Chapter 5

Embedded multi-resolution mixture models

Since, in practice, there is no way of rendering feature representation a trivial problem, it is important to always rely on the most expressive density models available. In this chapter, we review several models that have been proposed in the retrieval literature and show that they are particular cases of a parametric family of densities know as mixture models. A mixture model is in turn shown to define a collection of embedded probabilistic descriptions over subspaces of the original feature space.

When combined with a multi-resolution feature transformation, this embedded representation leads to an interesting extension of the color histogram that provides explicit control over the trade-off between spatial support and invariance. We present experimental evidence that the embedded multi-resolution mixture representation outperforms the standard methods for color- and texture-based retrieval, even on the specific domains for which these techniques were designed. This confirms that the new representation can account for both color and texture and should do well on generic image databases.

5.1 Spatially supported representations

The main challenge for a feature representation designed to account for both color and texture is to combine tractability on high dimensions with expressive power to model complicated densities. We have already established that the two most popular representations in current use cannot fulfill this goal: the Gaussian assumption is too simplistic, the histogram model does not scale to high dimensional spaces. There have been, however, some efforts to extend these representations into usable joint models of color and texture.

5.1.1 Extensions to the Gaussian model

The simplest among such solutions is to represent the class-conditional densities by finite collections of their *moments*

$$\gamma_i^p = E_{\mathbf{x}|Y}[\mathbf{x}^p|Y=i] = \int \mathbf{x}^p P_{\mathbf{X}|Y}(\mathbf{x}|i)d\mathbf{x},$$

where γ_i^p is the p^{th}-order moment of the density associated with image class i. This is an extension of the Gaussian model that can only account for second-order moments.

While, theoretically, any density can be characterized by a collection of moments [124], it may take a large number number of them for the characterization to be accurate. The difficulty of determining how many moments are enough in any given situation may be the reason why the approach has not been extensively pursued. While the use of more than two moments has been advocated by some authors [167, 101], the total number is usually kept small. It is therefore not clear that the resulting characterization will be expressive enough to model complex densities, especially in high dimensions.

One answer to this problem is to arbitrarily divide the image into regions and compute moments for each region [166]. If there are M regions, this is equivalent to the probabilistic model

$$P_{\mathbf{X}|Y}(\mathbf{x}|i) = \frac{1}{M}\sum_{r=1}^{M} \hat{P}_{\mathbf{X}|R,Y}(\mathbf{x}|r,i) \tag{5.1}$$

where $\hat{P}_{\mathbf{X}|R,Y}(\mathbf{x}|r,i)$ is the moment-based approximation to the density of the r^{th} region of the i^{th} image class. Since the total number of moments is proportional to the number of regions, the resulting representation is compact if and only if the segmentation is limited to a small number of large regions. This usually leads to arbitrary image partitions that are not tailored to the image statistics.

5.1.2 Extensions to the histogram model

It has been suggested that the solution to the limitations of color-based retrieval is to rely on histograms of spatially supported features [156, 125]. This fails to realize that the main limitation of color-based retrieval is the histogram itself. In result, feature spaces are constrained to small dimensions and a significant amount of information is lost. We have already shown that this is not a good idea.

A better extension to the histogram is to explicitly augment this model with spatial information. This is the rationale behind representations like *color coherence vectors* (CCVs) [126] and *color correlograms* [66]. CCVs divide the image pixels into two classes, coherent and non-coherent (where the coherency of a pixel is a function of the number of similar pixels that are spatially connected to it), and compute an histogram for each class. They provide a probabilistic model of the form

$$P_{\mathbf{X}|Y}(\mathbf{x}|i) = \sum_{c=0}^{1} P_{\mathbf{X}|\mathcal{C}(\mathbf{X}),Y}(\mathbf{x}|c,i) P_{\mathcal{C}(\mathbf{X})}(c), \tag{5.2}$$

where $\mathcal{C}(\mathbf{x}) = 1$ for coherent pixels, $\mathcal{C}(\mathbf{x}) = 0$ for non-coherent ones, and $P_{\mathbf{X}|\mathcal{C}(\mathbf{X}),Y}(\mathbf{x}|c,i)$ are the color histograms for each case.

Correlograms are spatial extensions of the histogram, registering the relative frequencies of occurrence of color-pairs on pixels separated by a pre-determined set of spatial distances. They are a simplification the co-occurrence matrices that have been widely studied in the texture literature during the seventies [62, 61, 178, 144]. Under the co-occurrence model a texture is characterized by a collection of matrices $\{\mathbf{M}_{\mathbf{d}_k}\}_{k=1}^{K}$. Each matrix is associated with a distance vector $\mathbf{d}_k = (d_k, \theta_k)$, of norm d_k and angle θ_k with the horizontal axis, and contains color co-occurrence probabilities for pixels separated by that distance. If $C_{l,m}$ is the color of pixel (l, m), then the element (i, j) of $M_{\mathbf{d}_k}$ contains the relative frequencies with which two pixels separated by $\mathbf{d}_k$ occur on the image, one with color i and the other with color j

$$[M_{\mathbf{d}_k}]_{i,j} = P((l,m),(p,q)|C_{l,m} = i, C_{p,q} = j, (l,m) = (p,q) + \mathbf{d}).$$

The main limitation of co-occurrence matrices is the large number of probabilities that have to be estimated, leading to significant computational complexity and poor estimates (there may not even be any observations for a significant number of the matrix elements). The color correlogram tries to avoid some of these difficulties by simply averaging co-occurrence matrices with respect to θ. In practice, this is usually not enough to make the implementation feasible and the correlogram is replaced by the *auto-correlogram* which only consider pairs of pixels of the same color. The auto-correlogram is the collection of vectors

$$[\mathbf{M}_{d_k}]_i = P((l,m),(p,q)|C_{l,m} = C_{p,q} = i, \rho((l,m),(p,q)) = d_k), \tag{5.3}$$

where ρ is usually the L^∞ norm.

For both CCVs and auto-correlograms, retrieval is based on straightforward extensions of the standard histogram similarity metrics. It has been shown experimentally that auto-correlograms achieve the best performance in this class of representations, significantly outperforming the histogram [66]. This is not surprising since the auto-correlogram accounts for both color and texture.

5.1.3 Vector quantization

Since both the exponential dependence on dimensionality and sparseness of the histogram are a consequence of the rigid rectangular partition of the feature space of (2.25), a final solution is to adapt this partition to the particular characteristics of the image data. One possible way to do this is to vector quantize [56, 91] the feature space. A *vector quantizer* (VQ) is a map

$$\mathcal{Q} : \mathcal{R}^n \rightarrow \mathcal{C},$$

where $\mathcal{C} = \{\mathbf{y}_1, \ldots, \mathbf{y}_C\}$ is a finite set of *reconstruction vectors*, or *codebook*, and $\mathbf{y}_i \in \mathcal{R}^n$. This map defines a partition of R^n into C regions $\{\mathcal{R}_1, \ldots, \mathcal{R}_C\}$, associating a reconstruction vector $\mathbf{y}_i$ with each region

$$\mathcal{Q}(\mathbf{x}) = \sum_{i=1}^{C} \mathbf{y}_i \, \chi_{\mathcal{R}_i}(\mathbf{x}), \tag{5.4}$$

where $\chi_{\mathcal{R}_i}(\mathbf{x})$ are set indicator functions (2.2).

