Available online at www.sciencedirect.com
ScienceDirect
Neurocomputing 70 (2007) 1040–1050
NEUROCOMPUTING
www.elsevier.com/locate/neucom

A new image classification technique using tree-structured regional features

Tommy W.S. Chow*, M.K.M. Rahman

Dept of Electronic Engineering, City University of Hong Kong, Hong Kong

Received 12 November 2004; received in revised form 26 January 2006; accepted 26 January 2006
Communicated by T. Heskes
Available online 10 October 2006

Abstract

Image classification is a challenging problem of computer vision. Conventional image classification methods use flat image features with fixed dimensions, which are extracted from a whole image. Such features are computationally effective but are crude representation of the image content. This paper proposes a new image classification approach through a tree-structured feature set. In this approach, the image content is organized in a two-level tree, where the root node at the top level represents the whole image and the child nodes at the bottom level represent the homogeneous regions of the image. The tree-structured representation combines both the global and the local features through the root and the child nodes. The tree-structured feature data are then processed by a two-level self-organizing map (SOM), which consists of an unsupervised SOM for processing image regions and a supervising concurrent SOM (CSOM) classifier for the overall classification of images. The proposed method incorporates both global image features and local region-based features to improve the performance of image classification. Experimental results show that this approach performs better than conventional approaches.

Keywords: Feature integration; Region-based image comparison; Image similarity measure; Self-organizing map; Image classification

1. Introduction

Recently, content-based image classification and retrieval received increasing attention through numerous applications [8,10–13,21] in the field of education, entertainment, military, and biomedicine. With the immense growth of computational power and the continuously declining hardware cost, image retrieval, image classification, and pattern recognition have become more demanding in the area of computer vision. The success of solving such problem lies in the issues of object-based image understanding, proper representation of image contents, and suitable learning algorithms. Traditionally, flat image features (fixed length features extracted from whole image), such as color, texture, and their combinations [8,10,12,13,17], are used for image retrieval and classification. Color histogram, which can be extracted efficiently, is the most popular feature. Color histogram is generally insensitive to small changes in camera positions, but it is a very coarse representation in a sense that it discards other important features like textures, shapes, sizes, and positions of the objects. Thus, color histogram lacks spatial information and is sensitive to intensity variation, color distortion, and cropping. As a result, images with similar histogram may exhibit substantially different semantics [14]. To overcome the limitation of color histogram, color layout approach was introduced to partition an image into blocks. Subsequently, the average colors [16], or Daubechies' wavelet coefficients, [25] of those blocks were used as features. The shape, the position, and the texture information can be maintained at a proper resolution, but the retrieval performance is sensitive to the procedures of shifting, cropping, or scaling, because local properties are emphasized. Gabor filter [10] is widely adopted to extract texture features from images and it has been shown to be an efficient approach for performing image retrieval and classification. Combination of color

*Corresponding author. Tel.: +852 27887756; fax: +852 27887791.
E-mail address: eetchow@cityu.edu.hk (T.W.S. Chow).

0925-2312/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2006.01.033

and texture features [17] is shown to be more effective in describing image content compared with color or texture alone. All these flat vector-type image features are excellent in comparing the overall image similarity, but they are still unable to describe the high level and the object-based property of the image content.

A meaningful image representation is the key for performing classification of images. Region-based approaches [3,9,20,21,24] were introduced for a better understanding of image contents. In the region-based approaches, instead of extracting features from the whole image, the image is first decomposed into regions for the subsequent features extraction. The similarity between two images is expressed as all the possible combinations of the similarities among their regions. However, region-based approach has been limited to image retrieval application for two main reasons. First, the comparison of two images is not as straightforward as comparing two feature vectors. Second, traditional neural learning is not directly applicable to such non-flat structured data. Therefore, one query image must be compared with every other image of the database. Tree-structured representation is shown to be an effective approach for image processing and image analysis [19]. There are evidences indicating that region-based image representation can be better encoded using a tree representation. Some successful works can be found in [1,2,15,18,26]. In [15,26], binary space partition (BSP) tree representation is used for coding images in terms of segmented regions. Recent works in [1,2,18] indicate that the BSP tree-based region-oriented image representation can be effective for image processing, classification, and retrieval. Though BSP provides efficient processing of data, partitioning a region into two sub-regions in BSP does not always provide a meaningful representation of the objects in a real image. Height of the BSP tree grows linearly with the number of objects in the image that lacks clarity in representing image contents.

In this paper, an improved image classification approach is proposed. Image contents are represented by integrating both the global image features and the local region-based features. Fig. 1 briefly illustrates the proposed approach. A two-level tree hierarchically contains all image features, where the root node represents the whole image and the child nodes represent the image regions. Thus, the global and local image features are represented by the root and child nodes, respectively. Two-level self-organizing map (SOM) networks are used to process the tree-structured data. First, all image regions of the whole database are processed by an unsupervised SOM. After the completion of training, image regions are compressed by the positions of winner neurons on the SOM map. The position vectors together with the global image features are then used to classify the images through a concurrent self-organizing maps (CSOM) classifier [11]. Experimental results indicate that the proposed method delivers better results in the application of image classification. It is also worth noting that the proposed method is able to maintain the computational cost at a reasonably low level.

This paper is organized as follows. In Section 2, the representation and the feature extraction of image contents are described. Section 3 elaborates a two-layer SOM-based image classification system. Section 4 presents the experimental results and discussions. Finally, conclusion is drawn in Section 5.

2. Representation of image contents

Feature extraction and representation is important in the sense that they serve as dimensionality reduction for enabling the analysis of image contents. However, features should also reflect the high-level semantic (perceived by human) in addition to visual similarity obtained by global features like color histogram. Thus, different region-based approaches were proposed to provide a better understanding of image semantic. In this work, a composite region-based image representation is developed to integrate both the visual and regional properties of the image content. To extract regional features, JSEG, a color image segmentation method [6], is used. In brief, JSEG first quantizes colors of an image to several representative classes. It then labels pixels with the color classes to form a class map of the image. At last, the image is segmented using multi-scale J-images [6]. Experimental results showed

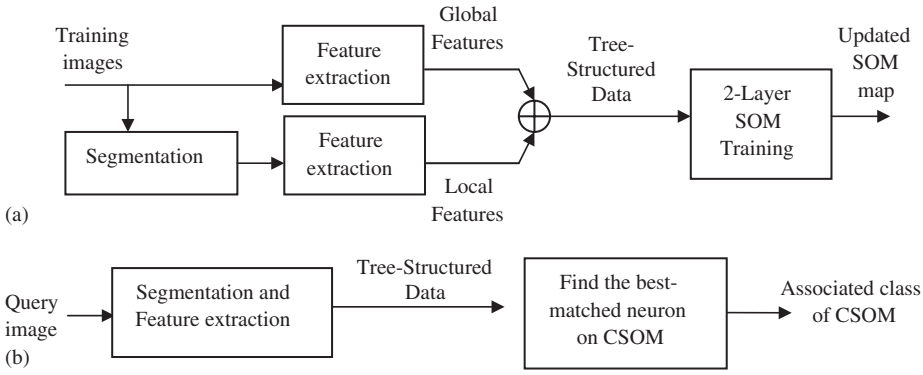

Fig. 1. Overview of the 2-layer SOM-based classification system (a) training phase and (b) classification phase.

