

## Chapter 2

# Kernel-Based Adaptive Image Retrieval Methods

**Abstract** This chapter presents machine learning methods for adaptive image retrieval. In a retrieval session, a nonlinear kernel is applied to measure image relevancy. Various new learning procedures are covered and applied specifically for adaptive image retrieval applications. These include the adaptive radial basis function (RBF) network, short term learning with the gradient-decent method, and the fuzzy RBF network. These methods constitute the likelihood estimation corresponding to visual content in a short-term relevance feedback (STRF). The STRF component can be further incorporated in a fusion module with contextual information in long-term relevance feedback (LTRF) using the Bayesian framework. This substantially increases retrieval accuracy.

## 2.1 Introduction

Adaptation of the traditional similarity function plays a vital role in enhancing the capability of image retrieval and broadening the domain of applications for machine learning. In particular, it is often necessary to adapt the traditional Euclidean inner-product to the more flexible and nonlinear inner products characterized by relevance feedback parameters. The new inner products lead to a new similarity metric. As a result, the image retrieval has to be necessarily conducted in a new space that is adaptively re-defined in accordance with different user preferences. This implies a greater flexibility for image retrieval. The topics addressed in this chapter are as follows:

Section 2.2 will look into the *linear kernel* that is implemented through the query adaptation method, metric adaptation method, and a combination of these methods. In a linear-based adaptive retrieval system, the similarity score of a pair of vectors may be represented by their inner product or Mahalanobis inner product.

Depending on the data cluster structure, either linear or nonlinear inner products may be used to characterize the similarity metric between two vectors. The linear metric would be adequate if the data distribution is relatively simple. To handle more complex data distributions, it is often necessary to adopt nonlinear inner products prescribed by nonlinear kernel functions, e.g., the *Gaussian radial basis function* (RBF). Section 2.3 introduce a single-class RBF method for adaptive retrieval.

To cope with the small size of training sample sets and convergence speed, new learning methods are required for the construction of the RBF network, instead of the direct application of traditional learning procedures. Section 2.4 introduces an adaptive RBF network to exploit the local context defied by query sessions, and aids in improving retrieval accuracy. This section follows by the optimization of network parameters by the gradient-descent-based learning procedure, then introducing fuzzy RBF network which offers a soft-decision choice to the users.

Section 2.5 establishes the fusion of content and context information, by the application of Bayesian theory. The content component is gathered from a short-term relevance feedback (STRF), which is the estimation of the likelihood of a specific query model. The context information is obtained by a long-term relevance feedback (LTRF), representing a user history or the *a priori* information.

## 2.2 Kernel Methods in Adaptive Image Retrieval

### 2.2.1 Adaptive Retrieval Framework

The most important part in the adaptive process is the analysis of the role of the user in perceiving image similarity according to preferred image selections. This is implemented by a mapping function,  $f_q : \mathbb{R}^P \rightarrow \mathbb{R}$ , which is given by:

$$y_q = f_q(\mathbf{x}) \quad (2.1)$$

where  $\mathbf{x} = [x_1, \dots, x_P]^T$  is called a feature vector in a  $P$ -dimensional Euclidean space  $\mathbb{R}^P$ , corresponding to an image in the database. The main procedure is to obtain the mapping function  $f_q$  (for the query class  $q$ ) from a small set of training images,  $\mathcal{T} = \{(\mathbf{x}_1, l_1), (\mathbf{x}_2, l_2), \dots, (\mathbf{x}_N, l_N)\}$ , where the class label  $l_i$  can be in binary or non-binary form. In the binary form, the training samples contains a set of positive samples,  $\mathcal{X}^+$  and a set of negative samples,  $\mathcal{X}^-$ :

$$\mathcal{T} = \mathcal{X}^+ \cup \mathcal{X}^- \quad (2.2)$$

$$\mathcal{X}^+ = \{\mathbf{x}'_i | l_i = 1\}, \quad i = 1, \dots, N_p \quad (2.3)$$

$$\mathcal{X}^- = \{\mathbf{x}''_j | l_j = 0\}, \quad j = 1, \dots, N_n \quad (2.4)$$

where  $N_p$  and  $N_n$  are the numbers of positive and negative samples, respectively.

The adaptive process for constructing the mapping function for retrieval is summarized in Table 2.1.

**Table 2.1** Summary of the adaptive retrieval algorithm

<i>Input:</i>	Query vector = $\mathbf{x}_q$ Set of vectors to be searched in the database = $\mathbf{x}_n, n = 1, \dots, T$
<i>Output:</i>	The final retrieval set, containing $k$ -relevant samples = $S_k(\mathbf{x}_q)$
<i>Computation:</i>	$d(\mathbf{x}_q, \mathbf{x}_n) = \left[ \sum_i^P  x_{qi} - x_{ni} ^2 \right]^{\frac{1}{2}}, n = 1, 2, \dots, T,$ $S_k(\mathbf{x}_q) = \{\mathbf{x}   d(\mathbf{x}_q, \mathbf{x}) \leq d(\mathbf{x}_q, \mathbf{x}_k)\}$ <p>where <math>S_k(\mathbf{x}_q)</math> is the set of nearest neighbors, and <math>\mathbf{x}_k</math> is the <math>k</math>-th nearest neighbor of <math>\mathbf{x}_q</math>.</p>
<i>Repeat:</i>	Obtain training sample: $\{\mathbf{x}_i\}_{i=1}^N \leftarrow S_k(\mathbf{x}_q)$ User selects class label: $l_i$ Calculate model parameters of the mapping function $f_q$ Calculate $f_q(\mathbf{x}_n)$ , for $n = 1, 2, \dots, T$ , and obtain $S_k(\mathbf{x}_q) = \{\mathbf{x}   f_q(\mathbf{x}) \geq f_q(\mathbf{x}_k)\}$
<i>Until:</i>	User is satisfied with the retrieval result.

### 2.2.2 Query Adaptation Method

Among the early attempts to conduct adaptive retrieval, Rui et al. [11, 12] implemented the query modification strategy, and the mapping function takes the form of the following linear function:

$$f_q(\mathbf{x}) = \frac{\mathbf{x} \cdot \mathbf{x}_{\hat{q}}}{\|\mathbf{x}\| \|\mathbf{x}_{\hat{q}}\|} \quad (2.5)$$

$$\propto K(\mathbf{x}, \mathbf{x}_{\hat{q}}) \quad (2.6)$$

where  $K$  is the linear kernel function:

$$K(\mathbf{x}, \mathbf{x}_{\hat{q}}) \equiv \langle \mathbf{x}, \mathbf{x}_{\hat{q}} \rangle \equiv \mathbf{x} \cdot \mathbf{x}_{\hat{q}} \quad (2.7)$$

and  $\mathbf{x} \cdot \mathbf{x}_{\hat{q}}$  denotes the Euclidean inner product,  $\mathbf{x}_{\hat{q}} = [x_{\hat{q}1}, \dots, x_{\hat{q}P}]^T$  is the modified query vector, and  $\|\cdot\|$  is the Euclidean norm. The linear kernel function represents the similarity metric for a pair of vectors,  $\mathbf{x}$  and  $\mathbf{x}_{\hat{q}}$ . The two vectors,  $\mathbf{x}$  and  $\mathbf{x}_{\hat{q}}$

are called orthogonal if  $\langle \mathbf{x}, \mathbf{x}_{\hat{q}} \rangle = 0$  in which case we write  $\mathbf{x} \perp \mathbf{x}_{\hat{q}}$ , i.e. they are geometrically perpendicular. Given two vectors, the smaller the magnitude of their inner-product, the less similar they are.

The modified query vector  $\mathbf{x}_{\hat{q}}$  discussed in Eq. (2.7), is obtained by the training samples as:

$$\mathbf{x}_{\hat{q}} = \alpha \mathbf{x}_q + \beta \left( \frac{\sum_{i=1}^{N_p} \mathbf{x}'_i}{N_p} \right) - \varepsilon \left( \frac{\sum_{i=1}^{N_n} \mathbf{x}''_i}{N_n} \right) \quad (2.8)$$

where  $\mathbf{x}_q = [x_{q1}, \dots, x_{qP}]^T$  denotes the original query vector, and  $(\alpha, \beta, \varepsilon)$  are suitable parameters [13]. The new query is obtained by adjusting the positive and negative *terms* of the original query. When adding the positive terms to the query, the modified query is close to the mean of the positive samples (i.e.,  $\mathbf{x}_{\hat{q}} \cong \bar{\mathbf{x}}'$ ), and the inner product  $\langle \mathbf{x}', \mathbf{x}_{\hat{q}} \rangle \cong 1$ . On the other hand, subtracting the negative terms from the query will make the modified query more dissimilar to the negative samples.

The query modification method has been widely used for information retrieval [13, 107] and image retrieval systems [14, 103]. However, one disadvantage of this model is the requirement of an indexing structure to follow term-weighting model, as in text retrieval for greater effectiveness. The models assume that the query index terms are sparse and are usually of a binary vector representation. However, as compared to text indexing, image feature vectors are mostly real vectors. Thus, a large number of terms can be applied for characterization of images in order to overcome this problem [103]. This also increases computational complexity.

### 2.2.3 Metric Adaptation Method

The Euclidean inner-product may be extended as the Mahalanobis inner product

$$K(\mathbf{x}, \mathbf{x}_q) = \langle \mathbf{x}, \mathbf{x}_q \rangle_{\mathbf{M}} = \mathbf{x}^T \mathbf{M} \mathbf{x}_q \quad (2.9)$$

with a weight matrix  $\mathbf{M}$ . The Euclidean inner product is a special case of the Mahalanobis inner product with  $\mathbf{M} = \mathbf{I}$ . In this case, we assume that all the features are equally weighted in their importance, and there exists no inter-feature dependence. However, when the features are mutually independent, but not isotropic, the Mahalanobis matrix takes the following form

$$\mathbf{M} = \text{Diag} \{w_i\}, \quad i = 1, \dots, P \quad (2.10)$$

where the weights  $\{w_i, i = 1, \dots, P\}$  reflect the importance of the respective features.

The Mahalanobis inner product leads to the following Mahalanobis distance between  $\mathbf{x}$  and  $\mathbf{x}_q$ , which is associated as the mapping function as in [17, 19–24, 103],

$$f_q(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_q\|_{\mathbf{M}} \equiv \sqrt{(\mathbf{x} - \mathbf{x}_q)^t \mathbf{M} (\mathbf{x} - \mathbf{x}_q)} \quad (2.11)$$

$$= \left( \sum_{i=1}^P w_i (x_i - x_{qi})^2 \right)^{\frac{1}{2}} \quad (2.12)$$

$$\equiv \left( \sum_{i=1}^P h(d_i) \right)^{\frac{1}{2}} \quad (2.13)$$

where  $\mathbf{x}_q = [x_{q1}, \dots, x_{qP}]^t$  is the feature vector of the query image, and  $h(d_i)$  denotes a transfer function of distance  $d_i = |x_i - x_{qi}|$ . The weight parameters  $\{w_i, i = 1, \dots, P\}$  are called relevance weights, and  $\sum_i w_i = 1$ . The weight parameters can be calculated by the standard deviation criterion [17, 20, 21] or a probabilistic feature relevance method [16].

Different types of distance function have also been exploited. These include the selection of Minkowski distance metrics according to a minimum distance within the positive class [23], the selection of metrics based on reinforcement learning [22] and on the interdependencies between feature elements [25].

### 2.2.4 Query and Metric Adaptive Method

In order to reduce time for convergence, the adaptive systems have been designed to combine the query reformulation model with the adaptive similarity function [26–30]. Apart from Eq. (2.8), the query modification model can be obtained by a linear discrimination analysis [30], and a probabilistic distribution analysis methods applied to the training samples [28].

The optimum solutions for query model and similarity function can be obtained by the optimal learning relevance feedback (OPT-RF) method [26]. The optimum solution for a query model, obtained by Lagrange multiplier, is given by the weighted average of the training samples:

$$\mathbf{x}_{\hat{q}}^t = \frac{\mathbf{v}^t \mathbf{X}}{\sum_{i=1}^N v_i} \quad (2.14)$$

where  $\mathbf{x}_{\hat{q}} = [x_{\hat{q}1}, \dots, x_{\hat{q}P}]^t$  denotes the new query,  $\mathbf{v} = [v_1, v_2, \dots, v_N]^t$ ,  $v_i$  is the degree of relevance for the  $i$ -th training sample given by the user,  $\mathbf{X}$  is the training sample matrix, obtained by stacking the  $N$  training vectors into a matrix, i.e.,  $\mathbf{X} = [\mathbf{x}_1 \dots \mathbf{x}_N]^t$ . The optimum solution for the weight matrix  $\mathbf{M}$  is obtained by:

$$\mathbf{M} = \begin{cases} (\det(C))^{\frac{1}{k}} C^{-1} & \text{if } \det(C) \neq 0 \\ \text{Diag} \left\{ \frac{1}{C_{11}}, \frac{1}{C_{22}}, \dots, \frac{1}{C_{PP}} \right\} & \text{Otherwise} \end{cases} \quad (2.15)$$

where  $C$  denotes the weight covariance matrix, given by:

$$C_{rs} = \frac{\sum_{i=1}^N v_i (x_{ir} - x_{\hat{q}r}) (x_{is} - x_{\hat{q}s})}{\sum_{i=1}^N v_i}, \quad r, s = 1, \dots, P \quad (2.16)$$

Based on Eq. (2.15), the weight matrix is switched between a full matrix and a diagonal matrix. This overcomes possible singularities when the number of training samples,  $N$ , is smaller than the dimensionality of the feature space,  $P$ .

Table 2.2 gives a summary of the OPT-RF method, where the relevance feedback process is conducted after the initial search.

