Learning Similarity Matching in Multimedia Content-Based Retrieval

Joo-Hwee Lim, *Member*, *IEEE Computer Society*, Jian Kang Wu, Sumeet Singh, and Desai Narasimhalu

Abstract—Many multimedia content-based retrieval systems allow query formulation with user setting of relative importance of features (e.g., color, texture, shape, etc) to mimic the user's perception of similarity. However, the systems do not modify their similarity matching functions, which are defined during the system development. In this paper, we present a neural network-based learning algorithm for adapting similarity matching function toward the user's query preference based on his/her relevance feedback. The relevance feedback is given as ranking errors (*misranks*) between the retrieved and desired lists of multimedia objects. The algorithm is demonstrated for facial image retrieval using the NIST Mugshot Identification Database with encouraging results.

Index Terms—Content-based retrieval, image retrieval, multimedia databases, learning, ranking, similarity matching, relevance feedback.

1 Introduction

SIMILARITY matching is crucial in our daily life. We rely on it to categorize the physical and information worlds and to understand new concepts and situations by comparison with existing ones. For multimedia databases wherein exact matching is not appropriate, similarity matching plays a very important role. Since humans originate the queries and have expectations of the retrieval results, it is necessary that the similarity measure implemented in a multimedia content-based retrieval system approximates the users' perception of similarity matching closely [6]. Many multimedia content-based retrieval systems support query by example, with specification of relative weights of features (e.g., color, texture, shape, etc.) to mimic a user's perception of similarity (e.g., [3], [7], [8]). However, these systems do not modify their similarity matching functions, which are determined during system development [5].

In this paper, we present a neural network-based learning algorithm for adapting similarity matching function toward the user's query preference based on his/her relevance feedback. The relevance feedback is given as ranking errors (*misranks*) between the retrieved and desired lists of multimedia objects. More specifically, a user presents a sample query object to the system. The system responds with an ordered sequence of retrieved objects. The user reorders the sequence according to his/her perception of similarity. The misrank information derived from the difference between the retrieved sequence and the desired sequence is fed back as an error signal to the system for learning.

There are two possible types of learning problems. The first type involves short-term learning using only a single misrank sequence. The aim here is to learn a user's perception of similarity from this sequence. The intention is that the system will be able to produce a newly retrieved sequence that is closer to the user's perception of similarity for query refinement. The probable consequence is that objects considered similar to the query by the user but not retrieved previously are now retrieved or ordered more to the front in the retrieved sequence. Objects considered dissimilar to the query by the user but retrieved previously are now not retrieved or ordered more to the rear of the retrieved sequence.

The second type of learning (dubbed long-term learning) is more ambitious. Its objective is to capture a user's perceptual consistency in similarity matching. Learning requires the user's misrank feedback on a number of retrieved sequences corresponding to the sample query objects. The intention is that the system will be able to learn the regularities in a user's perception of similarity for the multimedia objects in the target application domain. After learning, a new query sample will produce a retrieved sequence with less misrank compared to the one without learning.

In any case, learning is user-specific and a user is only required to provide desired ranking on a retrieved list of objects (not the whole database). The similarity matching function is either tuned toward a user's perceived similarity in a user query session or over a number of query sessions. As the learning is based on ranking errors, it is most effective when the domain objects can be ranked easily by the user. The learning algorithm is demonstrated for facial image retrieval using the NIST Mugshot Identification Database with encouraging results.

2 Related Works

Although we are using error-correcting supervised learning for both types of learning mentioned above, it is neither similar to conventional supervised learning for pattern classification [2] nor adaptive Information Retrieval (IR) [1].

To train a pattern classifier, N training samples of the form (X_i, d_i), where X_i is a feature vector and d_i is the desired class label for X_i, are presented to the classifier. For each X_i, the classifier will output a target class label t_i. The difference between d_i and t_i is treated as an error $E_i = d_i - t_i$ which can be used to modify the parameters in the classifier toward a less erroneous classification performance.

In our proposed approach, the training patterns and objectives are quite different. First, the input vectors for learning are not feature vectors, but similarities (or distances) between the feature vectors of two multimedia objects. Second, the output vectors for learning are not desired and target class labels, but misrank information. Last, but not least, the learning process modifies the representation of the similar matching function, which is fundamental in comparing two multimedia objects for retrieval purpose, instead of some discrimination function used in defining the class or decision boundaries.

