,j} \tag{4}$$

Inverse document frequency, idf_i: Sparck Jones (1972) noticed that rather than merely the number of times that a term occurs within a document, what is central for IR is the discriminatory power of a term within a corpus. A term that occurs in very few documents within the collection is likely to be important for representing those documents in which it occurs. Inverse document frequency weighting therefore weights a term inversely proportional to the logarithm of the number of documents in which it occurs:

$$w_{i,j} = idf_i = \log \frac{J}{df_i} \tag{5}$$

where df_i is the number of documents in the collection containing the term t_i. The constant of proportionality is J, the number of documents in the collection, but other normalising constants (such as the maximum document frequency) are sometimes used (Robertson and Jones, 1976; Sparck Jones, 1979a,b)

TFIDF $tf \times idf$: The well known $tf \times idf$ measure (Salton and Yang, 1973; Sparck Jones, 1972) simply combines the tf and the idf weights:

$$w_{i,j} = tf_{i,j} \times \log \frac{J}{df_i} \tag{6}$$

A term t_i that occurs frequently in document d_j but infrequently throughout the document collection is thought to be highly discriminatory for d_j and is thus highly weighted.

TFIDF normalised by document length $tf \times idf/ndl$: The TFIDF weight is often normalised by document length to prevent longer documents obtaining higher weights simply because they are verbose. Although more sophisticated document length normalisation methods are available (e.g., (Singhal et al., 1996)), we study the commonly used scheme which merely divides the $tf \times idf$ weights by the normalised document length. The resulting weight is:

$$w_{i,j} = \frac{tf_{i,j}}{ndl_j} \log \frac{J}{df_i} \tag{7}$$

where the normalised document length $ndl_j = dl_j / \left\{ \frac{1}{J} \sum_{j=1}^{J} dl_j \right\}$ is the length dl_j of document J normalised by the average document length.

BM25: Harter (1975) introduced a two Poisson model to account for the difference in the probability distributions of significant or elite words in a document collection and uninformative words. Although the 2-Poisson model involves several parameters which are difficult to estimate for a particular corpus, Robertson and Walker (1994) argue that by extracting details of the shape of the 2-Poisson model function, it can usefully replaced by simpler functions. Robertson and Walker suggested two replacement term weighting functions, BM11 and BM15, both of which proved successful in IR experiments. In (Robertson et al., 1994) a term weighting function called BM25 is suggested which combines the BM11 and BM15 functions. Interestingly, although derived from the 2-Poisson model, the BM25 weighing scheme can be seen as a variant of $tf \times idf$. BM25 is widely used, having proved successful in the TREC workshops[1] (Harman et al., 2004; Voorhees, 2001, 2002, 2003) and is the default weighting in the popular OKAPI system (Robertson et al., 1994).

The BM25 weighting (Robertson et al., 1994) incorporates nonlinear document length normalisation (by normalising with a measure of document length that increases more slowly than raw document length) and term frequency weights:

$$w_{i,j} = \frac{tf_{i,j} \times (k+1)}{k \times ((1-b) + (b \times ndl_j)) + tf_{i,j}} \times \frac{J}{df_i} \tag{8}$$

[1] TREC: `http://www.trec.nist.gov/`

Values for the b and k parameters are chosen according to the qualities of the document collection. The b parameter controls the degree of document length normalisation: $b = 1$ assumes verbose documents and $b = 0$ assumes multi-topic documents. The k term determines the sensitivity of $w_{i,j}$ to changes in the term frequency. In this work we compare the performance of BM25 with optimised k and b values to two commonly used sets of values. The first set, $k = 1.2$ and $b = 0.75$, are the OKAPI (Robertson et al., 1994) default weights, and the second, $k = 2$ and $b = 0.75$, are those suggested by Robertson and Sparck Jones (1997).

Neural networks: The term weighting methods above all combine document length, term frequency and document frequency to arrive at the term weight. Although a great deal of work has been devoted to deriving suitable term weighting functions, the ideal mapping for a particular corpus and set of queries is unknown. Here we adopt an empirical approach and use a neural network as a flexible nonlinear function approximator to map document length, term frequency and document frequency to a term weight dependent on the values of a vector of parameters or weights, $\boldsymbol{\theta}$:

$$w_{i,j} = g(\bar{dl}_j, \bar{tf}_{i,j}, \bar{df}_i; \boldsymbol{\theta}) \tag{9}$$

Here $\bar{dl}_j$ denotes the document length normalised so that the mean and variance of $\bar{dl}_j$ for the collection are 0 and 1 respectively; and likewise for $\bar{tf}_{i,j}$ and $\bar{df}_i$. Normalising the inputs to a neural network is common practise because it decreases training time. In this study we use a simple multi-layer perceptron network (MLP) (see, for example, Bishop, 1995) with a single hidden layer of three nonlinear hidden units and a linear output unit. Instead of using traditional MLP learning methods, such as gradient descent and back propagation, we use the multi-objective optimisation framework to learn the 16 weights and biases by training on the document collection.

2.5 Evaluation

In order to make connections between the DC and IR paradigms (Lewis, 1991), we focus on two-class classification problems, calling one class the *relevant* class and designating the other class as the *irrelevant* class. Results of a two-class document classification problem may be summarised by a confusion matrix:

		True class	
		relevant	irrelevant
Classified	relevant	TP	FP
class	irrelevant	FN	TN

Here TP (true positives) is the number of documents correctly classified as belonging to the *relevant* class, FP (false positives) is the number of documents incorrectly classified as belonging to the *relevant* class, FN (false negatives) denotes the number of documents belonging to the relevant class but not classified as such and finally TN (true negatives) is the number of documents correctly classified as not belonging to the relevant class. The overall classification rate thus is $(TP + TN)/J$.

In IR problems documents are assigned a weight or probability measuring the degree to which d_j belongs to the *relevant to the query* class. These weights are used to produce a ranked list of the documents within the collection. Returning the entire document collection as a ranked list to the user would be unreasonable so a cut-off point is chosen after the first n documents which are returned to the user. In a similar manner to the DC paradigm a confusion matrix may be calculated from these first n documents. An important difference, however, between IR and DC is that in document classification exemplars belonging to known classes are readily available for training, whereas the relevant and irrelevant classes in information retrieval are only defined in relation to each new query and usually the only known relevant exemplar is the single *document* formed by the query itself.

Precision, $\mathcal{P}$, and recall, $\mathcal{R}$, are usually used to evaluate information access tasks and can be calculated from the confusion matrix:

$$\mathcal{P} = \frac{TP}{TP + FP} \qquad \mathcal{R} = \frac{TP}{TP + FN} \tag{10}$$

Precision is the fraction of the documents classified as relevant that actually are relevant, while recall shows how many of all possible documents belonging to the relevant class have been classified as such.

High precision is important in situations such as information retrieval on the WWW, where there is such a vast pool of documents that it is more important to find the most relevant documents rather than all of the relevant documents. High recall is required when all the available information about a subject is required; for example, a patent search.

It is often found that it is difficult to obtain good values for both recall and precision, because to achieve better recall value a system can retrieve more documents, which in turn lowers the precision and vice versa.

If a fixed number, N, of documents are always returned note that $\mathcal{P}$ and $\mathcal{R}$ increase or decrease together, because the denominators in Equation (10) are both constant; $TP + FP = N$ and $TP + FN$ is the number of relevant documents in the collection.