**Table 2.2** Summary of the optimal learning relevance feedback algorithm

<i>Input:</i>	Query vector = $\mathbf{x}_q$
	Set of vectors to be searched in the database = $\mathbf{x}_n, n = 1, \dots, T$
	The training samples = $\{\mathbf{x}_i\}_{i=1}^N$
<i>Output:</i>	The final retrieval set, containing $k$ -relevant samples = $S_k(\mathbf{x}_{\hat{q}})$
<i>Repeat:</i>	User provides relevance scores of training samples, $v_1, v_2, \dots, v_N$
	Calculate new query: $\mathbf{x}_{\hat{q}}^t = \frac{\mathbf{v}^t \mathbf{X}}{\sum_{i=1}^N v_i}$
	Calculate weight parameter:
	$\mathbf{M} = \begin{cases} (\det(C))^{\frac{1}{k}} C^{-1} & \text{if } \det(C) \neq 0 \\ \text{Diag} \left\{ \frac{1}{C_{11}}, \frac{1}{C_{22}}, \dots, \frac{1}{C_{PP}} \right\} & \text{Otherwise} \end{cases}$
	Calculate $f_{\hat{q}}(\mathbf{x}_n) = \left( (\mathbf{x}_n - \mathbf{x}_{\hat{q}})^t \mathbf{M} (\mathbf{x}_n - \mathbf{x}_{\hat{q}}) \right)^{\frac{1}{2}}$ , for $n = 1, 2, \dots, T$ , and obtain
$S_k(\mathbf{x}_{\hat{q}}) = \{\mathbf{x}   f_{\hat{q}}(\mathbf{x}) \leq f_{\hat{q}}(\mathbf{x}_k)\}$	
where $S_k(\mathbf{x}_{\hat{q}})$ is the set of nearest neighbors and $\mathbf{x}_k$ is the $k$ -th nearest neighbor of $\mathbf{x}_{\hat{q}}$ .	
$\{\mathbf{x}_i\}_{i=1}^N \leftarrow S_k(\mathbf{x}_{\hat{q}})$	
<i>Until:</i>	User is satisfied with the retrieval result.

### 2.2.5 Nonlinear Model-Based Adaptive Method

The methods outlined above are referred to as linear-based learning and this restricts the mapping function to quadratic form, which cannot cope with a complex decision boundary. For example, the one-dimensional distance mapping function  $h(d_i)$  in Eq. (2.13) may take the following form:

$$h(d_i) = w_i d_i^2 \quad (2.17)$$

where  $d_i = |x_i - x_{qi}|$ . This function has a small degree of nonlinear behaviour, i.e.,

$$\frac{\partial f_q(\mathbf{x})}{\partial d_i} = 2w_i d_i \quad (2.18)$$

where  $w_i$  is *fixed* to a numerical constant for the respective feature dimension.

To simulate human perception, a radial basis function (RBF) network [31, 45] is employed in this chapter. The input–output mapping function,  $f(\mathbf{x})$ , is employed on the basis of a method called *regularization* [32]. In the context of a mapping problem, the idea of regularization is based on the *a priori* assumption about the form of the solution (i.e., the input–output mapping function  $f(\mathbf{x})$ ). In its most common solution, the input–output mapping function is *smooth*, in the sense that similar inputs correspond to similar outputs. In particular, the solution function that satisfies this regularization problem is given by the expansion of the radial basis function [33]. In this case, a new inner product is expressed as a nonlinear kernel function  $K(\mathbf{x}, \mathbf{z})$ :

$$\langle \mathbf{x}, \mathbf{z} \rangle = K(\mathbf{x}, \mathbf{z}) \quad (2.19)$$

The Gaussian-shaped radial basis function is utilized:

$$K(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{z}\|^2}{2\sigma^2}\right) \quad (2.20)$$

where  $\mathbf{z}$  denotes the center of the function and  $\sigma$  denotes its width. The activity of function  $K(\mathbf{x}, \mathbf{z})$  is to perform a Gaussian transformation of the distance  $\|\mathbf{x} - \mathbf{z}\|$ , which describes the degree of similarity between the input  $\mathbf{x}$  and center of the function. Under Gaussian distribution, this function reflects the likelihood that a vector  $\mathbf{x}$  may be mistaken to be another vector  $\mathbf{z}$ .

To estimate the input–output mapping function  $f(\mathbf{x})$ , the Gaussian RBF is expanded through both its center and width, yielding different RBFs which are then formed as an RBF network. Its expansion is implemented via a learning process, where the expanded RBFs can modify weighting, to capture user perception.

### 2.3 Single-Class Radial Basis Function Based Relevance Feedback

Whist in the later sections in this chapter, the  $P$ -dimensional RBF function is explored, in this section, a one-dimensional Gaussian-shaped RBF applied for the distance function  $h(d_i)$  in Eq. (2.13), i.e.,

$$f_q(\mathbf{x}) = \sum_{i=1}^P G(x_i, z_i) \quad (2.21)$$

$$= \sum_{i=1}^P \exp\left(-\frac{(x_i - z_i)^2}{2\sigma_i^2}\right) \quad (2.22)$$

where  $\mathbf{z} = [z_1, z_2, \dots, z_P]^t$  is the center of the RBF,  $\sigma = [\sigma_1, \sigma_2, \dots, \sigma_P]^t$  is the tuning parameter in the form of RBF width. Each RBF unit implements a Gaussian transformation which constructs a local approximation to a nonlinear input–output mapping. The magnitude of  $f_q(\mathbf{x})$  represents the similarity between the input vector  $\mathbf{x}$  and the center  $\mathbf{z}$ , where the highest similarity is attained when  $\mathbf{x} = \mathbf{z}$ .

Each RBF function is characterized by two adjustable parameters, the tuning parameters and the adjustable center:

$$\{\sigma_i, z_i\}_{i=1}^P \quad (2.23)$$

This results in a set of  $P$  basis functions,

$$\{G_i(\sigma_i, z_i)\}_{i=1}^P \quad (2.24)$$

The parameters are estimated and updated via learning algorithms. For a given query class, some pictorial features exhibit greater importance or *relevance* than others in the proximity evaluation [16, 30]. Thus, the expanded set of tuning parameters,  $\sigma = [\sigma_1, \sigma_2, \dots, \sigma_P]^t$  controlled the weighting process according to the relevance of individual features. If the  $i$ -th feature is highly relevant, the value of  $\sigma_i$  should be small to allow greater sensitivity to any change of the distance  $d_i = |x_i - z_i|$ . In contrast, a large value of  $\sigma_i$  is assigned to the non-relevant features. Thus, the magnitude of the corresponding function  $G_i$  is approximately equal to unity regardless of the distance  $d_i$ .

#### 2.3.1 Center Selection

The selection of query location is done by a modified version of the learning quantization (LVQ) method [31]. In the LVQ process, the initial vectors (in a codebook), referred to as Voronoi vectors, are modified in such a way that all



points partitioned in the same Voronoi cells have the minimum (overall) encoding distortion. The movement of the Voronoi vectors is based on the class labels provided in a training set, so as to improve the accuracy of classification. Let  $\{\mathbf{z}_j\}_{j=1}^J$  denote the set of Voronoi vectors. Also, let  $\{\mathbf{x}_i\}_{i=1}^N$  denote the set of training samples. First, for the input vector  $\mathbf{x}_i[t]$  at iteration index  $t$ , the class index  $c(\mathbf{x}_i)$  of the best-matching Voronoi vector  $\mathbf{z}_c$  is identified by:

$$c = \arg \min_j \{ \|\mathbf{x}_i - \mathbf{z}_j\| \} \quad (2.25)$$

The Voronoi vector  $\mathbf{z}_c$  is modified by the *reinforced learning* rule if the class indexes of  $\mathbf{z}_c$  and  $\mathbf{x}_i$  are in agreement,

$$\mathbf{z}_c[t+1] = \mathbf{z}_c[t] + \alpha[t](\mathbf{x}_i[t] - \mathbf{z}_c[t]) \quad (2.26)$$

Otherwise, the modification is obtained by the *anti-reinforced learning* rule:

$$\mathbf{z}_c[t+1] = \mathbf{z}_c[t] - \alpha[t](\mathbf{x}_i[t] - \mathbf{z}_c[t]) \quad (2.27)$$

where  $\alpha[t]$  is the leaning constant, which decreases monotonically with the number of iterations. All other Voronoi vectors remain unchanged, except the best-matching Voronoi vector.

In the adaptive image retrieval process, we have the training samples with two-class labels,  $\{\mathbf{x}_i, l_i\}_{i=1}^N, l_i \in \{0, 1\}$ , associated with the query vector,  $\mathbf{x}_q$ . This training set represents the set of points closest to the query, according to the distance calculation in the previous search operation. Consequently, each data point can be regarded as the vector that is *closest* to the Voronoi vector. Therefore, following the LVQ algorithm, it is observed that all points in this training set are used to modify only the best-matching Voronoi vector, that is,  $\mathbf{z}_c \equiv \mathbf{x}_q$ .

*Center shifting model 1:* The first model approximates the Voronoi vector (after the convergence) by the position that is close to the data points that are in the positive class ( $l_i = 1$ ), and away from those points that are in the negative class ( $l_i = 0$ ):

$$\mathbf{z}_c^{new} = \mathbf{z}_c^{old} + \alpha_R (\bar{\mathbf{x}}' - \mathbf{z}_c^{old}) - \alpha_N (\bar{\mathbf{x}}'' - \mathbf{z}_c^{old}) \quad (2.28)$$

$$\bar{\mathbf{x}}' = \frac{\sum_{i=1}^{N_p} \mathbf{x}_i'}{N_p} \quad (2.29)$$

$$\bar{\mathbf{x}}'' = \frac{\sum_{i=1}^{N_n} \mathbf{x}_i''}{N_n} \quad (2.30)$$

where  $\mathbf{z}_c^{old}$  is the previous RBF center,  $\mathbf{x}_i', i = 1, \dots, N_p$  are the positive samples,  $\mathbf{x}_i'', i = 1, \dots, N_n$  are the negative samples,  $\alpha_R$  and  $\alpha_N$  are suitable positive constants.

*Center shifting model 2:* We may reduce the procedural parameters and provide a direct movement of the RBF center towards the positive class. Equation (2.28) is reduced to:

$$\mathbf{z}_c^{new} = \bar{\mathbf{x}}' - \alpha_N (\bar{\mathbf{x}}'' - \mathbf{z}_c^{old}) \quad (2.31)$$

Since the positive class indicates the user's preferred images, the presentation of  $\bar{\mathbf{x}}'$  for the new RBF center will give a reasonable representation of the desired images. In particular, the mean value,  $\bar{x}' = \frac{1}{N_p} \times \sum_{i=1}^{N_p} x'_i$ , is a statistical measure providing a good representation of the  $i$ -th feature component since this is the value which minimizes the average distance  $\frac{1}{N_p} \times \sum_{i=1}^{N_p} (x'_i - \bar{x}')$ .

### 2.3.2 Width Selection

The RBFs are adjusted in accordance with different user preferences and different types of images. Through the proximity evaluation, differential biases are assigned to each feature, while features with higher relevance degrees are emphasized, and those with lower degrees are de-emphasized. To estimate the relevance of individual features, the training vectors associated with the set of positive images are used to form an  $N_p \times P$  feature matrix  $\mathbf{R}$ ,

$$\mathbf{R} = [\bar{\mathbf{x}}'_1 \dots \bar{\mathbf{x}}'_m \dots \bar{\mathbf{x}}'_{N_p}]^t \quad (2.32)$$

$$= [x'_{mi}], \quad m = 1, \dots, N_p, \quad i = 1, \dots, P \quad (2.33)$$

where  $x'_{mi}$  is the  $i$ -th component of the  $m$ -th feature vector  $\bar{\mathbf{x}}'_m$ ,  $P$  is the total number of features, and  $N_p$  is the number of positive samples. As the previous discussion, the tuning parameter  $\sigma_i$  should reflect the relevance of individual features. It was demonstrated, in [16, 34], that given a particular numerical value  $z_i$  for a component of the query vector, the length of the interval which complexly encloses  $z_i$  and a pre-determined number  $L$  of the set of values  $x'_{mi}$  in the positive set which falls into its vicinity, is a good indication of the relevancy of the feature. In other words, the relevancy of the  $i$ -th feature is related to the density of  $x'_{mi}$  around  $z_i$ , which is inversely proportional to the length of the interval. A large density usually indicates high relevancy for a particular feature, while a low density implies that the corresponding feature is not critical to the similarity characterization. Setting  $L = N_p$ , the set of turning parameters is thus estimated as follows:

*RBF width model 1:*

$$\sigma = [\sigma_1, \dots, \sigma_i, \dots, \sigma_P]^t \quad (2.34)$$

$$\sigma_i = \eta \cdot \max_m (|x'_{mi} - z_i|) \quad (2.35)$$

The factor  $\eta$  guarantees a reasonably large output  $G(x_i, z_i)$  for the RBF unit, which indicates the degree of similarity, e.g.,  $\eta = 3$ .

*RBF width model 2:* The feature relevancy is also related to sample variance in the positive set  $\{x'_{mi}\}_{m=1}^{N_p}$ , and thus, the RBF width can also be obtained by

$$\sigma_i = \exp((\beta) Std_i) \quad (2.36)$$

$$Std_i = \left( \frac{1}{N_p - 1} \sum_{m=1}^{N_p} (x'_{mi} - \bar{x}')^2 \right)^{\frac{1}{2}} \quad (2.37)$$

where  $Std_i$  is the standard deviation of the members in set  $\{x'_{mi}\}_{m=1}^{N_p}$  which is inversely proportional to their density (Gaussian distribution), and  $\beta$  is a positive constant. The parameter  $\beta$  can be chosen to maximize or minimize the influence of  $Std_i$  on the RBF width. For example, when  $\beta$  is large, a change in  $Std_i$  will be exponentially reflected in the RBF width  $\sigma_i$ .

Both models provide a small value of  $\sigma_i$  if the  $i$ -th feature is highly relevant. This allows higher sensitivity to any change in the distance  $d_i = |x_i - z_i|$ . In contrast, a high value of  $\sigma_i$  is assigned to the non-relevant feature, so that the corresponding vector component can be disregarded when determining the similarity. Table 2.3 summarizes the RBF-based relevance feedback algorithm using RBF center model 1 and RBF width model 1.

### 2.3.3 Experimental Result

This section reports the experimental results [35, 329] of the nonlinear RBF approach in comparison with linear-based adaptive retrieval methods. Table 2.4 describes the database and feature extraction methods used in the experiment. The Laplacian mixture model (LMM) demonstrated in [35] is applied to the texture images for feature characterization. Table 2.5 summarizes the learning procedure of all methods of comparison, which comprise of the RBF method, the query adaption method (QAM), and the metric adaption method (MAM). Table 2.6 summarizes the retrieval results in terms of average precision. The initial precision of 76.7 %, averaged over all queries, was obtained. The precision was significantly improved by updating weighting functions. During relevance feedback, most of the performance enhancement was achieved after the first iterations. A slight improvement was achieved after the second iteration. A significant improvement in the retrieval efficiency was observed by employing a nonlinear RBF method. The final results, after learning, show that RBF-1 gave the best performance with 88.12 % correct retrievals, followed by RBF-2 (87.37 %), and MAM (80.74 %) at a distant third. The QAM is also given for benchmarking purposes.

