Precision and Recall Optimisation for Information Access Tasks

Michelle J. Fisher **Jonathan E. Fieldsend** **Richard M. Everson**[1]

Abstract. Information access tasks involve setting a large number of model parameters (such as term weighting functions or which terms from the document collection to use); in addition, information access tasks frequently attempt to satisfy multiple competing objectives. We investigate the use of multi-objective evolutionary algorithms for finding model parameters that optimise both precision and recall. Methodologically, optimising precision/recall curves is equivalent to optimising ROC curves (describing the trade off between recall and false positive rate). 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 optimised term weighting functions in retrieval and classification tasks: tf, idf, $tf \times idf$, $(tf \times idf)/ndl$, BM25 and neural networks. We also optimise the range of terms used for document classification. We show that performance can be significantly enhanced beyond default parameter settings. This study presents a methodology for improving, analysing and evaluating performance for information access tasks.

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 (term weighting method and values of associated variables, stop-word removal, terms from the document collection to use, *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, leading to a trade-off curve similar to ROC (receiver operating characteristic) curves. Clearly, simultaneous optimisation of several objectives for a single solution is usually impossible 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. This has probably been the case as the only method to guarantee the return of optimal parameters, exhaustive search, is infeasible. Also, the most popular search algorithms for complex problems with many parameters, for example simulated annealing, genetic algorithms and evolution strategies (ES) [9] 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, [5, 26].

In this paper we present a methodology for improving, analysing and evaluating performance for information access tasks. We investigate the use of an MOEA to estimate optimal values for model parameters in information access tasks. 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 behaviour of the models.

Section 2, *information access tasks*, discusses the IR and DC tasks tackled in the experiments, term weighting methods and term ranges. It also introduces the evaluation measures (or objectives), precision and recall, used in the experiments and discusses the reasons that these are more suitable for information retrieval tasks than the measures, recall and false positive rate, used in ROC analysis. Section 3, *optimality and the MOEA*, discusses the meaning of Pareto optimality, how to compare Pareto fronts and the algorithm used for the experiments. Section 4 describes the design of the experiments and gives the results. Finally, Section 5 discusses the results and areas of future work.

2 INFORMATION ACCESS TASKS

The vector space model underlies many information access endeavours [2] 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 $\mathcal{D} = \{d_1, ..., d_J\}$ as a bag-of-words or terms, $t_i \in \mathcal{T} = \{t_1, ..., t_I\}$. Each document is represented by a vector of non-negative 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

[1] Department of Computer Science, University of Exeter email: {M.J.Fisher, J.E.Fieldsend, R.M.Everson}@exeter.ac.uk

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. IR and DC both 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, which is represented by a document d_q, where weights $w_{i,q}$ 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 the distance between documents is measured by the cosine similarity measure:

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

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

The terms $|d_q|$ 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. The popular BM25 weighting scheme [20] incorporates more sophisticated document length normalisation in its assignment of term weights, hence the unnormalised inner product is used to measure distance between documents when utilising the BM25 term weighting method:

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

2.2 Classification

In text classification each document d_j in the document collection is associated with one of M classes $c_j \in \{c_1, ..., c_M\}$. 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 straight forward and commonly used classification method 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 around d_q containing K neighbours. The K-nn classifier can be formulated to provide probabilities of class membership [3]: 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 at $P(c_m|d_q) = K_m/K$. K-nn classifiers can be shown to have an asymptotic error rate no worse than twice the optimal Bayes error [10]. In an identical way to IR the K-nearest neighbours are determined by distances derived from the cosine similarity (1) and (2).

2.3 Choosing terms

Luhn [16] predicted that the most and least frequently occurring terms in a document collection would not be particularly useful for information access tasks but, as illustrated in Figure 1, that the terms within a middle range would be most useful. 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

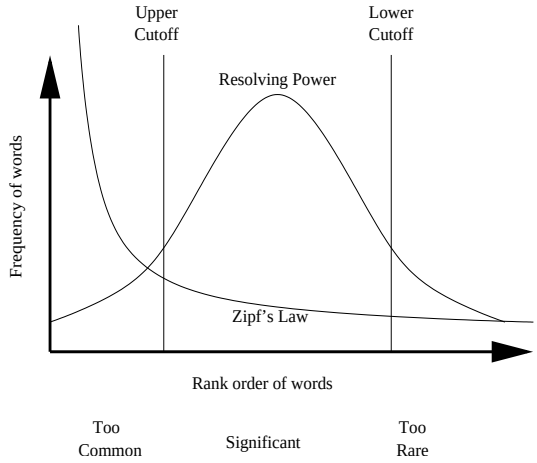

Figure 1. Resolving power of words (adapted from [22]).

