

# Toward Consistent Evaluation of Relevance Feedback Approaches in Multimedia Retrieval

Xiangyu Jin<sup>1</sup>, James French<sup>1</sup>, and Jonathan Michel<sup>2</sup>

<sup>1</sup> University of Virginia, Charlottesville VA 22903, USA

<sup>2</sup> Science Applications International Corporation, Charlottesville VA 22911, USA

**Abstract.** Many different communities have conducted research on the efficacy of relevance feedback in multimedia information systems. Unlike text IR, performance evaluation of multimedia IR systems tends to conform to the accepted standards of the community within which the work is conducted. This leads to idiosyncratic performance evaluations and hampers the ability to compare different techniques fairly. In this paper we discuss some of the shortcomings of existing multimedia IR system performance evaluations. We propose a common framework in which to discuss the differing techniques proposed for relevance feedback and we develop a strategy for fairly comparing the relative performance of the techniques.

## 1 Introduction

The information retrieval (IR) task is to find documents relevant to a searcher's information need. Relevance feedback may be offered by an IR system as a mechanism for more effectively capturing a user's information need. Usually relevance feedback is carried out by having the user select positive/negative examples from the current retrieval result to be used to modify the query for the next retrieval iteration.

Research in relevance feedback for multimedia retrieval is being carried out by several different disciplines, for example, computer vision [1, 2], database management [3, 4, 5], information retrieval [6, 7], human computer interaction [8], artificial intelligence, and even psychology. These groups tend to follow different traditions within their communities and, in particular, they often have different standards and approaches to evaluation. Unlike text IR, which is an established area with general agreement upon evaluation protocols such as TREC (Text Retrieval Conference),<sup>1</sup> relevance feedback research in multimedia retrieval still does not have a generally accepted evaluation methodology.

Take content-based image retrieval (CBIR) as an application example. On one hand, different research groups tend to use different image libraries. In the early days of CBIR research, many groups used the COREL<sup>2</sup> image collections [1, 9, 4]. Unfortunately, as pointed out by Müller [10], they tend to use different subsets which makes cross system comparison impossible. More recently, research groups have begun using the test collections of TRECVID<sup>3</sup> [11, 12]. Even if the

<sup>1</sup> <http://trec.nist.gov/>

<sup>2</sup> <http://www.corel.com/>

<sup>3</sup> <http://www-nlpir.nist.gov/projects/trecvid/>

same image collection is used, a different groundtruth can be used for evaluation purposes. Manual judgment (using human labeling[9, 4]), automatic judgment (using a reference retrieval system [1, 2]), and semi-automatic judgment (using system-assisted human labeling [13]) can be used as groundtruth. These different “correct answers” make it extremely difficult to perform cross system comparisons. On the other hand, relevance feedback tests can be executed by machine simulation according to a pre-defined groundtruth (system-oriented) or by real users via some interface (user-oriented). For example, in TRECVID each site can submit its runs based on any feedback approach they executed and usually this process involves human users. However, the user-oriented feedback test methodology is not ideal for cross system comparison purposes since this usually requires a large amount of human effort and a human judge is not as consistent as a machine is (i.e., human user’s experience can vary from person to person and time to time). These issues make cross system comparison almost impossible since too many factors could affect the feedback performance other than the feedback algorithm itself. Only very recently have testbeds which can provide stable human judgment and methods to balance human bias appeared. One such approach is the clarification forms in the TREC HARD track.<sup>4</sup>

In addition to these problems, rank normalization [7], which is considered almost a standard process in text IR, is seldom paid enough attention to in multimedia retrieval. Most current research work in this area neglects this problem and does not perform proper rank normalization when comparing relative performances. These problems make it hard for us to study the relations among different feedback approaches and hard to fairly compare competing techniques under a realistic application environment.

We make two contributions in this paper. First, we briefly summarize the relevance feedback approaches in multimedia retrieval by putting them in a common framework so that each approach can be treated as a special case and their intrinsic relations can be studied. Second, we point out the evaluation problems in previous research and demonstrate how we perform a fair comparison for three typical feedback approaches in large scale test beds (both text and image) which are indicative of real application. We also show that improper evaluation methodology can lead to quite different and even contradictory conclusions.

The rest of the paper is organized as follows. Section 2 provides an overview of several major approaches to relevance feedback in multimedia retrieval. In section 3 we point out several problems in their evaluation. In section 4 we describe our framework and how we perform a fair comparison among these approaches. Sections 5 and 6 discuss our experiments.

## 2 Relevance Feedback in Multimedia Retrieval

### 2.1 Overview

Approaches for relevance feedback in multimedia retrieval often involve a retrieval procedure to handle multiple query points, i.e., multi-query retrieval.

<sup>4</sup> <http://trec.nist.gov/tracks.html>

Many researchers working on multi-query retrieval do not directly connect them to relevance feedback. However, there is strong relation between these two notions. If we are given a solution to multi-query retrieval, we can implement a corresponding relevance feedback process, and vice versa. The process of relevance feedback can be abstracted as the following steps:

1. An initial query set  $S$  gives rise to an initial retrieval result  $L$
2. If the termination condition is met then END
3. Select new feedback examples from  $L$  and put them into  $S$
4. Issue a multi-query retrieval and get a new retrieval result  $L'$ , let  $L = L'$
5. Goto step 2

Our abstraction of the relevance feedback process is a little bit different from the traditional way. There is no independent “refinement” step. The refinement task is seamlessly carried out by the multi-query retrieval in step 4. This is achieved by issuing a new multi-query search whose query both includes the old query points and the new feedback ones. Therefore, in the following, we focus on the solution to multi-query retrieval. We note that our notion is not the same as a multi-modal query. While our queries could be multi-modal, there is no requirement that they be.

