VQ based Image Retrieval using Color and Position Features

Ajay H. Daptardar*and James A. Storer
Computer Science Department, Brandeis University,
Waltham, MA

Abstract

We present a new lower complexity approach for content based image retrieval based on a relative compressibility similarity measure using VQ codebooks employing feature vectors based on color and position. In previous work we have developed a system that employs feature vectors that are a combination of color and position. In this paper, we present a new approach that decouples color and position. We present this approach as two methods. The first trains separate codebooks for color and position features, eliminating the need for potentially application specific feature weightings during training. The second method achieves nearly the same performance at greatly reduced complexity by partitioning images into regions and training high-rate TSVQ codebooks for each region (i.e., position information is made implicit). Features extracted from query regions are encoded with the corresponding database region codebooks. The maximum number of codewords that a database region codebook may contain is determined at runtime and is a function of the query features. Region codebooks are then pruned appropriately before encoding query features. Experiments performed on the COREL image database show this new approach to provide almost equivalent retrieval precision to our previous method of jointly trained codebooks (and an improvement over previous methods) at much lower complexity.

1 Background

With the recent proliferation of digital images, there is a need for information systems that can organize and store images using models which support content based queries. In the query-by-example setting (Eakins and Graham [4]), a user presents the system with a query image and the system responds by retrieving a set of database images with (visually) similar content. Given the discriminative power of color features and the simple histogram model, global color histograms which are relatively invariant to spatial transformations such as translation and rotation, have been effectively used

*Contact: ajay@brandeis.edu

for CBIR related tasks (Swain and Ballard [10], Faloutsos et al. [5]) However, global color histograms, because of their sensitivity to bin width and bin placement, have a drawback: if images with different statistical properties (classes) are later added to the database, the histogram colors (labels) may need to be recomputed to maintain retrieval precision. When dealing with varying databases (the web, for example), it can be advantageous to use models generated individually for each image.

Vasconcelos [12] uses finite Gaussian mixture models (GMM) to represent image densities and maximum-likelihood (ML) classifiers for retrieval, i.e the most similar image (class) is the one that maximizes the posterior probability of the database image (class) given the query image. Image similarity is computed using an approximation, the asymptotic likelihood approximation (ALA), of the Kullback-Liebler divergence between two Gaussian mixtures. Jeong and Gray [8] introduced minimum distortion image retrieval (MDIR) using GMMs where, instead of comparing image densities, the total query distortion is computed by encoding the query features with database GMM. Database images are ranked based on this distortion, and experiments show that MDIR outperforms ALA in terms of retrieval precision, albeit at higher complexity. We introduced a similar approach using VQ codebooks and simple mean squared error (MSE) distortion [1]. Images are ranked based on the MSE when query features are encoded with database image codebooks. This can be viewed as the prototype 1-nearest neighbor (1-NN) rule [3] in an unsupervised learning setting where instead of assigning a label to a query vector, a MSE score is assigned. Experiments measuring retrieval precision show that for the MDIR similarity criteria, VQ models compare well with GMMs while operating at a lower complexity [2]. We also implemented a simple extension of this method which considers position information. Position features consisting of the XY coordinates of an image block are appended to color features for that block and a VQ codebook is trained. Suitable weightings for the position features are determined empirically and it was found that by using position features for retrieval, there was an overall modest improvement in retrieval precision for a minor increase in complexity (feature vector dimension increases by 2), while at the same time providing "insurance" where it makes little difference for many classes but can yield significant improvements for some.

In this paper, we present a new approach that decouples color and position information. The first method trains separate codebooks for color and position features, eliminating the need for specifying feature weighting during training and allowing for potentially application specific feature weightings during retrieval. The second method achieves nearly the same performance at greatly reduced complexity by partitioning images into regions and training high-rate TSVQ codebooks for each region. Global similarity is defined as the sum of the similarities for each region. However, instead of directly encoding query features with the database codebook for the corresponding region, we apply a preprocessing step which limits the maximum number of codewords contained in a particular region codebook during encoding. This method can be viewed as related to the work by Stricker and Dimai [9] where images were represented by fuzzy, partially overlapping regions. Feature vectors consisting of the first three color moments in the HSV color space are extracted from each region and retrieval is performed by computing the sum of weighted differences between color feature moments. Vaisey and Gersho employed segmentation based image compression where each region was associated with a class and a distinct coding procedure was used for each class [11].