While learning in multimedia content-based retrieval is new [5], adaptive IR has been an active research area [1]. In fact, Crestani and van Rijsbergen [1] have given a comprehensive and up-to-date survey of past works in adaptive IR in their paper. We shall not repeat the review here, but would like to point out some important observations.

First, in IR, the feature vectors are usually binary vector representation of query and document descriptors (i.e., index terms). But, in multimedia content-based retrieval, the feature vectors are mostly real vectors. They are much richer to capture color, shape, texture, and even spatial information. In fact, due to their complex nature, they are best described in a multilevel structure. Second, learning in IR typically uses relevance feedback directly in the form of class labels (i.e., relevant or irrelevant classes) as in the case of supervised learning for pattern classification. In this paper, errors are derived from ranking information provided by a user. Last, but not least, adaptive IR, as typified by the approach described in [1], attempts to acquire application domain knowledge by learning the association

• *The authors are with Kent Ridge Digital Labs, 21 Heng Mui Kent Terrace, Singapore 119613. E-mail: {joohwee, jainkang, sumeet, desai}@krdl.org.sg.*

Manuscript received 15 Nov. 1997; revised 10 Sept. 1998; accepted 16 Nov. 1999; posted to Digital Library 6 Apr. 2001.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 111123.

between the query descriptors and the document descriptors into a query adaptation or expansion function. The latter can then be used to modify a query for matching (by a "Matcher" [1]) with documents in the database during retrieval. However, in our approach, it is the similarity matching function (i.e., the"Matcher") that gets adapted using misrank feedback.

In short, the uniqueness in our approach surrounds a common notion of relativity: Since the similarity matching function is the function that determines the relative closeness between any two multimedia objects, it is the focus of adaptation. By the same token, the input and supervised information for learning are also computed and presented in relative sense, namely, similarities between pairwise feature vector elements of query and retrieved objects and ranking error of retrieved sequence with respect to the desired sequence, respectively.

3 Learning Similarity Matching Functions

3.1 Feature Measures and Similarity Matching Functions

In multimedia content-based databases, objects are characterized by a set of features, $F^1, F^2, \ldots, F^m$. A multimedia object "facial image," for example, is characterized by facial features such as chin, eyes, nose, and mouth. A feature F^i can be numerically represented by feature measures $\{F_1^i, F_2^i, \ldots\}$. For simplicity, let us just consider one feature and its measures and omit the superscript: $F_1, F_2, \ldots$. Here, F_i is one type of feature measures for a feature. For example, the size of eyes can be a measure of the feature "eye" and color and orientation can be the other two. Each type of feature measure forms a feature vector of, say, m components:

$$F_i = (f_{i1}, f_{i2}, \ldots f_{im})^T. \tag{1}$$

The similarity matching function for object retrieval can be written as

$$S = g(h_1(f_{11}, f_{11}^q; f_{12}, f_{12}^q; \ldots), h_2(f_{21}, f_{21}^q; f_{22}, f_{22}^q; \ldots), \ldots), \tag{2}$$

where h_i is the similarity matching function for feature measure F_i and g is the overall similarity measure. f^2 stands for feature measures of the sample query object q presented to the system to retrieve similar objects from the database. Note that neither h nor g are explicitly defined. h can be distance, correlation, or more complicated functions. g is a function which fuses similarity matching functions on different types of feature measures. The simplest similarity matching function is the linear combination of feature measures:

$$S = w_{11}\Delta f_{11} + w_{12}\Delta f_{12} + \ldots + w_{21}\Delta f_{21} + w_{22}\Delta f_{22} + \ldots, \tag{3}$$

where Δf_{ij} stands for $|f_{ij} - f_{ij}^q|$.

Equation (2) plays a central role in content-based retrieval. We can see, from this equation, that there are two key issues. The first is the effectiveness of feature measures, which is reflected in three levels of granularity: elements in a feature vector (f_{ij}), feature measure (F_i), and multiple feature vectors (F_i^k).

The second important issue is the suitability of the similarity matching functions g and hs for a given application. In other words, the similarity matching function should mimic the domain expert's view on similarity. To achieve that, first, the feature space defined by the feature vectors chosen should be rich enough to represent a human's perceptual space of the objects. In the case of facial retrieval, the salient features which human beings used to compare faces should be captured and well represented in the feature space. Assuming we have selected the right set of features, the remaining issue would be to find the right similarity matching function. To do so, we require appropriate training data set and training criteria.

