CONTRIBUTIONS TO CONTENT-BASED IMAGE RETRIEVAL

by

Hua Xie

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(ELECTRICAL ENGINEERING)

August 2005

Dedication

To my family.

Acknowledgments

I would like to express my deepest gratitude and thanks to my advisor, Dr. Antonio Ortega, for his guidance, inspiration, support and patience throughout the years I have been pursuing my Ph.D. degree at the University of Southern California. He has made my experience at USC greatly rewardable and memorable.

I would like to extend my gratitude to Prof. C.C. Jay Kuo and Prof. Cyrus Shahabi for serving on my Dissertation Committee, and Prof. Shrikanth S. Narayanan and Prof. Zhen Zhang for serving on my Qualifying Exam Committee.

I would like to thank all my friends in the lab, especially Raghavendra Singh, Zhourong Miao, Naveen Srinivasamurthy, Sang-Yong Lee, Phoom Sagetong, Hui-Sheng Wang, Hyukjune Chung and Jae Hoon Kim, for making it a pleasant work environment. I would like to thank Baltasar Beferull-Lozano for a very enjoyable collaboration.

Last, but not least, I thank my family for their consistent support through these years. I thank my parents for their unconditional love and letting me to pursue my dreams; I thank my parents-in-law for taking care of my daughter so that I had time to finish my work during my last year at school; I thank my daughter for bringing

so much joy to my life. I especially thank my husband, Jing Cao. Without his companionship, understanding, encouragement and support, it would be impossible for me to be who I am today.

This research has been funded in part by NASA under grant AIST-0122-0005 and by Integrated Media Systems Center, a National Science Foundation Engineering Research Center.

Contents

List of Tables

List of Figures

Abstract

Due to the proliferation of multimedia information over the Internet, users are confronted with large amounts of content from many sources around the world. Content-based retrieval system have been proposed to automatically annotate and index multimedia information based on their audio/visual contents instead of manually-entered text keywords. In this thesis, we investigate two major topics related to content-based retrieval.

First, we propose and analyze efficient compression techniques for distributed image retrieval systems. The first technique we design is a classified quantization system. A partial classification is first performed before compressing the data so that we are able to capture the special characteristics of the classes that are relevant to content-based retrieval. The pre-classifier and the quantization parameters for each class are jointly searched based on a rate-distortion-complexity optimization framework. Substantial improvement in terms of retrieval performance vs. bit rate, is achieved using the proposed compression scheme as compared to standard encoding.

The second technique we consider is to use linear discriminant analysis for transform coding in distributed image classification/retrieval systems. We examine the optimal transform which compacts the class discrimination information into the lowest dimensional space, and propose a greedy bit allocation algorithm to minimize the loss in class separability due to quantization. We analyze the relations between proposed transform coding and Likelihood Ratio Quantization, and develop high rate analysis for certain classes of Gaussian distributions.

The second topic addresses relevance feedback, a critical component for content-based retrieval systems. It has been shown that support vector machines (SVMs) can be used to conduct effective relevance feedback. In this work, we propose an approach to derive a novel information divergence based kernel given the user's preference. Our proposed kernel function naturally takes into account the statistics of the data that is available during relevance feedback. Experiments show that the new kernel achieves significantly higher (about 17%) retrieval accuracy than the standard radial basis function (RBF) kernel, and can thus become a valid alternative to traditional kernels for SVM-based active learning in relevance feedback applications.

Chapter 1

Introduction

1.1 Motivation

With the recent advances in networking technologies and devices for producing and storing multimedia data (image/video/audio), we have seen fast proliferation of multimedia information over the Internet during the past few years. Digital images and videos have become an integral part of human communications. Consider a few example applications:

- *Digital libraries.* Draw a sketch of what you have in your mind and find a set of images which contain similar contents. Or play a few notes and retrieve pieces of musics similar to the required tune.
- *Home entertainment.* Home video editing, systems for managing personal multimedia collections.
- *E-Commerce.* Personal advertising, directories of e-shops.

Now the question is: given these very large amounts of audio/visual information distributed all over the world, how can we *efficiently* locate and retrieve that information we are specifically interested in?

Content-based Information Retrieval (CBIR) systems are proposed for automatically indexing and accessing large amounts of information. In such systems, multiple features (color, texture, shape, etc.) are extracted from the multimedia data as a summarization of the information contained in the data. Retrieval is performed based on some similarity matching function, where given an input feature pattern the goal is to search for similar patterns in the database. Some well known CBIR systems include QBIC [45], MARS [42], Netra [41] and Photobook [50]. Many advances have been made in visual feature extraction [70],[73],[69], multi-dimensional indexing [4],[24],[13], etc. However, there are still many open research issues which need to be solved before content-based image retrieval can be put into practical use. In this thesis, we investigate two major topics related to content-based retrieval system. While we focus on image retrieval systems in our examples and experiments, the ideas and concepts can be easily applied to other media types.

First we investigate the limitations of having a centralized system architecture, where feature extraction, indexing and query processing are all done at a central database server. This architecture may require significant computation at the central node and may make it difficult to scale up the system. We argue that one approach to overcome this limitation is to design a distributed retrieval system, where the

data storage and query computation are shared by users over the network. Users search and exchange information by sharing with each other over the network the relevant content features, which contain sufficient information for retrieval. We propose that by compressing the features we are able to reduce both the transmission bandwidth and the storage space significantly, without degrading retrieval accuracy. Different from traditional compression techniques, which are designed to provide the best perceptual quality under given rate constraints, we design novel compression techniques tailored for specific classification purposes.

The second topic addresses relevance feedback, a critical component for content-based retrieval systems. Effective learning algorithms are needed to accurately and quickly capture the user's query concept, under the daunting challenges of high dimensional data and small number of training samples. It has been shown that support vector machines (SVMs) can be used to conduct effective relevance feedback in content-based image retrieval. Most recent work along these lines has focused on how to customize SVM classification for the particular problem of interest. However, not much attention has been to paid to the design of kernel functions specifically tailored for relevance feedback problems and traditional kernels have been directly used in these applications. In this work, we propose an approach to derive an information divergence based kernel given the user's preference. Our proposed kernel function naturally takes into account the statistics of the data that is available during relevance feedback for the purpose of discriminating between relevant and

non-relevant images. Experiments show that the new kernel achieves significantly higher (about 17%) retrieval accuracy than the standard radial basis function (RBF) kernel, and can thus become a valid alternative to traditional kernels for SVM-based active learning in relevance feedback applications.

1.2 Data compression for distributed image retrieval/classification

Most existing content-based retrieval systems are focused on signal processing techniques to extract meaningful features, and indexing methods to speed up the database search. These systems are developed in a centralized fashion as stand-alone applications where feature extraction, indexing and query processing are all done at the database server.

But there are many real applications which involve remote access to multimedia databases in order to retrieve useful visual information in the form of photographs, scanned articles, satellite images, etc. The remote access to such databases can be initiated by a client machine forming a query message, and sending this query information to the remote database server where the query is processed and the results are sent back to the clients.

Figure 1.1 shows the block diagram of a content-based image retrieval system based on a Client-Server architecture, where the client and the server are physically apart. The client may have access to a sample image and communicates with the remote database server looking for similar images (objects). The server keeps a collection of images (image database) and their corresponding features (feature database). Similarity matching is performed between the query feature and features in the database in order to retrieve similar images.

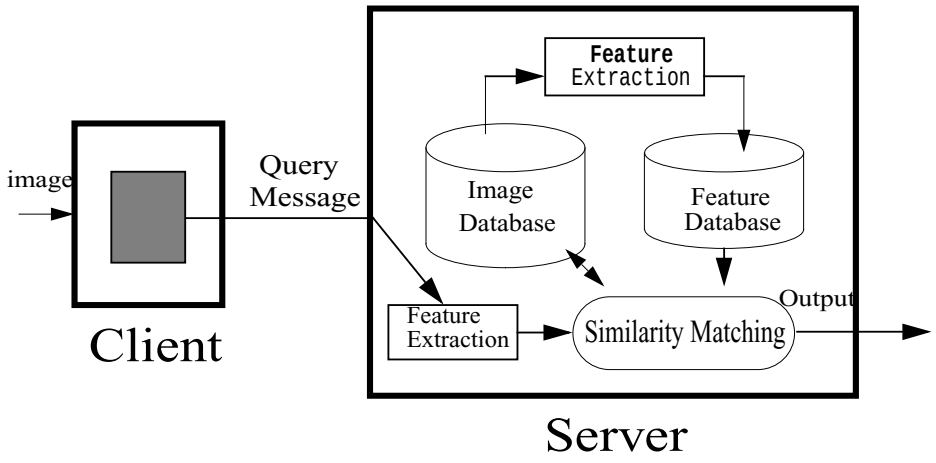


Figure 1.1: Block diagram of content-based Image Retrieval System. The image features are extracted to represent the content information and stored in a database of features. Similarity is measured by computing a distance metric between the feature vectors of query image and those of database images.

As compared to text-based indexing, content-based image querying involves more computation and leads to different design problems. In this thesis, we consider content-based image retrieval over:

- *Web-server based systems.* The images are located at each of the database servers. Clients with different processing power and connecting bandwidth access the database through a central web-server which processes the query and forwards it to the target databases.

- *Peer-to-Peer networks.* Each peer in the network stores a subset of the database and different files are shared by different peers. When a peer initiates a search, it broadcasts the query request to all its connecting peers. Each peer receiving the request will then process it and propagate it on to other peers. Each peer in the network acts as a client and as a server at the same time.

There are special constraints and requirements in these systems which we have to consider carefully in order to ensure they work efficiently. These are discussed in detail in what follows.

1.2.1 Web-server based access to distributed image databases

Current systems developed to allow users access to distributed multimedia databases over the Internet are modeled after traditional search engines such as Lycos, Alta Vista, etc., which, in order to find relevant images, use textual information provided by the webpages where images are included. Prototype architectures of content-based image retrieval systems over distributed image databases are proposed in [3] [44] to address the problem of integrated access in such environments. The

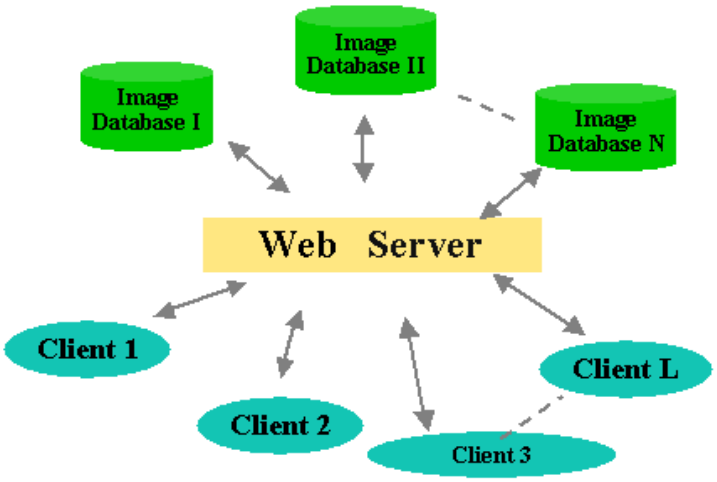


Figure 1.2: A Web-server based Image Retrieval system for content-based online search of large, distributed multimedia databases.

architecture of a Web-server based multimedia database retrieval system is shown in Figure 1.2.

There are three basic components in such systems: Image databases generated from many sources over the world; a Web server which serves as a common gateway linking clients to different image databases; and a number of clients running a querying interface. The web-server stores a meta database which contains predefined features as well as icons of the images from all the databases. The web-server contains all the information needed to decide which of the databases may contain images similar to the query image. When a query is received, the web server runs a similarity search locally by comparing the query features with the features stored in the meta database, calculates the likelihood that a given database contains the target images, and forwards the query request to those database servers that are

most likely to have a related image. As a query the client submits an image or a region of an image. Features from the query image are extracted at the Web-server for each type of query supported by the databases. Then similarity of the query features to the images in the database is computed using an appropriate matching method. There are several characteristics of this architecture which will affect the overall system performance. First, all the query processing computations are carried out at the web server. This can represent a significant computation load for the server, especially when a large number of clients are accessing the server at the same time. Second, sample images are sent to the web server as queries, which can cause bandwidth problems due to image sizes and the cumulative effect of multiple simultaneous queries.

1.2.2 Content-based image retrieval in Peer-to-Peer networks

As another example consider content-based retrieval in Peer-to-Peer (P2P) networks. Instead of having a dedicated server continuously processing queries from all the clients and routing the requests to target databases, the P2P network allows each individual computer to directly share information with each other. Each peer acts as a server and as a client simultaneously. This exactly fits the real scenario where very large amounts of multimedia information are produced every day from many sources all over the world. Each peer can join this information network by simply connecting to one or more peers. Figure 1.3 shows an example of a typical Peer-to-Peer network

topology. The block diagram shown inside Node A demonstrates the functionalities of each peer to support content-based retrieval. Each peer in the network keeps a collection of images and a feature database obtained by extracting features from those images. It maintains its own index structure of the image collection. When a peer initiates an image query, it sends the query message to all its connecting peers. The peers receiving this query request process it and forward it to all their connecting peers. This process continues as new peers receive a query.

Unlike Web servers, peers are highly transient, joining and leaving the network on short time scales. This makes the network topology highly dynamic. P2P systems have no single, well-defined indexing scheme to locate the content information over the network. Every query is broadcasted to every peer in the network. The query messages flooding into the network will potentially increase network traffic, causing slow system responses. For each query received from other peers, the recipient peer has to extract the features from the query image before any similarity is computed.

1.2.3 Distributed content-based Image Retrieval

As pointed out by several researchers [60] [67], a promising future trend in content-based image retrieval is to convert the centralized system model into a distributed computing model. The new model not only allows us to increase the size of image

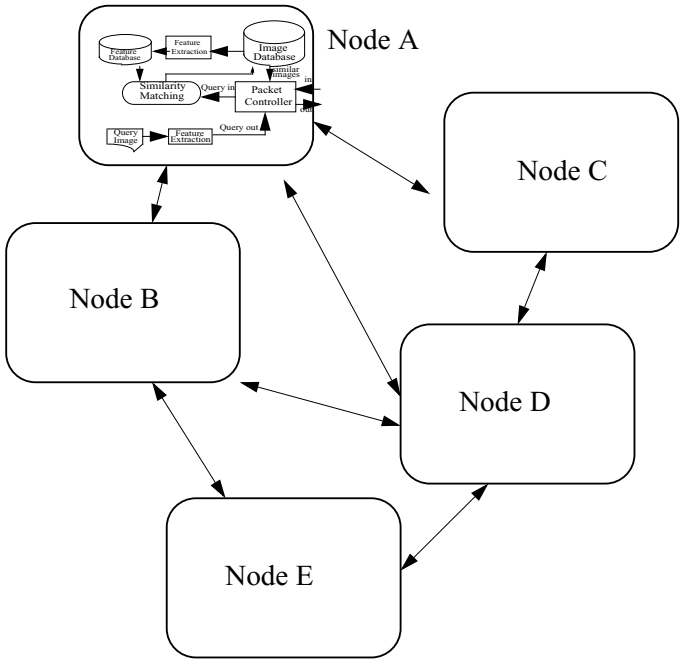


Figure 1.3: Content-based Image retrieval over Peer-to-Peer networks.

collections, but also overcomes the scalability bottleneck problem by distributing the image retrieval processing.

There are several options available to design a distributed content-based image retrieval system. One option is to keep a full-fledged classification engine locally at each client (or at each peer) and then contact the server (or the target peer that has the wanted information) only once the best match images have been identified. This means that retrieval, feature extraction and similarity matching are all performed at the clients. Obviously this method requires a minimum amount of transmission bandwidth since only the image labels need to be transmitted to the server. But it may be too complex for clients that have limited storage space and processing

power. Moreover, if the database at the central server is continually updated, the information at all the clients also has to be updated continuously, which would be complicated by the fact that clients are not continuously connected to the network. The second option is to have the image/video query data compressed and sent to the remote server, where it will be classified. This means that the black-box shown in Figure 1.1 at the client is an image encoder. This second method imposes lower processing burden on the clients, but it may be problematic if bandwidth is limited and the server response may be slow when a large number clients are connecting and sending requests to the server at the same time. We propose an intermediate solution where features are extracted and compressed at the client, then sent to the server, where they are decoded and used for classification. Performing the feature extraction at the client leads to a distributed image retrieval system (i.e., the query processing is distributed between the client and the server). This is shown in Figure 1.4. There are several advantages to employing such a distributed image retrieval architecture as compared with the first two options:

- Reduced computation at the server, because feature extraction is performed at the client and feature decoding (which still needs to be performed at the server) is less complex than feature extraction.

- Lower transmission bandwidth requirement for the clients to communicate with the server, because the features can be represented with far fewer bits than the image itself.

Note that in particular this approach may enable mobile devices to access the multimedia content over Internet, since for these devices transmitting requires more power than processing, and thus it is important to reduce the data volume to be transmitted.

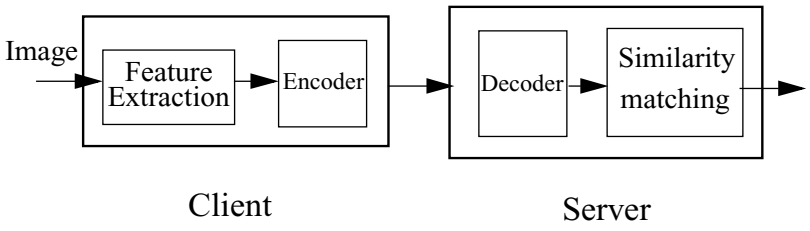


Figure 1.4: Distributed image retrieval system. When clients are accessing image databases which are remotely located, features are extracted and compressed to be sent to the database server.

Compression is essential in this scenario in order to reduce the overall bandwidth when a large number of clients are accessing a server at the same time. This problem becomes worse in peer-to-peer networks since the requests are broadcasted to the network, potentially increasing network congestion and therefore reducing the system performance. Traditional compression algorithms aim at achieving the best perceptual quality of the reconstructed media for a given rate. However in the scenario of distributed image retrieval we discussed above, the end user of the compressed data is not a human viewer, but a classifier. To ensure better retrieval/classification performance, it is beneficial to tailor the compression schemes to this specific application. In this thesis, we investigate two coding schemes for this purpose: classified coding and linear discriminant transform coding. These are introduced next.

1.2.4 Classified Encoding system

In most existing CBIR systems [13] [45] [42], similar indexing methods are used, in the sense that tree structures are employed to hierarchically partition the data space into a number of subregions, each containing a subset of the image objects. Similarity search corresponds to a range query or a nearest neighbor query on the tree structures. Hierarchical indexing structures can speed up the retrieval process in cases where there are a large number of images in the database. As an example, we show in Figure 1.5 an example of the 2-D layout of an image collection [13], the so called *similarity pyramid*. The images are hierarchically partitioned into subgroups. As we traverse down to the lower level of the pyramid, images within a certain subgroup are more similar to each other.

Due to the large variations of the image content in the database, designing a single compression scheme for all the images may be inefficient. Instead, the similarity structure of the images can be exploited to improve compression performance. Classified encoding for distributed content-based retrieval can be summarized as follows: If the client has some knowledge about the database, and thus is able to perform a rough classification of the image submitted by the user, then the query data can be more efficiently compressed using an encoder specifically designed for the corresponding image class. The classified compression scheme will increase the computation complexity at the client. However, the query data can be compressed with higher accuracy at a lower rate, as compared to using a single encoder for

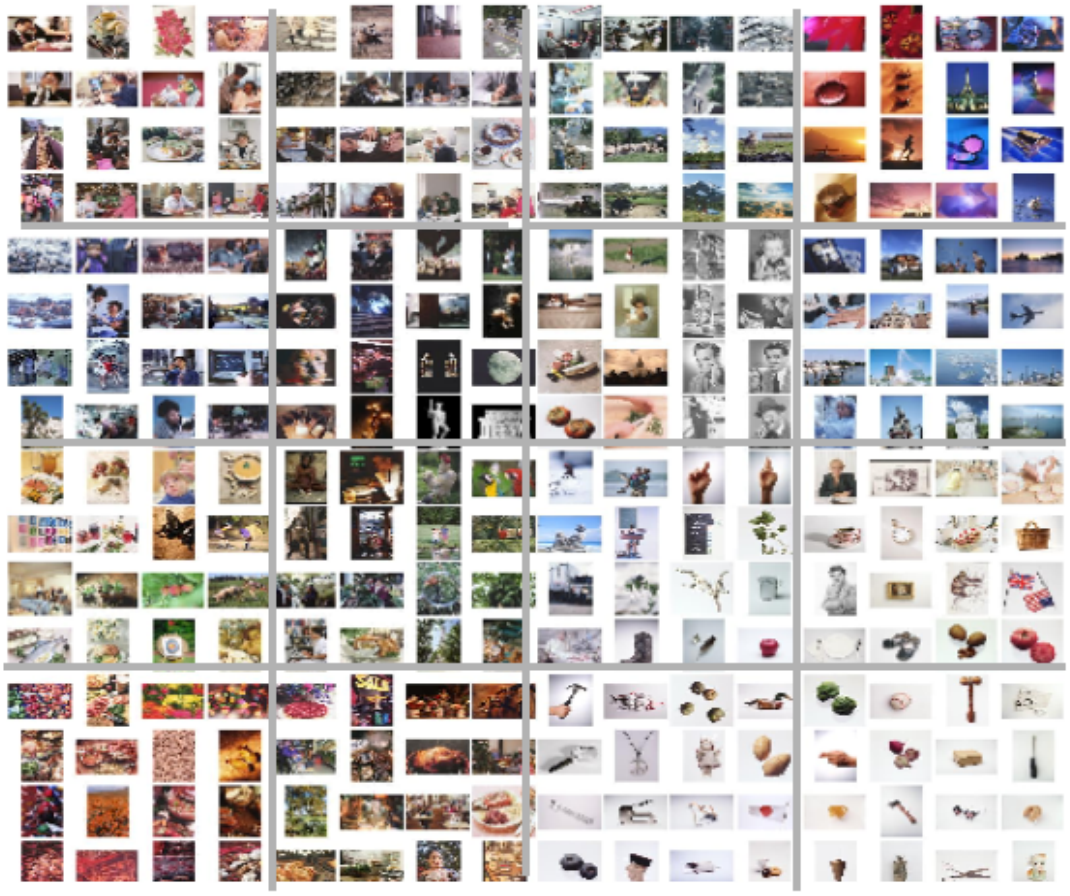


Figure 1.5: 2-D layout of an image collection. Visually similar Images are nearby to each other on this 2D grid.

all inputs. This improved compression performance may be enough to justify the added computation cost. We propose an optimization framework that automatically searches for the best tradeoff between coding efficiency and complexity. Next we introduce this optimization framework in the context of general decision tree classifier (DTC) [62].

The decision tree classifier (DTC) is a multistage approach that breaks up a complex decision into a union of several simpler decisions. Various multidimensional indexing techniques proposed for similarity search in content based retrieval, such as TV-trees [40], X-tree [5], SS-tree [85], SR-tree [39], M-tree [17], Hybrid-tree [10], similarity pyramid [13], etc., can be seen as examples of DTCs. A decision tree structure design strategy can be found in [62] and in references there in. In this thesis, we assume that the tree structure is known a priori.

A tree $\mathcal{T}$ is simply a finite set of nodes, $\mathcal{T} = \{t_0, t_1, t_2, \cdots\}$, where t_0 is the unique root node. The set of leaf nodes is denoted by $\tilde{\mathcal{T}}$. In general a subtree S is a subset of tree $\mathcal{T}$. A pruned subtree $S \preceq \mathcal{T}$ is a subtree rooted at the root node t_0, obtained by pruning some of the branches of $\mathcal{T}$. We denote the number of leaf nodes of tree S by $|\tilde{S}|$. For more detailed and rigorous definitions of these tree terminologies, refer to [6] and [62].

Figure 1.6 shows an example of a general decision tree and one of its pruned subtrees.

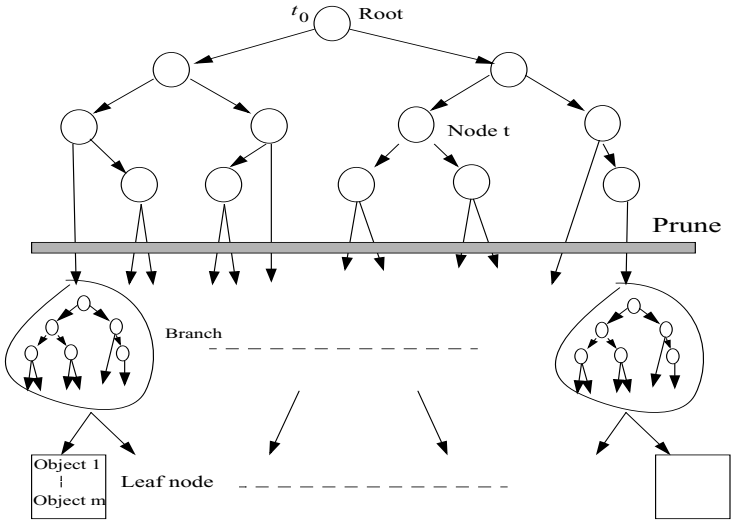


Figure 1.6: A generalized decision tree and its pruned subtree.

Starting from the root, the data space is hierarchically partitioned into a set of subregions. Each node t is split based on a given decision rule. Each of the leaf nodes represents a subregion of the data space and stores a subset of the image objects. We can think of the pruned subtree S as a pre-processing step, i.e., it allows partitioning the data space into $|\tilde{S}|$ classes. Then we can design separate encoders for each of the classes. Figure 1.7 shows the block diagram of a classified compression system. Inputs are first classified and then compressed with compressors specifically designed for each of the classes. By performing a partial classification using S before compressing the data, we are able compress in a similar manner feature vectors that have similar interpretation in a context-based retrieval system. We now list the main factors that need to be considered for designing such a classified compression system.

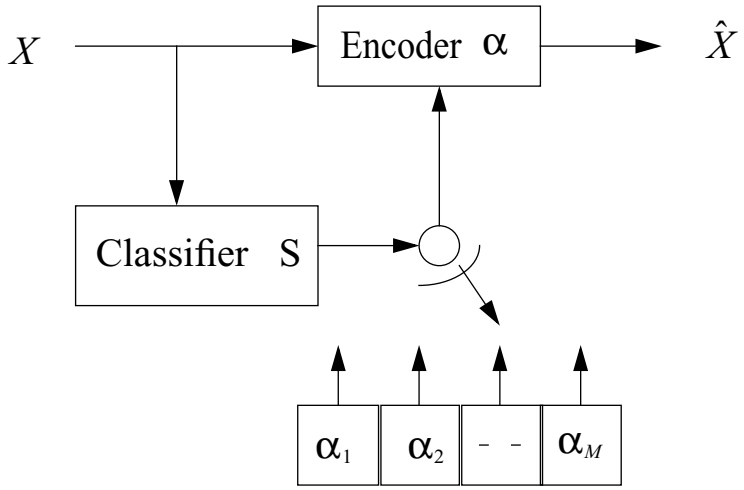


Figure 1.7: Block diagram of a classified quantization system. Separate encoders $\{\alpha_i\}$ are designed for the classes $i = 1, ..., M$. The input vector X is first classified and then encoded with an encoder specifically designed for its class.

Rate constraint. Clients have limited bandwidth to connect to the server and thus the rate required to represent the features should be limited. Moreover, in mobile communication environments rate constraints also correspond to the amount of power required at the client to transmit the query message. Assuming that node t is reached with probability $P(t)$ and that the rate for encoding the data corresponding to node t using encoder Δ_t is $r(t)$, then the overall rate R is the weighted sum of the rates of all the classes:

$$R(S, \{\Delta\}) = \sum_{t \in \tilde{S}} P(t) \times r(t) \tag{1.1}$$

Loss in retrieval/classification accuracy. The classification is based on a reconstructed version of the original query data. Thus, the distortion introduced by

lossy compression at the clients will lead to a degradation in classification accuracy. Thus, ideally we should use the classification error as a distortion metric in designing the system. However, the classification error can not be easily estimated in most applications. In this work we approximate the loss in classification performance by simply using the mean square error (MSE), $d(x, \hat{x}) = |x - \hat{x}|^2$ to measure the fidelity of reconstructed query data, where x and $\hat{x}$ are the original and reconstructed querying data, respectively. The overall distortion is then computed as:

$$D(S, \{\Delta\}) = \sum_{t \in \tilde{S}} P(t) \times \sum_{x \in t} d(x, \hat{x}) \tag{1.2}$$

where $x \in t$ means x belongs to the class in node t. Note that this information is inferred from training data.

Complexity constraint. As compared to not using compression, our proposed coarse classification followed by compression will increase the computational and storage complexity at the clients. Complexity is constrained by the processing power and memory available at each individual client. We define complexity as a weighted sum of (i)the expected length of the tree S, which indicates the amount of preprocessing to be performed, and (ii)the number of leaves of S, which is the number of models that have to be stored at the clients.

$$C(S) = \sum_{t \in \tilde{S}} P(t) \times l(t), \tag{1.3}$$

where $l(t)$ is the length of the path from root t_0 to node t.

Our goal is to find the optimal pruned subtree $S^* \preceq \mathcal{T}$ to be used as a pre-classifier and the set of encoders $\{\Delta_t^*, t \in \tilde{S}\}$, such that the overall distortion is minimized subject to a rate budget R_b and complexity constraint C_b. This can be written as follows:

$$D^* = \min_{S,\{\Delta_t\}} \sum_{t \in \tilde{S}} P(t) \times D(t) \tag{1.4}$$
$$\text{s.t. } R(S, \{\Delta(t)\}) \leq R_b \text{ and } C(S) \leq C_b,$$

where D^* is the optimal distortion, and we optimize the system by selecting the best subtree with its corresponding quantizers, one per class.

In previous proposed Classified Vector Quantization schemes [55][57] for image coding, the classifier and the codebook for each class are designed in a sequential manner and the complexity of the resulting system plays no role in the design process. Instead, in this thesis, we present a framework where the pre-classifier and the quantization parameter for each of the classes are searched in a joint manner under rate and complexity constraints.

1.2.5 Transform coding techniques

In Section 1.2.4, we assumed that the clients have some knowledge about the database, and the encoder is designed to take into account the information which is relevant for retrieval. In this section, we formulate content-based image retrieval as a *statistical*

classification problem, and consider transform coding techniques aiming at minimizing the *probability of classification error*. Instead of assuming that the classifier is known at the clients, we treat the classifier as a black box and design encoders so as to preserve discrimination information between classes. Ideally, we would like that feature vectors with different classes be quantized to different discrete values.

Assume a given feature extraction method and consider a set of labeled images: each image I in the database $\mathcal{D}$ is represented by a pair (X_I, Y_I), where X_I is the feature vector and Y_I the underlying semantic class label of image I. Image retrieval is performed by finding the K most similar objects to the given query feature vector $\mathcal{Q}$. This leads to a mapping $\mathcal{G}$:

$$\mathcal{G} : \mathcal{Q} \rightarrow \mathcal{S} = \{I_1^{\mathcal{Q}}, I_2^{\mathcal{Q}}, ..., I_K^{\mathcal{Q}}, \} \subset \mathcal{D} \tag{1.5}$$

The probability of retrieval error $\Pr(\mathcal{G}(X) \neq Y)$ is the probability that the system is provided a feature vector drawn from class Y and it returns images from classes other than Y. $\mathcal{G}(X)$ is the similarity function that the retrieval system employed to assign a class label of to a feature vector X.

Suppose we have M classes in the database, content-based image retrieval can be naturally posed as an M-ary statistical classification problem, where the optimal mapping is to select as classification label for observation X the label y^* having maximum a posteriori probability $P(y^*|X)$:

$$\mathcal{G}^*(X) = \arg\max_{y_i} \Pr(y_i|X), \quad i = 1, 2, ..., M. \tag{1.6}$$

This is the well known Bayes classifier [25]. The retrieval functions of most existing prototype CBIR systems can be thought of as special cases of this Bayesian classifier, where certain assumptions have been made about the class distributions.