2 VQ based Image Similarity

Vector quantization (VQ) is a well known technique for signal compression [6]. Given a set of training vectors, a training algorithm determines a set of vectors (codewords) that constitute a codebook. Compression is achieved in representing a source vector by transmitting or storing the index of the codeword that it is closest to. For our purposes, VQ codebooks for each database image are trained using Lloyd clustering, and similarity is determined by how well a given database codebook encodes query image feature vectors. Specifically, given a query image A and a database image B, the system computes a score which is defined as the mean squared error when the features extracted from image A are encoded with image B's codebook:

$$s(A,B) = \frac{1}{N}\sum_{i=1}^{N} \operatorname*{argmin}_{j} \|(a_i - b_j)\|_{\mathbf{W}}, \tag{1}$$

where $a_i, b_j \in \mathbb{R}^k$, $\{b_j\}_{j=1}^{M}$ is the codebook for image B, $\{a_i\}_{i=1}^{N}$ are the feature vectors from query image A, $\mathbf{W} \in \mathbb{R}^{k\times k}$ is a weight matrix and $\|x-y\|_{\mathbf{W}} \overset{\text{def}}{=} (x-y)^t\mathbf{W}(x-y)$.

2.1 Separate Training with Color-Position Codebooks

In our previous work [2], we extended the basic VQ approach to incorporate position information. Position features consisting of the XY coordinates of an image block are appended to color features extracted from that block. Full search VQ codebooks are then trained on these joint features. The key parameters: feature weighting and codebook size were determined empirically. Although experimentally, this method improved retrieval precision, it is not clear if joint training of disparate features uses the discrimination provided by each feature type optimally. Also, such feature feature weighting may be data dependent. Therefore, in order to avoid the feature weighting problem during the training phase, separate codebooks are trained for color and position features; that is, we can assign a set of position codewords to each color codeword. We begin by training a color codebook by considering only the color features of feature vectors in the training set. Given a color codeword, we then train a position codebook by considering only the position features of all feature vectors that map, based on their color features, to this color codeword. Thus, associated with every color codeword, there is a set of position codewords. When encoding a given a query vector, the best color position match is the one that jointly minimizes the weighted color and position distortion:

$$s(A,B) = \frac{1}{N}\sum_{i=1}^{N} \operatorname*{argmin}_{j,k} \left\{ \|(a_i^{(c)} - b_j^{(p)})\|_{\mathbf{W}^{(c)}} + \|(a_i^{(p)} - b_{jk}^{(p)})\|_{\mathbf{W}^{(p)}} \right\}, \tag{2}$$

where $a_i^{(c)}$ and $a_i^{(p)}$ are the color and position features (sub-vectors) respectively for feature vector a, $\mathbf{W}^{(c)}$ and $\mathbf{W}^{(p)}$ are the color and position weights respectively and $k \in \{1, \ldots M_j^{(p)}\}$ with $M_j^{(p)}$ the number of position codewords associated with color j. The double index $b_{jk}^{(p)}$ refers to the k-th position codeword for color codeword j. In terms of complexity, this method is cheaper than having an unstructured VQ codebook of size $M_1 M_2$ with M_1 colors and M_2 positions per color because the color distortion is fixed for different positions associated with a given color codeword.