queries. Normally, these upper and lower cut-off points are estimated by removing the most common terms in a document collection using a stop-word list and removing terms that occur only a few times throughout the collection [2]. Often cut-off points that have proven successful in previous systems are used or a few different cut-off points are tested and those which produce the best performance are used; however, these methods are clearly suboptimal. Here we investigate the use of multi-objective optimisation to find the upper and lower cut-off points that produce the best precision and recall performance for information access tasks.

2.4 Term weighting methods

Many term weighting methods have been introduced, see [19] for examples, but in this study we examine the six term weighting methods discussed below.

Term frequency, $tf_{i,j}$: Luhn [17] observed that: *repetition is an indication of emphasis.* Frequently used terms in a document are important for 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 i occurs in document j:

$$w_{i,j} = tf_{i,j} \quad (3)$$

Inverse document frequency, idf_i: This method is concerned with 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. Weights are assigned as:

$$w_{i,j} = idf_i = \log \frac{J}{df_i} \quad (4)$$

where J is the number of documents in the collection and df_i is the number of documents in the collection containing t_i.

TFIDF: The well known $tf \times idf$ measure [21] simply combines the tf and the idf weights:

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

TFIDF normalised by document length: The TFDIF weight is often normalised by document length to prevent longer documents obtaining higher weights simply because they are verbose. The resulting weight is:

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

where the normalised document length $ndl_j = dl_j/avdl$, the average document length $avdl = \frac{1}{J}\sum_j^J dl_j$ and the length of document j is denoted by dl_j. A more sophisticated document normalisation method is presented in [23].

BM25: The popular BM25 weighting [20] incorporates non-linear document length normalisation and term frequency weights:

$$w_{i,j} = \frac{tf_{i,j} \times (k+1)}{k \times ((1-b) + (b \times ndl_j)) + tf_{i,j}} \times \frac{J}{df_i} \tag{7}$$

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. Common values are $b = 0.75$ and $k = 2$ [19].

Neural networks: The term weighting methods above all combine document length, term frequency and document frequency to arrive at the term weight. Here we use a neural network as a flexible 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, $\boldsymbol{\theta}$:

$$w_{i,j} = g(\bar{dl}_j, \bar{tf}_{i,j}, \bar{df}_i; \boldsymbol{\theta}) \tag{8}$$

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 $\bar{df}_i$. In this study we use a simple multi-layer perceptron network (MLP) [3] with three hidden units. Instead of learning the weights in the traditional manner, of back propagation for example, we use the multi-objective optimisation framework to learn the 16 weights by training on the document collection.

2.5 Evaluation

In order to make connections between the DC and IR paradigms [15], 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 class	relevant	TP	FP
	irrelevant	FN	TN

Here TP (true positives) are the documents correctly classified as belonging to the *relevant* class, FP (false positives) are the documents incorrectly classified as belonging to the *relevant* class, FN (false negatives) are the documents belonging to the class but not classified as such and finally TN (true negatives) are documents correctly classified as not belonging to the class.

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 n documents, from which a confusion matrix may be calculated. 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:

$$\mathcal{P} = \frac{|TP|}{|TP| + |FP|} \qquad \mathcal{R} = \frac{|TP|}{|TP| + |FN|} \tag{9}$$

Precision is the fraction of the documents classified as relevant that actually are relevant, while recall quantifies 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 get a 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 9 are both constant; $|TP|+|FP| = n$, and $|TP| + |FN|$ is the number of relevant documents in the collection.

In IR it is usual to measure a system's performance by precision and recall. We wish to maximise both precision and recall – that is return to the user a list of documents containing all the relevant documents and only the relevant documents. Equivalently, the objectives of ROC analysis are to return all of the relevant documents and no irrelevant documents, but it does so by maximising recall (called the true positive rate) and minimising the false positive rate:

$$\mathcal{F} = \frac{|FP|}{|FP| + |TN|} \tag{10}$$

$\mathcal{F}$ shows how many of all possible documents belonging to the irrelevant class have been classified as relevant. $\mathcal{F}$ is equivalent to the, rarely used, fallout measure in IR [25].