In order to simplify our discussion, we restrict our topic to positive feedback examples. We do this for two reasons. First, approaches can be easily extended to negative examples. Second, negative examples may be harder to select in a real user interface since users may find it harder to judge the extent of dissimilarity than similarity.

In a distance based IR system, both the documents and queries can be abstracted as points in some space referred to as document points and query points in this paper. Each pair of points’ distance is defined by some distance function  $D$ . The process of retrieval can be interpreted as getting the document points in some neighborhood of the query points. When there is only a single query point to consider, the query  $q$  to each document  $d$ ’s distance is evaluated by  $D(q, d)$ . The retrieval is a nearest neighbor search process. When multiple query points are provided, we must extend the distance function  $D$  so that it can handle the distance between a query set  $Q$  and a document  $d$ . There are two possible solutions to construct such  $D'$ : combine queries and combine distances.

*Combine Queries Approaches.* The combine queries approach tries to synthesize a single query point from the given query point set and put the synthetic query into the distance calculation. The idea is to create a mapping to map a set of query points  $Q = \{q_1, q_2, \dots, q_T\}$  to a single query  $q$  and define  $D'(Q, d) = D(q, d)$ . The query  $q$  can be created either by selecting a representative from  $Q$  or generating a point which may be not in  $Q$ . The latter approach is usually applied when the space is a vector space, thus the new query point is usually generated via a linear combination of all points in  $Q$ :

$$q = \sum_{i=1}^T w_i * q_i \quad (1)$$

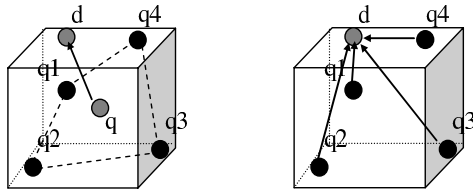
Where  $w_i$  is the corresponding weight for  $q_i$ . In this case,  $q$  must reside in the convex hull of  $Q$ . If all queries are regarded as having the same importance,  $q$  is actually the center of  $Q$ 's convex hull (referred to as query-center in this paper). Query-point-movement [6], re-weighting [1], and MindReader [5] all follow this track. They all use a synthetic single query point to retrieve the data. Figure 1(a) gives an example of using the query center. The query set  $Q$  is composed of four queries  $q_1, q_2, q_3, q_4$ . The combine queries approach first selects a point  $q$  (the query center of  $Q$ ) and uses the distance from  $q$  to  $d$  as the expected distance from  $Q$  to  $d$ .

*Combine Distances Approaches.* The combine distances approach uses an aggregate function [14] to combine the set of distances (each query point's distance to the document point) to a synthetic one. A  $T$ -ary aggregate function  $\varphi$  should satisfy:  $\varphi(x_1, x_2, \dots, x_T) \leq \varphi(x_{1'}, x_{2'}, \dots, x_{T'})$  if  $\forall i(x_i \leq x_{i'})$ . The power mean is usually employed for distance combination. A power mean is a generalized mathematical mean for a data set and is defined over a  $T$ -element set  $X = \{x_i | 1 \leq i \leq T\}$  as:  $mean_\alpha = (1/T \sum_{i=1}^T x_i^\alpha)^{\frac{1}{\alpha}}$ . The new distance function  $D'$  can be regarded as a weighted power mean of distances from individual query points to  $d$ :

$$D'(Q, d) = \left[ \sum_{i=1}^T w_i * D(q_i, d)^\alpha \right]^{\frac{1}{\alpha}} \tag{2}$$

We further define  $D'(Q, d) = 0$  if both  $\alpha < 0$  and  $\exists i(D(q_i, d) = 0)$  and thus avoid possible division by 0. This is a reasonable modification since points in  $Q$  would have 0 distance to themselves. Moreover, in a real implementation, if the combined distance is only used for ranking purpose, we can omit the last power operation and just simply sort in reverse order if  $\alpha < 0$ . Query-expansion in MARS [2] (not the same notion in text retrieval), FALCON [3], and QCluster [4] follow this track. Figure 1(b) shows an example of using arithmetic mean as the aggregate function. Instead of using a query center, the average of distances from all query points to  $d$  is now regarded as the distance from  $Q$  to  $d$ .

In the next section we describe some of the significant research efforts for relevance feedback in multimedia retrieval.



**Fig. 1.** Combine queries VS combine distances.  $q_1, q_2, q_3, q_4$  are four queries in  $Q$ ,  $d$  is a document.

## 2.2 Combine Queries Feedback

The intuition for combining queries is as follows: if the user-interested documents are clustered in the space, we could find the relevant documents by an “ideal” query (the cluster center). In this case, the feedback process can be abstracted as shifting and reshaping a query region in the space to fit the user-interested area. The two tasks are referred to as query-point-movement and reshaping in this paper. The first task aims at shifting the query region toward a proper location and the latter one tries to stretch the query region to fit the user interested area optimally.

