Learning Similarity for Texture Image Retrieval

Guodong Guo, Stan Z. Li, and Kap Luk Chan

School of EEE, Nanyang Technological University
Nanyang Avenue, Singapore 639798
egdguo@ntu.edu.sg

Abstract. A novel algorithm is proposed to learn pattern similarities for texture image retrieval. Similar patterns in different texture classes are grouped into a cluster in the feature space. Each cluster is isolated from others by an enclosed boundary, which is represented by several support vectors and their weights obtained from a statistical learning algorithm called support vector machine (SVM). The signed distance of a pattern to the boundary is used to measure its similarity. Furthermore, the patterns of different classes within each cluster are separated by several sub-boundaries, which are also learned by the SVMs. The signed distances of the similar patterns to a particular sub-boundary associated with the query image are used for ranking these patterns. Experimental results on the Brodatz texture database indicate that the new method performs significantly better than the traditional Euclidean distance based approach.

Keywords: Image indexing, learning pattern similarity, boundary distance metric, support vector machines.

1 Introduction

Image content based retrieval is emerging as an important research area with application to digital libraries and multimedia databases [9] [8] [10] [12]. Texture, as a primitive visual cue, has been studied for over twenty years. Various techniques have been developed for texture segmentation, classification, synthesis, and so on. Recently, texture analysis has made a significant contribution to the area of content based retrieval in large image and video databases. Using texture as a visual feature, one can query a database to retrieve similar patterns based on textural properties in the images.

In conventional approach, the Euclidean or Mahalanobis distances [8] between the images in the database and the query image are calculated and used for ranking. The smaller the distance, the more similar the pattern to the query. But this kind of metric has some problems in practice. The similarity measure based on the nearest neighbor criterion in the feature space is unsuitable in many cases. This is particular true when the image features correspond to low level image attributes such as texture, color, or shape. This problem can be illustrated in Fig. 1 (a), where a number of 2-D features from three different image clusters are shown. The retrieval results corresponding to query patterns “a” and “b” are

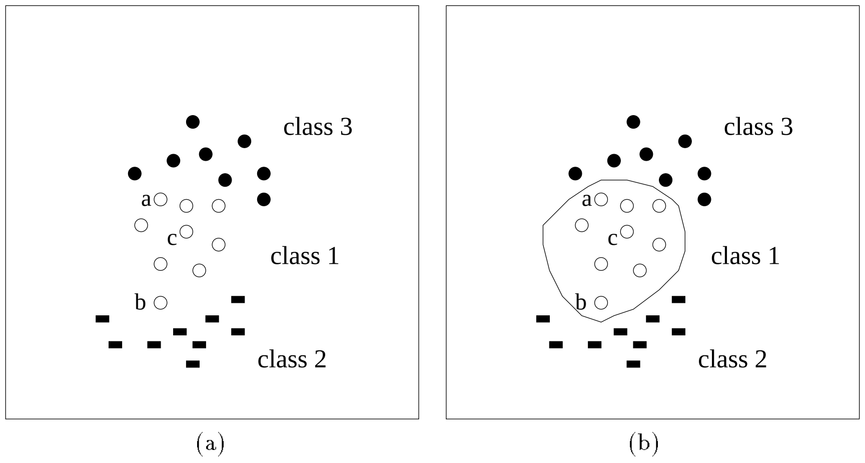


Fig. 1. (a). Examples of 2-D features of three different clusters: the circles belong to cluster 1, the balls belong to cluster 2, and the squares belong to cluster 3. Three points a, b and c are from cluster 1. (b). A nonlinear boundary separates the examples of cluster 1 from cluster 2 and 3.

much different. In addition, using Euclidean distance measures for the nearest neighbor search might retrieve patterns without any perceptual relevance to the original query pattern.

In fact, above problem is classical in pattern recognition, but not much effort has been made to address these issues in the context of image database browsing. Ma and Manjunath [7] present a learning based approach to retrieve the similar image patterns. They use the Kohonen feature map to get a coarse labeling, followed by a fine-tuning process using learning vector quantization. However, the performance of their learning approach is not good when evaluated by the average retrieval accuracy (see Fig. 6-2 on page 108 of [6]). In addition, there are many parameters to be adjusted heuristically and carefully for applications.