The problem we address is that of designing an efficient transform code, which consists of a linear transform T, a bank of uniform quantizers $\{\Delta_i\}$, and entropy coding γ of quantization indices. Our goal is to find the optimal linear transform T^* and the set of quantization stepsizes $\{\Delta_i^*\}$, such that the probability of classification error based on the compressed data is minimized for a given rate constraint:

$$(T^*, \Delta_i^*) = \arg\min_{T,\{\Delta_i\}} P_e(\hat{X}) \quad s.t. \quad R \leq R_b \tag{1.7}$$

However, in most practical applications, the probability of misclassification can not be easily evaluated and thus various alternative criteria and measures have been proposed and used in practice. In particular, researchers have been interested in a measure of the *overlap* or *class separability*.

There are two types of criteria which are frequently used in practice to evaluate class separability. One is based on a family of functions which give upper bounds of the Bayes classification error. An example of these criteria is the Chernoff Bound [76]:

$$P_e \leq \pi_0^{1-s} \pi_1^s e^{-D_s(f_0, f_1)} \tag{1.8}$$

where π_i is the a priori probability, f_i the distribution function of class i, and $0 \leq s \leq 1$ is the Chernoff exponent, respectively. D_s is the chernoff distance computed as:

$$D_s(f_0, f_1) = \ln \int f_0(x) \left(\frac{f_1(x)}{f_0(x)} \right)^s \tag{1.9}$$

In this work we consider a class separability criterion called scatter measure [25]. It is based on a second-order measure of quality that is defined completely in terms of second-order probabilistic parameters, i.e., means and covariances, of the empirical data. We consider the within-class scatter $\mathbf{S_w}$, which is the scatter of samples around their respective class means:

$$\begin{aligned} \mathbf{S_w} &= \sum_{i=1}^{M} \pi_i E\{(X - M_i)(X - M_i)^t | y_i\} \\ &= \sum_{i=1}^{M} \pi_i \mathbf{\Sigma_i} \end{aligned} \tag{1.10}$$

and the between-class scatter $\mathbf{S_b}$, which is the scatter of the expected vectors around the mixture mean, is defined as in [25]:

$$\mathbf{S_b} = \sum_{i=1}^{M} \pi_i (M_i - M)(M_i - M)^t \tag{1.11}$$

where $\mathbf{\Sigma_i}$ and M_i are the covariance matrix and mean vector for the ith class, respectively, π_i is the a priori probability of class i and M is the overall mean vector. The scatter ratio criterion is computed as:

$$\mathbf{S} = \mathbf{S_w}^{-1}\mathbf{S_b} \tag{1.12}$$

We follow the same philosophy underlying the Karhunen-Loève transform, i.e., the transform that is optimal in terms of preserving the maximum signal energy (measured by covariance) in the fewest transform coefficients. Thus we propose that the optimal transform in terms of preserving the class separability (measured by scatter ratio) be defined:

$$\mathbf{A}^* = \max_{\mathbf{A} \in R^{N \times L}} \frac{\mathbf{A^t S_b A}}{\mathbf{A^t S_w A}}, \tag{1.13}$$

which is the *Linear Discriminant Analysis* (LDA) transformation proposed in pattern recognition applications [25].

We propose to use LDA as a tool for transform coding in classification applications. In order to decide the quantization stepsizes $\{\Delta_i^*\}$, we propose a greedy bit allocation which best preserves the discrimination information between classes for

the given rate constraint. Different from traditional bit allocation, where total distortion is obtained as the sum of distortion in each dimension, here the classification information depends on all dimensions. Thus it is not possible to apply standard Lagrangian techniques to allocate rate to each dimension in order to maximize overall classification performance. To overcome this problem, we propose a greedy scheme where we start with finest quantization for all dimensions and at next iteration we choose the dimension to which a coarser quantization shall be applied such that the magnitude of the ratio of discrimination information loss (measured by class entropy) to decrease in entropy rate is minimized. This process repeats until the rate constraint is satisfied.

1.2.6 Related work and our contribution

Optimal transform coding of images for joint classification/reconstruction was considered in [35]. It was shown that assuming X is a stationary, periodic process under two hypothesis H_0 and H_1, the Discrete Cosine Transform (DCT) would produce uncorrelated components and the optimal transform would be the DCT followed by a diagonal transform, which can be absorbed into scalar quantization. In their latest work [36] the same authors showed that when a cost function combining Chernoff distance (one special case of Ali-Silvey distance) and MSE was used, the Karhunen-Loève transform (KLT) is the optimal transform, under the following assumptions:

1. High rate quantization.

2. The same unitary transform C is a decorrelating transform for data generated under either hypothesis, $H_i, i = 0, 1$.

3. Under hypothesis $H_i, i = 0, 1$, the probability density function $f_i(X)$ is a mixture of N_i Gaussian with a common mean (assumed to be zero):

$$H_i : X \sim f_i = \sum_{j=1}^{N_i} \alpha_{ij} \mathcal{N}(0, \Sigma_{ij}), \quad i = 0, 1 \tag{1.14}$$

where $\alpha_{ij} > 0$ and $\sum_{j=1}^{N_i} \alpha_{ij} = 1$. Each covariance matrix Σ_{ij} is diagonal, hence $X_k, 1 \leq k \leq N$, are uncorrelated under $H_i, i = 0, 1$.

4. The transform has to be unitary.

Although assumption 2 approximately holds for natural images, it might not hold anymore when we deal with image *features* that are extracted from images for the purpose of content-based image retrieval/classification. Furthermore, assumption 3 cannot be justified in general: neighboring pixels in natural images are highly correlated; the feature extraction for content-based retrieval are not necessarily to have uncorrelated features.

In this work, we design transform coding optimized strictly for a *classification* problem. Assuming that the distance between classes is measured by the class separability criterion based on the scatter measure of (1.12), which is based on the second-order statistics, i.e., means and covariances, of the empirical data, we present the following contributions:

- We demonstrate that the LDA is the optimal linear transform;
- We design an efficient bit allocation algorithm to split the rate budget among the transform components, based on classification criteria;
- We derive the relationship between the proposed transform coding and the optimal solution, Likelihood-Ratio Quantization [54] [52] [38], for Gaussian Markov sources;
- We extend the high rate analysis of Likelihood Ratio Quantization [53] for optimal bit allocation in the proposed transform coding for Gaussian Markov sources.

1.3 Relevance feedback in content-based image retrieval

There is a major difficulty associated with CBIR schemes: the semantic gap between low-level features and high-level human concepts. Thus substantial efforts have been devoted to designing techniques that introduce the user into the loop, so that the system can learn the user's particular query preferences. Relevance feedback provides a way for the user to interactively tune the system to her own interest by asking whether certain proposed images are relevant or not. The system then learns from these labeled examples to tune the parameters and returns a new set of similar images, iteratively repeating this process until the user is satisfied with the result.

The construction of such a query updating scheme can be regarded as a machine learning task.

A majority of proposed approaches for relevance feedback in CBIR systems have been developed based on various forms of feature re-weighting [61][49], where the weights associated with each feature for a typical K-Nearest-Neighbor classifier are adjusted based on user feedback. The intuition is to emphasize (i.e., by giving them a more significant weight in the distance computation) those features that are best at discriminating between positive samples and negative ones.

A more systematic formulation of the relevance feedback problem can be achieved by setting up an optimization problem [33], where the goal is to find the optimal linear transformation to map the feature space into a *new* space, that has the property of clustering together positive examples, making it easier to separate them from negative ones.

1.3.1 SVM learning in relevance feedback

More recently, several researchers have proposed the use of support vector machines as an active learning method for the relevance feedback problem in content-based retrieval [11] [32] [14] [31]. We shall briefly review the concept of support vector machine.

Let $\{\mathbf{x}_i, y_i\}, i = 1, \cdots, L, y_i \in \{-1, +1\}, \mathbf{x}_i \in R^n$ be the labeled training set. SVMs are hyperplanes that separate the training data by a maximal margin, with

all vectors labeled +1 lying on one side and all vectors labeled -1 lying on the other side (see Figure. 1.8):

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x}_i + b &\geq +1 \quad for\ y_i = +1 \\ \mathbf{w} \cdot \mathbf{x}_i + b &\leq -1 \quad for\ y_i = -1 \end{aligned} \tag{1.15}$$

where $\mathbf{w}$ is normal to the hyperplane H. The training vectors that lie on hyperplanes $H_0 : \mathbf{w} \cdot \mathbf{x}_i + b = 1$ and $H_1 : \mathbf{w} \cdot \mathbf{x}_i + b = -1$, are called *support vectors*. It can be shown that the margin between the two hyperplanes H_0 and H_1 is simply $\frac{2}{\|\mathbf{w}\|}$. Thus searching for the optimal separating hyperplane becomes a typical constrained optimization problem [9]: minimizing $\| \mathbf{w} \|^2$ subject to the constraints given by (1.15). By introducing Lagrange multipliers, this then leads to maximizing a Lagrangian objective function:

$$\max(\sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j) \tag{1.16}$$

with respect to positive Lagrange multipliers $\alpha_i, i = 1, \cdots, L$, subject to constrains $\sum_i \alpha_i y_i = 0$.

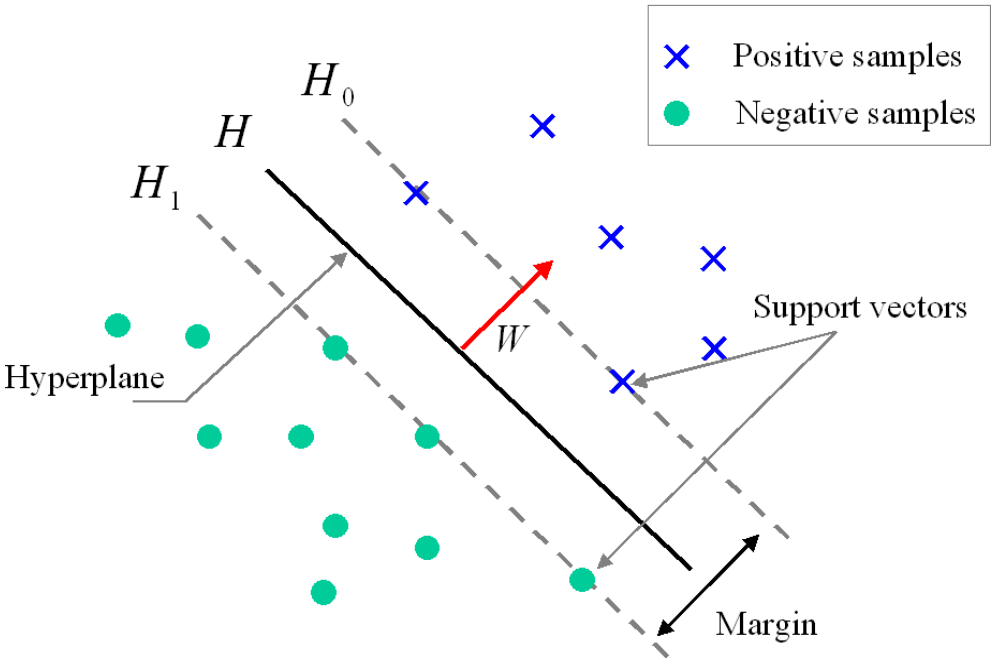


Figure 1.8: The optimal hyper-plane is the one that separates the positive samples from the negative ones with maximum margin.

If the training samples are not linearly separable in the original space χ, suppose that we first map the data to some other Euclidean space $\mathcal{H}$ (possibly infinite dimensional) using a mapping $\Phi : \chi \mapsto \mathcal{H}$. Since the training algorithm only depends on the inner products between sample vectors, we can define a kernel function K such that $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. Then we would only need to replace the inner product $\mathbf{x}_i \cdot \mathbf{x}_j$ by $K(\mathbf{x}_i, \mathbf{x}_j)$ everywhere in the training algorithm (1.16) and would never need to explicitly compute the mapping Φ. The resulting classifier takes the form of $g(\mathbf{x}) : \sum_{i=1}^{N_s} \alpha_i y_i K(\mathbf{x_i}, \mathbf{x}) + b$. $\{\alpha_i, i = 1, \cdots, N_s\}$ and b are parameters that can be learned using quadratic programming [9]. N_s is the number of support vectors.

In [32], SVMs were first incorporated as an automatic tool to evaluate the preference weights of the relative images (one relative image might be closer to the user's concept than another), which was then utilized to compute the parameters of a query refinement [61]. A one-class SVM scheme was developed in [14] that tries to fit a tight hyper-sphere in the non-linearly transformed feature space (through a kernel) to include most positive samples. This scheme only employs the positive samples while totally neglecting the information provided by the negative samples. As an extension, a biased SVM was proposed in [31] to incorporate negative information by employing a pair of hyper-spheres, the inner one includes most of the positive instances while the outer one pushes out most of the negative samples. The unlabeled samples will then be classified as relevant if falling inside the inner sphere and non-relevant if falling outside the outer sphere. We can see that a key assumption made in both schemes is that the positive samples will actually be clustered together in the transformed space. Clearly, there is no guarantee that this will always hold true. Whether clustering does occur (in which case these SVM techniques are likely to be successful) depends on the distribution of positive and negative samples and on the choice of kernel function.

Typical kernel functions that have been used in practice include: linear, polynomial and radial basis function (RBF):

$$Linear: \ K(\mathbf{x}, \mathbf{z}) = \mathbf{x} \cdot \mathbf{z} \tag{1.17}$$

$$Polynomial: K(\mathbf{x}, \mathbf{z}) = (A\mathbf{x} \cdot \mathbf{z} + B)^p \tag{1.18}$$

$$Radial\ Basis : K(\mathbf{x}, \mathbf{z}) \quad = \quad e^{-\gamma \|\mathbf{x}-\mathbf{z}\|^2} \tag{1.19}$$

where $\mathbf{z}$ is another vector of the same dimension as $\mathbf{x}$ and $(\cdot)$ denotes the inner product of two vectors. A, B, p and γ are constants which are set a priori.

1.3.2 Our contribution

Since the kernel function is a key factor to determine the discrimination ability of a SVM in this work we propose a kernel function based on the information divergence between the probabilities of positive and negative samples inferred from the user's preferences. To the best of our knowledge this approach has not been used for relevance feedback in content-based image retrieval systems. Our work is inspired by [43] where a Kullback-Leibler (KL) divergence was used to derive the kernel function for SVM classification in speaker identification and image classification. Note that in [43] domain knowledge is available to model the data distributions that are used in computing the KL divergence. Statistical models such as Gaussian Mixture Models (GMM) or Hidden Markov Models (HMM) can very well model the data and the Expectation Maximization(EM) algorithm can be employed to learn and estimate the parameters. A more theoretical analysis of the use of Kullback-Leibler divergence to derive similarities between image classes, where each image class is modeled as Gaussian Mixtures, can be found in [80]. Although the idea of applying the Kullback-Leibler divergence to SVM learning is not new, in this

work we propose an extension of the framework in [43] for cases where the data distribution model is not known *a priori* and has to be inferred from user feedback.

1.4 Outline and contributions of this thesis

The main contributions of this thesis are,

- *Novel classified quantization design for distributed image retrieval/classification.* We show how to use the database information in order to design a feature compression scheme. Inputs are first classified and then quantized with quantizers specifically designed for each of the classes. By performing a partial classification before compressing the data, we are able to capture the special characteristics of the classes that are relevant to content-based retrieval. We propose a nested optimization framework to jointly search for the optimal pre-classifier and quantization parameters for the classes, such that the overall distortion is minimized subject to both rate and complexity constraints.

- *Exploration of linear discriminant analysis for transform coding in distributed image classification system.* We examine the design of an optimal transform which compacts the maximum amount of class discrimination information in lowest dimensional space, and propose a greedy bit allocation algorithm to minimize the loss in class separability due to quantization. We analyze the relations between proposed transform codes and Likelihood Ratio Quantization

(LRQ), which is the optimal solution, and develop high rate analysis for certain classes of Gaussian distributions.

- *A Novel user preference information based kernel for SVM active learning in relevance feedback.* We propose an approach to derive an information divergence based kernel given the user's preference for SVM active learning in content-based image retrieval. Our proposed kernel function naturally takes into account the statistics of the data that is available during relevance feedback for the purpose of discriminating between relevant and non-relevant images. Experiments show that the new kernel achieves significantly higher (about 17%) retrieval accuracy than the standard radial basis function (RBF) kernel, and can thus become a valid alternative to traditional kernels for SVM-based active learning in relevance feedback applications.

Chapter 2

Feature compression for content-based image retrieval

2.1 Introduction

In content-based image/video retrieval systems, multiple visual features such as texture, color, shape and motion are extracted automatically and used as indexing keys. Consider accessing a multimedia database which contains large amounts of multimedia data. Media in the database is stored in compressed format, while the feature data is extracted from the media data and stored as meta data to represent the information content. Thus there is a need to represent efficiently the metadata that will be used in content-based retrieval. We argue that by compressing this metadata we will enable more efficient database management and fast retrieval [1].

[1] Work in this chapter was published in [86]

Since the JPEG, JPEG 2000, and MPEG standards have been widely used to compress image and video data, DCT or wavelet domain indexing has attracted a lot of attention. Shneier et al. [64] exploited the method of extracting indexing keys from partially decoded JPEG data (i.e., data where Huffman or Arithmetic decoding has been performed). Direct use of DCT coefficients facilitates the process of feature extraction by eliminating the need of decoding the image. Similarly, features can also be extracted from the wavelet domain if a wavelet codec is used for image compression. Vass et al.[83] proposed a new wavelet coding algorithm and utilized it to achieve the combined goal of compression and indexing. Compressed domain indexing reduces system complexity by avoiding decoding the image in the process of feature extraction, and thus seems to make it unnecessary to store separately from image. However, these systems are tightly coupled with the underlying coding schemes and can be indexed only by those features which are extractable from the compressed domain. Facing various types of queries in real applications, compressed domain indexing potentially lacks flexibility, because only a limited set of features are provided.

In order to fulfill different kinds of queries, various feature sets (color, texture, shape, motion,etc.) are extracted to represent the content information of images/videos and stored as metadata in some content-based retrieval systems [45] [68]. Besides the global feature information, local feature extraction is often performed

to provide localized feature information within homogeneous regions [41], thus enabling localized queries that search for given features in specific parts of an image. As databases get larger and larger, storing and transmitting metadata requires significant amount of space and transmission bandwidth and is no longer a trivial task. Consider for example a distributed querying system where the metadata has to be loaded to the remote query processing engine from the local server, the volume of metadata may become a big hurdle for efficient use of transmission bandwidth and fast response to the queries.

In this chapter we propose that by representing metadata in compressed format, we will reduce the storage requirement and transmission bandwidth for this side information. We show by a simple texture classification example that we can achieve an order of magnitude of savings in bit rate, by using compressed features rather than compressing images for querying.

As will be described in Section 2.2, we assume that vectors of features are used for retrieval. Thus vector quantization will be a natural choice if there are no complexity constraints. However in practice, low complexity compression schemes are preferred. We propose to employ bit allocation in the design of low complexity quantizer which operates on scalars or lower dimensional vectors.

2.2 Probabilistic model for content-based retrieval

In general, there are three fundamental components in an image retrieval system [81]: (i) a feature extraction algorithm, (ii) a feature representation algorithm, and (iii) a similarity matching function. Image content is represented by a set of feature vectors. Typical features include color, texture, shape, motion, position, etc. Color content of an image is mostly represented by color histograms, color sets, or coherence color vectors. Texture features can be generated in various ways such as wavelet transform coefficients or Fourier transform coefficients. For example, the texture feature vector of the kth class in the database $\mathbf{F}^k = \{\mu_{mn}^k, \sigma_{mn}^k\}$ can be constructed using the mean μ_{mn}^k and the standard deviation σ_{mn}^k of the energy distribution of the Gabor wavelet coefficients [41] for this image class. This set of feature vectors $\{\mathbf{F}^k\}$ forms a representation space $\mathcal{F}$ for the entries in the database.

In real applications a querying key $\mathbf{Q}$ is generated by the user which specifies her particular interest (and can be a composite of keys given by the system), then similarity matching is performed between the query object and a set of candidates from the database. This process can be viewed as a classification process, where our goal is to find in the representation set $\{\mathbf{F}^k\}$ the closest match to the querying key $\mathbf{Q}$ by a similarity function. The feature representation and similarity matching function should be designed in such a way that the probability of retrieval error is minimized.

A good formulation of the problem of selection of similarity criteria can be found in the work by Vasconcelos and Lippman in [82]. Given a feature vector $\mathbf{Q}$, the goal of retrieval is to find the mapping to the set of classes defined for the retrieval operation:

$$g : \mathbf{Q} \mapsto \mathcal{F} = \{\mathbf{F}^1, ..., \mathbf{F}^M\}. \tag{2.1}$$

The optimal mapping g^* in terms of minimizing the probability of retrieval error is the Bayes Classifier:

$$g^*(\mathbf{Q}) = \arg\max_k P(\mathbf{Q}|\mathbf{y} = \mathbf{F}^k)P(\mathbf{y} = \mathbf{F}^k), \tag{2.2}$$

where $P(\mathbf{Q}|\mathbf{y} = \mathbf{F}^k)$ is the likelihood of observing $\mathbf{Q}$ given that we are in the k-th class and $P(\mathbf{y} = \mathbf{F}^k)$ is the prior probability of class k. As demonstrated in [82], most of the similarity functions currently in use are special cases of the Bayesian criterion. In this chapter, we consider the simplest and most commonly used similarity metric, i.e., the nearest neighbor classifier.

When we assume that the features are Gaussian distributed with equal prior probabilities, (2.2) leads to a minimum distance classifier:

$$g^*(\mathbf{Q}) = \arg\min_k d(\mathbf{Q}, \mathbf{F}^k), \tag{2.3}$$

where $\mathbf{F}^k$ is the feature vector for the kth class, and d is a quadratic distance defined as:

$$d(\mathbf{Q}, \mathbf{F}^k) = (\mathbf{Q} - \mathbf{F}^k)^t \mathbf{B}^k (\mathbf{Q} - \mathbf{F}^k), \tag{2.4}$$

where $\mathbf{B}^k$ is a positive definite weighting matrix. For example, in the case of the Mahalanobis distance measure, it is assumed that all classes have the same covariance $\Sigma_i = \mathbf{B}, \forall i$. Particularly, if the covariance is the identity ($\mathbf{B} = \mathbf{I}$), we have a nearest neighbor classifier $d(\mathbf{Q}, \mathbf{F}^k) = (\mathbf{Q} - \mathbf{F}^k)^t (\mathbf{Q} - \mathbf{F}^k)$.

Figure 2.1 shows the structure of the query system using minimum distance feature matching. This system is basically a Vector Quantizer (VQ) based classifier γ with N outputs, where the goal is to find the best N matches $\{\mathbf{r}_i, i = 1, ..., N\}$ to a given query feature vector $\mathbf{Q}$ among a set of candidates $\{\mathbf{F}^k\}$ using the minimum distance criterion.

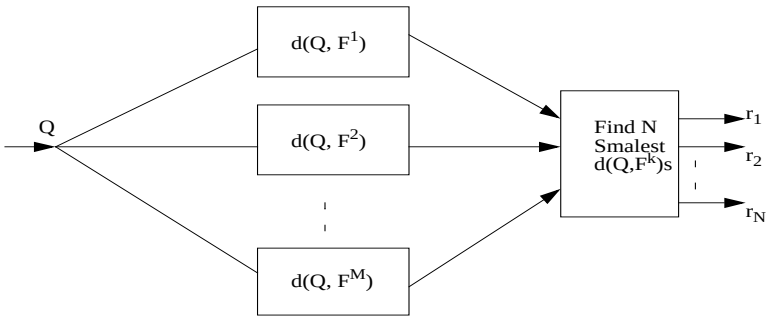


Figure 2.1: Structure of a retrieval system using nearest neighbor classifier.

2.3 Feature compression for minimum distance classifier

Let the original feature set be $\mathcal{F} = \{\mathbf{F}^1, \mathbf{F}^2, ..., \mathbf{F}^M\}$, where $\mathbf{F}^i$ is the feature vector for the i-th image class in the database. Throughout this chapter, we assume *query-by-example* for content based retrieval, which means that the system is provided with an example image represented by feature vector X and the goal is to find the most similar images to X in the database. We also assume in-database retrieval, which means the query image X belongs to one of the image classes in the database.

Suppose that the feature vectors $\mathbf{F}^k$ are L dimensional, then our goal is to design, under a given rate budget R_b, a good source code (consisting of encoder α and decoder β) which maps the L dimensional signal set $\{\mathbf{F}^k, k = 1, ..., M\} \in R^L$ to a smaller set $\{\hat{\mathbf{F}}^k, k = 1, ..., M'\} \in R^L$, such that after applying the classifier γ to the reconstructed feature vectors $\hat{\mathbf{F}}^k$, the performance degradation caused by the quantization is minimized.

$$\min_{\alpha^*,\ \beta^*} P_e(\hat{X}) \quad s.t \quad R \leq R_b \tag{2.5}$$

Although the probability of retrieval error would be the most desirable distortion metric to use for our purpose, estimating this metric so that the minimization can be performed may not be possible in most applications. Alternate metrics, such as class separability measures can be used, this will be addressed in Chapter 4. In

this chapter, we will focus on the mean squared error (MSE) distortion metric due to its simplicity and common usage in nearest neighbor classifier for content based retrieval. Our goal is to design encoder α and decoder β, such that the expected distortion is minimized:

$$\min_{\alpha^*,\beta^*} E(d(X,\beta(\alpha(X)))) \quad s.t \quad R \leq R_b \tag{2.6}$$

Several researchers have addressed the problem of quantization for classification. Vasconcelos and Lippman [81] studied feature representation with a mixture model and showed that Vector Quantization can be used to minimize the Bayes classification error. A joint design algorithm was developed by Perlmutter et al. [51] aiming at minimizing both the quantization error and Bayes classification risk. In both works the dimensions of quantizer and classifier are the same, and the quantizer was optimized for a Bayes classifier, i.e., encoder and classifier can be seen both as vector quantizers operating on input vectors of same dimension. For the particular examples we consider here, this would lead to using quantizers with fairly high dimension, which may not be practical if low complexity encoders are required. Instead, in our work, we address the design of a less complex quantizer which operates on scalars, or lower dimensional vectors, but still with the goal of minimizing the overall classification distortion (see (2.6)) resulting from quantization.

2.4 Experimental results

Here we demonstrate the advantages of representing image content by compressed features, as compared to compressed images, using a simple texture classification example [12]. General image retrieval using multiple features is demonstrated in Chapter 3.

Ten 512×512 texture images are drawn from the Brodatz's texture album [7] and a set of randomly selected sub-images from the original images are used for our experiment. Three-level Dyadic wavelet decomposition is performed on the sample images using 16-tap Daubechies filter and the averaged absolute energies of each wavelet subband compose the feature vector for texture classification [12]. Figure 2.4 shows the structure of the transformed image and the subband numbering used in our feature representation.

The i-th element of feature vector $\mathbf{F}^k$ for the k-th image is computed as the norm in the i-th subband of a Dyadic wavelet decomposition of the image. The simplified Mahalanobis distance is used to perform minimum-distance classification.

In this section we demonstrate that it is efficient to store compressed features as side information, and we evaluate various quantization schemes in terms of classification accuracy and complexity.

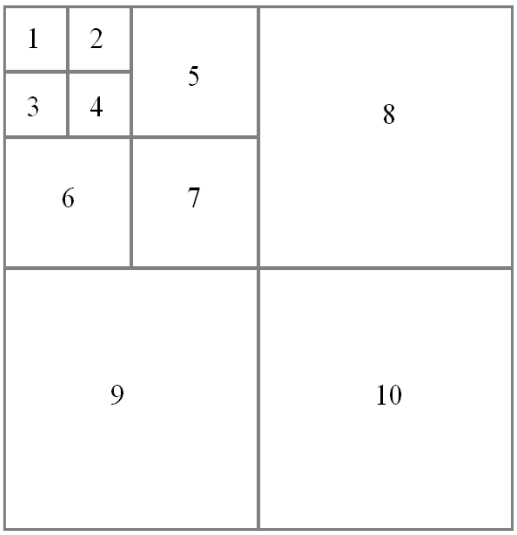


Figure 2.2: 3-level Dyadic wavelet decomposition of an image and the subband numbering used in this chapter.

2.4.1 Efficiency of storing compressed features

Wavelet based texture classification [12] is performed with JPEG and SPIHT [63] compressed images, as well as with explicitly stored features (in compressed format). We compare the classification performance of an ad hoc scalar classification quantization and uniform quantization (see Figure 2.3). Compared with storing uncompressed features (32 bits for each element), our ad hoc classification quantizer achieves the same performance at much lower rate (0.7 bits on average for each element). We also show in the same figure the efficiency of storing features by comparing with operating on SPIHT and JPEG compressed images. Our results

show that the same classification performance can be achieved using three orders of magnitude fewer bits than with SPIHT.

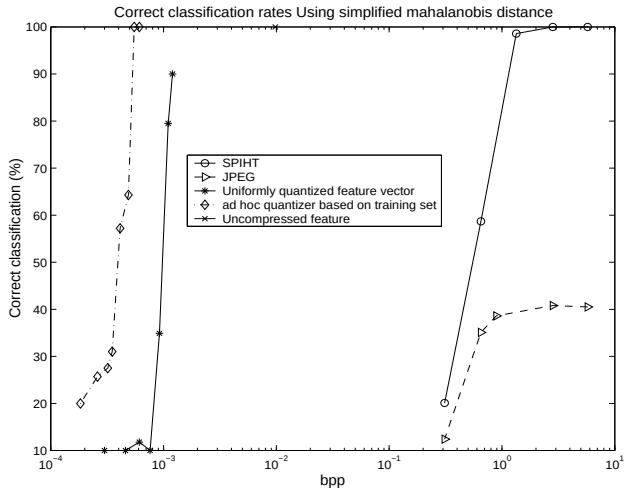


Figure 2.3: Comparison of classification performance using different techniques. When JPEG and SPIHT are used, features are extracted from decompressed images. Notice the gain in terms of bit rate by storing the explicit information. Original feature data is stored as 32 bits per element. We see that even with simplest uniform quantizer 90% correct classification can be achieved at 8 bits per element. Same performance (100% correct classification rate) as using original data can be achieved at 4 bits per element by employing our ad hoc quantizer. The simplified Mahalanobis distance is used for all cases.

2.4.2 Evaluation of quantization schemes

Given the feature vector, VQ would be a natural choice for quantization [26]. However, as the number of classes M gets larger, the coding complexity becomes higher due to the increasing variance in the input data. Here we consider several alternatives that are less computationally intensive. The goal is to design a quantizer α which maps an L dimensional signal set $\{\mathbf{F}^k, k = 1, ..., M\} \in R^L$ to a smaller set $\{\hat{\mathbf{F}}^k, k = 1, ..., M'\} \in R^L$ in a minimum distance sense, under the constraints of rate

budget R_b and complexity of the quantizer. The elements of the vector could be quantized independently of each other (independent scalar quantization) or partitioned into groups of lower dimensions and quantized using a product code. In the following sections we compare several alternatives in terms of Rate-Classification performance and coding complexity.

2.4.2.1 Uniform scalar quantization vs. Bayes VQ

Uniform scalar quantization is the simplest approach we consider. A uniform step size Δ can be used for each vector component, so that each component is quantized independently of each other. In general different stepsizes could be used.

Bayes Vector Quantization (VQ) [51] has demonstrated the possibility of a joint design of quantizer and classifier, where a Bayes risk is introduced into the distortion measure to minimize both the distortion and the classification error. Then the quantizer would assign to each input both a quantization index and a classification label. This algorithm gives superior classification performance as compared to other VQ-based designs for a given classifier and would represent a better approach to optimize quantization of the classification features than the scalar quantization approach described above.

However the Bayes VQ approach has the drawback that performance is optimized for a given query class. Thus, superior performance is achieved for the specific query, but performance could suffer significantly if the quantized feature set is used in a different query. Furthermore, the complexity of VQ grows exponentially with

the dimensionality of the input space. Thus, an approach such as simple uniform quantization, where no optimization has been performed, may be a good approach to quantize the feature vector if the query types are not known a priori.

2.4.2.2 Nonuniform scalar quantization

By nonuniform scalar quantization we mean that the elements of the L dimensional feature vectors are quantized independently of each other using a set of codebooks. The codewords are generated independently for each of the dimensions.