For a random variable $\mathbf{x}$ characterized by a density $P_{\mathbf{X}}(\mathbf{x})$ and a distortion measure $d(\mathbf{x}, \mathcal{Q}(\mathbf{x}))$, the average distortion introduced by a VQ is

$$\mathcal{D} = \int d(\mathbf{x}, \mathcal{Q}(\mathbf{x}))\, P_{\mathbf{X}}(\mathbf{x}) d\mathbf{x}.$$

It can be shown [56, 91] that, when the goal is to minimize this distortion, the optimal partition for a fixed codebook must satisfy the *nearest-neighbor condition*

$$\mathcal{R}_i = \{\mathbf{x} : d(\mathbf{x}, \mathbf{y}_i) \leq d(\mathbf{x}, \mathbf{y}_j), \forall j \neq i\},$$

while the optimal codebook for a given partition must satisfy the *generalized-centroid condition*

$$\mathcal{Q}(\mathbf{x}) = \min_{\mathbf{y}_i}\{E[d(\mathbf{x}, \mathbf{y}_i)|\mathbf{x} \in \mathcal{R}_i]\}.$$

In practice, the distortion measure is usually one of the quadratic distances discussed in Chapter 2. The mean squared error is a particularly popular choice leading to

$$\mathcal{R}_i = \{\mathbf{x} : ||\mathbf{x} - \mathbf{y}_i||^2 \leq ||\mathbf{x} - \mathbf{y}_j||^2, \forall j \neq i\}, \tag{5.5}$$

and

$$\mathbf{y}_i = \mathcal{Q}(\mathbf{x}) = \{E[\mathbf{x}|\mathbf{x} \in \mathcal{R}_i]\}. \tag{5.6}$$

Under this distance, given the codebook $\{\mathbf{y}_1, \ldots, \mathbf{y}_C\}$, the feature vectors extracted from each image are quantized by simply finding the reconstruction vectors that are closest to them under the Euclidean norm. Each feature vector is therefore replaced by a scalar (the index or label of the corresponding reconstruction vector) and, for retrieval, the entire image can be represented by the histogram of the labels. Standard histogram metrics, such as L^p norms, can then be used to evaluate image similarity [70, 183, 68, 118, 109, 193, 163, 139].

While vector quantization reduces the sparseness and dimensionality problems, there are a few significant problems associated with label histograms. The first is the assumption that it is possible to find a universal VQ that will be a good representation for all the images in the database. It is not clear that this is the case or, even when it is, if the resulting codebook will have a manageable size. Second, a learned VQ must typically be retrained whenever new image classes are added to the database. These problems can be solved by avoiding histogram similarity measures, and simply reverting back to the ML similarity criteria of (2.16), using VQ-based density estimates. In other words, instead of learning a universal VQ and evaluating similarity between label histograms (the quantization view), a VQ is learned for each image class and similarity evaluated through the ML criteria, using VQ-based density estimates (the probabilistic retrieval view) [136, 192, 190, 181]. For this, we need a probabilistic, or *generative*, interpretation for vector quantization. We address this issue in the next section, where we will consider a generic family of probability models that encompasses, as special cases, the majority of the representations discussed so far.

5.2 Mixture models

A mixture density [177, 143, 13] has the form[1]

$$P(\mathbf{x}) = \sum_{c=1}^{C} P(\mathbf{x}|\omega_c)P(\omega_c), \tag{5.7}$$

where C is a number of feature classes, $\{P(\mathbf{x}|\omega_c)\}_{c=1}^{C}$ a sequence of *feature class-conditional densities* or *mixture components*, and $\{P(\omega_c)\}_{c=1}^{C}$ a sequence of *feature class probabilities*. Mixture densities model processes with hidden structure: one among the C feature classes is first selected according to the $\{P(\omega_c)\}$, and the observed data is then drawn according to the respective feature class-conditional density. These densities can be any valid probability density functions, i.e. any set of non-negative functions integrating to one.

Because the complexity of the mixture model is proportional to the complexity of the feature class-densities, if the latter are tractable in high dimensions, the former will also be. This is indeed the case for most of the feature class-conditional densities in common use and, in particular, the Gaussian. Furthermore, because it is, by definition, a multi-modal model, the mixture density can easily capture the details of complex densities.

In particular, Li and Barron have recently shown [90, 88] that if $\mathcal{C}$ is the space of all convex combinations[2] of a density $P_{\mathbf{X}|\Theta}(\mathbf{x}|\theta)$ parameterized by θ and F_k a k-component

[1]In this section, we drop the dependence on the image class $Y = i$ which is always implicit and the subscript from all densities since the random variables are clear from the context. For example, we write $P(\mathbf{x}|\omega_c)$ instead of $P_{\mathbf{X}|\Omega}(\mathbf{x}|\omega_c)$. We also use the words "feature class" for the different feature sub-classes that may exist within each image class.

[2]This includes both finite and continuous mixtures models.

mixture of $P_{\mathbf{X}|\Theta}(\mathbf{x}|\theta)$ then, for any density F,

$$KL(F||F_k) \leq KL(F||F^*) + \frac{c_F^2 \gamma}{k},$$

where c_F is a constant that depends on F, γ depends on the family $P_{\mathbf{X}|\Theta}(\mathbf{x}|\theta)$, and $F^* = \inf_{G \in \mathcal{C}} KL(F||G)$. This bounds the difference between $KL(F||F_k)$ and $KL(F||F^*)$, which is a measure of the distance between F and $\mathcal{C}$. Obviously, if $F \in \mathcal{C}$ then $KL(F||F^*) = 0$. For Gaussian mixtures of covariance $\sigma \mathbf{I}$, $\gamma = O(n/\sigma^2)$, i.e. the bound is linear on the dimension of the space. Hence, by selecting k large enough, it is possible to make the bound arbitrarily tight as long as $\sigma > 0$. On the other hand, by making $\sigma \to 0$, it is always possible to make $KL(F||F^*) = 0$. This result therefore suggests that, by selecting σ small enough and then k large enough, it is always possible to approximate F arbitrarily well.

In practice, densities can usually be well approximated by mixtures with a small number of components. In this section, we show that most of the representations in current use for image retrieval are particular cases of the mixture model.

5.2.1 Some obvious relationships

By simply making $C = 1$, it is obvious from (5.7) that any parametric density is a particular case of the mixture model. Similarly, the representation of an image by a collection of global moments is nothing more than an approximation to a one-component mixture model. If the image is segmented and each region modeled by a different set of moments, we obtain the approximation of (5.1), where each feature class corresponds to an image region. Finally, a CCV is a 2-component mixture model, where the features are pixel colors, feature classes are determined by the coherence of those colors, and feature class-conditional densities are modeled by histograms.