Similarity measure is the key component for content-based retrieval. Santini and Jain [15] develop a similarity measure based on fuzzy logic. Puzicha *et al.* [14] compare nine image dissimilarity measures empirically, showing that no measure exhibits best overall performance and the selection of different measures rather depend on the sample distributions. In this paper, we propose a new metric called boundary distance to measure pattern similarities, which is insensitive to the sample distributions. The basic idea here is that a (non-linear) boundary separates the samples belonging to a cluster of similar patterns with the remaining. This non-linear boundary encloses all similar patterns inside. In Fig. 1 (b), a non-linear boundary separates all samples in cluster 1 with others belonging to cluster 2 and 3. The signed distances of all samples to this nonlinear boundary are calculated and used to decide the pattern similarities. This non-linear boundary can be learned from some training examples before we construct an image database.

How to learn the boundary? We argue that an appropriate similarity learning algorithm for application in content based image retrieval should have two properties: 1) good generalization; 2) simple computation. The first one is a common requirement for any learning strategy. While the second is very important for large image database browsing.

A statistical learning algorithm called support vector machine (SVM) [16], is used in our learning approach. The foundations of SVM have been developed by Vapnik [16]. The formulation embodies the Structural Risk Minimization (SRM) principle, which has been shown to be superior to traditional Empirical Risk Minimization (ERM) principle employed by conventional artificial neural networks [4]. SVMs were developed to solve the classification and regression problems [16] [4], and has been used recently to solve the problems in computer vision, such as 3D object recognition [13], face detection [11], and so on.

We adapt the SVMs to solve the image retrieval problem. The major difference from the conventional utilization of SVMs is that we use the SVMs to learn the boundaries.

The paper is organized as follows. In Section 2, we describe the basic theory of SVM and its use for discriminating between different clusters. In Section 3, we present the techniques for learning image pattern similarity and ranking the images. Section 4 evaluates the performance of the new method for similar image retrieval. Finally, Section 5 gives the conclusions.

2 Cluster Discrimination by Support Vector Machines

2.1 Basic Theory of Support Vector Machines

Consider the problem of separating the set of training vectors belonging to two separate classes, $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)$, where $\mathbf{x}_i \in R^n$, a feature vector of dimension n, and $y_i \in \{-1, +1\}$ with a hyperplane of equation $\mathbf{w}\mathbf{x} + b = 0$. The set of vectors is said to be *optimally separated* by the hyperplane if it is separated without error and the margin is maximal. In Fig. 2 (a), there are many possible linear classifiers that can separate the data, but there is only one (shown in Fig. 2 (b)) that maximizes the margin (the distance between the hyperplane and the nearest data point of each class). This linear classifier is termed the optimal separating hyperplane (OSH). Intuitively, we would expect this boundary to generalize well as opposed to the other possible boundaries shown in Fig. 2 (a).

A canonical hyperplane [16] has the constraint for parameters $\mathbf{w}$ and b: $\min_{\mathbf{x}_i} y_i(\mathbf{w} \cdot \mathbf{x}_i + b) = 1$.

A separating hyperplane in canonical form must satisfy the following constraints,

$$y_i \left[(\mathbf{w} \cdot \mathbf{x}_i) + b \right] \geq 1, \quad i = 1, \ldots, l \tag{1}$$

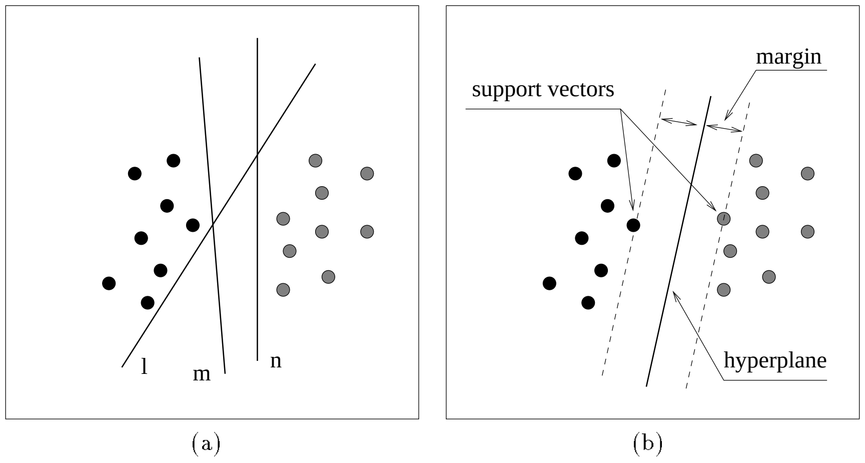


Fig. 2. Classification between two classes using hyperplanes: (a) arbitrary hyperplanes l, m and n; (b) the optimal separating hyperplane with the largest margin identified by the dashed lines, passing the two support vectors.