There are numerous other metrics for evaluating the performance of both document classification and information retrieval systems, most of which can be described in terms of the confusion matrix (Fawcett, 2004). Since the similarities between information retrieval and document classification were explicitly recognised only relatively recently (Lewis, 1991), metrics for information retrieval and document classification have been developed separately. Unsurprisingly, there is much overlap in the commonly used metrics of the two fields and all metrics can be applied to both tasks, although with varying degrees of utility.

A metric that is used in both fields is false positive rate (FPR), also known as fallout in information retrieval, $FPR = FP/(FP+TN)$. Notice the symmetry between FPR and recall – in classification circles recall is known as the true positive rate (TPR) or sensitivity. The research area of ROC analysis is concerned with optimising the competing objectives FPR and TPR in a similar manner to the optimisation of precision and recall (Fisher et al., 2004).

Other commonly used metrics, especially for document classification, include accuracy or classification rate and the specificity defined as $TN/(FP+TN) = 1-FPR$. Notice that, unlike precision, the measures commonly used in document classification do not take the threshold used in to consideration, because generally all documents are classified. In, information retrieval, however, the threshold or the number of documents returned to the user is important and this is reflected in the popular metrics (Lewis, 1997), which quote the precision or recall at a particular n, for example $\mathcal{P}$ at $\mathcal{R} = 0.1$, $FPR = 0.001$ and $\mathcal{P}$ at n documents retrieved, where a standard $n = 20$. Notice that these measures all enforce a threshold in order to evaluate the system.

As discussed at greater length in Section 3, comparison of systems attempting to meeting multiple objectives is difficult because it is not clear when one system is outperforming another. For example: how should two systems, the first of which has superior recall, the other having superior precision be compared? In an attempt to overcome this problem composite measures have been defined to combine the precision and recall objectives into one objective. One of the most popular composite measures is the F-measure (Jardine and van Rijsbergen, 1971):

$$F_\beta = \frac{\mathcal{P}\mathcal{R}}{(1-\beta)\mathcal{P} + \beta\mathcal{R}} \tag{11}$$

If $\beta = 0.5$ the F-measure is the harmonic mean of precision and recall. However, the variable β forces the system designer to give an *a priori* weight to the relative importance of precision and recall when, in fact, the relative importance is unlikely to be known in advance.

An alternative composite metric arising from decision theory, the utility (Lewis, 1997), can be used to evaluate performance:

$$u = \alpha TP + \beta FP \tag{12}$$

where α is the value associated with retrieving a relevant document and β is the cost of retrieving an irrelevant document. Here again the system designer must make an *a priori* judgement about the costs α and β, which are usually difficult to estimate in advance.

Instead of using a single objective, such as the F-measure, the goal of this paper is to show that multi-objective optimisation may be used to allow the system designer to investigate the properties of models on the optimal trade-off curve – the Pareto front – before deciding on the importance of each objective. We use MOEAs to find the term weighting method, the values of term weighting method parameters and threshold n that optimise the objectives $\mathcal{P}$ and $\mathcal{R}$ for IR and DC tasks. In addition, we find the lower l and upper u cut-off points for the index terms and the number of nearest neighbours K that optimise $\mathcal{P}$ and $\mathcal{R}$ for document classification. It should be emphasised that we use relatively naïve methods for the information access tasks and are not attempting to outperform the more sophisticated models, but rather to demonstrate the ability of the multi-objective optimisation methodology to allow the system designer to analyse, understand and improve the performance of their model.

3 Optimality and MOEAs

Many situations involve the trading-off of one objective against one or more other objectives. For example, the manufacturer of a product may want to minimise cost and maximise performance. An approach to solving these types of problems is to employ multi-objective evolutionary algorithms (MOEAs), which return a set of solutions to the problem describing the trade-off front by using evolutionary computation techniques.

The curve or surface that defines the optimal trade-off possibilities between objectives to be optimised (precision and recall here) is known as the Pareto front. By definition, a solution (an instance of a type of model) operating on the Pareto front cannot improve any objective without degrading at least one of the others. Two solutions neither of which wholly dominates the other are called *non-dominated* solutions, and the Pareto front is comprised of the non-dominated solutions that are not dominated by any other feasible solution. The purpose of multi-objective algorithms is therefore to locate the Pareto front of these *non-dominated* solutions.

We mention two of the advantages of using a multiple-objective optimisation technique over the traditional technique of aggregating the multiple objectives into a single objectives. First, the combination of several objectives into a single measure requires relative weightings between the objectives to be set, for example, the F-measure (11) and the utility (12), require the system designer assign a weight to each objective *a priori*. Secondly, it can be shown that once these relative weighting have been set the full trade-off surface is usually inaccessible to a uni-objective optimiser (Das and Dennis, 1997). Multiple objective evolutionary techniques, on the other hand, have been found to be robust and permit a full exploration of the trade-off surface; see (Coello, 1999; Deb, 2001; Fonseca and Fleming, 1995; Veldhuizen and Lamont, 2000) for reviews.

We first formally define the multi-objective optimisation problem and ideas of dominance before describing the MOEA used in here.

3.1 Pareto optimality

We seek to simultaneously maximise or minimise D objectives, y_i, which are functions, $f_i(\boldsymbol{\theta})$ of a vector of P variable parameters, or decision variables, $\boldsymbol{\theta}$:

$$y_i = f_i(\boldsymbol{\theta}), \qquad i = 1, \ldots, D \tag{13}$$

Here the objectives will be precision, $\mathcal{P}(\boldsymbol{\theta})$, and recall, $\mathcal{R}(\boldsymbol{\theta})$, so $D = 2$. The parameters are the parameters of the term weighting model, the range $[l, u]$ of index terms used and the number n

of documents returned, which is always the final element in the parameter vector. The precise parameter set depends upon which of the different models is being optimised; for example, 3 parameters $\boldsymbol{\theta} = (k, b, n)$, in the case of IR with BM25 term weighting and 6 parameters $\boldsymbol{\theta} = (k, b, K, l, u, n)$ for DC using a K nearest neighbours classifier and BM25 term weighting.

Without loss of generality we assume that the objectives are to be maximised, so that the multi-objective optimisation problem may be expressed as:

$$\text{Maximise} \quad \mathbf{y} = \mathbf{f}(\boldsymbol{\theta}) = (f_1(\boldsymbol{\theta}), \ldots, f_D(\boldsymbol{\theta})) \tag{14}$$

where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_P)$ and $\mathbf{y} = (y_1, \ldots, y_D)$.