2.2 Region VQ

Our second method partitions images into regions and trains high-rate TSVQ codebooks for these regions. Within a given region, spatial color distribution is ignored by training codebooks only on color features. Codebooks are trained by successively splitting leaf nodes with the largest distortion until the desired number of leaf codewords is reached. The global score is computed by summing the individual scores for each region:

$$s(A, B) = \sum_{i=1}^{r} \frac{w_i}{|R_i|} \sum_{j=1}^{|R_i|} \underset{k}{\operatorname{argmin}} \|a_{ij} - b_{ik}\|, \tag{3}$$

where r is the number of regions in an image, $|R_i|$ is the number of feature vectors in region R_i, $\{b_{ik}\}_{k=1}^{M_i}$ is image B's (color only) codebook of size M_i for region R_i, $\{a_{ij}\}_{j=1}^{|R_i|}$ is the set of query features from image A's region R_i, and w_i is the weight to be assigned to region R_i. Before encoding query features, region codebooks are pruned depending upon the query and region. A single parameter, *query threshold*, is used to determine the size to which a region codebook must be pruned. Given a query image and a query threshold, TSVQ codebooks are trained for each query image region such so that the MSE for each region is less than the query threshold. The sizes of the resulting region codebooks determine the sizes for the corresponding pruned database region codebooks.

The following steps summarize the querying process for an arbitrary region R with query features $A = \{a_i\}_{i=1}^{|R|}$ and database codebook $B = \{b_j\}_{j=1}^{M}$.

- Train a region codebook of size M' using the query feature set A, such that the MSE is less than the query threshold.
- Prune the database image's region codebook B to size M'. The pruning algorithm merges siblings u and v with parent w such that $D(w) - (D(u) + D(v))$ is minimal amongst all siblings. Where $D(n)$ is the (training) distortion at node n and is stored in the codebook.
- Encode query features A with the pruned region codebook $B' = \{b_j\}_{j=1}^{M'}$ using full search as defined in Equation 3.

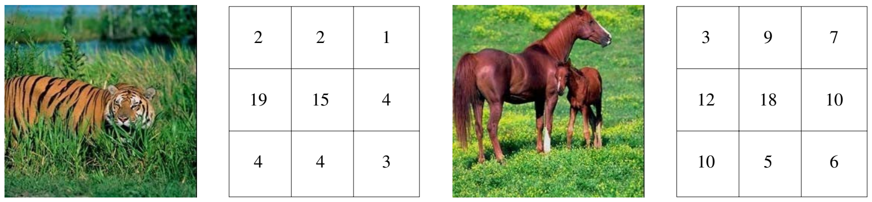

Figure 1: Sample database images for Region VQ (Tigers no. 24 and Horses no. 77); the figure to the right of each image shows the region boundaries and the prune sizes for each region corresponding to a query threshold of 1500.

2.3 MDIR using GMM

We compare the two methods presented here with the minimum distortion image retrieval (MDIR) method of Jeong and Gray. MDIR uses finite Gaussian mixture models (GMM) to represent database image densities, and determines similarity based on the distortion when query image features are encoded with database GMMs:

$$s(A,B) = \sum_{i=1}^{N} \operatorname*{argmin}_{j} \rho(a_i, \pi_j, g_j) \tag{4}$$

where $\{a_i\}_{i=1}^{N}$ are the query features, $\{\pi_j, g_j = \mathcal{N}(\mu_j, \Sigma_j)\}_{j=1}^{M}$ is a finite Gaussian mixture model with M components with priors π_j and component densities which are multivariate Gaussian with means μ_j and covariance matrices Σ_j. Database GMMs are trained using Gauss mixture VQ clustering [7]. The cost function, $\rho(a_i, \pi_j, g_j) = (d_{\mathrm{LL}}(a_i, g_j) - \ln \pi_j)$ is the penalized log-likelihood, where

$$d_{\mathrm{LL}}(a_i, g_j) = \frac{1}{2}\left(k \ln(2\pi) + \ln\det(\Sigma_j) + \|a_i - \mu_j\|_{\Sigma_j^{-1}}\right).$$

3 Experiments

For experiments, we used a subset of the COREL database (see Wang et al. [13], Jeong and Gray [8]), consisting of 1500 JPEG images, organized into 15 classes of 100 images each. Database images are either 256×384 or 384×256 in size. For convenience, the images were cropped to a central 256×256 region and then scaled down to 128×128.