Figure 2.1 illustrates retrieval examples with and without learning similarity. It shows some of the difficult patterns analyzed, which clearly illustrates the superiority of the RBF method.

**Table 2.3** Summary of the single-RBF based relevance feedback algorithm

<i>Input:</i>	Query vector = $\mathbf{x}_q$ The training samples = $\{\mathbf{x}_i\}_{i=1}^N$
<i>Output:</i>	The final retrieval set, containing $k$ -relevant samples = $S_k(\mathbf{x}_q)$
<i>Initialization:</i>	RBF center $\mathbf{z}_c \equiv \mathbf{x}_q$
<i>Repeat:</i>	<p>User labels training samples, i.e., <math>l_i, i = 1, \dots, N, l_i \in \{0, 1\}</math></p> <p>Calculate RBF center: <math>\mathbf{z}_c^{new} = \mathbf{z}_c^{old} + \alpha_R (\bar{\mathbf{x}}' - \mathbf{z}_c^{old}) - \alpha_N (\bar{\mathbf{x}}'' - \mathbf{z}_c^{old})</math></p> <p>Calculate RBF widths:</p> $\sigma_i = \eta \max_m  x'_{mi} - z_i , i = 1, \dots, P$ <p>Calculate <math>f_q(\mathbf{x}_n) = \sum_{i=1}^P G(x_{ni}, z_i)</math>, for <math>n = 1, 2, \dots, T</math>, and obtain</p> $S_k(\mathbf{x}_q) = \{\mathbf{x}   f_q(\mathbf{x}) \geq f_q(\mathbf{x}_k)\}$ <p>where <math>S_k(\mathbf{x}_q)</math> is the set of nearest neighbors and <math>\mathbf{x}_k</math> is the <math>k</math>-th nearest neighbor of <math>\mathbf{x}_q</math>.</p> $\{\mathbf{x}_i\}_{i=1}^N \leftarrow S_k(\mathbf{x}_q)$
<i>Until:</i>	User is satisfied with the retrieval result.

**Table 2.4** Database and feature extraction methods

Item	Description
Brodatz texture database	The database contains 1,856 texture images divided into 116 classes. Every class has 16 images
Laplacian mixture model (LMM) [35]	The images are decomposed to three levels using Daubechies wavelet filters (db4). The wavelet coefficients in each of high-frequency subbands are modeled as a mixture of two Laplacians. The parameters of the model are used as the features. The feature set composes of (1) the mean and standard deviation of the wavelet coefficients in the approximation subbands and (2) the variances of the two Laplacians in each of the nine high-frequency subbands. This results in 20-dimensional feature vector

In the second experiment, the adaptive retrieval methods are applied in photograph collection. Table 2.7 gives details of the database and the multiple types of visual descriptors, including color, texture, and shape. Table 2.8 gives details of the methods being compared. The average precision rates and CPU times required are summarized in Table 2.9. Evidently, the nonlinear RBF method exhibits significant retrieval effectiveness, while offering more flexibility than MAM and OPT-RF. With the large, heterogeneous image collection, an initial result obtained by the non-adaptive method had less than 50 % precision. With the application of the RBF learning method, the performance could be improved to greater than 90 % precision. Due to limitations in the degree of adaptability, MAM provides the

**Table 2.5** Comparison of adaptive retrieval methods

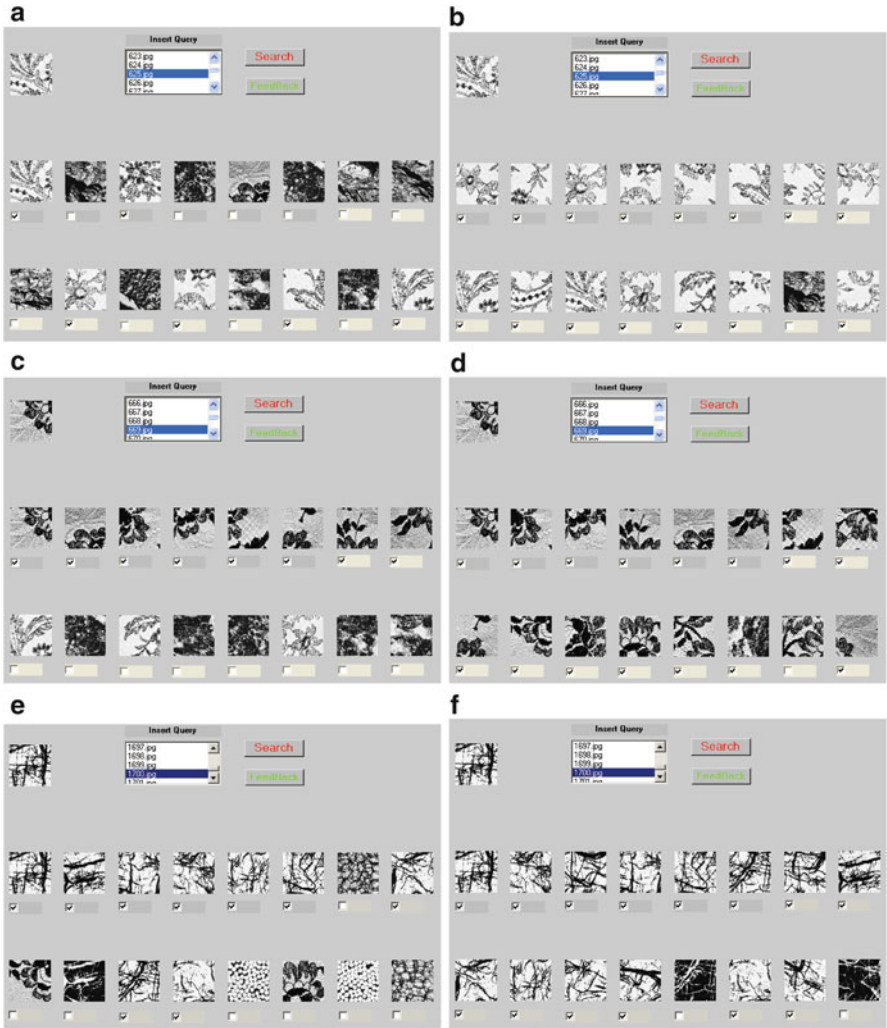
Method	Learning algorithm
RBF-1	RBF center model 1 ( $\alpha_R = 1.2, \alpha_N = 0.08$ )
	RBF width model 1 ( $\eta = 12$ )
RBF-2	RBF center model 2 ( $\alpha_R = 1.2$ )
	BRF width model 1 ( $\eta = 12$ )
QAM [12, 14, 18, 36, 51, 103]	Query modification in Eq. (2.8), Cosine similarity metric in Eq. (2.6) ( $\alpha = 1, \beta = 4, \gamma = 0.8$ )
MAM [17, 20, 21]	City-block distance is used for similarity metric. The feature weighting is obtained by the standard deviation criterion

**Table 2.6** Average precision (%)

Method	Iter. 0	Iter. 1	Iter. 2	Iter. 3
MAM	76.70	80.43	80.71	80.74
RBF-1	76.70	85.02	86.90	88.12
RBF-2	76.70	85.32	86.80	87.37
QAM	67.10	75.12	76.42	76.95

lowest performance gains and converges at about 62 % precision. It is observed that the learning capability of RBF is more robust than that of OPT-RF, not only in retrieval capability, but also learning speed. As presented in Table 2.9, results after one round of the RBF method are similar to results after three rounds of the OPT-RF method. This quick learning is highly desirable, since the user workload can be minimized. This robustness follows from imposing nonlinear discriminant capability in combination with positive and negative learning strategies.

Typical retrieval sessions are shown in Fig. 2.2, for the Yacht query. Figure 2.2a show the 16 best-matches images before applying any feedback, with query image display in the top-left corner. It was observed that some retrieved images were similar to the query image in terms of color composition. In this set, three retrieved images were marked as relevant subjects to the ground truth classes. Figure 2.2b shows the improvement in retrieval after three rounds of using the RBF learning method. This is superior to the results obtained by MAM (cf. Fig. 2.2c) and OPT-RF (cf. Fig. 2.2d). This query may be regarded as a “hard” query, which requires a high degree of nonlinear discrimination analysis. There are some queries that are relatively easier to retrieve, which are shown in Fig.2.3. Those queries have prominent features, such as a shape in the Rose query, and a combination of texture and color in the Polo query. In each case, it is observed that the MAM and OPT-RF methods show better performance than in the previous results. In these cases, however, the retrieval results obtained by RBF approached 100 % precision.



**Fig. 2.1** Top 16 retrievals obtained by retrieving textures D625, D669, and D1700 from the Brodatz database, using RBF-1. Images on the *left*, (a), (c), and (e) show results before learning, and images on the *right*, (b), (d), and (f), show results after learning

## 2.4 Multi-Class Radial Basis Function Method

In image retrieval, particularly in general image collections, the relevancy of images to a specific query is most appropriately characterized by a multi-class modeling approach. For example, when a user has a query for a plane, she or he may wish to have any image containing planes. The semantics of a plane is usually described by a variety of models, which are correlated, but each of which has its

**Table 2.7** Database and feature extraction methods

Item	Description
Corel database [73]	The database contains 40,000 real-life images divided into 400 classes. Every class has 100 images
Color histogram [38] and color moments	The first descriptor is a 48-bin color histogram in HSV color space. The second descriptor is a nine-dimensional vector, conducted from the mean, standard deviation, and skew of the three RGB-color channels
Gabor wavelet descriptor [91]	The 48-dimensional descriptor contains the mean and standard deviations of the Gabor wavelet coefficients from the filtering in four scales and six orientations
Fourier descriptor [37]	The nine-dimensional descriptor contains the Fast Fourier transform coefficients (at low frequency) of the edge information of an input image

**Table 2.8** Comparison of adaptive retrieval methods

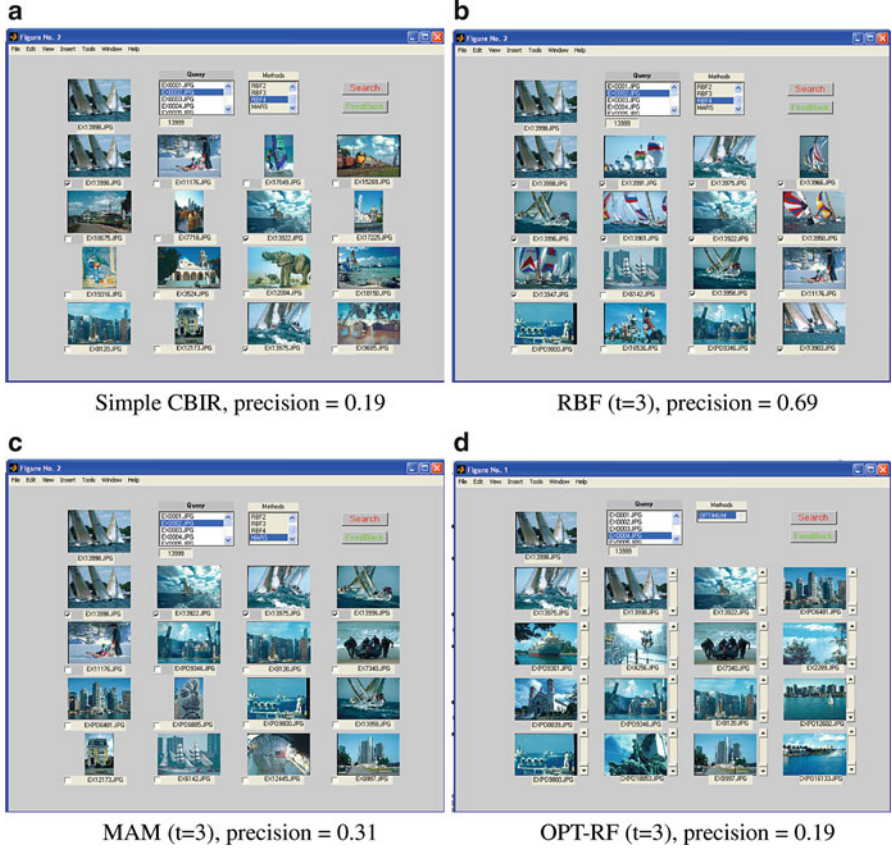
Method	Learning algorithm
RBF	RBF center model 2, RBF width model 2
OPT-RF [26]	Optimum query adaptation model Eq. (2.14), optimum weighting metric Eq. (2.15), Mahalanobis distance Eq. (2.11) as similarity function
MAM [17,20,21]	Mahalanobis distance Eq. (2.11) as similarity function, weight parameters Eq. (2.10) are obtained by the standard deviation criterion

**Table 2.9** Average precision rate (%) obtained by retrieving 35 queries selected from different categories, using the Corel database (columns 2–5)

Method	Iter. 0	Iter. 1	Iter. 2	Iter. 3	CPU time (Sec./Iter.)
RBF	44.82	79.82	88.75	91.76	2.34
MAM	44.82	60.18	61.61	61.96	1.26
OPT-RF	44.82	72.14	79.64	80.84	1.27
Non-adaptive method	44.82	—	—	—	0.90

Average CPU time obtained by retrieving a single query, not including the time to display the retrieved images, measured from a 1.8 GHz Pentium IV processor and a MATLAB implementation

own local characteristics. The difficulty in characterizing image relevancy, then, is identifying the local context associated with each of the sub-classes within the class plane. Human beings utilize multiple types of modeling information to acquire and develop their understanding about image similarity. To obtain more accurate, robust, and natural characterizations, a computer must generate a fuller definition of what humans regard as significant features. Through user feedback, computers do acquire knowledge of novel features which are significant but have not been explicitly specified in the training data. This implicit information constitutes subclasses within the query, permitting better generalization. In this case, a mixture of Gaussian models is used, via the RBF network, to represent multiple types of model information for the recognition and presentation of images by machines.