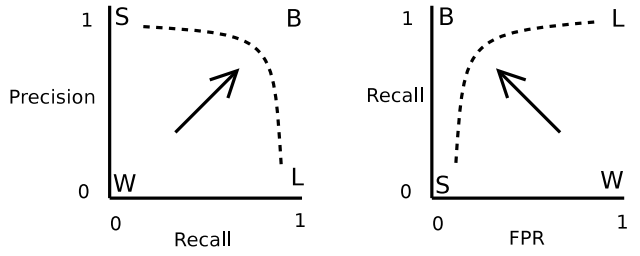

Figure 2. Left: precision/recall curve. Right: ROC curve. The arrows denote the direction from worst classifier (W) to best classifier (B) and S and L indicate equivalent points on each graph.

The optimal or *best classifier* is equal in both precision/recall and ROC curves, however, they differ in their definition of the *worst classifier*. The worst classifier on a precision/recall plot is defined as one that returns to the user a list of documents containing no relevant documents. The definition of the worst classifier in an ROC plot is one which returns no relevant documents and all of the irrelevant documents to the user. However, in IR any classifier returning no relevant documents is equally unhelpful to the user regardless of the number of irrelevant documents returned. Precision/recall curves are therefore well suited to IR task evaluation.

Figure 2 shows how precision/recall curves are related to ROC curves. The best classifier (B) on both plots is equivalent: $(\mathcal{R} = 1 \wedge \mathcal{F} = 0) = (\mathcal{R} = 1 \wedge \mathcal{P} = 1)$. For $\mathcal{R}$ to equal 1 the $|TP|$ must be at its maximum ($|TP|$ = the number of relevant documents) and, consequently, $|FN|$ must equal 0. Given $\mathcal{R} = 1$, for both $\mathcal{F}$ to equal 0 and $\mathcal{P}$ to equal 1 then $|FP|$ must equal 0 and, therefore, $|TN|$ must be at its maximum. The worst classifier (W) on both plots is not equivalent: $(\mathcal{R} = 0 \wedge \mathcal{F} = 1) \neq (\mathcal{R} = 0 \wedge \mathcal{P} = 0)$ because if $\mathcal{F} = 1$ then $|FP|$ must be maximised, whereas for $\mathcal{P}$ to equal 0 the only requirement is that $|TP| = 0$; $|FP|$ can have any value.

Points close to S on both the plots can be obtained by returning very few documents (assuming that most documents are irrelevant) for both precision/recall and ROC curves. S can be obtained in ROC curves by returning 0 documents to the user, however, in precision/recall curves that would produce the worst classifier (W) point because $|TP| = 0$. S cannot actually be obtained in precision/recall curves because $|TP|$ cannot equal 0 (to produce $\mathcal{R} = 0$) and a positive number (to produce $\mathcal{P} = 1$) at the same time, however, points close to S can be obtained by returning a small number of relevant documents to the user.

Points close to L can be obtained by returning a large number of documents to the user (assuming that most documents are relevant) for both precision/recall and ROC curves. L can be obtained for ROC curves by assuming that all documents are relevant, however, for precision/recall curves L cannot actually be obtained because for recall to be high then $|TP|$ must be high. However, assuming that all documents are relevant obtains the closest point possible to L because the denominator of $\mathcal{P}$ is at its largest.

As discussed in Section 3, comparison of multi-objective systems is difficult because it is not clear when one system is outperforming another. 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 [12]:

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

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.

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 Pareto front before deciding on the importance of each objective. A precision/recall curve may be obtained, for a single set of model parameters, simply by adjusting the number of documents, n, returned to the user; a larger n results in high recall and low precision and vice versa. We could imagine discovering the optimal set of model parameters by creating the precision/recall curves for all possible sets of model parameters and selecting only the convex hull of these precision/recall curves. In a similar manner the area under the ROC curve (AUC) [4, 11] might be computed for all possible sets of model parameters, the convex hull of these being the optimal recall/false-positive-rate trade-off curve. However, such exhaustive search is computationally infeasible for realistic problems, and we therefore use MOEAs to discover the thresholds n and model parameters (term weighting method, value of term weighting method parameters) that optimise the objectives precision $\mathcal{P}$ and recall $\mathcal{R}$ for IR and DC. 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 DC. 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 performance of their model.