3.1 Image features

We employed relatively simple features for experiments. Database images were transformed from the RGB to the nonlinear CIE-LUV color space in which the Euclidean distance between color vectors corresponds more closely with human perception (Wyszecki and Stiles [14]). Features are extracted by sliding a window across the image in a raster tiling fashion. Each feature vector comprises of the mean and

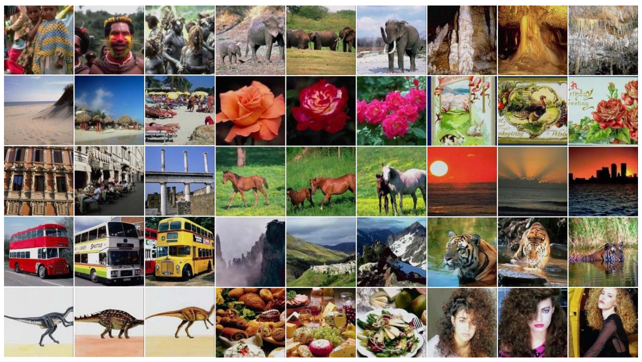

Figure 2: Samples database images from each class in raster order: *Africans, Elephants, Caves, Beach, Roses, Postcards, Architecture, Horses, Sunsets, Buses, Mountains, Tigers, Dinosaurs, Foods, Women.*

the variance for each color channel within that block along with its XY-coordinates. The mean and variance for each color channel are computed as:

$$\begin{aligned} \mu_c &= \tfrac{1}{4}\sum_{i=0}^{1}\sum_{j=0}^{1} p_{ij}^{(c)}, \\ \sigma 2_c &= \tfrac{1}{4}\sum_{i=0}^{1}\sum_{j=0}^{1} \left(p_{ij}^{(c)} - \mu_c\right)^2, \end{aligned}$$

where p_{ij} is the pixel value at row i, column j in the 2×2 window for color channel $c \in \{L, U, V\}$. The result is an 8-dimensional feature vector:

$$(\mu_L, \mu_U, \mu_V, \sigma 2_L, \sigma 2_U, \sigma 2_V, x, y)^t,$$

where the first six components are the color features and the last two are the position features.

3.2 Results

The same set of 210 query images as used previously in [8] and [2] are used in all experiments reported here. Standard precision recall curves were used as performance metrics. Precision and recall are shown on a single graph to display the change in precision as the recall increases. Since the precision typically drops as the recall increases, a retrieval system is said to be more effective if it has higher precision at the same recall values.

To test our first method using color-position codebooks, we trained codebooks with 8 colors and 8 positions per color. Figure 3 shows the precision recall plots of color-position codebooks with our previous work of jointly trained codebooks with 22 codewords (for comparable complexity - see section 3.3) and MDIR based on 8 component Gaussian mixtures (this was chosen so that results from [8] could be directly used). Although, retrieval precision for color-position codebooks closely follows that for jointly trained codebooks, separate training of color and position features does not require feature weightings during the training phase. In addition, the number