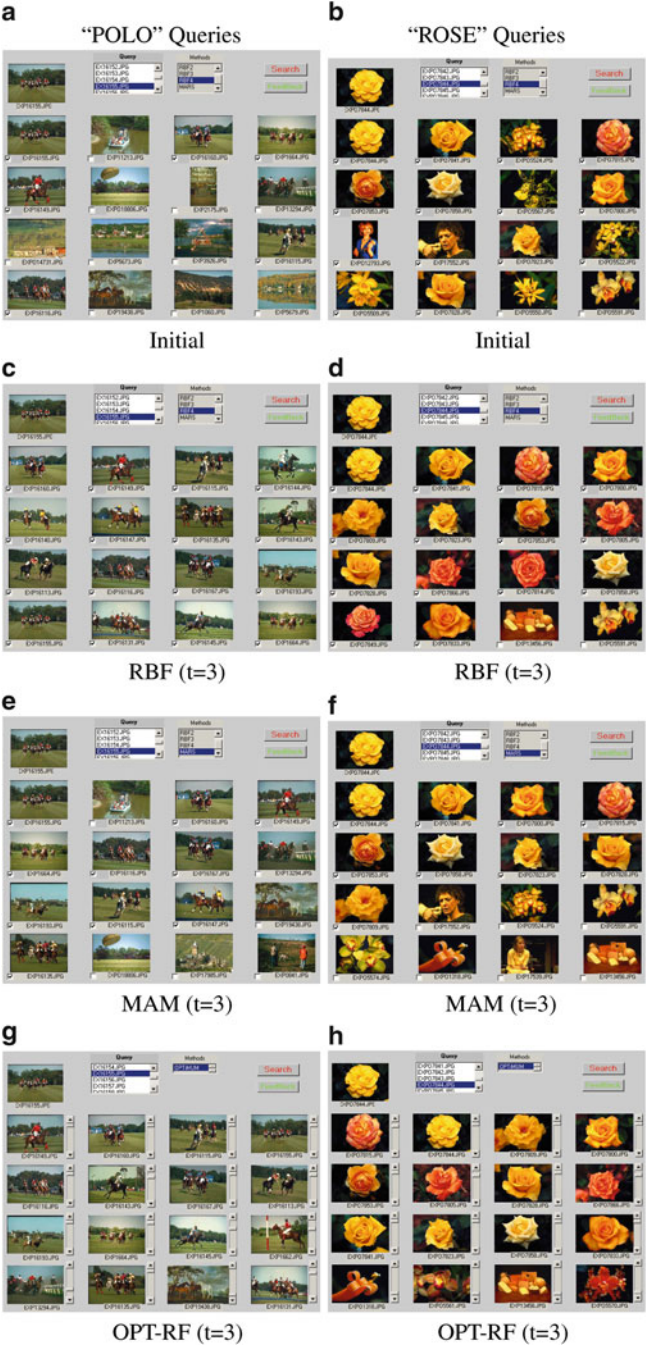


**Fig. 2.2** Top 16 retrieved images obtained by the Yacht query, using the Corel Database, (a) before RF learning, (b) after RF learning with the RBF method, (c) MAM, and (d) OPT-RF

Previously, Sect. 2.3 introduced a nonlinear input–output mapping function based on a single-RBF model. As discussed by most other works [15–17], this has been concerned with *global* modeling, in which a query image is described by one model, which is then associated with only a particular location in the input space. Furthermore, the similarity function is based on a single metric. This combination gives rise to a single model function  $f(\mathbf{x})$ , which cannot fully exploit the local data information. This section introduces a mixture of Gaussian models for adaptive retrieval that enables the learning system to take advantage of the information from multiple sub-classes. The learning system utilizes a highly local characterization of image relevancy in the form of a superposition of different local models, as  $\sum_i f_i(\mathbf{x})$ , to obtain the input–output mapping function.

The learning methods for constructing the RBF network include the adaptive RBF method [61], gradient-descent method [40], and fuzzy RBF method [39].





**Fig. 2.3** Retrieval results of POLO and ROSE queries, obtained by (a, b) non-adaptive retrieval method, (c, d) RBF, (e, f) MAM, and (g, h) OPT-RF

### 2.4.1 Local Model Network

The basic assumption underlying the use of learning systems is that the behavior of the system can be described in terms of the training set  $\{\mathbf{x}_i, y_i\}_{i=1}^N$ . It is therefore assumed that the system to be described by a model whose observable output  $y_i$ , at the time of step  $i$ , in response to an input vector  $\mathbf{x}_i$ , is defined by:

$$y_i = f(\mathbf{x}_i) + \varepsilon_i, \quad i = 1, 2, \dots, N \quad (2.38)$$

where  $\varepsilon_i$  is a sample drawn from a white noise process of zero mean and variance  $\sigma^2$ . The modeling problem is to estimate the underlying function of the model,  $f(\mathbf{x}_i)$ , from observation data, having already used the existing *a priori* information to structure and parameterize the model. Let  $\hat{f}(\mathbf{x}, \mathbf{z})$  be the estimate of  $f(\mathbf{x})$  for some values of the  $P$ -dimensional parameter vector  $\mathbf{z}$ . The model  $\hat{f}(\mathbf{x}, \mathbf{z})$  can be estimated in a number of ways. A Local Model Network (LMN), is adopted to achieve this purpose [41]. Figure 2.4 shows the network architecture. This type of network approximates the model function  $\hat{f}(\mathbf{x}, \mathbf{z})$  according to:

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{N_m} \lambda_i \hat{f}_i(\mathbf{x}, \mathbf{z}_i) \quad (2.39)$$

$$= \sum_{i=1}^{N_m} \lambda_i K_i(\mathbf{x}, \mathbf{z}_i) = \sum_{i=1}^{N_m} \lambda_i \exp\left(-\frac{\|\mathbf{x} - \mathbf{z}_i\|^2}{2\sigma_i^2}\right) \quad (2.40)$$

where  $\mathbf{x} = [x_1, \dots, x_p]^t$  and  $\mathbf{z} = [z_1, \dots, z_p]^t$  are the input vector and the RBF center, respectively. In addition,  $\lambda_i, i = 1, \dots, N_m$  are the weight, and  $K(\mathbf{x}, \mathbf{z})$  is a nonlinearity of hidden nodes.

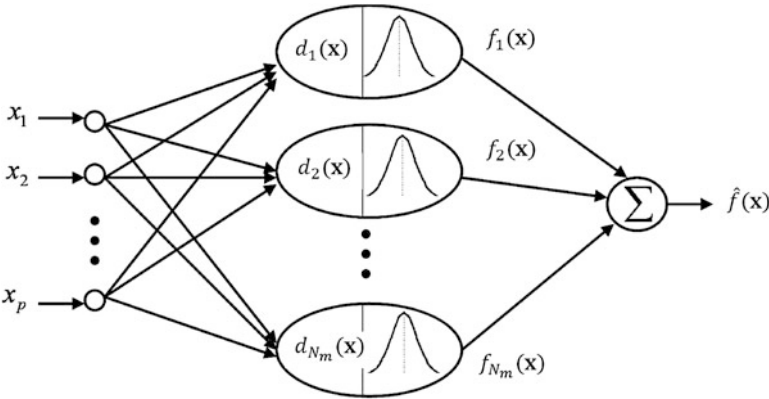


Fig. 2.4 RBF network architecture

The advantage of this network's use in the current application is that it finds the input-to-output map using local approximators; consequently, the underlying basis function responds only to a small region of the input space where the function is centered, e.g., a Gaussian response,  $K_i = \exp\left(-\frac{d^2}{2}\right)$ , where:

$$d(\mathbf{x}, \mathbf{z}_i, \sigma_i) = \sqrt{(\mathbf{x} - \mathbf{z})^t \sigma_i^{-2} (\mathbf{x} - \mathbf{z}_i)} \quad (2.41)$$

This allows local evaluation for image similarity matching.

The parameters to learn for the LMN are the set of linear weight  $\lambda_i$ , the center  $\mathbf{z}_i$ , and the width  $\sigma_i$  for each local approximator  $K_i, i = 1, \dots, N_m$ . The linear weights are usually estimated by the least-squared (LS) method [43]. When using the Gaussian function as the nonlinearity of hidden nodes, it has been observed that the same width of  $\sigma_i$  is sufficient for the RBF network to obtain universal approximation [42]. However, more recent theoretical investigations and practical results indicate that the choice of center  $\mathbf{z}_i$  is most significant in the performance of the RBF network [44]. As we shall see, this suggestion plays a central role in overcoming the variation in the performance of the network in the adaptive retrieval application.

### 2.4.2 Learning Methods for the RBF Network

Various learning strategies have been proposed to structure and parameterize the RBF network [41, 43–45]. This section will consider two of these beside the new learning strategy for adaptive image retrieval. For a given training set  $\{\mathbf{x}_i, y_i\}_{i=1}^N$ , the initial approaches [41], constructed the RBF network by associating all available training samples to the hidden units, using one-to-one correspondence. A radial-basis function centered at  $\mathbf{z}_i$  is defined as:

$$K(\mathbf{x}, \mathbf{z}_i) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{z}_i\|^2}{2\sigma_i^2}\right), \quad i = 1, \dots, N_m \quad (2.42)$$

where

$$\{\mathbf{z}_i\}_{i=1}^{N_m} = \{\mathbf{x}_i\}_{i=1}^N, \quad N_m = N \quad (2.43)$$

This solution may be expensive, in terms of computational complexity, when  $N$  is large. Thus, we may arbitrarily choose some data points as centers [43]. This gives an approximation to the original RBF network, while providing a more suitable basis for practical applications. In this case, the approximated solution is expanded on a finite basis:

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{N_m} \lambda_i K(\mathbf{x}, \mathbf{z}_i) \quad (2.44)$$

where

$$\{\mathbf{z}_i\}_{i=1}^{N_m} \subset \{\mathbf{x}_i\}_{i=1}^N, \quad N_m < N \quad (2.45)$$

The linear weights  $\lambda_i$ ,  $i = 1, \dots, N_m$  are determined by minimizing the following cost function,  $\xi(f)$ :

$$\xi(\hat{f}) = \sum_{i=1}^N \left( y_i - \sum_{j=1}^{N_m} \lambda_j K(\mathbf{x}_i, \mathbf{z}_j) \right)^2 + \gamma \|\mathbf{D}\hat{f}\|^2 \quad (2.46)$$

where  $\gamma$  is the regularization parameter, and  $\mathbf{D}$  is a differential operator. Based on the *pseudoinverse* method [43], the minimization of Eq. (2.46) with respect to the weight vector  $\lambda = [\lambda_1, \dots, \lambda_{N_m}]^T$ , yields:

$$\lambda = \mathbf{G}^+ \mathbf{y} \quad (2.47)$$

$$= (\mathbf{G}^T \mathbf{G})^{-1} \mathbf{y} \quad (2.48)$$

where

$$\mathbf{y} = [y_1, y_2, \dots, y_N]^T \quad (2.49)$$

The matrix  $\mathbf{G} \in \mathcal{M}_{N \times N_m}$  is defined as:

$$\mathbf{G} = \{K_{ij}\} \quad (2.50)$$

$$K_{ij} = \exp \left( -\frac{\|\mathbf{x}_i - \mathbf{z}_j\|^2}{2\sigma_j^2} \right), \quad i = 1, \dots, N; \quad j = 1, \dots, N_m \quad (2.51)$$

where  $\mathbf{x}_i$  is the  $i$ -th training sample.

Table 2.10 summarizes the RBF network learning with randomly selected centers, applied to image retrieval. The main problem with this method is that it cannot guarantee desired performance, because it may not satisfy the requirement that the centers should suitably sample the input domain. To overcome this problem, the orthogonal least squares (OLS) learning algorithm [44] is designed to select a suitable set of centers so that adequate RBF networks can be obtained. The OLS algorithm chooses centers one by one from the training data; that is, at each iteration the vector that results in the largest reduction in network errors is used to create the center. When the sum-squared error of the network computed is higher than a specified level, the next center is added to the network. The iteration process stops when the error falls beneath an error goal, or when the maximum number of centers is reached. This provides a simple and efficient means for fitting RBF networks.

**Table 2.10** Summary of the RBF network learning with randomly selected centers, applied to image retrieval

<i>Input:</i>	The training samples = $\{\mathbf{x}_i\}_{i=1}^N$ for a given query $\mathbf{x}_q$
<i>Output:</i>	The final retrieval set, containing $k$ -relevant samples = $S_k$
<i>Initialization:</i>	Number of RBF centers = $N_m$ Setting RBF width to a positive constant, $\sigma_i \leq 1, i = 1, \dots, N_m$
<i>Repeat:</i>	<p>User provides class label <math>l_i, i = 1, \dots, N, l_i \in \{0, 1\}</math></p> <p>Select RBF center <math>\{\mathbf{z}_i\}_{i=1}^{N_m} \subset \{\mathbf{x}_i\}_{i=1}^N</math></p> <p>Calculate weights <math>\lambda = [\lambda_1, \dots, \lambda_{N_m}]^t</math>:</p> $\lambda = (\mathbf{G}^t \mathbf{G})^{-1} \mathbf{y}$ <p>where</p> $\mathbf{G} = \{K_{ij}\}$ $K_{ij} = \exp\left(-\frac{\ \mathbf{x}_i - \mathbf{z}_j\ ^2}{2\sigma_j^2}\right), \quad i = 1, \dots, N; \quad j = 1, \dots, N_m$ $\mathbf{y} = [y_1, \dots, y_i, \dots, y_N]^t, \quad y_i \leftarrow l_i$ <p>For <math>n = 1, 2, \dots, T</math>, calculate <math>\hat{f}(\mathbf{x}_n) = \sum_{i=1}^{N_m} \lambda_i K(\mathbf{x}_n, \mathbf{z}_i)</math></p> <p>Obtain <math>k</math>-nearest neighbor:</p> $S_k(\mathbf{x}_q) = \{\mathbf{x}   \hat{f}(\mathbf{x}) \geq \hat{f}(\mathbf{x}_k)\}$ <p>where <math>S_k</math> is the set of top <math>k</math> ranked samples.</p> $\{\mathbf{x}_i\}_{i=1}^N \leftarrow S_k(\mathbf{x}_q)$
<i>Until:</i>	User is satisfied with the retrieval result.

### 2.4.3 Adaptive Radial-Basis Function Network

Problems in adaptive image retrieval are considered as a special case for function approximation. The characteristics of learning are quite different. First, the training data size for image retrieval is very small compared to the general approximation strategy. Second, the training samples available for image retrieval are highly correlated, i.e., each sample is selected from a specific area of the input space and is near to the next, in the Euclidean sense. When the training samples are highly correlated, the choice of centers is the most important factor. The RBF network will be ill-conditioned, owing to the near-linear dependency caused by some centers being too close together [44].

In order to circumvent the environmental restrictions in image retrieval, an adaptive learning strategy for the RBF network is introduced and referred to as adaptive RBF network (ARBFN). This is a special network for learning in image

retrieval where there is a small set of samples with a high level of correlation between the samples. This new strategy is based on the following points:

- The learning method formulates and solves the local approximator  $K(\mathbf{x}, \mathbf{z})$  from available positive samples.
- In order to obtain a dynamic weighting scheme, the Euclidean norm in  $\|\mathbf{x} - \mathbf{z}\|$  is replaced with the weighted Euclidean,  $\|\mathbf{x} - \mathbf{z}\|_M$ .
- In order to take advantage of negative samples to improve the decision boundary, a method of shifting centers is obtained, instead of employing linear weights.