The margin is $\frac{2}{\|\mathbf{w}\|}$ according to its definition. Hence the hyperplane that optimally separates the data is the one that minimizes

$$\Phi(\mathbf{w}) = \frac{1}{2} \parallel \mathbf{w} \parallel^2 \tag{2}$$

The solution to the optimization problem of (2) under the constraints of (1) is given by the saddle point of the Lagrange functional,

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2} \parallel \mathbf{w} \parallel^2 - \sum_{i=1}^{l} \alpha_i \left\{ y_i \left[(\mathbf{w} \cdot \mathbf{x}_i) + b \right] - 1 \right\} \tag{3}$$

where α_i are the Lagrange multipliers. The Lagrangian has to be minimized with respect to $\mathbf{w}$, b and maximized with respect to $\alpha_i \geq 0$. Classical Lagrangian duality enables the *primal* problem (3) to be transformed to its *dual* problem, which is easier to solve. The *dual* problem is given by,

$$\max_{\alpha} W(\alpha) = \max_{\alpha} \left\{ \min_{\mathbf{w}, b} L(\mathbf{w}, b, \alpha) \right\} \tag{4}$$

The solution to the *dual* problem is given by,

$$\bar{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{5}$$

with constraints $\alpha_i \geq 0, \quad i = 1, \ldots, l$, and $\sum_{i=1}^{l} \alpha_i y_i = 0$.

Solving Equation (5) with constraints determines the Lagrange multipliers, and the OSH is given by,

$$\bar{\mathbf{w}} = \sum_{i=1}^{l} \bar{\alpha}_i y_i \mathbf{x}_i, \qquad \bar{b} = -\frac{1}{2}\bar{\mathbf{w}} \cdot [\mathbf{x}_r + \mathbf{x}_s] \tag{6}$$

where $\mathbf{x}_r$ and $\mathbf{x}_s$ are support vectors, $\bar{\alpha}_r, \bar{\alpha}_s > 0, \quad y_r = 1, \quad y_s = -1$.

For a new data point $\mathbf{x}$, the classification is then,

$$f(\mathbf{x}) = sign\left(\bar{\mathbf{w}} \cdot \mathbf{x} + \bar{b}\right) = sign\left(\sum_{i=1}^{l} \bar{\alpha}_i y_i (\mathbf{x}_i \cdot \mathbf{x}) + \bar{b}\right) \tag{7}$$

To generalize the OSH to the non-separable case, slack variables ξ_i are introduced [2]. Hence the constraints of (1) are modified as

$$y_i\left[(\mathbf{w} \cdot \mathbf{x}_i) + b\right] \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, l \tag{8}$$

The generalized OSH is determined by minimizing,

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2} \parallel \mathbf{w} \parallel^2 + C\sum_{i=1}^{l} \xi_i \tag{9}$$

(where C is a given value) subject to the constraints of (8).

This optimization problem can also be transformed to its *dual* problem, and the solution is the same as (5), but adding the constraints to the Lagrange multipliers by $0 \leq \alpha_i \leq C, \quad i = 1, \ldots, l$.

2.2 Non-linear Mapping by Kernel Functions

In the case where a linear boundary is inappropriate, the SVM can map the input vector, $\mathbf{x}$, into a high dimensional feature space, $\mathbf{z}$. The SVM constructs an optimal separating hyperplane in this higher dimensional space. In Fig. 3, the samples in the input space can not be separated by any linear hyperplane, but can be linearly separated in the non-linear mapped feature space. Note that the feature space in SVMs is different from our texture feature space. According to the Mercer theorem [16], there is no need to compute this mapping explicitly, the only requirement is to replace the inner product $(\mathbf{x}_i \cdot \mathbf{x}_j)$ in the input space with a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ to perform the non-linear mapping. This provides a way to address the curse of dimensionality [16].

There are three typical kernel functions [16]:

Polynomial

$$K(\mathbf{x}, \mathbf{y}) = ((\mathbf{x} \cdot \mathbf{y} + 1))^d \tag{10}$$

where the parameter d is the degree of the polynomial.