5.2.2 Relationship to non-parametric models

It is also clear from (5.7) that, given a sample of observations $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\}$, by making the number of feature classes equal to the number of observations $|\mathbf{X}|$, assuming each class to be equally likely, and feature class conditional densities to be replicas of the same kernel

$\mathcal{K}_\Sigma(\mathbf{x})$ centered on the observations

$$P(\mathbf{x}) = \frac{1}{|\mathbf{X}|} \sum_{i=1}^{|\mathbf{X}|} \mathcal{K}_\Sigma(\mathbf{x} - \mathbf{x}_i) \tag{5.8}$$

we obtain what are usually called *Parzen* or *kernel* density estimates [39, 162, 48]. These models are traditionally referred to as non-parametric densities, even though they usually require the specification of a scale (or *bandwidth*) parameter Σ. One popular choice for the kernel $\mathcal{K}_\Sigma(\mathbf{x})$ is the Gaussian distribution, in which case Σ is a covariance matrix.

The kernel model can be seen as the limit case of the region-based moments approach, where a window is placed on top of each image pixel, a set of moments measured from the features extracted from that window, and the kernel defined by those moments. Of course, the mixture model supports any representation in between one single set of global moments and a different set of moments for each pixel.

5.2.3 Relationship to vector quantization

Several authors have pointed out the existence of relationships between mixture models and vector quantization [3, 192, 145, 136, 78, 13, 27]. However, this tends to be done by comparing the standard algorithms for learning mixture models (the *EM algorithm* [36, 143, 13]) and vector quantizers (the *generalized Lloyd* or *LBG* algorithm [56, 91]), and showing that the later is a special case of the former.

This is somewhat unsatisfying since the learning algorithm does not completely specify the probabilistic model. In fact, various algorithms have been proposed both for vector quantization [91, 82, 3, 197, 147] and mixture density estimation [36, 90, 143, 108]. The comparison of algorithms therefore leads to different views on the relationship between the underlying representations [3, 192, 136, 27]. Here, we seek an explicit relationship between the probabilistic models.

For this, we start by noticing that, associated with any mixture model, there is a *soft partition* of the feature space. In particular, given an observation $\mathbf{x}$, it is possible to assign that observation to each of the feature classes according to

$$P(\omega_i|\mathbf{x}) \quad = \quad \frac{P(\mathbf{x}|\omega_i)P(\omega_i)}{\sum_{k=1}^{C} P(\mathbf{x}|\omega_k)P(\omega_k)}$$

$$= \begin{cases} \frac{1}{1+\sum_{k\neq i}\frac{P(\mathbf{x}|\omega_k)P(\omega_k)}{P(\mathbf{x}|\omega_i)P(\omega_i)}}, & \text{if } P(\mathbf{x}|\omega_i)P(\omega_i) > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{5.9}$$

The following theorem makes explicit the relationship between vector quantization and mixture models.

Theorem 5 *If* $\mathbf{x}$ *is a random vector distributed according to a Gaussian mixture*

$$P_\epsilon(\mathbf{x}) = \sum_c P(\omega_c)\mathcal{G}(\mathbf{x}, \mu_c, \mathbf{\Sigma}_c(\epsilon))$$

with covariances

$$\mathbf{\Sigma}_c(\epsilon) = \epsilon\mathbf{I}, \forall c,$$

then

$$\lim_{\epsilon\to 0} P_\epsilon(\omega_i|\mathbf{x}) = \begin{cases} 1, & \textit{if } ||\mathbf{x}-\mu_i|| \leq ||\mathbf{x}-\mu_k|| \forall k < i \\ 0, & \textit{otherwise} \end{cases} \tag{5.10}$$

and

$$\lim_{\epsilon\to 0} P_\epsilon(\mathbf{x}) = \sum_{i=1}^{C} \delta(\mathbf{x}-\mu_i)P(\omega_i). \tag{5.11}$$

where $\delta(\mathbf{x})$ *is the Dirac delta function (2.7).*

Proof: Since a mixture model with C classes of which z have zero probability is the same as a model with $C - z$ classes of non-zero probability, we assume, without loss of generality, that all the classes have non-zero probability, i.e.

$$P(\omega_i) > 0, \ \forall i.$$

For Gaussian feature class-conditional densities, (5.9) then becomes

$$P(\omega_i|\mathbf{x}) = \frac{1}{1+\sum_{k\neq i}\sqrt{\frac{|\mathbf{\Sigma}_i|}{|\mathbf{\Sigma}_k|}}\frac{e^{||\mathbf{x}-\mu_i||^2_{\mathbf{\Sigma}_i} - \log P(\omega_i)}}{e^{||\mathbf{x}-\mu_k||^2_{\mathbf{\Sigma}_k} - \log P(\omega_k)}}},$$

and for $\mathbf{\Sigma}_i = \epsilon\mathbf{I}, \forall i$,

$$P_\epsilon(\omega_i|\mathbf{x}) = \frac{1}{1+\sum_{k\neq i}\frac{P(\omega_k)}{P(\omega_i)}e^{\frac{1}{\epsilon}(||\mathbf{x}-\mu_i||^2 - ||\mathbf{x}-\mu_k||^2)}}.$$

Hence

$$\lim_{\epsilon\to 0} P_\epsilon(\omega_i|\mathbf{x}) = \begin{cases} a, & \text{if } ||\mathbf{x}-\mu_i|| \leq ||\mathbf{x}-\mu_k|| \ \forall k \neq i \\ 0, & \text{otherwise.,} \end{cases} \tag{5.12}$$

where

$$a = \frac{P(\omega_i)}{P(\omega_i) + \sum_{\{k:||\mathbf{x}-\mu_k||=||\mathbf{x}-\mu_i||\}} P(\omega_k)}.$$

Since the set $\{\mathbf{x} : ||\mathbf{x}-\mu_k|| = ||\mathbf{x}-\mu_i||\}$ has measure zero, (5.12) is equivalent to (5.10) almost everywhere. Furthermore, because some arbitrary tie-breaking rule is always necessary to vector quantize the points that lie on the boundaries between different cells, the same rule can be applied to (5.12) and the two equations are equivalent. Equation (5.11) is a direct consequence of the fact that the Gaussian density converges to the delta function as its covariance tends to zero [123].□

Equations (5.10) and (5.11) are nothing more than a generative model for a VQ. While (5.10) is simply the nearest neighbor condition (5.5), (5.11) is the probabilistic version of (5.4) and (5.6): given a cell label, a vector quantizer draws a sample from the conditional density associated with that cell. Since this density is the delta function centered on the cell's centroid, this sampling operation is equivalent to the combination of (5.4) and (5.6). It is, therefore, clear that a VQ is a particular case of the Gaussian mixture model.