In contrast to the uni-objective situation, when there are two or more competing objectives to be optimised, it is clear that solutions exist for which performance on one objective cannot be improved without reducing performance on at least one other. Such solutions are said to be *Pareto optimal* (Veldhuizen and Lamont, 2000) and the set of all Pareto optimal solutions are said to form the Pareto set, $\mathcal{E}$. The notion of *dominance* can be used to make Pareto optimality clearer. A decision vector $\boldsymbol{\theta}$ is said to *strictly dominate* another $\boldsymbol{\phi}$ (denoted $\boldsymbol{\theta} \succ \boldsymbol{\phi}$) iff

$$\begin{gathered} f_i(\boldsymbol{\theta}) \geq f_i(\boldsymbol{\phi}) \quad \forall i = 1, \ldots, D \quad \text{and} \\ f_i(\boldsymbol{\theta}) > f_i(\boldsymbol{\phi}) \quad \text{for some } i. \end{gathered} \tag{15}$$

Less stringently, $\boldsymbol{\theta}$ *weakly dominates* $\boldsymbol{\phi}$ (denoted $\boldsymbol{\theta} \succeq \boldsymbol{\phi}$) iff

$$f_i(\boldsymbol{\theta}) \geq f_i(\boldsymbol{\phi}) \quad \forall i = 1, \ldots, D. \tag{16}$$

A set of decision vectors F is said to be a *non-dominated set* (an estimate of the Pareto front) if no member of the set is dominated by any other member:

$$F_k \not\succ F_j \quad \forall k, j = 1, \ldots, |F|. \tag{17}$$

In multi-objective optimisation problems we generally want the estimated Pareto front to be: *accurate* – close to the true Pareto front, *representative* – cover most of the true Pareto front and *well distributed* – solutions are well distributed across the true Pareto front.

3.2 The optimisation algorithm

Evolutionary computation has the ability to search for global solutions in high dimensional parameter space which may exhibit complex interactions in their relationship to the evaluation of the parameter space. There are two main approaches to evolutionary computation: evolution strategies (ESs) and genetic algorithms (GAs). In this paper, we use ES techniques.

ES emulates the biological evolutionary process of adjustment and selection. A problem is represented by a floating point vector with P adjustable parameters $\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_P)$. These *decision vectors* are perturbed at each iteration before their fitness is evaluated by calculating the effect of the new parameters on the process being modelled. Perturbation at each iteration, or generation, produces new decision vectors or solutions. A $(\mu + \lambda)$-ES process has μ decision vectors (or parents) available at the start of each generation, each parent is perturbed to generate λ offspring. The set of offspring and parents are then truncated or replicated to provide the μ parents for the following generation.

The MOEA used in this study is a based on a simple (1+1)-ES, similar to PAES (Knowles and Corne, 1999). In outline, the procedure for locating the Pareto front, $\mathcal{E}$ operates by maintaining an archive, F, of mutually non-dominating solutions, $\boldsymbol{\theta}$, which is the current approximation to the Pareto front. As shown in lines 4 and 5 of Algorithm 1, at each stage of the algorithm a single solution in F is copied and its parameters perturbed. Evaluating the precision and recall of the perturbed solution (line 6) at various numbers of documents returned n yields many new solutions, the set of which we denote in Algorithm 1 by M. Those perturbed solutions that are dominated by members of F are discarded, while the others are added to F (line 10) and any dominated solutions in F removed (line 10). In this way the estimated Pareto front F advances towards $\mathcal{E}$. This algorithm, unlike earlier versions (Knowles and Corne, 1999), maintains an archive which is unconstrained in size, permitting better convergence (Fieldsend et al., 2003).

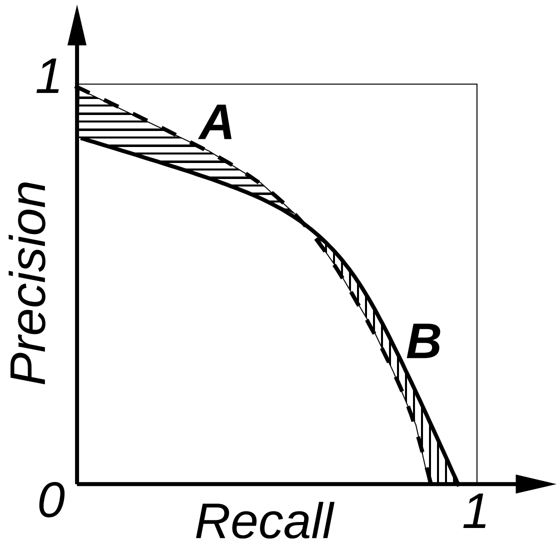

Figure 2: Comparison of Pareto fronts, A (dashed) and B (solid). Solutions in the horizontally hatched region (of area $\mathcal{V}(A, B)$) are dominated by A, but not by B, whereas solutions in the vertically hatched region (of area $\mathcal{V}(B, A)$) are dominated by B but not A.

It should be emphasised that we regard the number, n, of documents returned in response to a query as the final parameter θ_P to be optimised. Thus only the first $P-1$ parameters are perturbed, since it is computationally cheap to evaluate the precision and recall for a wide range of n. All that is needed is to evaluate the precision and recall for the most probable document returned, the second most probable document returned, *etc.* For the IR tasks we evaluate the precision and recall for thresholds up to $n = 5000$, beyond which the precision and recall change only very slowly with n; for the classification task $n = 1082$ (the number of documents in the test set). The optimiser therefore acts as an $(1+\lambda)-$ES, where $\lambda = n$ for IR and $\lambda = Kn$ for classifiers (an additional factor of $K = 50$ models are evaluated, one for each number of nearest neighbours). This is emphasised in line 6 of Algorithm 1, where the `evaluate` method is passed both the decision vector $\boldsymbol{\theta}$, and n, in order to generate the set of children M.

3.3 Comparison

The comparison of estimated Pareto fronts (and therefore information access models) is difficult as there are several ways in which a front or set of points could be judged as being be inferior or superior to another (Knowles and Corne, 2002). For example in Figure 2 the front B is superior when objective 1 is large, whereas A is superior when objective 1 is small. Since precision and recall lie in the range $[0, 1]$, we compare fronts A and B by measuring $\mathcal{V}(A, B)$, the volume of the unit square in objective space that is dominated by A and not by B (Fieldsend et al., 2003). Note that $\mathcal{V}(A, B) \neq \mathcal{V}(B, A)$. This measure is conceptually similar to the performance measured used by Laumanns *et al.* (Laumanns et al., 2000). The $\mathcal{V}$ measure may be estimated by Monte Carlo sampling of the unit square and counting the fraction of samples that are dominated exclusively by A or B; see (Fieldsend et al., 2003) for details.

4 Experiments

4.1 Information retrieval

For the IR task we use the TREC 7 and 8 *ad hoc* document collections (disks 4 and 5), topics (351 - 450) and relevance judgements[2]. The document collection consists of approximately 55,000 documents from the Federal Register (1994), 210,000 documents from the Financial Times (1992-1994), 130,000 documents from the Foreign Broadcast Information Service and 130,000 documents from the Los Angeles Times (1989-1990). All terms (no stop word removal) contained within the documents are used as index terms.

The topics (queries) were randomly split into a 75% training part and a 25% test part. This was performed 10 times to give 10 different folds to compare when evaluating the results. Results on the test data are averaged over all of the queries in the test set and results on the training

[2]TREC: `http://www.trec.nist.gov/`

data are averaged over all the queries in the training set.