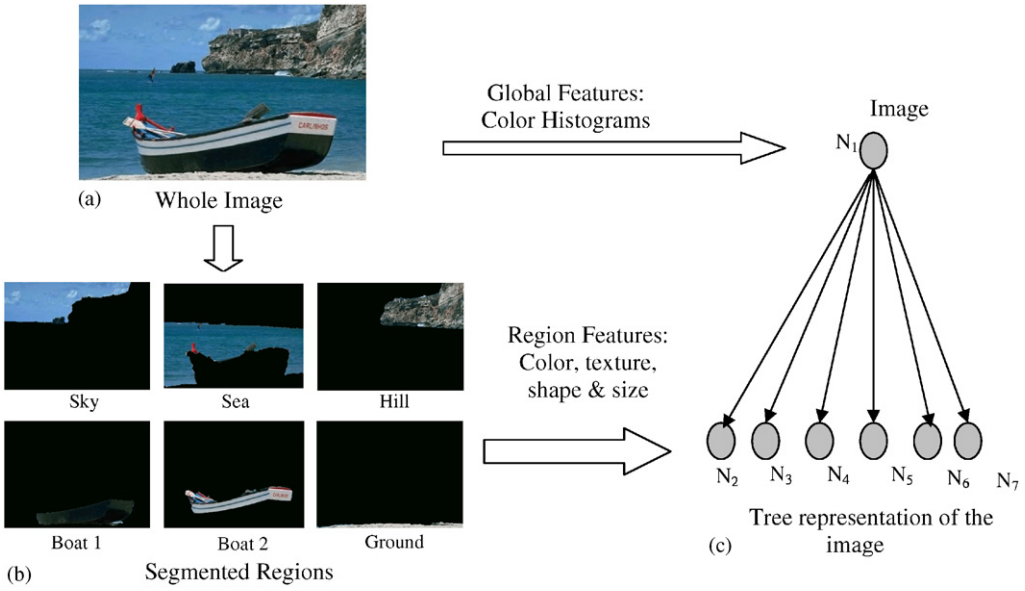

Fig. 2. Representation of image contents by integrating global features and local region-based features (a) whole image (b) segmented regions, and (c) tree representation of the image.

that JSEG provides good segmentation on a variety of color images [5].

After image segmentation, the image is decomposed into a number of homogeneous regions. In Fig. 2, it shows that the image is represented by a two-level tree, where the root node represents the whole image and child nodes represent the region-based objects. The root node is assigned to the global feature, which is the color histogram in this case. Local region-based features, such as color moment, texture, size, and shape, are assigned to the child nodes. This enables global and local image features to be integrated through a tree structure. Hue, saturation, and value (HSV) color space is used to calculate the color histograms. The histogram of each channel is computed as follows:

$$h_G = \frac{n_G}{n_T}, \quad G = 1, 2, \ldots, q, \tag{1}$$

where G represents a quantized level of an HSV color channel, n_G is the total number of pixels in that level, n_T is the total number of pixels, and q is the number of quantized levels. To avoid computational burden, only 16 or 8 quantized levels are used to calculate the histograms of each of the 3 HSV channels. The complete histogram vector is represented as follows:

$$\boldsymbol{H}_q = \left[h_{H1}, \ldots, h_{Hq}, \quad h_{S1}, \ldots, h_{Sq}, \quad h_{V1}, \ldots, h_{Vq}\right].$$

The HSV color space is also used to calculate the first and the second-order color moments of the image region:

$$\boldsymbol{CM}_2 = \left[\mu_{c1}\ \mu_{c2}\ \mu_{c3}\ \sigma_{c1}\ \sigma_{c2}\ \sigma_{c3}\right],$$

where μ_{c1}, μ_{c2}, and μ_{c3} are the means of the three channels of a region, and σ_{c1}, σ_{c2}, and σ_{c3} are their corresponding standard deviations. We also used the third-order moment for the whole image:

$$\boldsymbol{CM}_3 = \left[\mu_{c1}\ \mu_{c2}\ \mu_{c3}\ \sigma_{c1}\ \sigma_{c2}\ \sigma_{c3}\ \gamma_{c1}\ \gamma_{c2}\ \gamma_{c3}\right],$$

where γ_{c1}, γ_{c2}, and γ_{c3} represent the skewness of each color.

Three features are used to describe the shape properties of a region. They are normalized inertia [24] with orders from 1 to 3. For a region M in Euclidean space $\Re^2$, the normalized inertia of order α is

$$I_\alpha(M) = \frac{\sum_{(x,y)\in M}\left\{(x-\bar{x})^2 + (y-\bar{y})^2\right\}^{\alpha/2}}{\{n_M\}^{1+\alpha/2}}, \tag{2}$$

where $(\bar{x}, \bar{y})$ is the centroid of the region, and n_M is the number of pixels in the region M. The three shape features are represented as

$$\boldsymbol{S} = [s_1\ s_2\ s_3] = \left[\frac{I_1(M)}{I_1^o}, \frac{I_2(M)}{I_2^o}, \frac{I_3(M)}{I_3^o}\right], \tag{3}$$

where I_1^o, I_2^o, I_3^o are the minimum normalized inertia achieved by circular area. The size feature of a region M in the image I is defined as

$$A(M) = \frac{n_M}{n_T}, \tag{4}$$

where n_M is the number of pixels in a region M, and n_T is the total number of pixels in the image. Gabor filter is the most commonly used method in extracting texture features. It is a group of wavelets that capture energy at a specific scale (frequency) and a specific direction. It provides a localized frequency description and captures the local features/energy of the signal. Texture features can then be extracted from this group of energy distributions. The scale and orientation tunable property of Gabor filter makes it especially useful for texture analysis. The details can be found in [7,10]. Suppose $G_{uv}(x, y)$ is a discrete Gabor wavelet transform for a given image $I(x, y)$ at uth scale and

vth orientation, means and standard deviations of $G_{uv}(x, y)$ are computed to represent the Gabor features:

$$\mu_{uv} = \frac{1}{PQ}\sum_x \sum_y |G_{uv}(x,y)|, \tag{5}$$

$$\sigma_{uv} = \frac{1}{PQ}\sqrt{\sum_x \sum_y |G_{uv}(x,y)| - \mu_{uv}}, \tag{6}$$

where $P \times Q$ is the size of image. The complete Gabor features is then represented as

$$\boldsymbol{G}_{U,V} = \left[\mu_{00,}\sigma_{00}, \mu_{01,}\sigma_{01}, \ \ldots \ , \mu_{0(U-1)}, \sigma_{0(V-1)}, \right.$$
$$\left. \ldots \ , \mu_{(U-1)(V-1)}, \sigma_{(U-1)(V-1)}\right].$$