The learning strategy for the ARBFN consists of two parts. First, the local approximators  $K(\mathbf{x}, \mathbf{z})$  are constructed using *positive* samples. Second, in order to improve the decision boundary, *negative* samples are used for shifting the centers, based on anti-reinforced learning [331].

#### 2.4.3.1 Construction of Local Approximators

Given the set of positive samples,  $\mathcal{X}^+ = \{\mathbf{x}'_i\}_{i=1}^{N_p}$ , each positive sample is assigned to the local approximator  $K(\mathbf{x}, \mathbf{z}_i)$ , so that the shape of each relevant cluster can be described by:

$$K(\mathbf{x}, \mathbf{z}_i) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{z}_i\|^2}{2\sigma_i^2}\right), \quad (2.52)$$

$$\mathbf{z}_i = \mathbf{x}'_i, \quad \forall i \in \{1, \dots, N_m\}, \quad N_m = N_p \quad (2.53)$$

$$\sigma_i = \delta \cdot \min(\|\mathbf{z}_i - \mathbf{z}_j\|), \quad \forall j \in \{1, 2, \dots, N_p\}, \quad i \neq j \quad (2.54)$$

where  $\delta = 0.5$  is an overlapping factor.

Here, only the positive samples are assigned as the centers of the RBF functions. Hence, the estimated model function  $f(\mathbf{x})$  is given by:

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{N_m} \lambda_i K(\mathbf{x}, \mathbf{z}_i) \quad (2.55)$$

$$\lambda_i = 1, \quad \forall i \in \{1, \dots, N_m\} \quad (2.56)$$

The linear weights are set to constant, indicating that all the centers (or the positive samples) are taken into consideration. However, the degree of importance of  $K(\mathbf{x}, \mathbf{z}_i)$  is indicated by the natural responses of the Gaussian-shaped RBF functions and their superposition. For instance, if centers  $\mathbf{z}_a$  and  $\mathbf{z}_b$  are highly correlated (i.e.,  $\mathbf{z}_a \approx \mathbf{z}_b$ ), the magnitude of  $\hat{f}(\mathbf{x})$  will be biased for any input vector  $\mathbf{x}$  located near  $\mathbf{z}_a$  or  $\mathbf{z}_b$ , i.e.,  $\hat{f}(\mathbf{x}) \approx 2K(\mathbf{x}, \mathbf{z}_a) \approx 2K(\mathbf{x}, \mathbf{z}_b)$ .

### 2.4.3.2 Integrating Elliptic Basis Function

The basic RBF version of the ARBFN discussed in Eq.(2.52) is based on the assumption that the feature space is uniformly weighted in all directions. However, image feature variables tend to exhibit different degrees of importance which heavily depend on the nature of the query and the relevant images defined [16]. This leads to the adoption of an elliptic basis function (EBF):

$$\|\mathbf{x} - \mathbf{z}_i\|_{\mathbf{M}} = (\mathbf{x} - \mathbf{z}_i)^t \mathbf{M} (\mathbf{x} - \mathbf{z}_i) \quad (2.57)$$

where

$$\mathbf{M} = \text{Diag}[\alpha_1, \dots, \alpha_j, \dots, \alpha_P] \quad (2.58)$$

So, the parameter  $\alpha_j, j = 1, \dots, P$  represents the relevance weights which are derived from the variance of the positive samples in  $\{\mathbf{x}'_i\}_{i=1}^{N_p}, \mathbf{x}'_i \in \mathbb{R}^P$  as follows:

$$\alpha_j = \begin{cases} 1, & \zeta_j = 0 \\ \frac{1}{\zeta_j}, & \text{Otherwise} \end{cases} \quad (2.59)$$

where

$$\zeta_j = \left( \frac{1}{N_p - 1} \sum_{i=1}^{N_p} (x'_{ij} - \bar{x}'_j)^2 \right)^{\frac{1}{2}} \quad (2.60)$$

$$\bar{x}'_j = \frac{1}{N_p} \sum_{i=1}^{N_p} x'_{ij} \quad (2.61)$$

The matrix  $\mathbf{M}$  is a symmetrical  $\mathbf{M}_{P \times P}$ , whose diagonal elements  $\alpha_j$  assign a specific weight to each input coordinate, determining the degree of the relevance of the features. The weight  $\alpha_j$  is inversely proportional to  $\zeta_j$ , the standard deviation of the  $j$ -th feature component of the positive samples,  $\{x'_{ij}\}_{i=1}^{N_p}$ . If a particular feature is relevant, then all positive samples should have a very similar value to this feature, i.e., the sample variance in the positive set is small [17].

### 2.4.3.3 Shifting RBF Centers

The possibility of moving the expansion centers is useful for improving the representativeness of the centers. Recall that, in a given training set, both positive and negative samples are presented, which are ranked results from the previous search operation. For all negative samples in this set, the similarity scores from the previous search indicate that their clusters are close to the positive samples retrieved.

Here, the use of negative samples becomes essential, as the RBF centers should be moved slightly away from these clusters. Shifting the centers reduces the similarity scores for those negative samples, and thus more favorable similarity scores can be obtained for any positive samples that are in the same neighborhood area, in the next round of retrieval.

Recall that the set of negative samples is denoted by  $\mathcal{X}^- = \{\mathbf{x}_i''\}_{i=1}^{N_n}$ , and  $N_n$  is the number of these samples. At the  $n$ -th iteration, let the input vector  $\mathbf{x}''$  (randomly selected from the negative set) be the closest point to  $\mathbf{z}_{i^*}$ , such that:

$$i^* = \arg \min_i (\|\mathbf{x}'' - \mathbf{z}_i\|_{\mathbf{M}}), \quad i \in \{1, \dots, N_m\} \quad (2.62)$$

Then, the center  $\mathbf{z}_{i^*}$  is modified by the anti-reinforce learning rule:

$$\mathbf{z}_{i^*}(n+1) = \mathbf{z}_{i^*}(n) - \eta(n)[\mathbf{x}'' - \mathbf{z}_{i^*}(n)] \quad (2.63)$$

where  $\eta$  is a learning constant which decreases monotonically with the number of iteration, and  $0 < \eta < 1$ . The algorithm is repeated by selecting a new sample from the set of input samples,  $\{\mathbf{x}_i''\}_{i=1}^{N_n}$ , and stops after a maximum number of iterations is reached.

Table 2.11 summarizes the learning procedure of the ABRF network for image retrieval. This includes learning steps explained in Sects. 2.4.3.1–2.4.3.3.

## 2.4.4 Gradient-Descent Procedure

Apart from the ARBFN model, the procedural parameters for RBF can be obtained by a gradient-descent procedure [39,40]. This procedure is employed to optimize all three parameters,  $\mathbf{z}_i$ ,  $\sigma_i$ , and  $\lambda_i$  for each RBF unit. Here, all training samples (both positive and negative) are assigned to the RBF centers, and the linear weights are used to control the output of each RBF unit. Thus, the mapping function becomes:

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^N \lambda_i K(\mathbf{x}, \mathbf{z}_i) = \sum_{i=1}^N \lambda_i \exp\left(-\frac{\|\mathbf{x} - \mathbf{z}_i\|_{\mathbf{M}}^2}{2\sigma_i^2}\right) \quad (2.64)$$

where  $\{\mathbf{z}_i\}_{i=1}^N = \mathcal{X}^+ \cup \mathcal{X}^-$ . During relevance feedback learning, the network attempts to minimize the following error function:

$$\xi(\hat{f}) = \frac{1}{2} \sum_{j=1}^N e_j^2 = \frac{1}{2} \sum_{j=1}^N \left( y_j - \sum_{i=1}^N \lambda_i K(\mathbf{x}_j, \mathbf{z}_i) \right)^2 \quad (2.65)$$

where  $e_j$  is the error signal for the training sample  $\mathbf{x}_j$ , and  $y_j$  represents the desired output of the  $j$ -th training sample. The network parameters can be obtained by the



**Table 2.11** Summary of the learning algorithm of the ARBF network for adaptive retrieval

<i>Input:</i>	The training samples = $\{\mathbf{x}_i\}_{i=1}^N$ for a given query $\mathbf{x}_q$
<i>Output:</i>	The final retrieval set, containing $k$ -relevant samples = $S_k(\mathbf{x}_q)$
<i>Initialization:</i>	Setting smoothing parameter $\delta = 0.5$ Maximum number of iterations = $N_{max}$ Setting anti-reinforce learning parameter $\eta$
<i>Repeat:</i>	User provides labels for training vectors, $l_i, i = 1, \dots, N, l_i \in \{0, 1\}$ Construct $\mathcal{X}^+$ and $\mathcal{X}^-$ Assigning RBF center $\mathbf{z}_i \leftarrow \mathbf{x}'_i, \forall i \in \{1, \dots, N_p\}$ Obtain weight matrix $\mathbf{M}$ For $n = 1 : N_{max}$ , adjust RBF centers <ol style="list-style-type: none"> <li>1. Randomly select the input vector <math>\mathbf{x}''</math> from <math>\mathcal{X}^-</math></li> <li>2. Select winning node <math>\mathbf{z}_{i^*}</math>, such that:               <math display="block">i^* = \arg \min_i (\ \mathbf{x}'' - \mathbf{z}_i\ _{\mathbf{M}}), \quad i \in \{1, \dots, N_m\}</math> </li> <li>3. Update               <math display="block">\mathbf{z}_{i^*}(n+1) \leftarrow \mathbf{z}_{i^*}(n) - \eta(n)[\mathbf{x}'' - \mathbf{z}_{i^*}(n)]</math> </li> </ol> End for-loop For $i = 1, 2, \dots, N_p$ , calculate RBF width $\sigma_i = \delta \min \ \mathbf{z}_i - \mathbf{z}_j\ , \quad \forall j \in \{1, 2, \dots, N_p\}, \quad i \neq j$ For $j = 1, 2, \dots, T$ , calculate $\hat{f}(\mathbf{x}_j) = \sum_{i=1}^{N_m} \exp\left(-\frac{\ \mathbf{x}_j - \mathbf{z}_i\ _{\mathbf{M}}^2}{2\sigma_i^2}\right)$ Obtain $k$ -nearest neighbor: $S_k(\mathbf{x}_q) = \{\mathbf{x}   \hat{f}(\mathbf{x}) \geq \hat{f}(\mathbf{x}_k)\}$ where $S_k(\mathbf{x}_q)$ is the set of top $k$ ranked samples. $\{\mathbf{x}_i\}_{i=1}^N \leftarrow S_k(\mathbf{x}_q)$
<i>Until:</i>	User is satisfied with the retrieval result.

gradient-descent method to minimize the cost function, i.e.,

$$\{\mathbf{z}_i, \boldsymbol{\sigma}_i, \lambda_i\}_{i=1}^N = \arg \min (\xi) \quad (2.66)$$

The learning procedure starts with the initialization of the linear weights:

$$\lambda_i = \begin{cases} 1, & \text{if } \mathbf{z}_i \text{ is conducted from positive sample} \\ -0.5, & \text{if } \mathbf{z}_i \text{ is conducted from negative sample} \end{cases} \quad (2.67)$$

and the RBF widths:

$$\sigma_i = \delta \min (\|\mathbf{z}_i - \mathbf{z}_j\|_{\mathbf{M}}), j \in \{1, 2, \dots, N\} \quad i \neq j \quad (2.68)$$

Based on the gradient-descent method, the parameters for the  $i$ -th RBF unit are updated in the iterative process, as follows:

1. For  $t = 1, 2, \dots, N_{max}$ :
2. Update

$$\lambda_i(t+1) \leftarrow \lambda_i(t) - \eta_1 \frac{\partial \xi(t)}{\partial \lambda_i(t)} \quad (2.69)$$

where  $\frac{\partial \xi(t)}{\partial \lambda_i(t)} = -\sum_{j=1}^N e_j(t) K(\mathbf{x}_j, \mathbf{z}_i)$

3. Update

$$\mathbf{z}_i(t+1) \leftarrow \mathbf{z}_i(t) - \eta_2 \frac{\partial \xi(t)}{\partial \mathbf{z}_i(t)} \quad (2.70)$$

where  $\frac{\partial \xi(t)}{\partial \mathbf{z}_i(t)} = -\lambda_i(t) \sum_{j=1}^N e_j(t) K(\mathbf{x}_j, \mathbf{z}_i) \frac{\mathbf{M}(\mathbf{x}_j - \mathbf{z}_i(t))}{\sigma_i^2(t)}$

4. Update

$$\sigma_i^2(t+1) \leftarrow \sigma_i^2(t) - \eta_3 \frac{\partial \xi(t)}{\partial \sigma_i(t)} \quad (2.71)$$

where  $\frac{\partial \xi(t)}{\partial \sigma_i(t)} = -\lambda_i(t) \sum_{j=1}^N e_j(t) K(\mathbf{x}_j, \mathbf{z}_i) \frac{(\mathbf{x}_j - \mathbf{z}_i(t))^t M(\mathbf{x}_j - \mathbf{z}_i(t))}{\sigma_i^2(t)}$

5. Return

where  $N_{max}$  is the maximum iteration count, and  $\eta_1$ ,  $\eta_2$ , and  $\eta_3$  are the step sizes.

The adjustment of the RBF models proceeds along many relevance feedback sessions. The training samples are gathered from the first to the last retrieval sessions, and only selective samples are used to retrain the network. In each feedback session, newly retrieved samples which have not been found in the previous retrieval are inserted into the existing RBF network. In the next iteration, the updating procedure is performed on the newly inserted RBF units, thus improving training speed.

