Similarity Search Using Multiple Examples in MARS*

Kriengkrai Porkaew[1], Sharad Mehrotra[2],
Michael Ortega[1], and Kaushik Chakrabarti[1]

[1] Department of Computer Science
University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
[2] Department of Information and Computer Science
University of Cailifornia at Irvine, Irvine, CA 92697, USA
{nid,sharad,miki,kaushik}@ics.uci.edu

Abstract. Unlike traditional database management systems, in multimedia databases that support content-based retrieval over multimedia objects, it is difficult for users to express their exact information need directly in the form of a precise query. A typical interface supported by content-based retrieval systems allows users to express their query in the form of examples of objects similar to the ones they wish to retrieve. Such a user interface, however, requires mechanisms to learn the query representation from the examples provided by the user. In our previous work, we proposed a query refinement mechanism in which a query representation is modified by adding new relevant examples based on user feedback. In this paper, we describe query processing mechanisms that can efficiently support query expansion using multidimensional index structures.

1 Introduction

In a content-based multimedia retrieval system, it is difficult for users to specify their information need in a query over the feature sets used to represent the multimedia objects [10, 7, 12]. Motivated by this, recently, many content-based multimedia retrieval systems have explored a *query by example* (QBE) framework for formulating similarity queries over multimedia objects (e.g., QBIC [4], VIRAGE [1], Photobook [9], MARS [6]). In QBE, a user formulates a query by providing examples of objects similar to the one s/he wishes to retrieve. The system converts this into an internal representation based on the features extracted from the input images. However, a user may not initially be able to provide the system with "good" examples of objects that exactly capture their information needs. Furthermore, a user may also not be able to exactly specify the relative

* This work was supported by NSF awards IIS-9734300, and CDA-9624396; in part by the Army Research Laboratory under Cooperative Agreement No. DAAL01-96-2-0003. Michael Ortega is supported in part by CONACYT grant 89061 and MAVIS fellowship.

importance of the different features used to represent the multimedia objects to the query.

To overcome the above limitations, in the *Multimedia Analysis and Retrieval* (MARS) project, we explored techniques that allow users to refine the initial query during the retrieval process using *relevance feedback* [10]. Given an initial query, the system retrieves objects that are most similar to the query. The feedback from the user about the relevance of the retrieved objects is then used to adjust the query representation.

Relevance feedback in MARS serves two purposes as follows. *Query Reweighting* adjusts the relative importance of the different components to the query. It allows the system to learn the user's interpretation of similarity between objects. *Query Modification* changes the underlying representation of the query to incorporate new relevant information from the user's feedback. It overcomes the deficiency of having started from examples that only partially capture the user's information need.

In [11, 12, 7, 10], various models for query reweighting and query modification were explored and compared over diverse multimedia collections. Specifically, two different strategies for query modification have emerged. The first, referred to as *query point movement* (QPM) [7, 11], attempts to move the query representation in the direction where relevant objects are located. At any instance, a query is represented using a single point in each of the feature spaces associated with the multimedia object. In contrast to QPM, in [10] we proposed a *query expansion model* (QEM) in which the query representation is changed by selectively adding new relevant objects (as well as deleting old and less relevant objects). In QEM, the query may consist of multiple points in each feature space.

Our experiments over large image collections illustrated that QEM outperforms QPM in retrieval effectiveness (based on precision/recall measures) [10]. However, in QEM, its potential drawback is that the cost of evaluating the query grows linearly with the number of objects in the query if done naively.

In this paper, we explore efficient strategies to implement QEM that overcome the above overhead. The key is to traverse a multidimensional index structure (e.g., X-tree [2], hybridtree [3], SS-tree [15], etc.) such that best N objects are retrieved from the data collection without having to explicitly execute N nearest neighbor queries for each object in the query representation. We conduct an experimental evaluation of our developed strategies over a large image collection. Our results show that the developed algorithms make QEM an attractive strategy for query modification in content-based multimedia retrieval since it provides better retrieval effectiveness without extensive overhead.

The rest of the paper is developed as follows, Sect. 2 describes the content-based retrieval in MARS. Section 3 describes the proposed approaches to implementing QEM. Section 4 compares the approaches and shows experimental results. Conclusions are given in Sect. 5.