To optimize the codebook design, the Generalized Lloyd algorithm(GLA) [26] can be run in each dimension i to minimize the expected distortion. The Lloyd iteration is terminated when a certain rate has been attained. The total bit budget is distributed by optimal bit allocation, i.e., we find the allocation $R = \{b_i\}$ for each dimension such that the overall distortion

$$D = \sum_{i=1}^{L} D(b_i) \tag{2.7}$$

is minimized subject to

$$\sum_{i=1}^{L} b_i \ \leq \ R_b \tag{2.8}$$

$D(b_i)$ is the distortion in ith dimension associated with rate b_i allocated to it. Lagrangian optimization techniques can be used to solve this problem [23] [65].

2.4.2.3 Product code

In real cases there exists some correlation between the elements of the feature vector. Gains in compression can be achieved by exploiting it. Product codes provide a way of exploiting the correlation between the elements and yet are less computationally expensive than vector quantization operating on the original input vector. A scalar quantizer is a particular case of a product code where all subspaces have dimension 1, but in general the dimensions of the subspaces used in the product code are larger than one.

The method for partitioning the feature vectors is important in the design of a product code. A product code is optimal if the sub-vectors are independent. One approach is to estimate pairwise correlation and group correlated elements into sub-vectors. In this experiment, we partition the elements of the 10-dimensional wavelet texture feature vector $T = \{t_1, t_2, \cdots, t_{10}\}$ into a set of non-overlapping sub-vectors according to their correlation, so that the more correlated feature components (as determined experimentally) are grouped into sub-vectors:

$$\begin{aligned} T_1 &= \{t_1\} \\ T_2 &= \{t_2, t_5, t_8\} \\ T_3 &= \{t_3, t_6, t_9\} \\ T_4 &= \{t_4, t_7, t_{10}\} \end{aligned} \tag{2.9}$$

A codebook C_i is designed for each sub-vector T_i using the GLA algorithm with respect to a given distortion function $D(T_i, C_i)$. Here the rate budget R_b is optimally allocated among these sub-vectors so as to minimize the overall distortion. We compare the performances achieved when using Euclidean distance or modified distortion function defined in (2.4). The results in Figure 2.4 show that the performance is significantly better when this modified distance measure is used.

2.4.2.4 Comparison of performance of different quantizers

Figure 2.4 shows the performance achieved with various quantizers. We see that after optimization (bit allocation with simplified Mahalanobis distortion), significantly better performance can be achieved for both scalar quantization and product code. This can be explained by the fact that finer quantization is chosen along the dimensions where classification distortion is higher after bit allocation. The product code approach always performs better than scalar quantization at low bit rates but results in higher encoding complexity. So the choice of quantizer in a real application should be made based on the relative importance of coding effectiveness and complexity.

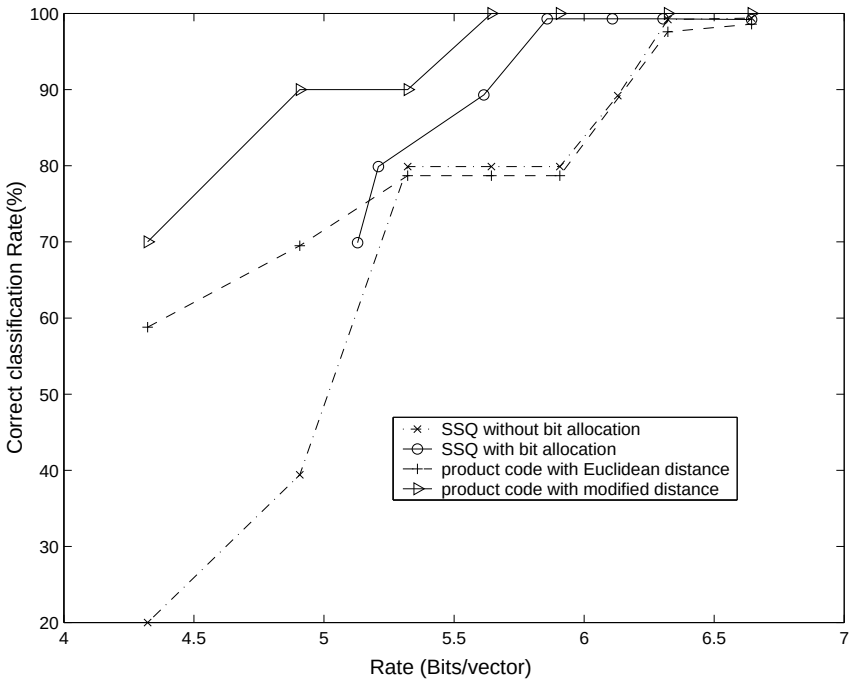


Figure 2.4: Comparison of classification performance using different quantizers. Scalar quantizers are nonuniform, the GLA algorithm is run to get the optimal partition along each dimension with respect to the modified distortion (Mahalanobis distance is this case). Notice the gain achieved by bit allocation. The intuition is that this leads to finer partition along the dimension where classes are prone to mix together (as evaluated by the Mahalanobis distortion).

Chapter 3

Entropy- and Complexity-constrained Classified Quantization Design

3.1 Introduction

In this chapter we address the design of classified encoding schemes for feature compression in distributed image retrieval/classification systems [1]. The idea is to tailor the compression scheme to the specific retrieval task by exploiting information available about the database.

In most existing CBIR systems [13] [45] [42], similar indexing methods are used, in the sense that tree structures are employed to hierarchically partition the data space into a number of subregions, each containing a subset of the image objects. Similarity search corresponds to a range query or a nearest neighbor query on the tree structures. The tree structure breaks down a complex decision (an exhaustive

[1] The work in this chapter was published in [87] [21] [2]

similarity matching with all the image objects in the database) into stages of simpler decisions (traversing down the tree, making a decision at each intermediate node along the path), thus allowing the user to access the database information more efficiently.

Figure 3.1 shows an example of a general decision tree and one of its pruned subtree. A tree $\mathcal{T}$ is simply a finite set of nodes, $\mathcal{T} = \{t_0, t_1, t_2, \cdots\}$, where t_0 is the unique root node. The set of leaf nodes is denoted by $\tilde{\mathcal{T}}$. In general a subtree S is a subset of tree $\mathcal{T}$. A pruned subtree $S \preceq \mathcal{T}$ is a subtree rooted at the root node t_0, obtained by pruning some of the branches of $\mathcal{T}$. We denote the number of leaf nodes of tree S by $|\tilde{S}|$. For more detailed and rigorous definitions of these tree terminologies, refer to [6] and [62].

Starting from the root, the data space is hierarchically partitioned into a set of subregions. Each node t is split based on some decision rule. Each of the leaf nodes represents a subregion of the data space and stores a subset of the image objects. As shown in Figure 3.1, S is one of the pruned subtree which lies above the pruning line. Taking the subtree S as the pre-classifier, we have a classified quantization system shown in Figure 3.2. Inputs are first classified and then quantized with quantizers specifically designed for each of the classes. By performing a partial classification before compressing the data, we are able to capture the special characteristics of the classes that are relevant for content-based retrieval.

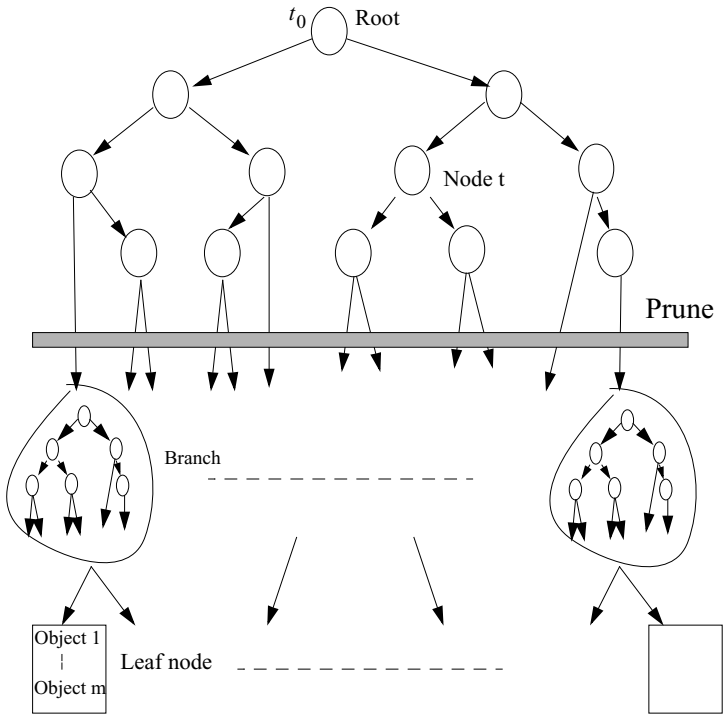


Figure 3.1: A generalized decision tree and its pruned subtree.

In previously proposed Classified Vector Quantization schemes [55][57], the classifier and the codebook for each class are designed in a sequential manner and the complexity of the resulting system plays no role in the design process. Instead, in this chapter, we present a framework where the pre-classifier and the quantization parameter for each of the classes are searched in a joint manner under rate and complexity constraints using the Generalized BFOS algorithm [16].

This chapter is organized as follows: Section 3.2 provides a brief description of classified quantization techniques and introduces the novelty of our work. Section 3.3 states the problem we are addressing and Section 3.4 describes our proposed algorithm to solve the problem. The performance of our proposed method is evaluated in Section 3.5 based on both the Brodatz texture album [7] and a set of natural scenes from the Corel image set [18].

3.2 Classified quantization systems

In Classified Quantization systems [55][57], inputs are first classified and then quantized with quantizers specifically designed for each of the classes. Figure 3.2 shows the block diagram of a generic classified quantization system. There are two components involved in a classified quantization system design:

- *Source modeling.* A classifier is designed to best model the source distributions;

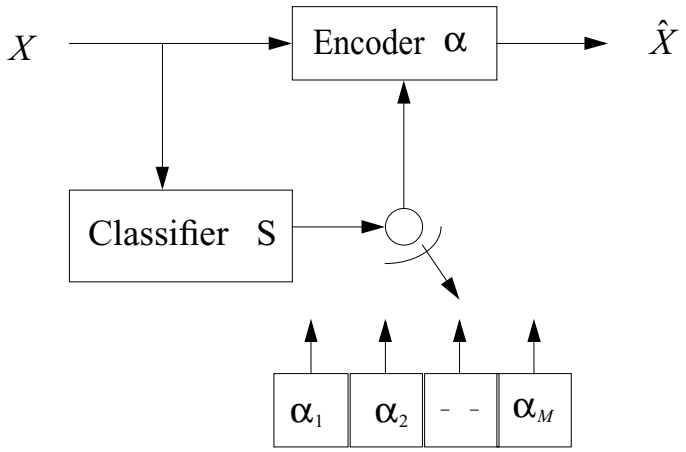


Figure 3.2: The block diagram of a classified quantization system. Separate encoders $\{\alpha_i\}$ are designed for the classes $i = 1, ..., M$. The input vector X is first classified and then encoded with encoder specifically designed for the class.

- *Quantization design.* Optimal quantization design for each of the classes to best exploit the source model.

In this work, we address the issue of optimal design of a classified quantizer in a rate-distortion-complexity framework, for a given decision tree classifier. We especially focus on decision tree classifiers which are built by nearest neighbor rule [90][13][59]. The rate and complexity constraints play the role of deciding how much pre-classification we can afford at the transmitter and what parameters to use for each of the quantizers.

Classified Vector Quantization (CVQ) was proposed by Ramamurthi and Gersho [55] for image coding. An edge-oriented classifier was first designed to classify image blocks into either edge blocks or non-edge blocks. Then codebooks were designed specifically for the resulting classes using the LBG algorithm. CVQ achieved

better perceptual quality with significantly lower complexity as compared to ordinary VQ. Riskin [57] proposed an algorithm to optimally allocate bits among classes using the Generalized BFOS algorithm for an M-class CVQ. The bit allocation was modeled as an $(M, 1)$ tree, where the root had M children and each child was a unary subtree representing a linked list of codebook sizes for the corresponding class in increasing size order. The process of optimal bit allocation among the M classes involves assigning a maximum number of bits to each source and then reducing the number bits in order of increasing magnitude of the ratio of increase in distortion to decrease in rate. Note that the main difference between [57] and our work is that we jointly search for the optimal pre-classifier and bit allocation among classes under both rate budget and complexity constraints. Furthermore, an M-class classifier was assumed to be fixed during the design of quantizers in [57] and [55], and no complexity constraint was imposed in the system design. In this chapter, we focus on the optimal design of (i)the pre-classifier, as a pruned version of the original classifier, and (ii)the quantizers for each of the classes. We assume the original (non-pruned) decision tree classifier is given.

Various multidimensional indexing techniques have been proposed for similarity search in content based retrieval, including TV-trees [40], X-tree [5], SS-tree [85], SR-tree [39], M-tree [17], Hybrid-tree [10], and similarity pyramid [13]. All these tree indexing methods can be seen as generalized decision tree classifiers [62]. The idea of classified encoding for distributed content-based retrieval is to allow the

client perform a coarse classification of the query data, based on a subtree of the original classification tree. After classification, the query data is compressed using an encoder that is specifically designed for the corresponding coarse image class. The classified compression scheme will increase the computation complexity at the client. However, the query data can be compressed with higher accuracy and lower rate by using a classified encoder, as compared to a unified encoder, and as will be seen this improved compression performance may justify the complexity increase in scenarios where bandwidth is scarce.

3.3 Problem formulation

Reasonable structural restrictions are imposed on the quantizers to simplify our formulation. A uniform scalar quantizer is employed as the baseline quantizer due to its simplicity, and entropy coding such as Huffman coding is applied to losslessly encode the quantization indexes. The stepsizes are chosen from a pre-defined discrete set. Figure 3.3 shows the structure of such a coding system. Each classified encoder α_i consists of a bank of uniform quantizers $\{\Delta_{i,1}, \Delta_{i,2}, ..., \Delta_{i,N}\}$ followed by entropy coding $\{\gamma_{i,j}, j = 1, ..., N\}$ of the quantization indexes. The decoder $\beta_{i,j}$ consists of entropy decoding followed by inverse quantization. $\hat{X} \in R^N$ is the reconstructed vector for input $X \in R^N$.

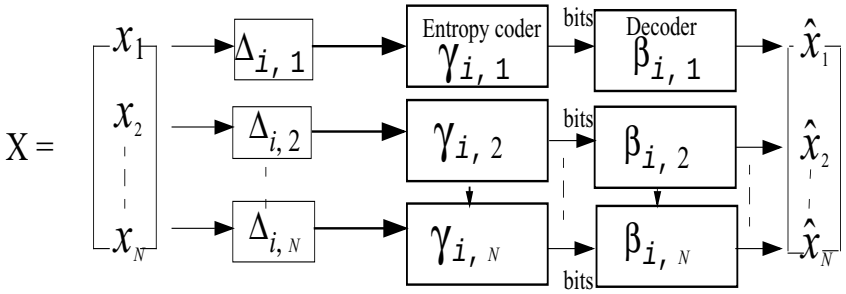


Figure 3.3: Structure of the coding system. A bank of stepsizes $\{\Delta_{i,1}, \Delta_{i,2}, ..., \Delta_{i,N}\}$ are first applied to the scalar components $\{x_1, x_2, ..., x_N\}$ of X. Then independent entropy coding $\{\gamma_{i,j}, j = 1, ..., N\}$ were used to code the quantization indexes. The decoder $\beta_{i,j}$ consists of entropy decoding followed by inverse quantization. $\hat{X} \in R^N$ is the reconstructed vector for input $X \in R^N$.

We review *tree functionals* following the description in [16]. Tree functionals are real-valued functions defined on trees and their subtrees. Examples of tree functionals include the average length $l(S)$, number of leaf nodes $|\tilde{S}|$, and average distortion $d(S)$. A tree functional is *monotonic* if it increases or decreases monotonically as the tree grows. We now formalize notations of some tree functionals that will be used in this chapter.

We use $\tilde{S}$ to denote the set of leaf nodes of S. The number of leaf nodes is represented as $|\tilde{S}|$. In the context of a classified encoder, $|\tilde{S}|$ equals the number of different encoders that will be used; $P(t)$ is the probability that a given input vector traverses node t; $l(t)$ is the length of the path from root t_0 to node t, and reflects the cost of traversing the classification tree S from root to node t; $r(t, \{\Delta_{i,j}\})$ and $d(t, \{\Delta_{i,j}\})$ are the operational entropy rate and distortion, respectively, of the

quantized output for the sample space at node t, with the set of quantization stepsizes $\{\Delta_{i,j}\}$ applied to quantize the data at node t:

$$\begin{aligned} r(t, \{\Delta_{i,j}\}) &= \sum_{k=1}^{N} H(\hat{x}_k | X \in t) \\ d(t, \{\Delta_{i,j}\}) &= \sum_{k=1}^{N} d(x_k, \hat{x}_k | X \in t) \end{aligned} \tag{3.1}$$

where $X \in t$ means that random vector X traverses node t, $H()$ represents the entropy of the quantization indexes.

We define tree functionals $D()$ (distortion), $C()$ (complexity) and $R()$ (rate) as follows:

$$\begin{aligned} D(S, \{\Delta_{i,j}\}) &= \sum_{t \in \tilde{S}} P(t) \times d(t) \\ R(S, \{\Delta_{i,j}\}) &= \sum_{t \in \tilde{S}} P(t) \times r(t) \\ C(S, \{\Delta_{i,j}\}) &= \sum_{t \in \tilde{S}} P(t) \times l(t) + w \times |\tilde{S}| \end{aligned} \tag{3.2}$$

where w is a positive weighting factor. Note that $C()$, our complexity metric, is a weighted sum of computational complexity and the memory required by the classified encoder, roughly estimated by the number of leaf nodes in the pre-classification tree, and hence different encoders used.

Our goal is to find the optimal pruned subtree $S^* \preceq \mathcal{T}$ to be used as a pre-classifier and the set of stepsizes $\{\Delta^*_{i,j}, j = 1, ..., N\}$ for quantization of each class i, such that the overall distortion is minimized subject to a rate budget R_b and complexity constraint C_b.

$$D^* = \min_{S^*, \{\Delta^*_{i,j}\}} \sum_{i=1}^{|\tilde{S}|} P_i \times D_i(\Delta_{i,1}, \Delta_{i,2}, ..., \Delta_{i,N}) \qquad (3.3)$$
$$\text{s.t. } R(S, \{\Delta_{i,j}\}) \leq R_b \text{ and } C(S) \leq C_b$$

where P_i is the probability that a sample belongs to the ith class (the ith leaf node of S). Instead of solving the constrained problem (3.3), we use Lagrange multipliers and solve the dual problem [47]:

$$\min_{S^*}[\min_{\{\Delta^*_{i,j}\}} \{D(S, \{\Delta_{i,j}\}) + \lambda \times R(S, \{\Delta_{i,j}\}) + \mu \times C(S)\}] \qquad (3.4)$$

where $\lambda, \mu \geq 0$. The trade-offs between rate $R()$, distortion $D()$ and complexity $C()$ can be explored by adjusting the two multipliers λ and μ. Now we need to find the optimal multipliers λ and μ such that the rate and complexity constraints are satisfied with equality. This problem is not straightforward since we have two Lagrange multipliers instead of one.

Recent work in the literature has addressed the problem of bit allocation with multiple constraints. Cheung proposed a generalized Gersho-Shoham algorithm in [15] to find the optimal bit allocation among the source coder and the channel coder,

such that the overall distortion is minimized subject to both source and channel rate budgets, for known channel conditions.

$$\min\{D + \lambda R_S + \mu R_C\} \quad s.t. \quad R_S \leq R_{Sb} \quad and \quad R_C \leq R_{Cb} \tag{3.5}$$

where R_S and R_C are the source coding rate and channel coding rate, R_{Sb} and R_{Cb} the source rate budget and channel rate budget, respectively. The basic idea in Cheung's work is to extend the constant-slope concept [47] to 2-dimensional space. In 1-D case, finding the optimal point which minimizes the cost function $J = D + \lambda R$ is equivalent to finding the point in the $R - D$ characteristic that is "hit" first by a "line wave" of slope λ; the Gerhsho-Shorham algorithm [65] addresses the problem of exactly how the slope λ, should be adjusted to approach the target rate budget. Similarly, in the 2-dimensional case, the optimal solution is the point which is first "hit" by a "plane wave" with slope (λ, μ) as shown in Fig. 3.4. The Generalized Gersho-Shoham algorithm proposed by Cheung in [15] iteratively makes adjustments to the slopes (λ, μ) such that the operating pair (R_S, R_C) converges to the target rate constraints.

The major disadvantage of the generalized Gersho-Shoham algorithm is that it takes many iterations to converge if the initial source-channel pair is very far from the optimal.

Since we are working with tree structures, this motivates us to look for optimal pruning methods to solve our problem taking advantage of structure that is specific

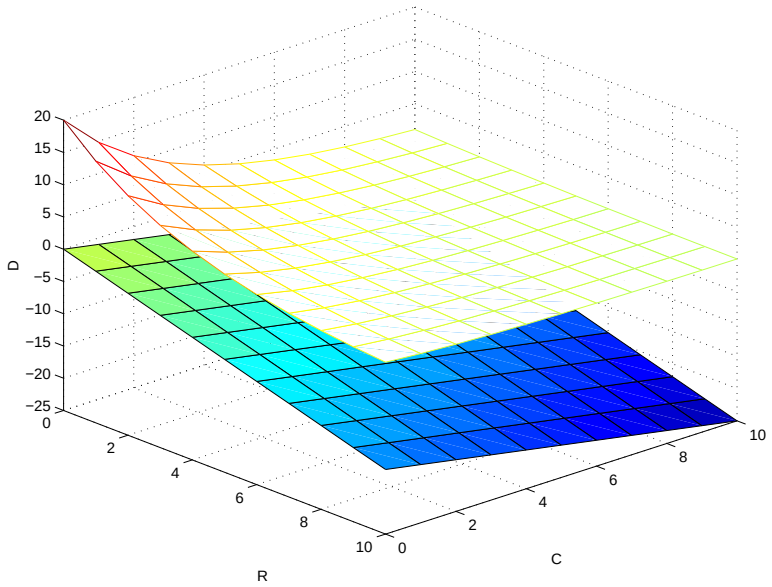


Figure 3.4: For given multipliers λ and μ, minimizing the cost function $J = D + \lambda R + \mu C$ is equivalent to finding the point on the $R - D - C$ surface that is first "hit" by a "plane wave" of slope (λ, μ).

to our problem. In what follows we propose a nested algorithm to iteratively search for the optimal solution of the problem in (3.4).

3.4 Proposed Algorithm

The BFOS algorithm proposed by Friedman et al. [6] is a Lagrangian technique based on minimizing the functional $J(S) = \delta(S) + \lambda \times l(S)$ over all pruned subtrees, with $\delta(S)$ and $l(S)$ being the average distortion and rate, respectively, of the tree structured vector quantizer defined on $\mathcal{T}$ and pruned to S. Chou et al. [16] extended the BFOS algorithm by generalizing the two components of the cost functional to any monotonic tree functionals $U_1(S)$ and $U_2(S)$. This approach prunes off branches T_t of tree $\mathcal{T}$ in order of increasing slope $\mu = \frac{-\Delta(U_1,t)}{\Delta(U_2,t)}$, so as to obtain successive

optimal pruned subtrees. $\Delta(U_i, t)$ is the change of the tree functional U_i (increase for U_1 and decrease for U_2) if the branch rooted at node t is pruned off. It was proven that the generalized BFOS (G-BFOS) algorithm is capable of tracking out the extreme points which lie on the convex hull of the operating (U_1, U_2) pairs over all possible pruned subtrees $S \preceq \mathcal{T}$.

In our problem, we define the following two tree functionals:

$$\begin{aligned} U_1(S, \{\Delta_{i,j}\}) &= D(S, \{\Delta_{i,j}\}) + \lambda \times R(S, \{\Delta_{i,j}\}) \\ U_2(S, \{\Delta_{i,j}\}) &= C(S, \{\Delta_{i,j}\}) \end{aligned} \tag{3.6}$$

As the sample space is hierarchically partitioned by the Nearest-Neighbor classification tree $\mathcal{T}$ defined in Euclidean space, within each node t data tends to be clustered. Due to this clustering property of the classifier, for a fixed multiplier λ, the tree functional U_1 tends to be monotonically decreasing as the tree grows. This is because, as the clusters become "smaller" when the tree grows, the distortion achievable at a given rate is also reduced. On the other hand, the complexity of the system is monotonically increasing since both the depth of the tree and the number of encoders become larger. Because U_1, U_2 are monotonic tree functionals as those described in [16], the generalized BFOS (GBFOS) algorithm can be used for optimal pruning.

This monotonicity of U_1 and U_2 leads to an operational $U_1 - U_2$ function (with a fixed multiplier λ) having the form shown in Figure 3.5. The upper left corner of the

curve corresponds to the singleton tree consisting of just the root node t_0. It means there is no pre-classification before quantization. In this case, we have the smallest encoder complexity (U_2 is minimum) and worst coding performance (U_1 is maximum); The lower right corner of the curve corresponds to the full tree $\mathcal{T}$, where we have maximum encoding complexity (U_2 is maximum), but best coding performance (U_1 is minimum). Starting from the lower right corner $\vec{U}(\mathcal{T}) = (U_1(\mathcal{T}), U_2(\mathcal{T}))$, if we go clockwise along the convex hull, we get: $U_1(\mathcal{T}) \leq U_1(S_1) \leq U_1(S_2) \leq ... \leq U_1(S_m) \leq U_1(t_0)$ and $U_2(\mathcal{T}) \geq U_2(S_1) \geq U_2(S_2) \geq ... \geq U_2(S_m) \geq U_2(t_0)$. And we get a list of nested subtrees: $t_0 \preceq S_m \preceq ... \preceq S_2 \preceq S_1 \preceq \mathcal{T}$. Thus we shall be able to trace out the lower boundary of the convex hull by starting from the full tree and pruning back to the root. This process is for a fixed multiplier λ.

We propose a nested optimization algorithm to jointly search for the optimal subtree S^* and the set of quantization stepsizes $\{\Delta_{i,j}\}$ for a given rate budget and complexity constraint. Refer to Figure 3.6 for a geometric interpretation of this nested optimization.

The basic idea is as follows: First initialize the multiplier λ; then for this fixed multiplier, we choose the set of stepsizes which minimize the functional $u_1(t)$, $\forall t \in \mathcal{T}$. We call this the "tree population" process; then the G-BFOS algorithm is used to prune the original tree $\mathcal{T}$ until the complexity constraint C_b is satisfied. This process corresponds to the curve starting from point $\lambda_{initial}$ with the full-length tree (maximum complexity) to point $R_{initial}$. In Figure 3.5 (with multiplier $\lambda = \lambda_{initial}$), the

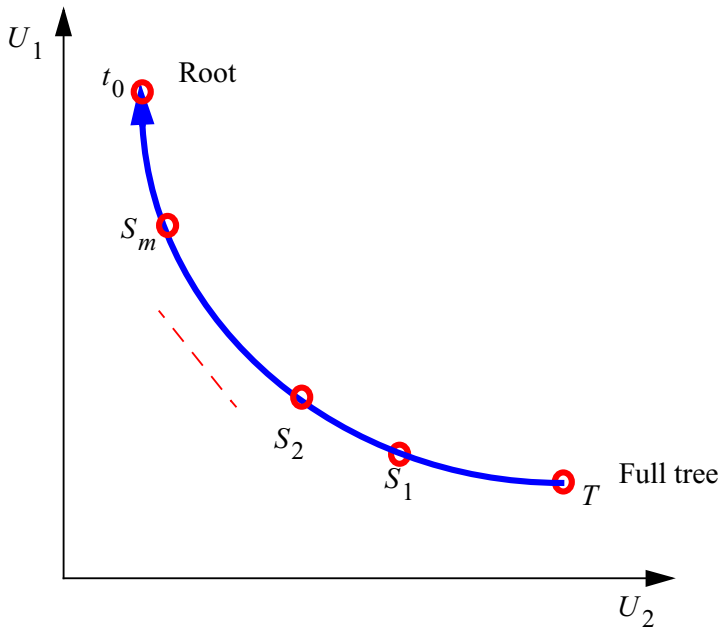


Figure 3.5: Operational (U_1, U_2) pairs for fixed multiplier λ. Each circle represents one pair and one corresponding pruned subtree.

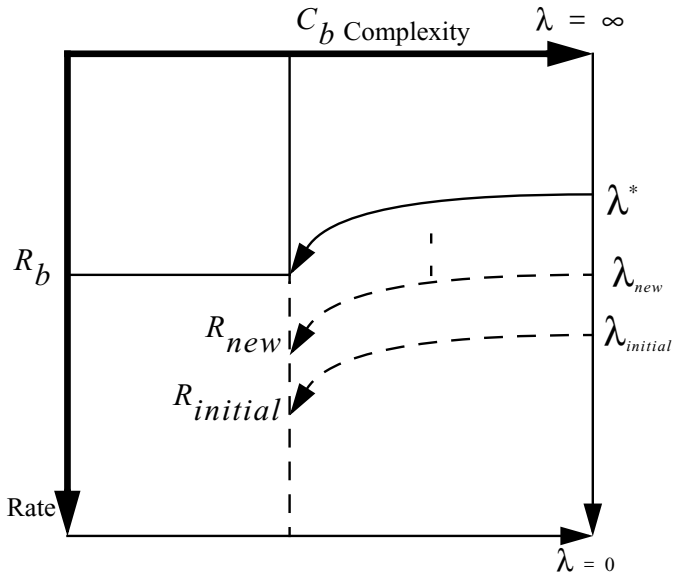


Figure 3.6: We start with multiplier $\lambda_{initial}$, prune the tree until C_b is satisfied. Then update λ to λ_{new} using the bisection method. Repeat the process until R_b is satisfied.

slope of this point will equal to multiplier μ. Now that we have found the optimal multiplier μ to meet the complexity constraint given our choice of λ, we need to adjust λ to approach the rate constraints. We compute the resulting operational rate R with these two multipliers, adjust the multiplier λ to λ_{new} using the bisection method [56] and select the quantization parameters at each node with λ_{new}. Then we prune the original tree $\mathcal{T}$ with the functional $u_1^{new}(t)$ updated with multiplier λ_{new} until the target complexity budget is satisfied, and repeat the process until convergence. This is shown in Figure 3.6 by the curve $\lambda_{new} \rightarrow R_{new}$. Convergence is shown by the curve $\lambda^* \rightarrow R_b$. Note that every time we prune, we always start from the original tree $\mathcal{T}$. The detailed algorithm is as follows:

Proposed Algorithm : *Optimal design of a classified quantizer under entropy and complexity constraints*

Step 0: *Initialize* $\lambda_l \leq \lambda_u$.

Step 1: *Select the quantization parameters at each node* t *such that* $u_1(t)$ *is minimized; prune the tree using G-BFOS until* $C(S) \leq C_b$. *If* $R_{\lambda_l}(S) < R_b$, *let* $\lambda_l^{new} = a \times \lambda_l$ $(a < 1)$ *and repeat* **Step 1**; *otherwise go to* **Step 2**.

Step 2: *Select the quantization parameters at each node* t *such that* $u_1(t) = d(t) + \lambda_u r(t)$ *is minimized, prune the tree using G-BFOS until* $C(S) \leq C_b$. *If* $R_{\lambda_u}(S) > R_b$, *let* $\lambda_u^{new} = \lambda_u \times \frac{1}{a}$ $(a < 1)$ *and repeat* **Step 2**; *otherwise go to* **Step 3**.

Step 3: $\lambda_{next} = |\frac{D_{\lambda_l} - D_{\lambda_u}}{R_{\lambda_l} - R_{\lambda_u}}|$, *Select the quantization parameters at each node* t *such that* $u_1(t) = d(t) + \lambda_{next} r(t)$ *is minimized. Prune the tree until the complexity constraint is satisfied. If* $R_{next} = R_{\lambda_u}$, *stop; Else if* $R_{next} > R_b$, *let* $\lambda_l = \lambda_{next}$, *repeat* **Step 3**; *Else let* $\lambda_u = \lambda_{next}$, *repeat* **Step 3**.

3.4.1 Optimality of proposed algorithm

We denote the pre-classification tree by S, the set of quantizers for the classes by $\{\alpha_i, i = 1, ..., |\tilde{S}|.\}$. Let the admissible quantizer set be designated by $\mathcal{Q}$.