5.2.4 Relationship to histograms

Equations (5.10) and (5.11) also provide an interpretation of a VQ as an histogram since the vector $\mathbf{H} = [P(\omega_1) \ldots P(\omega_C)]^T$ is estimated with normalized counts of the number of samples that land on each of the quantization cells. Because this is also the definition of histogramming, it is clear that histograms are a particular case of the mixture model. A special case of interest occurs when the reconstruction vectors μ_i are constrained to lie on a rectangular grid of size $h_1, \ldots, h_n$. In this case, the quantization cells become rectangles

$$P(\omega_i|\mathbf{x}) = \begin{cases} 1, & \text{if } |\mathbf{x}_1 - \mu_{i,1}| \leq \frac{h_1}{2}, \ldots, |\mathbf{x}_n - \mu_{i,n}| \leq \frac{h_n}{2} \\ 0, & \text{otherwise,} \end{cases}$$

and we obtain the standard histogram model of (2.25).

5.2.5 A unified view of feature representations

The relations between the various representations are depicted in Figure 5.1. Starting from the generic mixture model and a sample $\mathbf{X}$, if the number of classes C is set to the sample size and the classes assumed to be equally likely, we obtain a non-parametric model. If, on the other hand, $C = 1$ we have the parametric density defined by the particular kernel of choice. For C in between these two extrema, selecting a Gaussian kernel of zero covariance, leads to a vector quantizer and further restriction of the class centroids to lie on a rectangular grid leads to the standard histogram.

Independently of the number of classes C, the mixture model can always be approximated by storing a collection of moments instead of the complete class-conditional densities. In particular, for $C = 1$ we obtain the approximation by a collection of global moments and for C equal to a pre-defined number R of regions we obtain a region-based moment approximation. Finally, if there are only two classes (determined by the coherence of pixel colors) and each class-conditional density is represented by an histogram we obtain a CCV.

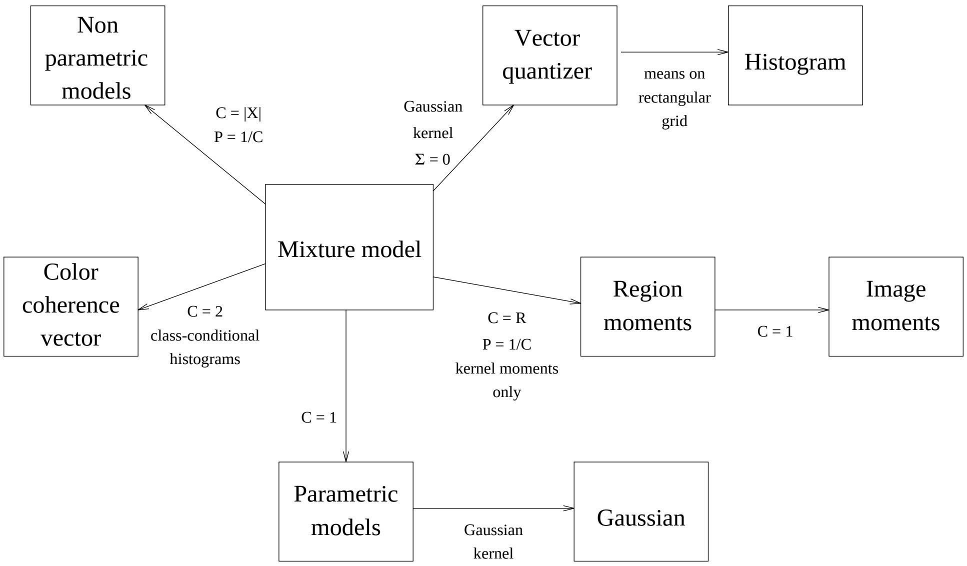


Figure 5.1: Relations between different feature representations. C is the number of feature classes and P the probability of each class.

Since all these feature representations are particular cases of the mixture model, it is expected that they will lead to suboptimal performance when applied to the retrieval problem. We now analyze this issue in greater detail.

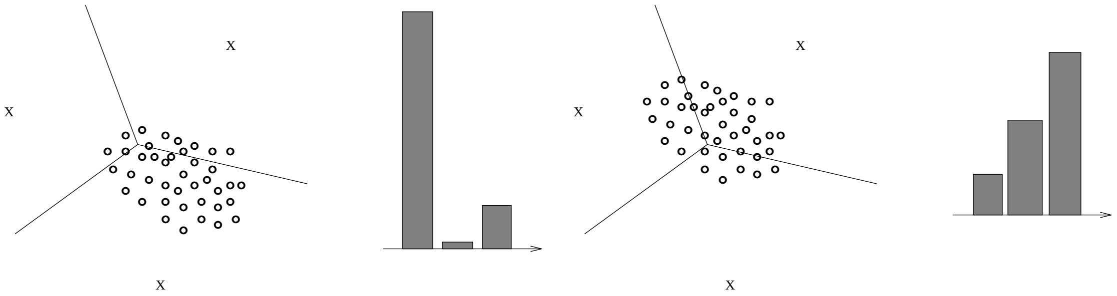
We start by noticing that because non-parametric models have as many degrees of freedom as the number of observations, these models are not compact. Consequently, the evaluation of equation (5.8) is computationally expensive (complexity proportional to the number of training features). On the other hand, there is no guarantee that the density estimates will be better than those provided by the mixture model, and there is usually no easy way to set the bandwidth parameter [162]. It is, therefore not clear that relying on a non-parametric model will justify the increase in retrieval complexity.

Relying on moment approximations is usually also not a good idea. While global moments provide a very crude approximation to the underlying density; region-based moment descriptions tend to be fairly arbitrary when the segmentation is not dictated by the image itself, and automated segmentation is well known to be a very difficult problem. This criticism is also valid for CCVs, where the classification into coherent and non-coherent pixels is rather arbitrary.

While non-parametric models have too many degrees of freedom, parametric ones have too few. Most parametric densities are unimodal and, as seen in the previous chapter, do not have enough expressive power to capture the details of the densities associated with real images. Standard histograms overcome this limitation but, given their intractability in high dimensions, are ineffective for joint modeling of color and texture. On the other hand, by adapting the partition of the space to the characteristics of the data, vector quantization can outperform the standard histogram with lower complexity. While this enables estimation on high-dimensional feature spaces, the fact that VQ-based estimates rely on a hard partition of the space restricts their robustness to small image variations. In fact, as illustrated in Figure 5.2, slight feature perturbations may lead to drastic changes in quantization and, consequently, image similarity.

Because mixture models rely on a soft partition of the feature space, they eliminate this problem. Furthermore, by allowing arbitrary covariances for each of the feature classes, Gaussian mixtures provide a much better approximation to the true density than the train of delta functions inherent to a VQ. The difficulty of accounting for more than the first-order moments of each cell is a well known problem for VQ-based density models [170, 3, 106, 145, 107, 26].