In this study, another smaller set of wavelet features are used for the image regions that have been successfully used in region-based image retrieval [24]. This set contains the moments of wavelet coefficients in high-frequency bands as a texture feature representing energy in the high-frequency bands by Haar wavelet transform. After one-level wavelet transform of the intensity image, a 4×4 image block is decomposed into the four frequency bands, namely LL, LH, HL, and HH bands [4]. Each band contains 2×2 coefficients. Suppose the coefficients in the LH band are (a_1, a_2, a_3, a_4), the texture feature of that block in the LH band is computed as

$$t_1 = \left[\tfrac{1}{4}\left(a_1^2 + a_2^2 + a_3^2 + a_4^2\right)\right]^{1/2}. \tag{7}$$

The other two features t_2 and t_3 are computed similarly from the HL and HH bands. The mean and standard deviation in the LH, HL, and HH bands are thus used as texture feature:

$$\boldsymbol{T} = \left[\mu_{t_1}\ \mu_{t_2}\ \mu_{t_3}\ \sigma_{t_1}\ \sigma_{t_2}\ \sigma_{t_3}\right].$$

3. Image classification by two-layer SOM networks

3.1. Encoding of data

Traditional neural network-based classifiers only deal with fixed vector-type input data. In this application, the tree-structured image data cannot be encoded with fixed input vector simply by adding features of all nodes. It is because the number of segmented regions is not fixed for every image and the simple addition of nodes' features can result in too high a dimensional input vector. To solve this problem, we use the SOM's property of dimensionality reduction. The child nodes' (regions) features are first processed by a SOM to reduce the dimension of the final input vector. Thus, special arrangement of a two-layer SOM network is used to process the tree. The bottom layer consists of a SOM network to process the child nodes, while the top layer that consists of a CSOM classifier processes the root nodes. Fig. 3 demonstrates the processing of tree-structured data. To start the training process, the bottom SOM layer is first trained by the child node inputs from all tree data. After the completion of training, each child node is associated with its best-matched neuron on the SOM. Positions of these neurons are then used for inputs encoding the root node together with the features of the root nodes. The root nodes, the final identity of the images, are then processed with the CSOM classifier [11].

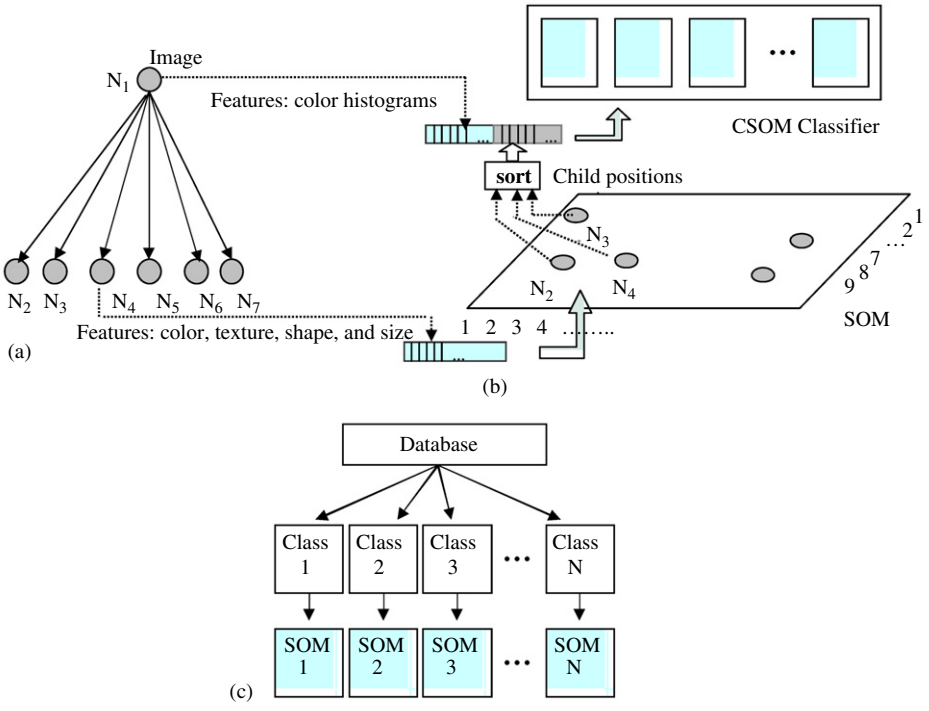

Fig. 3. (a–b) Describing the mapping of image tree into the 2-layer SOM. (c) Training phase of CSOM classifier indicating that each SOM block is trained with one data class only.

Thus, the bottom layer SOM is used for region encoding, while the top layer CSOM is used for image classification task.

In this paper, a tree structure is represented by a set of nodes as $G = \{N_1, N_2, \ldots, N_T\}$, where T is the total number of nodes of the tree, N_1 is the root node, and others are child nodes. Input encoding of a 2nd level (child) node (N_2 to N_T) can be simply represented as $X \leftarrow \{f_1, f_2, \ldots, f_m\}$, where f represents a feature value, and m is the number of features used for the image regions. However, the root node encoding is represented by $X \leftarrow \{f_1, f_2, \ldots, f_k, \quad p_1, p_2, \ldots, p_{c_{\max}}\}$, where k is the total number of global features, and $p_i (= [x_i, y_i]^{\mathrm{T}})$ is a 2-D position vector. The position vectors contain the positions of the best-matching neurons on the SOM map for all the child nodes of the tree. The value of $c_{\max}$ is equal to or greater than the maximum number of the child nodes found in the database. Thus, some p_i can contain zero vectors for a root node when the number of its child nodes is less than $c_{\max}$. These position vectors are sorted according to their spatial positions on the SOM map, which is discussed in the following section.