We use the term weighting methods described in Section 2.4, but we do not attempt to optimise the upper and lower cut-off points for terms since we have only 99 queries and would almost certainly overfit the data. Term weighting methods tf, idf, $tf \times idf$ and $(tf \times idf)/ndl$ have no associated variables so they are used simply for comparison. Method BM25 has two variables b and k, commonly used values for these are $b = 0.75$ & $k = 2$ (Robertson and Sparck Jones, 1997) and $b = 0.75$ & $k = 1.2$ (Robertson et al., 1994). We denote BM25 term weighting with these traditional, fixed values as BM25(0.75, 2.0) and BM25(0.75, 1.2); we also evaluate the BM25 method in which b and k are adjusted as part of the optimisation process, these results are denoted BM25*. Finally, we show results using the neural network term weighting, denoted MLP, where the weights and biases producing optimal $\mathcal{P}$ and $\mathcal{R}$ values are found using the MOEA.

There are short and long query types for each of the 99 topics, both types representing the same information need; for example, topic number 351 has the short query 'Falkland petroleum exploration' and the long query 'What information is available on petroleum exploration in the South Atlantic near the Falkland Islands?' We compare the performance of the different term weighting methods on both query types.

Optimisations for all term weighting methods were run for 1000 generations, after which there was little change in the estimated Pareto front, except for the neural network term weighting method for which we used 2000 generations.

4.1.1 Results

Figure 3 shows an example of the the fronts found for the various term weighting methods using the first fold of the TREC data with short queries. Pareto fronts for each term weighting method on the training data are shown in the top figure and the evaluation of the various model sets, optimised with respect to the training data, on the test data are shown in the bottom figure. At high recall most of the term weighting methods perform in a similar fashion, however the benefit of the MOEA approach is visually manifest in areas of high precision. Both the optimised BM25* method and the MLP method outperform the other methods dramatically. This result has interesting repercussions for information retrieval on the Web where high precision is required but recall is less important: using the methodology presented in this paper for Web information retrieval, where queries are typically composed of only 1 or 2 terms (Xu and Croft, 1996), could result in significant improvements in performance. The colours in this and subsequent figures denote the threshold n for each model on the Pareto front; the colour bars on the right show that colder colours are associated with lower thresholds and warmer colours with higher thresholds. Since the most rapid variation along the front is found at lower thresholds, all models with a threshold of 1000 and above are displayed by the same colour. Unsurprisingly, for all term weighting methods we find that models on the Pareto front with high thresholds, n, (returning many documents) generally have high recall and low precision and models with low thresholds have low recall and high precision. It is interesting to note that although the MLP term weighting has far more flexibility and adjustable parameters than the BM25 model, it is unable to significantly improve performance over BM25 with optimised parameters.

Figure 4 shows the optimal precision/recall curves obtained using the first fold of the TREC data with long queries, Pareto fronts on the training data are shown in the top graph and evaluation on the test data is shown in the bottom figure. The performance of each term weighting method on the long queries is very similar to the performance on the short queries, except for neural networks which do not perform as well. As with the short queries the *idf* term weighting method obtains the lowest precision and recall values followed by *tf* term weighting method. However, although still achieving the second highest precision values, the values obtained along rest of the MLP term weighting method's precision/recall curve only improve upon those found using the *idf* and *tf* methods. The $tf \times idf$ and $(tf \times idf)/ndl$ perform in a similar manner, at high precision values these methods outperform the BM25 method with standard parameter values $k = 2$ and $b = 0.75$. At high recall, however, the BM25(0.75, 2.0) method outperforms the $tf \times idf$ and $(tf \times idf)/ndl$ methods. Notice that for both long and short queries the BM25 method with $k = 1.2$ and $b = 0.75$ (that is, placing less emphasis on term frequency) outperforms BM25(0.75, 2.0) demonstrating the effect on performance of even small changes in parameter values. The BM25* with optimised k and b values substantially outperforms all other methods at every point along the precision/recall curve demonstrating the importance of carefully chosen parameter values. As with short queries, the colours on Figure 4 show the threshold of each model on the Pareto front. Again, for all of the term weighting methods lower thresholds generally result in high precision and higher thresholds result in high recall.

The results of $\mathcal{V}$ measure comparison of the IR term weighting methods of the test data is shown in the form of boxplots in Figure 5. These boxplots show the evaluation of weighting methods over all 10 folds of the test data. Each of the two plots contains seven rows, where each row represents $\mathcal{V}$ for one method with respect to all the other methods; that is the volume of the $[0, 1] \times [0, 1]$ precision-recall square that is dominated by the term weighting method corresponding to the row, compared with each of the other methods. For example the top row of the left plot gives the boxplots, left to right, for $\mathcal{V}(tf, tf)$, $\mathcal{V}(tf, idf)$, $\mathcal{V}(tf, tf \times idf)$, $\mathcal{V}(tf, (tf \times idf)/ndl)$, $\mathcal{V}(tf, BM25)$, $\mathcal{V}(tf, BM25^*)$ and $\mathcal{V}(tf, MLP)$ and shows that, for the short queries, the *tf* method is better than only the *idf* method. On the other hand, again for the short queries, the BM25* row shows that BM25* outperforms all other methods except neural networks.

Interestingly, for both short and long queries, the $tf \times idf$ and $(tf \times idf)/ndl$ methods are similar in their performance, suggesting that the partial document length normalisation in Equation (1) is sufficient.

On the short queries for all folds of the data, the BM25* method with optimised parameters significantly outperforms all other term weighting methods for all values of precision and recall. The MLPs perform better than all term weighting methods other than BM25*. The BM25(0.75, 1.2) method has the best performance of a method with non-optimised parameters. The $tf \times idf$, $tf \times idf/ndl$ and BM25(0.75, 2.0) methods obtain roughly equivalent results outperforming both tf and idf.

The performance of each term weighting method for the long queries on all folds of the data is similar to the performance on the short queries with the exception that the MLP performs less well. The BM25* method with optimised k parameters outperforms all other methods at all precision and recall values. BM25(1.2, 0.75) outperforms all methods except BM25*. Over all of the folds, the $tf \times idf$, $(tf \times idf)/ndl$, BM25 with standard parameters $k = 2$ & $b = 0.75$ and the MLP perform equally well, outperforming the tf and idf term weighting methods.

For both query types, the optimised BM25 method performs significantly better than the BM25 methods with standard parameter values ($k = 2$, $b = 0.75$ and $k = 1.2$, $b = 0.75$), suggesting that significantly enhanced performance can be obtained by optimising the IR model and its parameters for each particular dataset.

Figure 6 shows the optimised k (left) and b (right) values obtained from all folds using the BM25* method plotted against precision, for both short (top) and long (bottom) queries. Colour indicates the thresholds associated with each of the k and b values. These plots demonstrate that k and b follow the same general trends for both short and long queries; however, there are differences especially for b. For short queries, the average value of b is very close to 0 when fewer than about 400 documents are returned, and as $b \to 0$ higher precision is obtained indicating that very little document length normalisation is required. If more documents are to be returned in response to the query (that is, at higher recall) the degree of document normalisation is much less crucial as may be expected because the system should return as many matching documents as possible, with little regard for precision.

For long queries, small b still obtains higher precision but there is a wider spread of b values at low thresholds and high precision, although document length normalisation is more important here than for short queries. In contrast to the short queries, b close to 1 is necessary for high recall when more than about 200 documents are returned.

The plots of k indicate that, although the optimal value of k is insensitive to the precision or the number of documents being returned, the optimal value for this corpus and set of queries is significantly smaller than the values recommended in the literature. Here we find $k \approx 0.5$ is optimal, in contrast to the typically used values of 2.0 (Robertson and Sparck Jones, 1997) and 1.2 (Robertson et al., 1994). This result may be due to the lack of stopword removal in this system, stopwords often have high term frequencies and so a lower sensitivity to term frequency (k) may result in more reliable term weights.