Despite the appealing properties of mixture modeling, very small attention has been devoted to their application to CBIR. To the best of our knowledge only two retrieval



a) b)

Figure 5.2: a) Partition of the feature space by a 3-cell VQ, a set of feature vectors, and the corresponding label histogram. b) For a universal VQ, small perturbations of the feature vectors can lead to entirely different label histograms.

systems have used mixture density estimates. One is the *Blob-world* system [7] which relied on Gaussian mixtures for image segmentation. These mixtures were not used, however, for feature representation. Instead, image regions were characterized by a low-dimensional feature vector and retrieval based on the Mahalanobis distance. The other is the *Candid* system [80, 79] that does rely on Gaussian mixtures for feature representation but does not apply them to high dimensional spaces of spatially supported features nor relies on a probabilistic similarity function. In this situation, the mixture representation does not have any significant advantage over the histogram.

5.3 Embedded multi-resolution mixture models

The discussion in the previous section suggests that there is no strong justification for relying on any of the above feature representations instead of the mixture model. In this section, we connect feature representation with feature transformation and show that, also from this point of view, there are good reasons to rely on the Gaussian mixture. A property of particular interest is that a projection of a high-dimensional Gaussian mixture into a linear subspace is still a Gaussian mixture.

Lemma 1 *Let* $\mathbf{X} \in R^n$ *be a random vector distributed according to the Gaussian mixture* *density*

$$P_{\mathbf{X}|Y}(\mathbf{x}|i) = \sum_{c=1}^{C} \pi_c \mathcal{G}(\mathbf{x}, \mu_c, \mathbf{\Sigma}_c), \tag{5.13}$$

and consider the projection map $\mathbf{V} = \pi_k(\mathbf{X}) = \mathbf{\Gamma}_k \mathbf{X}$ *defined in (4.3). Then*

$$P_{\mathbf{V}|Y}(\mathbf{v}|i) = \sum_{c=1}^{C} \pi_c \mathcal{G}(\mathbf{\Gamma}_k \mathbf{x}, \mathbf{\Gamma}_k \mu_c, \mathbf{\Gamma}_k \mathbf{\Sigma}_c \mathbf{\Gamma}_k^T). \tag{5.14}$$

Proof: Consider a Gaussian random vector $\mathbf{x}$ with

$$P_{\mathbf{X}|Y}(\mathbf{x}|y = i) = \mathcal{G}(\mathbf{x}, \mu_{\mathbf{x}}, \mathbf{\Sigma}_{\mathbf{x}}),$$

and define $\mathbf{V} = \pi_k(\mathbf{X})$. Since π_k is a linear transformation, $\mathbf{V}$ is also Gaussian distributed. Therefore, $P_{\mathbf{V}}(\mathbf{v})$ is uniquely determined by its mean and covariance. Using the relationships (4.9) and (4.10)

$$\mu_{\mathbf{v}} = \mathbf{\Gamma}_k \mu_{\mathbf{x}}$$
$$\mathbf{\Sigma}_{\mathbf{v}} = \mathbf{\Gamma}_k \mathbf{\Sigma}_{\mathbf{x}} \mathbf{\Gamma}_k^T,$$

it follows that

$$P_{\mathbf{V}|Y}(\mathbf{v}|i) = \mathcal{G}(\mathbf{\Gamma}_k \mathbf{x}, \mathbf{\Gamma}_k \mu_{\mathbf{x}}, \mathbf{\Gamma}_k \mathbf{\Sigma}_{\mathbf{x}} \mathbf{\Gamma}_k^T).$$

Applying this result to each of the C components of (5.13) we obtain (5.14).□

This lemma implies that the Gaussian mixture of (5.13) not only defines a probability density on R^n, but also a family of Gaussian mixture densities $\{P_{\mathbf{V}|Y}(\pi_k(\mathbf{x})|i)\}_{k=1}^{n-1}$ on the subspaces $R^1, \ldots, R^{n-1}$. We denote this collection as the family of *embedded mixture models* associated with the original distribution.

When, as is the case of the DCT features, the underlying feature space results from a multi-resolution decomposition this leads to an interesting interpretation of the mixture density as a family of densities defined over multiple image scales, each adding higher resolution information to the characterization provided by those before it. Disregarding the dimensions associated with high-frequency basis functions is therefore equivalent to modeling densities of low-pass filtered images. In the extreme case where only the first, or DC, coefficient is considered the representation is equivalent to the histogram of a smoothed version of the original image. This is illustrated in Figure 5.3.

The *embedded multi-resolution mixture* (EMM) model (embedded mixtures on a multi-resolution feature space) can thus be seen as a generalization of the color histogram, where

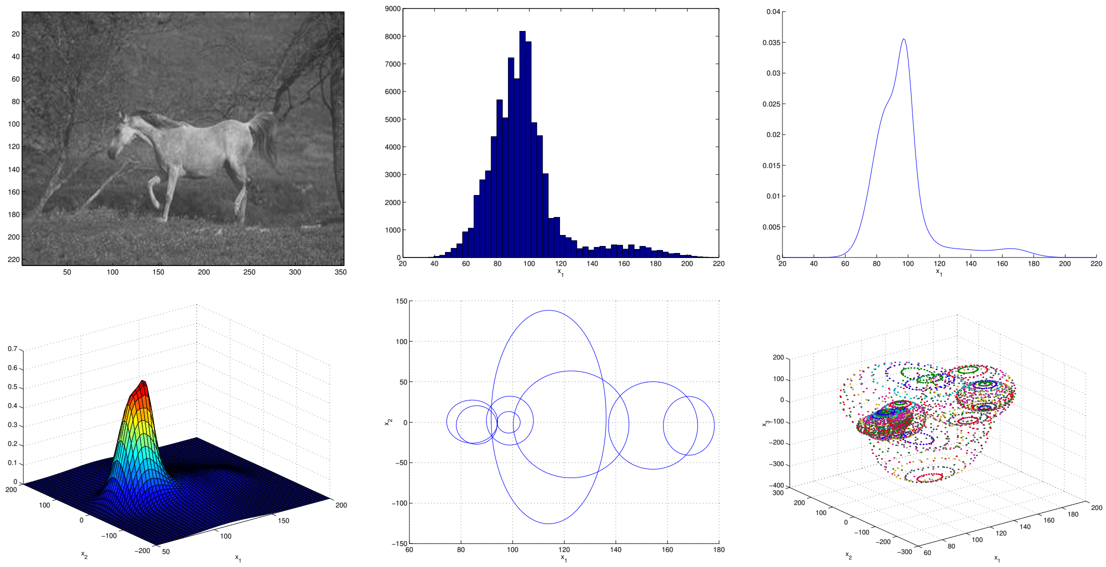


Figure 5.3: An image from the Corel database (top left), its histogram (top center), and projections of the corresponding 64-dimensional embedded mixture onto the DC subspace (top right), the subspace of the two lower frequency coefficients (density on the bottom left, contours where the likelihood drops to 60% on the bottom center), and the subspace of the three lower frequency coefficients (contours shown on bottom right).

the additional dimensions capture the spatial dependencies that are crucial for fine image discrimination (as illustrated in Figure 3.3). One of the interesting consequences of this generalization is that it enables fine control over the invariance properties of the image representation.