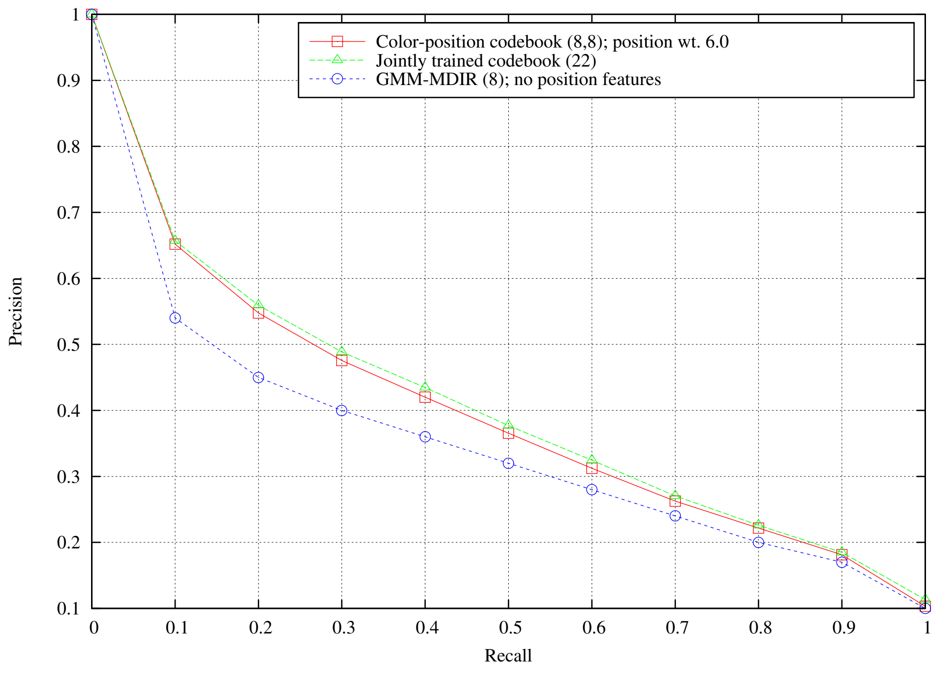

Figure 3: Comparing color-position codebooks with 8 colors and 8 positons per color, jointly trained codebooks with 22 codewords and MDIR based on GMMs with 8 components (higher is better).

of positions for a given color may also be varied depending on the image statistics. The performance of GMM-MDIR is expected since the model does not contain any position information.

