JOINT APPEARANCE AND LOCALITY IMAGE REPRESENTATION BY
GAUSSIANIZATION

BY

XI ZHOU

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

Doctoral Committee:

Professor Thomas S. Huang, Chair
Associate Professor Mark A. Hasegawa-Johnson
Professor Stephen E. Levinson
Assistant Professor Feng Liang
Abstract

A novel image representation is proposed in this thesis to capture both the appearance and locality information for image classification applications. First, we model the feature vectors, from various granularity levels including the corpus level, the image level and image patch level, in a hierarchical Bayesian framework using mixtures of Gaussians. After such a hierarchical Gaussianization, each image is represented as a Gaussian mixture model (GMM) for its appearance, and several Gaussian maps for its spatial layout. Then we extract the appearance information from the GMM parameters, and the locality information from the global and the local statistics over Gaussian maps. Finally, we employ a supervised dimension reduction technique called DAP (discriminant adaptive projection) to remove noise directions and to further enhance the discriminating power of our representation.

To validate the argument that the new representation is a general representation for images and video frames, we evaluate the representation on several important applications. Firstly, we apply the new presentation to classification and regression tasks taking whole images as inputs. These tasks include object recognition, scene category classification, face recognition, age estimation, pose estimation, gender recognition, and video event recognition. Then we test it for the object detection and image parsing tasks, where the new representation takes partial images as inputs. The experimental results show that, for various types of images and tasks, the performances using the proposed representation were the best in all the applications compared with other state-of-the-art algorithms.
To Na
I am grateful to all the members of my Ph.D. thesis committee, and especially to my two advisers, Professor Thomas S. Huang and Professor Mark Hasegawa-Johnson, for all their help and advice during my studies at UIUC. Over the past four years it has been my great honor to collaborate with (in alphabetical order) Jui-Ting Huang, Zhen Li, Ming Liu, Hao Tang, Minqiang Xu, Shuicheng Yan, and Xiaodan Zhuang.

Many people have contributed in various ways to make this Ph.D. study an exciting and memorable journey. I owe sincere gratitude to Jiri Navratil, Jason W. Pelecanos, Kai Yu, Jinjun Wang, Fengjun Lv, Yihong Gong, and members of the UIUC IFP Lab. I also thank all my friends, near and far, for their support and care.
# Table of Contents

List of Tables .............................................................................................................. viii
List of Figures ............................................................................................................... ix

Chapter 1  Introduction ................................................................................................. 1
  1.1  Background ........................................................................................................... 1
  1.2  Classification Tasks ............................................................................................. 3
    1.2.1  Scene image .................................................................................................. 3
    1.2.2  Facial image classification .......................................................................... 4
    1.2.3  Video classification ..................................................................................... 6
  1.3  Novel Hierarchical Gaussianization Representation ........................................ 7
  1.4  Contributions and Outline .................................................................................. 9

Chapter 2  Related Work ............................................................................................... 11
  2.1  Local Features .................................................................................................... 11
    2.1.1  Scale-Invariant Feature Transform ............................................................. 12
    2.1.2  Coordinate patches ..................................................................................... 14
  2.2  Comparing Distributions of Local Features ....................................................... 15
    2.2.1  Spatial pyramid matching for combining spatial information ................. 15

Chapter 3  Hierarchical Gaussianization Representation ............................................. 17
  3.1  Flow Chart of Hierarchical Gaussianization Representation ............................ 17
  3.2  GMMs for Appearance Representation .............................................................. 18
    3.2.1  Histogram as a special case of GMMs ....................................................... 21
  3.3  Similarity of Distributions .................................................................................. 22
    3.3.1  GMM mean supervector ............................................................................. 22
  3.4  GMM Mean-Covariance Supervector .................................................................. 23
  3.5  Gaussian Maps for Spatial Representation ....................................................... 26
    3.5.1  SPM as a special case of Gaussian maps ................................................... 27
  3.6  Discriminant Adaptive Projection ...................................................................... 28
  3.7  Fast Gaussianization by Random Forest ............................................................ 29
    3.7.1  Building the random forest ....................................................................... 31
    3.7.2  Obtaining the top Gaussians ..................................................................... 33
Chapter 8  The Application of HG in Image Parsing  81
8.1  Introduction  81
8.2  Graph Adaptive Dynamic Programming for Multi-Hypotheses Generation  84
  8.2.1  Graph adaptive dynamic programming  85
8.3  Implementation Detail  89
8.4  Experimental Results  90
  8.4.1  Object segmentation performance  91
  8.4.2  Object detection performance  92
  8.4.3  Image classification performance  93