5.3.1 Invariance

There are many ways to encode invariance into an image representation. While the features can themselves be made invariant to common image transformations by reducing their spatial support (e.g. color histograms), filtering out high-frequency information [156, 158, 103, 166], or application of arbitrary invariant transformations [104, 163, 66], invariance can also be achieved at the level of similarity function [161, 187, 142, 159] or feature representation. In the case of representations that are learned from data, the latter can be achieved by simply including a large number of (real or artificial) examples covering all types of variation in the training set [148, 179, 115, 120, 166].

Explicitly modeling all the transformations in the similarity function usually implies a significant complexity increase in the evaluation of similarity and is not recommended in the context of CBIR. On the other hand, learning invariance does not affect retrieval complexity and is the optimal solution from the Bayes error point of view (since no information is discarded). However, the complexity of learning the individual models is usually combinatorial in the number of degrees of freedom that must be accounted for, and it may be impossible to rely on it uniquely. The best solution is, therefore, to combine learning with explicit encoding of invariance in the features.

The EMM representation provides automatic support for this combination. On one hand, by considering less or more of the subspaces, the features can be made more or less invariant to image transformations. In particular, since the histogram is approximately invariant to scaling, rotation, and translation, when only the DC subspace is considered the representation is invariant to all these transformations. As high-frequency coefficients are included, invariance is gradually sacrificed. On the other hand, invariance can always be improved by including the proper examples in the training sample used to learn the parameters of the model.

5.4 Experimental evaluation

In this section, we present results of experiments on the performance of ML retrieval with EMM models. We start by showing that this approach can outperform the standard methods for texture and color-based retrieval even in the specific domains for which these methods were designed, i.e. texture (Brodatz) and color (Columbia) databases. Improvements are shown in terms of both objective (precision/recall) and subjective evaluation. A detailed analysis of the invariance properties of the embedded mixture representation is then carried out both in terms of the trade-off between invariance and spatial support, and how invariance can be encoded in the learning process.

5.4.1 Embedded mixtures

We start by comparing EMM/ML with MRSAR/ML and HI on the Brodatz and Columbia databases. Once again, the DCT features were obtained with an 8×8 window sliding by increments of two pixels. Mixtures of 8 Gaussians were used for the Brodatz database and 16 for Columbia. Only the first 16 subspaces (DCT coefficients) were considered for retrieval. Because in the Columbia database objects are presented against a black background, we restricted Columbia queries to the central square of 1/2 the image dimensions. Diagonal covariances were used for all Gaussians, and all the mixture parameters were learned with the EM algorithm [36]. The implementations of the other techniques were the same as in the previous chapters.

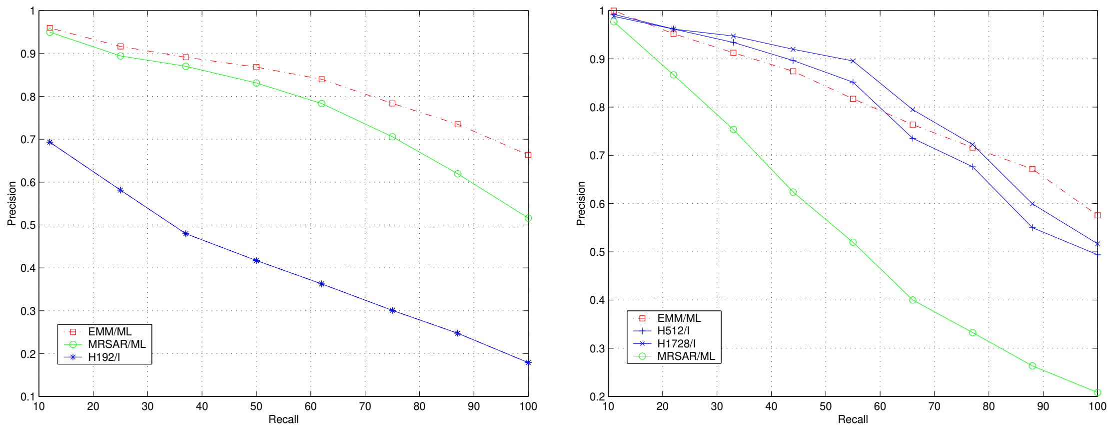


Figure 5.4: Precision/recall curves of EMM/ML, MRSAR/ML, and HI on Brodatz (left) and Columbia (right).

Figure 5.4 presents precision/recall curves obtained on the two databases. It is clear that the EMM/ML combination achieves equivalent performance or actually outperforms the best of the two other approaches in each image domain (MRSAR for texture, HI for color). This is a significant achievement because EMM is a generic representation, not specifically tailored to any of these domains, and proves that EMM/ML can handle both color and texture. The new representation should therefore do well across a large spectrum of databases.

5.4.2 Perceptual relevance

The visual observation of all the retrieved images suggests that, also along the dimension of perceptual relevance, EMM/ML clearly beats the MRSAR and histogram-based approaches. In this subsection, we present retrieval examples that are representative of the three major advantages of EMM/ML: 1) when it makes errors, these errors tend to be perceptually less annoying than those originated by the other approaches, 2) when there are several visually similar classes in the database, images from these classes tend to be retrieved together, and 3) even when the performance in terms of precision/recall is worse than that of the other approaches, the results are frequently better from a perceptual point of view.

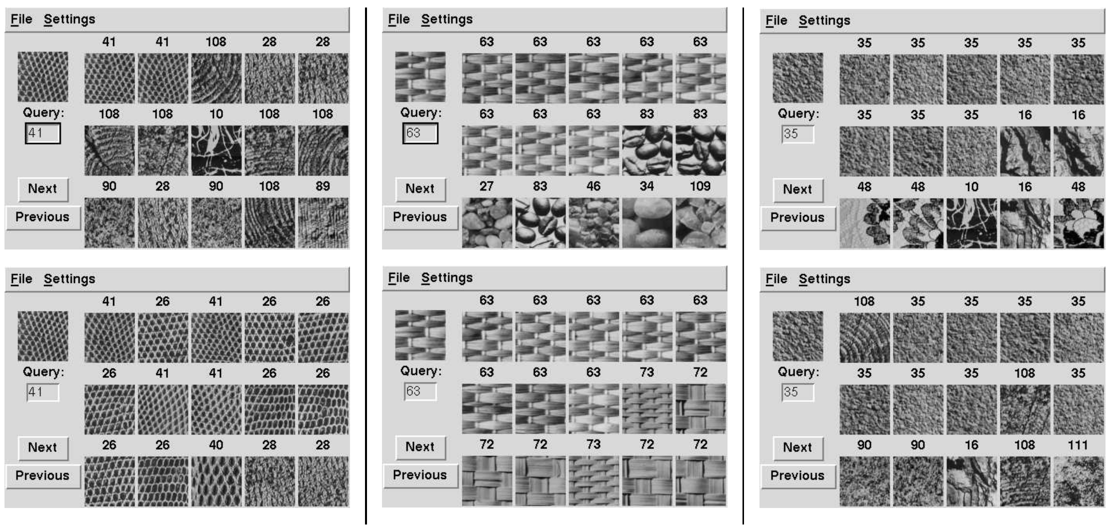


Figure 5.5: Three queries from Brodatz. MRSAR/ML results are shown at the top, EMM/ML results at the bottom. The number above each image indicates the class to which it belongs.