3 OPTIMALITY AND THE MOEA

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, $\boldsymbol{\theta}$:

$$y_i = f_i(\boldsymbol{\theta}), \qquad i = 1, \ldots, D \tag{12}$$

Here the objectives are precision, $\mathcal{P}(\boldsymbol{\theta})$, and recall, $\mathcal{R}(\boldsymbol{\theta})$, so $D = 2$. The parameters are the parameters of whichever term weighting model is employed, 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

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

	Inputs.
T	Maximum number of ES generations.
Ω	Perturbation distribution; normal with mean zero and variance $(0.1)^2$.
α	Probability of perturbation ($\alpha = 0.8$).
n_{max}	Number of different thresholds, for IR $n_{max} = 5000$, and in classification $n_{max} = 54100$.
1:	Initialise random decision vector $\boldsymbol{\theta}$, with model parameters lying in valid ranges. Generate set M_1 by replicating $\boldsymbol{\theta}$ n_{max} times but with different thresholds in its last element. Initialise archive $F := \emptyset$, and generation $t := 1$.
2:	Evaluate precision and recall for elements of M_t. That is, evaluate $\mathbf{f}(\boldsymbol{\theta}_i) \forall \boldsymbol{\theta}_i \in M_t$.
3:	Insert into F those individuals $\boldsymbol{\theta}_i \in M_t$, that are not dominated: $F := F \bigcup \{\boldsymbol{\theta} \in M_t \,|\, \boldsymbol{\theta} \nprec M_t \bigcup F\}$.
4:	Remove from F any dominated individuals.
5:	Copy a decision vector, $\boldsymbol{\theta} \in F$: $\boldsymbol{\phi} := \boldsymbol{\theta}$. (Partitioned quasi-random selection method [7] is used to select $\boldsymbol{\theta}$.)
6:	Perturb the elements of $\boldsymbol{\phi}$ with probability α: $\phi_p := \phi_p + \Omega \;\; \forall p = 1, \ldots, P-1$, Generate M_{t+1} by replicating $\boldsymbol{\phi}$ n_{max} times with different thresholds in its last element.
7:	$t := t + 1$. If $t = T$, end, else goto 2.

$\boldsymbol{\theta} = (k, b, K, l, u, n)$ for DC with BM25 term weighting and using K nearest neighbours.

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

$$\text{Maximise} \quad \mathbf{y} = \mathbf{f}(\boldsymbol{\theta}) = (f_1(\boldsymbol{\theta}), \ldots, f_D(\boldsymbol{\theta})) \tag{13}$$

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

When concerned with a single objective an optimal solution is one which maximises the objective given any model constraints. However, 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* [26] and the set of all Pareto optimal solutions is 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

$$\begin{aligned} f_i(\boldsymbol{\theta}) \geq f_i(\boldsymbol{\phi}) \quad \forall i = 1, \ldots, D \quad \text{and} \\ f_i(\boldsymbol{\theta}) > f_i(\boldsymbol{\phi}) \quad \text{for some } i. \end{aligned} \tag{14}$$

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:

$$F_k \nprec F_j \quad \forall k, j = 1, \ldots, |F|. \tag{15}$$

3.2 The optimisation algorithm

Anastasio, Kupinski & Nishikawa [1] introduced the use of multi-objective evolutionary algorithms (MOEAs) to optimise

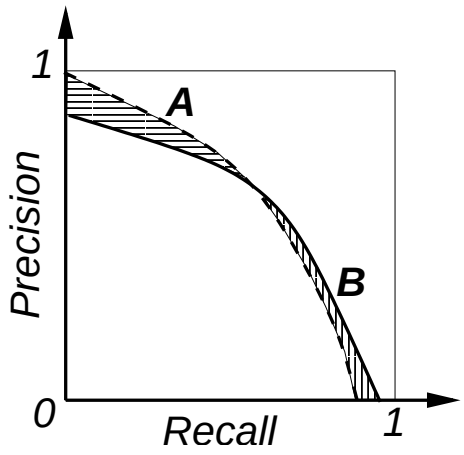

Figure 3. Comparison of Pareto fronts, A (dashed) and B (solid). Solutions in the horizontally hatched region (of area $\mathcal{V}(A, B)$) are dominated by A, but not by B, whereas solutions in the vertically hatched region (of area $\mathcal{V}(B, A)$) are dominated by B but not A.

