Adaptable Similarity Search using Non-Relevant Information

Ashwin T.V.
vashwin@in.ibm.con

Rahul Gupta
rahulgup@in.ibm.con

Sugata Ghosal
gsugata@in.ibm.con

IBM India Research Lab, Hauz Khas, New Delhi 110016, India

Abstract

Many modern database applications require content-based similarity search capability in numeric attribute space. Further, users' notion of similarity varies between search sessions. Therefore online techniques for adaptively refining the similarity metric based on relevance feedback from the user are necessary. Existing methods use retrieved items marked relevant by the user to refine the similarity metric, without taking into account the information about non-relevant (or unsatisfactory) items. Consequently items in database close to non-relevant ones continue to be retrieved in further iterations. In this paper a robust technique is proposed to incorporate non-relevant information to efficiently discover the feasible search region. A decision surface is determined to split the attribute space into relevant and non-relevant regions. The decision surface is composed of hyperplanes, each of which is normal to the minimum distance vector from a non-relevant point to the convex hull of the relevant points. A similarity metric, estimated using the relevant objects is used to rank and retrieve database objects in the relevant region. Experiments on simulated and benchmark datasets demonstrate robustness and superior performance of the proposed technique over existing adaptive similarity search techniques.

Keywords : *Information retrieval, Database browsing, Database navigation, Ellipsoid query processing, Relevance feedback, Non-relevant judgement*

Proceedings of the 28th VLDB Conference,
Hong Kong, China, 2002

1 Introduction

In many modern database applications, it is necessary to be able to pose queries in terms of similarity of data objects rather than relational operations like equality or inequality. Examples include finding sets of stocks that behave in approximately the same (or for that matter opposite) way in a temporal database [16]; searching for structurally similar proteins from a spatial database which can cause a particular medical condition [13]; retrieval of 3D objects from a CAD database [2], finding similar objects based on content from multimedia databases containing audio, image or video [11]. Also, several approximation schemes have been developed to efficiently process similarity queries using multidimensional indexing structures [19, 1, 3]. In this paper, we focus on accuracy improvement of similarity based retrieval of database objects using relevance feedback.

To support similarity based modern database applications, multidimensional attribute or feature vectors are extracted from the original object, and stored in the database. Given a query object (associated with it an attribute vector), database objects whose attribute vectors are most similar to the query vector are retrieved for the user. Usually k top matches are retrieved. For text applications, *vector space model* with cosine similarity metric is being widely used [17],[4]. The cosine similarity is defined as, $\mathcal{S}(\vec{x},\ \vec{q}) = \frac{\vec{x}.\vec{q}}{||\vec{x}||||\vec{q}||}$, where $\vec{u}.\vec{v}$ stands for inner-product of $\vec{u}$ and $\vec{v}$, and $||\vec{v}||$ denotes the magnitude of $\vec{v}$. On the other hand, in *metric space model* of information retrieval, second-order (L_2) distance metrics are typically used. The second-order L_2 norm is a quadratic distance metric and is defined as $\mathcal{D}(\vec{x},\ \vec{q},\ Q) = (\vec{x} - \vec{q})^T Q(\vec{x} - \vec{q})$. Imposing constraints on the structure of the matrix Q, metrics like Euclidean (Q is the identity matrix), weighted Euclidean (diagonal Q) and generalized Euclidean (symmetric Q with off-diagonal entries) are obtained. Examples of application of quadratic distance metrics include Euclidean metric in discrete wavelet transform (DWT) based attribute space for time-series stock data [16], color histogram space for color image databases [11], Fourier descriptors (FD) for shape databases [2]; weighted Euclidean metric for multimedia object retrieval [18], and generalized Euclidean distance metric for spatial databases [10]. Applicability of L_2 norms requires that the users' *information need* be modeled by a compact convex set in the feature space.

The *query by example* paradigm is typically used for similarity based search and retrieval from databases. In many emerging database applications, the notion of similarity cannot be predetermined, and needs to vary across search sessions to satisfy information need of the user (it is not likely that a user would be able/willing to supply the similarity metric). For example in an online car shopping scenario, a buyer starting with an initial query (e.g. Jeep Cherokee), may be interested in "weekend getaway vehicle" (based on cargo capacity, wheelbase, and torque considerations), whereas another buyer starting with the same initial query may instead be looking for an inexpensive family vehicle (based on engine size, weight, and price considerations). In these scenarios the retrieval processes proceeds as follows. The system presents a set of objects from the database to the user based on an initial similarity metric. The user expresses his liking/disliking of the retrieved set of objects. Based on this relevance feedback from the user, the similarity metric is adapted, and a new set of objects that are likely to be more relevant to the user are retrieved. This process continues until the user is satisfied. Most of the existing systems [18, 10] use only relevant objects to refine the similarity metric. Since information about the non-relevant objects is not used, other database objects close to the non-relevant objects are also typically retrieved.

In this paper we propose a novel means of incorporating non-relevant judgement for improving the performance of similarity based retrieval. Non-relevant information is not used for ranking similar items in our approach. We, instead use non-relevant information to define a feasible region in the feature space. Relevant objects are used to estimate the parameters of the similarity metric. Similar objects are ranked and retrieved from within the feasible region only.

The remainder of this paper is organized as follows. Existing work that attempted to incorporate non-relevant judgement in similarity retrieval is presented in Section 1.1. A mathematical formulation of our approach is presented in Section 2. The proposed solution and retrieval algorithm is presented in Section 3. The performance of the proposed approach is experimentally demonstrated using simulated and benchmark datasets in Section 4. Finally we summarize our contributions and mention some future directions for research in Section 5.

1.1 Related Work

We now analyze approaches considered in the past to incorporate non relevant information. Relevance feedback was first introduced by Rocchio in the context of incremental (iterative) text categorization using keywords [17]. Each document is associated with an attribute vector, each component (term) of which represents the normalized frequency of occurrence of the corresponding keyword. Rocchio presented the following formula to compute the new query,

$$\vec{q}_{new} = \alpha \vec{q}_{curr} + \frac{\beta}{|\mathbb{G}|} \sum_{\vec{x}_i \in \mathbb{G}} \vec{x}_i - \frac{\gamma}{|\mathbb{B}|} \sum_{\vec{y}_i \in \mathbb{B}} \vec{y}_i \qquad (1)$$

where $\vec{q}_{curr}$ is the current query vector, $\mathbb{G}$ is the set of relevant objects and $\mathbb{B}$ the set of non relevant objects. Values for α, β and γ are empirically chosen for the document set. Documents in the database are ranked based on their cosine distances from the estimated query $\vec{q}_{new}$. The parameter values that give good overall performance for a document set requires careful fine tuning. Even in case of 2-D feature space, it is easy to construct cases where for a fixed set of parameters, a large number of non-relevant judgements move the query vector toward the non-relevant region. Singhal *et.al.* [5] propose a dynamic query zoning scheme for learning queries in Rocchio's framework by using a restricted set of non-relevant documents in place of the entire set of non-relevant documents. In a recent study [6], Dunlop concludes that with Rocchio's formula using non-relevance feedback, the results show behaviors that can hardly be justified and vary widely. Hence recent approaches have chosen to ignore non-relevant documents [12]. Even though Rocchio's formula is primarily designed for use in the context of vector space models, it is straightforward to extend the formula for metric space models [10]. Problems with unpredictable performance similar to the vector space model still persist.