3.2. Training process

The training process runs in a bottom up fashion. First, all the child nodes are collected from the whole database and are fed for training the bottom layer SOM. The input vectors of the root nodes are formed prior to the training of the CSOM. In CSOM [11], instead of using a big SOM map, N smaller SOMs are used for the classification of N classes of data. Each SOM of the CSOM is labeled and trained only with one specific class of data (Fig. 3c). Training is performed in the same way as the ordinary SOM. In the recognition/classification phase, the test pattern is compared with every SOM of CSOM. The SOM that provides the lowest quantization error indicates the class of the test pattern. The overall training process of the two-layer SOM network can be summarized as follows:

SOM training algorithm

Step 1. Randomly ***initialize*** the weight vector of bottom layer SOM. Collect all child nodes as the training inputs for bottom layer SOM.
Step 2. Set *Iteration* = 0.
Step 3. Randomly select an input data.
Step 4. Find a winner neuron on the SOM for the input data using ***similarity measure***.
Step 5. Set the learning rate and the neighborhood function according to iteration number.
Step 6. ***Update*** the winner neuron and its neighbors on the SOM map.
Step 7. Set *Iteration* = *Iteration* + 1
Step 8. If *Iteration* reaches its maximum value go to next step, otherwise go to Step 3.
Step 9. Find the final winner neuron for each child node and save the information.
Step 10. ***Make up the input vectors*** for root nodes by combining features and child positions.
Step 11. Randomly ***initialize*** the weight vector of each SOM of the CSOM.
Step 12. Set class $j = 1$.
Step 13. Select the jth SOM of CSOM as active SOM for training. Select root nodes only from the jth class as the training input data.
Step 14. Do Steps (2–8) for training the jth SOM of CSOM.
Step 15. Set $j = j + 1$. If $j > N$ (N is the number of classes) go to Step 16, otherwise go to Step 13.
Step 16. End program.

The basic steps involved in the above training procedure are elaborated below:

Initialization: The weight vector $W = [w_1, w_2, \ldots, w_m]^{\mathrm{T}}$ of each neuron of the bottom layer SOM is randomly initialized with a value ranged between 0 and 1, where m is the total number of features used for child nodes. However, for the neurons of the CSOM, the dimension of the weight vector $W = [w_1, w_2, \ldots, w_k, p_1\{w_{x_1}, w_{y_1}\}, \quad p_2\{w_{x_2}, w_{y_2}\} \ldots p_{c_{\max}}\{w_{x_{c_{\max}}}, w_{y_{c_{\max}}}\}]^{\mathrm{T}}$ is equal to $(k + 2 \times c_{\max})$, where k is the number of features used for root nodes. From our practical experience, a good choice of $c_{\max}$ lies between Z and 2Z, provided $c_{\max} \geqslant C_{\max}$, where $C_{\max}$ is the maximum number of child nodes found on the database, and $Z \times Z$ is the size of the bottom layer SOM. In the case of $2Z < C_{\max}$, $c_{\max}$ can simply be set to $C_{\max}$. In our studied database, $C_{\max}$ is found to be 22 and $c_{\max}$ is set to 66.

Similarity matching and winner neuron: For an input node X, the best-matching neuron on the SOM is found by using the following maximum similarity criterion:

$$(x_{\mathrm{w}}, y_{\mathrm{w}}) = \arg \max_{x,y} S(X, W_{x,y}), \quad x, y = 1, 2, \ldots, Z, \tag{8}$$

where (x_w, y_w) is the position of the winner neuron on the SOM map, Z is the size of SOM layer, and S is the similarity measure between a root node X and the weight vector W of a neuron.

$$S(X, W) = \frac{1}{k}\sum_{i=1}^{k}\{1 - \mathrm{abs}(f_i - w_i)\} + \frac{1}{\sum_{j=1}^{c_{\max}} \mathbb{S}\left(p_j^x\right)} \sum_{j=1}^{c_{\max}} \mathbb{S}\left(p_j^x\right)\left\{1 - d\left(p_j^x, p_j^w\right)\right\} \tag{9}$$

where,

$$\mathbb{S}\left(p_j^x\right) = \begin{cases} 1 & \text{if } p_j^x \neq (0,0), \\ 0 & \text{otherwise,} \end{cases}$$

where $d(\cdot)$ is the Euclidean distance, p_j^x and p_j^w are the position vector in X and W, respectively. The first part in Eq. (9) is the similarity measure in global feature space, while the second part contains the similarity measure in the

local region-based feature space. It should be noted that these global and local similarities are weighted equally. Inclusion of the mean operator in each part enables them to be evaluated equally independent of the number of features or child positions. For the child nodes, the similarity function becomes

$$S(X, W) = \frac{1}{m}\sum_{i=1}^{m}\{1 - \mathrm{abs}(f_i - w_i)\}. \quad (10)$$

Updating: After the winner neuron is found for the input node X, the weight vectors are updated as follows:

$$W_{x,y}(t+1) = W_{x,y}(t) + \eta(t)\ h_{x,y,x_w,y_w}(t)\ [X(t) - W_{x,y}(t)],$$
$$x, y = 1, 2, \ldots, Z, \quad (11)$$

where t is the iteration number, $\eta(t)$ is the learning rate parameter, and $h(t)$ is the neighborhood function centered on winning neuron at position (x_w, y_w). Both $\eta(t)$ and $h(t)$ decrease gradually during the learning process.