Gaussian Radial Basis Function

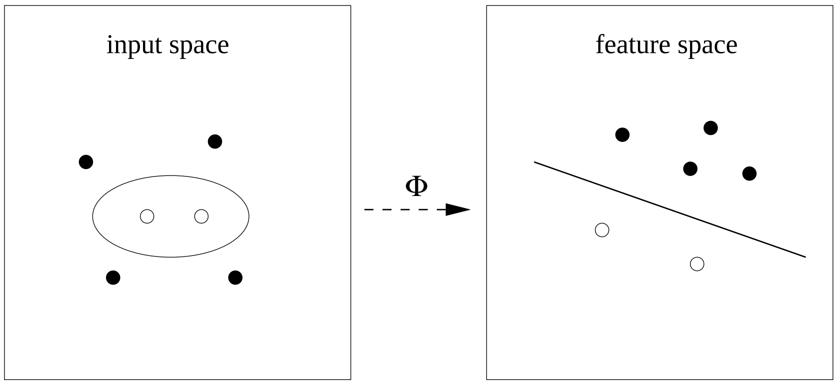


Fig. 3. The feature space is related to input space via a nonlinear map $\mathbf{\Phi}$, causing the decision surface to be nonlinear in the input space. By using a nonlinear kernel function, there is no need to do the mapping explicitly.

$$K(\mathbf{x},\mathbf{y}) = \exp\left(-\frac{(\mathbf{x}-\mathbf{y})^2}{2\sigma^2}\right) \tag{11}$$

where the parameter σ is the width of the Gaussian function.
Multi-Layer Perception

$$K(\mathbf{x},\mathbf{y}) = \tanh\left(scale.(\mathbf{x}\cdot\mathbf{y}) - offset\right) \tag{12}$$

where the $scale$ and $offset$ are two given parameters.

For a given kernel function, the classifier is thus given by,

$$f(\mathbf{x}) = sign\left(\sum_{i=1}^{l}\bar{\alpha}_i y_i K(\mathbf{x}_i,\mathbf{x}) + \bar{b}\right) \tag{13}$$

2.3 Discrimination Between Multiple Clusters

Previous subsections describe the basic theory of SVM for two-class classification. For image retrieval in a database of multiple image clusters, for instance, c clusters, we can construct c decision boundaries. Note that a cluster may contain more than one image class. The perceptually similar images in different classes are considered as one cluster. Each boundary is used to discriminate between the images of one cluster and all the remaining belonging to other clusters. The images belonging to cluster k are enclosed by the k^{th} boundary. This is a one-against-all strategy in the context of pattern recognition. To our knowledge, only the recently proposed support vector machines can be used to obtain the boundary optimally by quadratic programming.

3 Similarity Measure and Ranking

The basic idea of learning similarity is to partition the original feature space into clusters of visually similar patterns.

The pair $(\mathbf{w}, b)$ defines a *separating hyperplane* or boundary of equation

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{14}$$

By kernel mapping, the boundary $(\mathbf{w}, b, K)$ is,

$$\sum_{j=1}^{m} \bar{\alpha}_j^* y_j K(\mathbf{x}_j^*, \mathbf{x}) + \bar{b}^* = 0 \tag{15}$$

where $\mathbf{x}_j^* (j = 1, \cdots, m)$ are support vectors, $\bar{\alpha}_j^*$ are the linear combination coefficients or weights, and $\bar{b}^*$ is a constant. Usually, $m < l$, *i.e.*, the number of support vectors is less than that of the training examples.

Definition 1 (signed distance):

The signed distance $D(\mathbf{x}_i; \mathbf{w}, b)$ of a point $\mathbf{x}_i$ from the boundary $(\mathbf{w}, b)$ is given by

$$D(\mathbf{x}_i; \mathbf{w}, b) = \frac{\mathbf{w} \cdot \mathbf{x}_i + b}{\| \mathbf{w} \|} \tag{16}$$

Definition 2 (signed distance with kernel):

The signed distance $D(\mathbf{x}_i; \mathbf{w}, b, K)$ of a point $\mathbf{x}_i$ from the boundary $(\mathbf{w}, b, K)$ with kernel function $K(\cdot, \cdot)$ is given by

$$D(\mathbf{x}_i; \mathbf{w}, b, K) = \frac{\sum_{j=1}^{m} \bar{\alpha}_j^* y_j K\left(\mathbf{x}_j^*, \mathbf{x}_i\right) + \bar{b}^*}{\| \sum_{j=1}^{m} \bar{\alpha}_j^* y_j \mathbf{x}_j^* \|} \tag{17}$$

Combining Definitions 1 and 2 with equation (1), we have

$$y_i D(\mathbf{x}_i; \mathbf{w}, b, K) \geq \frac{1}{\| \sum_{j=1}^{m} \bar{\alpha}_j^* y_j \mathbf{x}_j^* \|} \tag{18}$$

for each sample $\mathbf{x}_i$, and $y_i = \pm 1$, $i = 1, \cdots, l$. Therefor, we have,

Corollary: The lower bound of the positive examples $(y_i = 1)$ to the boundary $(\mathbf{w}, b, K)$ is $\frac{1}{\|\sum_{j=1}^{m} \bar{\alpha}_j^* y_j \mathbf{x}_j^*\|}$; the upper bound of the negative examples $(y_i = -1)$ is $-\frac{1}{\|\sum_{j=1}^{m} \bar{\alpha}_j^* y_j \mathbf{x}_j^*\|}$. The boundary $(\mathbf{w}, b, K)$ is between these two bounds.