Theorem 1 *For any real, non-negative multipliers,* $\lambda \geq 0$ *and* $\mu \geq 0$, *the solution* $\{S^*, \{\alpha_i^*\}\}$ *to the unconstrained problem*

$$\min_{S \preceq \mathcal{T}, \{\alpha_i\} \in \mathcal{Q}} \{D(S, \{\alpha_i\}) + \lambda R(S, \{\alpha_i\}) + \mu C(S, \{\alpha_i\})\} \tag{3.7}$$

is also the solution to the constrained problem (3.3) with the constraints

$$R(S, \{\alpha_i\}) \leq R_b = R(S^*, \{\alpha_i^*\}) \tag{3.8}$$

$$C(S, \{\alpha_i\}) \leq C_b = C(S^*, \{\alpha_i^*\})$$

The proof of the theorem is an extension of the proof in [65].

Proof: For the solution $\{S^*, \{\alpha_i^*\}\}$, we have

$$D(S^*, \{\alpha_i^*\}) + \lambda R(S^*, \{\alpha_i^*\}) + \mu C(S^*, \{\alpha_i^*\}) \tag{3.9}$$

$$\leq D(S, \{\alpha_i\}) + \lambda R(S, \{\alpha_i\}) + \mu C(S, \{\alpha_i\})$$

for all $S, S^* \preceq \mathcal{T}$ and $\{\alpha_i\}, \{\alpha_i^*\} \in \mathcal{Q}$. Thus we have:

$$\begin{aligned} & D(S^*, \{\alpha_i^*\}) - D(S, \{\alpha_i\}) \qquad (3.10) \\ \leq \; & \lambda(R(S, \{\alpha_i\}) - R(S^*, \{\alpha_i^*\})) + \mu(C(S, \{\alpha_i\}) - C(S^*, \{\alpha_i^*\})) \end{aligned}$$

This is true for the choices of S and α which satisfy the rate and complexity constraints (3.8). Since $\lambda \geq 0$ and $\mu \geq 0$, we have:

$$D(S^*, \{\alpha_i^*\}) - D(S, \{\alpha_i\}) \leq 0 \qquad (3.11)$$

That is, $\{S^*, \{\alpha_i\}\}$ is the optimal solution to the constrained problem.

■

Proposition 1 *The problem of finding:*

- *An optimal pruned subtree $S^* \preceq \mathcal{T}$, and*
- *the associated optimal set of quantizers $\{\alpha_i^*\}$ for each of the classes corresponding to the leaf nodes of S.*

can be solved by proposed nested algorithm.

Proof: The nested algorithm first fixes the multiplier λ and prunes the tree until the complexity constraint C_b is satisfied. This process corresponds to finding: the optimal multiplier μ^* for the given λ; the optimal pruned subtree S^* and the set of quantizers $\{\alpha_i\}$ which minimize the cost function $J(\lambda, \mu) = D + \lambda R + \mu C$.

Assume there is another pruned subtree S' which has the same complexity as S^*, but better coding performance:

$$D' + \lambda R' < D^* + \lambda R^*. \tag{3.12}$$

This means that S' lies on the convex hull of operational (U_1, U_2) pairs. $U_1(S') < U_1(S^*)$ implies that S^* is a pruned subtree of S', i.e., $S^* \preceq S'$. Thus $C^* < C'$, which contradicts our assumption.

■

3.5 Experimental results

We evaluate the performance of the proposed method based on three image databases: the Brodatz texture album [7], a subset of the Corel images [18], and a rotated set of Brodatz texture [8].

3.5.1 Texture classification

In this experiment, we use Brodatz texture images. We extract $n = 200$ sub-images (of size 128×128) from different locations of each original image (of size 512×512). We use half of these images for training and the other half for testing. Feature vector is computed as the energy in the subbands of the wavelet decomposed images using Daubechies-16 filter. We apply 3 level of decomposition so that the feature vector X is 10 dimensional. We assume that each Brodatz texture represents a class. We first build a binary decision tree $\mathcal{T}$ in a top-down manner using the K-means algorithm until only one class is left at each leaf node. The clustering for K-means algorithm is based on the Euclidean distance between feature vectors. The quantization stepsizes $\{\Delta_{i,j}\}$ are chosen from a predefined discrete set $\{32, 16, 8, 4, 2\}$. Figure 3.7 shows examples of operational (U_1, U_2) pairs for different values of multiplier λ. These points are obtained by two methods of pruning: Depth-first and In-Order-Walk [19]. We can clearly see the monotonicity of U_1 and U_2.

Figure 3.8 shows the Rate-Distortion-Complexity surface we obtain. We can see the tradeoff between rate, distortion and complexity of the system and we can verify the convexity of the surface, as predicted by Goyal and Vetterli [27]. In Figure 3.9, we show the rate-distortion performance for this coding system with and without pre-classification. Substantial coding gain is obtained by employing a classified quantizer instead of a single quantizer, or by using a higher-complexity classification tree. For example, at coding rate of 2 bits/element, we reduce the MSE from around 700

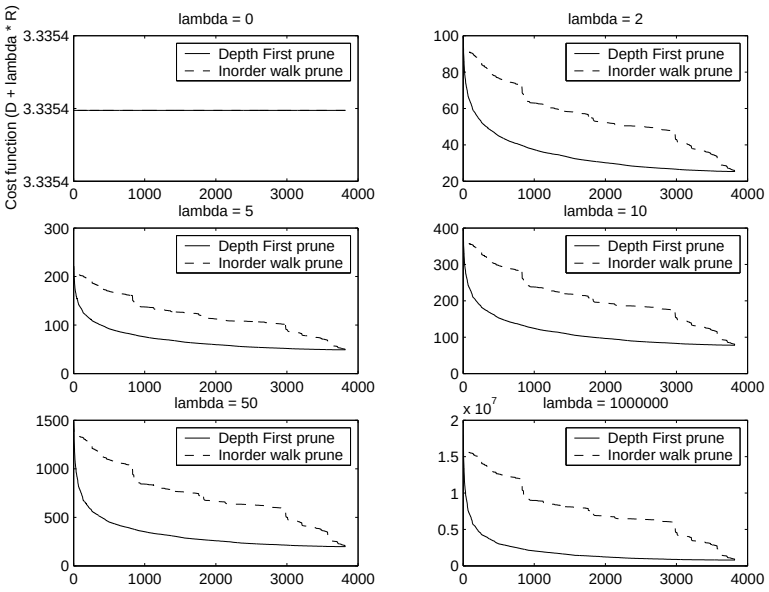


Figure 3.7: Tree functionals U_1 and U_2 for different values of multiplier λ. The x-axis is the complexity U_2 with the weighting factor w equal to 400. The y-axis shows the cost functional U_1. The operating points are obtained by computing the two functionals of arbitrary subtrees through two different pruning methods: depth-first and in-order-walk.

to 150 using a pre-classification tree of average length 3.88, to around 20 using a pre-classification tree of average length 6.47. In real applications, depending on how much complexity is allowed for the encoder, we are able to choose the best subtree as the pre-classifier which gives maximum coding gain within the given complexity constraint.

We also perform texture classification with the compressed data using the decision tree classifier $\mathcal{T}$. The result is shown in Figure 3.10. As is clearly shown in the figure, a lower classification error rate was achieved by using a classified encoder, as compared to using a single encoder. At transmission rate of 20 bits/vector, a reduction of 11% in the classification error rate is achieved by using the classified

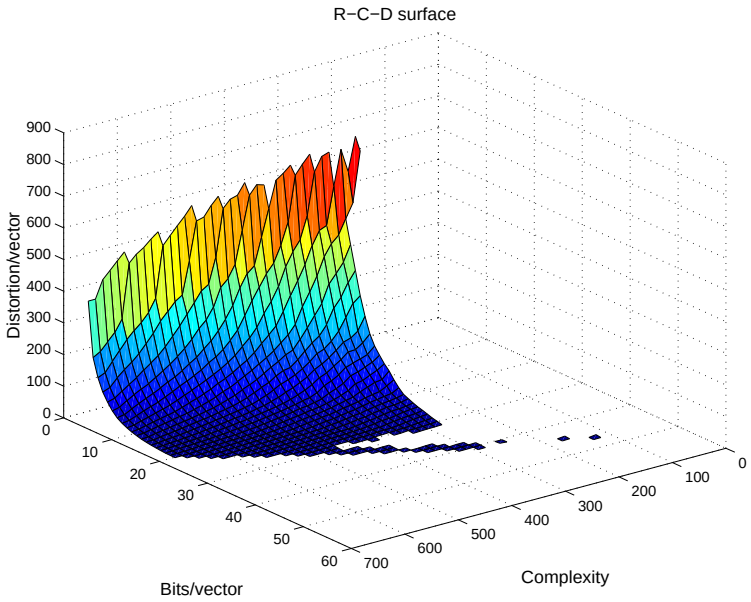


Figure 3.8: The Rate-Distortion-Complexity surface obtained by proposed optimization framework. We employed traditional mean square error in this example. The complexity is evaluated as the cost of traversing the tree plus a weighted storage cost for storing the encoders, with the weighting factor $w = 1$ for this example. Rate is computed as the entropy rate of the quantization outputs.

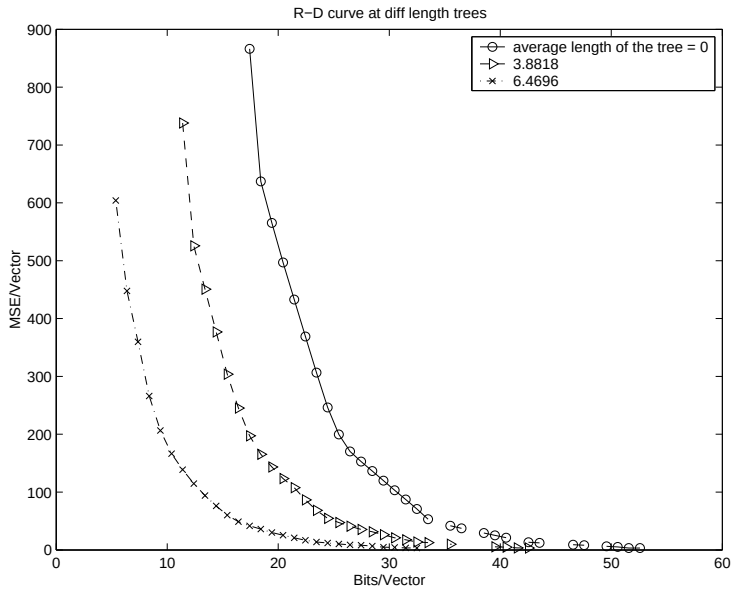


Figure 3.9: Comparison of R-D performance between systems with pre-classification (using tree lengths 3.8818 and 6.4696) and without pre-classification (tree length 0).

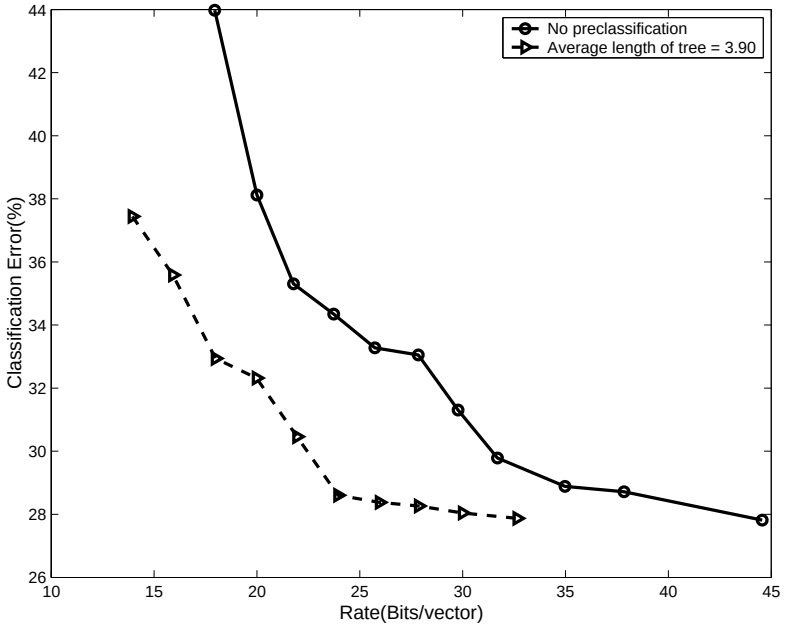


Figure 3.10: Comparison of classification performance for systems with pre-classification (where tree length is 3.90) and without pre-classification.

encoder, with an average pre-classification tree length of 3.9. Although we assume the traditional mean squared error distortion in this experiment, our results could be extended to other distortion measures, although further study would be needed to identify alternative metrics, such as Kullback-Leibler divergence used in [71].

3.5.2 Corel Image retrieval

We use the feature extraction algorithm developed in [13]. The features consist of color histogram, texture histogram and edge histogram in La^*b^* color space. Table 3.1 shows the structure of the feature vector we used for Corel images. The dimensionality of the feature vector is 99.

color histogram		Dynamic range	Number of bins
	L^*	$0 \cdots 100$	8
	a^*	-100 $\cdots$ 100	16
	b^*	-100 $\cdots$ 100	16
texture histogram		Dynamic range	Number of bins
	L^*	$0 \cdots 100$	8
	a^*	$0 \cdots 200$	16
	b^*	$0 \cdots 200$	16
edge histogram		Dynamic range	Number of bins
	L^*	$0 \cdots 2\pi$	9
	a^*	$0 \cdots 2\pi$	9
	b^*	$0 \cdots 2\pi$	9

Table 3.1: Structure of feature vector.

We use 15 classes [2] from the Corel image database [18], with 100 images per class. We use 80% of each class for training and 20% for testing. K nearest neighbors to the query vector are returned as the retrieval set. The distance function between two feature vectors is computed as the l_1 norm:

$$D(Q, X) = \sum_{i=1}^{N} |Q_i - X_i| \tag{3.13}$$

Experimental results are shown in figure 3.11. We performed retrieval for $K = 20,\ 50$. The quality of the retrieval result is measured by two quantities: *precision* and *recall*. Precision is the percentage of relevant objects in the retrieved set, it measures the purity of the retrieval. Recall is a measurement of completeness of the retrieval, computed as the percentage of retrieved relevant objects in the total

[2]The classes include Flowers II, Exotic cars, Sunrise and sunsets, Religious stained glass, Ski scenes, Painting, English country garden, Land of the pyramids, Mayan and Aztec ruins, Divers and diving, Glacier and mountains, Owls, Arabian horses, Coasts and Fireworks.

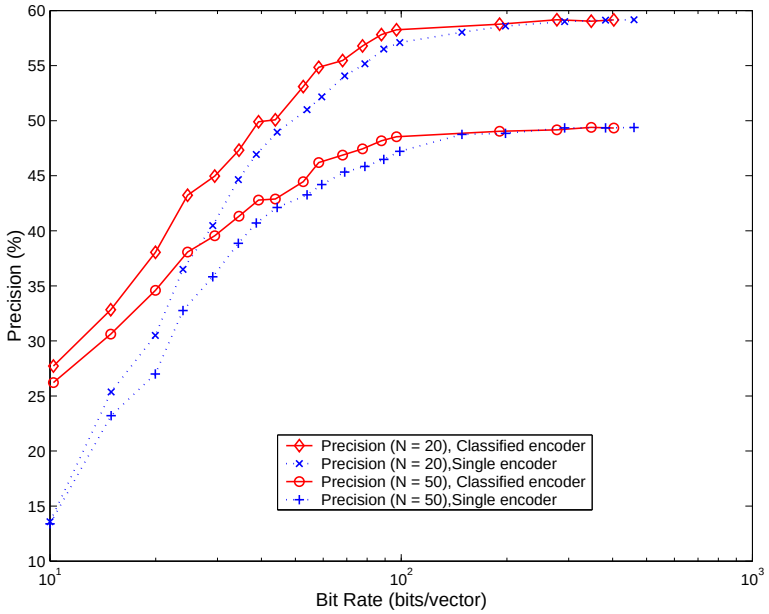


Figure 3.11: Retrieval Precision vs. rate curve comparing classified encoder and single encoder

relevant set in the database. Figure 3.11 shows the precision of 20 and 50 retrieved images using the proposed classified encoding and a single encoder. Figure 3.12 shows the precision-recall curve of the two encoding systems operating at various bit rates. Again, we see that substantial gain is achieved by employing the proposed classified encoder, especially at low operating bit rates. At bit rate around 0.2 bits/sample, our proposed encoding scheme achieves about 7% higher retrieval precision than a system based on a single encoder. The retrieval performance of proposed encoder is significantly better than standard single encoder, especially at low bit rates.

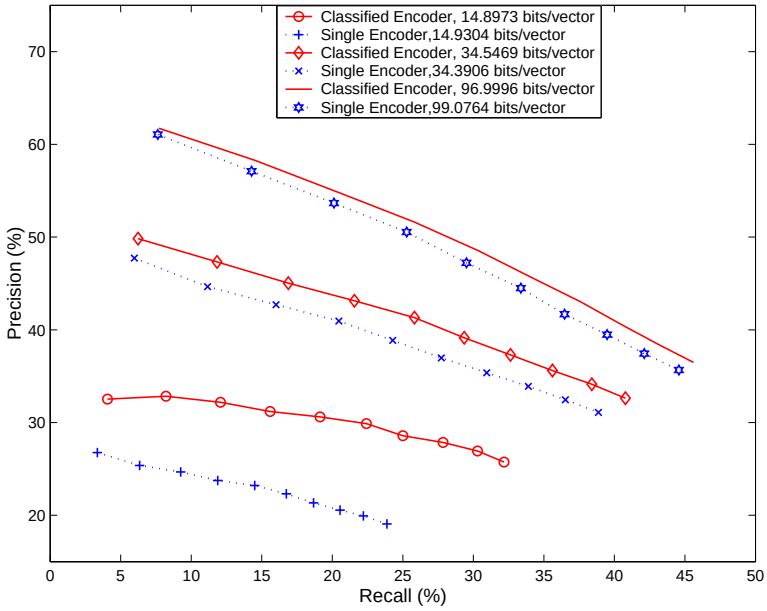


Figure 3.12: Precision-Recall curve comparing the two encoding schemes at different operating rates.

3.5.3 Rotation-invariant texture classification using steerable features

In this section, we address the compression of steerable features proposed by Beferull-Lozano et al.[2] for rotation-invariant texture image retrieval, using classified encoding [3]. The feature extraction is based on the subbands obtained from a steerable pyramid [66]. Steerable representations are such that from the output produced by projection onto basis functions, one can obtain output of projection onto rotated basis functions. Beferull-Lozano proved that the same is true for the computation of the features [2]: Given the features of an image oriented at any angle ϕ, the features

[3]This was work done in collaboration with Dr.Baltasar Beferull-Lozano.

corresponding to the same image but oriented at an angle ϕ', can be computed from the features at angle ϕ.

The feature vector is computed as the set of correlation matrices $\{\mathbf{C}^l\}_{l=1}^{L}$, where $\mathbf{C}^l$ is the correlation matrix of level l decomposition with elements:

$$C_{ij}^l = \left(\frac{1}{N_l}\right) \sum_{k=1}^{N_l} c^l(\mathbf{x}_k, \phi_i) c^l(\mathbf{x}_k, \phi_j) = C_{ji}^l, \quad l = 1, \cdots, L. \tag{3.14}$$

We cite the Proposition in [2] without proof.

Proposition 1 *Given a steerable representation with J basic angles, the correlation matrices $\boldsymbol{C}_{I_\theta}^l$ and $\boldsymbol{C}_I^l$, both evaluated with respect to the same set of basic angles $\{\phi_1, \ldots, \phi_J\}$, are related as follows:*

$$\boldsymbol{C}_{I_\theta}^l = \boldsymbol{R}(\theta) \boldsymbol{C}_I^l \boldsymbol{R}^T(\theta),$$

$$\boldsymbol{R}(\theta) = \begin{pmatrix} \alpha_1(\phi_1 - \theta) & \alpha_2(\phi_1 - \theta) & \cdots & \alpha_J(\phi_1 - \theta) \\ \alpha_1(\phi_2 - \theta) & \alpha_2(\phi_2 - \theta) & \cdots & \alpha_J(\phi_2 - \theta) \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_1(\phi_J - \theta) & \alpha_2(\phi_J - \theta) & \cdots & \alpha_J(\phi_J - \theta) \end{pmatrix} \tag{3.15}$$

In the particular case where the J basic angles are taken to be equally spaced, then $\boldsymbol{R}(\theta)$ becomes an orthogonal matrix for any θ, and therefore, $\boldsymbol{C}_{I_\theta}^l$ and $\boldsymbol{C}_I^l$ become orthogonally equivalent.

We propose a The similarity measurement $D(I_1, I_2)$ between 2 different images I_1 and I_2 as follows:

$$D(I_1, I_2) = \mathrm{Min}_\theta \left(\sum_{l=1}^{L} \|\mathbf{C}_{I_1}^l - \mathbf{R}(-\theta)\mathbf{C}_{I_2}^l \mathbf{R}^T(-\theta)\|_F \right) \tag{3.16}$$

In words: an angular alignment is first carried out between the two images and the distance is the minimum one obtained, i.e., when the angular alignment is most accurate. In this section, we incorporate the rotation-invariance in the DTC based retrieval by performing angular alignment at each node in the tree and defining an appropriate distance between an image and a tree node, which ensures that a best-first-search method works correctly.

In the experiment, the decision tree classifier $\mathcal{T}$ is built the same way as in Section 3.5.1 based on the feature $\{\mathbf{C}^l\}_{l=1}^L$. The distance used for clustering is the l_1 norm between two feature vectors. Let Q be the feature vector of a query image I_q and t be a node in the tree $\mathcal{T}$. We use $\underline{D}(Q, t)$ to denote the distance of the query feature vector Q with node t .

In order to ensure that this search algorithm finds the correct closest matches, we need to define the distance $\underline{D}(Q, t)$ satisfying the property that $\underline{D}(Q, t)$ is a lower bound of the distances of Q to all the images in node t, and we need to take into account the angular alignment process. Notice that:

$$\begin{aligned} \underline{D}(Q, I) &= \min_\theta d(Q_\theta, I) \qquad (3.17) \\ &\geq \min_\theta d(Q_\theta, I_c) - d(I_c, I) \text{ (triang. inequality)} \end{aligned}$$

$$\geq \min_{\theta} d(Q_\theta, I_c) - R(t) \qquad \text{(upper bound)}$$

where I_c is the centroid in node t, $R(t)$ the radius of node t given by $R(t) = \max_{I \in t} d(I_c, I)$ and $d(Q_\theta, I)$ is given by the expression inside the parenthesis in (3.16). Thus, defining $\underline{D}(Q,t)$ as $\underline{D}(Q,t) = \min_\theta d(Q_\theta, I_c) - R(t)$, then, it is guaranteed that the best-first-search method will find the correct closest match. Thus, we see that a crucial differential point in our work is that in the retrieval process using the DTC, at each node of the tree, alignments between the query (quantized) feature vector and each of the two representing vectors (corresponding to the two branches) have to be performed using (3.16). After these two alignments, two distance measurements are performed and a branch is chosen.

We have evaluated the performance of our proposed method applied to the Brodatz texture images [8]. The feature vectors are quantized using:(i) Simple uniform quantization (same stepsize); (ii) Non-uniform quantization with optimal bit allocation in a rate-distortion sense [65]; (iii) Classified quantization. We use 2 collections of texture samples of size 128×128. The first collection, which forms the *non-rotated* image database, is obtained by partitioning each of the 13 Brodatz (512×512) non-rotated texture images [8] into 16 non-overlapping texture subimages of size 128×128 with a total of 208 texture samples. This set is used in training of the DTC for retrieval. The second collection, which forms the *rotated* set, is obtained by partitioning (for each of the 13 texture classes) 4 large texture images oriented at 30, 60, 90 and 120 degrees also into non-overlapping subimages of size 128×128

and taking the 4 central subimages. In this way, in the second database, there are also 16 textures for each class and therefore, also the same total number of 208 textures. A query texture sample is taken from the *rotated* set and the feature vector is extracted and quantized using the three quantization schemes mentioned above. We assume that each quantized component of the feature vector is independently entropy coded. The $M = 16$ closest textures from the *non-rotated* set are obtained and the average retrieval precision over all the rotated texture samples is measured.

Fig. 3.13 and Fig. 3.14 show the retrieval performance of compressed steerable feature vectors for $J = 2$ and $J = 4$. We can clearly see that the classified quantizer achieves the best performance among the three quantization schemes. By using the classified quantizer with average tree length $l = 2$ (complexity constraint), the retrieval performance degrades very gracefully. Even with the bit rate reduced to around 1 bit/element, we can still achieve about the same precision as when using uncompressed features.

With respect to the retrieval complexity reduction by employing a DTC instead of a linear search, we have computed the number of distance computations that have to be performed to find the $M = 16$ closest matches. Instead of 208 distance computations as in the case of linear search, the DTC requires on average 121.97 for $J = 2$ and 39.82 for $J = 4$.

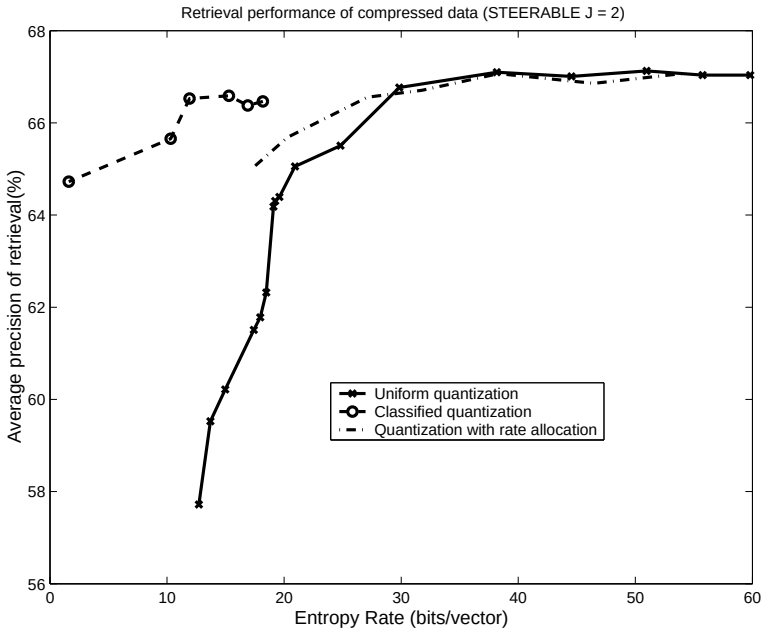


Figure 3.13: Average Retrieval Performance with compressed steerable features using uniform quantization, single encoder with bit allocation, and classified encoding. The feature extraction uses a 3 level steerable pyramid with 2 basic angles.

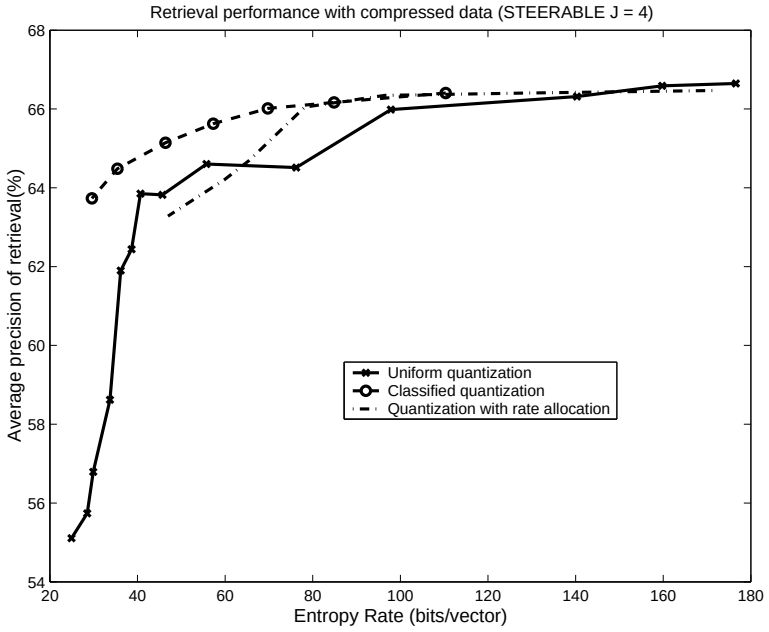


Figure 3.14: Average Retrieval Performance with compressed steerable features using uniform quantization, single encoder with bit allocation, and classified encoding. The feature extraction uses a 3 level steerable pyramid with 4 basic angles.

Chapter 4

Transform Coding for Distributed Image Classification/Retrieval

4.1 Introduction

In Chapter 3, we designed a classified quantization scheme for feature compression in distributed image retrieval systems. In this chapter, we formulate content-based image retrieval as a *statistical classification* problem, and consider transform coding techniques aimed at minimizing the *probability of classification error*[1]. We assume a given feature extraction method and consider a set of labeled images: each image I in the database $\mathcal{D}$ is represented by a pair (X_I, Y_I), where X_I is the feature vector and Y_I the underlying semantic class label of image I. Here we do not explicitly assume any specific model for X, different application-specific features can be considered within our framework. For example, these could be the transform coefficients of

[1]Part of the work is this chapter was published in [88]

the image in DCT domain [81], pixels in image domain [30], color histograms [45], Gabor textures [41], etc. Image retrieval is performed by finding the K most similar objects to the given query feature $\mathcal{Q}$ through a mapping $\mathcal{G}$:

$$\mathcal{G} : \mathcal{Q} \rightarrow \mathcal{S} = \{I_1^{\mathcal{Q}}, I_2^{\mathcal{Q}}, ..., I_K^{\mathcal{Q}}, \} \subset \mathcal{D} \tag{4.1}$$

The probability of retrieval error $\Pr(\mathcal{G}(X) \neq Y)$ is the probability that the retrieval system is provided a feature vector drawn from class Y and it returns images from classes other than Y. $\mathcal{G}(X)$ is the similarity function that the retrieval system employed to find underlying class label of observation X. Suppose we have M classes in the database, content-based image retrieval can be naturally posed as an M-ary statistical classification problem, where the optimal mapping is to select as classification label for observation X the label y^* having maximum a posteriori probability $P(y^*|X)$:

$$\mathcal{G}^*(X) = \arg\max_{y_i} \Pr(y_i|X) \quad i = 1, 2, ..., M \tag{4.2}$$

This is the well known Bayes classifier [25]. The retrieval functions of most existing prototype CBIR systems can be thought of as special cases of this Bayesian classifier, where certain assumptions have been made about the class distributions. For example, the Mahalanobis distance is used for the Nearest Neighbor Classifier chosen in QBIC [45]. The nearest-neighbor classification employed by [42] [41] is a special case of a Bayes classifier assuming Gaussian distributions with identity

covariance. Vasconcelos and Lippman provided an unifying view of image similarity in [82]. Refer to Appendix A for an evaluation of the similarity functions currently used by most CBIR systems.

The Bayes formulation of content-based image retrieval is motivated by the observation that most CBIR system focus on extracting meaningful features to represent the image contents, but fail to make good use of this information for classification. Simple nearest-neighbor classifiers are often used without taking into account the data distributions, which may reduce the benefits of selecting good features for extraction.

In a distributed image retrieval/classification system such as that shown in Figure 4.1, the client acquires the data X, encodes it and transmits to the server. The server decodes the data and selects the class label for the reconstructed data $\hat{X}$. The classifier $\mathcal{G}$ does not have access to the original data X and instead operates on compressed data.

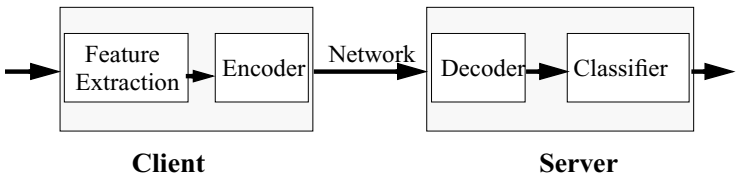


Figure 4.1: Source coding for distributed image classification.