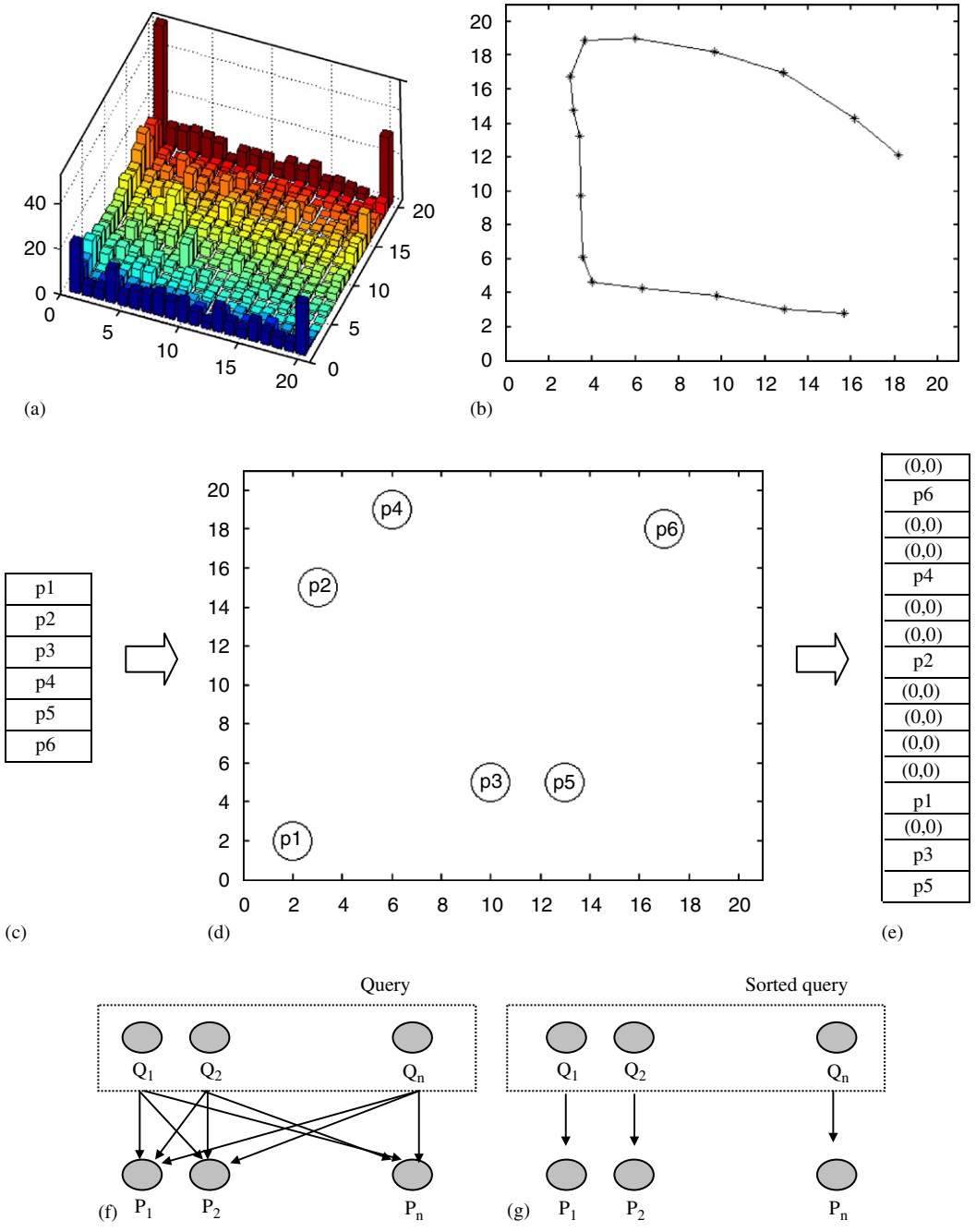

Fig. 4. Illustration of sorting position vectors using 1D SOM (a) mapping of child nodes on SOM, where the bars indicate the number of nodes, (b) weights of 1-dimensional SOM in data space (node's position) after training by the child nodes' positions and the line shows connection between two neighborhood neurons, (c) unsorted position vectors for a set of child nodes, (d) mapping of child nodes on the SOM, (e) sorted position vectors for the input vector of the root node. Illustration of distance measure for unsorted (f) and sorted (g) position vectors.

The neighborhood function is:

$$h_{x,y,x_w,y_w}(t) = \exp\left(-\frac{[(x-x_w)^2+(y-y_w)^2]}{2\sigma^2(t)}\right), \quad (12)$$

where $\sigma(t)$ is the width of the neighborhood function that decreases with iteration.

$$\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\tau_1}\right), \quad (13)$$

where σ_0 is the initial width, $\tau_1 = (\tau_2/\log\ \sigma_0)$, and τ_2 is the maximum number of iterations. The learning rate can be selected by

$$\eta(t) = \eta_0\ \exp\left(-\frac{t}{\tau_2}\right), \quad (14)$$

where η_0 is the initial learning rate.

Making inputs for root nodes: After the training is complete for the bottom layer SOM, each of the second level child nodes is associated with the best-matched neuron. Positions of these neurons $\{p_1, p_2, \ldots, p_c\}$ are combined with the global features of root nodes to make the inputs for the root nodes. Prior to the use of position vectors, they are normalized through dividing a position vector by the length of the square SOM grid. Before the position vectors are combined with the global features, the position vectors are sorted according to their spatial positions on the SOM map and are replaced with a fixed length vector $\{p_1, p_2, \ldots, p_{c_{\max}}\}$. The purpose of this sorting lies in the similarity matching between two position sets of children nodes that appears in the second part of Eq. (9). The basic idea is to compare a child node from the first set with only one similar child node from another set. Thus, by performing the sorting procedure, the two position sets can be compared by a meaningful and fast "one to one" matching (see Fig. 4(g), instead of "many to many" matching (see Fig. 4f). The sorting of the position vectors is conducted using a simple 1-dimensional SOM that has $c_{\max}$ number of neurons. This 1-dimensional SOM is trained by all the position vectors $(p = \{x,\ y\})$ of the child nodes over the database. The neurons' weights are saved for latter use. At last, the following sorting procedures are applied to obtain the sorted position vector, $\left[p_1^s, p_2^s, \ldots, p_{c_{\max}}^s\right]$. Fig. 4 shows that the trained weights of the neurons of the 1-dimensional SOM follow the data clusters in a spatially ordered sequence.

Fig. 5. Samples of the different classes of images.

Sorting process for a set of position vectors $\{p_1, p_2, \ldots, p_c\}$ **using 1D SOM of** $c_{\max}$ **neurons:**
Set $p_j^s \leftarrow [0,0]$, $j = 1, \ldots, c_{\max}$
Loop for **each** p_i, $i = 1 \ldots c$
done $\leftarrow 0$
Find winner neuron j for p_i
  While done $= 0$
    If $p_j^s = [0,0]$ (p_j^s is not occupied before)
      $p_j^s \leftarrow p_i$
      done $\leftarrow 1$
    else
      Find next best-matched neuron l and set $j \leftarrow l$
  End while
End

After the two-layer SOM network has been trained, the classification of a query image is quite simple. First, the tree data for the query image are extracted. Second, the child nodes are processed in the bottom SOM layer and the positions $\{p_1(x_1, y_1), p_2(x_2, y_2), \ldots, p_c(x_c, y_c)\}$ of their winner neurons are found. The sorted position vector $\left[p_1^s, p_2^s, \ldots, p_{c_{\max}}^s\right]$ is then formed using the 1-dimensional SOM described in the above procedures. Finally, appending this position vector to global features forms the input vector $\left[f_1, f_2, \ldots, f_k, \quad p_1^s, p_2^s, \ldots, p_{c_{\max}}^s\right]$ for the root node. The root node input is then compared at CSOM, where each SOM is labeled with one class only. The query is then assigned to the class of a SOM block containing the best-matched neuron.