In our cluster-based similarity measure, the perceptually similar patterns are grouped into one cluster and labeled as positive examples, while the other patterns are treated as being dissimilar to this cluster. Thus we give,

Definition 3 (similarity measure):

The patterns $\mathbf{x}_i$, $i = 1, \cdots, l'$, are said with perceptual similarity if they are considered as positive samples $(y_i = 1)$ and hence located inside the boundary $(\mathbf{w}, b, K)$; the samples outside are said dissimilar to the patterns enclosed by the boundary.

In the case of c clusters, we have c boundaries.

Definition 4 (signed distance to the k^{th} boundary):

If the boundary separating cluster k from others is $(\mathbf{w}_k, b_k, K)$, the signed distance of a pattern $\mathbf{x}_i$ to the k^{th} boundary is

$$D(\mathbf{x}_i; \mathbf{w}_k, b_k, K) = \frac{\sum_{j=1}^{k_m} \bar{\alpha}^*_{kj} y_{kj} K\left(\mathbf{x}^*_{kj}, \mathbf{x}_i\right) + \bar{b}^*_k}{\| \sum_{j=1}^{k_m} \bar{\alpha}^*_{kj} y_{kj} \mathbf{x}^*_{kj} \|} \tag{19}$$

where $\mathbf{x}^*_{kj}$, $(j = 1, \cdots, k_m)$ are the support vectors, $\bar{\alpha}^*_{kj}$ are the optimal Lagrange multipliers for the k^{th} boundary, and $\bar{b}^*_k$ are some constants, $k = 1, \cdots, c$.

Equation (19) is used to calculate the signed distances of patterns to the k^{th} boundary. The pattern similarities (dissimilarities) are measured by Definition 3.

How to connect the c boundaries to each pattern in a database? It is realized as follows: when an image pattern $\mathbf{x}_i$ is ingested into the database during its construction, the signed distances of $\mathbf{x}_i$ to the stored c boundaries are calculated firstly by equation (19), and then the index k^* is selected by,

$$k^* = argmax_{1 \leq k \leq c} D(\mathbf{x}_i; \mathbf{w}_k, b_k, K) \tag{20}$$

The index k^* is therefore connected to the image pattern $\mathbf{x}_i$. Basically, this is a classification problem. Each pattern in the database will be associated with a boundary index.

In retrieval, when a query image pattern is given, the boundary index k^* connected to the query pattern is first found. Then, we use equation (19) to calculate the signed distances of all samples to the k^{*th} boundary. According to Definition 3, the images in the database with positive distances to the k^{*th} boundary are considered similar. Thus we obtain the similar image patterns to the query.

How to rank these similar images? The similar images obtained above belong to different texture classes. To rank these images, the class information should be taken into consideration. Assume that cluster k $(k = 1, \cdots, c)$ contains q_k texture classes, the feature space of cluster k is further divided into q_k subspaces. Each subspace is enclosed by a sub-boundary containing the patterns of the same class. Thus, we partition the feature space in a hierarchical manner: in the higher level, the database is divided into c clusters, with each contains the perceptually similar patterns inside; in the lower level, each cluster k is further divided into q_k texture classes. The signed distances to the sub-boundary q^*_k of all image patterns enclosed by the boundary k^* are used for ranking, if the query image pattern is located inside the sub-boundary q^*_k.

In summary, each (query) image is associated with two-level boundary indexes. The images selected by the higher level boundary are ranked by their signed distances to the lower level boundary.

The hierarchical approach to texture image retrieval has two advantages: one is to retrieve the perceptually similar patterns in the top matches; the other is to speed up the retrieval process further. Note that there is no need to compute the Euclidean distance between two points as in [7].

4 Image Retrieval Performance

The Brodatz texture database [1] used in the experiments consists of 112 texture classes. Each of the 512×512 images is divided into 49 sub-images (with overlap), which are 128×128 pixels, centered on a 7×7 grid over the original image. The first 33 sub-images are used as the training set and the last 16 for retrieval [7]. Thus we create a database of 3696 texture images for learning, and a database of 1792 texture images to evaluate the retrieval performance. A query image is one of the 1792 images in the database.