We start by presenting three retrieval examples from the Brodatz database. The three images in the top row of Figure 5.5 present the results of queries performed under the MRSAR/ML combination. The three images in the bottom row present the corresponding results for queries based on EMM/ML. The two images on the left illustrate how the errors of EMM/ML are usually less annoying than those of MRSAR/ML. Notice that even though EMM/ML makes several errors, most of these would actually be hard to detect by a human if the image class were not indicated on top of each retrieved image. On the other hand, the errors of MRSAR/ML are very clear.

The two images on the center are an example of situations where both approaches perform perfectly in terms of precision/recall, yet the perceptual retrieval quality is very different. MRSAR/ML ranks all the images in the query class at the top, but produces nonsensical matches after that. On the other hand, EMM/ML retrieves images that are visually similar to the query after all the images in its class are exhausted. This observation is frequent and derives from the fact that the MRSAR features have no perceptual justification.

Finally, the two images on the right are an example of how, even when better in terms of precision/recall, the performance of MRSAR/MD can actually be worse from a perceptual viewpoint. Notice how, even though it gets the first image wrong, EMM/ML produces images that are visually similar to the query. On the other hand, MRSAR/ML has perfect precision/recall but produces poor matches after all the images in the class of the query are retrieved.

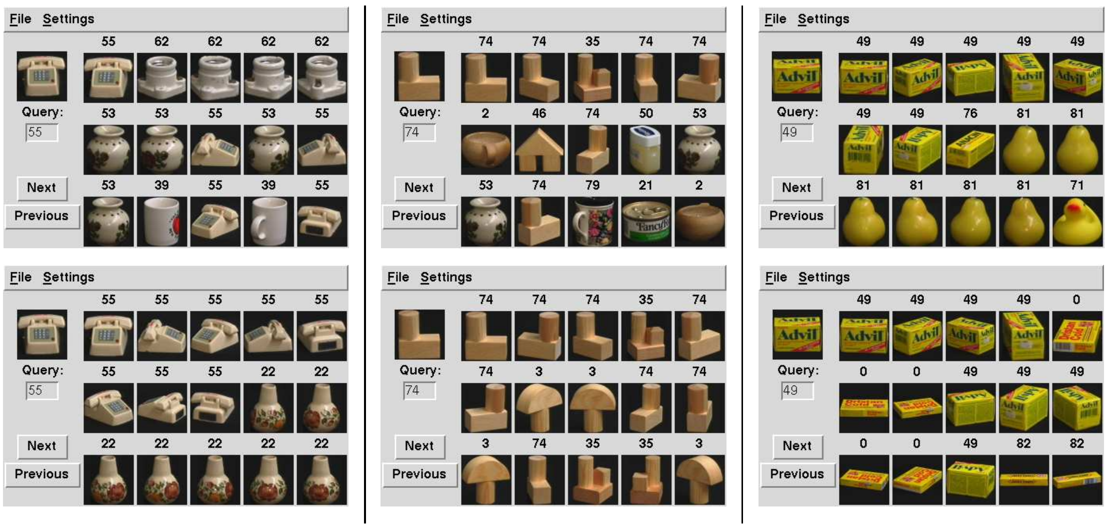


Figure 5.6: Three queries from Columbia. HI results are shown at the top, EMM/ML results at the bottom.

The Columbia database leads to similar observations with respect to the perceptual superiority of EMM/ML. Figure 5.6 presents examples comparing the performance of HI (top row) with EMM/ML (bottom one). The images on the left depict a situation in which

EMM/ML is better under both the objective and the perceptual performance criteria. This example illustrates well how color similarity is not enough for good recognition. Because there are several objects with color distributions close to that of the query, variations due to the simple rotation of the object are enough to originate poor retrieval results. In particular, the histogram-based approach does not appear to lead to a perceptually consistent retrieval strategy. Notice how light bulb sockets, vases, and coffee mugs are all returned before the desired telephone images.

On the other hand, EMM/ML not only ranks all the telephones at the top, but also consistently ranks a vase as the most similar object to the telephone query. While it is difficult to say if a person would agree with this judgment (there are no more pictures of telephones in the database), its consistency is a positive sign. The main reason why EMM/ML does better is that, by relying on features with spatial support, it is able to capture the local appearance of the object surface. It will thus tend to match surfaces with the same shape, texture, and reflection properties. This is not possible with color histograms.

The pictures on the center exemplify how capturing local appearance can lead to perceptually pleasing retrievals by EMM/ML, even when the precision/recall performance is only mediocre. In this case, while HI retrieves several objects unrelated to the query, EMM/ML only returns objects that, like the query, are made of wood blocks. Finally, the pictures on the right illustrate how, even when it has higher precision/recall, HI frequently leads to perceptually poorer results than EMM/ML. In this case, images of a pear and a duck are retrieved by HI after the images in the right class ("Advil box"), even though there are several boxes with colors similar to those of the query in the database. On the other hand, EMM/ML only retrieves boxes, although not in the best possible order.

5.4.3 Invariance vs spatial support

As discussed in section 5.3.1, EMMs provide two ways to encode invariance into the retrieval operation: learning and low-pass filtering (by discarding high-frequency subspaces). We now carry out a quantitative analysis of these two mechanisms.

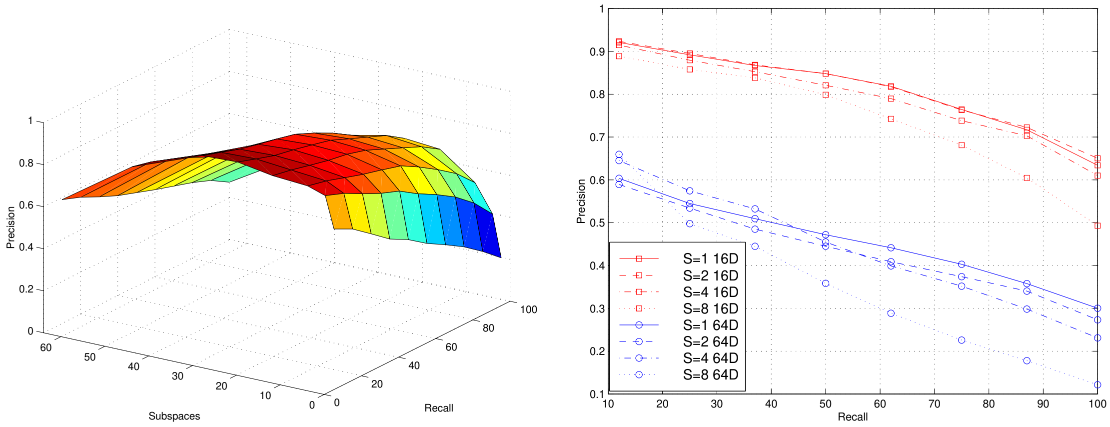


Figure 5.7: Analysis of invariance on Brodatz. Left: surface spanned by precision/recall as a function of the number of subspaces considered during retrieval. Right: precision/recall as a function of block spacing (S) during learning when 16 and 64 subspaces are considered.