Nastar *et al.* [15] use two non-parametric density estimates to model the probability distribution for the relevant and non-relevant classes. The relevance score of a given database object is defined as,

$$RelScore(x) = \frac{Prob(\vec{x} \in relevant)}{Prob(\vec{x} \in nonrelevant)}. \qquad (2)$$

The intuition behind this formula is that points in feature space either having large probability of being relevant or small probability of being non-relevant should receive high relevance scores. It is easy to see that even though the probabilities obtained through maximum likelihood estimates are well behaved, the relevance score being a ratio is not guaranteed to give consistent results. For example consider a point in the feature space with a small estimated probability of being relevant and with nearly zero estimated probability of being non-relevant, this point will receive a very high relevance score which may far exceed that of a point whose probability of being relevant is high (≈ 1) and probability of being non-relevant is a small non zero value.

A similar heuristic has also been proposed by Brunelli *et al.* [7] where they suggest the following distance function,

$$\mathcal{D}(\vec{x}, G, B) = D(\vec{x}, G)\left[\frac{D(\vec{x}, G)}{D(\vec{x}, B)}\right]^{\gamma} \quad (3)$$

$$D(\vec{x}, S) = \sum_{\vec{y} \in S} ||\vec{x} - \vec{y}||^2. \quad (4)$$

The database points that are either close to relevant objects or far from non-relevant objects should have small net distance $\mathcal{D}$. Since the distances D are obtained through summation over all relevant or non-relevant objects, the net distance is sensitive to the size of relevant and non-relevant sets. Also the resulting distance $\mathcal{D}$ function is not well behaved and can lead to unpredictable results.

Our key contribution in this paper is the use of non-relevant judgements to delineate the relevant region in the feature space, ensuring that the restricted search space does not contain any non-relevant objects. Relevant judgements are used to estimate a similarity metric which then is used to rank and retrieve database objects in the relevant region. In machine learning literature, decision trees [8] are routinely used to achieve a partitioning of the feature space. Inducing a decision tree on the relevant and non-relevant objects may result in multiple disconnected relevant regions. Disconnected relevant regions are clearly incompatible for ranking using a single quadratic distance metric. One can envision a technique wherein a similarity metric is estimated independently for each relevant region. However it would be difficult to obtain robust estimates for the similarity metrics given the small number of relevant judgements in each of the relevant regions. Decision trees are also known to perform poorly with small training sets.

Support Vector Machines are popularly used to solve 2-class classification problems [9]. SVMs transform the original feature space using a kernel (usually a Gaussian kernel) and estimate a optimal hyperplane to separate the two classes in the new feature space. The distance from the hyperplane is used as a measure of belonging to a class. The local nature of the mapping results in the ranking function (in the original feature space) having bumps at relevant objects and being relatively flat elsewhere, to attenuate this effect a large and representative training set is essential.

To overcome these issues, the partitioning of the feature space in the proposed algorithm is achieved by using a piecewise linear decision surface that separates the relevant and non-relevant objects. Each of the hyperplanes constituting the decision surface is normal to the minimum distance vector from a non-relevant point to the convex hull of the relevant points. Our algorithm robustly estimates the hyperplanes that constitute the decision surface even when the size of feedback is small. The relevant region is obtained as the result of intersection of half spaces and hence forms a convex subset of the feature space. This ensures that we can use any similarity metric to rank and retrieve database objects inside the relevant region. Since the estimated relevant region is convex and the quadratic distance metric is a convex function on the feature space, we are ensured that there are no 'pockets' in the feature space where unpredictable relevance scores are possible.

Symbol	Definition
$\mathbb{D}$	The database
n_d	Dimensionality of the database
$\mathbb{G}$	Relevant set retrieved in the current iteration
$\vec{v}$	Goodness scores assigned by user to relevant objects. (1 if user only identifies relevant objects).
$\mathbb{B}$	Non-relevant set retrieved in the current iteration
k	Number of data objects retrieved per iteration
$\vec{q}_{curr}, (\vec{q}_{opt})$	Query center for the current iteration, optimal solution with current set of feedback
$Q_{curr}, (Q_{opt})$	Inverse covariance matrix for the current iteration, optimal solution with current set of feedback
$d(.,.,.)$	Generalized Euclidean distance function

Table 1: Symbol usage

We experimentally demonstrate the effectiveness of the proposed algorithm to improve precision and recall. In over 50% of the experiments there is a significant performance improvement over using only relevant objects. In rest of the experiments improvement was not visible because relevant sets were compact enough compared to the distribution of non-relevant points in their neighborhood. In small fraction of the experiments, there is an inconsequential (around 0.05) performance degradation due to approximation of the feasible region using piecewise linear surfaces.

2 Problem formulation

At each iteration of relevance feedback the judgements provided by the user constitute a relevant set $\mathbb{G}$ and a non-relevant set $\mathbb{B}$. If the user provides different degrees of desirability for the relevant objects, then this information is available as a vector of goodness scores $\vec{v}$. If user only marks relevant objects then the goodness scores for all relevant objects are set to 1. All objects seen by the user and marked neither relevant nor non-relevant are not considered for further computation.

Let $\vec{x}$ and $\vec{q}$ represent the feature vectors corresponding to a database object and the estimated center of the query[1]. Then a quadratic distance function to measure the distance of $\vec{x}$ from $\vec{q}$ is defined as

$$d(\vec{x}, \vec{q}, Q) = (\vec{x} - \vec{q})^T Q(\vec{x} - \vec{q}) \tag{5}$$

MindReader [10] estimates the parameters $(\vec{q}, Q)$ to minimize the total distance of objects in the relevant set $\mathbb{G}$. This can be written as,

$$\min_{\vec{q},Q} \sum_{\vec{x}_i \in \mathbb{G}} v_i d(\vec{x}_i, \vec{q}, Q). \tag{6}$$

Subject to:

$$det(Q) = 1 \tag{7}$$

Consider the ellipsoid $\mathcal{E}$ defined by the estimated parameters $(\vec{q}_{opt}, Q_{opt})$ and radius equal to the largest value of d_{opt} distance for relevant objects. k items now need to be presented to the user to obtain the next set of relevance judgements. The items to be presented are obtained by expanding $\mathcal{E}$ till k database items are enclosed inside the ellipsoid. However, during this expansion, the exclusion of non-relevant objects is not guaranteed.

To ensure that non-relevant objects are not retrieved, we formulate a new optimization problem with additional constraints as follows, (abbreviating $d(\vec{x}, \vec{q}, Q)$ as $d(\vec{x})$)

$$\min_{\vec{q},Q,c} \sum_{\vec{x}_i \in \mathbb{G}} v_i d(\vec{x}_i) \tag{8}$$

Subject to :

$$\begin{aligned}
\forall \vec{x} \in \mathbb{G},\ d(\vec{x}) &\le c && (9)\\
\forall \vec{x} \in \mathbb{B},\ d(\vec{x}) &> c && (10)\\
|\{\vec{x} : \vec{x} \in \mathbb{D},\ d(\vec{x}) \le c\}| &\ge k && (11)\\
c &> 0 && (12)\\
det(Q) &= 1 && (13)
\end{aligned}$$

Let the optimal solution be $(\vec{q}_{opt}, Q_{opt})$ and the optimal distance function be d_{opt}. Consider the ellipsoid $\mathcal{E}$ defined by $(\vec{q}_{opt}, Q_{opt})$ with a radius corresponding to the largest value of d_{opt} for relevant objects (we can also use c_{opt} as the radius for $\mathcal{E}$). Equations 9 and 10 ensure that $\mathcal{E}$ partitions the feature space with all relevant points inside and non-relevant points outside. Equation 11 ensures that $\mathcal{E}$ captures enough objects to present to the user, and along with Equation 10 this also ensures that no non-relevant objects are shown to the user. The minimization of distances of relevant objects ensures that relevant objects are ranked higher than other items in the database.