The classical query-point-movement is expressed by Rocchio’s formula [6], which is based on the vector space model. The new query point is the previous query point plus some movement toward positive queries and some movement away from negative queries. Therefore query-point-movement provides continuous movement of the query point in the space, thus shifting the location of the query region to a proper place. In our framework, Rocchio method’s implementation as multi-query retrieval is to use a linear combination to combine the queries as in Eqn. (1).

Reshaping approaches all have some assumption on the distance function. Most current research requires that the distance function be a weighted  $L_p$  metric, e.g., MindReader requires squared Euclidian distance. Suppose each document is represented by an  $M$ -dimensional feature vector (if multiple features are employed, just concatenate them to be a single long vector), all dimensions are normalized in the same scale  $0-1$ . Define the general squared Euclidian distance function to be

$$D(q, d) = [q - d]^T P [q - d] \quad (3)$$

where  $P$  is called the distance matrix, which is an  $M \times M$  symmetric matrix satisfying  $\det(P) = 1$ . The query region is a hyper-ellipse in the space. Modifying the distance matrix  $P$  will change the shape of the hyper-ellipse. Therefore, reshaping can be interpreted as a process to refine the distance matrix.

The classical Rocchio method shifts the location of the query region but does not change the shape of it. Therefore it keeps  $P$  as an identity matrix. This may not capture the user’s information need well. Suppose we have 2D database composed of people’s heights and weights. If we want to find people whose weight is around 140 pounds, the user interested region is actually a sharp band along the line  $weight = 140$ . But Rocchio method always forms a circular query region which cannot fit the intended area. In order to remedy this problem, a natural thought is to allow the query region to be an ellipse. Rui *et al.*[1] proposed a standard deviation approach to adjust the distance matrix in their retrieval system MARS. In this work, they assume the distance matrix to be a diagonal matrix (the standard deviation approach is not restricted to squared Euclidian distance). The data on the diagonal are the weights for different feature components. The basic idea is the higher the variance of queries points along an axis, the lower is the importance of this dimension. So the corresponding weight for this dimension should be low. Therefore, the query region is now stretched to an ellipse.

The re-weighting approach regards each dimension to be independent and orthogonal. Therefore, the resulting hyper-ellipse should be aligned to the axes. However, there may exist queries that actually infer some relation between axes. For example, the “diagonal query”, which is presented in [5], infers a linear relation between axes. Still consider the previous 2D weight-height database as an example. If we want to find “good shape” people, the expected region is a band along the diagonal, where weight/height varies within a range. In this case, re-weighting cannot form an optimal ellipse since neither aligns to weight axis nor height axis. In the MindReader system, Ishikawa et al. loosens the restriction on  $P$  further ( $P$  is a symmetric matrix) so that the hyper-ellipse can be arbitrarily shaped in space.

### 2.3 Combine Distances Feedback

Combine query approaches all hold strong assumptions which are hard to achieve in operational systems. For example, the Rocchio method requires the space is vector and reshaping requires that the squared Euclidian distance function is used. To avoid these restrictions, combine distances approaches are deeply explored recently.

The idea of combining the distances of a document to multiple query points can be traced to combining multiple evidence in text retrieval. Several investigators have explored improving retrieval effectiveness by combining multiple information sources, such as different search strategies [15] and different query representations [16]. These combining techniques (referred to as fusion, merging) can result in improved retrieval effectiveness. In our early work [9] we showed if we treat each query point’s retrieval result as an independent information source, the fused result can be regarded as the refined search result. Therefore, we can introduce techniques which proved to be effective in text fusion into multi-query retrieval. For example, we can use rank merge instead of raw similarity merge to avoid the complex normalization issue. We can also truncate each query point’s retrieval result to improve merge efficiency. In [17], some merging techniques for text retrieval are summarized. The most commonly used are COMBSUM, COMBMIN and COMBMAX which are just special forms of the aggregate function in Eqn. (2) where  $\alpha = 1$ ,  $\alpha \rightarrow -\infty$ , and  $\alpha \rightarrow +\infty$  respectively.

The early work to apply combine distances in multimedia retrieval is MARS’s query expansion [2]. Apart from the difference in application, the algorithm of MARS’s query expansion is exactly the same as the COMBSUM approach in Shaw’s paper [17]. In [2], a weighted arithmetic mean (where  $\alpha = 1$  in Eqn. (2)) to combine the distance. They claim their approach can adapt to any arbitrary shape in space, such as an triangular query region. Hence, it is superior to query-point-movement.

Wu et al. [3] proposes a much more sophisticated approach for merging the distance where a general power mean is used (c.f. Eqn. (2)). They further distinguish the situations of  $\alpha > 0$  and  $\alpha < 0$  as fuzzy AND merge and fuzzy OR merge, respectively. For fuzzy AND merge, if a document is judged as dissimilar by any query point, then the document is regarded as dissimilar. On the

contrary, for fuzzy OR merge, if a document is judged as similar by any query point, then the document is regarded as similar. These two categories of merging algorithm behave differently in forming the query region. For fuzzy AND merge, any retrieved document should be close enough to all the query points. If the distance function is metric by triangle inequality this ensures the retrieved documents have small pair-wise distances, i.e., they are clustered in the space. But for fuzzy OR merge, this property doesn't hold. This means generally in a metric space fuzzy AND merge tends to form a continuous region which covers the central region of query points but the fuzzy OR merge can form several disjoint regions. Applying power mean in distance measure is not new to IR. A similar idea can be traced to the extended Boolean IR approach proposed by Salton et al. [18], where a  $p$ -norm model is proposed to make fuzzy judgment toward the degree of document's matching to a query. Other similar ideas can be found in Korfhage's distance metrics [19].