In this paper we use a similar Gabor filter banks [3] as that derived in [8], where four scales and six orientations are used. Applying these Gabor filters to an image results in 24 filtered images. The mean and standard deviation of each filtered image are calculated and taken as a feature vector

$$\bar{f} = [\mu_{00}, \mu_{01}, \cdots, \mu_{35}, \sigma_{00}, \cdots, \sigma_{35}] \tag{21}$$

where the subscripts represent the scale ($s = 0, \cdots, 3$) and orientation ($o = 0, \cdots, 5$). The dimension of the feature vector is thus 48.

The 112 texture image classes are grouped into 32 clusters, each containing 1 to 8 similar textures. This classification was done manually and Table 1 shows the various clusters and their corresponding texture classes. Note that we use all the 112 texture classes.

In the learning stage, we use the Gaussian RBF kernel function with the parameter $\sigma = 0.3$ and $C = 200$. Figure 4 illustrates an evaluation based on the average retrieval accuracy defined as the average percentage number of patterns belonging to the same image class as the query pattern in the top 15 matches [8]. The comparison is between our hierarchical approach to learning similarity and ranking and that based on the Euclidean distance measure. Note that the significant better result achieved by our method. This figure demonstrates that our hierarchical retrieval can give better result than the traditional Euclidean distance based approach. Note that the learning approach in [6] gives nearly the same result as that based on Euclidean distance (Fig. 6-2 on page 108 of [6]).

Since the average retrieval accuracy does not consider the perceptual similarity, another evaluation is done, based on the 32 clusters instead of just the top 15 matches [7]. Figure 5 illustrates the second evaluation result. Here the differences are quite striking. The performance without learning deteriorates rapidly after the fist $10 \sim 15$ top matches, however, the retrievals based on our learning similarity perform very well consistently.

Figure 6 and 7 show some retrieval examples, which clearly demonstrate the superiority of our learning approach. Another important issue is the hierarchical retrieval structure which speed up the search process.

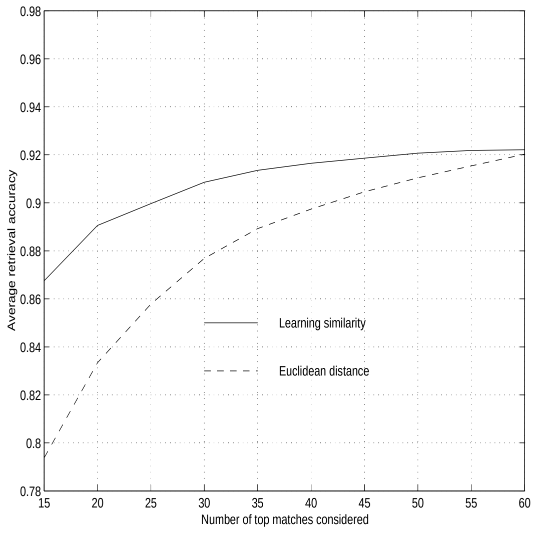


Fig. 4. The retrieval performance in terms of obtaining the 15 correct patterns from the same texture class as the top matches. The performance improves explicitly on the small number of top matches considered. Note that this evaluation does not refer to the visual similarity.

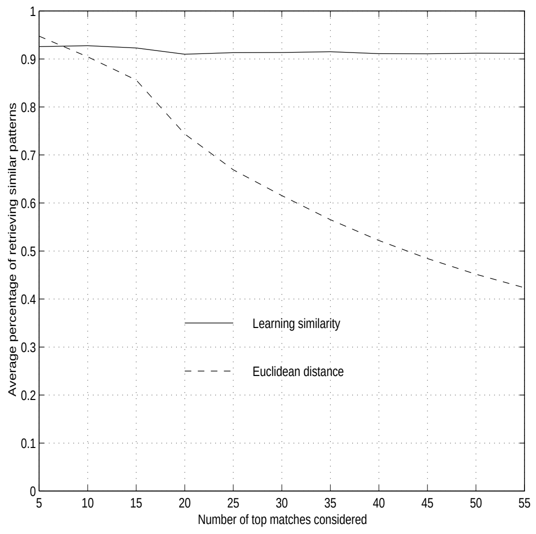


Fig. 5. The retrieval performance comparison between our learning similarity approach and that based on Euclidean distance. The evaluation takes the perceptual similarity into consideration. Notice the significant improvement in our retrieval with learning similarity.