4.2 Classification

For the classification task we used the Cora document collection (McCallum et al., 2000) which was collected by intelligent web spiders. The data set is comprised of approximately 37000 papers, all of which have been semi-automatically classified into hierarchical categories such as /Artificial_Intelligence/Machine_Learning/Theory. In common with others (Cohn and Hofmann, 2001; Fisher and Everson, 2003), we use the 4330 documents in the 7 sub-categories of Machine Learning. Although there are seven document categories, we treat the data as seven two-class DC problems and report precision and recall averaged over the seven problems.

All terms (no stop-word removal) occurring in the title, abstract, author and affiliation are used as index terms. The documents within the collection were randomly split into a 75% training part and a 25% test part 10 times to obtain the 10 folds used in these experiments.

As in the IR task, we calculate the average precision and average recall at different thresholds

n, $1 \leq n \leq N = 1082$; the upper limit being set by the number of documents in the test set. To ensure comparative consistency between the training and test sets the average precision and recall are calculated using every third document on the training set. For the classification task we use a K-nn classifier with $1 \leq K \leq 50$. In summary, for every term weighting parameter combination evaluated we return 54100 average precision and average recall results, 1082 for each K.

As in the IR task, we compare the term weighting methods tf, idf, $tf \times idf$, $(tf \times idf)/ndl$, BM25(0.75, 1.2), BM25(0.75, 2.0) and find the values for the variables, k and b, in the BM25* method and the weights and biases for the MLP method that optimise the average precision and average recall objectives. We also search for the ranges of terms from the Cora document collection that produce the maximum average precision and maximum average recall. There are approximately 16000 terms in all.

4.2.1 Results

Figure 7 shows the fronts obtained on the classification task using the first fold of the Cora document collection. Pareto fronts for the training data are shown in the bottom figure and evaluation on the test data is shown in the top. The colours denote the thresholds of each of the models on the Pareto front. The lower thresholds are found in high precision models and higher thresholds are found in high recall models. In this case all 1082 thresholds are shown and hence colder colours are found until relatively high recall in comparison with the information retrieval figures. As with the information retrieval task, the tf and idf methods are outperformed by all other methods. The performance of the $tf \times idf$ and $(tf \times idf)/ndl$ methods is again very similar, demonstrating that the simple document length normalisation (Equation 1) is sufficient, although there is a relatively small range of document lengths in the Cora collection. As discussed in section 2.5, classification tasks are normally easier than information retrieval tasks in the sense that many exemplars from the few known classes are available for training; this explains why the BM25(0.75, 1.2), BM25(0.75, 2.0) and BM25* methods show similar performance and why the MLP shows only a slight performance improvement. The supervised nature of the document classification task is also the reason for the change in shape of the precision/recall curves, all of the precision/recall curves obtained for the document classification task are closer to the optimal classifier ($\mathcal{P} = 1$ and $\mathcal{R} = 1$).

Figure 8 displays the boxplots resulting from the $\mathcal{V}$ measure comparison of the DC term weighting methods; each plot represents the average over all 10 folds of the test data. The MLP, BM25*, BM25(0.75, 1.2) and BM25(0.75, 2.0) methods all have similar performance over all folds and are superior to $tf \times idf$ and $(tf \times idf)/ndl$. It is unsurprising that document length normalisation is unimportant here because all the documents have similar lengths. Indeed we find that the value of b for optimised BM25* is not crucial (not shown). Finally, the poorest performance is given by the tf and idf schemes, which on average are not significantly different from each other. As found on the IR task the $tf \times idf$ and $(tf \times idf)/ndl$ methods perform similarly. Surprisingly, the idf method outperforms the tf, $tf \times idf$ and $(tf \times idf)/ndl$ on some folds, probably because it emphasises the discriminatory potential of a term which is particularly important for classification. Finally, the tf term weighting method only outperforms the idf method.

Although we used $K = 1, \ldots, 50$ neighbours for the K-nn classifier, it was found that most of the models on the Pareto front had values of K close to 1.

Figure 9 displays histograms of the range of terms used by the different BM25(0.75, 1.2) (dotted), BM25(0.75, 2.0) (dashed) and BM25* (solid) models on the trade-off front. The proportion of models on the Pareto front that used the terms is plotted against the terms ordered by decreasing frequency. The positions of stop-words from a standard stop-word list (about 250 words) are indicated in the ordered list of terms by a dash on the abscissa. Reassuringly, this figure is similar to Figure 1 showing very common and very rare words are ineffective discriminators for classification and that the standard stop words occur predominantly in the unused term ranges. Although it appears that a great many more than the standard 250 words are ineffective for classification, in this specialised collection of machine learning abstracts many of the frequent terms are general terms about machine learning. We observe that (authors') surnames become common at rank 1500-2000, suggesting that an effective classification strategy, for this collection, is to assign a document to the category containing another document with the same author(s) and probably similar words.

5 Conclusion

We have presented a novel methodology for discovering the optimal term ranges and weighting parameters of term weighting methods in information access tasks. This optimality is defined both in terms of precision and recall, leading us to seek an optimal *set* of model parameters which describes the trade-off between these objectives on the training and test data used.

We have demonstrated that, on TREC test collections, multi-objective optimisation can locate parameters for the BM25 model that yield significantly better performance than popular default parameter settings. Even with the simple term weighting methods used in this study, quite significant performance improvements can be obtained through the use of an MOEA optimisation framework. We have also shown that simple MLPs perform very well both on the classification and the information retrieval problem. The performance hierarchy of term weighting methods is very similar for both information retrieval and document classification tasks.

Our experiments show that particular standard parameter values may be not be optimal for particular collections. Indeed, the designer of an information retrieval or document classification system can seldom be sure in advance what balances between accuracy (precision) and coverage (recall) may be available. Multi-objective optimisation algorithms provide a straightforward method of investigating these trade-offs and, importantly, allow the designer to set the operating point without making *a priori* assumptions.

Parameter tuning techniques, whether they are *ad hoc* (for example, plugging in a few values) or sophisticated (for example, using multi-objective optimisation), and model selection are straight-forward to apply to supervised information access tasks, such as DC, because the objectives can be evaluated. However, applying them to the, normally unsupervised, IR task is more difficult because the relevant documents from the document collection, or a representative subset of it, must be discovered given a variety of queries in order to evaluate the performance of various models and parameter sets. There are techniques to make this process easier, for example pooling (Sparck Jones and van Rijsbergen, 1975). Ideally, parameters that optimise precision, recall and possibly other objectives could be learnt automatically with no need for experimental evaluation. Zhai *et al.* (Zhai and Lafferty, 2002) use Bayesian decision theory to automatically find the parameters of two-stage language model that minimise risk using the EM algorithm given the knowledge currently available.

Although including prior information is important and Zhai *et al.* have shown that good performance is obtained using their model, the prior information known is limited. We expect that by using our multi-objective optimisation framework to discover the parameters leading to optimal precision and recall further performance improvements could be made. In addition the trade-off precision/recall front obtained via multi-objective optimisation can be analysed by the system designers before a final model with associated parameter values is chosen.