ROC curves. Here we use to similar methodology, albeit with improved convergence properties, to optimise precision/recall curves.

The MOEA used in this study is a based on a simple (1+1)-ES, similar to that introduced in [13]. 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. At each stage of the algorithm some solutions in F are copied and perturbed. Those perturbed solutions that are dominated by members of F are discarded, while the others are added to F and any dominated solutions in F removed. In this way the estimated Pareto front F advances towards $\mathcal{E}$. Algorithm 1 shows in more detail how this MOEA works in the context of optimising information access models. This algorithm, unlike earlier versions [13], maintains an archive which is unconstrained in size, permitting better convergence properties [7].

In a $(\mu+\lambda)-$ES, μ decision vectors are perturbed to generate λ new decision vectors. That is, μ parameter vectors are selected (whose performance have already been evaluated); these *parents* are copied and have their parameter values perturbed in order to generate λ *children*. This perturbation is typically the addition of a random value from a Normal distribution (part 6 of of Algorithm 1). These children are then evaluated and compared to their parents. Children that are not dominated by any other child or by a decision vector already in F are added to F; otherwise the child is discarded. Any elements of F that are dominated by the new children are removed.

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, but 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 re-

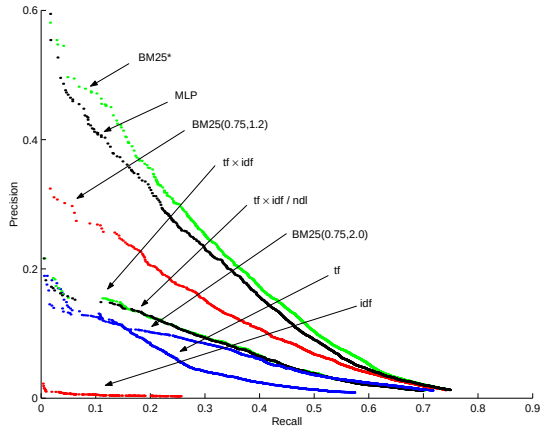

Figure 4. Pareto fronts for the various term weighting methods in the information retrieval task, using the short queries on the training data, fold 1.

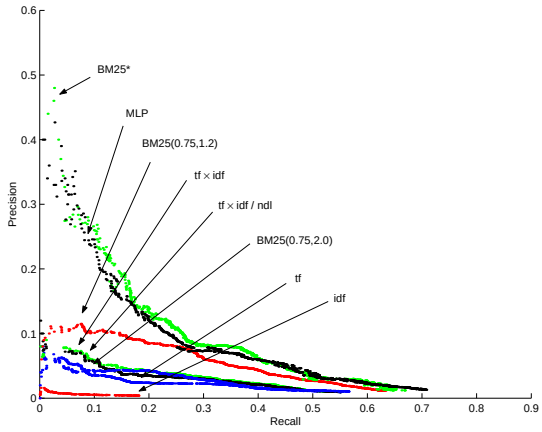

Figure 5. Evaluation on the test data of the various model sets, optimised with respect to the training data. Short query IR data, fold 1.

turned, *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 there are 1028 documents in the test set and hence thresholds up to $n = 1082$ are evaluated. The optimiser therefore acts as an $(1 + n_{max})$-ES, where $n_{max} = 5000$ for IR and $n_{max} = 54100$ for classifiers (an additional factor of $K = 50$ models are evaluated, one for each number of nearest neighbours).

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 inferior or superior to another [14]. For example as illustrated in Figure 3 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 [7]. Note that $\mathcal{V}(A, B) \neq \mathcal{V}(B, A)$. 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 [7] for details.