The problem we address is that of designing good source codes (which consists of encoder α and decoder β), such that after applying the classifier to the reconstructed

data $\hat{X} = \beta(\alpha(X))$, the classification error due to data compression is minimized for a given rate constraint R_b:

$$\min_{\alpha^*,\beta^*} P_e(\hat{X}) \quad s.t \quad R \leq R_b \tag{4.3}$$

From (4.2) we see that the M a posteriori probability functions $\{\Pr(y_i|X),$ $i = 1, ..., M\}$ carry sufficient information for classification. An unconstrained solution would be to classify the input at the client using uncompressed data X, and send only the classification label to the server. However there are situations where client-based classification is not an option, including:

- cases where the client has limited computational and memory resources and is unable to support a complex classifier [72][46](e.g., Image database, Speech recognition engine, etc.)

- cases without those complexity constraints but such that the database used for information retrieval is constantly changing. In this situation it is desirable to keep the database in a central location, without having to also maintain up to data version at every single client.

- sensor network scenarios where the data acquired from all the sensors are fused in a central node in order to estimate the required parameter about target, the classifier (estimator) has to be kept in the central node [84].

Thus, in any of the above scenarios it would be preferable to compress the feature data so that recognition/classification can be remotely handled.

Several authors have studied the problem of optimal quantization for binary signal detection [54] [52] [38], and have shown that for several choices of performance criteria, likelihood ratio quantizers (LRQ's) are optimal.

Definition 1 *Given a set of all vectors* $t = (t_1, ..., t_{q-1}) \in [0, \infty]^{q-1}$, *satisfying* $0 \leq t_1 \leq ... \leq t_{q-1} \leq \infty$. *A likelihood ratio quantizer is the partition* $\Delta = \{\delta_i, i = 1, ..., q\}$ *of the sample space* χ, *with each domain* δ_i *defined by*

$$\delta_i = \{x | t_{i-1} < \ell(x) \leq t_i\} \tag{4.4}$$

where $\ell(x)$ is the Likelihood Ratio $\frac{d\mathcal{P}_1}{d\mathcal{P}_0}(x)$.

Tsitsiklis studied the extremal properties of Likelihood-Ratio Quantizers in [77] and established optimal properties of LRQ for quantization problems involving the maximization of an Ali-Silvey distance [1] and the Neyman-Pearson variant of the decentralized detection.

In spite of the fact that LRQ is optimal, such a solution is often impractical for real applications. In a client-server communication model, employing a likelihood ratio quantizer is equivalent to keeping a full-fledged classifier at the client, which is what we try to avoid in the first place. Furthermore, it is unknown from the literature how to generalize the likelihood ratio quantizer to the cases where $M > 2$ hypothesis need to be classified. In this chapter, based on the idea of maximizing

the class separability, we show how to design a transform-based encoding system for distributed classification applications. We focus on transform coding due to its simplicity, flexibility, and good practical performance.

Traditional transform coding is designed to achieve minimum reconstruction error for a given bit rate, where the distortion is often evaluated by in terms of mean square error (MSE), i.e., $E(d(X, \hat{X}))$. It has been shown that a good approach for this purpose is to use an optimal decorrelation transform (Karhunen Loève Transform) followed by quantization and entropy coding [28]. The main principle is to compact most of the signal energy in fewest dimensions (energy compaction), and then allocate bits based on signal variances in each of the dimensions. However, for quantization in signal classification applications, it is the class separability information that we shall preserve, not the representation precision measured by MSE.

Optimal transform coding of the images for joint classification/reconstruction was considered in [35]. It was shown that assuming X is a stationary, periodic process under two hypothesis H_0 and H_1, the Discrete Cosine Transform (DCT) would produce uncorrelated components and the optimal transform for classification/reconstruction would be the DCT followed by a diagonal transform, which can be absorbed into the scalar quantization. In their latest work [36], the same authors showed that when a cost function combining Chernoff distance (one special case of Ali-Silvey distance) and MSE is used, the Karhunen-Loève transform (KLT) is the optimal transform, under the following assumptions:

1. High rate quantization.

2. The same unitary transform C is a decorrelating transform for data generated under either hypothesis, $H_i, i = 0, 1$.

3. Under each hypothesis $H_i, i = 0, 1$, the probability density function $f_i(X)$ is a mixture of N_i Gaussian with a common mean (assumed to be zero):

$$H_i : X \sim f_i = \sum_{j=1}^{N_i} \alpha_{ij} \mathcal{N}(0, \Sigma_{ij}), \quad i = 0, 1 \tag{4.5}$$

   where $\alpha_{ij} > 0$ and $\sum_{j=1}^{N_i} \alpha_{ij} = 1$. Each covariance matrix Σ_{ij} is diagonal, hence $X_k, 1 \leq k \leq N$, are uncorrelated under $H_i, i = 0, 1$.

4. A Unitary transform is used.

Although Assumption 2 approximately holds for natural images, it might not hold anymore when we deal with image *features* that are extracted from images for the purpose of content-based image retrieval. Furthermore, Assumption 3 can not be justified in general: neighboring pixels in natural images are highly correlated; the features extracted for content-based retrieval are not necessarily going to be uncorrelated.

In this chapter, we design transform coding approach optimized *only* for the classification problem. Assuming that the distance between classes is measured by the class separability criterion namely the scatter measure, which is based on the

second-order statistics, i.e., means and covariances, of the empirical data, we present the following contributions:

- We demonstrate that the LDA is the optimal linear transform;
- We design an efficient bit allocation algorithm to split the rate budget among the transform components, based on classification criteria;
- We derive the relationship between the proposed transform coding and the optimal solution, likelihood ratio quantization [54] [52] [38], for Gaussian Markov sources;
- We extend the high rate analysis of Likelihood Ratio Quantization [53] for optimal bit allocation in the proposed transform coding for Gaussian Markov sources.

This chapter is organized as follows. In Section 4.2 we define the problem of designing transform coding for pure classification application. Section 4.3 presents the optimal transform and proposes a greedy bit allocation scheme. In Section 4.4 we demonstrate the optimality of the proposed scheme for Gaussian Markov sources by deriving the relation to the LRQ, and extending the high rate analysis by Poor et. al [53] for optimal bit allocation under the proposed transform coding scheme. Finally, Section 4.5 provides simulation results using both a real texture classification example and synthesized Gaussian Markov sources.

4.2 Problem definition

4.2.1 The standard model of transform coding

Let us define the system under study. The "information source" is denoted by a real column vector $X \in R^N$, or a sequence of such vectors. Each source vector is assumed to be a realization of M possible random processes with distributions $f_i, i = 0, 1, \cdots, M-1$. In general, these distributions need not to be known as closed form models and could be empirical, i.e., they could be characterized by a set of sample vectors.

Transform coding comprises three relatively simple steps: Computation of a linear transformation, scalar quantization of the transform coefficients, and entropy coding. Each of the steps operates independently. The structure of transform coding is shown in Fig. 4.2. The input data $X \in R^N$ is transformed to $\{\bar{x}_1, \bar{x}_2, ..., \bar{x}_N\}$ by a linear transform $T \in R^{N \times N}$. The resulting transform coefficients are separately quantized by N scalar quantizers α_k, $k = 1, ..., N$. The quantization indices $\{i_1, i_2, ..., i_N\}$ are represented using an entropy coder by γ. At the decoder, entropy decoding γ^{-1} produces the quantization indices, β_k reconstructs the quantized transform coefficient $\hat{\bar{x}}_k$. And finally the synthesis transform U (usually T^{-1}) is applied to obtained reconstructed data $\hat{X}$.

Without loss of generality, we assume a linear transform T followed by a bank of uniform quantizers $\{\Delta_i\}$, and independent entropy coding of quantization indices.

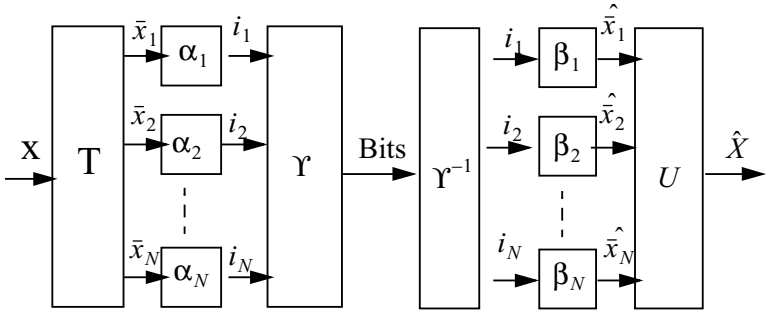


Figure 4.2: Structure of a standard transform coding system.

Our goal is to find the optimal linear transform T^* and the set of quantization step-sizes $\{\Delta_i^*\}$, such that the probability of classification error based on the compressed data $P_e(\hat{X})$ is minimized for a given rate constraint R_b:

$$(T^*, \Delta_i^*) = \arg \min_{T, \{\Delta_i\}} P_e(\hat{X}) \quad s.t. \quad R \leq R_b \tag{4.6}$$

4.2.2 Quantization and cost criteria

Information loss in transform coding comes from quantization. The quantizer α introduces a partition $\mathcal{A}$ which consists of a set of non-overlapping cells $\mathcal{A} = \{a_i, i = 1, ..., q\}$. Each cell a_i is associated with a reconstruction value v_i. Any sample X falling in cell a_i after quantization $\alpha(X) \in a_i$ is reconstructed as $\hat{X} = v_i$ and thus will be classified to the same classification label as v_i.

Quantization performance is measured by the information loss it introduces and the entropy rate of the quantization indices. For reproduction purposes, MSE is typically used to describe information loss. Alternate performance criteria must be

considered when the reconstructed data is to be used for classification. Probability of misclassification P_e is certainly the most desirable criterion. However, in most practical applications, the probability of misclassification may not be easy to compute and manipulate and thus various alternative criteria and measures have been proposed and used in practice. In particular, researchers have been interested in a measure of the *class overlap* or alternatively *class separability*.

There are two types of criteria which are frequently used in practice to evaluate class separability. One is based on a family of functions which give upper bounds of the Bayes classification error, such as the Chernoff Bound [76]:

$$P_e \leq \pi_0^{1-s} \pi_1^s e^{-D_s(f_0, f_1)} \tag{4.7}$$

where π_i is the a priori probability, f_i the distribution function of class i, and $0 \leq s \leq 1$ is the Chernoff exponent. D_s is the chernoff distance computed as:

$$D_s(f_0, f_1) = \ln \int f_0(x) \left(\frac{f_1(x)}{f_0(x)} \right)^s . \tag{4.8}$$

In this work we consider a class separability criterion called scatter measure [25]. It is based on a second-order measure of quality that is defined completely in terms of second-order probabilistic parameters, i.e., means and covariances, of the empirical data. We consider the within-class scatter $\mathbf{S_w}$, which is the scatter of samples around their respective class mean:

$$\begin{aligned} \mathbf{S_w} &= \sum_{i=1}^{M} \pi_i E\{(X - M_i)(X - M_i)^t | y_i\} \\ &= \sum_{i=1}^{M} \pi_i \mathbf{\Sigma_i}, \end{aligned} \tag{4.9}$$

and the between-class scatter $\mathbf{S_b}$, which is the scatter of the expected vectors around the mixture mean [25]:

$$\mathbf{S_b} = \sum_{i=1}^{M} \pi_i (M_i - M)(M_i - M)^t, \tag{4.10}$$

where $\mathbf{\Sigma_i}$ and M_i are the covariance matrix and mean vector for the ith class, respectively, π_i is the a priori probability of class i and M is the overall mean vector. The scatter ratio criterion is computed as:

$$\mathbf{S} = \mathbf{S_w}^{-1} \mathbf{S_b} \tag{4.11}$$

Figure 4.3 shows a visualization of the meaning of the scatter measure. The intuition here is that to have good class separation, we would like different classes to be as far apart from each other as possible ($\mathbf{S_b}$ is large) and, at the same time, samples belonging to the same class to be as closely clustered together as possible ($\mathbf{S_w}$ is small).

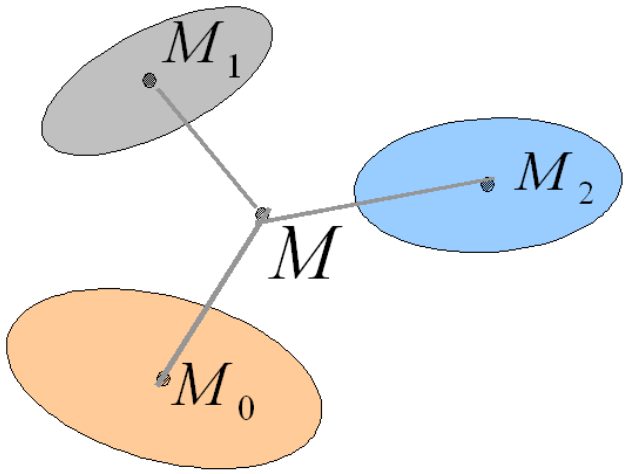


Figure 4.3: A simple example showing the visualization of the scatter metric.

4.2.3 Linear discriminant transform

A signal given as an N dimensional vector $X \in R^N$ is implicitly represented with respect to the canonical basis for R^N. A linear invertible transform $\mathbf{T}$ basically changes the basis through rotation and scaling. A change of basis does not alter the information in the signal. The reason that transforms are useful for compression is that simple coding, e.g., scalar quantization and entropy coding, may be more effective in the transform domain than in the original domain [28].

Typically, the transform coefficients are separately quantized with uniform quantizers. The induced partitioning consists of parallelogram-shaped cells aligned with the transform bases. It was shown in [28] that the optimal transform for signal representation is the Karhunen-Loève transform. Let $\mathbf{S_m}$ denote the covariance matrix of the information source (which is a mixed distribution from M possible random processes). The KLT transform is computed as:

$$\mathbf{T}^* = \max_{\mathbf{T} \in R^{N \times N}} |\mathbf{T^t S_m T}| \tag{4.12}$$

where $|\cdot|$ denotes the determinant of a matrix. We can see that the KLT computes the linear transform that provides optimal energy compaction.

Following the same philosophy as of Karhunen-Loève transform, which is designed to preserve the signal energy (measured by covariance), we propose that the optimal transform in terms of preserving the class separability (measured by scatter ratio) can be computed as:

$$\mathbf{A}^* = \max_{\mathbf{A} \in R^{N \times L}} \frac{\mathbf{A}^t \mathbf{S_b} \mathbf{A}}{\mathbf{A^t S_w A}}, \tag{4.13}$$

which is the *linear discriminant analysis* (LDA) matrix proposed in pattern recognition application [25].

The idea of exploiting the LDA matrix for transform coding is inspired by the optimality of LRQ. LRQ leads to a space partition where partition boundaries are aligned with classification boundaries. An LDA classifier generates a hyperplane that provides optimal separation between two classes. If the decision boundaries are to be approximated by linear hyperplanes, LDA will be the optimal transform.

LDA has been proposed in pattern recognition applications to avoid the curse-of-dimensionality problem. Classification procedures that are analytically or computationally manageable in low-dimensional spaces can become totally impractical when dimensionality reaches 50 or higher.

It has been shown in [22] that the matrix $\mathbf{A}$ which maximizes the scatter ratio shown in (4.13) must satisfy:

$$\mathbf{S_b A} = \lambda \mathbf{S_w A}, \tag{4.14}$$

which is a generalized eigensystem problem. Assuming that $\mathbf{S_w}$ is non-singular, we have:

$$\mathbf{S_w}^{-1}\mathbf{S_b A} = \lambda \mathbf{A}. \tag{4.15}$$

This is a standard eigensystem problem. Thus the transform $\mathbf{A}^*$ that is optimal in terms of "compacting" the class discrimination information measured by scatter $\mathbf{S_w}^{-1}\mathbf{S_b}$ is a matrix whose columns consist of the eigenvectors of the matrix $\mathbf{S_w}^{-1}\mathbf{S_b}$.

To visualize the fundamental difference between signal representation and signal classification, let us look at a simple example taken from [25]. Figure 4.4 shows the distributions of height and weight for males and females. The principal axis ϕ_1 correspond to the basis vector of the KLT transform with largest eigenvalue. It captures the most energy of the mixture distributions. However, if we project the distributions to ϕ_1, the marginal density functions of two classes (male and female) are heavily overlapped. On the other hand, classes are well separated along direction ϕ_2. In this sense ϕ_2 preserves more classification information and should be treated as being more important than ϕ_1. However, traditional transform coding will allocate more bits to ϕ_1 than to ϕ_2, due to the larger signal variance along ϕ_1. This

is good for representing the signal but bad for preserving classification information. Thus, new transform and bit allocation schemes are needed to better preserve class discrimination information.

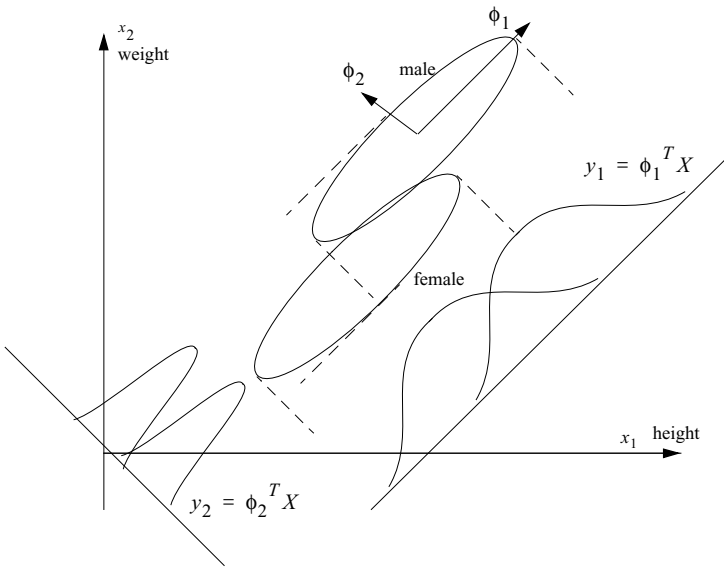


Figure 4.4: An example comparing transforms for signal classification and signal representation [25].

4.3 Proposed scheme

We propose to an approach based on LDA for transform coding in classification applications. Since the scatter matrix $\mathbf{S_w}^{-1}\mathbf{S_b}$ is not necessarily symmetric, the eigensystem computation in (4.14) could be unstable. We use the method proposed in [74] for the eigensystem computation.

Suppose the diagonal factorization of the within-class scatter is $\mathbf{S_w} = \mathbf{H\Lambda H}^t$ ($\mathbf{S_w}$ is symmetrical). Compute a new symmetric matrix $\mathbf{V}$ by multiplying the between-class scatter matrix $\mathbf{S_b}$ by $\mathbf{H\Lambda}^{-\frac{1}{2}}$ and its transpose $(\mathbf{H\Lambda}^{-\frac{1}{2}})^t$:

$$\mathbf{V} = (\mathbf{H\Lambda}^{-\frac{1}{2}})^t \mathbf{S_b} (\mathbf{H\Lambda}^{-\frac{1}{2}}) \tag{4.16}$$

Since $\mathbf{V}$ is also symmetric, it again can be factored in terms involving a unitary matrix $\mathbf{U}$ and a diagonal matrix $\mathbf{E}$: $\mathbf{V} = \mathbf{UEU}^t$. Without loss of generality, we assume that $\mathbf{E}$ is sorted in descending order of eigenvalues. We have:

$$\mathbf{S_b} = \mathbf{H\Lambda}^{\frac{1}{2}} \mathbf{UEU}^t \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{H}^t \tag{4.17}$$

then the scatter matrix can be represented as:

$$\begin{aligned} \mathbf{S_w}^{-1}\mathbf{S_b} &= \mathbf{H\Lambda}^{-1}\mathbf{H}^t\mathbf{H\Lambda}^{\frac{1}{2}}\mathbf{UEU}^t\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{H}^t \\ &= \mathbf{H\Lambda}^{-\frac{1}{2}}\mathbf{UEU}^t\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{H}^t \\ &= \mathbf{SES}^{-1} \end{aligned} \tag{4.18}$$

where $\mathbf{S} = \mathbf{H\Lambda}^{-\frac{1}{2}}\mathbf{U}$. Thus $\mathbf{S}$ is the eigenmatrix of $\mathbf{S_w}^{-1}\mathbf{S_b}$ and $\mathbf{E}$ contains the eigenvalues.

The key idea of linear discriminant analysis is that classes become well separated after projecting to the most discriminant directions (the bases of $\mathbf{S}$ that have

larger eigen values), and the class separability information is "compacted" in fewest dimensions in the transform domain.

As an example, we show in Figure 4.5 the histograms of projected data of the wavelet feature vectors for two Brodatz textures [7] $D1$ and $D2$. The wavelet feature vector is computed as the norm in subbands of a Dyadic wavelet decomposition of the image. Details about wavelet feature extraction can be found in Section 2.4. The example in Figure 4.5 uses as a feature vector the norm in the LL and LH bands of third level decomposition.

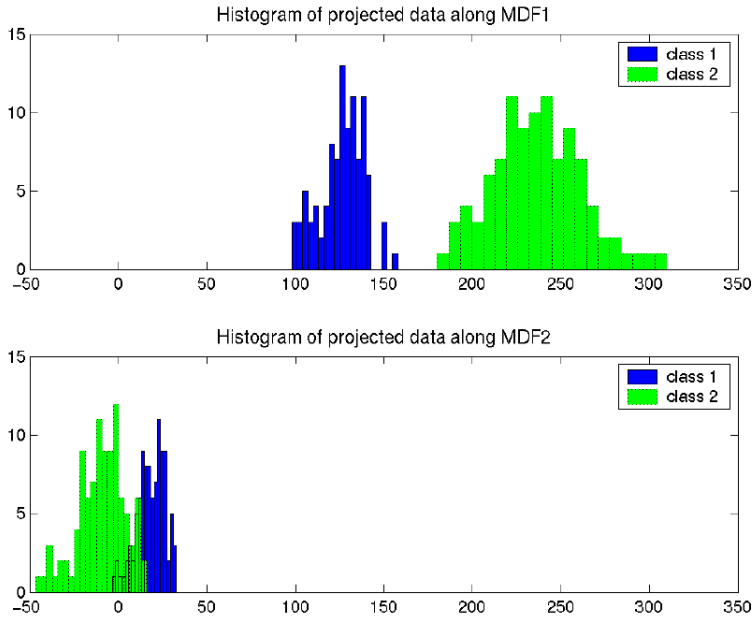


Figure 4.5: Histograms of features projected onto the eigensystem of the scatter matrix.

We can see that after projecting onto the eigensystem of the scatter matrix, most of the class discrimination information is preserved in the direction which

corresponds to the eigenvector with the largest eigenvalue (the top histogram in Figure 4.5). Quantization in the transform domain will induce partitions that align with the directions of class separation. Keeping this in mind, we would expect that the representation entropy (representing class separability information) in linear discriminant transform domain can be reduced as compared to the original domain. This nice property can be used to enable efficient quantization.

After transformation, we need a bit allocation scheme to decide the quantization stepsizes $\{\Delta_i\}$ for each feature dimension i, such that the discrimination information between classes is best preserved for the given rate constraint. Different from traditional bit allocation, where total distortion is obtained as the sum of distortion in each dimension, here the classification performance depends on all dimensions, and it is not possible to create separable metric to capture this performance. Thus it is not possible to apply standard Lagrangian techniques to allocate rate to each dimension in order to maximize overall classification performance. To overcome this problem, we propose a greedy bit allocation scheme where we start by using the finest quantization for all dimensions and at each iteration we choose the dimension to which a coarser quantization shall be applied such that the magnitude of the ratio of increase in average class entropy to decrease in entropy rate is minimized. This process repeats until the rate constraint is satisfied. Following is a formal description of the proposed bit allocation scheme.

Let $\alpha^t = \{\Delta_i^t, i = 1, ..., N\}$ denote the quantizer at iteration t, which applies quantization stepsize Δ_i^t to dimension i, and let $\mathcal{A}^t$ be the resulting partition of the sample space using quantizer α^t. At iteration $t+1$, let $\alpha_k^{t+1} = \{\Delta_1^t, \Delta_2^t, ..., \Delta_k^{t+1}, ..., \Delta_N^t\}$ be the quantizer which has the same stepsizes as quantizer α^t except for the $k-$th component, where a larger stepsize than at iteration t is applied ($\Delta_k^{t+1} > \Delta_k^t$). We denote the resulting partition induced by quantizer α_k^{t+1} as $\mathcal{A}_k^{t+1}$. Note that at iteration $t+1$, there are N alternative partitions $\{\mathcal{A}_k^{t+1}, k = 1, ..., N\}$ to choose from. The goal of proposed greedy bit allocation scheme is to decide which partition to choose at each iteration t.

A partition $\mathcal{A}$ consists of many individual cells denoted as a. The samples falling in the same cell will be reconstructed as the same value (usually the centroid of the cell), and thus will be classified to the same class by any classifier. We introduce a misclassification cost which we call *class entropy*, to be used in the context of quantization.

Suppose the total number of classes is C and let $\{p_1(a), p_2(a), \cdots, p_C(a)\}$ denote the probabilities that vectors corresponding to each class fall within partition cell a. In practice given a training set, this can be computed as the ratio of the number of class i samples $n_i(a)$ and the total number of samples $n(a)$ in cell a. The class entropy in cell a is computed as:

$$H(a) = -\sum_{i=1}^{C} p_i(a) \times \log(p_i(a)), \tag{4.19}$$

and the average class entropy for a partition $\mathcal{A}$ is computed as:

$$H(\mathcal{A}) = \sum_{a \in \mathcal{A}} \Pr(a) \times H(a). \tag{4.20}$$

The *average entropy rate* is computed as the sum of the Shannon entropy [29] of the quantizer output for each vector component. Let $\bar{x}_i$ be the coefficient and Δ_i be the quantization stepsize for the $i-$th component, respectively. $P_k = P(k\Delta_i < \bar{x}_i \leq (k+1)\Delta_i)$ is the probability that the source sample, $\bar{X} = \{\bar{x}_i, i = 1, \cdots, N\}$, lies in the quantization interval $(k\Delta_i, (k+1)\Delta_i]$. Note that in order to compute the Shannon entropy, we consider all source samples and do not separate them by class. Then we have the Shannon entropy rate for $i-$th feature vector component computed as:

$$R_i = -\sum_k P_k \log P_k \tag{4.21}$$

and the average entropy rate is the sum over all feature vector components:

$$R = \sum_i R_i \tag{4.22}$$

We need to find at iteration $t+1$ the component i^* along which to apply a coarser quantization than at iteration t, such that the slope $\rho_i = \frac{H(\mathcal{A}_i^{t+1}) - H(\mathcal{A}^t)}{R(\mathcal{A}^t) - R(\mathcal{A}_i^{t+1})}$ is minimized. $R(\mathcal{A})$ is the entropy rate associated with partition $\mathcal{A}$ and is computed using (4.22). This process is continued until the rate constraint is satisfied, i.e., $R(\mathcal{A}) \leq R_b$.

Let there be Q admissible quantization stepsizes $\{\delta_1, ..., \delta_Q\}$, where $\delta_1 < \delta_2 < ... < \delta_Q$. We associate an index i with each entry δ_i. Thus at iteration t, if we have quantizer $\alpha^t = \{1, 3, ..., 1\}$, we mean that the corresponding stepsizes are: δ_1 for x_1, δ_3 for x_2,..., and δ_1 for x_N. Following is the proposed greedy bit allocation algorithm:

Proposed Algorithm : *Greedy bit allocation for quantization in signal classification*

Step 0: *Initialize* $t = 0$, $\alpha^0 = \{k_1^0, k_2^0, ..., k_N^0\} = \{1, 1, ..., 1\}$. *Compute* $H(\mathcal{A}^0)$ *and* $R(\mathcal{A}^0)$.

Step 1: *If* $R(\mathcal{A}^t) \leq R_b$, *stop; Else* $t = t + 1$, $i = 0$, *goto* **Step 2**.

Step 2: $i = i + 1$. *Let* $k_i^t = k_i^{t-1} + 1$ *and* $k_j^t = k_j^{t-1} \quad \forall j \neq i$. *The corresponding quantizer is* $\alpha_i^t = \{k_1^t, ..., k_i^t, ..., k_N^t\}$. *Compute* $H(\mathcal{A}_i^t)$ *and* $R(\mathcal{A}_i^t)$ *using (4.20) and (4.22). Then we have* $\rho_i = \frac{H(\mathcal{A}_i^t) - H(\mathcal{A}^{t-1})}{R(\mathcal{A}^{t-1}) - R(\mathcal{A}_i^t)}$. *If* $i < N$, *repeat* **Step 2**;

else, goto **Step 3**.

Step 3: *Choose* i^* *which has the minimum slope* ρ : $i^* = \min_{i \in \{1,2,...,N\}} \rho_i$. *Let* $\alpha^t = \alpha_{i^*}^t$, *and goto* **step 1**.

4.4 Relation to Likelihood Ratio Quantization (LRQ)

In this section we review the concept of Likelihood Ratio Quantization (LRQ) and show the optimality of LDA transform coding for a certain class of Gaussian distributions. We develop the asymptotic analysis of proposed coding scheme under the assumption of high rate quantization.

4.4.1 Review of Likelihood Ratio Quantization

Suppose we have two hypothesis H_1, H_2, and X is a random vector with probability distribution $\mathcal{P}_i(X)$ under hypothesis H_i. In classical signal detection theory, one observes one or more realizations of the random vector X and attempts to predict the hypothesis that generated the observation. However there are many applications where the observation must be quantized and the classification is constrained to operate on quantized data. For example, an imaging radar captures and transmits information to an user whose interest is the likelihood of presence of a particular target; Or consider decentralized detection applications where sensors obtain and transmit their observations to a central node that fuses all the information to make a final decision [75]. Compression is essential to reduce transmission rates. The problem is to find the optimal quantizer with respect to a performance measure

of interest. In this chapter, we focus on a performance measure called *Ali-Silvey distance* [1].

Let $f : [0, \infty) \to R$ be a continuous convex function satisfying:

$$\lim_{x \to \infty} \frac{f(x)}{x} = 0 \tag{4.23}$$

The Ali-Silvey distance of two probability measures $\mathcal{P}_0$, $\mathcal{P}_1$ is defined as [77]:

$$D_f(\mathcal{P}_0, \mathcal{P}_1) = \int f(\ell(x)) d\mathcal{P}_0 = E_0(f(\ell(x))) \tag{4.24}$$

where $\ell(x) = \frac{d\mathcal{P}_0}{d\mathcal{P}_1}$ is the likelihood ratio function. Let us use a quantizer α, and let the quantized data be a random variable with probability mass function $q(\alpha|H_i)$, under hypothesis H_i, $i = 0, 1$. The quality of quantizer α in terms of discriminating two hypothesis (H_0, H_1) can be measured by the Ali-Silvey distance of the distributions of the quantized random variables:

$$F(\alpha) = D_f(q(\alpha|H_0), q(\alpha|H_1)) \tag{4.25}$$

Several authors have studied this problem[54] [52] [38], and have shown that for several choices of performance criteria, LRQs are optimal. Tsitsiklis studied the extremal properties of LRQs in [77] and established optimality of LRQ for quantization problems involving the maximization of an Ali-Silvey distance. We now cite the results in [77] without proof.

Definition 2 *Given a set of all vectors* $t = (t_1, ..., t_{q-1}) \in [0, \infty)^{q-1}$, *satisfying* $0 \leq t_1 \leq ... \leq t_{q-1} \leq \infty$. *A Likelihood Ratio Quantizer is the partition* $\mathcal{A} = \{a_i, i = 1, ..., q\}$ *of the sample space, with each domain* a_i *defined by*

$$a_i = \{x | t_{i-1} < \ell(x) \leq t_i\} \tag{4.26}$$

Proposition 2 *The problem of finding a quantizer* α *that maximizes the Ali-Silvey measure* $F(\alpha)$ *has an optimal solution which is a Likelihood Ratio Quantizer (LRQ).*

Many criteria used to evaluate the distance between distributions are special cases of the Ali-Silvey distance. For example, we get the Kullback-Liebler divergence when $f(x) = -\log(x)$; the Chernoff distance:

$$D_s(\mathcal{P}_0, \mathcal{P}_1) = \ln \int d\mathcal{P}_0(x) \left(\frac{d\mathcal{P}_1(x)}{d\mathcal{P}_0(x)} \right)^s \tag{4.27}$$

is obtained by setting $f(x) = -x^s$ with $s \in (0, 1)$; the work by Picinbono and Duvaut [52] is also a special case of the Ali-Silvey distance with $f(x) = \frac{1}{x}$.