Whereas the above formulation is sufficient to utilize non-relevant information effectively, a straightforward solution of Equation 8 is difficult to obtain. The formulation is a quadratic optimization problem with quadratic constraints. Considering the quadratic nature of the constraints it is likely that the feasible region for the parameters $(\vec{q}_{opt}, Q_{opt})$ is not convex, leading to numerous local minima. Also constraint Equation 11 involves items from the database. When this constraint is expanded, we get an additional constraint for each database item. This makes the problem very expensive to solve in the current form.

To decrease the computation required, we simplify the above formulation as follows. The problem is first split into two independent subproblems,

Subproblem 1 Find a decision boundary that separates the relevant from non-relevant objects. The boundary should be sufficiently close to the non-relevant objects to maximize the size of the relevant region.

Subproblem 2 Find a distance function that minimizes the total distance of relevant objects.

We approximate the decision boundary by a piecewise linear surface. This reduces computation time and also allows the convexity constraint for the relevant region to be easily incorporated. The second subproblem is the same as MindReaders' formulation and hence their results hold in this case.

It is easy to see that the convex hull ($\mathcal{CH}$) of the relevant points is one of the many possible piecewise linear surfaces that satisfy Equations 9 and 10. To satisfy Equation 11 we need to "expand" the convex hull so as to obtain k database items inside. Note that during expansion we must also ensure that Equations 9 and 10 hold for the new surface. This process is not efficient since items from the database need to be accessed in each expansion step.

2.1 Proposed solution

Rather than using an incremental scheme to refine the decision surface, we create a decision surface that maximizes the size of relevant region. For each non-relevant point ($\vec{b}$) we create a hyperplane ($\mathcal{H}$) normal to the shortest distance vector from $\vec{b}$ to $\mathcal{CH}$. $\mathcal{H}$ is positioned a small distance ($\epsilon > 0$) from $\vec{b}$. Figure 1 illustrates an example. The positive halfspace of each such $\mathcal{H}$ contains the relevant objects and the negative halfspace contains the non-relevant example. The relevant region $\mathcal{R}$ is obtained as the intersection of the positive halfspaces obtained for all non-relevant objects. A distance metric d_{new} estimated using only the relevant objects is then used to present top k of those database items that belong to $\mathcal{R}$. Actual computation of $\mathcal{CH}$ is not necessary, the point in $\mathcal{CH}$ closest to $\vec{b}$ is obtained by solving an optimization problem

[1]The symbols used are defined in Table 1.

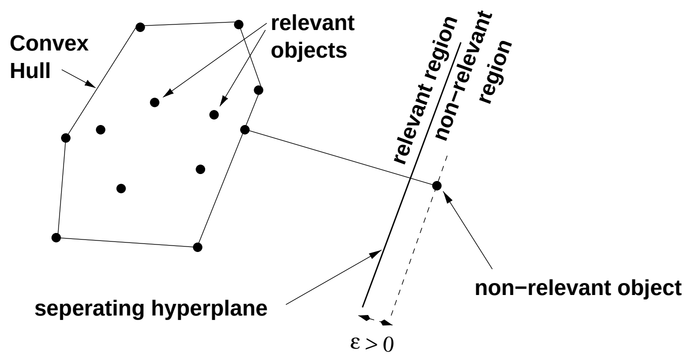


Figure 1: Illustrates feature space partitioning using one non-relevant object, see Section 4.3.1 and Figure 8 for an realistic example.

(Equation 14). Note that in some cases $\mathcal{R}$ may not satisfy Equation 11, i.e, there are fewer than k database items in $\mathcal{R}$. Since this result is the best possible with a piecewise linear surface, a set of size less than k can be presented to the user, in our experiments we obtain the remaining objects by picking top ranked objects from the non-relevant partition.

3 Proposed Algorithm

In this section we describe the proposed method. As described in Section 2.1, the proposed algorithm has two independent steps. The first step is that of obtaining a surface that separates the relevant from non-relevant examples thereby partitioning the feature space into relevant and non-relevant regions, this step is detailed in Section 3.1. The second step involves using relevant examples to estimate a distance metric, this step is detailed in Section 3.2. The final step involves ranking and retrieval of items from the database to present to the user and is detailed in section 3.3.

3.1 Partitioning the feature space

The proposed method separates the relevant and non-relevant objects with a piecewise linear surface. Each of the non-relevant objects is used to create a hyperplane as described in Section 2 and illustrated in Figure 1. For each non-relevant point $\vec{b}_i \in \mathbb{B}$, the closest point $\vec{p}_i$ in the convex hull $\mathcal{CH}$ of the relevant points is computed as follows. Let $G = [\vec{g}_1, \ldots, \vec{g}_{|\mathbb{G}|}]$, columns of G represent the feature vectors of the relevant objects in $\mathbb{G}$. The vector $\vec{p}_i$ can be written as a linear combination of the relevant points as $\vec{p}_i = G\vec{\lambda}$, where $\vec{\lambda}$=$[\lambda_1, \ldots, \lambda_{|\mathbb{G}|}]^T$ and $\sum_j \lambda_j = 1$. The computation of $\vec{p}_i$ can hence be formulated as,

$$\min_{\lambda} \quad |G\vec{\lambda} - \vec{b}_i|^2 \tag{14}$$

subject to

$$\sum_{j=1}^{|\mathbb{G}|} \lambda_j = 1 \tag{15}$$

$$\forall j, \quad \lambda_j \geq 0 \tag{16}$$

Equation 14 is a convex quadratic problem with linear constraints. We use the reduced gradient method outlined in Appendix A to obtain the optimal value of $\vec{\lambda}$. The closest point on the hull $\vec{p}_i$ can now be obtained. The corresponding hyperplane ($\mathcal{H}_i$) can be represented as in Equation 17.

$$\mathcal{H}_i \quad = \quad \{\ \vec{x}\ :\ (\vec{p}_i - \vec{b}_i) \cdot (\vec{x} - \vec{b}_i) = \epsilon\ \} \tag{17}$$

$$\text{where } \vec{a} \cdot \vec{b} = \frac{\vec{a}^T\vec{b}}{|\vec{a}|\ |\vec{b}|} \tag{18}$$

$$\mathcal{H}_i^+ \quad = \quad \{\ \vec{x}\ :\ (\vec{p}_i - \vec{b}_i) \cdot (\vec{x} - \vec{b}_i) > \epsilon\ \} \tag{19}$$

$$\mathcal{H}_i^- \quad = \quad \{\ \vec{x}\ :\ (\vec{p}_i - \vec{b}_i) \cdot (\vec{x} - \vec{b}_i) < \epsilon\ \} \tag{20}$$

Here, ϵ is a small positive constant. In cases where the non-relevant point $\vec{b}_i$ lies inside $\mathcal{CH}$, the closest point $\vec{p}_i$ will equal $\vec{b}_i$, no hyperplanes are constructed for such cases. Each hyperplane $\mathcal{H}_i$ partitions the feature space into a positive halfspace ($\mathcal{H}_i^+$, Equation 19) containing the relevant objects and a negative halfspace ($\mathcal{H}_i^-$, Equation 20) containing the non-relevant point $\vec{b}_i$. The intersection of the positive halfspaces $\mathcal{H}_i^+$ defines the relevant region and the union of negative halfspaces $\mathcal{H}_i^-$ defines the non-relevant region.