3.2 Training Data Set and Criteria

In similarity-based retrieval systems, the list of K retrieved objects for a query sample q_i are ranked in descending order of similarity matching values S as computed by (2). Let $t_i = t_{i1}, t_{i2}, \cdots, t_{iK}$ be their ranking. In general, this ranking may not be the same as the ranking of the user based on his/her perception of similarity. Suppose we let $d_i = d_{i1}, d_{i2}, \cdots, d_{iK}$ (a permutation of t_i) be the desired ranking of the retrieved objects for query sample q_i. Then, we can define a *misrank*, $m_{ij} = d_{ij} - t_{ij} (j = 1, 2, \cdots, K)$, as the difference between a desired rank and its actual rank. Consequently, a misrank list for a query sample q_i is $m_i = m_{i1}, m_{i2}, \cdots, m_{iK}$. Note that $\sum_j^K m_{ij} = 0$. In the situation of an *ideal recall*, $m_{ij} = 0, j = 1, 2, \cdots, K$.

For simplicity, and without loss of generality, we can let $t_i = 1, 2, \cdots, K$ for each query q_i (i.e., a rank coincides with its position in the retrieved list). Below are three examples of t_i, d_i, and m_i, where $K = 5, i = 1, 2, 3$.

Example 1.

$$\begin{array}{rcrrrrr} t_1 & : & 1, & 2, & 3, & 4, & 5. \\ d_1 & : & 5, & 4, & 3, & 2, & 1. \\ m_1 & : & 4, & 2, & 0, & -2, & -4. \end{array}$$

Example 2.

$$\begin{array}{rcrrrrr} t_2 & : & 1, & 2, & 3, & 4, & 5. \\ d_2 & : & 3, & 1, & 4, & 5, & 2. \\ m_2 & : & 2, & -1, & 1, & 1, & -3. \end{array}$$

Example 3.

$$\begin{array}{rcrrrrr} t_3 & : & 1, & 2, & 3, & 4, & 5. \\ d_3 & : & 5, & 1, & 2, & 3, & 4. \\ m_3 & : & 4, & -1, & -1, & -1, & -1. \end{array}$$

Each misrank sequence m_i constitutes a training sequence for the system to learn. As we shall see in our experiments, each m_i for a query q_i is a training data set for short-term learning to revise t_i toward d_i (i.e., to achieve $m_i = 0, 0, \cdots, 0$). Collectively, the training sequences $m_i (i = 1, 2, \cdots, N)$ for N queries are used as a training data set for long-term learning to capture a user's consistent perception of similarity in the queries. With the training data set defined in terms of misrank as above, an appropriate training criteria would be to minimize the following *error function* (i.e., cost function) E:

$$E = \frac{1}{2}\sum_i^N \sum_j^K (m_{ij})^2. \tag{4}$$

Note that, in short-term learning, as shown later, i will be a constant for a given query q_i. For long-term learning, *Mean Squared Error* $\widehat{E}$ can also be defined as:

$$\widehat{E} = \frac{E}{N}.$$

As an illustration, if Examples 1 to 3 for three queries are treated as a training data set, then the values for E and $\widehat{E}$ over this training data set are 38 and 12.67, respectively.

3.3 Learning Algorithm

Similar to discriminant function in pattern classification, similarity matching function in content-based retrieval can be complex and nonlinear. As shown in (2), the similarity matching function should

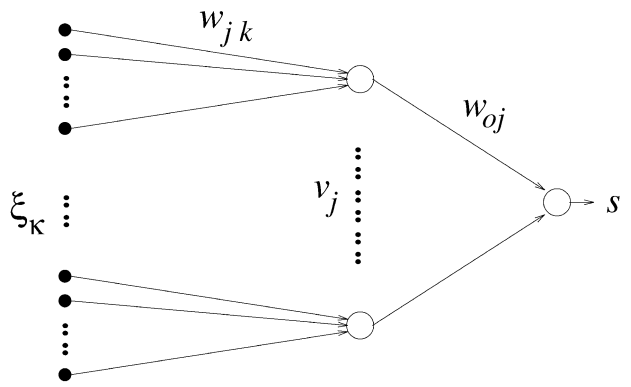

Fig. 1. A multilevel similarity matching function.

initially consist of a sufficiently rich set of primitives (i.e., h_i) before any customization or learning can take place. More often than not, the similarity matching function is multilevel (or hierarchical), where similarity matching at level k is defined in terms of similarity matching at level $k+1$ until the most primitive distance measures or correlation metrics are used at the leaf level. The multilevel structure reflects the application domain knowledge used in characterizing the multimedia objects in terms of features, feature measures, and feature vectors, etc. An illustration of such a similarity matching tree is shown in Fig. 1.