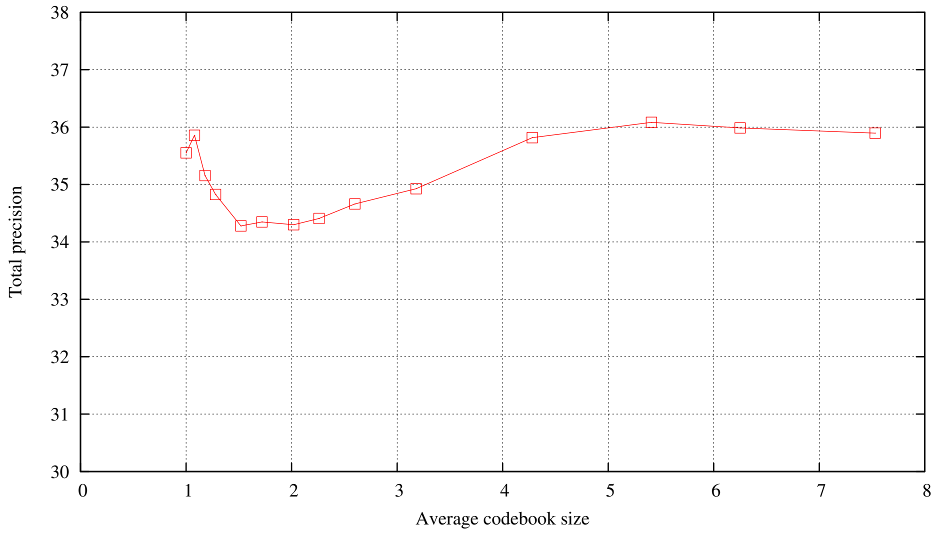

Figure 4: Total precision vs. Average codebook size for region VQ

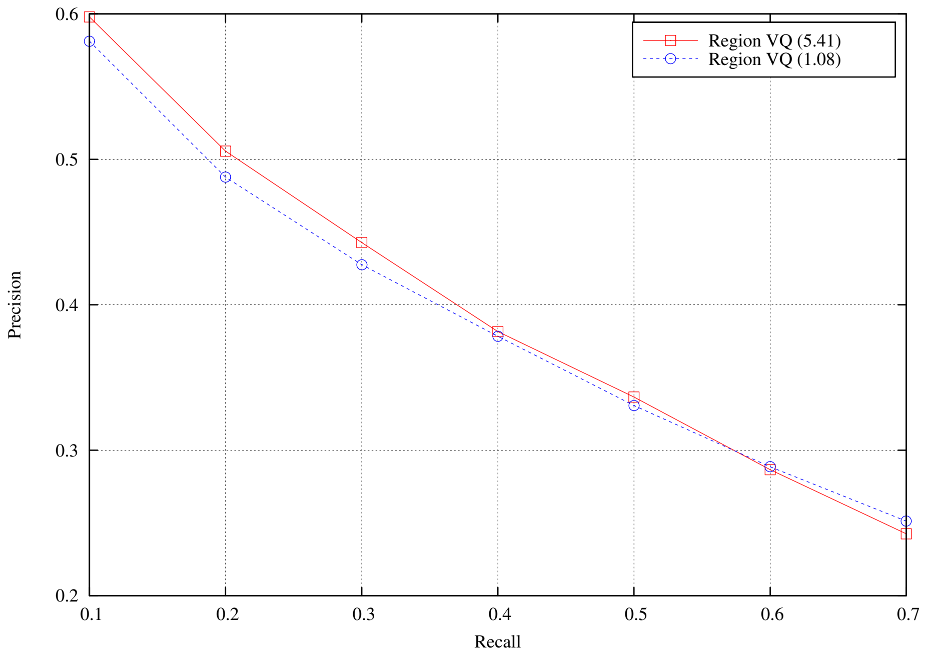

Figure 5: Precision vs. Recall for region VQ; numbers in parenthesis are the average codebook sizes

For region VQ experiments, we partitioned each database image into 9 regions where the 4 corner regions were of size 43×43 (42×42 for the remaining regions). For each region, a TSVQ codebook was trained with 32 leaf codewords and tests were performed at various query thresholds. Figure 4, plots the *total precision*, defined as the precision sum at all recall values ($P_t = \sum_{r=1}^{100} P_r$ where P_r is the precision at recall r), against the average region codebook size, which is computed as the average codebook size for all 210 queries for all 9 regions. The dip in the graph, although visually quite significant at this scale, amounts to a 5% drop in total precision. Figure 5 shows the precision recall performance of region VQ. The best performance occurs at an average codebook size of 5.41 (corresponding to a query threshold of 1500). Also region VQ with average codebook size of 1.08, has comparable retrieval precision (within 3%).

3.3 Complexity

A key advantage of the methods presented here is reduced complexity at comparable retrieval precision. As an example, consider the number of multiplications for an elementary step that searches for the best codeword in a codebook. A color-position codebook with 8 colors and 8 positions per color, uses $8(6 + 2 \times 8) = 176$ multiplications since for each color codeword, we need 6 multiplications to compute the color distance and 2×8 multiplications to find the best position for this color codeword. The equivalent jointly trained codebook would have $176/8 = 22$ codewords. The best performing region VQ (with an average codebook size of 5.41) uses

$5.41 \times 6 = 32.46$ multiplications since once a feature vector is assigned to a region, we need only compute the best matching color codeword. Recall that region VQ with an average codebook size of 1.08 achieves nearly the same precision (within 3%); this reduces the number of multiplications to only $1.08 \times 6 = 6.48$. In contrast, MDIR with 8 components per Gaussian mixture and full covariances uses one vector-matrix multiplication (8×8) and one inner product (8) for each of the 8 Gaussians for a total of $8(8 \times 8 + 8) = 576$ multiplications, nearly two orders of magnitude greater complexity.

4 Conclusion and Future Work

We have presented alternatives for incorporating position features into the VQ based image retrieval framework. The first method with separately trained codebooks alleviates the problem of feature weightings during the training phase and performs comparably with our previous jointly trained codebook method. The region VQ method at slightly reduced retrieval precision offers a significantly lower complexity alternative. Figure 6 summarizes this performance by showing the best points for the two methods presented here, our previous work with jointly trained codebooks and MDIR with GMMs. Further investigation in terms of region weighting, region size, region shape and soft partitioning of images is of interest.