3.2 Estimating the similarity metric

The parameters of the distance metric in Equation 5 are estimated using only the relevant objects ($\mathbb{G}$) along with their associated goodness scores ($\vec{v}$). The parameters ($\vec{q}, Q$) are estimated independent of the feature space partitions. This permits the usage of any scheme like MindReader [10] or MARS [18] to estimate the new set of parameters.

The estimates used by MindReader [10] are as follows,

$$\vec{q}_{new} \quad = \quad \frac{\sum_{\vec{x}_i \in \mathbb{G}} v_i \vec{x}_i}{\sum_{\vec{x}_i \in \mathbb{G}} v_i} \tag{21}$$

$$Q_{new} \quad = \quad det(C)^{\frac{1}{n_d}} C^{-1} \tag{22}$$

C is the weighted covariance matrix of the relevant objects, given by

$$C = [c_{jk}], \quad c_{jk} = \sum_{\vec{x}_i \in \mathbb{G}} v_i (x_{ij} - q_j)(x_{ik} - q_k) \tag{23}$$

MARS [18] is a special case of MindReader, where

$$Q_{new} = [Q_{jj}], \quad Q_{jj} \alpha\ \frac{1}{{\sigma_j}^2} \tag{24}$$

```
Input: $\mathbb{G}, \mathbb{B}$ the set of relevant and non-relevant objects.
Output: The next set of $k$ data objects.
$\forall b_i \in \mathbb{B}$ {
    Find $\vec{p}_i$ by solving (14)
    $\mathcal{H}_i$ defined as $(\vec{p}_i - \vec{b}_i) \cdot (\vec{x} - \vec{b}_i) = \epsilon$, (Eqn. 17)
}
Compute ($\vec{q}_{new}$,$Q_{new}$) using Equations 21 and 22.
$\forall \vec{x}_i \in \mathbb{D}$ {
    if ( $(\vec{p}_i - \vec{b}_i) \cdot (\vec{x} - \vec{b}_i) > \epsilon$ , $\forall b_i \in \mathbb{B}$ ){
        /* $x_i$ lies in the relevant region */
        $Dist_i = d(\vec{x}_i, \vec{q}_{new}, Q_{new})$, Equation. 5
    } else {
        /* $x_i$ lies in the non-relevant region */
        $Dist_i = \infty$
    }
}
Return top $k$ objects in $\mathbb{D}$ in increasing order of their distances ($Dist$).
```

Figure 2: Algorithm to retrieve top k relevant database items.

${\sigma_j}^2$ being the variance of the j^{th} feature over all objects in $\mathbb{G}$.

3.3 Ranking and retrieval

Figure 2 illustrates one iteration of the retrieval process. Given a vector $\vec{x}$ representing an item in the database, it is determined if $\vec{x}$ belongs to the relevant region by checking if $\vec{x}$ lies in the positive halfspace for each hyperplane $\mathcal{H}_i$. Distances for items in the relevant region are computed using the estimated distance function d_{new}. k items having the smallest distances are presented to the user for further feedback.

4 Experimental Setup

We tested our relevance feedback algorithm incorporating non-relevant objects on synthetic and real datasets. The experiments demonstrate that our algorithm effectively uses the non-relevant objects to restrict the search space. Many non-relevant objects that would be retrieved by conventional algorithms are rejected, improving the retrieval accuracy.

Four datasets were used for the experiments, one dataset was synthetically generated and the other three were real. In our experiments, feedback is provided by labeling a retrieved object as either relevant or non-relevant. To enable simulation of real users, following assumptions are made. Each object in a database is associated with a class. Note that these class labels are used for the purpose of simulation only, in a real world setting each user has a distinct notion of the set of objects matching her requirements and hence assigning class labels is not useful. For each simulation run, an object class is chosen to be the target class. An object from the target class is used to start the relevance feedback loop. The user's feedback is simulated by labelling objects from the target class as relevant and labelling objects from other classes as non-relevant. We also assume that the user provides feedback on a fixed number of retrieved objects, the objects to be labelled picked randomly from the retrieved set.

As discussed in Section 1, changing the structure of the Q matrix of the quadratic distance function, we obtain different distance metrics. In practice there is a tradeoff between increased flexibility and the robustness with which parameters can be estimated. We chose MARS over the MindReader metric as the number of parameters to be estimated is an order smaller and hence can be more robustly estimated when there are very few ($< n_d$) relevant objects. To demonstrate that it is still feasible to use MindReader for small dimensions, we use the generalized ellipsoid distance metric for experiments with the synthetic 2D dataset.

4.1 Datasets

A brief description of the datasets used for our experiments follows,

- **Synthetic 2D dataset** To make analysis and visualization easier, we synthesized a two-dimensional dataset. This dataset is plotted in Figure 3. There are 400 relevant points and 2500 non-relevant points.

- **Digits dataset** The public domain *PEN dataset*[2] for pen-based recognition of handwritten digits has 10 classes, each class represents a digit. Digits are represented by 16 dimensional attribute vectors. We use 100 objects from each class to create our dataset.

- **Letter recognition** The *Letter Recognition Dataset*[2] has 26 classes corresponding to the letters of the English alphabet. Letters are represented by 16 attributes, we choose 100 objects from each class to constitute the dataset.

- **CAR dataset** This dataset consists of automobile specifications extracted from online car stores. A car is represented by 24 numerical attributes. *Price, torque, weight* are examples of the attributes present. We do not use nominal attributes such as *transmission* (takes values manual or automatic). We use the vehicle type (ex. *Midsize Sedan, Fullsize SUV, Sport Hatchback*) as the class label for a car. We choose 21 vehicle types, each having atleast 25 distinct cars to constitute a dataset of 1270 cars.

[2]UCI Machine Learning Repository, http://www.ics.uci.edu/~mlearn/MLRepository.html

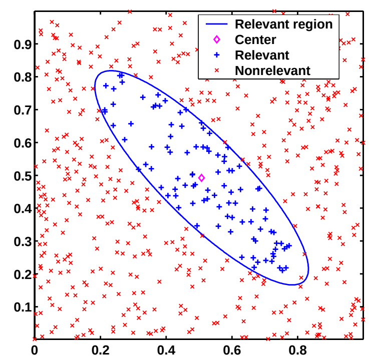


Figure 3: Synthetic 2D dataset

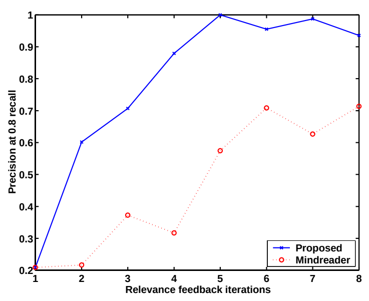


Figure 4: Precision at 0.8 recall in successive iterations of relevance feedback.

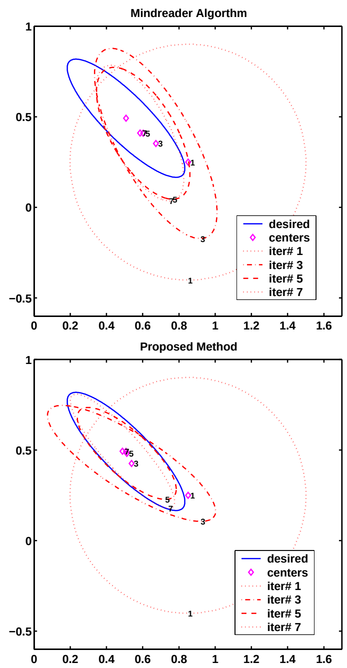


Figure 7: Ellipses represent retrieved regions necessary to achieve 80% recall at different iterations of relevance feedback. For the proposed algorithm the actual retrieved region is obtained by intersecting the ellipse with half-spaces, see Section 4.3.1 and Figure 8.