QCluster [4] is a recent work that tries to unite the merit of combine queries and combine distances approaches. First, the feedback documents are clustered into several groups by some clustering algorithm. Then a combine queries approach is applied within each group, so that each group has its query center and an optimal distance function. MindReader [5] is employed for this purpose. These query centers form a new query set  $Q$ , and each query  $q_i$  has an associated distance matrix  $p_i$  with it. Finally, combine distances approach is applied on query set  $Q$  to retrieve documents. A query point  $q_i$  in  $Q$  will evaluate the distance to a document according to its own distance function. From this viewpoint, it is a mixture of combine queries and distances approaches. The major concern of QCluster is to separate intra-cluster and inter-cluster feedback processes, where combine queries approach is applied to intra-cluster feedback and combine distances approach is applied to inter-cluster feedback. This is a very complex approach whose reliability highly depends on the clustering effect.

### 3 Evaluation Problems

Performance evaluation is critical for properly understanding relevance feedback research in multimedia retrieval. However, as we have already noted, the fragmented nature of the research effort across different domain communities leads to evaluations which cannot be compared to one another. Unfortunately, we seldom see continuous evaluation work to test these ideas under a practical environment either by the same group or others. This makes it extremely difficult for us to assess their usability for real applications. We now list several common evaluation problems that often appear in the published literature for multimedia evaluation.

*Problems with the dataset.* This is the most commonly occurring problem and appears in the following guises.

1. An algorithm is proposed based on some assumption. Instead of verifying this assumption on real application data, the algorithm is evaluated against an arbitrarily constructed dataset where the assumption already holds. But the assumption is still left unverified.

2. The experiment is performed on a real dataset, but the dataset is small, low dimensional, or highly structured and is far from the real application environment. Under this “real” dataset, the proposed approach will work well but it is still an open question whether it would work well with a much larger, higher dimensional, and unstructured dataset.
3. The dataset itself has no problem, but the groundtruth is arbitrarily generated and is biased in favor of the proposed approach. This is the most complicated case facing the reader and it is usually hard to find out what is really going on.

For example, MindReader [5] is based on the assumption that “diagonal” queries exist. However, instead of verify this assumption, they perform a comparison between their approach and the MARS re-weighting approach in arbitrarily constructed datasets and a real Montgomery Country dataset. The latter one is very small and a structural 2D map, where points near I-270 are of interest, i.e., a diagonal query exists. The experiments cannot be used to prove MindReader will be more useful than pure query-point-movement in a real multimedia retrieval environment since we do not know whether such diagonal queries would exist. Actually, the diagonal query does not have an intuitive interpretation in a multimedia retrieval environment.

Another example is the MARS re-weighting [1] approach. Re-weighting is based on the assumption that the “ellipse” query is more suitable for a user’s information need than “sphere” query in a multimedia retrieval application. However, instead of verifying this point, [1] generates the groundtruth by arbitrarily constructed hyper-ellipse in the feature space <sup>5</sup> and then show us that their re-weighting finally converges on the “ideal” distance matrix. However, this evaluation cannot answer the question of whether re-weighting can perform better than pure query-point-movement in a real CBIR application. [1] did not provide any comparison between these two approaches. It is unknown how much improvement (or even if there is such improvement) for the re-weighting approach in a real retrieval system. Even if the assumption is correct, we must notice multimedia retrieval systems usually employ very high dimensional features. This makes it very hard to mine the relation among hundreds of dimensions via several dozen feedback examples. It is highly possible that the noise can overwhelm the real user intention and result in a lower performance. A similar problem also exists in [2], where the groundtruth is generated via a similar technique rather than from semantic level user judgments.

*Problem with the comparison.* Another category of problems is relevant to comparison. We further classify them as *no comparison*, *misused comparison*, and *unfair comparison*. In the first case, the author proposes some algorithm without any comparison to others, but gives evaluation scores according to their own experiments, including convergence speed, precision-recall graph, etc. However, as noted earlier because we lack a standard evaluation test bed in this area these

---

<sup>5</sup> This is done by constructing some arbitrary distance matrix  $P$  and then performing real image retrieval to get the retrieval result as the “ideal” result in user’s mind.

scores seldom give the reader an intuitive feeling for how well the proposed approach works. In the second case, the author proposes a new algorithm  $B$  based on some modification of an algorithm  $A$ . But instead of comparing  $B$  with  $A$  it is compared with another poorer algorithm  $C$  and concludes that  $B$ 's performance is much better. In the third case, the evaluation treats feedback examples inconsistently, thus resulting in unfair comparison. Suppose the retrieval result is in the format of a ranked list. Normalization issues arise when we compare the retrieval results directly. On one hand, when we compare different feedback approaches, some feedback approaches (such as fuzzy-OR-merge) will automatically shift those feedback examples to the head of the result list, while others (such as query-point-movement) will not. Comparing them directly is unfair because any approach can shift the feedback examples in post-processing. On the other hand, when we compare different feedback iterations, it is unclear how much of the performance improvement is contributed from the user and how much is from the retrieval system since those training samples also join the evaluation. Rank normalizations, such as "fluid" and "frozen" [7], are the techniques dealing with these problems. Without a rank normalization process, we would exaggerate the improvement of some feedback approaches. Although rank normalization is generally accepted in text IR community, it is often neglected by multimedia retrieval.