2 Content-Based Retrieval in MARS

This section briefly describes the content-based retrieval mechanism supported in MARS which is characterized by the following models:

Multimedia Object Model: a multimedia object is a collection of features and the functions used to compute the similarity between two objects for each of those features.

Query Model: A query is also a collection of features. In QEM, a query may be represented by more than one instance (point) in each feature space. Furthermore, weights are associated with each feature, as well as, with each instance in the feature representation. These weights signify the relative importance of the component to the query. Figure 1 illustrates the query structure which consists of multiple features f_i and each feature consists of multiple feature instances r_{ij}.

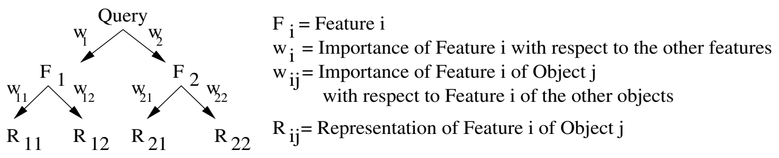

Query
w1
w2
F1
F2
w11
w12
w21
w22
R11
R12
R21
R22
Fi = Feature i
wi = Importance of Feature i with respect to the other features
wij = Importance of Feature i of Object j with respect to Feature i of the other objects
Rij = Representation of Feature i of Object j

Fig. 1. Query Model

Retrieval model: The retrieval model defines how similarity Sim between a query Q and an object O is computed. Similarity is computed hierarchically over the query tree. That is $Sim = \sum_{i=1}^{n} w_i Sim_i$, where $\sum_{i=1}^{n} w_i = 1$, n is the number of features used in the queries, and Sim_i is the similarity between the object and the query based on feature i which is computed as: $Sim_i = \sum_{j=1}^{m} w_{ij} Sim_{ij}$, where $\sum_{j=1}^{m} w_{ij} = 1$, m is the number of feature instances in the feature i in the query, and Sim_{ij} is the similarity between instance j and the object based on feature i. Sim_{ij} is computed using the similarity function determined by the object model. The retrieval process begins with some initial weights associated with nodes at each level of the query tree. For simplicity, initially weights associated with nodes of the same parent are equal.

Refinement Model: The refinement model adjusts the query tree and the similarity functions used at different levels of the tree based on the user's feedback. As discussed in the introduction, the refinement process consists of query reweighting and query modification using query expansion model. The details of the reweighting models, and the query modification models are not critical for the discussion of implementation techniques in this paper and hence omitted due to space restrictions. Details can be found in [10].

3 Query Processing

At each iteration of query refinement, the system returns to the user N objects from the database that have the highest similarity to the current query representation. Instead of ranking each object in the database and then selecting the best N answers, the query is evaluated in a hierarchical bottom up fashion. First, the best few objects based on each feature individually are retrieved. The similarity values of these objects on individual features are then combined (using the weighted summation model) to generate a ranked list of objects based on the entire query. The process continues until the best N matching objects have been retrieved. We next discuss how feature nodes of the query are evaluated, and the answers are combined to obtain the best N answers for the query.

3.1 Evaluating Feature Nodes

In a query tree, let f be a feature node and $r_1, \ldots, r_m$ be the instances (points) under the feature space F. The objective of evaluating the feature node is to retrieve N objects from the database that best match f. We will use the notion of distance instead of similarity since the evaluation of the feature node will use multidimensional indexing mechanisms that are organized based on distances. Let $d_{r_j,x}$ be the distance between r_j and a point x in F and $D_{f,x}$ be the distance between f and x in F where $D_{f,x} = \sum_{j=1}^{m} w_j d_{r_j,x} and \sum_{j=1}^{m} w_j = 1$. Thus, the best N matches to f correspond to objects which are closest to f based on the above definition of distance.

In the following two subsections, we describe two different strategies of evaluating the best N objects for a given feature node. Both strategies assume that the feature space is indexed using a multidimensional data structure that supports range and k-nearest neighbor queries.

Centroid Expansion Search (CES): The idea is to iteratively retrieve next nearest neighbors of some point c (close to $r_1, \ldots, r_m$) in the feature space F using the feature index until the N best matches to f are found.