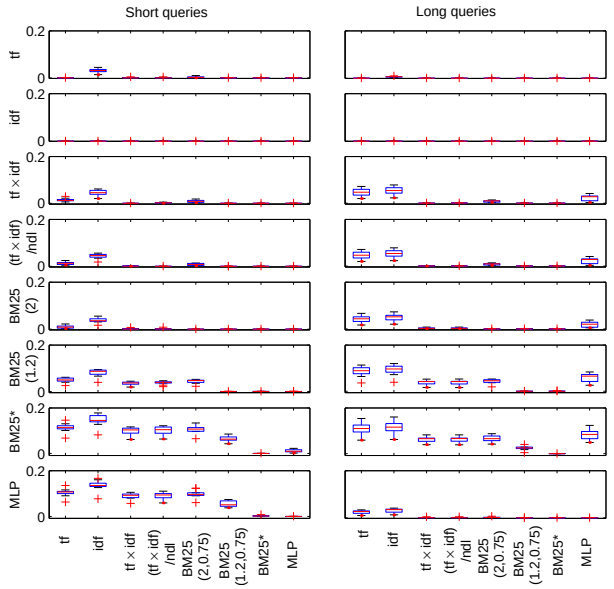

Figure 6. Box-plots of the volume measure comparing the seven different IR term weighting methods using test folds for both short (left) and long (right) queries.

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 [24]. 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 partitioning was performed 10 times to give 10 different folds to compare when evaluating the results. Since the training and test sets are constant for all classifiers being compared, the precision/recall curves are as stable as ROC curves. Results on the test data are averaged over all of the queries in the test set and results on the training 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 over-fit the data. Term weighting

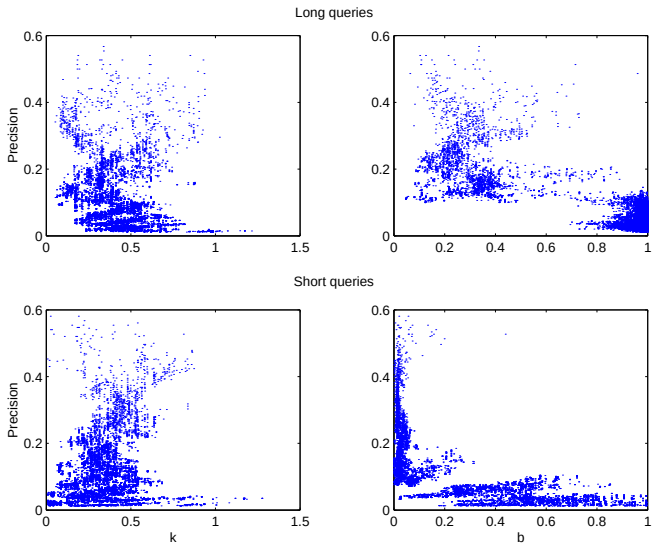

Figure 7. Plots of k and b versus precision, taken from models on the trade-off front of the BM25* model.

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 $b = 0.75$ and $k = 2$ [19]. We denote BM25 term weighting with these traditional, fixed values as BM25; 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 method, denoted by MLP, for which the weights and biases producing optimal $\mathcal{P}$ and $\mathcal{R}$ values are found.

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.

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 which was run for 2000 generations to ensure convergence.

4.1.1 Results

Figures 4 and 5 show examples of the fronts found for the first fold of the TREC data. 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 could result in significant improvements in performance. Interestingly, 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 MLPs perform significantly better under the Wilcoxon non-parametric signed ranks test [27] than all other term weighting methods. The BM25* method with optimised parameters significantly outperforms the tf, idf, $tf \times idf$, $tf \times idf/ndl$ and BM25 methods for all values of precision and recall. The $tf \times idf$, $tf \times idf/ndl$ and BM25 methods obtain roughly equivalent results outperforming both tf and idf. 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. This result is probably due to the increased complexity of the problem with long queries: increasing the flexibility of the neural networks by increasing the number of hidden units should yield better performance.

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

The results of $\mathcal{V}$ measure comparison of the IR term weighting methods on test data is shown in the form of boxplots in Figure 6. These boxplots show the evaluation of weighting methods over 10 folds. Each of the two plots contains seven rows, where each row represents $\mathcal{V}$ for one method with respect to all 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× idf)*, $\mathcal{V}$*(tf,(tf× 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.