4.4.2 LDA transform coding and LRQ

Consider data source whose class conditional probability densities are Gaussian $\mathcal{P}_i(x) \sim \mathcal{N}(M_i, \mathbf{\Sigma_i})$, and assume their covariance matrix are identical for each hypothesis ($\mathbf{\Sigma_i} = \mathbf{\Sigma}, i = 0, 1$). We show that LDA transform followed by scalar quantization is the optimal quantization for signal classification.

Under above assumptions, the solution to the eigenvalue problem in (4.15) for LDA is the *Fisher's Linear Discriminant* [25]:

$$\mathbf{A}^* = {\mathbf{S_w}}^{-1}(M_0 - M_1) \tag{4.28}$$

Assuming equal a priori probabilities, the log Likelihood Ratio between the two hypothesis H_0 and H_1:

$$\log(\ell(x)) = -\frac{1}{2}(x - M_0)^t {\mathbf{\Sigma_0}}^{-1}(x - M_0) + \frac{1}{2}(x - M_1)^t {\mathbf{\Sigma_1}}^{-1}(x - M_1) \tag{4.29}$$

Example 1 The first case we consider is when the covariance matrix is an identity matrix, i.e., elements from different dimensions are uncorrelated. In this case we have $\mathbf{\Sigma_0} = \mathbf{\Sigma_1} = \sigma^2 I$. The log likelihood ratio function becomes a linear machine [22]:

$$\log(\ell(x)) = \mathbf{W}^t x + w_0 \tag{4.30}$$

where $\mathbf{W} = M_0 - M_1$ and $w_0 = -\frac{1}{2\sigma^2}(M_0^t M_0 - M_1^t M_1)$. The LRQ is equivalent to a linear transform W followed by scalar quantization. We now show that this linear transform W is the same as the LDA transform $\mathbf{A}^*$ in this case.

Here, the within-class scatter matrix $\mathbf{S_w} = \frac{1}{\sigma^2}\mathbf{I}$, and thus can be written as the LDA transform in (4.28) becomes:

$$\mathbf{A}^* = \frac{1}{\sigma^2}(M_0 - M_1) \tag{4.31}$$

We can clearly see that $\mathbf{W}$ and $\mathbf{A}^*$ are actually the same linear transform, therefore linear discriminant analysis results in the same decision boundaries as the optimal Likelihood Ratio test.

Now let us look at the principal axis of KLT, which are the eigenvectors of the average covariance matrix $\mathbf{S_m} = \mathbf{S_w} + \mathbf{S_b}$. From LDA we have:

$$\mathbf{S_w}^{-1}\mathbf{S_b}\mathbf{A} = \lambda\mathbf{A} \tag{4.32}$$

Since $\mathbf{S_w} = \sigma^2 I$, from (4.32) we easily see that $\mathbf{S_b}\mathbf{A} = \lambda\sigma^2\mathbf{A}$. After some simple manipulation we get:

$$\begin{aligned} \mathbf{S_m}\mathbf{A} &= (\mathbf{S_w} + \mathbf{S_b})\mathbf{A} \\ &= \sigma^2(\lambda + 1)\mathbf{A}, \end{aligned} \tag{4.33}$$

i.e, the covariance matrix of the mixed distribution shares the same eigensystem with the scatter ratio matrix $\mathbf{S_w}^{-1}\mathbf{S_b}$. Figure 4.6 shows an example of the direction vectors from Likelihood Ratio test (4.30), Linear Discriminant Analysis transform (4.31), and KLT (4.33) for 2-dimensional Gaussian random vectors with identity covariance matrices:

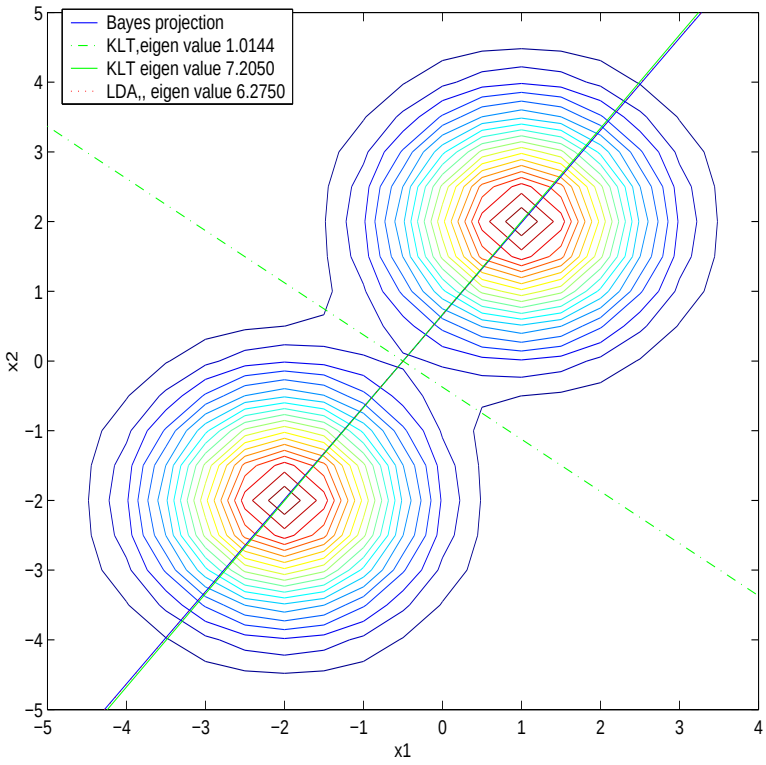


Figure 4.6: An example showing the direction vectors from Likelihood Ratio test, Linear Discriminant Analysis transform, and KLT for 2-dimensional Gaussian random vectors with identity covariance matrices. The mean vectors for the two hypothesis are $M_0 = [1 \;\; 2]^t$ and $M_1 = [3 \;\; 3]^t$.

$$\mathbf{\Sigma_0} = \mathbf{\Sigma_1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{4.34}$$

The mean vectors for the two hypothesis in this example $M_0 = [1 \;\; 2]^t$ and $M_1 = [3 \;\; 3]^t$.

Example 2 The second case we consider is that when the two hypothesis have same covariance matrix, but there exists correlation between elements of different dimensions. Decision boundaries from Likelihood Ratio test are represented by a linear transform $\mathbf{W} = \mathbf{\Sigma}^{-1}(M_0 - M_1)$, which is the same as the LDA transform $\mathbf{A}^*$.

We shall show that the principal axis ϕ_1 from KLT is different from $\mathbf{W}$ and $\mathbf{A}^*$ for this case, and that as the correlation increases, ϕ_1 becomes more deviated from $\mathbf{W}$.

Using the diagonal factorization of the scatter matrix $\mathbf{S_w}$ and $\mathbf{S_b}$ in (4.16) and (4.17), the eigensystem of the the mixture covariance matrix can be computed as:

$$\begin{aligned} \mathbf{S_m} &= \mathbf{H\Lambda H}^t + \mathbf{H\Lambda}^{\frac{1}{2}}\mathbf{UEU}^t\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{H}^t \\ &= \mathbf{H\Lambda}^{\frac{1}{2}}\mathbf{U}(\mathbf{E}+I)(\mathbf{H\Lambda}^{\frac{1}{2}}\mathbf{U})^t \end{aligned} \tag{4.35}$$

Compared to the factorization of the scatter matrix in (4.18), we see that the principal axis ϕ_1 from KLT, which corresponds to the first column of matrix $\mathbf{H\Lambda}^{\frac{1}{2}}\mathbf{U}$, will be generally different from the Fishers' linear discriminant which corresponds to the first column of matrix $\mathbf{H\Lambda}^{-\frac{1}{2}}\mathbf{U}$. While LDA penalizes the directions where the within class scatter is large, KLT emphasizes these direction since the within class scatter contributes to the over signal energy.

Example 3 We now show an example of the deviation of KLT from LDA and Likelihood Ratio test. In this example the covariance matrix is given by:

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma^2 & \rho \\ \rho & \sigma^2 \end{bmatrix} \tag{4.36}$$

where ρ is the correlation coefficient between dimensions. We plot in Figure 4.7 an example of the basis vectors $\mathbf{W}$, $\mathbf{A}^*$ and the KLT transform, with correlation coefficient $\rho = 0.2$. The mean vectors are $M_0 = [1 \quad 2]^t$ and $M_1 = [-2 \quad -2]^t$.

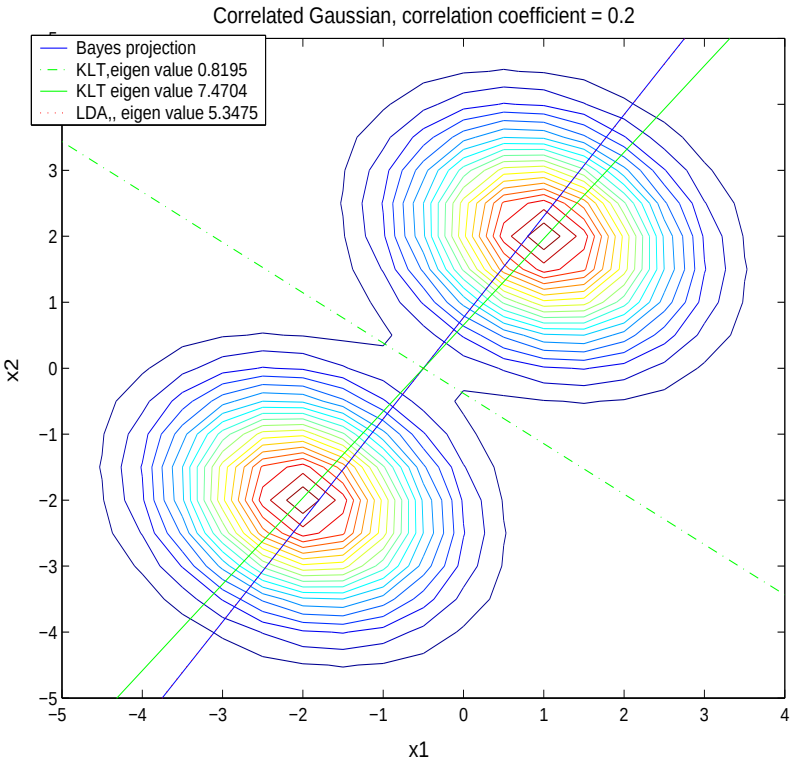


Figure 4.7: Example of basis vectors of Likelihood Ratio test, Linear Discriminant Analysis transform, and KLT for 2-dimensional correlated Gaussian source with $\rho = 0.2$. The mean vectors for the two hypothesis is $M_0 = [1\ \ 2]^t$ and $M_1 = [-2\ \ -2]^t$.

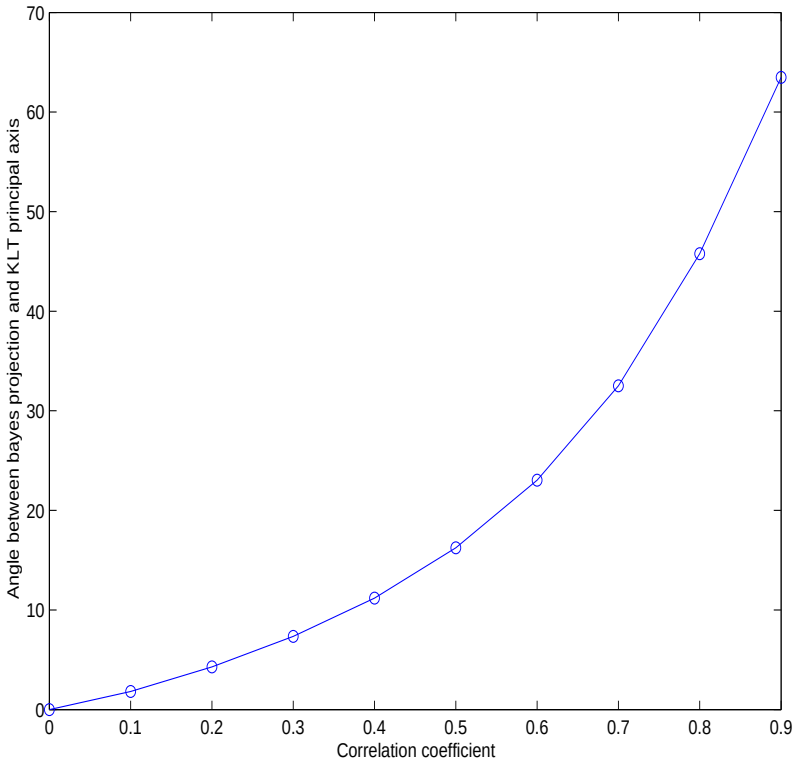


Figure 4.8: Angle between the principal axis ϕ_1 and W as a function of the correlation coefficient ρ for correlated Gaussian Sources.

We show in Figure 4.8 the angle between the principal axis ϕ_1 of KLT and the basis of W ($\mathbf{A}^*$) with largest eigenvalue, as a function of the correlation coefficient ρ. We can clearly see that the more correlated the dimensions are, the more deviated the principal axis of the KLT transform are from the optimal transform of Likelihood Ratio test. Linear Discriminant transform, on the other hand, is the same as the optimal transform.

4.4.3 Optimal bit allocation under high rate analysis

Assume that the two hypotheses are Gaussian distributed with identical covariance matrices $\mathbf{\Sigma_0} = \mathbf{\Sigma_1} = \mathbf{\Sigma}$. Then from Section 4.4.2 we know that LDA transform coding is optimal in terms of maximizing the distances of the quantized distributions. Let $\bar{X} = \mathbf{A}^t X$, where $\mathbf{A} = \mathbf{\Sigma}^{-1}(M_0 - M_1)$ is the LDA transform. The optimal quantization (LRQ) is given by scalar quantization in the transform domain (the likelihood ratio function is linear):

$$\mathcal{A}^* = \{b_{j-1} < \bar{X} \leq b_j, j = 1, ..., L\} \tag{4.37}$$

where $\{b_j, i = 1, ..., L\}$ is a set of real numbers defining the partition lines and L is the total number of quantization bins.

Consider the scatter ratio of the quantized data in the transform domain:

$$D(Q) = \frac{(E_1(Q) - E_0(Q))^2}{\frac{1}{2}(\sigma_1^2(Q) + \sigma_0^2(Q))} \tag{4.38}$$

where E_i is the expectation operator under hypothesis H_i, and σ_i^2 is the variance under hypotheses H_i. Since the covariance matrices are the same for H_0 and H_1, clearly the denominator in (4.38) will become $\sigma_0^2(Q)$. Then the scatter criterion is the same as the deflection criterion used by Picinbono in [52].

It was shown in [52] that the optimal partition, in terms of maximizing the deflection criterion $D(Q)$, is achieved by a "quantization by likelihood ratio" procedure,

where the partition boundaries are surfaces where the likelihood ratio $\ell(X) = \frac{d\mathcal{P}_0}{d\mathcal{P}_1}(X)$ is constant. This is exactly the LRQ concept we introduced in Section 4.4.1. The optimal reconstruction value v_i associated with partition cell a_i which maximizes the deflection criterion is given by $v_i = \frac{P_1(\bar{X} \in a_i)}{P_0(\bar{X} \in a_i)}$. The resulting deflection is given by:

$$D(Q) = \sum_{a_i \in \mathcal{A}} \frac{P_1^2(\bar{X} \in a_i)}{P_0(\bar{X} \in a_i)} \tag{4.39}$$

where $P_i(\bar{X} \in a_j)$ is the probability that a sample $\bar{X}$ (transform coefficients of sample X) from hypothesis H_i falls in cell a_j. We are interested in how the performance loss (loss in class discrimination information) behaves under fine quantization. By a simple manipulation, we get:

$$\begin{aligned} D(Q) &= \sum_{a_i \in \mathcal{A}} \frac{P_1(\bar{X} \in a_i)}{P_0(\bar{X} \in a_i)} \times P_1(\bar{X} \in a_i) \\ &= E_1(\ell^{-1}(\bar{X})) \end{aligned} \tag{4.40}$$

which is the generalized divergence $E\{f(\ell(\bar{X}))\}$ [53] with $f(\ell) = \ell^{-1}$(convex function) and the expectation is taken with respect to probability measure P_1. Then as the quantization stepsize becomes arbitrarily small $\Delta \to 0$, it was shown in [53] that the discrimination loss due to quantization is quadratic in Δ:

$$E\{f(\ell(\bar{X}))\} - E\{f(\ell(\hat{\bar{X}})\} \sim \frac{\Delta^2}{24} E\{\| \bigtriangledown \ell(\bar{X}) \|^2 f''(\bar{X})\} \tag{4.41}$$

Note that after projecting to the transform space $\bar{X} = \mathbf{A}^t X$, $\bar{X}$ is a normal random variable with distribution $N(\mathbf{A}^t M_i, \mathbf{A}^t \mathbf{\Sigma} \mathbf{A})$ under hypotheses H_i. Without loss of generality, we assume that $M_1 = m$ and $M_0 = 0$. Then in the transform domain, the distributions under the two hypotheses are $\mathcal{N}(0, r^2)$ and $\mathcal{N}(r^2, r^2)$ for H_0 and H_1, respectively, and $r^2 = m^t \mathbf{\Sigma}^{-1} m$. Under the assumption of fine quantization, we calculate the loss in discrimination information $D(\Delta)$ using (4.41) with likelihood function given as:

$$\ell(x) = e^{-x + \frac{1}{2} r^2} \tag{4.42}$$

After computation, we get:

$$D(\Delta) = \frac{e^{r^2}}{12} \times \Delta^2 \tag{4.43}$$

Recall that the entropy rate of a uniform scalar quantizer with small cell size Δ is approximately [29]:

$$H(\Delta) \cong h(Y) - \log(\Delta), \tag{4.44}$$

where $h(Y)$ is the differential entropy of random variable Y. In our case the random variable is a mixture of Gaussian $\pi_0\mathcal{N}(0, r^2) + \pi_1\mathcal{N}(r^2, r^2)$. Then the optimal quantization is to find stepsize Δ^* such that cost function $J = D(\Delta) + \mu H(\Delta)$ is minimized subject to rate constraint $H(\Delta) \leq R_b$. For a given multiplier μ, taking the differential of J and letting $\frac{\partial J}{\partial \Delta} = 0$, we get the optimal stepsize by:

$$\Delta^* = \sqrt{\frac{6\mu}{e^{r^2}}} \tag{4.45}$$

The optimal stepsize can be formed by adjusting μ such that rate constraint R_b is satisfied.

4.5 Experimental results

Applying the proposed LDA transform and greedy bit allocation to a four-class (D1, D2, D3, D4) Brodatz texture example, we show in Figure 4.9 the partition of the sample space introduced by the proposed scheme. As a comparison, under the same rate constraint, Figure 4.10 shows the partition resulting from applying bit allocation using MSE as the distortion metric in the original domain. We can clearly see that a "purer" partition is achieved at a given rate when the proposed transform coding was employed.

We show the performance of the proposed method using both a real texture classification example and synthesized Gaussian Markov sources. The feature vector for

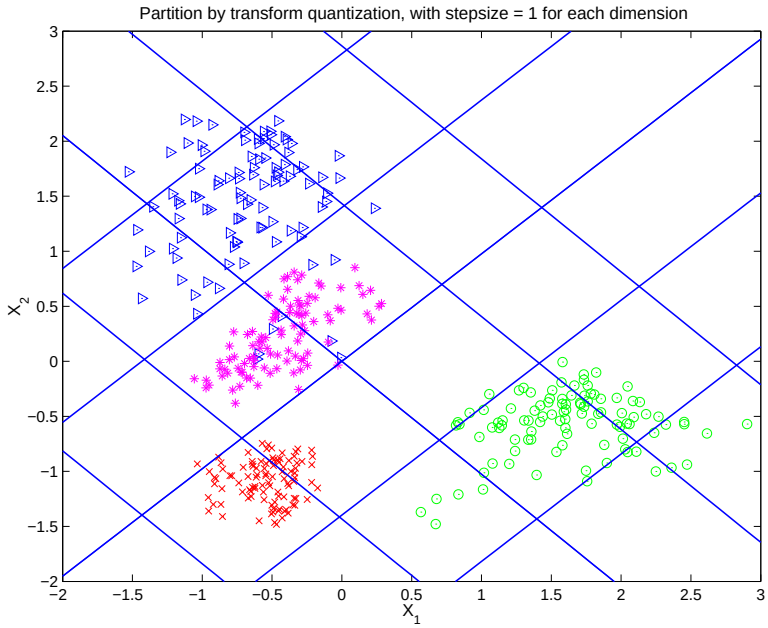


Figure 4.9: Partition induced by applying the proposed greedy bit allocation in the LDA transform domain.

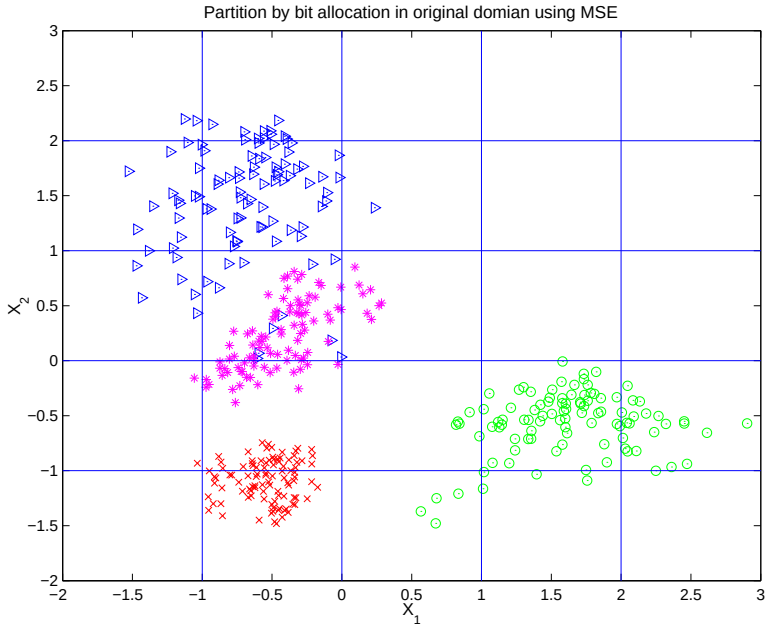


Figure 4.10: Partition induced by applying bit allocation using MSE as distortion metric in the original domain.

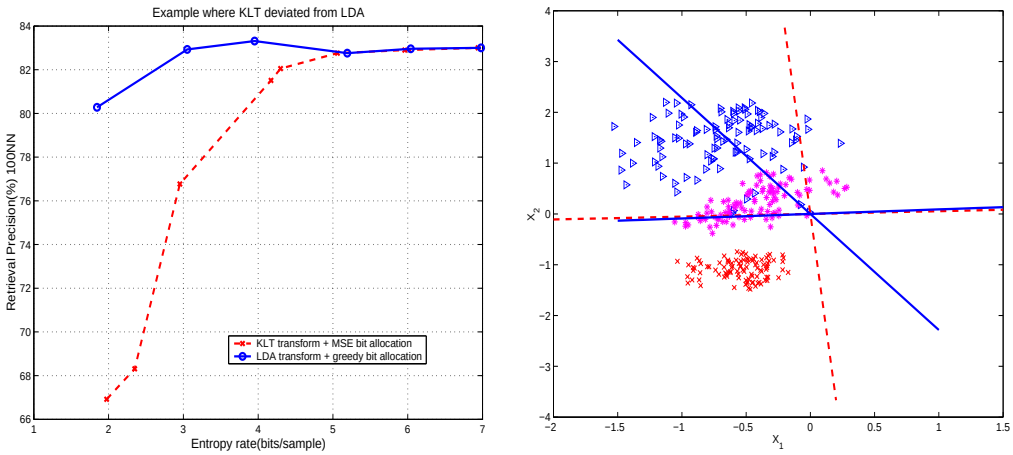
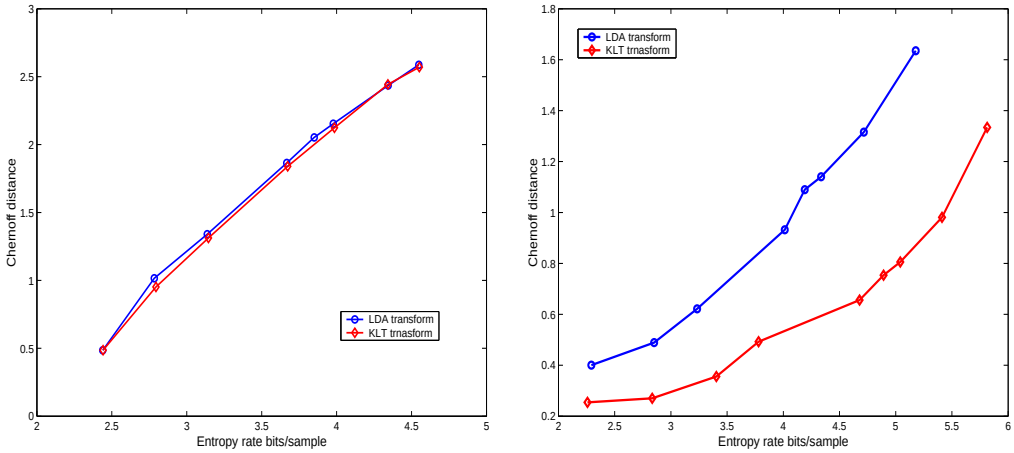


Figure 4.11: Example where KLT bases deviate from the bases of LDA. (a) Comparison of the classification performance based on two transform coding schemes. Solid circle: Proposed LDA transform coding with greedy bit allocation based on classification criteria; Dashed star: KLT transform coding with bit allocation based on Mean Square Error. Dashed square: (b)The bases of KLT and LDA for this example.

texture classification is computed as the energy in wavelet subbands. Details of how the features are obtained can be found in [86]. The three texture classes we used are: D1, D3, and D4 of Brodatz album [7]. We show in Figure 4.11(a) the classification performance comparing KLT transform coding and the proposed LDA transform coding. Bit allocation for KLT transform coding is performed based on MSE. We can clearly see that our proposed transform coding scheme achieves significantly better classification performance than the traditional KLT method, especially at low rates. At bit rate of 2 bits/sample, the proposed scheme achieves about 13% higher classification accuracy than KLT transform coding.

We show in Figure 4.12 the Chernoff distance (4.27) between the empirical probability mass functions $\hat{\mathcal{P}}_0$ and $\hat{\mathcal{P}}_1$ generated from the quantized data, using KLT



(a) (b)

Figure 4.12: Chernoff distance between the empirical distributions generated from quantized data for a synthesized Gaussian Markovian source with correlation coefficient (a) 0.2 (b) 0.8

transform coding and our proposed LDA transform coding for a synthesized Gaussian Markov source with covariance matrices given by:

$$\mathbf{\Sigma_0} = \mathbf{\Sigma_1} = \begin{bmatrix} \sigma^2 & \rho \\ \rho & \sigma^2 \end{bmatrix} \tag{4.46}$$

The Chernoff distance is the same as used by [36]. We can clearly see that the proposed algorithm provides much better separation between classes after compression than the KLT transform coding scheme, when compared at the same rates. The more deviated the KLT bases are from the bases of LDA, the larger the performance gap between the two schemes. When correlation exists between dimensions within each class, KLT is no longer the optimal transform to use for compression of the data in classification applications.

Chapter 5

A user preference information based kernel for SVM active learning in content-based image retrieval

5.1 Introduction

In this chapter we address relevance feedback which is a critical component for content-based retrieval systems[1]. A major difficulty associated with content-based image retrieval (CBIR) systems is the semantic gap between low-level features and high-level human concepts. While in some cases it is easy to map a high level concept (such as sunset) to low level features (color and shape), in some other cases such a mapping does not exist. Furthermore, different users may have different interpretations for the same image, and even the same user may have a different

[1]Part of the work in this chapter was published in [89]

interpretation or "feeling" about the same image depending on the applications and the context of usage.

Thus, substantial efforts have been devoted to designing techniques that place the "user in the loop", so that the system can learn the user's particular query preferences from time to time.

Relevance feedback allows the user to interactively tune the system to her own interest by indicating whether certain images provided by the system are relevant (positive samples) or not (negative samples)[2]. From the labeled examples, the system then learns how to tune the image selection parameters and returns a new set of similar images, iteratively repeating this process until the user is satisfied with the result. The construction of such a query updating scheme can be regarded as a machine learning task. However, relevance feedback learning is different from traditional machine learning in the sense that the class membership information is not available beforehand since it is both *user-dependent and time varying.* As compared to traditional machine learning, the major challenges associated with relevance feedback are:

1. The number of training samples is *small* relative to the dimensionality of the data.

2. There exists *no a priori model* that can provide a statistical characterization of positive and negative samples.

[2]Throughout this chapter we will consider relevance feedback provided by the user is restricted to two categories

3. Implicitly, the learning algorithm is expected to be *low complexity* since the user is interacting with the system in real time.

A majority of proposed approaches for relevance feedback in CBIR systems have been developed based on various forms of feature re-weighting [61][49], where the weights associated with each feature for a typical K-Nearest-Neighbor classifier are adjusted based on user feedback. The intuition is to emphasize those features that are best at discriminating between positive samples and negative ones by giving them a more significant weight in the distance computation.

A more systematic formulation of the relevance feedback problem can be achieved by setting up an optimization problem [33], where the goal is to find the optimal linear transformation that can map the feature space into a *new* space, having the property of clustering together positive examples, thus making it easier to separate them from negative ones.

More recently, several researchers have proposed the use of support vector machines (SVMs) as an active learning method for the relevance feedback problem in content-based retrieval [11] [32] [14] [31]. In [32], after each relevance feedback, a SVM is trained based on the resulting user-labeled images. Then the SVM classifier is applied to all other images, and produces as an output a class index as well as a distance score (a margin with respect to the classification boundary). This distance score can then be utilized for query refinement [61]. A one-class SVM scheme was developed in [14] that tries to fit a tight hyper-sphere in the non-linearly transformed

feature space (through a kernel) to include most positive samples. This scheme was named *one-class SVM* since it only employs the positive samples, while totally neglecting the information provided by the negative samples. As an extension, a biased SVM was proposed in [31] to incorporate negative information by employing a pair of hyper-spheres where the inner one includes most of the positive instances, while the outer one pushes out most of the negative samples. The unlabeled samples are then classified as relevant if they fall inside the inner sphere and non-relevant if they fall outside the outer sphere. We can see that a key underlying assumption made in these schemes is that the positive samples will actually be clustered together in the transformed space. Clearly, there is no guarantee that this will always hold true for an arbitrarily chosen kernel function. Whether clustering does occur (in which case these SVM techniques are more likely to be successful) depends on the distribution of positive and negative samples *and* on the choice of kernel function.

Using the training data set, a support vector machine finds the boundary which can best separate the classes. The goodness of this separation is measured in terms of the distance between training data vectors and the hyperplane that separates the classes. Each choice of kernel function leads to a "modified" distance in the feature space, and thus to a different ability to discriminate between classes for the corresponding SVM. Ideally, the kernel should be chosen so as to *best reflect the specific characteristics of the data being classified.* Thus, in the relevance feedback application, a good kernel would tend to minimize the distance between feature

vectors in the positive class, while increasing the distance between positive vectors and negative vectors. This chapter proposes an adaptive technique where the kernel function, and thus the resulting modified distance in the feature space, is updated with each successive user-provided relevance feedback.

In order to motivate our choice of kernel consider first a simple example where the feature space can be partitioned into non-overlapping regions, such that all the objects in any given region belong to a single class, i.e., they are *either* positive or negative vectors. If this is the case, then the obvious classification strategy is to, given a feature vector, determine to which region it belongs and produce as a classification output the class label of samples in that region. Note that in this case one could define a "modified distance" that would tend to cluster vectors of equal class. Trivially, the modified distance would be very small for two vectors located in regions belonging to the same class, and very large otherwise. This modified distance between two vectors would be based on the difference in class conditional probability in the regions each vector belongs to (in this trivial case, class conditional probabilities for a region are equal to 1 for one class and zero for the other).

Consider now extending this idea to cases where each region contains samples of both classes, i.e., a given region i will contain samples of class $+1$ with probability p^i and samples of class -1 with probability q^i, $p^i = 1 - q^i$. The question now is to define an appropriate distance between vectors, i.e., a distance that takes into account the different statistical characteristics of their respective regions. Essentially, two regions

can be considered more "similar", and thus the vectors in those regions "closer" in distance, if their statistical characteristics are similar. In this chapter we will show that this can be achieved with a distributional distance. Moreover, we will argue that our proposed method will lead to the optimal maximum likelihood solution as knowledge of the actual underlying probability model improves (i.e., as the feature space is partitioned into arbitrarily small "regions", as those we have been using in this example, and accurate models are known for all regions).