Table 1. Texture clusters used in learning similarity. The visual similarity within each cluster are identified by the people in our research group.

Cluster	Texture Class	Cluster	Texture Class
1	d001 d006 d014 d020 d049	17	d069 d071 d072 d093
2	d008 d056 d064 d065	18	d004 d029 d057 d092
3	d034 d052 d103 d104	19	d039 d040 d041 d042
4	d018 d046 d047	20	d003 d010 d022 d035 d036 d087
5	d011 d016 d017	21	d048 d090 d091 d100
6	d021 d055 d084	22	d043 d044 d045
7	d053 d077 d078 d079	23	d019 d082 d083 d085
8	d005 d033 d032	24	d066 d067 d074 d075
9	d023 d027 d028 d030 d054 d098 d031 d099	25	d101 d102
10	d007 d058 d060	26	d002 d073 d111 d112
11	d059 d061 d063	27	d086
12	d062 d088 d089	28	d037 d038
13	d024 d080 d081 d105 d106	29	d009 d109 d110
14	d050 d051 d068 d070 d076	30	d107 d108
15	d025 d026 d096	31	d012 d013
16	d094 d095	32	d015 d097

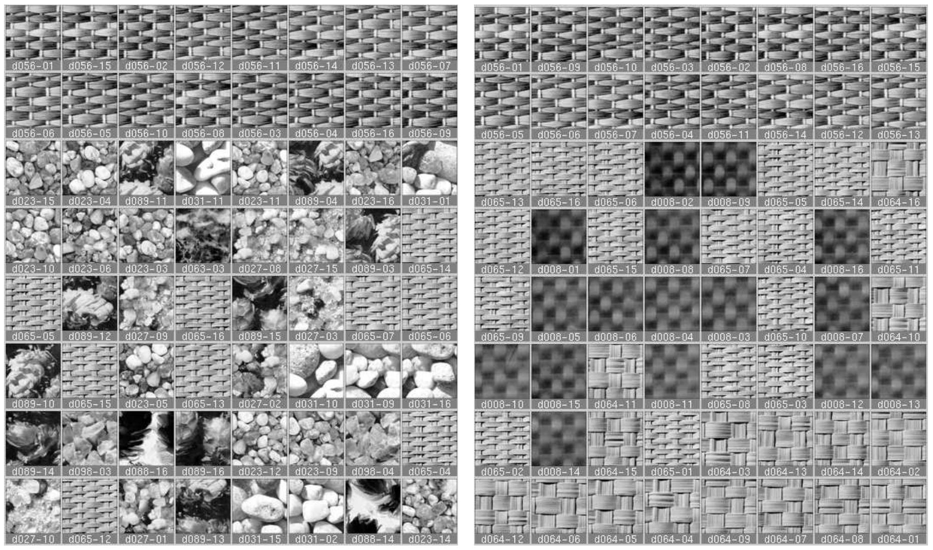
5 Conclusions

We have presented a new algorithm to learn pattern similarity for texture image retrieval. The similar patterns are grouped into a cluster in the feature space. The boundaries isolating each cluster with others can be learned efficiently by support vector machines (SVMs). Similarity measure and ranking are based on the signed distances to the boundaries, which can be simply computed. The performance of similar pattern retrieval is significantly improved as compared to the traditional Euclidean distance based approach.