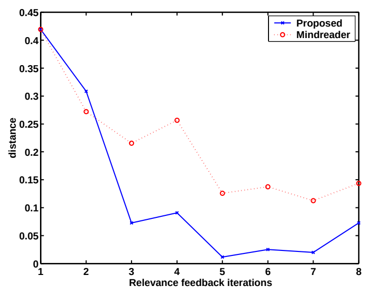


Figure 5: Euclidean distance between centers of estimated and target similarity metrics at successive iterations of relevance feedback.

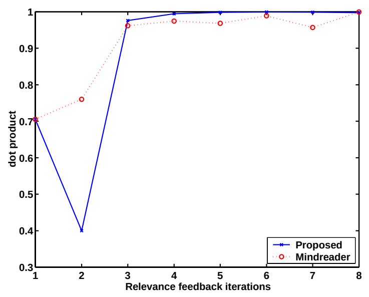


Figure 6: Dot product of major axis of target and estimated similarity metrics at successive iterations of relevance feedback.

4.2 Measuring retrieval effectiveness

Precision and recall are standard metrics used for measuring retrieval effectiveness. Recall and precision are defined as

$$recall = \frac{|relevant \cap retrieved|}{|relevant|} \quad (25)$$

$$precision = \frac{|relevant \cap retrieved|}{|retrieved|} \quad (26)$$

4.3 Results and Discussion

4.3.1 Results for Synthetic 2D dataset

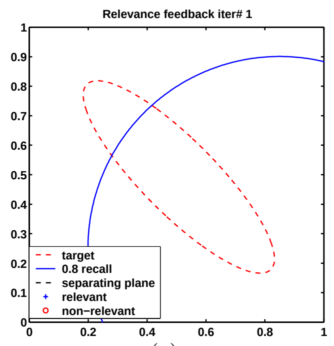
In this section we compare performance of the proposed retrieval algorithm using generalized ellipsoid distance metric with MindReader for the Synthetic 2D dataset. In Figure 4, the precision obtained at 0.8 recall in successive iterations is compared. Using non-relevant objects to restrict the search space significantly improves precision. Figures 7 provide a visual representation of the distance metrics for the two experiments. An ellipse represents the region to be retrieved to achieve 0.8 recall. The target ellipse represents all objects of the target class.

In the case of the proposed algorithm, the actual retrieved region is obtained by intersecting the ellipse with the estimated relevant region. A set of hyperplanes each corresponding to a non-relevant point is determined as detailed in Section 3. The relevant region is then obtained as the intersection of the positive half spaces of each of the hyperplanes (Figure 8). The parameters of the MindReader metric estimated using the relevant objects is used to rank objects in the relevant region. Referring to Figure 8, each separating plane corresponds to a non-relevant object. The unshaded area represents the relevant region obtained as the intersection of the positive half-spaces. The shaded area forms the *non-relevant region.* The solid ellipse represents the distance metric to achieve 0.8 recall and corresponds to the ellipses drawn in Figure 8. The intersection of the interior of the solid ellipse and the unshaded area represents the retrieved region to achieve the specified recall.

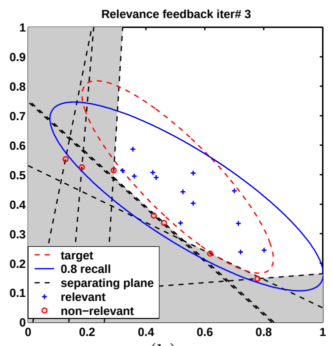
We now compare the convergence of the estimated distance metrics to the target metric. We chose two parameters to evaluate convergence.

1. *Query point movement:* This parameter captures how quickly the centers of the estimated distance metric converge to the target center. Figure 5 plots the Euclidean distance between the estimated centers and the target center for the two algorithms. The faster convergence of the proposed algorithm is evident.

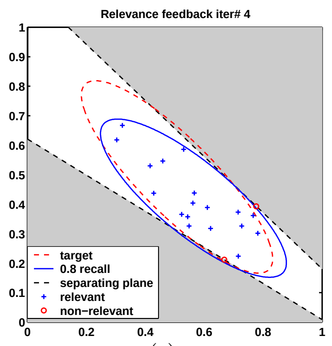
2. *Alignment:* This parameter is used to compare



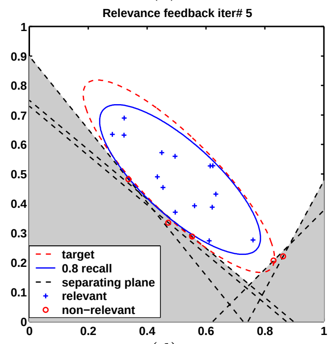
(a)



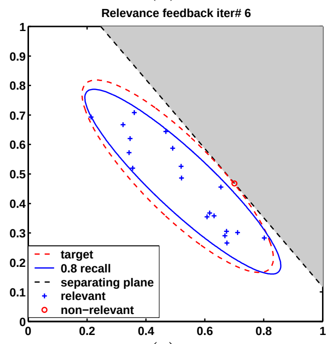
(b)



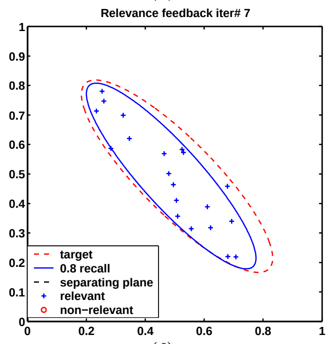
(c)



(d)



(e)



(f)

Figure 8: Restricted search space and similairty metric at different iterations of relevance feedback for 2d dataset.

the speed of convergence of the matrix parameter Q (of the distance metric) to the target Q. The dot product of the major axis of the estimated and the major axis of the target ellipse indicates the degree of alignment of the two metrics. The dot products for the two algorithms are compared in Figure 6. Here again the proposed algorithm shows faster convergence. Note that even though the difference in dot products is quite small the larger difference in angle is manifested in Figure 7.