### 2.4.5 Fuzzy RBF Network with Soft Constraint

The error function  $\xi(\hat{f})$  defined in Eq.(2.65) is based on the binary labeling or hard-decision. The desired network output  $y_j$  is equal to 1 for positive samples, and zero for negative samples. For a soft-decision, a third option, “fuzzy” is used to characterize a vague description of the retrieved (image) samples [39]. Thus, in a retrieval session, users have three choices for relevance feedback: relevant, irrelevant, and fuzzy. The error function is then calculated by:

$$\xi(\hat{f}) = \frac{1}{2} \sum_{j=1}^N \left( y_j - \sum_{i=1}^N \lambda_i K(\mathbf{x}_j, \mathbf{z}_i) \right)^2 \quad (2.72)$$

$$y_j = \begin{cases} 1, & \mathbf{x}_j \text{ is relevant} \\ 0, & \mathbf{x}_j \text{ is irrelevant} \\ P(\mathcal{X}^+|\mathbf{x}_j), & \mathbf{x}_j \text{ is fuzzy} \end{cases} \quad (2.73)$$

where  $P(\mathcal{X}^+|\mathbf{x}_j)$  is the probability that a fuzzy sample  $\mathbf{x}_j$  belongs to the relevant class  $\mathcal{X}^+$ . This represents the degree of relevancy for the corresponding fuzzy sample. The learning problem is the problem in estimating the desired output  $y_j = P(\mathcal{X}^+|\mathbf{x}_j)$  of the fuzzy sample  $\mathbf{x}_j$  by the *a posteriori* probability estimator. Let  $\mathbf{x}_j$  be defined by feature vector that is concatenated from  $M$  sub-vector, i.e.,  $\mathbf{x}_j \equiv [\mathbf{v}_{j1}, \dots, \mathbf{v}_{ji}, \dots, \mathbf{v}_{jM}]$ , where  $\mathbf{v}_{ji}$  is a  $d_i$ -dimensional feature sub-vector such as a color histogram, a set of wavelet moments or others. To deal with the uncertainty, the probability estimator takes into account the multiple features, by using the following estimation principle:

$$P(\mathcal{X}^+|\mathbf{x}_j) = \frac{1}{M} \sum_{i=1}^M P(\mathcal{X}^+|\mathbf{v}_{ji}) \quad (2.74)$$

where  $P(\mathcal{X}^+|\mathbf{v}_{ji})$  is the *a posteriori* probability for the  $i$ -th feature vector  $\mathbf{v}_{ji}$  of the fuzzy sample  $\mathbf{x}_j$ . The Bayesian theory is applied to  $P(\mathcal{X}^+|\mathbf{v}_{ji})$ ,

$$P(\mathcal{X}^+|\mathbf{v}_{ji}) = \frac{P(\mathbf{v}_{ji}|\mathcal{X}^+)P(\mathcal{X}^+)}{P(\mathbf{v}_{ji}|\mathcal{X}^+)P(\mathcal{X}^+) + P(\mathbf{v}_{ji}|\mathcal{X}^-)P(\mathcal{X}^-)} \quad (2.75)$$

where  $P(\mathcal{X}^+)$  and  $P(\mathcal{X}^-)$  are, respectively, the prior probabilities of the positive and negative classes, which can be estimated from the feedback samples;  $P(\mathbf{v}_{ji}|\mathcal{X}^+)$  and  $P(\mathbf{v}_{ji}|\mathcal{X}^-)$  are the class conditional probability density functions of  $\mathbf{v}_{ji}$  for the positive and negative classes, respectively. Assuming the Gaussian distribution, the probability density function for the positive class is given by:

$$P(\mathbf{v}_{ji}|\mathcal{X}^+) = \frac{1}{(2\pi)^{\frac{d_i}{2}} |\Sigma'_i|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\mathbf{v}_{ji} - \mu'_i)' \Sigma'^{-1}_i (\mathbf{v}_{ji} - \mu'_i)\right] \quad (2.76)$$

where  $\mu'_i$  is the  $d_i$ -component mean vector and  $\Sigma'_i$  is the  $d_i$ -by- $d_i$  covariance matrix for the  $i$ -th feature vector, and  $|\Sigma'_i|$  and  $\Sigma'^{-1}_i$  are its determinant and inverse, respectively. These variables can be estimated using the positive training vectors in the positive class  $\mathcal{X}^+$ ,

$$\mu'_i = \frac{1}{N_p} \sum_{j=1}^{N_p} \mathbf{v}'_{ji} \quad (2.77)$$

$$\Sigma'_i = \frac{1}{(N_p - 1)} \sum_{j=1}^{N_p} (\mathbf{v}'_{ji} - \mu_i) (\mathbf{v}'_{ji} - \mu_i)^t \quad (2.78)$$

where  $\mathbf{v}'_{ji}$  is the  $i$ -th sub-vector of  $j$ -th positive sample, and  $N_p$  is the number of positive samples. For simplicity we abbreviate Eq. (2.76) as  $P(\mathbf{v}_{ji}|\mathcal{X}^+) \sim N(\mu'_i, \Sigma'_i)$ . Similarly, the probability density function for the negative class is given by:  $P(\mathbf{v}_{ji}|\mathcal{X}^-) \sim N(\mu''_i, \Sigma''_i)$  where  $\mu''_i$  is the mean vector and  $\Sigma''_i$  is the covariance matrix for the  $i$ -th feature vector, which can be estimated using the negative training vectors in the negative class  $\mathcal{X}^-$ .

After the desired output  $y_j$  of the fuzzy sample is estimated, the gradient-descent procedure of Eqs. (2.69)–(2.71) are applied to construct the learning parameters of the RBF network.

#### 2.4.6 Experimental Result

The Corel database was used in the experiments reported in Sect. 2.3.3. All 40,000 images in the database were used, each of which was characterized by a multi-feature representation (explained in Table 2.7). This section begins by implementing the RBF network using the ARBFN leaning method and comparing its performance with two other learning strategies. This is followed by examining the ARBFN and the single-class learning methods discussed in Sects. 2.2–2.3.

The first objective is to verify that the ARBFN is able to meet the demands of adaptive retrieval applications; in particular, where there is a small set of training samples with a high level of correlation between the samples. A learning session with this condition may be observed in Fig. 2.3d, where the top sixteen retrieved images are returned to the user who provides relevance feedback. It is seen that at later iterations the learning system can improve the result sets, which means that the more times the interactive retrieval is implemented, the higher the level of correlation retrieved images.

The ARBFN method was compared with two learning strategies that have been successfully used in other situations to construct the RBF network. Table 2.12 summarizes the methods being compared. The first learning method, the orthogonal least square (OLS) learning procedure described in [44], was used to identify a RBF network model. In the second learning method [43], each vector in a

**Table 2.12** Comparison of RBF learning methods

Method	Learning algorithm
ARBFN	Table 2.11, RBF centers: using positive samples in $\mathcal{X}^+ = \{\mathbf{x}'_i\}_{i=1}^{N_p}$ ; RBF width: Eq. (2.54)
EDLS [43]	The weight and bias of the second layers were calculated by the least squares criterion; RBF centers: using all samples in $\{\mathbf{x}'_i\}_{i=1}^{N_p} \cup \{\mathbf{x}''_i\}_{i=1}^{N_n}$ ; RBF width: $\sigma = 0.8$ for all RBF units
OLS [44]	RBF centers: selecting from $\{\mathbf{x}'_i\}_{i=1}^{N_p} \cup \{\mathbf{x}''_i\}_{i=1}^{N_n}$ using the orthogonal least square method. The RBF center selection starts zero centers, and new centers were iteratively picked in the subsequent selection procedure. Each time, the network's mean square error was checked and compared to the pre-defined tolerance set at 0.0001; RBF width: $\sigma = 0.8$ for all RBF units

**Table 2.13** Average precision rate (%) as a function of iteration,  $\bar{Pr}(Iter.)$ , obtained by retrieving 35 queries, using Corel dataset

Method	Iter. 0	Iter. 1	Iter. 2	Iter. 3
ARBFN	44.82	80.72	90.36	92.50
EDLS	44.82	50.18	43.39	43.04
OLS	44.82	66.07	73.21	76.61

retrieved set was associated with the RBF centers [Eq. (2.43)], using a one-to-one correspondence. This is named as EDLS (exact design network using the least squares criterion). For both methods, the final RBF network model can be written as:

$$\hat{f}(\mathbf{x}_j) = \lambda_0 + \sum_{i=1}^{N_m} \lambda_i \exp \left( -\frac{\|\mathbf{x} - \mathbf{z}_i\|^2}{2\sigma_i^2} \right) \quad (2.79)$$

where  $N_m = 16$  for the EDLS method, and  $N_m \leq 16$  for the OLS learning method, since the size of retrieved samples is set to 16 at each feedback iteration.

The query image set used here is identical to the experiments reported in Sect. 2.3.3. Precision ( $Pr$ ) was recorded after each query iteration. Table 2.13 summarizes the average precision results,  $\bar{Pr}(Iter.)$ , as a function of iteration, taken over the 35 test queries. It can be seen from the results that the ARBFN significantly improved the retrieval accuracy (up to 92 % precision). The first iteration showed an improvement of 35.9 %. The ARBFN outperformed the OLS (76.61 %) and the EDLS. This result confirms that the ARBFN learning strategy offers a better solution for the construction of an RBF network for adaptive image retrieval, compared to the two standard learning strategies.

Both the OLS and the EDLS strategies usually perform well under the opposite condition, where the training samples are sufficiently large [46], and where the data samples may not correlate closely to each other. In this experiment, it was observed that the EDLS achieved improvement after the first iteration (i.e.,  $\bar{Pr}(Iter. = 1) = 50.2\%$ ), because the retrieved data at  $Iter. = 0$  usually has a low

degree of correlation. Its performance, however, was reduced after two iterations as the retrieved samples became correlated more strongly. This suggests that the EDLS may not be suitable for constructing the RBF network under this learning condition. Using the same RBF widths, the OLS learning strategy was more stable and much better than the EDLS.

It was observed that the RBF centers critically influenced the performance of the RBF classifier, and that the RBF classifier constructed by matching all retrieved samples exactly to the RBF centers degraded the retrieval performance. The OLS algorithm was fairly successful at resolving this problem, by choosing the subset of the retrieved samples for the RBF centers. However, the OLS provided a less adequate RBF network, compared to the ARBFN. In ARBFN learning, each available positive sample was considered as important. Also, the centers were shifted by negative samples with the weighted norm parameters being updated during adaptive cycles. The ARBFN also managed well with the small set of samples encountered. The ARBFN is thus the most adequate model for the current application.

The retrieval performance of the ARBFN was next compared to the single-class model discussed in Sects. 2.2–2.3, using a new query set, which contained 59 images randomly selected from different categories. The methods compared include ARBFN, single-RBF, OPT-RF, and MAM. Two criteria were employed for performance measures: precision  $Pr$  measured from the top  $N_c$  images, where  $N_c$  was set to 10, 16, 25, and 50; and second, a precision versus recall graph. However, the relevance feedback was done only on the *top 16* retrieved images.

Table 2.14 summarizes the precision results averaged over all queries, measured from the top 10, 16, 25, and 50 retrieved images. It can be seen that the learning methods provided a significant improvement in each of the first three iterations. The ARBFN achieved the best precision results in all conditions, compared to the other methods discussed. At  $N_c = 10$ , ARBFN reached a near-perfect precision of 100 % after three iterations. This means that all the top ten retrieved images were relevant. The results also show that, at  $N_c = 16$ , more than 14 relevant images were presented in the top 16 ranking set. The most important precision results are perhaps those after the first iteration, since users would likely provide only one round of relevance feedback. It was observed that the ARBFN provided a better improvement than the other methods for this requirement.

Figure 2.5a–c illustrates the average precision versus recall figures after one, two, and three iterations, respectively. The behavior of the system without learning and the strong improvements with adaptive learning can easily be seen. In all cases, the precision at 100 % recall drops close to 0. This fact indicates that it was not possible to retrieve all the relevant images in the database, which had been pre-classified by the Corel Professionals. It is observed from Fig. 2.5a that the ARBFN was superior to the single-RBF at the higher recall levels, while both provided similar precision at the lower recall levels. Also, the ARBFN achieved better improvements than the single-RBF by up to 8.6 %, 7.3 % and 6.5 %, at one, two and three iterations, respectively.

**Table 2.14** Average precisions,  $\bar{P}r$ , compared at four settings of top matches ( $N_c$ ), obtained by retrieving 59 queries, using the Corel Database

$N_c$	Method	Average precision (%), $\bar{P}r(N_c)$			
		t = 0	t = 1	t = 2	t = 3
10	ARBFN	55.93	+32.03	+42.03	+43.56
	Single-RBF	55.93	+27.97	+39.15	+42.03
	MAM	55.93	+17.12	+19.32	+19.66
	OPT-RF	55.93	+24.07	+30.51	+32.37
16	ARBFN	47.67	+30.83	+39.30	+41.21
	Single-RBF	47.67	+26.48	+34.64	+38.45
	MAM	47.67	+13.88	+16.00	+16.21
	OPT-RF	47.67	+20.97	+23.83	+25.00
25	ARBFN	39.93	+26.44	+30.44	+31.19
	Single-RBF	39.93	+21.36	+26.58	+28.14
	MAM	39.93	+11.46	+12.47	+12.07
	OPT-RF	39.93	+17.02	+19.73	+20.00
50	ARBFN	30.03	+19.08	+20.58	+20.75
	Single-RBF	30.03	+15.29	+17.76	+18.44
	MAM	30.03	+8.24	+8.31	+8.17
	OPT-RF	30.03	+11.86	+12.17	+12.51

Interactive results are quoted relative to the  $\bar{P}r$  observed with the initial retrieval

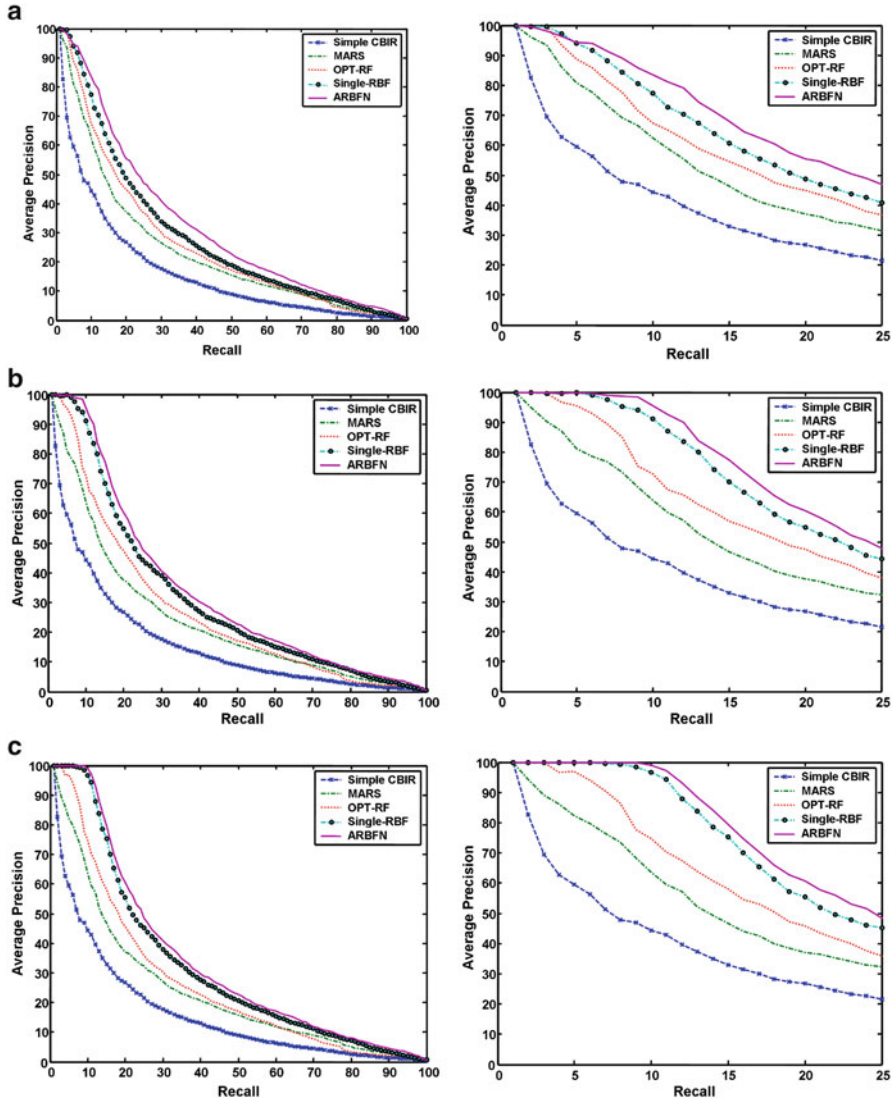
## 2.5 Bayesian Method for Fusion of Content and Context in Adaptive Retrieval

Adaptive retrieval method can be implemented to integrate visual content and contextual information through relevance feedback [47,48]. Contextual information refers to the statistical correlation across multiple images. In this section, a Bayesian framework is developed for fusion of content and context components. Specifically, the visual content analysis is associated with the likelihood evaluation, whereas the contextual information is represented by the *a priori* probability, learned through a maximum entropy algorithm.