Figure 7 shows the optimised k (left) and b (right) values obtained from all folds using the BM25* method plotted against precision, for both short (bottom) and long (top) queries. 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 and as $b \to 0$ the higher the precision obtained, indicating that very little document length normalisation is required. For long queries, a small b still obtains higher precision but there is a wider spread of b values with the average b value close to 1, high recall is obtained when b is very close to 1. For long queries, it appears that document length normalisation is more important.

A typically used value of k is 2 [19]; it is surprising then that we find, for our data and model, that optimal k values are much lower: the average is $k \approx 0.5$ for both short and long queries, indicating that BM25 does not need to be as sensitive to changes in term frequency.

Unsurprisingly, we find that models on the Pareto front with high thresholds, n (returning many documents) generally have high recall and low precision, and vice versa.

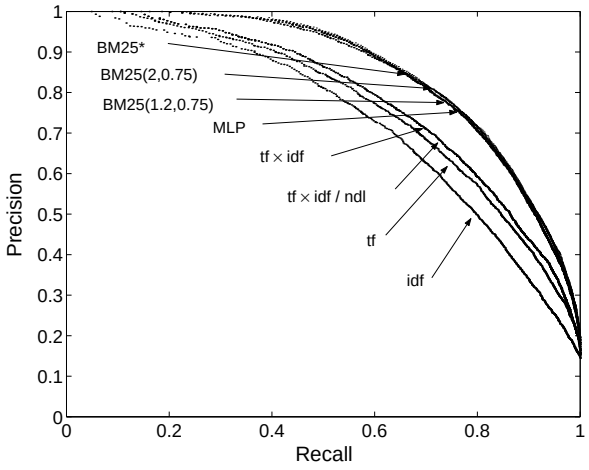

Figure 8. Pareto fronts for the various term weighting methods in the classification task on the training data.

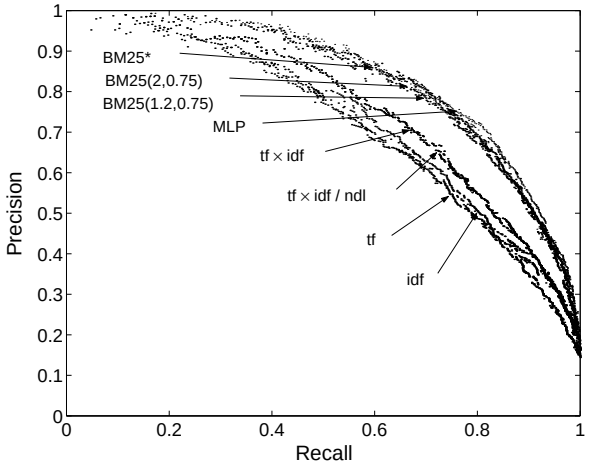

Figure 9. Evaluation of the various model sets, optimised with respect to the training data, on the test data for the various term weighting methods in the classification task.

4.2 Classification

For the classification task we used the Cora document collection [18] which was collected automatically by intelligent web spiders. The data set is comprised of approximately 37000 papers, all of which have been automatically classified into hierarchical categories such as /Artificial_Intelligence/-Machine_Learning/Theory. In common with others [6, 8], 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, and in a similar manner to the IR task the results are averaged over 10 such folds.

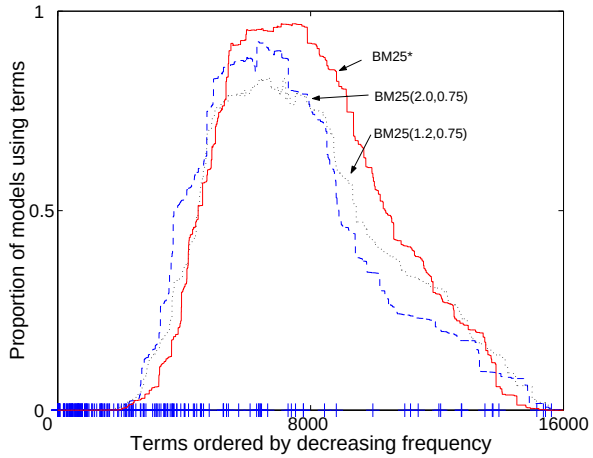

Figure 10. Histogram showing the terms used by the different BM25 models on the trade-off front. The abscissa shows the rank of terms ordered by term frequency, with stop-word position indicated by ticks.

As in the IR task, we calculate the average precision and average recall at different thresholds i, $1 \leq i \leq n = 1082$; 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 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 15000 terms in all.