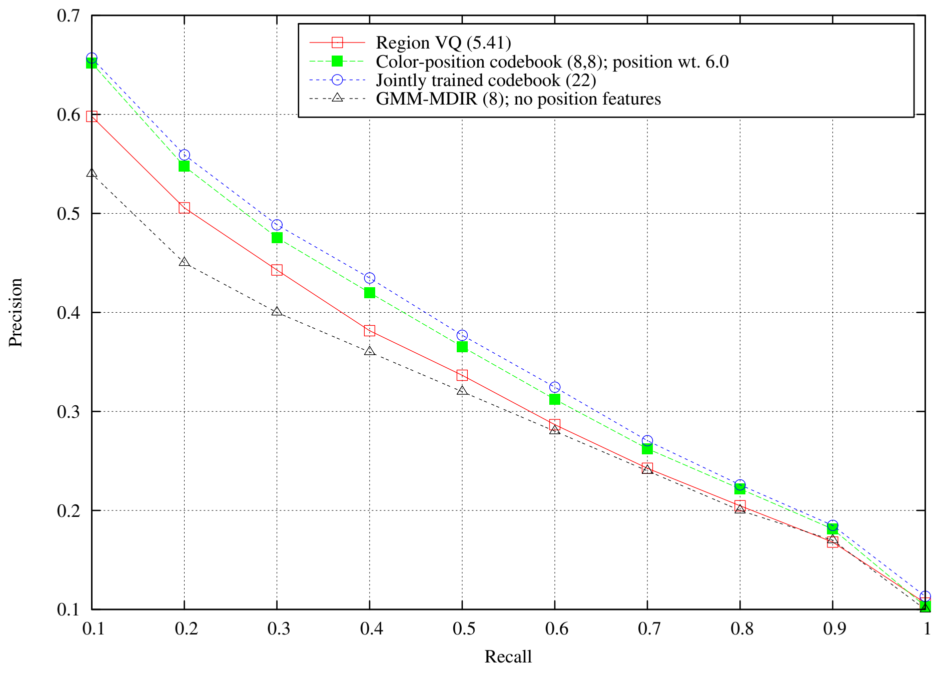

Figure 6: Precision vs. Recall plots (higher is better) comparing region VQ with an average of 5.41 codewords per region, color-position codebooks with 8 colors and 8 positions per color, jointly trained codebooks with 22 codewords and MDIR based on GMM with 8 components (no position features).

References

[1] Ajay H. Daptardar and James A. Storer. Content-based image retrieval via vector quantization. In *ISVC*, volume 3804 of *Lecture Notes in Computer Science*, pages 502–509. Springer, 2005.

[2] Ajay H. Daptardar and James A. Storer. Reduced complexity content-based image retrieval using vector quantization. In *Data Compression Conference*, pages 342–351. IEEE Computer Society, 2006.

[3] Luc Devroye, László Györfi, and Gábor Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

[4] J. P. Eakins and M. E. Graham. Content-based image retrieval. Technical report, JISC Technology Applications Programme, 1999.

[5] Christos Faloutsos, Ron Barber, Myron Flickner, Jim Hafner, Wayne Niblack, Dragutin Petkovic, and William Equitz. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3(3-4):231–262, 1994.

[6] Allen Gersho and Robert M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1992.

[7] Robert M. Gray. Gauss mixture vector quantization. In *Proceedings of IEEE ICASSP*, volume 3, pages 1769–1772, 2001.

[8] Sangoh Jeong and Robert M. Gray. Minimum distortion color image retrieval based on Lloyd-clustered Gauss mixtures. In *Data Compression Conference*, pages 279–288. IEEE Computer Society, 2005.

[9] Markus Stricker and Alexander Dimai. Color indexing with weak spatial constraints. In *Storage and Retrieval for Image and Video Databases IV*, volume 2670, February 1996.

[10] Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[11] J. Vaisey and Allen Gersho. Image compression with variable block size segmentation. *IEEE Transactions on Signal Processing*, 40(8):2040–2060, 1992.

[12] Nuno Vasconcelos. Minimum probability of error image retrieval. *IEEE Transactions on Signal Processing*, 52(8):2322–2336, 2004.

[13] James. Z. Wang, Jia Li, and Gio Wiederhold. Simplicity: Semantics-sensitive integrated matching for picture libraries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(9):947–963, 2001.

[14] Günther Wyszecki and W.S. Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Wiley-Interscience, 2 edition, 2000.