In this chapter we propose a kernel function based on the *information divergence* between the probabilities of positive and negative samples inferred from the user's preferences. Our proposed approach first infers a simple, non-parametric probabilistic model that can "explain" the training data and characterize the user's interest. The model is obtained by quantizing independently each component of each feature vector and forming histograms of the frequency of occurrence of each class in each quantization bin (multiple quantization bins per feature vector dimension). For each quantization bin we thus have an estimate of the marginal probability of positive and negative samples in the bin. The kernel obtained from these probabilistic models leads to a distance between vectors that increases if their respective probability models differ.

By incorporating the probabilistic information into the kernel function, we believe that some of the advantages of minimum Bayes error classifiers can be combined with those of trained SVM classifiers. As we will show later in this chapter, a SVM

classifier with Kullback-Leibler divergence based kernel can match the performance of an optimal maximum likelihood classifier in extreme cases where the true underlying probabilistic model is known. However, when the number of training samples is small relative to the dimensionality of the data, the performance of a maximum likelihood ratio classifier suffers, since the probabilistic models can not be effectively estimated under this scenario. SVMs are adequate tools to address these challenges since they do not suffer from Hughes phenomenon [9]. In effect, by decoupling probability model estimation from classification, we are able to select a model that is appropriate to the dimensionality of the feature vectors and the size of the training set (i.e., the total number of images on which feedback is provided), and then make use of it to train the SVM.

To the best of our knowledge this approach has not been used for relevance feedback in content-based image retrieval systems. Our work is inspired by [43], where a Kullback-Leibler (KL) divergence was used to derive the kernel function for SVM classification in speaker identification and image classification. Note that in [43] domain knowledge is available to model the data distributions that are used in computing the KL divergence. Statistical models such as Gaussian Mixture Models (GMM) or Hidden Markov Models (HMM) can very well model the data and the Expectation Maximization (EM) algorithm can be employed to learn and estimate the parameters. A more theoretical analysis of the use of Kullback-Leibler divergence to derive similarities between image classes, where each image class is

modeled as a Gaussian Mixture, can be found in [80]. Although the idea of applying the Kullback-Leibler divergence to SVM learning is not new, in this chapter we propose an extension of the framework in [43] for cases such as that of a relevance feedback application, where the data distribution model is not known *a priori* and has to be inferred from user feedback.

We present experiments based on a variety of image categories from the Corel set (from natural scenes such as Sunsets, Coasts, to human civilizations such as Mayan & Aztec, Land of the Pyramids). The results show that the new kernel achieves significantly higher (about 17%) retrieval accuracy than the RBF kernel (the best among standard SVM kernels), and even better than other kernel choices. Near 100% top-50 retrieval accuracy is achieved using the proposed kernel function after 6 relevance feedback iterations.

The chapter is organized as follows. In Section 5.2 we briefly review the concept of active learning for relevance feedback using support vector machines. Then we present our algorithm in Section 5.3. Experimental results are provided in Section 5.4, and Section 5.5 concludes our work.

5.2 Support vector machines for relevance feedback

In what follows we assume a feature extraction mechanism has been chosen so that each image (or in general each media object) is represented by a feature vector. A user is presented with a set of images as initial response to a query, and can then

attach relevance labels to each of the images. These labels are then returned to the CBIR system in order for a new set of images to be presented to the user.

More formally, suppose that we are given L observations, with each observation consisting of a pair: a feature vector $\mathbf{x}_i \in R^n$, $i = 1, ..., L$, and the associated semantic class label y_i, which can be either +1 (relevant) or -1 (irrelevant), based on the user feedback. $\mathbf{x}$ can be modeled as a random variable drawn from an unknown, user- and query-dependent distribution with probabilities $\{P(\mathbf{x}|y = +1), P(\mathbf{x}|y = -1)\}$. The goal of relevance feedback is to learn the mapping $g : \mathbf{x}_i \mapsto y_i$ based on the labeled training set.

In the ideal case where we are able to estimate $\{P(\mathbf{x}|y = +1), P(\mathbf{x}|y = -1)\}$, the optimal mapping is simply a maximum likelihood classifier (5.1):

$$g(\mathbf{x}) = \arg\max_i P(\mathbf{x}|y = i). \tag{5.1}$$

Estimation of the underlying probability model is difficult in relevance feedback applications, where we are confronted with a small sample problem [91], i.e., the number of available training samples is quite small relative to the dimensionality of the data. Thus it will be unrealistic to use traditional density estimation techniques in order to estimate $P(\mathbf{x}|y)$. One of the results of this work is that even simple statistical characterizations, which fall well short of providing a good estimate of $\{P(\mathbf{x}|y = +1), P(\mathbf{x}|y = -1)\}$, can in fact provide substantial performance gains

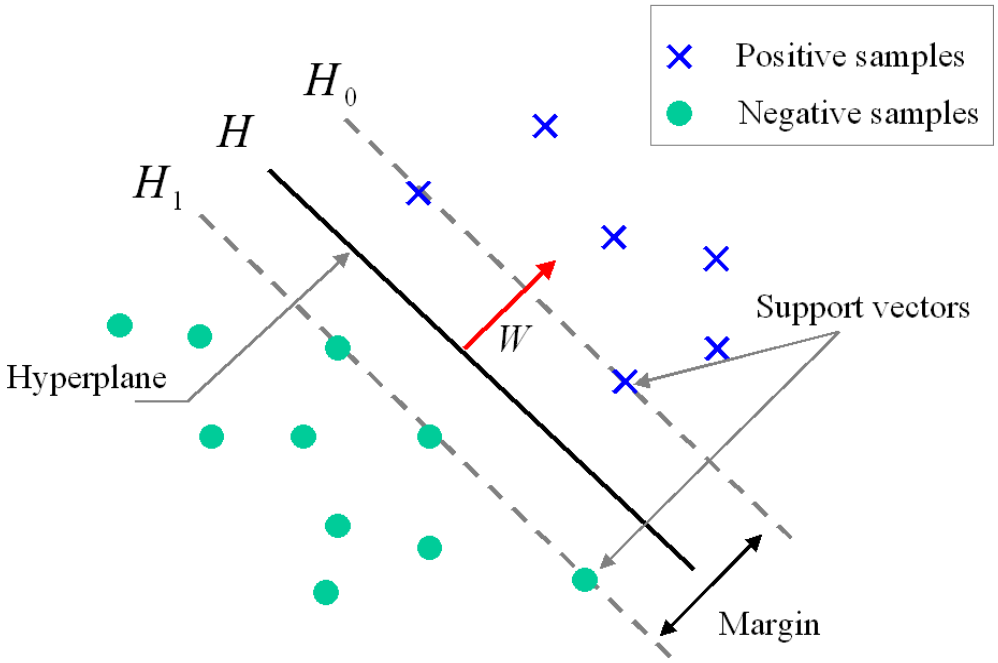


Figure 5.1: The optimal hyperplane is the one that separates the positive samples from the negative ones with maximum margin.

over methods that do not take into account the underlying statistical properties of the user query.

We start by providing a brief review of the basic concepts of Support Vector Machines, while referring to [78] and [9] for excellent tutorial descriptions on this subject. In the following, bold typeface will represent vector quantities and normal typeface will be used for vector components or scalars. Let $\{\mathbf{x}_i, y_i\}, i = 1, \cdots, L, y_i \in \{-1, +1\}, \mathbf{x}_i \in R^n$ be the labeled training set. SVMs are hyperplanes that separate the training data by a maximal margin between the two data classes, with all vectors labeled +1 lying on one side and all vectors labeled -1 lying on the other side (see Fig. 5.1).

The constraints that have to be met for this separation to occur can be written as follows:

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x}_i + b &\geq +1 \quad for \quad y_i = +1 \\ \mathbf{w} \cdot \mathbf{x}_i + b &\leq -1 \quad for \quad y_i = -1 \end{aligned} \tag{5.2}$$

where $\mathbf{w}$ is normal to the hyperplane H. The training vectors that lie on hyperplanes $H_0 : \mathbf{w} \cdot \mathbf{x}_i + b = 1$ and $H_1 : \mathbf{w} \cdot \mathbf{x}_i + b = -1$, are called *support vectors*. It can be shown that the margin between the two hyperplanes H_0 and H_1 is simply $\frac{2}{\|\mathbf{w}\|}$, thus searching for the optimal separating hyperplane becomes a typical constrained optimization problem [9]: minimizing $\| \mathbf{w} \|^2$ subject to the constraints given by (5.2).

Introducing non-negative Lagrange multipliers $\alpha_i, i = 1, \cdots, L$, one for each of the constraints in (5.2), leads to an unconstrained optimization, where the goal is to maximize the following Lagrangian objective function:

$$\max(\sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j), \tag{5.3}$$

with respect to non-negative Lagrange multipliers α_i, subject to constraints $\sum_i \alpha_i y_i = 0$. $\mathbf{x}_i \cdot \mathbf{x}_j$ is the inner product between vectors $\mathbf{x}_i$ and $\mathbf{x}_j$. Note that there is a Lagrange multiplier α_i associated with each training point $\{\mathbf{x}_i, y_i\}$. The solution (the optimal hyperplane) is given by:

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i \tag{5.4}$$

In the solution, those points for which $\alpha_i > 0$ are support vectors. The support vectors lie closest to the decision boundary (see Fig. 5.1). Typically they represent a small portion of the training data.

One important property of the Lagrangian formulation of the problem, as can be seen from (5.3), is that the training data only appear in the form of inner products. This is a crucial property which will allow us to generalize the training procedure to non linear transformations of the feature space by introducing appropriate kernel functions.

If the training samples are not linearly separable in the original space χ, suppose that we first map the data to some other Euclidean space $\mathcal{H}$ (possibly infinite dimensional) using a mapping $\Phi : \chi \mapsto \mathcal{H}$. The expectation is that, with a suitable mapping, the training samples will be more sparsely distributed and therefore can be more easily separated by a linear hyperplane. In the SVM literature, $\mathcal{H}$ is usually considered to be a Hilbert space, a Euclidean space where an inner product has been defined. Since the training algorithm only depends on the inner products between

sample vectors, we can define a kernel function K such that $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$, so that we only need to replace the inner product $\mathbf{x}_i \cdot \mathbf{x}_j$ by $K(\mathbf{x}_i, \mathbf{x}_j)$ everywhere in the training algorithm (5.3), without ever having to explicitly compute the mapping Φ. Any functional satisfying Mercer's condition [78] [9] represents a valid kernel. Through the use of kernels, SVMs may be built and tested in high (possibly infinite) dimensional feature spaces, while maintaining low computational complexity for training and testing. The resulting classifier takes the form:

$$g(\mathbf{x}) : sgn(\sum_{i=1}^{N_s} \alpha_i y_i K(\mathbf{x_i}, \mathbf{x}) + b). \tag{5.5}$$

where $\{\alpha_i, i = 1, \cdots, N_s\}$ and b are parameters that can be learned using quadratic programming [9]. N_s is the number of support vectors.

Most of the flexibility and classification power of SVMs resides in the kernel functions, since these make it possible to discriminate within challenging data sets, e.g., those where linear discrimination may be suboptimal. Typical kernel functions include linear, polynomial and radial basis function (RBF), which are defined as follows:

$$Linear: \ K(\mathbf{x}, \mathbf{z}) = \mathbf{x} \cdot \mathbf{z} \tag{5.6}$$

$$Polynomial: K(\mathbf{x}, \mathbf{z}) = (A\mathbf{x} \cdot \mathbf{z} + B)^p \tag{5.7}$$

$$Radial\ Basis: K(\mathbf{x}, \mathbf{z}) = e^{-\gamma ||\mathbf{x} - \mathbf{z}||^2} \tag{5.8}$$

where $\mathbf{z}$ is another vector of the same dimension as $\mathbf{x}$ and $(\cdot)$ denotes the inner product of two vectors. A, B, p and γ are constants which are set a priori. It is important to note that these kernels are generic and do not explicitly take into account the statistics of the data set being classified. The Fisher kernel proposed by Jaakkola [34] is a data-dependent kernel which aims at taking data statistics into account. Assume there is a generative model $P(\mathbf{x}|\theta)$ that explains well all possible data. The original input vector $\mathbf{x}$ is first mapped to a *Fisher score* $F_{\mathbf{x}}$ as follows:

$$F_{\mathbf{x}} = \bigtriangledown_{\theta} \log(P(\mathbf{x}|\theta)), \tag{5.9}$$

then standard kernels can be applied on this *score space*. Instead of mapping each object to the gradient log-likelihood vector space as in Fisher Kernel, in [43], the Kullback-Leibler divergence was directly computed from the distributions of the objects and was combined with the RBF kernel to compute inner products between vectors for speech recognition applications. In relevance feedback of content-based retrieval systems, however, generative models that can be used the underlying probabilistic distribution, e.g., HMMs/GMMs, may not be available. The Fisher kernel method [34] is inherently parametric, therefore can not be generalized to cases where only non-parametric estimation is available (e.g., estimation from quantized bins). We propose to employ non-parametric methods to capture statistical characteristics of the user's preference from the positive and negative samples, and we derive a new kernel called User Preference Information Divergence (UPID). Our scheme makes no

prior assumptions about the data distribution, which is exactly what we are trying to learn.

5.3 Kernel based on User Preference Information Divergence

Our proposed method aims at learning the user's preferences empirically, through probabilistic information contained in the user's feedback, and then using this information to derive a kernel that is customized for the specific user and task. We will first show that if the probabilistic estimate is accurate, performance can approach that of a maximum likelihood classifier. We will then propose a simple and practical technique to model the probabilistic information provided by the user feedback.

Consider first the ideal case where we assume the a posteriori probabilities $P(y = \pm 1|\mathbf{x})$ are known for any observation $\mathbf{x}$. Then, given any two observations $\mathbf{x}$ and $\mathbf{z}$ we can compute the Kullback-Leibler divergence between $P(y = \pm 1|\mathbf{x})$ and $P(y = \pm 1|\mathbf{z})$:

$$D(\mathbf{x}||\mathbf{z}) = P(y = +1|\mathbf{x}) \log(\frac{P(y = +1|\mathbf{x})}{P(y = +1|\mathbf{z})}) + P(y = -1|\mathbf{x}) \log(\frac{P(y = -1|\mathbf{x})}{P(y = -1|\mathbf{z})}). \tag{5.10}$$

Note that $D(\mathbf{x}||\mathbf{z})$ is not symmetric. Thus, in order to have a valid distance, a symmetric Kullback-Leibler divergence can be defined as:

$$D_s(\mathbf{x}, \mathbf{z}) = D(\mathbf{x}||\mathbf{z}) + D(\mathbf{z}||\mathbf{x}). \tag{5.11}$$

This distance can then be used to replace the inner product or the Euclidean distance in the standard kernels in equations (5.6), (5.7) and (5.8) to form a valid kernel. Under our assumption that an accurate model is available (in a practical case this would entail having enough training data) we are interested in evaluating the performance of the SVM classifier obtained using the new kernel.

Consider all the points $\mathbf{x}$ which have a given likelihood ratio of a posteriori probabilities, ℓ, i.e., such that:

$$\mathcal{L}(\mathbf{x}) = \frac{P(y = +1|\mathbf{x})}{P(y = -1|\mathbf{x})} = \ell. \tag{5.12}$$

it can be easily seen that these points will have a posteriori probabilities $P(y = +1|\mathbf{x}) = \frac{\ell}{\ell+1}, P(y = -1|\mathbf{x}) = \frac{1}{\ell+1}$. The distance between any such point $\mathbf{x}$ and and a support vector $\mathbf{x}_i, i = 1, \cdots, N_s$ can be computed by using equations (5.10) and (5.11) as follows:

$$\begin{aligned} D_s(\mathbf{x}, \mathbf{x}_i) = \quad & -H_2 \quad (P_i^+) - P_i^+ \log \ell + \log(\ell + 1) \\ & -H_2 \quad (\frac{\ell}{\ell+1}) + \frac{\ell}{\ell+1} \log \frac{P_i^+}{P_i^-} - \log P_i^- \end{aligned} \tag{5.13}$$

where P_i^+ and P_i^- are the positive and negative a posteriori probabilities for support vector $\mathbf{x}_i$, respectively. $H_2(p)$ is the binary entropy [20] computed as $-p\log p-(1-p)\log(1-p)$.

We can see that this distance is only a function of the likelihood ratio ℓ and is independent of $\mathbf{x}$. Thus any function $h()$ of such a distance (e.g., a function of the distance that might be used to form the kernel) will be independent of $\mathbf{x}$. Therefore all the points that have a given likelihood ratio will have the same score with respect to the support vectors, i.e.,

$$\begin{aligned} f(\mathbf{x}) &= \sum_{i=1}^{N_s} \alpha_i y_i h(D_s(\mathbf{x},\mathbf{x}_i)) + b \\ &= C(\ell), \quad \forall \mathbf{x} \;\; s.t. \;, \frac{P(y=+1|\mathbf{x})}{P(y=-1|\mathbf{x})} = \ell. \end{aligned} \tag{5.14}$$

This is equivalent to saying that all the points lying on the decision boundary of a Maximum likelihood ratio classifier, as represented by equation (5.12) [22] [52], are mapped to a hyperplane in the high-dimensional feature space (through the kernel) represented by $f(\mathbf{x}) = C(\ell)$, which is parallel to the SVM separating hyperplane $H : f(\mathbf{x}) = 0$. Thus, as is desirable, the decision hyperplanes will contain constant likelihood ratio feature vectors.

Assume now that all the positive support vectors (which lie on the hyperplane H_0, as shown in Figure 5.1) have likelihood ratio ℓ^+ and Lagrange multiplier α_+, the negative support vectors (which lie on the hyperplane H_1) have likelihood ratio

$\ell^- = \frac{1}{\ell^+}$ and Lagrange multiplier α_-. Assume further that the number of positive and negative support vectors are equal, $N_+ = N_-$. Then we are interested in evaluating the score for all the points which have likelihood ratio $\ell = 1$. From [9] we know that in this case $b = 0$. Then the distance score for those points can be computed as:

$$\begin{aligned} f(\mathbf{x}) &= \sum_{i=1}^{N_s} \alpha_i y_i h(D_s(\mathbf{x}, \mathbf{x}_i)) \\ &= \sum_{i=1}^{N_+} \alpha_+ y_i h(-H_2(\frac{\ell^+}{\ell^+ + 1}) - \frac{1}{2}\log(\frac{\ell^+}{(\ell^+ + 1)^2})) \\ &+ \sum_{i=1}^{N_-} \alpha_- y_i h(-H_2(\frac{\ell^-}{\ell^- + 1}) - \frac{1}{2}\log(\frac{\ell^-}{(\ell^- + 1)^2})) \\ &= h(-H_2(\frac{\ell^+}{\ell^+ + 1}) - \frac{1}{2}\log(\frac{\ell^+}{(\ell^+ + 1)^2})) \sum_{i=1}^{N_s} \alpha_i y_i \end{aligned} \tag{5.15}$$

Note that the argument of function $h()$ is the same for ℓ^+ and $\frac{1}{\ell^+}$, thus it can be factored out of the summation. Taking into account the fact that $\sum_{i=1}^{N_s} \alpha_i y_i = 0$, we have:

$$f(\mathbf{x}) = 0, \quad \forall \mathbf{x} \quad s.t. \ , \frac{P(y = +1|\mathbf{x})}{P(y = -1|\mathbf{x})} = 1. \tag{5.16}$$

which means the SVM classifier using the KL divergence kernel produces an optimal partition of the space in terms of minimizing Bayes classification risk.

Clearly, this result is important, as it indicates that our proposed method can approach the optimal classifier if a sufficiently good estimation of the underlying

probability model is available. However, in relevance feedback applications, there is no such probabilistic model available since the query concept is unknown and time-varying. Furthermore, the amount of available training samples is usually very small relative to the dimensionality of the data, which prevents us from using traditional methods to estimate the joint probabilistic models. Given these challenges, we propose to use marginal distributions for each component and derive a kernel based on marginal divergences between positive and negative probabilities. We show that by combining this approximate probabilistic model with an SVM, we are able to effectively capture information about the user's query concept, even though the number of training samples is small.

One important conceptual advantage of our proposed technique is that it separates the estimation of the underlying probabilistic model from the classification task itself. Thus, while in our experiments we describe a non-parametric method for density estimation, application specific probabilistic estimation tools could be brought to bear in other cases. For example, in some cases a combination of parametric and non-parametric tools might be used.

For a given feature vector $\mathbf{x} = (x_1, x_2, \cdots, x_n)^t$, we define the marginal probability of each label for each component of the feature vector x_l as $\{P(y = +1|x_l), P(y = -1|x_l)\}$. These marginal distributions for each component x_l can be empirically estimated from the training data (i.e., from the feedback data in our case). Clearly, this estimation process is challenging because i) x_l can in general take values in

either a large discrete set or a continuous range, and ii) limited amounts of data are available for training.

Given these challenges, parametric models for $\{P(y = +1|x_l), P(y = -1|x_l)\}$ could be considered but this in turn would imply that some prior assumptions need to be made about the distributions. Thus, in order to have as much flexibility as possible, we choose a non-parametric probability estimation approach. In what follows training data will refer to data obtained by accumulating successive iterations of user feedback.Thus the amount of available training data increases every time the user provides feedback.

For each feature vector component x_l we define a quantizer $\mathcal{A}_l$ that consists of B_l reconstruction levels r_{lk} with $B_l - 1$ decision boundaries denoted as $\{b_1, \cdots, b_{B_l-1}\}$. Thus the quantizer $\mathcal{A}_l$ partitions the input space into B_l non-overlapping regions. We estimate the probabilities $\{P(y = +1|x_l), P(y = -1|x_l)\}$ by counting the number of samples that fall in each bin:

$$P(y = \pm 1|x_l = r_{lk}) = \frac{\sum_{i=1}^{L} 1(y_i = \pm 1)1(|x_{il} - r_{lk}| \leq \Delta_{lk})}{\sum_{i=1}^{L} 1(|x_{il} - r_{lk}| \leq \Delta_{lk})} \tag{5.17}$$

where the indicator function $1(\cdot)$ is equal to one when its argument is true and zero otherwise. L is the number of labeled training data. x_{il} is the l-th component of training vector $\mathbf{x}_i$. $2\Delta_{lk}$ is the size of the quantization interval along dimension l centered at reconstruction value r_{lk}. Figure 5.2 shows an example of the partition

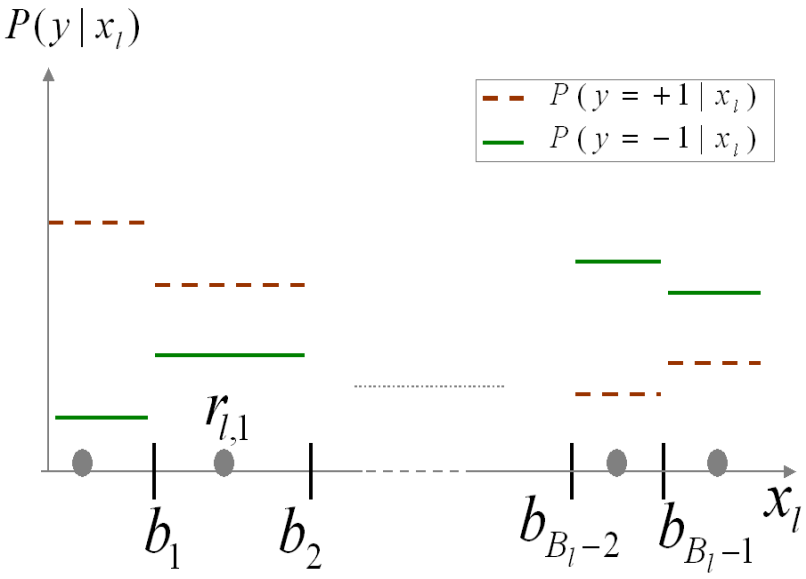


Figure 5.2: An example of the partition induced by quantizer $\mathcal{A}_l$ for component x_l. The dashed and solid bars above each bin represent the marginal probabilities $P(y=+1|x_l)$ and $P(y=-1|x_l)$, respectively.

induced by quantizer $\mathcal{A}_l$ for component x_l. The dashed and solid bars above each bin represent the marginal probabilities $P(y=+1|x_l)$ and $P(y=-1|x_l)$, respectively.

For those quantization bins where there is no training data, we simply set the marginal probabilities to zero since they make no contribution to differentiating classes.

Obviously the design of the quantizers, $\mathcal{A}_l$, in general may play an important role in probability estimation. In this chapter we focus on a simple uniform quantization scheme where all quantization bins in a given feature dimension have the same size $2\Delta_{lk}$, which is computed from the dynamic range of the data $[\max(x_l), \min(x_l)]$

(note that this range may change from iteration to iteration) and the number of quantization levels applied B_l:

$$\Delta_{lk} = \Delta_l = \frac{\max(x_l) - \min(x_l)}{2 \times B_l} \tag{5.18}$$

We also compare uniform quantization with a non-uniform quantization scheme which has unequal bin widths. The bin widths and the number of the bins are determined by minimization of the MSE (mean square error) using the Lloyd algorithm [26]. Piecewise linear approximations to the marginal probability densities could be used [48] instead of piecewise constant ones as proposed here. Also more sophisticated techniques, such as the MDL (minimum description length) principle [58], can potentially be used. In this chapter, we focus on uniform quantization due to its low complexity (which is a desired property for online learning).

With the probability model we just described we can view a feature vector $\mathbf{x} = (x_1, x_2, \cdots, x_n)^t$ as a sample drawn from a random source, which has relevance statistics given by $P^+(\mathbf{x}) = (p_1^+, \cdots, p_n^+)$ and $P^-(\mathbf{x}) = (p_1^-, \cdots, p_n^-)$. The $p_l^\pm = P(y = \pm 1 | \mathcal{A}_l(x_l))$ are estimated by quantizing the component x_l using $\mathcal{A}_l$ based on the training data obtained from relevance feedback.

Assume that we wish to estimate the distance between $\mathbf{x}$ and $\mathbf{z}$, where $\mathbf{z}$ is another feature vector with probability vectors $Q^+ = (q_1^+, \cdots, q_n^+)$ and $Q^- = (q_1^-, \cdots, q_n^-)$.

Here, our proposed distance based on the Kullback-Leibler divergence of the probability vectors P and Q can be obtained as:

$$D(\mathbf{x}||\mathbf{z}) = \sum_{l=1}^{n} p_l^+ \log(\frac{p_l^+}{q_l^+}) + \sum_{l=1}^{n} p_l^- \log(\frac{p_l^-}{q_l^-}) \tag{5.19}$$

We assume $0 \times \log(0) = 0$ by continuity arguments. Since the KL divergence is not symmetric we define based on (5.19) a symmetric distance measure $D_s(\mathbf{x}, \mathbf{z})$ using (5.11) to obtain a symmetric distance D_s. We then define our proposed UPID kernel function in the generalized form of RBF kernels with the original Euclidean distance $d()$ replaced by the proposed distance of (5.11):

$$K(\mathbf{x}, \mathbf{z}) = e^{-\rho D_s(\mathbf{x}, \mathbf{z})} \tag{5.20}$$

The distance (5.20) is a positive definite metric [43], thus the proposed UPID kernel satisfies Mercer's condition [9]. As the model parameters α_i, b and N_s are learned from the training set, we evaluate the likelihood that an unknown object $\mathbf{x}$ is relevant to the query by computing its score $f(\mathbf{x})$:

$$f(\mathbf{x}) = \sum_{i=1}^{N_s} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b, \tag{5.21}$$

where $\mathbf{x}_i$ is the ith support vector and there are a total of N_s support vectors (which is determined by the learning process). The larger the score is, the more

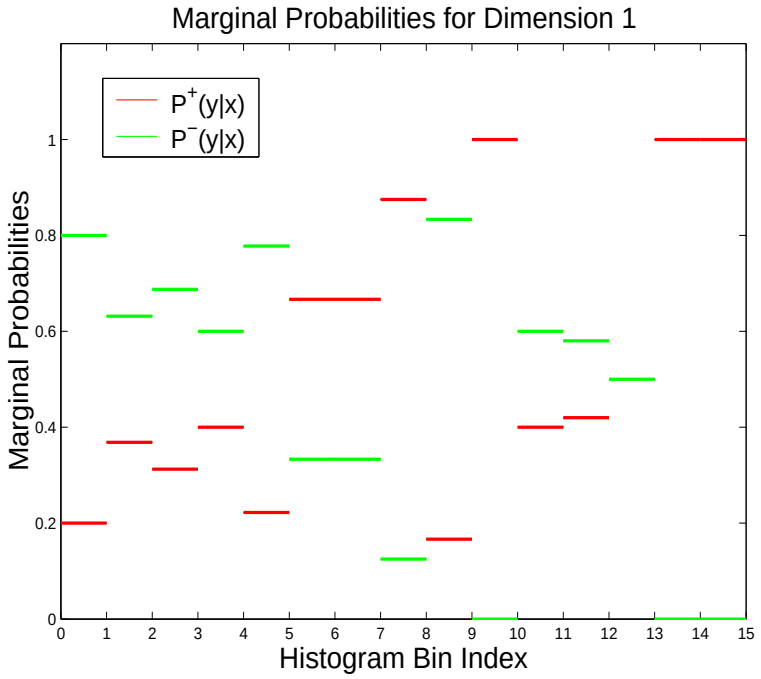


Figure 5.3: An example of the estimated marginal probabilities $P(y = +1|x_i)$ and $P(y = -1|x_i)$.

likely it is that the unknown object belongs to the relevant class and thus shall be returned and displayed to the user.

As an example, we show in Figure 5.3 the estimated marginal probabilities $\{P(y = +1|x_l), P(y = -1|x_l)\}$ based on a 15-bin uniform quantization for one dimension. The training data is collected from accumulation of two relevance feedback iterations. Figure 5.4 shows KL divergence based on the estimated probabilities shown in Figure 5.3, and the standard Euclidean distance (computed as the squared difference between the centroids of bins).

In this example, the reference point (query point) lies in the second bin (which is indicated as a dot in figure 5.4). We can see that the standard Euclidean distance

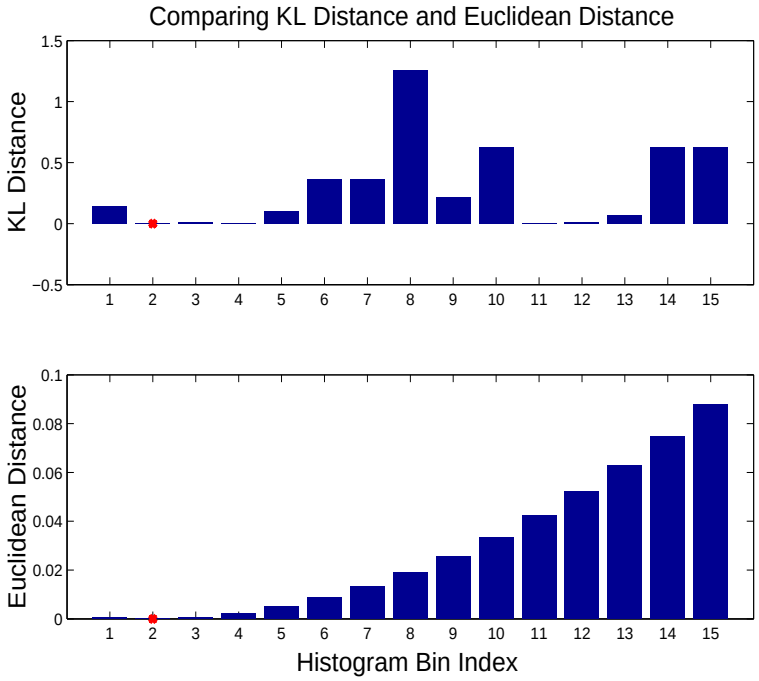


Figure 5.4: An example showing different properties between KL divergence and standard Euclidean distance. The KL divergence is computed based the probabilities estimated shown in Figure 5.3. The dot indicates the location of the query point. We can see that the KL divergences between the query point and the points located in bins 11 and 12 are small, although bins 11 and 12 are physically distant from the bin 2. Euclidean distance merely reflects the physical distance in the low-level feature space.

solely depends on the physical distance between two points. Instead, the KL distance is able to capture the probabilistic similarities between two points, even though they are physically far apart from each other. This can be verified by noting the smaller KL distances between the query point (located in bin 2) and the points located in bins 11 and 12.