We emphasise that the experiments presented here used unsophisticated information retrieval models – the vector space model and K-nearest neighbours; we used no query expansion or (pseudo) relevance feedback, for example. More sophisticated term selection or weighting methods could be employed. Nonetheless the MOEA framework is equally applicable to these more sophisticated information access models incorporating more parameters with more complex interactions. The multi-objective algorithms used are readily implemented, although considerable computer time may be needed to fully investigate complicated models.

Acknowledgements

Michelle Fisher is supported by a CASE studentship with BT and the EPSRC. Jonathan Fieldsend is supported by the EPSRC, grant GR/R24357/01.

References

R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

C.M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

C.A.C Coello. A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. *Knowledge and Information Systems. An International Journal*, 1(3):269–308, 1999. URL `citeseer.nj.nec.com/coello98comprehensive.html`.

D. Cohn and T. Hofmann. The missing link – a probabilistic model of document content and hypertext connectivity. In *Advances in Neural Information Processing Systems*, volume 13, pages 430–436, 2001. URL `citeseer.nj.nec.com/cohn01missing.html`.

I. Das and J. Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. *Structural Optimization*, 14(1): 63–69, 1997.

K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley, 2001.

T. Fawcett. ROC Graphs: Notes and Practical Considerations for Researchers. Technical Report HPL-2003-4, HP Laboratories, Palo Alto, CA, USA, 2004.

J.E. Fieldsend, R.M. Everson, and S. Singh. Using Unconstrained Elite Archives for Multi-Objective Optimisation. *IEEE Transactions on Evolutionary Computation*, 7(3):305–323, 2003.

M. J. Fisher and R. M. Everson. When are links useful? experiments in text classification. In *Advances in IR, 25th European Conf. on IR research, ECIR*, pages 41–56, 2003.

M. J. Fisher, J. E. Fieldsend, and R. M. Everson. Precision and recall optimisation for information access tasks. In *Proceedings of ROCAI 2004, part of the 16th European Conference on Artificial Intelligence*, pages 45–54, Valencia, August 2004.

D. Fogel. An introduction to simulated evolutionary optimization. *IEEE Transactions on Neural Networks*, 5(1):3–14, 1994.

C.M. Fonseca and P.J. Fleming. An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation*, 3(1):1–16, 1995.

K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, 1990.

D. Harman. Ranking algorithms. In *Information Retrieval: Data Structures and Algorithms*, pages 363–392. Prentice Hall, 1992.

D. Harman, E. Voorhees, and L. Buckland, editors. *Proceedings of TExt Retrieval Conference (TREC 1-9, 2001-2003)*, 2004. NIST Special Publications. `http://trec.nist.gov/pubs.html`.

S. P. Harter. A probabilistic approach to automatic keyword indexing. *Journal of the American Society for Information Science*, (35):197–206 and 280–289, 1975.

T. Hofmann. Learning the Similarity of Documents: An Information-Geometric Approach to Document Retrieval and Categorization. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 12, pages 914–920, 2000.

N. Jardine and C. J. van Rijsbergen. The use of hierarchic clustering in information retrieval. *Information storage and retrieval*, 7:217–240, 1971.

S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

J. Knowles and D. Corne. The Pareto archived evolution strategy: A new baseline algorithm for Pareto multiobjective optimisation. In *1999 Congress on Evolutionary Computation*, pages 98–105, 1999. URL `citeseer.nj.nec.com/knowles99pareto.html`.

J. Knowles and D. Corne. On Metrics for Comparing Nondominated Sets. In *2002 Congress on Evolutionary Computation*, pages 711–716, 2002.

M. Laumanns, E. Zitzler, and L. Thiele. A Unified Model for Multi-Objective Evolutionary Algorithms with Elitism. In *Proceedings of the 2000 Congress on Evolutionary Computation*, pages 46–53, 2000.

D. Lewis. The TREC-5 Filtering Track. In D. K. Harman and E. M. Voorhees, editors, *The Fifth Text REtrieval Conference (TREC-5)*, pages 75–96. NIST Special Publication 500-238, 1997.

D. D. Lewis. Evaluating Text Categorization. In *Proceedings of Speech and Natural Language Workshop*, pages 312–318. Morgan Kaufmann, 1991.

H.P. Luhn. The creation of literature abstracts. *IBM Journal of Research and Development*, 2: 159–165, 1958.

H.P. Luhn. The automatic derivation of information retrieval encodements from machine-readable texts. In A. Kent, editor, *Information retrieval and machine translation*, volume 3, pages 1021–1028. Interscience, 1961.

A. MacFarlane, S. E. Robertson, and J. A. McCann. PLIERS at TREC-8. In *The Eighth Text REtrieval Conference*, 1999.

A. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval Journal*, 3:127–163, 2000. `www.research.whizbang.com/data`.

S. Robertson and K. Sparck Jones. Simple proven approaches to text retrieval. Technical Report TR356, Cambridge University Computer Laboratory, 1997.

S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In W. B. Croft and C. J. va Rijsbergen, editors, *Proceedings of the 17th Internation Conference on Research and Development in IR*, pages 232–241, 1994.

S.E. Robertson and K. Spark Jones. Relevance weighting of search terms. *Journal of the American Society for Information Sciences*, 1976.

S.E. Robertson, S. Walker, S. Jones, and M. M. Hancock-Beaulieu. Okapi at TREC-3. In *In 3rd Text REtrieval Conference (TREC-3)*, 1994.

G. Salton. The smart retrieval system - experiments in automatic document processing. *Prentice Hall Inc.*, 1971.

G. Salton and C.S. Yang. On the specification of term values in automatic indexing. *J. Documentation*, 1973.

C.K. Schultz. *H.P. Luhn: Pioneer of Information Science - Selected Works.* Macmillan, London, 1968.

A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. In *Proceedings of the 19th ACM Conference on Research and Development in information retrieval (SIGIR'96)*, pages 21–29, 1996.

K. Sparck Jones. A statistical interpretation of term specificity and its applications in retrieval. *Journal of Documentation*, 1972.

K. Sparck Jones. Experiments in relevance weighting of search terms. *nformation Processing and Management*, 15:133–144, 1979a.

K. Sparck Jones. Search term relevance weighting given little relevance information. *Journal of Documentation*, 35:30–48, 1979b.

K. Sparck Jones and C. van Rijsbergen. Report on the need for and provision of an "ideal" information retrieval test collection. Technical Report British Library Research and Development Report 5266, Computer Laboratory: University of Cambridge, 1975.

C.J. van Rijsbergen. *Information Retrieval.* Butterworths, 1975.

D. Van Veldhuizen and G. Lamont. Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art. *Evolutionary Computation*, 8(2):125–147, 2000.

E. M. Voorhees. Overview of TREC 2001. In *The Tenth Text Retrieval Conference (TREC 2001)*, 2001.

E. M. Voorhees. Overview of TREC 2002. In *The Eleventh Text Retrieval Conference (TREC 2002)*, 2002.

E. M. Voorhees. Overview of TREC 2003. In *The Twelfth Text Retrieval Conference (TREC 2003)*, 2003.

J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th Annual International SIGIR Conference on Research & Development in IR*, pages 4–11, 1996. URL `citeseer.nj.nec.com/xu96query.html`.