Let x and y be two objects in the feature space F. x is a better match to f compared to y if and only if $D_{f,x} \leq D_{f,y}$, or equivalently

$$\sum_{j=1}^{m} w_j d_{r_j,x} \leq \sum_{j=1}^{m} w_i d_{r_j,y} \tag{1}$$

Since distance functions are metric, the triangle inequality dictates that $d_{r_j,x} \leq d_{c,x} + d_{c,r_j}$ and $d_{r_j,y} \geq |d_{c,y} - d_{c,r_j}|$. Substituting $d_{r_j,x}$, $d_{r_j,y}$ in (1):

$$\sum_{j=1}^{m} w_j \left(d_{c,x} + d_{c,r_j}\right) \leq \sum_{j=1}^{m} w_j |d_{c,y} - d_{c,r_j}| \tag{2}$$

$$\text{Since } \sum_{j=1}^{m} w_j = 1, \text{ we get:} \quad d_{c,x} + \sum_{j=1}^{m} w_j d_{c,r_j} \leq \sum_{j=1}^{m} w_j |d_{c,y} - d_{c,r_j}| \tag{3}$$

Thus, if (3) holds, then (1) also holds. To remove the absolute value from (3), let $R = \{ r_1, \ldots, r_m \}$, $R_1 = \{ r_j \in R \mid d_{c,r_j} \leq d_{c,y} \}$, and $R_2 = R - R_1 = \{ r_j \in R \mid d_{c,r_j} > d_{c,y} \}$. Replace R_1 and R_2 in (3),

$$d_{c,x} + \sum_{r_j \in R_1} w_j d_{c,r_j} + \sum_{r_j \in R_2} w_j d_{c,r_j}$$

$$\leq \sum_{r_j \in R_1} w_j (d_{c,y} - d_{c,r_j}) + \sum_{r_j \in R_2} w_j (d_{c,r_j} - d_{c,y}) \tag{4}$$

$$d_{c,x} \leq d_{c,y} - 2 \left(\sum_{r_j \in R_2} w_j d_{c,y} + \sum_{r_j \in R_1} w_j d_{c,r_j} \right) \tag{5}$$

$$d_{c,x} \leq d_{c,y} - 2 \sum_{j=1}^{m} w_j \min(d_{c,y}, d_{c,r_j}) \tag{6}$$

Equation (6) provides the strategy to retrieve the best N answers based on the match to f. The strategy works as follows. We find the nearest neighbors to c incrementally. Let $x_1, \ldots, x_P$ be the objects seen so far. We determine the target $M, 1 \leq M \leq P$ such that $D_{c,x_M} \leq D_{c,x_P} - 2\sum_{j=1}^{m} w_j \min(d_{c,x_P}, d_{c,r_j})$. By (6), $D_{f,x_M} \leq D_{f,x_{P+k}}, k = 1, 2, \ldots$. Let $\alpha = \max\{D_{f,x_i} | i = 1, \ldots, M\}$. We then determine the set $\{x_i | i = 1, \ldots, P \wedge D_{f,x_i} \leq \alpha\}$. All such x_i are better matches to f than any object $x_{P+k}, k = 1, 2, \ldots$ and are hence returned. If N objects have not yet been returned, the process continues iteratively by retrieving the next closest object to c (i.e., x_{P+1}) and repeating the above algorithm.

Notice that c can be any point. However, the optimal choice of c minimizes $\sum_{j=1}^{m} w_j d_{c,r_j}$; i.e. c should be the weighted centroid of $r_1, \ldots, r_m$.

This approach does not require any change to the incremental nearest neighbor search algorithm associated with the original multidimensional data structure. However, it does not perform well when query changes dramatically due to the relevance feedback process since the starting centroid is optimal for the original query.

Multiple Expansion Search (MES): In this approach, N nearest neighbor for a feature node f is determined by iteratively retrieving next nearest neighbors for each instance $r_1, \ldots, r_m$ associated with f. Let R_j be the set of ranked results for the instance $r_j, j = 1, \ldots, m$. That is, for all $x \in R_j$ and $y \notin R_j$, $d_{r_j,x} \leq d_{r_j,y}$. Furthermore, let α_j be the maximum distance between r_j and any object in R_j in the feature space; that is, $\alpha_j = \max\{d_{r_j,x} | x \in R_j\}$. R_j contains all objects that are in the range of α_j from r_j. Note that if $y \notin \bigcup_{j=1}^{n} R_j$, then $d_{r_j,y} > \alpha_j$ for all j. So $\sum_{j=1}^{m} w_j d_{r_j,y} > \sum_{j=1}^{m} w_j \alpha_j$, that is, $D_{f,y} > \sum_{j=1}^{m} w_j \alpha_j$. As a result, $y \in \bigcup_{j=1}^{m} R_j$ if $D_{f,y} \leq \sum_{j=1}^{m} w_j \alpha_j$.