4. Results

To evaluate the proposed approach, an image database consisting of 1000 images was used in this study. The images were categorized into 10 classes, each containing 100 images. The 10 classes are namely "people", "beach",

Table 2
Classification performances achieved by different feature sets

Classification performance (in percentage) on testing set											
	Average	Image classes									
		1	2	3	4	5	6	7	8	9	10
Flat feature sets											
FF1	69	62	66	56	84	100	70	88	76	32	56
FF2	70	64	64	60	98	98	58	98	84	22	56
FF3	68	64	60	60	84	98	70	86	70	30	54
Region feature sets											
RF1	51	42	14	6	80	54	56	96	74	8	76
RF2	63	56	46	40	86	92	60	84	62	34	70
RF3	56	46	4	34	62	64	56	86	98	22	88
Tree-structured feature sets											
TS1	80	80	64	54	82	100	82	96	100	48	92
TS2	81	80	50	72	92	100	82	94	98	48	90
TS3	78	76	36	64	78	100	86	90	100	52	94
TS4	78	74	52	66	84	100	86	84	100	48	88
TS5	80	78	52	60	86	100	80	90	100	60	92

Table 1
Lists of the features sets used in the experiments.

		Features
Flat feature sets		
FF1		Color moments + Gabor texture [$\boldsymbol{CM}_3$ $\boldsymbol{G}_{4,6}$]
FF2		Binary histogram + Gabor texture [$\boldsymbol{H}_{B64}$ $\boldsymbol{G}_{4,6}$]
FF3		Color histogram + Gabor texture [$\boldsymbol{H}_{64}$ $\boldsymbol{G}_{4,6}$]
Region feature sets		
RF1		Color moments + Gabor texture + shape + size [$\boldsymbol{CM}_2$ $\boldsymbol{G}_{1,6}$ $\boldsymbol{S}$ $\boldsymbol{A}$]
RF2		Color moments + Haar wavelet + shape + size [$\boldsymbol{CM}_2$ $\boldsymbol{T}$ $\boldsymbol{S}$ $\boldsymbol{A}$]
RF3		Color histogram + Gabor texture + shape + size [$\boldsymbol{H}_8$ $\boldsymbol{G}_{1,6}$ $\boldsymbol{S}$ $\boldsymbol{A}$]
Tree-structured feature sets	Global features	*Region features*
TS1	Color histogram [$\boldsymbol{H}_{16}$]	Color moments + Gabor texture + shape + size [$\boldsymbol{CM}_2$ $\boldsymbol{G}_{1,6}$ $\boldsymbol{S}$ $\boldsymbol{A}$]
TS2	Color histogram [$\boldsymbol{H}_{16}$]	Color moments + Haar wavelet + shape + size [$\boldsymbol{CM}_2$ $\boldsymbol{T}$ $\boldsymbol{S}$ $\boldsymbol{A}$]
TS3	Color histogram [$\boldsymbol{H}_{16}$]	Color histogram + Gabor texture + shape + size [$\boldsymbol{H}_8$ $\boldsymbol{G}_{1,6}$ $\boldsymbol{S}$ $\boldsymbol{A}$]
TS4	Color histogram [$\boldsymbol{H}_{16}$]	Color histogram + Haar wavelet + shape + size [$\boldsymbol{H}_8$ $\boldsymbol{T}$ $\boldsymbol{S}$ $\boldsymbol{A}$]
TS5	Color histogram [$\boldsymbol{H}_{16}$]	Color moments + Color histogram + Gabor texture + Haar wavelet + shape + size [$\boldsymbol{CM}_2$ $\boldsymbol{H}_8$ $\boldsymbol{G}_{1,6}$ $\boldsymbol{T}$ $\boldsymbol{S}$ $\boldsymbol{A}$]

The features used for each set shown on the right side.

"buildings", "busses", "dinosaurs", "elephants", "flowers", "horses", "mountains" and "food" (Fig. 5). Before extracting the features, each image was resized into 10,000 pixels (approximate) while maintaining their original width-height ratio. The image database was divided into two equal parts, namely training and testing image sets. In order to evaluate the generalization capability of the proposed approach, only the training set was used in the training process. The training and the testing sets are evaluated separately for the classification results.

We have also used a traditional set of flat image features extracted over the whole image. In addition, we compared our proposed approach with the purely region-based approach in which the same tree structure of our method is used without assigning global features to root node. Different feature combinations were investigated for use as region features in our approach. As a global feature, color histogram was used as the root node's attribute. Table 1 summarizes only the best-performing feature sets, while others are excluded due to their poor classification performance. SOMs are trained separately for evaluating each feature set, and the features used in each set are detailed in Table 1. The flat image features include color histogram [8,12], binary histogram [8], Gabor texture