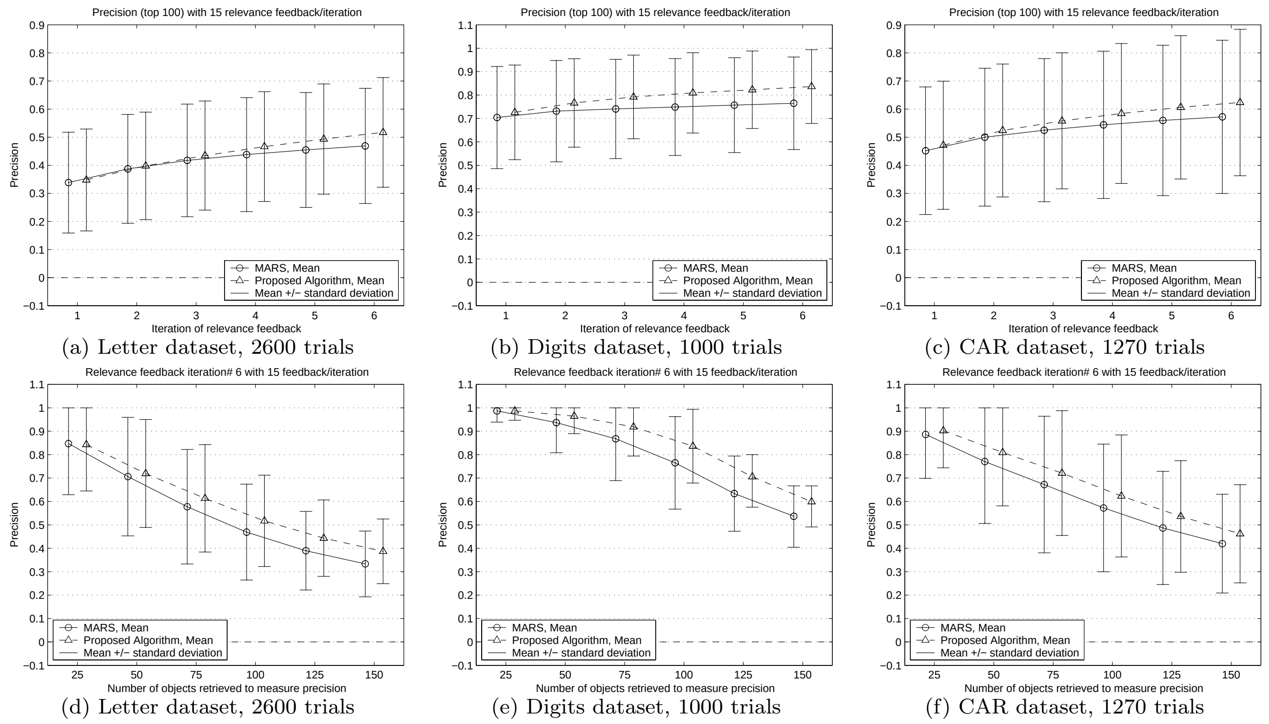
4.3.2 Real Dataset

We now demonstrate the improvement in performance of the proposed algorithm using the weighted Euclidean distance metric over the MARS algorithm on three real datasets.

4.3.3 Improvement in average precision

Figures 9(a),9(b),9(c) plot the average precision for the three datasets. The size of relevance feedback per iteration is 15, the feedback provided is accumulated over successive iterations. Precision is computed from 100 top ranked objects. The precision statistics for a dataset are computed using trials conducted with each class in the dataset as the target class. For a particular target class, trials are repeated using each object in the target class as the starting point for the relevance feedback loop. For the same set of experiments, Figures 9(d),9(e),9(f) plot the average precision after 6 relevance feedback iterations with increasing number of retrieved objects. This is equivalent to computing precision with increasing recall values. The proposed algorithm shows a consistent improvement in precision for all datasets, both with relevance feedback iterations and with large number of retrieved objects.

In Figure 10, we plot the average size of the relevant partition computed by the proposed algorithm. In the Digits dataset with 10 classes, each class constitutes 10% of the dataset, whereas in the Letter dataset each class constitutes $\approx 4\%$ of the whole dataset. The average size of a class in the CAR dataset is $\approx 5\%$. These variations in the size of the target class as fraction of the dataset lead to differences in the size of the estimated relevant regions in the above plot. The accuracy of the estimated relevant region is shown in Figure 11. The fraction of objects from the target class that lie in the estimated relevant region serves as a measure of accuracy of the partioning scheme. It is clear from the above plots that the proposed algorithm effectively utilizes non-relevant objects to accurately constrain the search space. The accuracy of the estimated relevant region improves with relevance feedback iterations demonstrating that the proposed algorithm is able to refine the relevant region to match the true distribution. The relatively poor accuracy in the case of the Letter dataset is due to the the multimodal distribution of the objects in a class, i.e. the objects in a class do not constitute a convex set.



(a) Letter dataset, 2600 trials (b) Digits dataset, 1000 trials (c) CAR dataset, 1270 trials

(d) Letter dataset, 2600 trials (e) Digits dataset, 1000 trials (f) CAR dataset, 1270 trials

Figure 9: Precision for MARS and the proposed algorithm (Section 4.3.3). (a)-(c) plots precision in top 100 retrieved objects at successive relevance feedback iterations, (d)-(f) compares precision measured with increasing number of retrieved objects at the 6th relevance feedback iteration.

4.3.4 Improvement in precision, (0.4 recall, 50 feedback per iteration)

Here, the parameter of interest is the improvement $\mathcal{I}$ in precision achieved by the proposed algorithm over MARS after the same number of relevance feedback iterations with the same starting point.

$$\mathcal{I} = Precision_{Proposed} - Precision_{MARS}. \qquad (27)$$

The range of $\mathcal{I}$ over all experiments is split into a fixed number of bins. Each experiment is assigned to a bin based on the value of $\mathcal{I}$ for the experiment.

In order to accurately quantify the improvement, we plot in Figures 13(a),13(b),13(c) the improvement in precision $\mathcal{I}$ against the precision achieved by MARS ($Precision_{MARS}$) for the same test. In Figure 13(a) we see that the percentage of cases where $\mathcal{I} > 0.3$ reduces with increase in $Precision_{MARS}$. This is expected since the scope for improvement decreases at higher precision values, such cases are shifted to smaller bins resulting in an increase in the % of tests showing smaller improvements ($0-0.3$) at larger values of precision.

4.3.5 Improvement in precision at different recall values across multiple tests

These experiments are similar to those of Section 4.3.4 except that the experiments are repeated for different values for recall. The improvement in precision $\mathcal{I}$ (Equation 27) is binned to create a histogram for each value of recall. Consider Figure 14(c), at larger values of recall there is an increase in percentage of cases where the proposed algorithm gives lower precision ($\mathcal{I} < 0$). Suppose that the objects in the target class are not contained in a convex set, i.e. objects from other classes overlap with the target class. A partition estimated by the proposed algorithm in this case will incorrectly prune some relevant objects causing lower precision than MARS (which is not prevented from retrieving those examples at large recall). The percentage of cases with $\mathcal{I} = 0$ (no improvement) is larger at small recall values (Figure 14(c),14(b)). In most of these cases MARS achieves a precision of 1 and hence no further improvement is possible.

4.3.6 Improvement in precision with different sizes of feedback (fixed recall)

In real systems users typically provide only a small number of relevance judgements at each iteration. Hence a retrieval algorithm must perform consistently with very small sizes of feedback. We performed experiments with number of relevance judgements varying from 5% to 50% of the size of target class. Note that both relevant and non-relevant feedback count as relevance judgements. Plots in Figure 15 show percentage of cases with $\mathcal{I} > 0$ and percentage cases with $\mathcal{I} < 0$.

The improvement achieved by the proposed algorithm at all feedback sizes is clear. Larger feedback

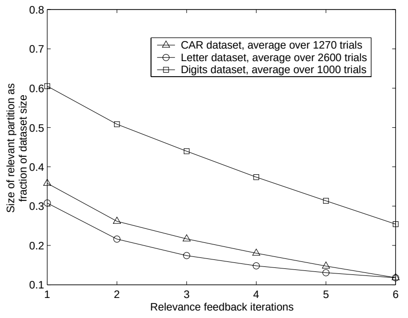


Figure 10: Plots the average size of relevant region for the proposed algorithm with 15 feedback provided per iteration (Section 4.3.3). The fraction of database objects in the relevant partition is used as an estimate for the size of the relevant region.

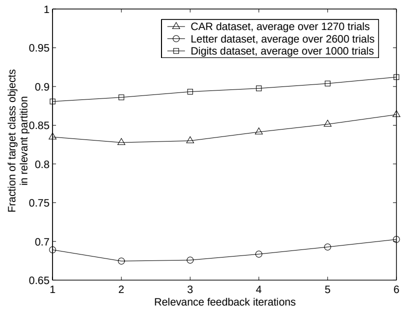


Figure 11: Plots the average accuracy of the relevant region estimated by the proposed algorithm (Section 4.3.3). The fraction of objects from the target class that lie in the estimated relevant region determines the accuracy.