The experiments of QCluster [4] are an example of misused comparison. [4] compare the performance of QCluster approach with query-point-movement and query expansion in MARS and conclude that QCluster is the best. However, they should at least compare their work to FALCON's fuzzy-OR merge, which their approach is based on, but not any earlier approach. Therefore, it is very hard to see their contribution in this experiment. In our experiments (c.f. sec. 5 and 6), we show that the COREL database has some attributes which will make any approach similar to fuzzy-OR outperform those similar to fuzzy-AND. Therefore, it is not convincing that merit is gained via such a complex approach.

None of the work listed in section 2 performs rank normalization. The problem is more severe in the case when the relevant set is small. For example, [4] use COREL groundtruth so for each query there are 100 relevant images. This is a small number which makes the accumulated feedback examples take a large portion of the relevant set. Recall will show great improvement by any approach which shifts the feedback examples to the head. FALCON [3] is another example but the problem is less severe since the arbitrarily generated dataset has a large relevant set for each query, so the improvement in recall is relatively objective, because the number of accumulated feedback examples only takes a small portion of the relevant set.

*Impractical parameter settings.* Sometime a paper uses an evaluation methodology which is unlikely to appear in a real application usually by assuming the user to be "diligent." For example, [1] suppose the user would look through the top 1100 retrieval results to find feedback examples. [3] compares feedback iterations after 30 times, but we know a real user would seldom have the patience to do that. [4] and [2] assume the user would feed back all relevant images in the top

100 retrieval result. We note that this is a possible reason why most feedback contribution appears in the first iteration. Because the COREL database only has 100 relevant images for each query, feeding back all relevant images in the top 100 retrieval result may already include most of them.

## 4 Evaluation Framework and Rank Normalization

In the following, we describe the evaluation framework we use to perform comparisons among different relevance feedback approaches fairly, including our implementation of rank-shifting and rank-freezing. Then we give an example of evaluating three most typical feedback approaches under large scale image and text retrieval test beds. Here our purpose is not to make extensive empirical studies to compare these approaches but rather to show that evaluation methodology is so important that by itself it can lead to the appearance of performance improvement when none exists or cause one to draw erroneous and/or contradictory conclusions when comparing systems.

First, we define the following notions and operations:

1. A document list  $L = (L_1, L_2, \dots)$  is a finite, non-duplicated list of document
  - (a) Define  $s(L) = \{L_1, L_2, \dots\}$  is the set of elements in  $L$
  - (b) Define  $len(L) = |s(L)|$  as the length of  $L$
  - (c) Define  $rank(d, L)$  ( $d \in s(L)$ ) return an integer indicating the rank of a document  $d$  in  $L$
  - (d) Define  $top(L, K) = (L_1, L_2, \dots, \min(len(L), K))$  as a truncation to keep top up to  $K$  elements in  $L$
  - (e) Define  $filter(L, S)$  ( $S$  is a document set) as a function to exclude elements in  $s(L) \cap S$  from  $L$
  - (f) Define  $L_1 \circ L_2$  as the operation to concatenate two lists
  - (g) Define  $sort(S, L)$  ( $S \subseteq s(L)$ ) as the function to sort a set  $S$  into a list according to the order of  $L$
  - (h) Define  $insert(d, L, p)$  ( $d \notin s(L)$ ) as an operation to insert a document  $d$  in the  $p$ -th position of a list  $L$
2. Define relevance function  $rel(d)$  as a Boolean function to show whether the document  $d$  is relevant for the current information need
3. Define feedback selection function  $S' = f(L, k, S)$ , which returns a relevant document set  $S'$  from  $s(L) - S$ , which contains upto  $k$  relevant documents, i.e.,  $(S' \subseteq s(L) - S) \wedge (|S'| \leq k) \wedge (\forall d(d \in S' \rightarrow rel(d)))$
4. The evaluation function  $E(L)$  returns a non-negative real number reflect some measure on  $L$ , e.g., precision at 100, average non-interpolated precision.

Now we give our evaluation framework of relevance feedback process:

1. The retrieval system returns  $L$  as the initial retrieval result;  $S = \{\}$
2.  $L = top(L, K)$ , output  $E(L)$
3. if some termination condition met then END.; else  $S = S \cup f(top(L, \hat{k}), k, S)$
4. Issue  $S$  as a multi-query search and get the retrieval result as  $L'$

5. Perform rank normalization on  $L'$  and get the normalized result  $L''$
6.  $L = L''$ ; goto step 2

We offer two possible implementations for rank normalization: rank-shifting and rank-freezing. Rank-shifting moves all feedback examples to the top of the refined retrieval result. Rank-freezing keeps those feedback examples' ranks in the previous retrieval result unchanged in the refined retrieval result, as if they are "frozen" there. These two approaches both "normalize" the performance improvement contributed from the user feedback examples. Rank-shifting makes them equal by maximizing them. Rank-freezing makes them equal by minimizing them. We should note that these two techniques are not necessarily performance order preserving under arbitrary performance measures, although in general they differ only when performance is very similar. We can always give negative examples that would generate different order for some measure. But since the variation is very small, we still claim either technique can be used to compare feedback approaches (due to space limitation, we skip the discussion here). Rank-shifting is relatively easier to implement and rank-freezing is more objective when comparing the performance improvement over iterations. The rank normalization process in step 5 can be implemented either as:

<p><b>Rank-shifting</b></p> $\hat{L}' = filter(L', S);$ $\hat{S} = sort(S, L);$ $L'' = \hat{S} \circ \hat{L}';$	<p>or <b>Rank-freezing</b></p> $\hat{L}' = filter(L', S);$ $\hat{S} = sort(S, L);$ <p>for <math>i = 1</math> to <math>len(\hat{S})</math> do</p> $insert(\hat{S}_i, \hat{L}', rank(\hat{S}_i, L));$
---	---

The retrieval system always outputs a truncated document list of length up to  $K$  for evaluation. Each feedback iteration adds up to  $k$  new documents to the feedback set  $S$ . Then  $S$  is issued as a multi-query search to get a "refined" retrieval result. Finally, evaluation is performed on the truncated "refined" result and if some condition is met we terminate the iterations. There are several parameters need to be decided when implementing a feedback test on a retrieval system.  $K$  is the length of the document list that the retrieval system returns.  $\hat{k}$  is the length of documents that user is supposed to look through to find relevant documents.  $k$  is the maximum number of documents that a user would like to feed back to the retrieval system. Obviously,  $K \geq \hat{k} \geq k$ . Another thing to be decided is the selection strategy used in feedback selection function  $f(L, k, S)$ , which is relevant to the user scenario.

## 5 Experimental Environment

### 5.1 Test Beds

*CBIR test bed.* In this test bed, 3400 color images (34 categories, each with 100 images) from the COREL database are indexed by our basic CBIR system [9]. This CBIR system is based on 7 global visual features and each image is

represented by a vector. In [9], we have shown our basic CBIR system provides a baseline which is competitive to top CBIR systems such as Simplicity [20] in the same test bed. We use our basic CBIR system because combine query approaches can be implemented.<sup>6</sup> The COREL category is regarded as the ground truth, i.e., all the images in an image category are relevant to all the other images in the same category and not relevant to any other images. Six images from each category are randomly chosen as the query images, hence there are 204 queries in total. In [21] we have shown this test bed is a good representative to a much larger scale test bed (composed of 60K images) for relevance feedback experiments.

*Text IR testbed.* In this test bed, Lucene<sup>7</sup>, which is an open source text search engine offered by Apache Jakarta project, is used to index Tipster data used in TREC-3 evaluation (disks 1 and 2). The dataset consists of about 750K documents. TREC topics 151-200 are used for query formulation. These topics have been assessed against the data on disk 1 and 2 and relevance judgments are provided. Since Lucene uses a VSM (Vector Space Model) like scoring algorithm to rank the documents, each document is represented as a vector in its implementation. The data is indexed with a standard analyzer provided by Lucene. The queries are created by a user by reading all these 50 TREC topics<sup>8</sup> One query for each topic. These queries are short and typically 2-5 words in length. We assign these words equal importance and form them into vector queries. This test bed is close to real queries input by a common user, hence simulating a “practical” environment.

## 5.2 Feedback Approaches

We implemented and compared three feedback approaches which typically represent today’s relevance feedback technology in multimedia retrieval:

*Combine queries: Query-Point-Movement (QPM).* In this implementation, we use the classical Rocchio method and give all query points equal importance (i.e., the initial query is weighted the same as the feedback ones). Re-shaping is not considered at this time because the distance function is not a squared Euclidian distance. The old query points and new feedback ones are averaged to get a new query point.

*Combine distances: AND-merge and OR-merge.* In this implementation, each feedback document is issued as a query. In the CBIR test bed, feedback images are directly fed into the CBIR system for they directly support QBE (query-by-example). In the text test bed, where QBE is not directly supported, the document’s representing vector is issued as a query. For efficiency purpose, only 20 most significant components of the representing vector are considered in

<sup>6</sup> We also conducted the same experiments with Simplicity and drew similar conclusions. Due to space limitations, we do not report all the results here.

<sup>7</sup> <http://jakarta.apache.org/lucene/>

<sup>8</sup> We also performed the same experiments using different query sets formulated from the same topics and drew similar conclusions. Due to space limitations, we do not report all the results here.

formulating the query. Retrieval results and previous search result are merged by assigning the same weight for the initial query and each feedback example. We use mid-rank [22] merge instead of merging the raw distance output by the search engine. First, using rank can avoid complex normalization issues arising from the retrieval system itself. Second, in our experience rank merge is competitive in performance to those complex merging approaches which use the raw distances. For AND-merge we choose  $\alpha = 1.0$  and for OR-merge we choose  $\alpha = -1.0$ . For other choices of  $\alpha$  we have similar results.