Y. Yang and J.O. Pedersen. A comparative study on feature selection in text categorization. In *ICML-97, 14th International Conference on Machine Learning*, pages 412–420. Morgan Kaufmann Publishers, 1997.

C. Zhai and J. Lafferty. Two-stage models for information retrieval. In *Proceedings of the 25th Annual International SIGIR Conference on Research & Development in IR*, 2002.

Algorithm 1 A MO $(1+\lambda)-$ES for information access optimisation.

Inputs:

T Maximum number of ES generations

$F :=$ initialise()

$t := 0$

`while` $t < T$:

    $\boldsymbol{\theta} :=$ select(F)

    $\boldsymbol{\theta}' :=$ perturb$(\boldsymbol{\theta})$

    $M :=$ evaluate$(\boldsymbol{\theta}', n)$

    `for all` $\boldsymbol{\psi} \in M$

        `if` $\boldsymbol{\psi} \npreceq \boldsymbol{\phi} \ \forall \boldsymbol{\phi} \in F$:

            $F := \{\boldsymbol{\phi} \in F \,|\, \boldsymbol{\phi} \nprec \boldsymbol{\psi}\}$

            $F := F \cup \boldsymbol{\psi}$

        `end`

    `end`

    $t := t + 1$

`end`

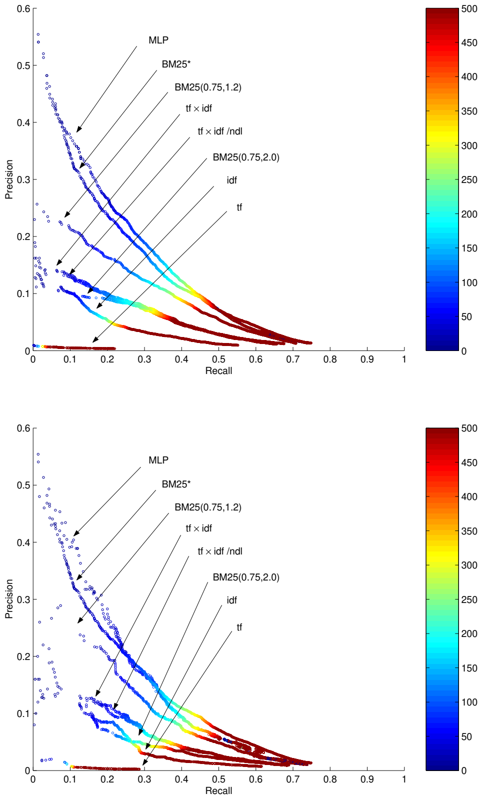

Figure 3: Pareto fronts for the various term weighting methods in the information retrieval task (short query TREC, fold 1.) on the training data (top) and evaluation of the various model sets, optimised with respect to the training data, on the test data (bottom). Colours indicate the threshold n at which the recall/precision performance is achieved.

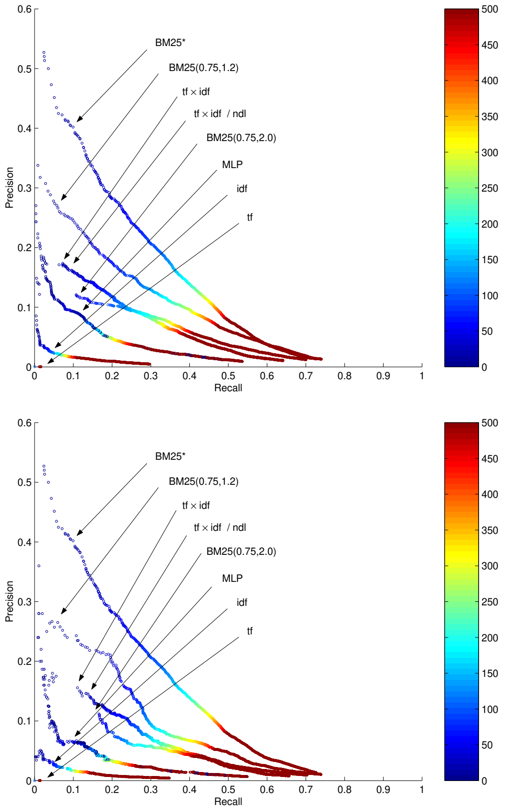

Figure 4: Pareto fronts for the various term weighting methods in the information retrieval task (long query TREC, fold 1) on the training data (top) and evaluation of the various model sets, optimised with respect to the training data, on the test data (bottom).

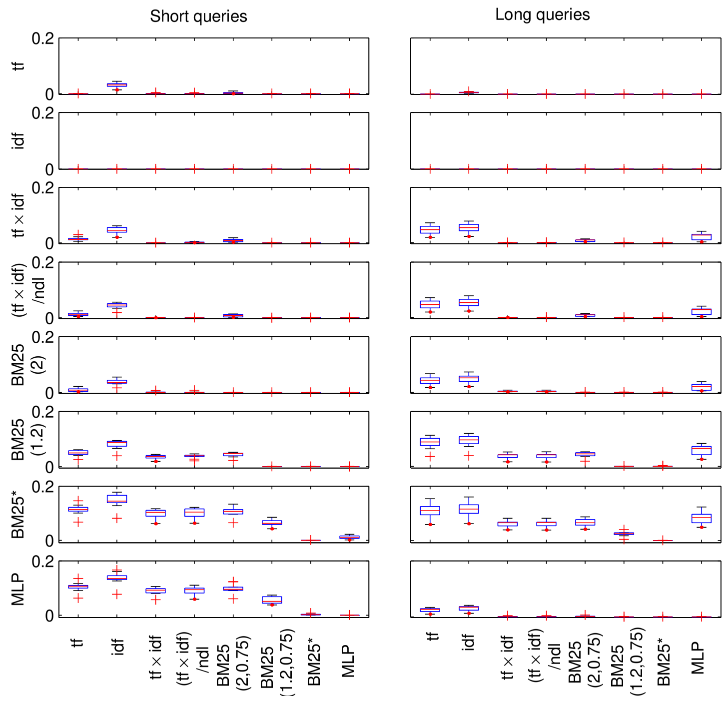

Figure 5: Box-plots of the volume measure comparing the seven different IR term weighting methods using test folds for both short (left) and long (right) queries.