Figure 5.7 presents 1) the surface spanned by precision/recall as a function of the number of subspaces considered during retrieval, and 2) the impact on precision/recall of the spatial distance between adjacent features vectors used for learning. The precision/recall surface clearly illustrates the trade-off between invariance and spatial support. When too few subspaces are considered, the spatial support is not enough to capture the correlations that characterize each texture class. Performance increases as the number of subspaces grows, but starts to degrade as high frequencies are included. At this point, because the representation is much more detailed, good recognition requires precise alignment between the query and the database feature vectors. Hence, the recognition becomes very sensitive to small spatial deformations between the textures.

The plot on the right confirms that increasing the number of examples in the training set also improves the retrieval accuracy. When the image blocks are non-overlapping, the representation is invariant only to translations by multiples of the block size. As the space between samples decreases, invariance happens for smaller displacements and retrieval accuracy increases. Notice that, while this is true with both small and large number of subspaces, the relative gain is much higher in the latter case. This was expected since invariance is a bigger problem when high frequencies are included.

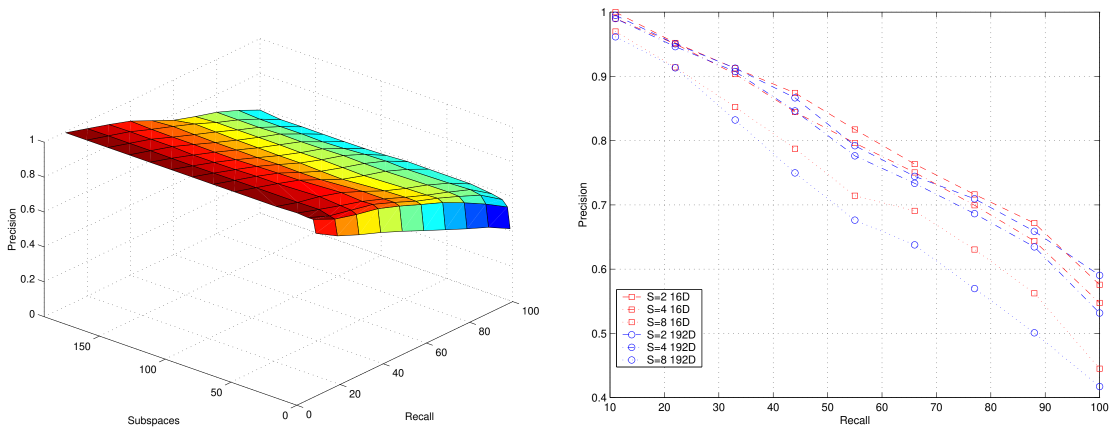


Figure 5.8: Analysis of invariance on Columbia. Left: precision/recall surface. Right: precision/recall as a function of block spacing (S) during learning when 16 and 192 subspaces are considered.

Figure 5.8 presents similar plots for Columbia. While the general behavior is the same, there are interesting differences in the details. In particular, precision/recall increases much faster with the number of subspaces and remains approximately constant after that. This indicates that the intrinsic dimensionality of the images in this database is significantly smaller than the textures in Brodatz and explains why approaches based on low-dimensional feature spaces can perform well for object recognition [172, 156, 103, 158]. This intrinsic low-dimensionality is a consequence of a much smaller high-frequency content and responsible for the flatness of the precision/recall surface: since there is no high-frequency energy, adding or deleting high-frequency subspaces does not have a significant impact on the retrieval accuracy.

On the other hand, as seen by the plot on the right, the spacing between training samples still has a significant impact on the retrieval performance, independently of the number of subspaces considered. We should note here that, while there is a significant amount of variation in scale and rotation on the Columbia database, the results above were obtained without explicit encoding of invariance to these transformations. This is encouraging since one would expect the inclusion of rotated and scaled copies of the feature vectors in the learning sample to further reduce the slope of the precision-recall curve. The main limitation of this approach is the inherent increase in the computational complexity of the learning and, for this reason, we have not conducted any experiments along these lines.

Overall, the results above lead to two main conclusions regarding invariance. First, there seems to be a relatively large set of subspaces for which the representation achieves a good trade-off between spatial support and invariance. In particular, any number between 16 and 32 subspaces per color channel seems to work well. This implies that a precise selection of the number of subspaces is, in practice, not crucial for good performance. Second, while the interval between consecutive samples in the training set clearly affects the performance, there seems to be no need to consider all possible image positions. In fact, while it is important to use a sampling grid where there is overlap between spatially neighboring samples, a spacing of half the block size already achieves performance equivalent to the best results.

5.5 Discussion

While we have shown that mixture models generalize most of the feature representations that have been proposed in the retrieval literature, we do not claim that they are the definitive answer to the problem of feature representation. The main limitation of the EMM image representation is that it can only model short-term image dependencies. In particular, it has no ability to model the dependencies that occur between DCT coefficients of spatially adjacent image neighborhoods.

One obvious extension to account for these dependencies would be to rely on *Markov Random Fields* (MRFs) [25]. These are models that define a probability measure over the entire image lattice by imposing a Markov condition: given the observation of its neighbors, each pixel is independent of the remainder of the image. The complexity of the MRF model depends on the size of the neighborhood underlying this Markov condition and, in practice, only very small neighborhoods are computationally feasible. This limits the ability of MRFs to capture long range structure and the success of these models has been, for the most part, limited to modeling random texture [33, 178, 37]. Since, in this case, correlation decreases quickly with distance, it is not clear that an MRF would provide vast improvements over EMM.

Recently, however, Zhu et al. [151] have shown that it is possible to design MRFs that can account for long-range and periodic structures. Given a multi-resolution decomposition of an image, they compute histograms on each of the frequency subbands and search for the distribution whose marginal densities matches those histograms. This search is formulated as a *maximum entropy* problem which leads to a MRF solution. While their results were impressive, the technique is extraordinarily slow and infeasible in the context of image databases. Nevertheless, it has prompted great interest on the question of modeling the statistics of multi-resolution representations [135, 138, 14].

While it has not yet been shown that it is possible to derive a true probabilistic model, or a suitable approximation, capable of accounting for long-range *spatial* dependencies between frequency coefficients and having tractable complexity, the question of whether such a model exists remains open for further debate.