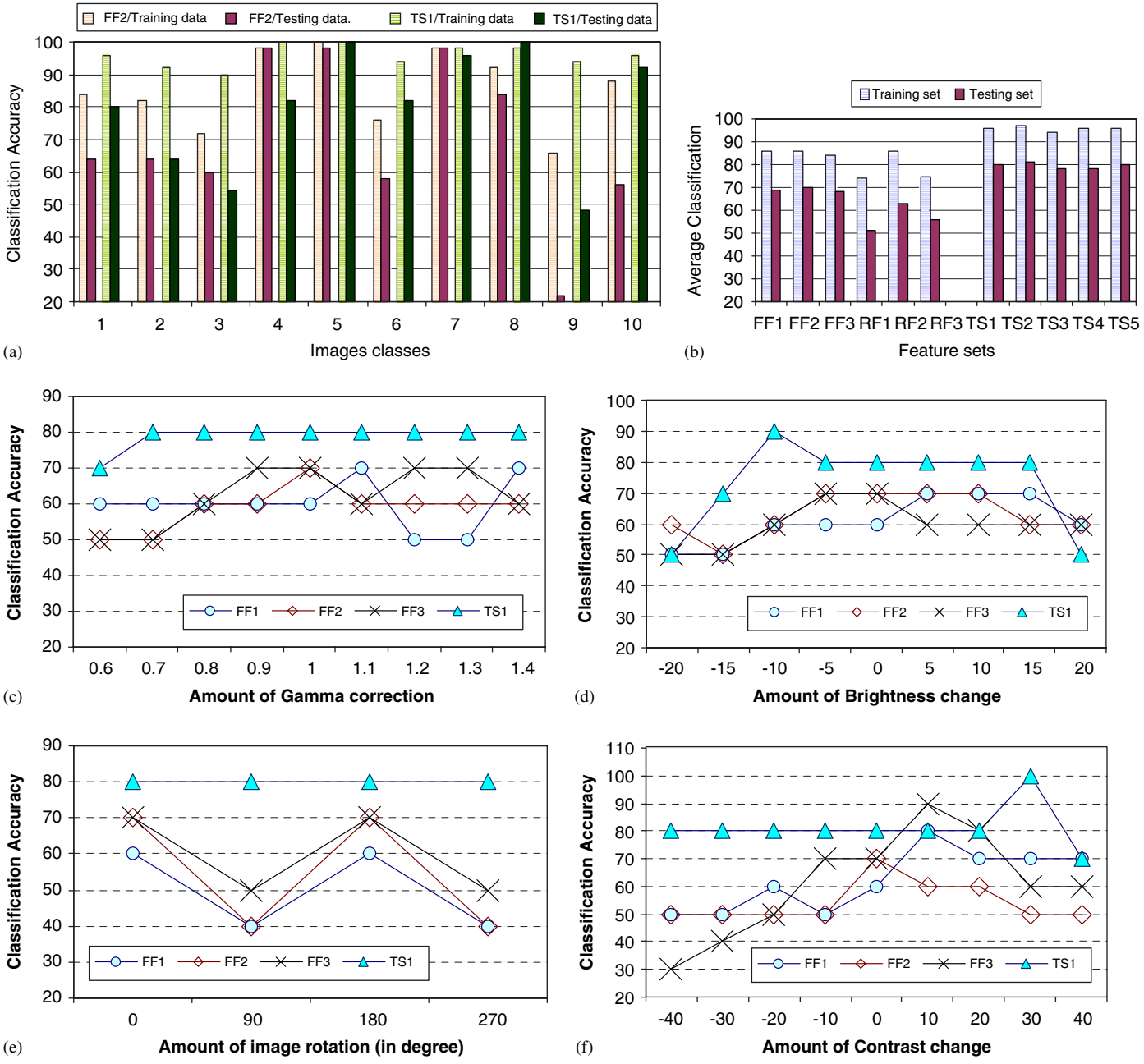

Fig. 6. (a) Comparative performance of FF2 and TS1 on every class of training and testing sets, (b) average performances of all feature sets on training and testing sets, and (c–f) classification accuracy under different kinds of image alterations: (c) gamma correction, (d) brightness change, (e) image rotation, and (f) contrast change.

features [7,10], and color moments [22]. For color and binary histograms, 64 bins were used for each of the HSV channels that were found the best performing for this data set. As the common practice, 4 scales and 6 orientations were used for Gabor texture features. Means and standard deviations of the complex-valued output from Gabor filters were used that make the feature dimensions 48. For the flat global features, 9 color moments of a different order from the 3 HSV color channels were used. The first moment stored the average color of the images. The second and the third moments stored the standard deviation and skewness of each color. However, we used smaller dimensions of features in the proposed tree structure to reduce the computational cost. Only 16 and 8 bins were used in the color histogram for the global and the region feature, respectively. Also, Gabor texture features were calculated using 1 scale and 6 orientations for the regions. Three shapes and one size feature (region based) were used in all combinations of the tree-structured features. For all the flat feature sets, a one-layer CSOM classifier was used. It should be noted that the additional bottom layer SOM used in the proposed approach was only to encode the image regions, but not for performing classification. A 40×40 size was used in the bottom layer SOM for region encoding and 10×10 size was used for each SOM in the top layer CSOM. In our experiment, the above SOM sizes appeared to be a good choice in compromising the performance and the computational cost. For all the cases, training was run for $(10 \times N_d)$ iterations (N_d is the number of training data) with the initial learning rate set to 0.3 and the initial radius of the neighborhood function set to half of the lattice dimension.

Table 2 summarizes the image classification results on different image classes using different feature sets. As shown in the table, the average performance of the tree-structured feature sets on the testing sets are around 80%, while the flat feature sets achieve around 70% and the purely region feature sets are around 60%. It is worth analyzing the class results using the image semantics of different classes. It is noticed from Fig. 5 that the image classes 2, 3, and 9 are scenery-like images where a high variation of semantics took place. The rest of the 10 classes are close images focusing on specific objects like human, elephants, flowers, horses, etc. Apparently, objects that appeared in the latter group contain almost unique features that are well perceived using region-based approach. Table 2 indicates that the region-based features perform moderately well on object-oriented close images. But they perform badly on the scene-like images, whereas global flat features perform much better. Fig. 6(a) compares the performance of a flat feature set (FF2) with the tree-structured feature set (TS1) for each image class, both on the training and the testing image sets. Fig. 6(b) summarizes the average performance over all the classes of different feature sets on the training and the testing sets. It is observed for both the training and the testing sets that performances of all the tree-structured feature sets are about 10% higher than that of the flat feature sets and 10–20% higher than that of the purely region feature sets. The obtained results using tree-structured feature sets show clear improvement over the traditional flat features because the flat features are crude representations of the image contents. The flat features are unable to deliver object-based image properties, while the proposed tree-structured features are able to encode the image contents in a better way by integrating the global and the region-based image characteristics.

Table 3
The relative computational time required for different feature sets

	Training time (min)	Testing time per query (sec)
Flat/Region feature sets		
FF1	1.7813	0.0907
FF2	14.1882	1.2154
RF2	30.7367	0.1532
Tree-structured feature sets		
TS1	71.3647	0.4249

The computational times for different feature sets are detailed in Table 3. All the implementations were performed in a Pentium III 1 GHz PC using MATLAB. The proposed approach requires additional training time for image regions. The testing time per query is, however, negligible, which makes the proposed approach suitable for real-time application. At last, the robustness of the proposed approach is tested against different types of image alterations. The images of the same scene can be different depending on the camera settings and the environmental conditions. For instance, image blurring can take place due to the improper camera focus, darkening, or brightening due to improper exposure/lighting conditions, etc. The performance evaluation was then further extended under different kinds of image alteration conducted by the software "FotoCanvas, version 1.1, ACD system" [23]. To evaluate the performance under these image alterations, image samples were randomly selected from all classes and the average classification accuracy on each alteration was calculated. Fig. 6(c–f) shows the classification performance against the gamma correction, brightness change, image rotation, and contrast change. The obtained results corroborate the robustness of the tree-structured features under those conditions.