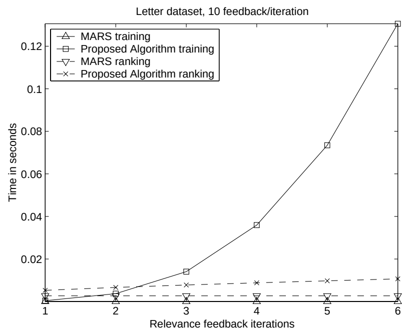
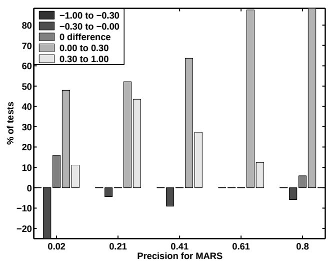
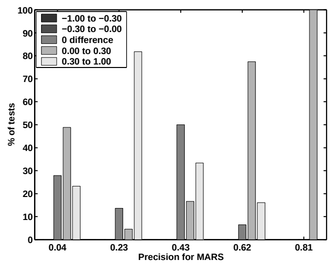


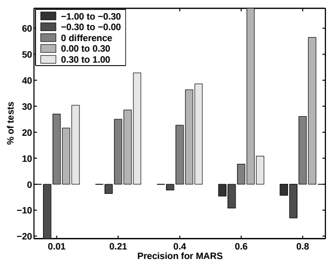
Figure 12: Comparison of average running times for MARS and the proposed algorithm (Section 4.3.7). Training time represents the execution time for parameter estimation. Ranking time is the time required to compute and sort similarity scores for all database objects.



(a) Letter dataset



(b) Digits dataset



(c) CAR dataset

Figure 13: Plots illustrate the histogram of $\mathcal{I}$ for different values of the precision of MARS. Refer to Section 4.3.4.

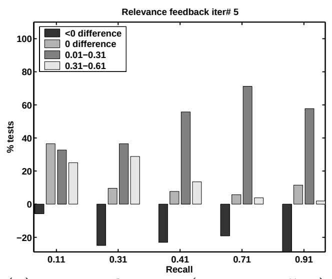
sizes lead to more cases with $\mathcal{I} > 0$. This demonstrates that the proposed algorithm has successfully utilized additional non-relevant information to accurately refine the search space. The percentage of cases with $\mathcal{I} > 0$ increases with iterations of relevance feedback demonstrating the effectiveness proposed algorithm.

4.3.7 Execution time of the proposed algorithm

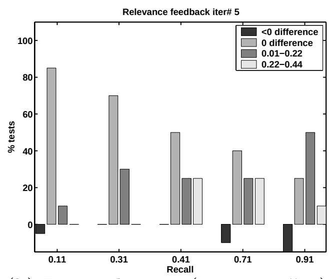
Figure 12 plots the time required for parameter estimation and for ranking and retrieval using the Letter dataset with 10 feedback per iteration on a 1000Mhz, Intel PIII processor. Since feedback is accumulated over iterations, the size of the training set increases with relevance feedback iterations. The MARS algorithm has the least processing requirement and shows no visible performance degradation for both parameter estimation and ranking with larger training sets. The training time for the proposed algorithm grows rapidly with the number of relevant objects in the training set. This is due to the complexity of the constrained optimization procedure. Since an object in the database is evaluated against hyperplanes corresponding to each of the non-relevant objects, the retrieval time grows linearly with the number of non-relevant objects in the training set.

5 Conclusion

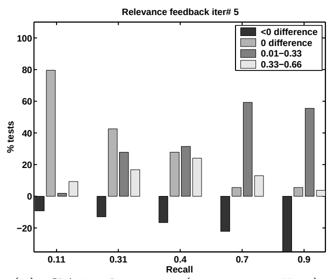
We have proposed a novel technique for improving the accuracy of adaptable similarity based retrieval by incorporating negative relevance judgement, and demonstrated excellent performance and robustness of the proposed scheme with a large number of experiments. The experiments also demonstrate that the proposed scheme improves performance when the size of feedback is small. Ad-hoc techniques have been proposed and studied in the past for using both relevance and non-relevant judgements during similarity based retrieval. Past studies have reported that such methods frequently lead to unfavorable performance, because incompatible information conveyed by the relevance and non-relevance judgements are combined to derive the ranking function. Instead we have proposed a two-step approach, where non-relevant objects in conjunction with relevant objects have been used to define the feasible search space. The ranking function, estimated using only the relevant objects was used to retrieve top k matches from inside the feasible search region. This enables the search to explicitly move away from the non-relevant region, while keeping close to the rel-



(a) Letter dataset (iteration# 5)

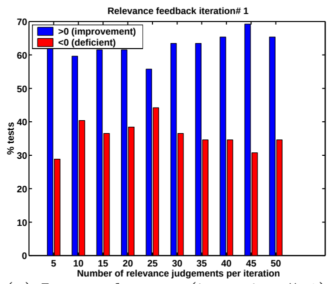


(b) Digits dataset (iteration# 5)

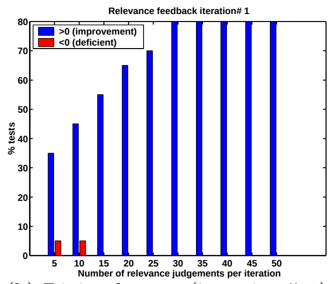


(c) CAR dataset (iteration# 5)

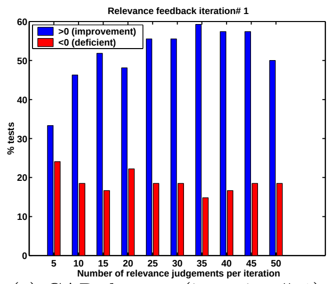
Figure 14: Illustrates the histogram of $\mathcal{I}$ for different recall values. In all the above plots, each bar represents the % of tests where the difference in precision ($\mathcal{I}$) falls in a specified range. Bars corresponding to negative differences shown inverted for clarity.



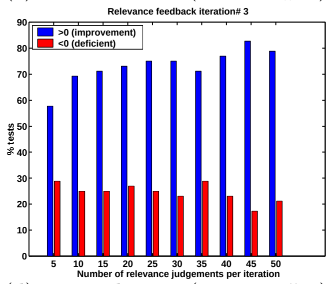
(a) Letter dataset (iteration# 1)



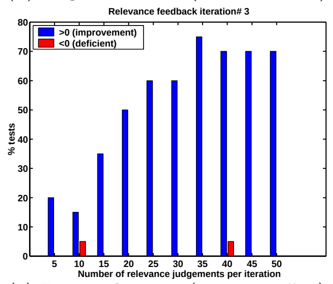
(b) Digits dataset (iteration# 1)



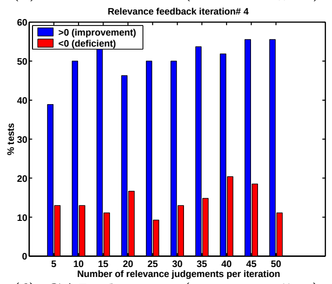
(c) CAR dataset (iteration# 1)



(d) Letter dataset (iteration# 3)



(e) Digits dataset (iteration# 3)



(f) CAR dataset (iteration# 4)

Figure 15: Effect of size of feedback on precision. Plots % cases where the difference $Precision_{Proposed} - Precision_{MARS}$ is postive (improvement), negative (deficient) for different sizes of feedback. Precision computed at 0.4 recall.

evant region. Note that our proposed method does not depend on database-specific parameter tuning. Moreover, it is usable on top of existing schemes, e.g., MindReader and MARS.