5.4 Experiments

As an experimental evaluation of the proposed scheme, we compare the performance of five learning methods: i) the query refinement and re-weighting (QRR) algorithm [61], ii) SVM using polynomial kernel (Polynomial), iii) SVM using Radial Basis function kernel (RBF), iv) SVM using our proposed probabilistic kernel (UPID), and v) SVM using linear kernel (Linear). 1500 real world images are chosen from the COREL Image CDs [18]. The image set includes 15 different categories[3], with 100 images for each category. Our experimental set up is very similar to that of [79], the only difference being that we replace the categories *Auto racing* and *Roses*, with *Exotic cars* and *Flowers*, respectively, since we do not have access to the former. We use 80% of each category (1200 images in total) as the database, and 20% (300 images in total) as the query images. The splitting of the data (80% and 20%) is chosen to be consistent with standard practice in machine learning.

[3]Sunset, Coasts, Flowers (volume II), Exotic cars, Mayan & Aztec, Fireworks, Ski scenes, Owls, Religious Stained glass, Arabian horses, Glaciers & Mountains, English country gardens, Divers & diving, Land of the pyramids, and oil paintings.

We employ the feature extraction algorithm in [13]. Three different features are extracted to represent the images: color, texture and shape. The color features are computed as the histograms in CIELab color space. The texture feature is formed by applying the Sobel operator to the image and histograming the magnitude of the local image gradient. The shape feature is characterized by histograming the angle of the edge. Dimensionality of the feature vector we used in our experiments is 72 (the number of bins used for each histogram is 8).

For the experiments, we assume that the query feedback is based on the actual image categories, i.e., all images corresponding to the same category as as the image being queried will be deemed relevant and those from other categories will not be relevant. The quality of the retrieval result is measured by two quantities: *precision* and *recall*. Precision is the percentage of relevant objects in the retrieved set, it measures the purity of the retrieval. Recall is a measurement of completeness of the retrieval, computed as the percentage of retrieved relevant objects in the total relevant set in the database.

When the system is presented with a query image, it will first search for the K nearest neighbors based on the Euclidean distance between the query image and each of the images in the database. Then the returned images that belong to the same category as the query image will be labeled as positive, and all the others in the returned set labeled as negative. The system learns the new model parameters and returns a new round of images and repeats this process. The labeled images

accumulate from iteration to iteration as the system gets more feedback from the user. For SVM based methods, the parameters α_i and b are learned using (5.3) based on the labeled images. The next round of retrieval will be carried out using the classifier (5.21) with the new set of parameters. The images with highest score are most likely to be the target images for the user. For the Query Refinement and Re-weighting method, we implemented the algorithm proposed in [61]. Then the new query vector and new weights are used to perform a K-Nearest-Neighbor classification. The precision and recall are averaged over all the test images.

The SVM learning algorithms are implemented based on the SVM^{light} library [37]. Figure 5.6 shows the precision-recall curves comparing proposed method (with parameter ρ set to 1), Query Refinement and Re-weighting (QRR), SVM with RBF kernel (with parameter γ set to 1), SVM with polynomial kernel (degree p = 4, $A = B = 1$), and linear kernel. In proposed scheme, we fix the number of quantization bins for all dimensions to be the same.

Top-K retrieval precision as a function of the number of returned images are plotted in Figure 5.5. Top-K retrieval precision is defined as the precision evaluated for the K returned images which have highest similarities to the query image. We can clearly see that the proposed method achieves significantly higher search accuracy than the other methods.

Figure 5.6 presents the Precision-Recall curves after 3 relevance feedback iterations, comparing five different methods.

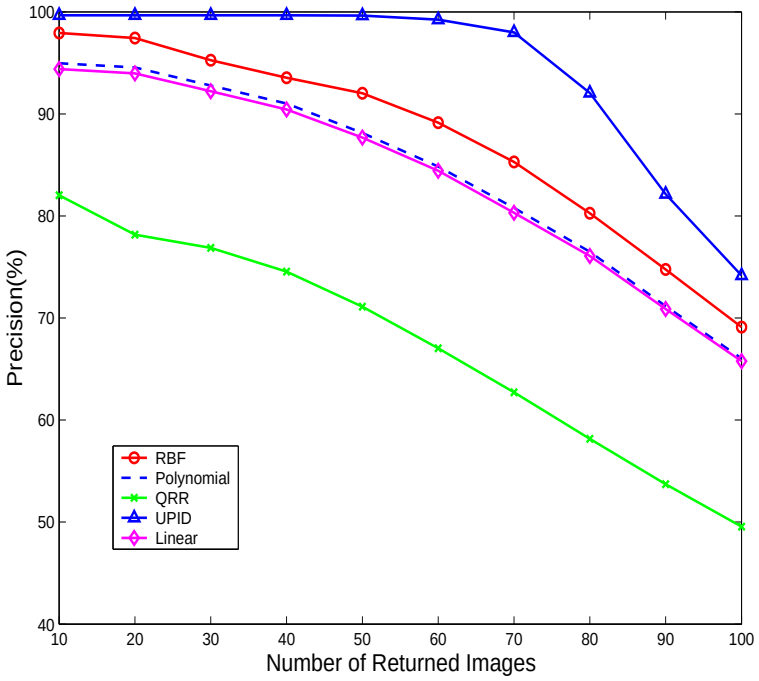


Figure 5.5: Top-K accuracy as a function of the number of returned images after 6 relevance iterations. We can see that compared to other methods, proposed method has a more compact display of the relevant images (Precision is relatively flat in the beginning and gets a sharper tail off).

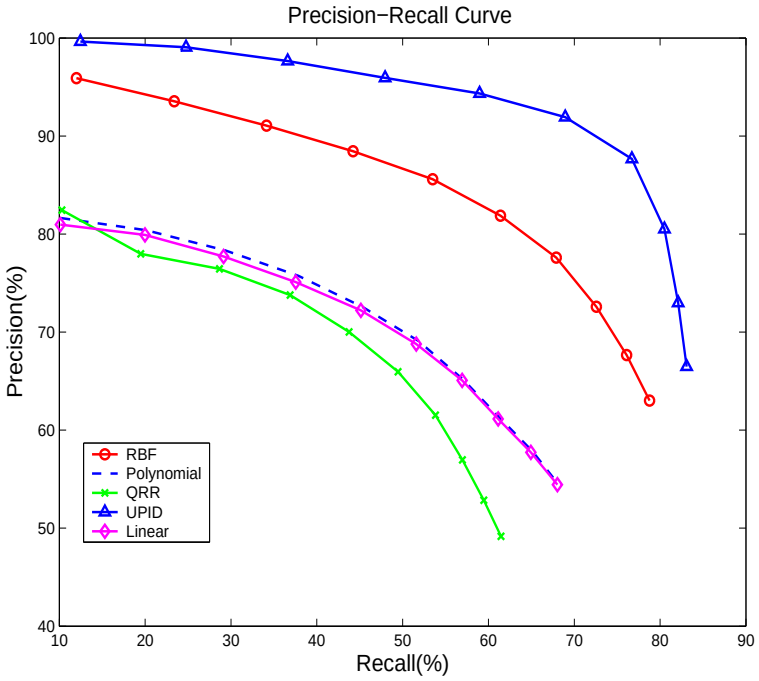


Figure 5.6: Precision-Recall curves after 3 relevance feedback iterations, comparing five methods: SVM with RBF kernel (Circles), SVM with Polynomial degree 2 (Dashed lines), Query Refinement and Re-weighting (Cross), SVM with Proposed UPID kernel (Triangles), and SVM with Linear Kernel (Diamonds).

Table 5.1 shows the top-K accuracy (mean and variance) after 6 relevance feedback iterations. We can see that SVM based active learning methods perform significantly better than the query refinement/re-weighting method. SVM with proposed empirical probabilistic kernel function is the best performer among all SVM based methods. It achieves almost 100% top-50 accuracy, while RBF kernels get around 92%. Our results are consistent with the results reported in literature using the standard kernels. The performance of RBF and polynomial were reported for a 4-category corel dataset in [11]. The top-50 accuracy after 3 iterations are 92.7% for polynomial degree 4, and 96.8% for RBF. Considering that learning is more accurate with smaller number of image classes, our results are consistent with theirs.

Algorithm	Top-20	Top-50	Top-80
RBF	97.43±0.44	92.03±0.77	80.27±0.79
Polynomial	94.60±0.64	88.69±0.85	77.12±0.78
UPID	**99.67±0.33**	**99.64±0.33**	**92.05±0.73**
Linear	93.97±0.82	87.69±1.07	76.08±0.83
QRR	78.17±7.4	71.11±7.56	58.15±4.96

Table 5.1: Top-K accuracy (mean and variance) after 6 relevance feedback iterations for various methods. Bold numbers indicate the best performer. The parameters chosen are: $\gamma = 1$ for the RBF kernel, $p = 4, A = B = 1$ for the polynomial kernel, and $\rho = 1$ for proposed UPID kernel. We implemented the query refinement and re-weighting based on the algorithm by Rui et.al.

We also show in figure 5.7 the improvement of the retrieval accuracy as a function of the number of interaction rounds. It basically gives us an idea about the speed (how many interactions are needed in order to achieve a certain accuracy) with which

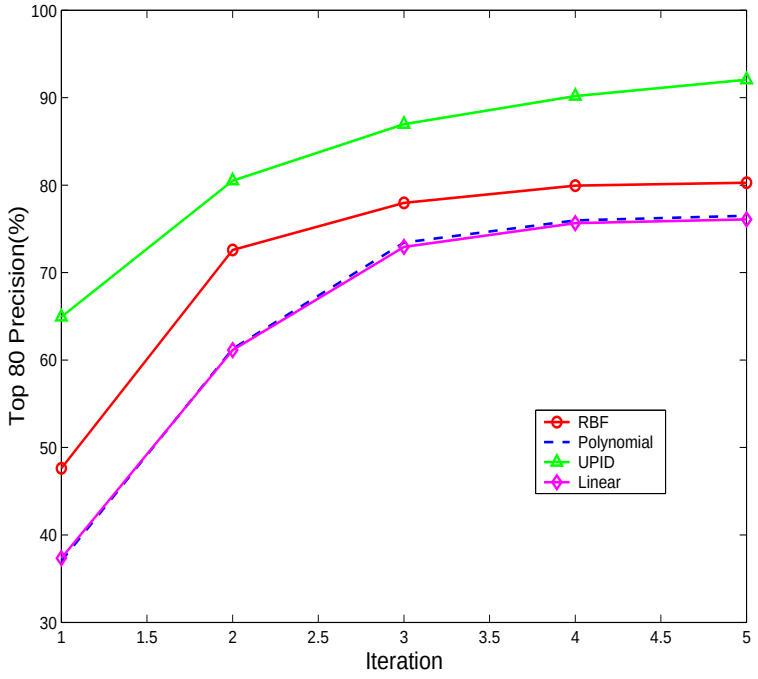


Figure 5.7: Comparison of learning accuracy of three different kernels (evaluated as the top-80 retrieval precision) as a function of the number of relevance feedback iterations. The accuracy without relevance feedback is 40.78%, it is obtained by a K-Nearest-Neighbor classifier with the weights equal for all feature components.

the system is able to capture the query concept through the information provided at each interaction round.

We see that proposed kernel outperforms the other three most popular kernels. About 17% higher accuracy is achieved after the first iteration using proposed kernel as compared to the RBF kernel. This is encouraging since usually very a very small number of positive samples are available at the beginning of the interaction.

We also investigate the reliability of the proposed empirical estimation scheme (5.17) to different quantization schemes. We compare uniform quantization at different bin sizes and non-uniform quantization. We test on $B_l = 10, 15, 20$ and figure

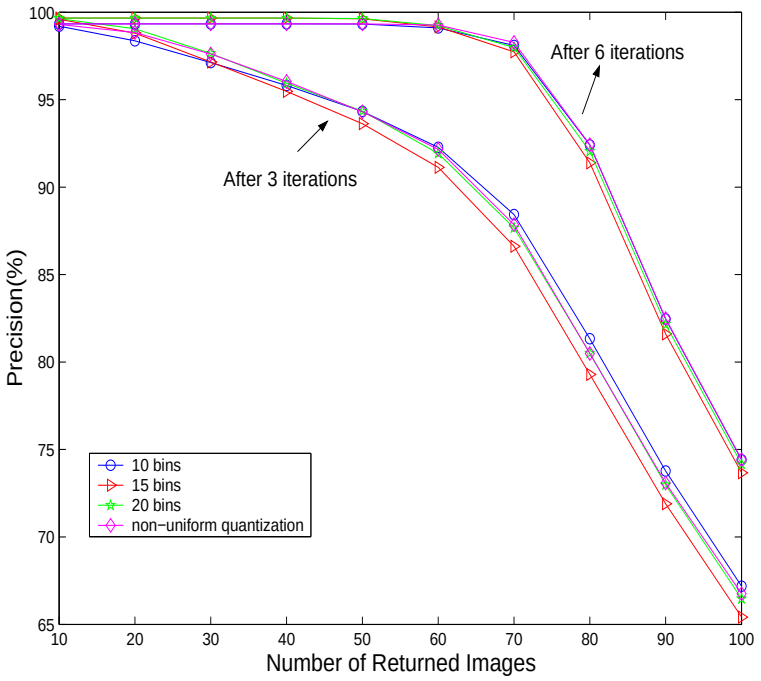


Figure 5.8: Precision-Recall curves of proposed scheme after 3 and 6 relevance feedback iterations using different number of quantization bins for probability estimation. We can see that neither varying the number of quantization bins nor having a different quantization scheme has much effects on the learning performance, and thus the proposed empirical estimation scheme is very reliable.

5.8 shows the precision-recall curves of proposed scheme after 3 and 6 relevance feedback iterations. We can see that neither varying the number of quantization bins nor having a different quantization scheme has a significant effect on the learning performance. Thus the estimation is reliable, at least for the purpose of improving classification based on received feedback. Another fact worth noticing is that the number of labeled positive samples is relatively small in the beginning of the learning process, and still the accuracy improvement is remarkable after only one relevance feedback using proposed method (17% higher than RBF kernel, see Fig. 5.7).

5.5 Conclusions and Future work

In this chapter we proposed a new method of employing the data statistics for active learning using SVM in content-based image retrieval. The derivation of the new kernel is empirical and requires no domain knowledge, it is thus a practical approach for relevance feedback learning tasks where the query concept is not known and can be time varying. Our experiments have shown promising performance using proposed scheme compared with other kernels. Our future work includes designing adaptive methods for estimating the marginal distributions (current version uses a simple uniform quantization to estimate the probabilities), taking into account the data imbalance problem (the number of negative samples is much larger than the number of positive samples), and speeding up of the learning. We are also investigating how to incorporate the ranking information (i.e., cases when the user has different degrees of preference for the relevant images) into our framework.

Reference List

[1] S. M. Ali and S. D. Silvey. A general class of coefficients of divergence of one distribution from another. *Journal of Royal Statistical Society, Series B*, 28:131–142, 1966.

[2] B. Beferull-Lozano, H. Xie, and A. Ortega. Rotation-invariant features based on steerable transforms with an application to distributed image classification. *IEEE International conference on Image processing*, 2003.

[3] M. Beigi, A. Benitez, and S. F. Chang. Metaseek: A content-based meta search engine for images. *Proc. of SPIE Storage and Retrieval for Image and Video Databases*, 1998.

[4] S. Berchtold, D. Keim, and H. P. Kriegel. The x-tree: An index structure for high dimensional data. In *Proc. of Int. Conference on Very Large Databases*, pages 28–39, 1996.

[5] S. Berchtold, D. A. Keim, and H. P. Kriegel. The x-tree: An index structure for high-dimensional data. *Proc.of VLDB*, 1996.

[6] L. Breiman, J. H. Freidman, R. A. Olsehn, and C. J. Stone. *Classification and Regression Trees.* Wadsworth, 1984.

[7] P. Brodatz. Textures: A photographic album for artists and designers. Dover, New York, 1966.

[8] http://sipi. usc. edu/services/database/Database. html, University of Southern California. Brodatz rotated textures.

[9] C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge discovery*, pages 121–167, 1998.

[10] K. Chakrabarti and S. Mehrotra. High dimensional feature indexing using hybrid trees. *Proc. of international conference on data engineering*, 1999.

[11] E. Chang and S. Tong. Support vector machine active learning for image retrieval. *ACM International Conference on Multimedia*, pages 107–118, October 2001.

[12] T. Chang and C. C. J. Kuo. Texture analysis and classification with tree-structured wavelet transform. *IEEE Trans. on Image processing*, 2:429–441, Oct. 1993.

[13] J.-Y. Chen, C. A. Bouman, and J. Dalton. Similarity pyramids for browsing and organization of large image databases. *Proc. of SPIE Conf. on Human Vision and Electronic Imaging III*, 3299, Jan. 1998.

[14] Y. Chen, X. Zhou, and T. S. Huang. One-class support vector machine for learning in image retrieval. *IEEE International Conference on Image Processing*, 2001.

[15] G. Cheung. Optimal bit allocation strategy for joint source/channel coding of scalable video. Master's thesis, University of California at Berkeley, 1998.

[16] P. A. Chou, T. Lookabough, and R. M. Gray. Optimal pruning with applications to tree-structured source coding and modeling. *IEEE Trans. on Info. Theory*, IT-35:299–315, March. 1989.

[17] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity. *IEEE International Conference on Image Processing*, 1995.

[18] Corel stock photo library. Corel Corp. Ontario, Canada.

[19] T. H. Corman, C. E. Leiserson, and R. L. Rivest. *Introduction to algorithms.* McGraw-Hill, 1990.

[20] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley & Sons, INC, 1991.

[21] S. Dolinar, A. Kiely, M. Klimesh, R. Manduchi, A. Ortega, S. Lee, P. Sagetong, H. Xie, G. C. J. Harel, S. Shambayati, and M. Vida. Region-of-interest data compression with prioritized buffer management (ii). *Earth Science Technology Conference (ESTC)*, Aug. 2002.

[22] R. O. Duda and P. E. Hart. *Pattern classification and scene analysis.* Wiley-interscience Publication, 1973.

[23] H. Everett. Generalized lagrangian multiplier method for solving problems of optimum allocation of resources. *Operations research*, 11:399–417, 1963.

[24] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, and A. E. Abbadi. Vector approximation based indexing for nun-uniform high dimensional data sets. *Proceedings of the 9th ACM Int. Conf. on Information and Knowledge Management*, pages 202–209, November 2000.

[25] K. Fukunaga. *Introduction to statistical pattern recognition.* Academic Press, 1991.

[26] A. Gersho and R. M. Gray. *Vector Quantization and Signal Processing.* Kluwer Academic publishers, 1992.

[27] V. Goyal and M. Vetterli. Computation distortion characteristics of block transform coding. In *Proc. of ICASSP*, Munich, Germany, April 1997.

[28] V. K. Goyal. Theoretical foundations of transform coding. *IEEE Signal Processing Magazine*, 18:9–21, Sep. 2001.

[29] R. M. Gray and D. L. Neuhoff. Quantization. *IEEE Trans on Information Theory*, 44:2325–2384, 1998.

[30] R. M. Gray, J. C. Young, and A. K. Aiyer. Minimum discrimination information clustering: modeling and quantization with gauss mixtures. In *IEEE Intl. Conf on Image Processing*, volume 2, pages 14–17, 2001.

[31] C. Hoi, C. Chan, K. Huang, M. R. Lyu, and I. King. Biased support vector machine for relevance feedback in image retrieval. *International joint conference on Neural Networks*, 2004.

[32] P. Hong, Q. Tian, and T. S. Huang. Incorporate support vector machines to content-based image retrieval with relevance feedback. *IEEE International Conference on Image Processing*, 2000.

[33] T. S. Huang and X. S. Zhou. Image retrieval with relevance feedback: From heuristic weight adjustment to optimal learning methods. *IEEE International Conference on Image Processing*, 2001.

[34] T. Jaakkola, M. Diekhans, and D. Haussler. Using the fisher kernel method to detect remote protein homologies. *Proc. of the International Conference on Intelligent Systems for Molecular Biology*, Aug. 1999.

[35] S. Jana and P. Moulin. Optimal design of transform coders and quantizers for image classification. *IEEE International conference on Image processing*, 2000.

[36] S. Jana and P. Moulin. Optimal transform coding of gaussian mixtures for joint classification/reconstruction. *Data Compression Conference*, 2002.

[37] T. Joachims. *Advances in Kernel Methods - Support Vector Learning*, chapter 11: Making Large-scale support vector machine Learning Practical. MIT Press, 1998.

[38] S. A. Kassam. Optimum quantization for signal detection. *IEEE Trans on Communication*, COM-25:479–484, 1977.

[39] N. Katayama and S. Satoh. The sr-tree: An index structure for high dimensional nearest neighbor queries. *Proc. of SIGMOD*, 1997.

[40] K. Lin, H. V. Jagadish, and C. Faloutsos. The tv-tree - an index structure for high dimensional data. *VLDB Journal*, 1994.

[41] W. Y. Ma. *Netra : A Toolbox for Navigating Large Image Databases.* PhD thesis, University of California, Santa Barbara, 1997.

[42] S. Mehrotra, Y. Rui, M. Ortega, and T. S. Huang. Supporting content-based queries over images in MARS. *Proceedings of IEEE Int. Conf. on Multimedia Computing and Systems*, pages 632–633, 1997.

[43] P. J. Moreno, P. Ho, and N. Vasconceles. A kullback-leibler divergence based kernel for svm classification in multimedia applications. *Neural Information Processing Systems*, 2003.

[44] D. Murthy and A. Zhang. Webview: A multimedia database resource integration and search system over web. *WebNet 97: World Conference of the WWW, Internet, and Intranet*, Oct 1997.

[45] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, and C. Faloutsos. The QBIC project: Querying images by content using color, texture, and shape. *Proceedings of SPIE Storage and Retrieval for Image and Video Databases*, 1908:173–187, 1993.

[46] A. Ortega, B. Beferull-Lozano, N. Srinivasamurthy, and H. Xie. Compression for recognition and content based retrieval. *EUSIPCO*, 2000.

[47] A. Ortega and K. Ramchandran. Rate-distorion methods for image and video compression. *IEEE Signal processing magazine*, pages 23–50, 1998.

[48] A. Ortega and M. Vetterli. Adaptive scalar quantization without side information. *IEEE Transactions on Image Processing*, 6(6):665–676, May 1997.

[49] J. Peng, B. Bhanu, and S. Qing. Probabilistic feature relevance learning for content-based image retrieval. *Computer Vision and Image understanding*, 75:150–164, 1999.

[50] A. Pentland, R. W. Picard, and S. Scaroff. Photobook: tools for content-based manipulation of image databases. *Proceedings of SPIE*, 2185:34–47, 1994.

[51] K. O. Perlmutter, S. M. Perlmutter, R. M. Gray, and R. A. Olsen. Bayes risk weighted vector quantization with posterior estimation for image compression and classification. *IEEE Trans. on Image processing*, 5:347–360, Feb. 1996.

[52] B. Picinbono and P. Duvaut. Optimum quantization for detection. *IEEE Trans on Communications*, 36:1254–1258, 1988.

[53] H. V. Poor. Fine quantization in signal detection and estimation. *IEEE Trans on Information Theory*, 34:960–972, 1988.

[54] H. V. Poor and J. B. Thomas. Application of ali-silvey distance measures in the design of generalized quantizers for binary decision systems. *IEEE Trans on Communication*, COM-25:893–900, 1977.

[55] B. Ramamurthi and A. Gersho. Classified vector quantization of images. *IEEE Trans. on Communications*, 34:1105–1115, Nov. 1986.

[56] K. Ramchandran and M. Vetterli. Best wavelet packet bases in a rate-distortion sense. *IEEE Trans. on Image processing*, 2:160–175, April. 1993.

[57] E. A. Riskin. Optimal bit allocation via the generalized BFOS algorithm. *IEEE Trans. on Information Theory*, 37:400–402, March. 1991.

[58] J. Rissanen, T. P. Speed, and B. Yu. Density estimation by stochastic complexity. *IEEE Trans. on Information Theory*, pages 315–323, March 1992.

[59] J. Robinson. A search structure for large multidimensional dynamic indexes. *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, pages 10–18, 1981.

[60] Y. Rui, T. S. Huang, and S. Chan. Image retrieval: Current techniques, promising directions and open issues. *Journal of Visual Communication and Image Representation*, 10:39–62, March 1999.

[61] Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: A power tool in interactive content-based image retrieval. *IEEE trans on Circuits and Systems for Video Technology*, 8:644–655, 1998.

[62] S. R. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. *IEEE Trans. on Systems, Man, and Cybernetics*, IT-21:660–674, May. 1991.

[63] A. Said and W. A. Pearlman. A new fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. on Circuits and Systems for Video Technology*, 6:243–250, June. 1996.

[64] M. Shneier and M. Abdel-Mottaleb. Exploiting the JPEG compression scheme for image retrieval. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18:849–850, 1996.

[65] Y. Shoham and A. Gersho. Efficient bit allocation for an arbitrary set of quantizers. *IEEE Trans. on Acoust. Speech Signal Process.*, 36:1445–1453, Jan. 1988.

[66] E. P. Simoncelli, W. Freeman, E. Adelson, and D. Heeger. Shiftable multiscale transform. *IEEE Transactions on Information Theory*, 2:587–607, 1992.

[67] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jai. Cotent-based image retrieval and the end of the early years. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:1349–1380, 2000.

[68] J. Smith and S. F. Chang. Visualseek: A fully automated content-based image query system. *Proceedings of ACM Multimedia*, 2420:506–517, 1996.

[69] J. R. Smith and S. Chang. Transform features for texture classification and discrimination in large image databases. *IEEE International Conference on Image Processing*, 1994.

[70] J. R. Smith and S. Chang. Tools and techniques for color image retrieval. *IS & T SPIE Proc. on storage and retrieval for image and video databases*, 2670, 1995.

[71] N. Srinivasamurthy, A. Ortega, and S. Narayanan. Towards optimal encoding for classification with applications to distributed speech recognition. *Eurospeech*, September 2003.

[72] N. Srinivasamurthy, A. Ortega, Q. Zhu, and A. Alwan. Towards efficient and scalable speech compression schemes for robust speech recognition applications. *IEEE Intl. Conf. on Multimedia*, July. 2000.

[73] M. Stricker and M. Orengo. Similarity of color images. *IS & T SPIE Proc. on storage and retrieval for image and video databases*, 2670, 1995.

[74] D. L. Swets and J. Weng. Using discriminant eigenfeatures for image retrieval. *IEEE Trans. on patter analysis and machine intelligence*, 18, 1996.

[75] R. R. Tenney and N. R. Sandell. Detection with distributed sensors. *IEEE Trans. on Aerospace & Electronic Systems*, 17:501–510, 1981.

[76] H. L. V. Trees. *Detection, Estimation and Modulation Theory.* John Wiely & Sons, 1968.

[77] J. N. Tsitsiklis. Extremal properties of likelihood-ratio quantizers. *IEEE Trans. on Communications*, 41:550–558, April 1993.

[78] V. Vapnik. *The nature of statistical learning theory.* Springer-Verlag, New York, 1995.

[79] N. Vasconcelos. Exploiting group structure to improve retrieval accuracy and speed in image databases. In *International conference on Image Processing*, 2002.

[80] N. Vasconcelos. On the efficient evaluation of probabilistic similarity functions for image retrieval. *IEEE Trans. on Information Theory*, pages 1482–1496, July 2004.

[81] N. Vasconcelos and A. Lippman. Feature representations for image retrieval: Beyond the color histogram. *IEEE International Conference on Multimedia and Expo*, 2:899–902, 2000.

[82] N. Vasconcelos and A. Lippman. A unified view of image similarity. In *In proceedings of international conference on pattern recognition*, Barcelona, Spain, 2000.

[83] J. Vass, J. Yao, A. Joshi, K. Palaniappan, and X. Zhuang. Interactive image retrieval over the internet. *IEEE Symposium on Reliable Distributed Systems*, 1:461–466, 1998.

[84] L. Vasudevan and A. Ortega. Processing-aware compression for sensor networks. *2nd IEEE Sensor Array and Multichannel Signal Processing Workshop*, Aug 2002.

[85] D. White and R. Jain. Similarity indexing with the ss-tree. *Proc. of ICDE*, 1995.

[86] H. Xie and A. Ortega. Feature representation and compression for content-based image retrieval. *VCIP*, 4310:111–122, Jan. 2001.

[87] H. Xie and A. Ortega. Entropy and complexity constrained classified quantizer design for distributed image classification. *Proc. of Multimedia Signal Processing (MMSP)*, 2002.

[88] H. Xie and A. Ortega. Exploration of linear discriminant analysis for transform coding in distributed image classification. *37th Asilomar Conference on Signals, systems, and Computers.*, 2003.

[89] H. Xie and A. Ortega. An user preference information based kernel for svm active learning in content-based image retrieva. *6th ACM SIGMM International Workshop on Multimedia Information Retrieval.*, 2004.

[90] B. Zhang and S. N. Srihari. Fast k-nearest neighbor classification using cluster-based trees. *IEEE Trans. on Pattern analysis and machine intelligence*, 2004.

[91] X. S. Zhou and T. S. Huang. Relevance feedback in image retrieval: A comprehensive review. *ACM Multimedia Systems Journal*, 2002.

Appendix A

Relations between Similarity Functions for Content-based Image Retrieval

Define a set of discriminant functions $g_i(x), i = 1, ..., M$. The classifier assign a feature vector x to class y_i, if

$$g_i(x) > g_j(x) \quad for \ \ all \ \ j \neq i. \tag{A.1}$$

The statistical classification can be viewed as a machine that computes M discriminant functions and outputs the classification label with the largest discriminant [22]. For Bayes classifier, $g_i(x)$ is the a posteriori probability $\Pr(y_i|x)$ or and monotonically increasing function of $\mathcal{F}(\Pr(y_i|x))$.

Define:

$$\begin{aligned} g_i(x) &= \log(\Pr(y_i|x)) \\ &= \log p(x|y_i) + \log P(y_i) \end{aligned} \tag{A.2}$$

A.1 Quadratic distance for Normal density functions

If the densities $p(x|y_i)$ are multivariate normal: $p(x|y_i) \sim N(\mu_i, \Sigma_i)$, then from (A.2), we have quadratic discriminant functions:

$$g_i(x) = -\frac{1}{2}(x - \mu_i)^t \Sigma_i^{-1}(x - \mu_i) - \frac{d}{2}\log(2 \times \pi) - \frac{1}{2}\log|\Sigma_i| + \log P(y_i) \tag{A.3}$$

A.1.1 Mahalanobis distance

If we assume that the covariance matrices are identical for all classes, and assume equal a priori probabilities, from (A.3) we get:

$$g_i(x) = -\frac{1}{2}(x - \mu_i)^t \Sigma_i^{-1}(x - \mu_i) \tag{A.4}$$

This is to say: To classify a feature vector x, compute the Mahalanobis distance $(x-\mu_i)^t\Sigma_i^{-1}(x-\mu_i)$, and find the nearest one. This similarity function was employed by QBIC [45] and MARS [42] for nearest neighbor retrieval. Furthermore if the covariance is identity $\Sigma_i = I$, we obtained an Euclidean distance $(x-\mu_i)^t(x-\mu_i)$

A.2 Kullback-Leibler divergence

If we assume that the query consists of a collection of N independent query features $x = \{x_1, ..., x_N\}$, the discriminant function in (A.2) can be written as:

$$g_i(x) = \frac{1}{N}\sum_{j=1}^{N} \log P(x_j|y_i) \tag{A.5}$$

Applying the law of large numbers we get:

$$\begin{aligned} g_i(x) &\xrightarrow{N\to\infty} E_q[\log \Pr(x|y_i)] \\ &= \int P(x|q)\log P(x|y_i)dx \\ &= -KL(Q||P_i) \end{aligned} \tag{A.6}$$

where $KL(Q||P_i)$ is the Kullback-Leibler divergence between the query density and that associated with the i^{th} image class.