### 2.5.1 Fusion of Content and Context

Let  $C$  represent the set of class labels and  $C = \{1, 2, \dots, C\}$ , where  $C$  is the number of classes. The class label of a particular image in a database is denoted  $c$ , where  $c \in C$ . Based on the *maximum a posteriori* probability (MAP) criterion which minimizes the classification error, the true class label is estimated with:

$$\hat{c} = \arg \max_{c \in C} P(c|\mathbf{x}, I) \quad (2.80)$$



**Fig. 2.5** Average precision versus recall figures, obtained by retrieving 59 queries, using the Corel database. Figures on the right are the zoom versions of the figures on the left. Note that, in each case, results obtained by the non-adaptive retrieval method are fixed, and used as a benchmark for other adaptive retrieval methods. (a) Results after first RF. (b) Results after second RF. (c) Results after third RF

where  $\hat{c}$  is the estimate of  $c$ ,  $\mathbf{I}$  is the background information, which exists with a well-formulated problem. In the context of the subsequent description, it represents a set of indexes of query images. Therefore,  $\mathbf{I}$  can be defined as  $\mathbf{I} = \{I_i | i =$



$1, 2, \dots, |\mathbf{I}|$ }, where  $|\mathbf{I}|$  is the number of query images,  $I_i \in \mathbf{C}$ ,  $i = 1, 2, \dots, |\mathbf{I}|$ . Using Bayes' theorem, the *a posteriori* probability can be written as

$$P(c|\mathbf{x}, \mathbf{I}) \propto p(\mathbf{x}|c, \mathbf{I}) P(c|\mathbf{I}), \quad (2.81)$$

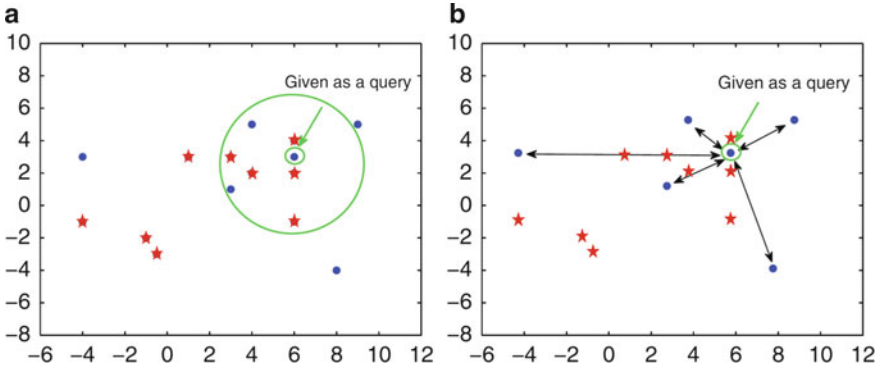
with the equality replaced by the proportionality due to the unimportance of the probability density function (PDF) of an observation, i.e.  $p(\mathbf{x}|\mathbf{I})$ , when the theorem is employed to solve a classification problem. Based on the meaning of the background information  $\mathbf{I}$ , we can assume the conditional independence between the observation  $\mathbf{x}$  and  $\mathbf{I}$  given the class label of the observation, i.e.  $\mathbf{x} \perp \mathbf{I}|c$ . Therefore, the *a posteriori* probability in Eq. (2.81) can be calculated through

$$P(c|\mathbf{x}, \mathbf{I}) \propto p(\mathbf{x}|c) P(c|\mathbf{I}) \quad (2.82)$$

The first term on the right-hand side of Eq. (2.82) is the PDF of the feature vector of the class  $c$ , which is considered as the content model characterizing the visual properties of that class. The second term is essentially a distribution of one class or candidate image, say  $c$ , conditional on a set of other classes or query images, collectively represented by  $\mathbf{I}$ . This is exactly the contextual information that characterizes the statistical relation between different classes or images. It will be shown that such contextual information can be learned from past user feedback for image retrieval. According to Eq. (2.82), the content and contextual information are integrated through the decision-level fusion in a multiplicative fashion.

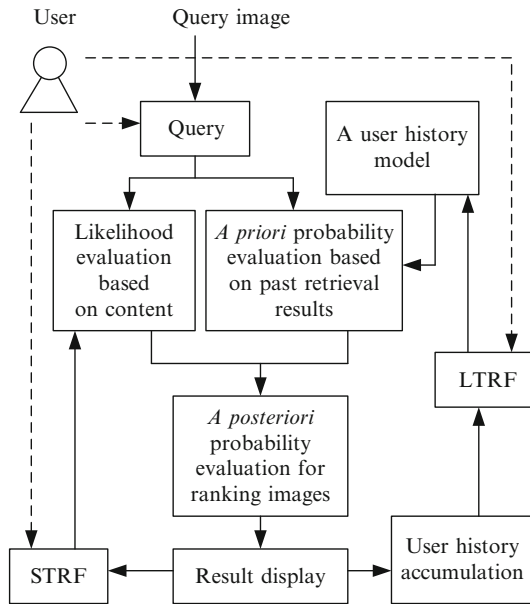
The Bayesian framework is applied to tackle the semantic gap of image retrieval by integrating short-term relevance feedback (STRF) and long-term relevance feedback (LTRF). STRF refers to the user interaction during a retrieval session consisting of a number of feedback iterations, such as query shifting and feature re-weighting. On the other hand, LTRF is the estimation of a user history model from past retrieval results approved by previous users. LTRF plays a key role in refining the degree of relevance of the candidate images in a database to a query. The STRF and LTRF play the roles of refining the likelihood and the *a priori* information, respectively, and the images are ranked according to the *a posteriori* probability. By exploiting past retrieval results, it can be considered as a retrieval system with memory, which incrementally learns the high level knowledge provided by users.

The underlying rationale of applying the Bayesian framework to image retrieval can be illustrated using Fig. 2.6, of which the gist is to boost the retrieval performance using some information extracted from the retrieval history. The two types of similarity measure are complementary to each other. Specifically, the similarity measure by the content-based component illustrated by the low-level feature space in Fig. 2.6a suffers from the semantic gap which can be alleviated using the contextual information. The links between relevant images in Fig. 2.6b are estimated by utilizing the co-occurrence of relevant images in the past retrieval results. At the same time, the contextual information can only be acquired by learning from the knowledge accumulated through the content-based component. The retrieval system, illustrated in Fig. 2.7, seamlessly integrates the content-based



**Fig. 2.6** The similarity measure in the content and context domains. (a) Semantic gap exists in the content domain. (b) There might not be sufficient data to extract accurate contextual information

**Fig. 2.7** Block diagram of the integration of STRF and LTRF in an adaptive retrieval system. The *solid and dashed directed lines* indicate the information flow and the user-controlled components, respectively



and the context-based methods into a mathematically justifiable framework. In the beginning, there is no available retrieval history from which to learn the context model but the system can still work using the content-based component and incrementally accumulate the retrieval results. When past retrieval results are available, the context component of the system performs LTRF by extracting information from the data gradually, which can be considered as a knowledge accumulation process. When a user presents a query, the content component of the system learns the user's information needs from the query through similarity measures and STRF. If the context component has been trained by the time a user queries the database,

the system is capable of integrating the useful information predicted using the context component and that learned using the content component. The *a posteriori* probability evaluated by the system is used to rank the images in the database.

### 2.5.2 Content-Based Likelihood Evaluation in Short-Term Learning

The visual content model of a certain semantic class, e.g.  $c$ , is the parametric form of the distribution of the visual features of that class. The parameters of the model are adapted to a given set of training data of class  $c$  through a supervised learning procedure. A visual content model plays the role of evaluating the likelihood of a visual feature with respect to a certain class. The support vector machine (SVM) is selected as the key component of the content model to evaluate the likelihood. L1 norm is also employed in addition to SVM for calculating the likelihood using the content model. At the same time, it should be noted that the formulation of the Bayesian framework requires that the output of the visual content model comply with the definition of a PDF. To this end, the exponential function is employed, i.e.  $h(s) = \exp(s)$ ,  $s \in \mathbb{R}$ , to convert the discriminant function of SVM into a PDF. The selection of the above exponential function is based on the following consideration. First, it is monotonically increasing, resulting in the preservation of the physical interpretation of the algebraic distance between a sample and the decision boundary. Second, it is positive. Since the total integral of a function must be equal to unity, appropriate normalization is necessary. Finally, representing the discriminant function of SVM corresponding to the  $c$ -th class as  $f_c(\mathbf{x})$  and substituting it for the variable  $s$  in the exponential function followed by normalization, we obtain

$$p(\mathbf{x}|c) = \frac{1}{A} \exp(f_c(\mathbf{x})) \quad (2.83)$$

where  $A = \int \exp(f_c(\mathbf{x})) d\mathbf{x}$ .

#### 2.5.2.1 Using the Nearest Neighbor (NN) Method

The nearest neighbor (NN) method returns the top  $K$  images on the list, which is ranked based on the similarity measure between the feature of the query and that of each of the candidate images, where  $K \ll C$ . The L1-norm is used as the distance function for the NN method. In adaptive retrieval, the query is refined using the method of query point movement [i.e., Eq. (2.8)]. To calculate the likelihood, the exponential function in Eq. (2.83) converts the L1-Norm into a similarity function, i.e.

$$p(\mathbf{x}_q|c) = \frac{1}{A} \exp(f_c(\mathbf{x}_q)) = \frac{1}{A} \exp(-|\mathbf{x}_q - \mathbf{x}_c|) \quad (2.84)$$

where  $A = \int \exp(-|\mathbf{x}_q - \mathbf{x}_c|)$ , is the normalization constant,  $\mathbf{x}_q$  denotes the feature vector of a query, and  $\mathbf{x}_c$  denotes a candidate image. When the likelihood is calculated using the L1 norm, the corresponding negative distance function should be substituted into the exponential function because the similarity is a decreasing function of the distance between features.

### 2.5.2.2 Using the Support Vector Machine Active Learning (SVMAL) Method

SVM is a powerful tool for pattern recognition because it maximizes the minimum distance between the decision hyperplane and the training samples so as to minimize the generalization error. Given training samples  $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ , where  $\mathbf{x}_i \in \mathbb{R}^P$ ,  $y_i \in \{-1, 1\}$  is the ground-truth label of  $\mathbf{x}_i$ , the optimal hyperplane can be represented as  $f(\mathbf{x}) = \sum_{i=1}^N \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$  where  $K(\mathbf{x}_i, \mathbf{x})$  is the kernel function,  $\alpha_i$  is the Lagrangian multiplier, and  $b$  is the bias. Due to the sparse sample problem of the relevance feedback learning, the active learning method was introduced into the learning process, whereby the most informative images are shown to request user-provided labeling, resulting in the support vector machine active learning (SVMAL)-CBIR [49]. Since the output of an SVM with respect to a sample is the oriented distance from the sample to the hyperplane, the value could be either positive or negative. Therefore, the exponential function is employed again to convert the value of the discriminant function. When selecting radial basis functions as the kernel, we obtain

$$p(\mathbf{x}_q|c) = \frac{1}{A} \exp(f_c(\mathbf{x}_q)) = \frac{1}{A} \exp\left(\sum_{i=1}^N \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_q) + b\right) \quad (2.85)$$

where  $A = \int \exp(\sum_{i=1}^N \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_q) + b)$  is the normalization constant.

### 2.5.3 Context Model in Long-Term Learning

This part aims at calculating the  $P(c|\mathbf{I})$  in Eq.(2.82), which is the contextual information about  $c$  inferred based on the  $\mathbf{I}$ . Without  $\mathbf{I}$ , the probability mass of  $c$  is uniformly distributed over the class ensemble  $\mathbf{C}$  without  $\mathbf{I}$ . Due to the statistical dependence across different classes, however, the distribution of  $c$  conditional on  $\mathbf{I}$  will deviate from the uniform distribution once  $\mathbf{I}$  is available. As a result, the classes that are more strongly correlated with  $\mathbf{I}$  have higher probabilities than the others do. Since the problem is essentially the estimation of a conditional probability mass function (PMF), a typical train of thought leads to the conventional approach that calculates the conditional probability through  $P(c|\mathbf{I}) = P(c, \mathbf{I})/P(\mathbf{I})$ , for which we need a set of training samples belonging to the Cartesian product of  $|\mathbf{I}| + 1$   $\mathbf{C}$ 's.

Regardless of the approach to estimating  $P(c, \mathbf{I})$  and  $P(\mathbf{I})$ , there are two problems with above estimation of  $P(c|\mathbf{I})$ . First, the background information  $\mathbf{I}$  may include different numbers of indexes, which requires separate estimations of the model for different sizes of  $\mathbf{I}$ . Second, when collecting training data, we cannot guarantee enough or even available samples for a certain configuration of  $c$  and  $\mathbf{I}$ , where the configuration refers to a particular instantiation of the number of random variables of  $c \cup \mathbf{I}$  and their values.