5. Conclusion

An improved image classification approach is proposed by integrating both global and local image features through a two-level tree. The child nodes of the tree contain the region-based local features, while the root node contains the global features. The region-based features exhibit object-oriented properties of image contents that are overlooked by the global image features. In the proposed architecture, a two-layer SOM network is used to process

the tree-integrated features. When the system is performing image classification, this type of arrangement enables us to process object-based properties of the image contents as well as the global image features. As a result, the proposed approach can enhance the classification performance. Also, it is worth noting that this approach is not computationally demanding, which makes it suitable for real-time application. Our study shows that tree-structured features are robust under different kinds of image alteration.

Acknowledgment

The work described in this paper was fully supported by a grant from the City University of Hong Kong [SRG: 7001599-570].

References

[1] S.Y. Cho, Z. Chi, Genetic evolution processing of data structures for image classification, IEEE Trans. Knowl. Data Eng. 17 (2) (2005) 216–231.
[2] S.Y. Cho, Z. Chi, W.C. Siu, A.C. Tsoi, An improved algorithm for learning long-term dependency problem in adaptive processing of data structures, IEEE Trans. Neural Networks 14 (2003) 781–793.
[3] C. Carson, M. Thomas, S. Belongie, J.M. Hellerstein, J. Malik, Blobworld: a system for region-based image indexing and retrieval, in: D.P. Huijsmans, A.W.M. Smeulders (Eds.), Third International Conference on Visual Information Systems, Amsterdam, the Netherlands, Springer, 1999.
[4] I. Daubechies, Ten Lectures on Wavelets, Capital City Press, 1992.
[5] Y. Deng, B.S. Manjunath, Unsupervised segmentation of color-texture regions in images and video, IEEE Trans. Pattern Anal. Mach. Intell. 23 (8) (2001) 800–810.
[6] Y. Deng, B. S. Manjunath, H. Shin, Color image segmentation, in: Proceedings of the CVPR '99, Vol. 2, Fort Collins, CO, (1999, pp. 446-451.
[7] D.J. Field, Relations between the statistics of natural images and the response properties of cortical cells, J. Opt. Soc. Am. A 4 (12) (1987) 2379–2394.
[8] L. Kunttu, L. Lepistö, J. Rauhamaa, A. Visa, Binary histogram in image classification for retrieval purposes, J. WSCG 11 (1) (2003).
[9] W.Y. Ma, B. Manjunath, NaTra: a toolbox for navigating large image databases, in: Proceedings of the IEEE International Conference on Image Processing, 1997, pp. 568–571.
[10] B.S. Manjunath, W.Y. Ma, Texture features for browsing and retrieval of large image data, IEEE Trans. Pattern Anal. Mach. Intell. (special issue on Digit. Libr.) 18 (8) (1996) 837–842.
[11] V.E. Neagoe, A.D. Ropot, Concurrent self-organizing maps for pattern classification, in: Proceedings of the First IEEE International Conference on Cognitive Informatics, 2002, pp. 304–312.
[12] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, G. Taubin, The QBIC project: querying images by content using color, texture, and shape, in: Proceedings of the SPIE: Storage and Retrieval for Image and Video Database, Vol. 1908, 1993, pp. 173, 187.
[13] V.E. Ogle, M. Stonebraker, Chabot: retrieval from a relational database of images, IEEE Comput. 28 (9) (1995) 40–48.
[14] G. Pass, R. Zabih, Histogram refinement for content-based image retrieval, in: Proceedings of the IEEE Workshop on Applications of Computer Vision, Sarasota, FL, 1996.
[15] H. Radha, M. Vetterli, R. Leonardi, Image compression using binary space partitioning trees, IEEE Trans. Image Process. 5 (12) (1996) 1610–1624.
[16] Y. Rubner, L.J. Guibas, C. Tomasi, The earth mover's distance, Shimulti-dimensional scaling, and color-based image retrieval, in: Proceedings of the ARPA Image Understanding Workshop, New Orleans, LA, 1997, pp. 661–668.
[17] E. Saber, A.M. Tekalp, Integration of color, edge, shape, and texture features for automatic region-based image annotation and retrieval, J. Electron. Imaging 7 (3) (1998) 684–700.
[18] P. Salembier, L. Garrido, Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval, IEEE Trans. Image Process. 9 (4) (2000) 561–576.
[19] A. Sanfeliu, R. Alquézar, J. Andrade, J. Climent, F. Serratosa, J. Vergés, Graph-based representations and techniques for image processing and image analysis, Pattern Recogn. 35 (2002) 639–650.
[20] J.R. Smith, S.F. Chang, VisualSEEk: a fully automated content-based image query system, in: Proceedings of the ACM Multimedia 96, no. 11, Boston, MA, November 1996.
[21] J.R. Smith, C.S. Li, Image classification and querying using composite region templates, Comput. Vision Image Understanding 75 (1/2) (1999) 165–174.
[22] M. Stricker, M. Orengo, Similarity of color images, in: Proceedings of the SPIE on Storage and Retrieval for Image and Video Databases, vol. 2420, San Jose, USA, 1995, pp. 381–392.
[23] ACD Systems: http://www.acdsystems.com/.
[24] J.Z. Wang, J. Li, G. Wiederhold, SIMPLIcity: semantics sensitive integrated matching for picture libraries, IEEE Trans. Pattern Anal. Mach. Intell. 23 (2001) 947–963.
[25] J.Z. Wang, G. Wiederhold, O. Firschein, X.W. Sha, Content-based image indexing and searching using Daubechies' wavelets, Int. J. Digital Librar. 1 (4) (1998) 311–328.
[26] X. Wu, Image coding by adaptive tree-structured segmentation, IEEE Trans. Inf. Theory 38 (6) (1992) 1755–1767.

Tommy W.S. Chow (IEEE M'93–SM'03) received the B.Sc. (First Hons.) and Ph.D. degrees from the University of Sunderland, Sunderland, U.K. He joined the City University of Hong Kong, Hong Kong, as a Lecturer in 1988. He is currently a Professor in the Electronic Engineering Department. His research interests include machine fault diagnosis, HOS analysis, system identification, and neural network learning algorithms and applications.

M.K.M. Rahman received his B.Eng. degree in the Department of Electrical and Electronic Engineering from Bangladesh University of Engineering and Technology in 2001. He is currently working towards the Ph.D. degree at City University of Hong Kong, Hong Kong. His research interests are structural data processing, neural network, pattern recognition, and their applications.