Note that if $\bigcup_{j=1}^{m} R_j$ contains at least N objects $x_1, \ldots, x_N$ such that for all x_k, $D_{f,x_k} \leq \sum_{j=1}^{m} w_i \alpha_j$, then it is guaranteed that N best matches to the feature node f are contained in $\bigcup_{j=1}^{m} R_j$. Thus, in order to evaluate the best

N matches to f, MES incrementally evaluates the nearest neighbor for each of the instances $r_1, \ldots, r_n$ thereby increasing the value of at least one α_j in each step, $j = 1, \ldots, m$ until there are at least N objects within $\bigcup_{j=1}^{m} R_j$ for which $D_{f,x_k} \leq \sum_{j=1}^{m} w_i \alpha_j$.

Many different strategies can be used to expand α_js. The optimal strategy determines α_j that minimize $\bigcup_{i=1}^{n} R_i$ since then the least number of objects are explored to retrieve the best N objects based on the match to the feature. We try different strategies for determining α_js and compare them in Sect. 4.

3.2 Evaluating the Query Node

Given the best matching answers for each of the feature nodes $f_1, \ldots, f_n$, the objective in evaluating the query node is to combine the results to determine the best N objects to the overall query. That is, we need to determine the N objects with the least distance to the query, where the distance between object and the query is defined as $D_{Q,x} = \sum_{i=1}^{n} w_i D_{f,x}$ where $\sum_{i=1}^{n} w_i = 1$. MES discussed for the feature node evaluation can also be used for this purpose and is hence not discussed any further.

4 Experiments

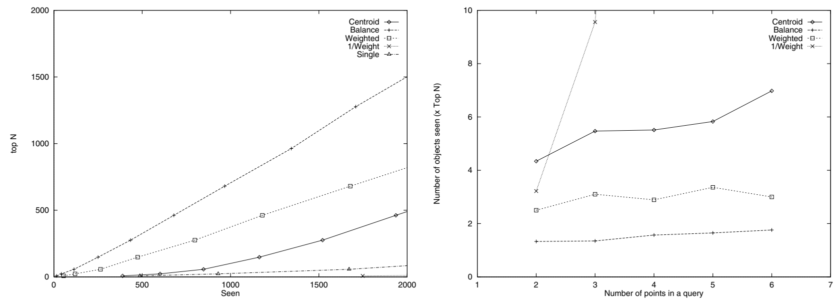
To explore the effectiveness of the algorithms, we performed experiments over a large image dataset (65,000 images) obtained from the Corel collection. Images features used to test the query processing are color histogram [14], color histogram layout [8], color moments [13], and co-occurrence texture [5]. Manhattan distance is used for the first two features and Euclidean distance is used for the last two features [8].

The purposes of this experiment are to compare various approaches we proposed, and to show that QEM can be implemented efficiently. The effectiveness is measured by the number of objects seen before the best N answers are found. A good approach should not need to explore so many objects to guarantee the best N answers and it should not degrade significantly when multiple objects are added to the query.

We performed experiments on CES and MES with various parameters. Specifically, CES searches from the centroid of the query point set. In MES, we explored 4 expansion options as follows. *Single Search* searches only in one of the query points. *Balanced Search* searches on all query points with equal ranges. *Weighted Search* searches on all query points with the ranges proportional to the weights of the query points. *Inverse Weighted Search* searches on all query points with the ranges proportional to the inverse of the weights of the query points.

In the experiments, we do not use any index structure in order to avoid hidden effects caused by the specific index structure. Instead, we simulate a k-nearest neighbor search by scanning the dataset and ranking the answers.

The experimental result shows that single search performs the worst. Intuitively, one may expect the weighted search to perform the best among the four approaches. However, surprisingly, even though the weights are not balanced, the balanced search performed better than any search techniques including the centroid expansion search.