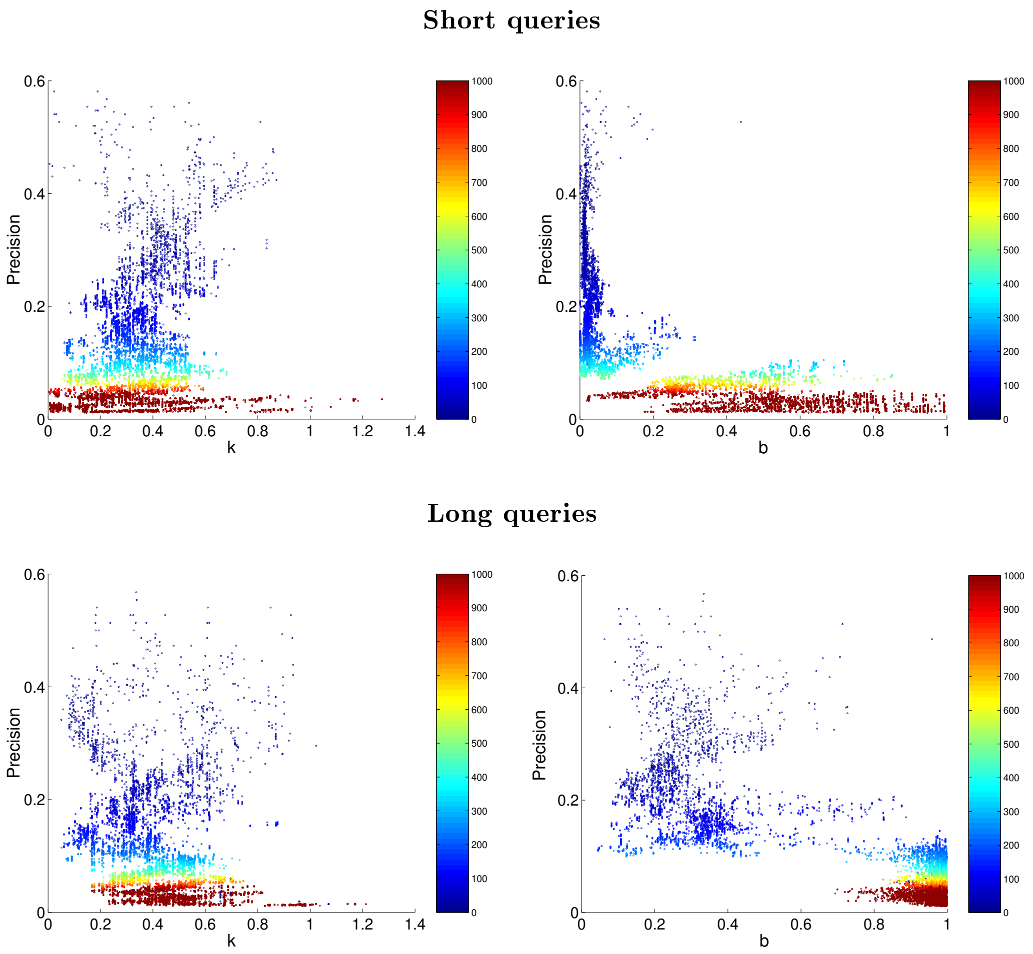

Figure 6: Plots of optimised BM25 parameters k and b versus precision, taken from models on the trade-off front of the BM25* model.

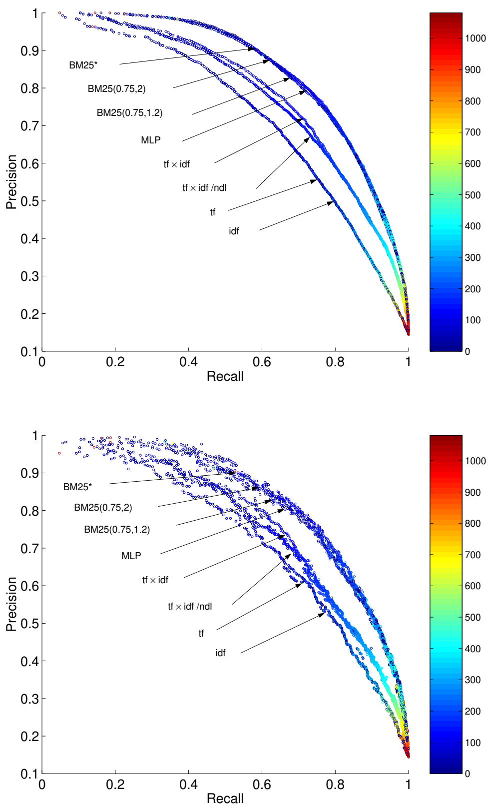

Figure 7: Top: Pareto fronts for the various term weighting methods in the classification task on the training data. Bottom: Evaluation of the various model sets, optimised with respect to the training data, on the test data for the various term weighting methods.

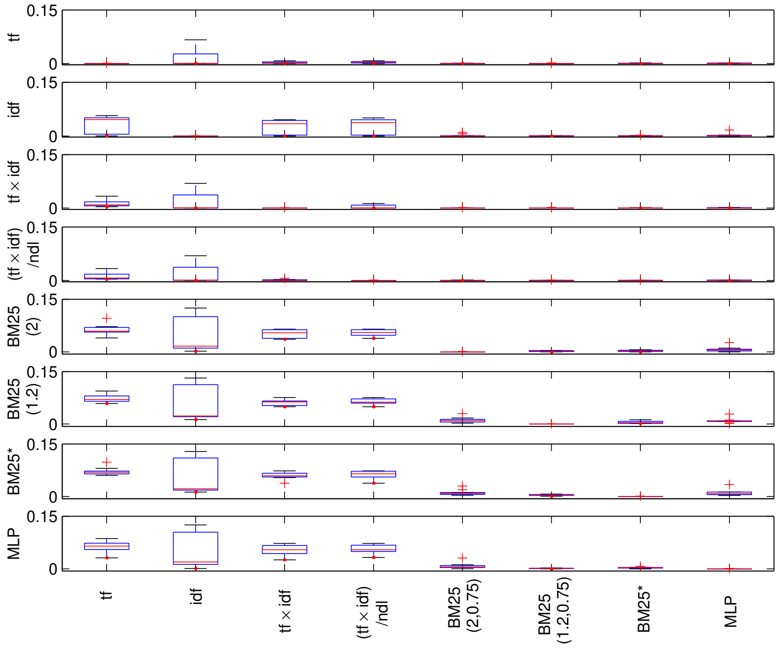

Figure 8: Box-plots of the volume measure comparing the seven different DC term weighting methods calculated using all 10 test folds.

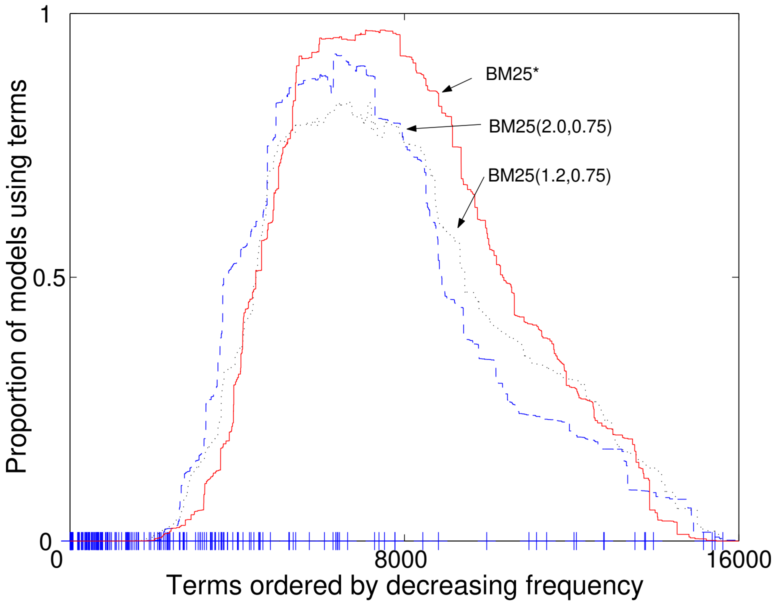

Figure 9: Histogram showing the terms used by the different BM25 models on the trade-off front. The abscissa shows the rank of terms ordered by term frequency, with stop word positions indicated by ticks.