### 5.3 Other Experimental Settings

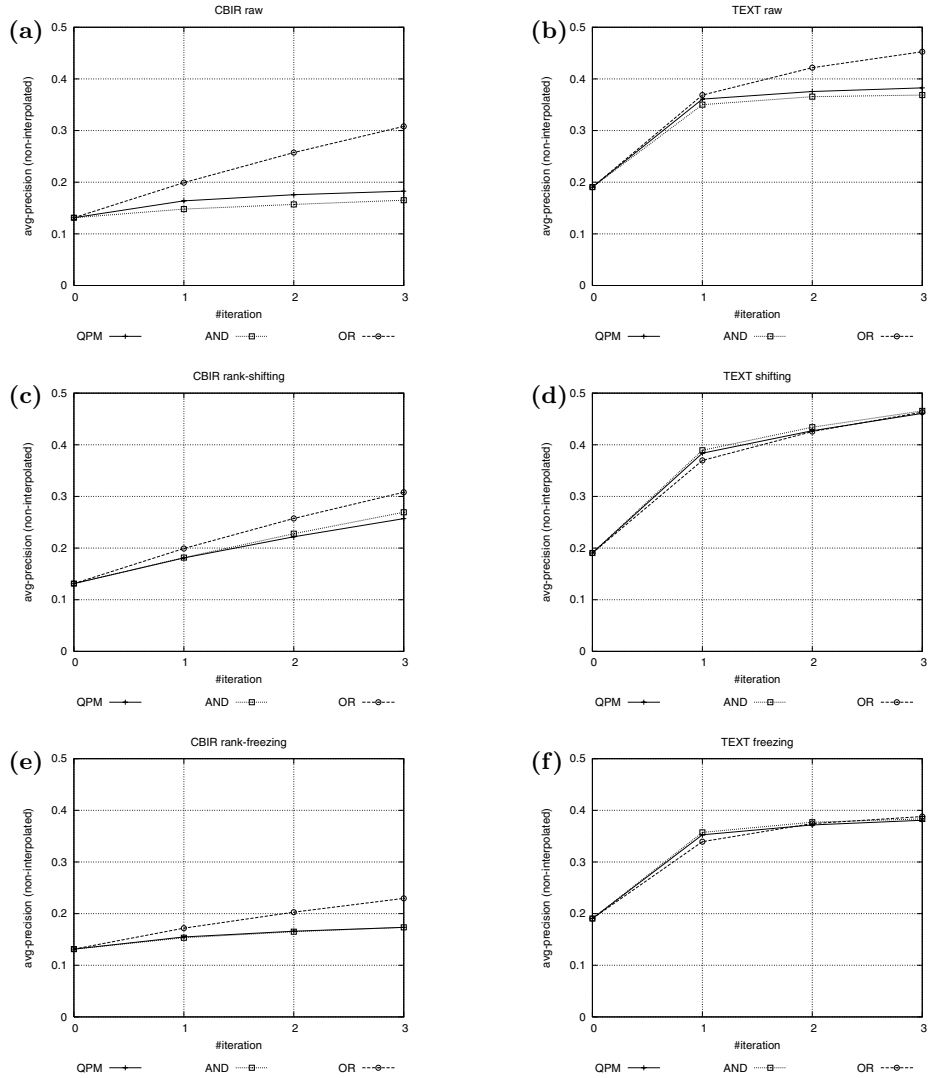
We let  $K = 150$  for the CBIR testbed and  $K = 1000$  for the text-IR testbed. These are commonly used values for evaluation in the corresponding area. We assign  $\hat{k} = 150$  for both test beds because it is a reasonable length for a user to look through to find relevant documents. We also assign  $k = 8$ , for we think it is a reasonable number for a user to feed back into the system. We use a sequential scan for feedback selection. That is, we suppose the user reads at most the top 150 documents and selects the first eight (actually, up to 8) new relevant documents encountered which have not yet been selected in any earlier feedback iteration.

## 6 Experimental Results

The evaluation metric we use is average non-interpolated precision which is a rank sensitive measure. We only compare the performance of the first three feedback iterations. The results are shown in Fig. 2. The left and right columns show the results from the CBIR and TEXT test beds, respectively. The three rows show the results without rank-normalization, with rank-shifting, and with rank-freezing, respectively.

First we compare the difference among relevance feedback approaches. In our test beds, without rank normalization, we would draw the conclusion QPM is slightly better than AND-merge, but OR-merge is much better than the other two, c.f. Fig. 2(a)(b). However, if rank normalization is considered, we would see that the difference is not so large, c.f. Fig. 2(c)-(f). In the CBIR test bed, OR-merge is better than QPM and AND-merge, and the other two are almost the same. While in the TEXT test bed, the three all perform similarly.

The reason why the OR-merge performs better in the CBIR test bed comes from the data distribution of the COREL image collection. In the COREL database, the relevant images (in the same collection) are scattered into several disjoint regions in the perceptual space. For example, for a semantic category flower, there are red, yellow, and white flowers. The OR-merge can form disjoint query regions which fit these areas. But AND-merge and QPM cannot. This makes OR-merge perform better. On the other hand in the text test beds, relevant documents are clustered in some continuous region in space. This makes the three approaches perform quite similarly but OR-merge converges more slowly than the other two approaches initially. In Fig. 2(d)(f), we can see that in the first feedback



**Fig. 2.** Comparisons of relevance feedback approaches with/without rank normalization in image and text retrieval environment

iteration, the OR-merge performance line is a little bit lower than the other two. This shows us there is no generally “better” approach, but different approaches are suitable for different applications. For databases where relevant objects are scattered into small clusters, such as a personal digital photo collections, the OR-merge is more suitable to carry out the task. On the other hand, for databases where such clustering phenomenon is not strongly present, like a text collection, QPM is favored because it converges fast and has a light computational cost. If

the retrieval system is a black box or the combine query approach is hard to implement, the AND-merge may be the best approach to choose.