To allow learning in such a similarity matching function and to derive a learning rule for each level uniformly from a global error function, we can treat the similarity matching tree as a feedforward neural network and apply the Backpropagation (BP) learning rule for multilayer perceptrons, ([4] p. 117). However, the BP learning rule cannot be applied directly as the error function in our case is different.

In standard BP learning, the error measure or cost function is defined as ([4],

$$E = \frac{1}{2}\sum_{\mu}(\zeta^{\mu} - s^{\mu})^2, \tag{5}$$

where ζ^{μ} is the desired value (i.e., supervised label) of the output node for the μth training pattern and s^{μ} is the actual value of the output node for the μth training pattern which is defined as (assuming only one hidden layer as in Fig. 1:

$$\begin{aligned} s^{\mu} &= g\left(\sum_{j} w_{oj} v_j^{\mu}\right) \\ &= g\left(\sum_{j}\left\{ w_{oj} g\left(\sum_{k} w_{jk}\xi_k^{\mu}\right)\right\}\right) \end{aligned} \tag{6}$$

where μ indexes the μth training pattern, v_j^{μ} is the output of the jth hidden node, w_{oj} is the weight connecting the jth hidden node to the output node, ξ_k^{μ} is the kth element of the input vector $\vec{\xi}^{\mu}$, w_{jk} is the weight connecting the kth input node to the jth hidden node, and g is a nonlinear transfer function at the hidden and output nodes.

In our case, the user does not provide the relevance feedback as the desired value for the output node directly. Instead, based on the actual outputs s^{μ}, which are the similarity matching values between the retrieved objects and the query object, the system produces a retrieved sequence in descending order of s^{μ}. The user rearranges the relative positions of the retrieved sequence into a desired sequence. There is no information about the desired values for each s^{μ}, though their desired relative ranking information is available. Thus, we adopt the error function defined earlier on

$$E = \frac{1}{2}\sum_{\mu}(d^{\mu} - t^{\mu})^2,$$

where μ indexes the μth training pattern, d^{μ} is the desired rank given by the user, t^{μ} is the rank actually obtained by the system.

Now, using gradient descent as in, BP learning rule and the chain rule, we have

$$\begin{aligned} \Delta w &= -\eta\frac{\partial E}{\partial w} = -\eta\frac{\partial E}{\partial s}\cdot\frac{\partial s}{\partial w} \\ &= -\eta\sum_{\mu}\frac{\partial E}{\partial s}\Big|_{\cdot s^{\mu}}\cdot\frac{\partial s}{\partial w}|_{s^{\mu}} \end{aligned}. \tag{7}$$

and

$$\frac{\partial E}{\partial s}|_{s^{\mu}} = -(d^{\mu} - t^{\mu})\frac{\partial t}{\partial s}|_{s^{\mu}}. \tag{8}$$

Now, since t^u is the rank of a retrieved multimedia object, it is integral and, hence, discrete. Therefore, t is not differentiable in general. Without further information, we assume that ∂t is inversely proportional to ∂s since the obtained rank of a retrieved object will always improve (i.e., reduce in value) with an increase in its similarity value s. Hence, we heuristically replace the term $\frac{\partial t}{\partial s}$ in (8) with a negative number $-\kappa$ (κ is a positive constant). Then, (7) becomes:

$$\Delta w = -\kappa\eta\sum_{\mu} m^{\mu}\frac{\partial s}{\partial w}|_{s^{\mu}} \tag{9}$$

since $m^{\mu} = (d^{\mu} - t^{\mu})$, the misrank for the μth training pattern. Combining all the constants into one and simply call it η, we get

$$\Delta w = -\eta\sum_{\mu} m^{\mu}\frac{\partial s}{\partial w}|_{s^{\mu}}, \tag{10}$$

where η, the learning rate, is a small positive number.

More precisely,

$$\Delta w_{oj} = -\eta\sum_{\mu} m^{\mu}\, g'(h^{\mu})\, v_j^{\mu} \quad \text{where} \quad h^{\mu} = \sum_{j} w_{oj}\, v_j^{\mu} \tag{11}$$

and

$$\Delta w_{jk} = -\eta\sum_{\mu} m^{\mu}\, g'(h^{\mu})\, w_{oj}\, g'(h_j^{\mu})\, \xi_k^{\mu} \text{ where } h_j^{\mu} = \sum_{k} w_{jk}\,\xi_j^{\mu}. \tag{12}$$

In the experiments we conducted, we have scaled m^u by $\tanh(m^u)$ to fall within $[-1, 1]$.