4.2.1 Results

Figures 8 and 9 show the fronts obtained on the classification task using the first fold of the Cora document collection. 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 document length normalisation (Equation 1) is sufficient, although there is a relatively small range of document lengths in this collection. As discussed in section 2.5, the classification task is easier than the information retrieval task in the sense that many exemplars from the few known classes are available for training; this explains why the BM25 and BM25* methods show similar performance and why the MLP shows only a slight performance improvement.

As found for the IR task the recall increases as the cut-off n increases. 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 10 displays a histogram of the range of terms used by the different BM25 (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 quantitatively that 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 since these are more suitable objectives for IR than other possible measures (see Section 2.5). We obtained optimal *sets* of model parameters which describe the trade-off between these objectives on the training and test data used.

We have demonstrated that, on TREC data, multi-objective optimisation can locate parameters for the BM25 model that yield significantly better performance than the popular default parameter settings. Even with the simple term weighting methods used in this study, significant performance improvements can be obtained through the use of an MOEA optimisation framework. We have also shown that simple MLPs, trained using multi-objective optimisation, perform very well both on the classification and the information retrieval problems. 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, in a similar manner to ROC curve analysis, allow the designer to set the operating point without making *a priori* assumptions.

We have defined optimality in terms of two objectives: precision and recall, it would be interesting future work to investigate the inclusion of a third objective: fallout or false positive rate. However, increasing the number of objectives would also increase the computational burden.

We emphasise that the experiments presented here used unsophisticated information retrieval models – the vector space model and K-nearest neighbours classification; we used no query expansion or (pseudo) relevance feedback, for example. Nonetheless the methods are 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

[1] M. Anastasio, M. Kupinski, and R. Nishikawa, 'Optimization and froc analysis of rule-based detection schemes using a multiobjective approach', *IEEE Transactions on Medical Imaging*, **17**, 1089–1093, (1998).

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

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

[4] A. P. Bradley, 'The use of area under the roc curve in the evaluation of machine learning algorithms', *Pattern Recognition*, **30**(7), 1145–1159, (1997).

[5] 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).

[6] 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, pp. 430–436, (2001).

[7] 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).

[8] 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*, pp. 41–56, (2003).

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

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

[11] J. A. Hanley and B. J. McNeil, 'The meaning and use of the area under a receiver operating characteristic (roc) curve', *Radiology*, **143**, 29–36, (1982).

[12] N. Jardine and C. J. van Rijsbergen, 'The use of hierarchic clustering in information retrieval', *Information storage and retrieval*, **7**, 217–240, (1971).

[13] J. Knowles and D. Corne, 'The Pareto archived evolution strategy: A new baseline algorithm for Pareto multiobjective optimisation', in *1999 Congress on Evolutionary Computation*, pp. 98–105, (1999).

[14] J. Knowles and D. Corne, 'On Metrics for Comparing Non-dominated Sets', in *2002 Congress on Evolutionary Computation*, pp. 711–716, (2002).

[15] D. D. Lewis, 'Evaluating Text Categorization', in *Proceedings of Speech and Natural Language Workshop*, pp. 312–318. Morgan Kaufmann, (1991).

[16] H.P. Luhn, 'The creation of literature abstracts', *IBM Journal of Research and Development*, **2**, 159–165, (1958).

[17] H.P. Luhn, 'The automatic derivation of information retrieval encodements from machine-readable texts', in *Information retrieval and machine translation*, ed., A. Kent, volume 3, 1021–1028, Interscience, (1961).

[18] 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`.

[19] S. Robertson and K. Sparck Jones, 'Simple proven approaches to text retrieval', Technical Report TR356, Cambridge University Computer Laboratory, (1997).

[20] 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).

[21] G. Salton and C.S. Yang, 'On the specification of term values in automatic indexing', *J. Documentation*, (1973).

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

[23] 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)*, pp. 21–29, (1996).

[24] Text REtrieval Conference (TREC) Home Page. `www.trec.nist.gov/`.

[25] C.J. van Rijsbergen, *Information Retrieval*, Butterworths, 1975.

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

[27] F. Wilcoxon and R.A. Wilcox, *Some Rapid Approximate Statistical Procedures*, Lederle Labs, New York, 1964.