Second, we analyze the performance improvement over iterations. With rank-shifting, looking at QPM and AND-merge, we would draw the conclusion that the performance is greatly improved during feedback iterations. However, with rank-freezing, we would say that in the CBIR test bed, there is almost no improvement in these two techniques after the first feedback iteration. And it is even small in the first iteration. In this case, the improvement mainly comes from those feedback examples which are included in the evaluation.

Consequently, if our purpose is to compare performance among different feedback approaches, it seems that both rank-freezing and rank-shifting can preserve the correct relative order. Since rank-shifting is much easier to be implemented, it is more suitable under this case. But if we want to compare real performance improvement along with feedback iterations, rank-freezing can give us more objective scores.

## 7 Conclusions

This paper grew out of an effort to understand the performance contribution of several different techniques for incorporating relevance feedback into multimedia information retrieval. Because the techniques came from different domain specialties with very little interaction across communities, the existing work and evaluation was hard to put in context. Consequently, it is difficult to know whether a newly proposed technique does, in fact, make any difference to retrieval effectiveness. We have made an effort here to put the evaluation of relevance feedback techniques in a framework where actual performance can be teased out. We have shown the relationship between multi-query retrieval and relevance feedback and demonstrate a framework in which these systems can be evaluated.

We have also shown very clearly how improper rank normalization can lead to very erroneous conclusions. Rank normalization has been used extensively in text IR but as far as we know, not at all in multimedia evaluation. The incorporation of rank normalization in the retrieval evaluation affords us a better understanding of the retrieval effectiveness of systems employing relevance feedback. This is extremely important if we are to properly understand the behavior of adaptive IR systems.

## References

1. Rui, Y., Huang, T., Mehrotra, S.: Relevance feedback techniques in interactive content-based image retrieval. In: *Storage and Retrieval for Image and Video Databases (SPIE 1998)*. (1998) 25–36
2. Porkaew, K., Ortega, M., Mehrotra, S.: Query reformulation for content based multimedia retrieval in mars. In: *Proc. of ICMCS'99, San Diego, CA, USA (1999)* 747–751
3. Wu, L., Faloutsos, C., Sycara, K., Payne, T.: Falcon: Feedback adaptive loop for content-based retrieval. In: *Proc. of VLDB'00, Cairo, Egypt (2000)* 297–306

4. Kim, D., Chung, C.: Qcluster: Relevance feedback using adaptive clustering for content-based image retrieval. In: Proc. of ACM SIGMOD'03, San Diego, CA, USA (2003) 599–610
5. Ishikawa, Y., Subramanya, R., Faloutsos, C.: MindReader: Querying databases through multiple examples. In: Proc. of VLDB'98. (1998) 218–227
6. Rocchio, J.: Relevance feedback in information retrieval. In Salton, G., ed.: *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall (1971) 313–323
7. Williamson, R.: Does relevance feedback improve document retrieval performance? In: ACM SIGIR'78. (1978) 151–170
8. Liu, W., Dumais, S., Sun, Y., Zhang, H., Czerwinski, M.: Semi-automatic image annotation. In: Proc. of INTERACT'01. (2001) 326–333
9. Jin, X., French, J.: Improving image retrieval effectiveness via multiple queries. In: Proc. of ACM MMDB'03, New Orleans, LA (2003) 86–93
10. Müller, H., Marchand-Maillet, S., Pun, T.: The truth about corel - evaluation in image retrieval. In: CIVR '02. (2002) 38–49
11. Yan, R., Jin, R., Hauptmann, A.: Multimedia search with pseudo-relevance feedback. In: CIVR'03. (2003)
12. Westerveld, T., de Vries, A.P.: Experimental result analysis for a generative probabilistic image retrieval model. In: SIGIR '03. (2003) 135–142
13. Liu, W., Su, Z., Li, S., Y.F.Sun, H.J.Zhang: Performance evaluation protocol for content-based image retrieval algorithms/systems. In: IEEE CVPR Workshop on Empirical Evaluation Methods in Computer Vision. (2001)
14. Fagin, R.: Combining fuzzy information: an overview. In: ACM SIGMOD Record. (2002) 109–118
15. Fox, E., Shaw, J.: Combination of multiple searches. In: Proc. of TREC2. (1994)
16. Belkin, N., Cool, C., Croft, W., Callan, J.: The effect of multiple query representations on information retrieval performance. In: Proc. of ACM SIGIR'03. (1993) 339–346
17. Shaw, J., Fox, E.: Combination of multiple searches. In: Proc. of TREC3. (1995)
18. Salton, G., Fox, E., Wu, H.: Extended boolean information retrieval. In: comm. of the ACM. Volume 26. (1983) 1022–1036
19. Korfhage, R.: *Information Storage and Retrieval*. John Wiley and Sons, New York (1994)
20. Wang, J., Du, Y.: Scalable integrated region-based image retrieval using irm and statistical clustering. In: Proc. of JCDL'01, Roanoke, VA (2001)
21. French, J., Jin, X., Martin, W.: An empirical investigation of the scalability of a multiple viewpoint cbir system. In: Proc. of CIVR'04, Dublin, Ireland (2004) 252–260
22. French, J., Watson, J., Jin, X., Martin, W.: Using multiple image representations to improve the quality of content-based image retrieval. In: Tech. report CS-2003-10, Dept. of Computer Science, Univ. of Virginia. (2003)