Implementation of the proposed ellipsoid query processing with search space pruning on multidimensional indexing structures is of further interest to us for improving the processing speed. Our ongoing work includes dimensionality reduction to address inadequate number of relevance judgements in high-dimensional feature space during a typical search session, and query expansion with non-relevant information so that multimodal (multiple disjoint ellipsoids) information need of a user can be supported.

References

[1] M. Ankerst, B. Braunmuller, H.P. Kriegel, and T. Seidl. Improving adaptable similarity query processing by using approximations. In *Proc. of VLDB*, New York, NY, 1998.

[2] S. Berchtold, and H.-P. Kriegel. S3: Similarity search in CAD database systems. In *Proc. of SIGMOD*, pages 564–567, Phoenix, AZ, 1997.

[3] C. Bohm, B. Braunmuller, F. Krebs, and H.-P. Kriegel. Epsilon grid order: An algorithm for the similarity join on massive high-dimensional data. In *Proc. of SIGMOD*, pages 379–388, Santa Barbara, CA, 2001.

[4] C. Buckley, and G. Salton. Optimization of relevance feedback weights. In *Proc. of SIGIR*, pages 351–357, Seattle, WA, 1995.

[5] A. Singhal, M. Mitra, and C. Buckley. Learning routing queries in a query zone. In *Proc. of SIGIR*, Philadelphia, PA, 1997.

[6] M.D. Dunlop. The effect of accessing non-matching documents on relevance feedback. In *ACM Trans. on Information Systems*, 1997.

[7] R. Brunelli, and O. Mich. Image retrieval by examples. In *IEEE Trans. on Multimedia*, volume 2, number 3, pages 164–171, 2000.

[8] J. Quinlan. *C4.5: Programs for Machine Learning*, San Mateo, CA: Morgan Kaufmann, 1993.

[9] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, volume 2, pages 121–167, 1998.

[10] Y. Ishikawa, R. Subramanya, and C. Faloustos. MindReader: Query databases through multiple examples. In *Proc. of VLDB*, 1998.

[11] M. Flickner, H. Sawhney, W. Niblack, and J. Ashley et al. Query by image and video content: The QBIC system. *IEEE Computer*, pages 23–32, volume 28, number 9, 1994.

[12] M. Iwayama. Relevance feedback with a small number of relevance judgements: Incremental relevance feedback vs. document Clustering. In *Proc. of SIGIR*, pages 10–16, Athens, Greece, 2000.

[13] H.-P. Kriegel, and T. Seidl. Approximation-based similarity search for 3-D surface segments. *GeoInformatica Journal.* Kluwer Academic Publishers, 1998.

[14] D. G. Luenberger. *Introduction to linear and nonlinear programming.* Addison-Wesley Publishing Company, Stanford University, 1973.

[15] C. Meilhac, and C. Nastar. Relevance feedback and category search in image databases. In *Proc. of IEEE Conf. Multimedia Computing Systems*, Florence, Italy, 1999.

[16] D. Rafiei, and A. Mendelzon. Similarity based queries for time-series data. In *Proc. of SIGMOD*, pages 13–25, Phoenix, AZ, 1997.

[17] J. Rocchio. Relevance feedback in information retrieval. *The SMART retrieval system – experiments in Automatic Document Processing*, pages 313–323, 1971.

[18] Y. Rui, T. Huang, and S. Mehrotra. Content-based image retrieval with relevance feedback in MARS. In *Proc. of ICIP*, 1997.

[19] Y. Sakurai, M. Yoshikawa, R. Kataoka, and S. Uemura. Similarity search for adaptive ellipsoid queries using spatial transformation. In *Proc. of VLDB*, Rome, Italy, 2001.

A Reduced gradient algorithm

Consider the minimization problem:

$$\min_{\lambda} (G\vec{\lambda} - \vec{b})^T (G\vec{\lambda} - \vec{b}) \qquad (28)$$

$$\text{Subject to : } \vec{e}^{\,T} \vec{\lambda} = 1 \text{ and } \lambda_i \geq 0, \ \forall i \qquad (29)$$

where $\vec{e} = [1 \dots 1]^T$. The problem can be rewritten as,

$$\min_{\lambda} \frac{1}{2} \vec{\lambda}^T D \vec{\lambda} - H^T \vec{\lambda} \qquad (30)$$

$$\text{Subject to : } \vec{e}^{\,T} \vec{\lambda} = 1 \text{ and } \lambda_i \geq 0, \ \forall i \qquad (31)$$

where $D = 2G^T G$ and $H = 2\vec{b}^{\,T} G$.

Let $n = |\mathbb{G}|$. Specifying any $n-1$ λ_i's uniquely determines the value of the n^{th} λ (using Equation 31). Hence we split $\vec{\lambda}$ into $(\lambda_y, \vec{\lambda}_z)$, where $\vec{\lambda}_z$ is an independent $(n-1)$ sized vector and λ_y is a scalar dependent on λ_z. λ_y is chosen to be one of the strictly positive components of $\vec{\lambda}$. See [14] for a detailed description of the algorithm. The problem now reduces to :

$$\min_{\lambda_y, \vec{\lambda}_z} \frac{1}{2} \vec{\lambda}^T D \vec{\lambda} - H^T \vec{\lambda} \qquad (32)$$

$$\text{Subject to : } \lambda_y + \vec{e}^{\,T} \vec{\lambda}_z = 1 \ , \ \lambda_y \geq 0, \ \vec{\lambda}_z \geq 0 \qquad (33)$$

where $\vec{e} = [11 \dots 1]^T$. We now use a modified steepest descent method using the reduced gradient. The reduced gradient at a point $\lambda = (\lambda_y, \vec{\lambda}_z)$ is obtained as

$$r = \nabla_{\vec{\lambda}_z} f(\lambda_y, \vec{\lambda}_z) - \nabla_{\lambda_y} f(\lambda_y, \vec{\lambda}_z) B^{-1} C \qquad (34)$$

The centroid of the relevant points whose corresponding $\vec{\lambda}$ is given by $\vec{\lambda}_{initial} = [\frac{1}{n} \frac{1}{n} \dots \frac{1}{n}]$, can be used as a feasible starting point for the iterative process.
One iteration of the reduced gradient method is as follows :

1. Compute $r(\vec{\lambda})$ using Equation 34
2. Let $\Delta\lambda_{z_i} = \begin{cases} -r_{z_i} & \text{if } r_{z_i} < 0 \text{ or } \lambda_{z_i} > 0 \\ 0 & \text{otherwise} \end{cases}$
   If $\Delta\vec{\lambda}_z = 0$, then return current $\vec{\lambda}$ as the solution, else find $\Delta\lambda_y = -B^{-1} C \Delta\vec{\lambda}_z$.
3. Find $\alpha_1, \alpha_2, \alpha_3$ so that,
   $$\begin{aligned} \alpha_1 &= \max\{\alpha : \lambda_y + \alpha\Delta\lambda_y \geq 0\} \\ \alpha_2 &= \max\{\alpha : \vec{\lambda}_z + \alpha\Delta\vec{\lambda}_z \geq 0\} \\ \alpha_3 &= \min\{\alpha_1, \alpha_2, \alpha'\} \end{aligned}$$
   where $\alpha' = \frac{-\Delta\vec{\lambda}^T (D\vec{\lambda} - H)}{\Delta\vec{\lambda}^T D \Delta\vec{\lambda}}$.
   Set $\vec{\lambda} = \vec{\lambda} + \alpha_3 \Delta\vec{\lambda}$.
4. If $\alpha_3 < \alpha_1$ then goto step 2 else incorporate λ_y into $\vec{\lambda}_z$ and mark one of the strictly positive λ_z's as λ_y.