References

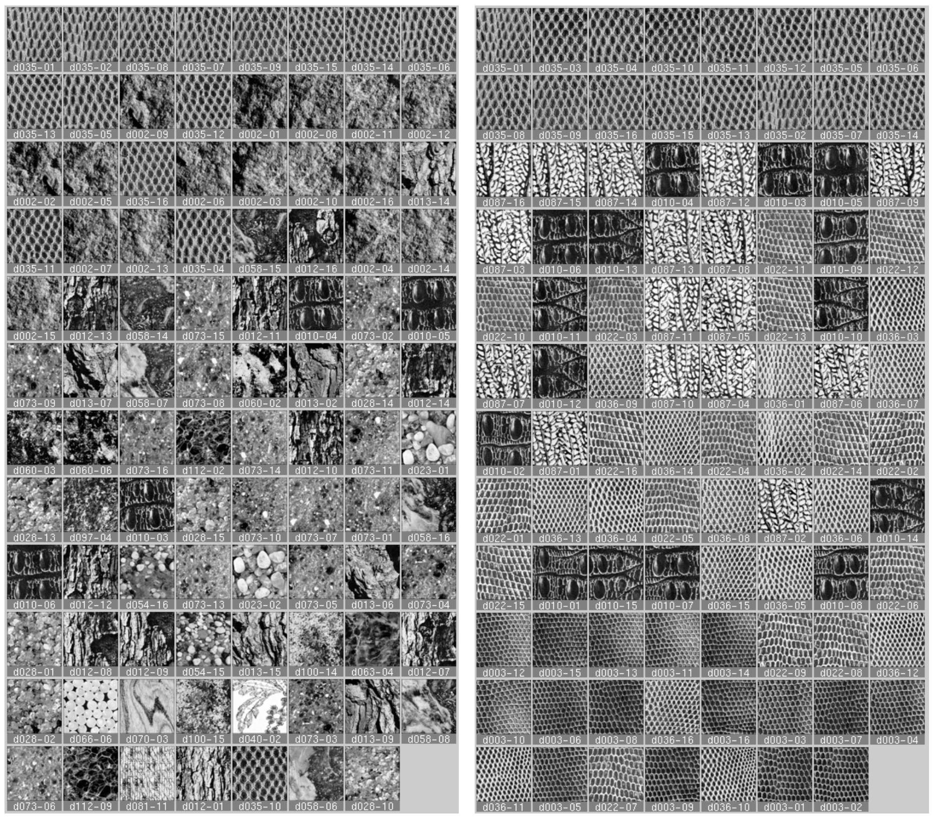
1. P. Brodatz, *Textures: A Photographic Album for Artists & Designers.* New York: Dover, 1966.
2. C. Cortes and V. Vapnik, Support vector networks, *Machine Learning*, 20, 273-297, 1995.
3. J. G. Daugman, Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression, *IEE Trans. ASSP*, vol 36, 1169-1179, July 1988.
4. S. Gunn, M. Brown and K. M. Bossley, Network performance assessment for neurofuzzy data modeling. *Lecture Notes in Computer Science*, 1280: 313-323, 1997.
5. S. Gunn, Support vector machines for classification and regression, ISIS Technical Report, May, 1998.
6. W. Y. Ma, *NETRA: A Toolbox for Navigating Large Image Databases.* PhD thesis, University of California at Santa Barbara, 1997.

7. W. Y. Ma and B. S. Manjunath, Texture features and learning similarity. *Proc. CVPR*, 425-430, 1996.
8. B. S. Manjunath and W. Y. Ma, Texture features for browsing and retrieval of image data. *IEEE PAMI*, vol. 18, No, 8, 837-842, 1996.
9. T. P. Minka and R. W. Picard, Interactive learning using a "society of models", Technical Report No. 349, MIT Media Laboratory, 1995.
10. W. Niblack, *et al*, The QBIC project: querying images by content using color, texture, and shape, *Proc. of SPIE*, Storage and Retrieval for Image and Video Databases-*II*, vol. 1908, San Jose, CA, 173-187, Feb. 1993.
11. E. Osuna, R. Freund and F. girosi, Training support vector machines: an application to face detection. *Proc. CVPR*, 1997.
12. A. Pentland, R. W. Picard, and S. Sclaroff, Photobook: tools for content based manipulation of image databases, *Proc. of SPIE*, Storage and Retrieval for Image and Video Databases-*II*, No. 2185, San Jose, CA, 34-47, Feb. 1994.
13. M. Pontil and A. Verri, Support vector machines for 3-D object recognition, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, 637-646, 1998.
14. J. Buzicha, J. M. Buhmann, Y. Rubner and C. Tomasi, Emperical evaluation of dissimilarity measures for color and texture, *Proc. ICCV*, vol. *II*, 1165-1172, 1999.
15. S. Santini and R. Jain, Similarity measures, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 21, No. 9, 871-883, 1999.
16. V. N. Vapnik, *Statistical learning theory*, John Wiley & Sons, New York, 1998.



(a) Euclidean distance measure (b) learning similarity

Fig. 6. Image retrieval comparison. Each query image has 15 other similar images in the database. The query image (d065-01) is shown at the top left in each case. Note that the degradation in visual similarity in the case of Euclidean distance measure. The images are ordered according to decreasing similarity from left to right and top to bottom. In the case of learning similarity, the performance continues without any marked degradation in perceptual similarity, even after 60 Images are retrieved.



(a) Euclidean distance measure (b) learning similarity

Fig. 7. Image retrieval comparison. Each query image has 15 other similar images in the database. The query image (d035-01) is shown at the top left in each case. Note that the degradation in visual similarity for the case of Euclidean distance measure. In the case of learning similarity, the performance continues without any marked degradation in perceptual similarity, even after 90 images are retrieved.