4 Experimental Results

4.1 Data Set and Feature Extraction

The data set for our experiments is comprised of face images taken from the NIST Special Database 18: Mugshot Identification Database (MID). The details of MID can be found at http://www.nist.gov/. Only those candidates who have three or more facial photographs (images) have been selected for our data set. The facial images of such a candidate, taking one of the images as a query sample, constitute a training sequence. We have 130 such training sequences in our data set.

Every facial image is marked with a set of 17 points called landmarks. These points are a subset of anthropometric landmarks

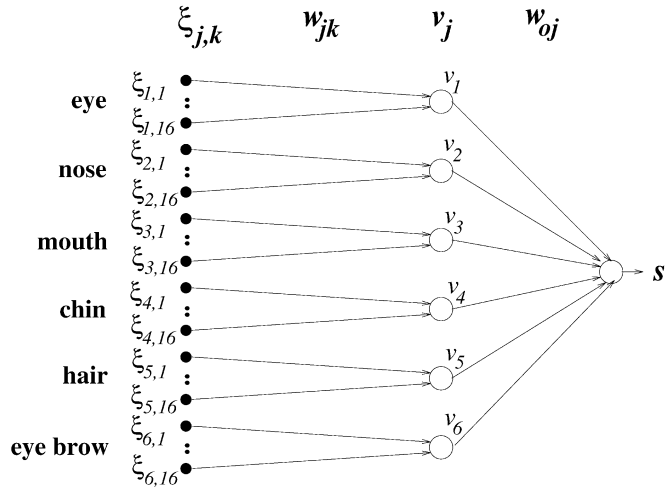

Fig. 2. A multilevel similarity matching function for face retrieval.

of the face. They identify salient points of the face like corners of eyes, tip of nose, chin, etc. Using these landmarks as anchor points, six different regions each containing one of the six facial features namely, eye, nose, mouth, eyebrow, hair, and chin, are extracted from each image. These six regions are then individually subject to principle component analysis (K-L transform) and 16 components corresponding to the 16 largest eigenvalues are extracted. Hence, in all, we have 96 features.

Let the features be represented by $f_{j,k}$, where $j \in \{1, 2 \cdots, 6\}$ represents one of the facial features and $k \in \{1, 2 \cdots, 16\}$ represents one of the 16 principle components. Each image is represented by its corresponding feature vector $\vec{f}$. Then, the input $\vec{\xi}^{u,v}$ to a multilevel similar matching function (which can be viewed as a feedforward multilayer neural network as shown in Fig. 2), is given by

$$\begin{aligned} \xi_{j,k}^{u,v} &= 1 - \left| \frac{z_{j,k}^{u,v}}{\lambda_j} \right| \qquad and \\ z_{j,k}^{u,v} &= f_{j,k}^{u} - f_{j,k}^{v}, \end{aligned} \tag{13}$$

where u and v represent the two images which are to be compared for similarity and λ_j is a normalizing constant for facial feature j given by

$$\lambda_j = \max\Big(|\max(\vec{z}_j^{u,v})|, |\min(\vec{z}_j^{u,v})| \Big) \tag{14}$$

taken over all pairs (u, v) where $\vec{z}_j^{u,v}$ is the vector $\{z_{j,1}, z_{j,2}, \cdots, z_{j,16}\}$. Note that $\vec{\xi}^{u,v}$ is a measure of similarity between images u and v.

4.2 Short-Term Learning

This set of experiments concerns short-term learning. Each of the 130 training sequences is a training data set. Using the first facial image in such a training sequence as a query sample, the system ranks the facial images in the training sequence according to descending similarity matching values. Desired ranking on the training sequence is given by a human user to the system to compute the misranks and, thus, the error function. Using the learning rule in (10), the system modifies its similarity matching function to tune its rankings toward that fed back by the user. The experiment is repeated for each of the 130 training data sets (sequences). The results reported here are averages over the 130 training data sets and each set was run 20 times with different initial weights for the neural network.

TABLE 1
Results of Short-Term Learning of Similarity Matching Function

	$\widehat{E}$
Before learning	0.400
After learning	0

TABLE 2
Results (on Training Set) of Long-Term Learning of Similarity Matching Function

	partly connected	fully connected
	$\widehat{E}$	$\widehat{E}$
Before learning	0.180	0.180
After learning	0.007	0.007

Table 1 shows the experimental results of short-term learning using (10). The neural network architecture parallels the one shown in Fig. 2. The table shows that the mean squared error $\widehat{E}$ in each of the 130 training data sets is corrected after learning. The average number of training iterations is 25.9 and the learning rate $\eta = 0.008$.