(a) Seen vs best N

(b) Query size vs Seen/best N ratio

Fig. 2. Experimental Result

Figure 2 compares the different approaches and shows that the number of objects in the query representation has very little impact on the balanced search and the weighted search which are the best searches. The reason is simply because the feature space is sparse and the multiple query points are close together due to the query expansion model which selectively adds relevant query points and removes less relevant ones. Other approaches do not perform well since they may have seen best answers but they cannot guarantee that those answers are among the best ones unless they explore further.

5 Conclusions

Content-based multimedia retrieval and multidimensional indexing are among the most active research areas in the past few years. The two research areas are closely related. The supporting index structure has a big impact on the efficiency of the retrieval. In this paper, we proposed algorithms to extend index structures to support complex queries efficiently in the MARS weighted summation retrieval model. We focussed on an efficient implementation to support QEM proposed in [10].

QEM modifies the query by selectively adding new relevant objects to the query (as well as deleting old and less relevant objects). In contrast, QPM modifies the query by moving the query point in the direction of the relevant objects.

Our previous work showed that QEM outperforms QPM in retrieval effectiveness. This paper further illustrates that QEM can be efficiently implemented using multidimensional index structures. As a result, we believe that QEM is a viable approach for query refinement in multimedia content based retrieval.

References

[1] Jeffrey R. Bach, Charles Fuller, Amarnath Gupta, Arun Hampapur, Bradley Horowitz, Rich Humphrey, Ramesh Jain, and Chiao fe Shu. The Virage image search engine: An open framework for image management. In *SPIE Conf. on Vis. Commun. and Image Proc.*, 1996. 68

[2] S. Berchtold, D. A. Keim, and H. P. Kriegel. The x-tree: An index structure for high-dimensional data. In *VLDB*, 1996. 69

[3] Kaushik Chakrabarti and Sharad Mehrotra. High dimensional feature indexing using hybrid trees. In *ICDE*, 1999. 69

[4] M. Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley, Q. Huang, Byron Dom, Monika Gorkani, Jim Hafine, Denis Lee, Dragutin Petkovic, David Steele, and Peter Yanker. Query by image and video content: The QBIC system. *IEEE Computer*, Sep 1995. 68

[5] Robert M. Haralick, K. Shanmugam, and Its'hak Dinstein. Texture features for image classification. *IEEE Trans. on Sys., Man, and Cyb.*, SMC-3(6), 1973. 73

[6] Thomas S. Huang, Sharad Mehrotra, and Kannan Ramchandran. Multimedia analysis and retrieval system (MARS) project. In *Annual Clinic on Library Application of Data Processing - Digital Image Access and Retrieval*, 1996. 68

[7] Yoshiharu Ishikawa, Ravishankar Subramanya, and Christos Faloutsos. Mindreader: Querying databases through multiple examples. In *VLDB*, 1998. 68, 69

[8] Michael Ortega, Yong Rui, Kaushik Chakrabarti, Sharad Mehrotra, and Thomas S. Huang. Supporting similarity queries in MARS. In *ACM Multimedia*, 1997. 73

[9] A. Pentland, R.W. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. *Int'l Journal of Computer Vision*, 18(3), 1996. 68

[10] Kriengkrai Porkaew, Sharad Mehrotra, and Michael Ortega. Query reformulation for content based multimedia retrieval in MARS. In *IEEE Int'l Conf. on Multimedia Computing and Systems*, 1999. 68, 69, 70, 74

[11] Yong Rui, Thomas S. Huang, and Sharad Mehrotra. Content-based image retrieval with relevance feedback in MARS. In *IEEE Int'l Conf. on Image Proc.*, 1997. 69

[12] Yong Rui, Thomas S. Huang, Michael Ortega, and Sharad Mehrotra. Relevance feedback: A power tool for interactive content-based image retrieval. *IEEE Trans. on Circuits and Video Technology*, Sep 1998. 68, 69

[13] Markus Stricker and Markus Orengo. Similarity of color images. In *SPIE Conf. on Vis. Commun. and Image Proc.*, 1995. 73

[14] Michael Swain and Dana Ballard. Color indexing. *Int'l Journal of Computer Vision*, 7(1), 1991. 73

[15] D. White and R. Jain. Similarity indexing with the ss-tree. In *ICDE*, 1995. 69