To deal with the estimation of the context model efficiently, the  $P(c|\mathbf{I})$  is approximated using a distribution of a set of binary random variables estimated based on the maximum entropy (ME) principle. In this approach, an image is represented using a  $C$ -dimensional vector of binary random variables, denoted  $\mathbf{Y} = (Y_1, Y_2, \dots, Y_C)^t$ , where the value of each variable  $Y_c$  is defined by

$$Y_c = \begin{cases} 1 & \text{if the } c\text{-th image is relevant to a query} \\ 0 & \text{otherwise} \end{cases} \quad (2.86)$$

Instead of being from the Cartesian product of  $|\mathbf{I}| + 1$   $C$ 's, the data utilized by the context modeling procedure belong to the set of vertices of a  $C$ -dimensional hypercube. Given a set of  $N$  training samples, denoted  $\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_N$ , we can estimate the  $P(\mathbf{Y})$  and then calculate the conditional probability  $P(Y_c | Y_{I_1}, Y_{I_2}, \dots, Y_{I_{|\mathbf{I}|}})$ , which is represented as  $P(Y_c | \mathbf{Y}_I)$  in what follows. To approximate the  $P(c|\mathbf{I})$  in Eq. (2.82), the following formula is utilized

$$P(c|\mathbf{I}) = \frac{P(Y_c | \mathbf{Y}_I)}{\sum_{v=1}^C P(Y_v | \mathbf{Y}_I)} \quad (2.87)$$

As the size of the concept ensemble, i.e.  $C$ , grows, the computational intensity of the calculation of  $P(Y_c | \mathbf{Y}_I)$  increases exponentially. Therefore, it would be more efficient if we can directly estimate  $P(Y_c | \mathbf{Y}_I)$  based on a set of training samples. To this end, the ME approach demonstrated in [50] is employed, which estimates a conditional distribution by maximizing its Rényi entropy. Essentially, the ME principle states that the optimal model should only respect a certain set of statistics induced from a given training set and otherwise be as uniform as possible. The ME approach searches for the conditional distribution  $P(Y_c | \mathbf{Y}_I)$ , with the maximum entropy, among all the distributions which are consistent with a set of statistics extracted from the training samples. Therefore, it can be considered as constrained optimization, which is formulated as

$$\max_{P(Y_c | \mathbf{Y}_I) \in [0,1]} - \sum_{y_c, \mathbf{y}_I} \hat{P}(\mathbf{Y}_I = \mathbf{y}_I) P(Y_c = y_c | \mathbf{Y}_I = \mathbf{y}_I)^2, \quad (2.88)$$

subject to:

$$\frac{\sum_{y_I} \hat{P}(Y_I = y_I) P(Y_c = y_c | Y_I = y_I) f_k}{\hat{P}(f_k)} = \hat{P}(f_c | f_k), k \in \{0\} \cup I, \quad (2.89)$$

where  $c \in C$  and  $c \notin I$  because  $P(Y_c = 1 | Y_I = 1) \equiv 1$  for  $c \in I$ . In addition,  $\hat{P}(\cdot)$  represents the empirical probabilities directly estimated from the training samples,  $f_c = Y_c$  and  $f_k = Y_k$  when  $k \neq 0$  and  $f_k = 1$  otherwise. Using a matrix-based representation, solving the above optimization leads to the result that

$$P = M \times N^{-1} \times f \quad (2.90)$$

where

$$P = [P(Y_{a1} | Y_I), P(Y_{a2} | Y_I), \dots, P(Y_{a_{|C/I|}} | Y_I)]^t \quad (2.91)$$

$$M = \begin{bmatrix} \hat{P}(f_{a1} | f_0) & \hat{P}(f_{a1} | f_{I_1}) & \cdots & \hat{P}(f_{a1} | f_{I_{|I|}}) \\ \hat{P}(f_{a2} | f_0) & \hat{P}(f_{a2} | f_{I_1}) & \cdots & \hat{P}(f_{a2} | f_{I_{|I|}}) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{P}(f_{a_{|C/I|}} | f_0) & \hat{P}(f_{a_{|C/I|}} | f_{I_1}) & \cdots & \hat{P}(f_{a_{|C/I|}} | f_{I_{|I|}}) \end{bmatrix} \quad (2.92)$$

$$N = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \hat{P}(f_{I_1} | f_0) & 1 & \cdots & \hat{P}(f_{I_1} | f_{I_{|I|}}) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{P}(f_{I_{|I|}} | f_0) & \hat{P}(f_{I_{|I|}} | f_{I_1}) & \cdots & 1 \end{bmatrix} \quad (2.93)$$

$$f = [f_0, f_1, \dots, f_{I_{|I|}}]^t \quad (2.94)$$

and  $|C/I| = \{a_1, a_2, \dots, a_{|C/I|}\}$ .

### 2.5.4 Experimental Result

In the experiment, four methods summarized in Table 2.15 were compared. A total of 200 classes of images was selected from the COREL image collection, with 50 images in each class. The resulting 10,000 images and the vendor-defined categories were used as the database and the ground truth for evaluating the performance. From the database, 10 queries are selected from each of the 200 classes, resulting in 2,000 queries being selected, each of which is composed of two different images. Under

**Table 2.15** Comparison of learning methods

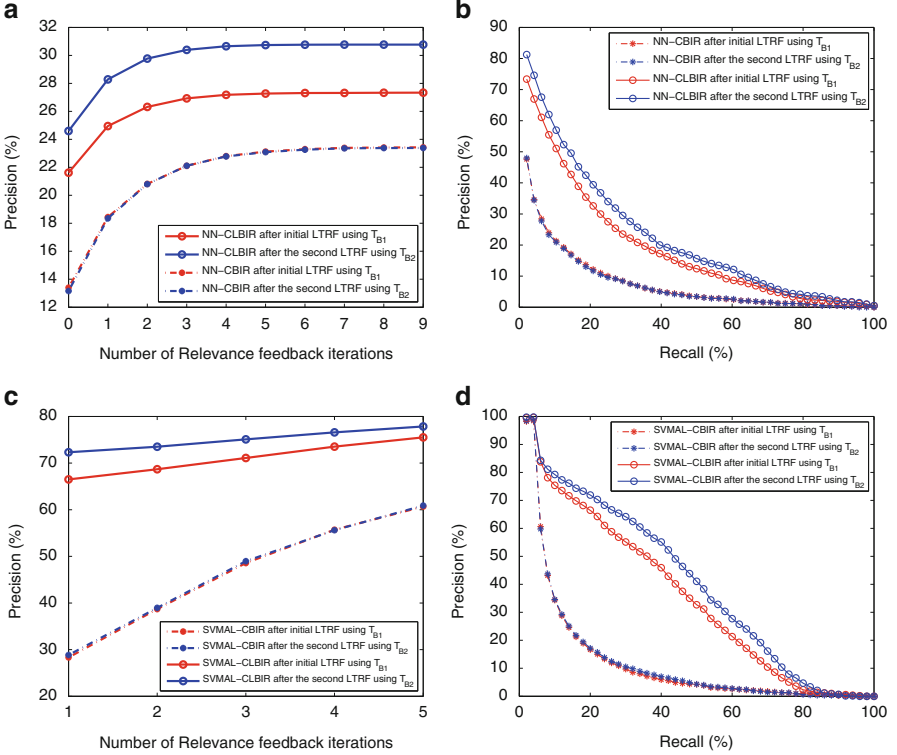
Method	Learning criterion	STRF	LTRF
NN-CBIR	Adaptive image retrieval method, using <i>nearest neighbor</i> (NN) criterion, where L1-norm is employed as the distance function and Eq. (2.8) is employed for a query modification	✓	×
SVMAL-CBIR	Adaptive image retrieval method using <i>support vector machine active learning</i> (SVMAL)	✓	×
NN-CLBIR	Collaborative Bayesian Image Retrieval (CLBIR), using Eq. (2.84) for estimation of $p(\mathbf{x}_q c)$ and Eq. (2.87) for estimation of $P(c I)$	✓	✓
SVMAL-CLBIR	Collaborative Bayesian Image Retrieval (CLBIR), using Eq. (2.85) for estimation of $p(\mathbf{x}_q c)$ and Eq. (2.87) for estimation of $P(c I)$	✓	✓

the query-by-example retrieval paradigm, the average of the features of the two images is used as the feature of an exemplar image.

To facilitate the subsequent elaboration, the query subsets which consist of the first five queries, the sixth through the eighth, and the ninth and the tenth in each class, are denoted  $T_A$ ,  $T_{B,1}$ , and  $T_{B,2}$ , where  $|T_A| = 1,000$ ,  $|T_{B,1}| = 400$ , and  $|T_{B,2}| = 600$ . Such a query set selection guarantees that the system trained using the LTRF will be tested based on previously unseen samples.  $T_A$  was used when there is no accumulated high-level knowledge, i.e. before LTRF happens. In such a case, only STRF is involved, and the “nearest neighbor collaborative Bayesian image retrieval” (NN-CLBIR) and the “support vector machine active learning collaborative Bayesian image retrieval” (SVMAL-CLBIR) are essentially the same as the NN-CBIR and SVMAL-CBIR because the *a priori* distribution of the candidate images is uniform. After the initial LTRF, the CLBIR systems are expected to present better performance in general thanks to the accumulated knowledge, while the STRF still improves the results with respect to each specific query.  $T_{B,1} \cup T_{B,2}$ , comprising 1,000 images, was used to verify the improvement after the initial LTRF. During the operation of the CLBIR systems, the new retrieval results after the initial LTRF are gradually accumulated, and a second LTRF can be carried out upon a certain point. The retrieval results corresponding to  $T_{B,1}$  were used to perform an incremental update of the system, i.e. the second LTRF, after which the performance was evaluated using  $T_{B,2}$ .

To capture various visual properties of the images, three types of low-level descriptors are selected, including global color histogram in Hue-Saturation-Value (HSV) space, color layout in YCbCr space [92], as well as Gabor wavelet [91].

Shown in Fig. 2.8a is the comparison between NN-CBIR and NN-CLBIR in terms of the average precision  $\bar{Pr}$  as a function of the number of iterations of STRF. The precision is given by  $Pr = N_C/N_R$ , where  $N_C$  and  $N_R$  are the numbers of relevant images and retrieved images, respectively. The precision is measured in the top  $N_R = 48$  in this case. Using the query set  $T_{B,1}$ , the improvement due to LTRF based on past retrieval results with respect to the query set  $T_A$  is obvious, and the effect of STRF can also be observed. After the second LTRF, the performance of NN-CLBIR using query set  $T_{B,2}$  is further enhanced due to more accumulated



**Fig. 2.8** (a) Comparison between the performance of NN-CBIR and NN-CLBIR in terms of the average precision versus the number of relevance feedback iterations; (b) comparison between the performance of NN-CBIR and NN-CLBIR in terms of the precision versus recall after the first retrieval iteration; (c) comparison between the performance of SVMAL-CBIR and SVMAL-CLBIR in terms of the average precision versus the number of relevance feedback iterations; (d) comparison between the performance of SVMAL-CBIR and SVMAL-CLBIR in terms of the precision versus recall after the first retrieval iteration

knowledge through the LTRF. Based on the same query set, the performance of NN-CBIR remains unchanged. To test the performance in terms of ranking ability, the precision-versus-recall curve (PRC) is employed. The recall is defined as  $R = N_C / N_G$ , where  $N_G$  is the number of images in the same classes as that of the query. The precision is averaged over all queries at each different recall value. The PRC after the initial retrieval is shown in Fig. 2.8b. Higher precision values at a certain recall indicates more relevant images being ranked ahead of irrelevant ones, i.e. to reach the recall value, a smaller set of retrieved images has to be processed. Based on this fact, the advantage of the integration of user history as high-level knowledge with the content analysis can be demonstrated based on the comparison in Fig. 2.8b.





**Fig. 2.9** Retrieval results for the subjective evaluation of the performance improvement resulting from extended user history; (a) based on the user history model trained using 2,000 past retrieval results; (b) based on the user history model trained using 3,200 past retrieval results

The comparison shown in Fig. 2.8c, d is for the same purpose of performance evaluation as that described above, and the difference lies with the approach to the content analysis for the likelihood computation, which is based on the output of the SVM employed for the active learning-based STRF. In this case,  $N_R = 20$  was adopted for the evaluation of precision as a function of the number of STRF iterations, and  $N_C = 50$  for the evaluation of PRC. Since the initial retrieval is just random ranking, the precision was evaluated starting from the first STRF iteration. Still, we can observe the improvement resulting from the integration through the Bayesian framework.

An interface with the NN-CLBIR enabled has been implemented to demonstrate the effectiveness of the system in terms of performance improvement by the accumulation of user history. Illustrated in Fig. 2.9a, b are the top 20 images retrieved using NN-CLBIR. Shown in the figure on the left is the result obtained using a system whose *a priori* knowledge was extracted from 1,000 user data, while on the right, the result is based on the *a priori* knowledge learned from 1,400 user data. The query is selected from the semantic class of the theme soldier, and the last four images do not belong to this class in Fig. 2.9a. Nonetheless, all of the top 20 images are relevant to the query.

## 2.6 Summary

The kernel approach makes use of a nonlinear kernel-induced inner product, instead of the traditional Euclidean inner product, to measure the similarity metrics of two vectors. In a relevance feedback session, the nonlinear kernel approach implements the nonlinear mapping function to analyze the role of the users in perceiving image similarity. This results in a high performance machine that can cope with the small size of the training sample set and the convergence speed. The new learning algorithms for the nonlinear kernel-based RF can be categorized into two groups. The first group includes the single-class RBF, the adaptive RBF, the gradient-descent-based learning, where hard constraints are used to force a clear separation on the RF samples. Then, in the second group, soft constraints are used to allow more support vectors to be included in the so-called fuzzy RBF formulations. Much of the chapter is meant to build the theoretical footing for the machine learning models in the subsequent chapters.

In addition, the nonlinear-kernel approach in a STRF is extended to a Bayesian fusion model. The STRF represents a content component that can be incorporated with a context component in a LTRF, through a Bayesian framework. This can be considered as a retrieval system with a memory, which can incrementally accumulate high-level semantic knowledge, assisting in bridging the semantic gap in future retrieval performed by prospective users.

Multimedia Database Retrieval

Technology and Applications

Muneesawang, P.; Zhang, N.; Guan, L.

2014, XII, 350 p. 142 illus., 111 illus. in color.,

Hardcover

ISBN: 978-3-319-11781-2