4.3 Long-Term Learning

4.3.1 Learning Capability

In these experiments, all the 130 training sequences constitute a training data set. Similar to the query and feedback procedure as described in short-term learning, each training sequence has an associated misrank sequence. To capture a user's perceptual consistency in similarity, the same user is asked to provide desired sequences for all the 130 training sequences. The error function, summed over all the misrank sequences, is used for learning. Using the learning rule in (10), the system looks across all the training sequences and modifies its similarity matching function to reduce the error and thus improve the rankings.

Table 2 shows the results of long-term learning using (10) averaged over 20 runs (with different neural network initial weights). The table shows the mean squared error $\widehat{E}$ before and after learning for two different neural network architectures. The "partly connected" label indicates the network architecture of Fig. 2 and "fully connected" means full connection is allowed between the input and hidden layers. While the partly connected network architecture incorporates the domain knowledge of the problem in the form of a multilevel similarity matching tree, a fully connected network is treated as a black box function mapping. The average numbers of training

TABLE 3
Results (on Test Set) of Long-Term Learning of Similarity Matching Function

	partly connected	fully connected
	$\widehat{E}$	$\widehat{E}$
Before learning	0.178	0.180
After learning	0.132	0.130

iterations are 837.0 (partly connected) and 258.5 (fully connected), respectively. The learning rates η are set as 0.01 (partly connected) and 0.005 (fully connected), respectively. The stopping condition for training is $\widehat{E} \leq 0.01$

The results in Table 2 show that long-term learning based on (10) is able to reduce the misrank errors in the training data set significantly by mimicking the similarity matching in the training data set.

4.3.2 Generalization Capability

In this set of experiments, our objective is to apply the learning algorithm (10) on part of the 130 training sequences and test the retrieval error on the rest of the training sequences. Since our data set is relatively small, we have used the leave-one-out method to evaluate how well our learning algorithm generalizes. One hundred thirty experiments are conducted with partly connected and fully connected neural network architectures as described above. In each experiment, a unique sequence is chosen as a test data set and the other 129 sequences as the training data set. Table 3 shows the mean squared error $\widehat{E}$ before and after learning on the test data set averaged over 130 runs. The average numbers of training iterations are 983.2 (partly connected) and 385.2 (fully connected), respectively. The learning rates η are set as 0.01 (partly connected) and 0.005 (fully connected), respectively. The stopping condition for training is $MS\widehat{E} \leq 0.01$.

The results in Table 3 show that the errors on unseen queries after long-term learning of similarity matching function from the training set have been reduced by more than 25 percent (25.8 percent for partly connected and 27.8 percent for fully connected).

5 Conclusions and Future Enhancements

In this paper, we have pointed out the importance of mimicking human perception of similarity in multimedia content-based retrieval and proposed a learning algorithm that adapts similarity matching function from ranking errors fed back by the user. The learning algorithm is demonstrated on facial retrieval with encouraging results for both short-term and long-term contexts. This may prompt us to build a *personalized* multimedia search engine that can be customized to a user's perceptual preference.

In the future, we will continue to improve the learning algorithm and the richness of feature set. More data needs to be collected for further experimentation.

References

[1] F. Crestani and C.J. van Rijsbergen, "A Model for Adaptive Information Retrieval," *J. Intelligent Information Systems*, vol. 8, pp. 29-56, 1997.
[2] R.O. Duda and P.E. Hart, *Pattern Classification and Scence Analysis.* John Wiley & Sons, 1973.
[3] M. Flickner et al., "Query by Image and Video Content: The QBIC System," *Computer*, vol. 28, no. 9, pp. 23-32, Sept. 1995.
[4] J. Hertz, A. Krogh, and R.G. Palmer, *Introduction to the Theory of Neural Computation.* Addison-Wesley, 1991.
[5] R. Jain, "Infoscopes: Multimedia Iinformation Systems," *Multimedia Systems and Techniques.* B. Furht, ed., pp. 217-253, Kluwer Academic Publishers, 1996
[6] S. Santini and R. Jain, "Similarity Matching," *Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 9, pp. 871-883, Sept. 1999.
[7] *Virage*, http://www. virage. com.
[8] J.K. Wu et al. "Content-Based Retrieval for Trademark Registration," *Multimedia Tools and Applications*, vol. 3, no. 3, pp. 245-267, 1996.