Chapter 9  Conclusion  96

References  98
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>The influence of codebook sizes (</td>
<td>C</td>
</tr>
<tr>
<td>4.2</td>
<td>Comparison of our method with top performers in PASCAL VOC 2007.</td>
<td>39</td>
</tr>
<tr>
<td>4.3</td>
<td>Comparison of our method with top performers in PASCAL VOC 2009.</td>
<td>40</td>
</tr>
<tr>
<td>4.4</td>
<td>Performance comparison on Caltech101 (single descriptor).</td>
<td>42</td>
</tr>
<tr>
<td>4.5</td>
<td>Performance comparison on scene category database.</td>
<td>44</td>
</tr>
<tr>
<td>4.6</td>
<td>The classification results on scene category database.</td>
<td>44</td>
</tr>
<tr>
<td>5.1</td>
<td>The recognition error rates of the proposed HG representation on the PIE dataset compared with the state-of-the-art.</td>
<td>50</td>
</tr>
<tr>
<td>5.2</td>
<td>Face recognition error rate (%) on PIE database.</td>
<td>50</td>
</tr>
<tr>
<td>5.3</td>
<td>MAEs of different algorithms on the YGA database.</td>
<td>54</td>
</tr>
<tr>
<td>5.4</td>
<td>MAEs of different algorithms on the FG-NET aging database. Note that (BM) below signifies the bilinear model used in [Yan 2007a].</td>
<td>55</td>
</tr>
<tr>
<td>5.5</td>
<td>MAEs of the algorithms PCA, LEA, SSE and RPK on the CHIL data from the CLEAR07 evaluation.</td>
<td>60</td>
</tr>
<tr>
<td>6.1</td>
<td>Comparison of Average Precision (%) using different algorithms. Note that: (1) TAPM-1 is the TAPM algorithm with same weights for all the three levels; [Xu 2008] (2) TAPM-2 refers to the TAPM algorithm with different weights for the three levels; [Xu 2008] (3) Hist+SVM refers to Bag-of-Words quantization with SVM classification; (4) HG+NN is the algorithm based on HG representation and Nearest Neighbor classifier; (5) HG+SVM means HG representation with SVM classification; (6) HG+NC refers to the centroid-based algorithm using DAP; and (7) NC+SVM refers to the algorithm based on the fusion of two classifiers. The last row, referred to as Mean AP, is the mean of APs over ten events.</td>
<td>64</td>
</tr>
<tr>
<td>8.1</td>
<td>Object segmentation results.</td>
<td>92</td>
</tr>
<tr>
<td>8.2</td>
<td>Object detection results.</td>
<td>93</td>
</tr>
<tr>
<td>8.3</td>
<td>Object classification results.</td>
<td>95</td>
</tr>
</tbody>
</table>
List of Figures

2.1 An illustration of the three image representations: (a) holistic image, where an appearance feature is assigned for each fixed coordinate; (b) coordinate patches, where certain appearance features may appear in a flexible area, and the attached ball for each local patch means that the coordinate of the patch is changeable; and (c) free patches, where coordinate information is discarded entirely [Lucey 2004].

3.1 (a) is an input image. (b) shows the patch features in the feature space. Each ”+” denotes a feature vector, whose distribution is approximated by a GMM. (c) shows a set of Gaussian maps, each of which corresponds to one Gaussian component in (b). A supervised dimension reduction algorithm, DAP, is performed in (d) to form the final image representation, hierarchical Gaussianization vector.

3.2 Sample Gaussian maps of three images from the Caltech 101 dataset.

3.3 A decision tree.

4.1 Example images from the PASCAL VOC 2007 database.

4.2 Visualization of the learned patch-level function on image examples from PASCAL-09.

4.3 Example images from the scene category database.

4.4 Confusion matrix on scene category database for the HG representation. The average classification accuracy is 88.4%. The entry in the $i^{th}$ row and $j^{th}$ column is the percentage of images from class $i$ that were misidentified as class $j$. For better viewing, please see the pdf file.

4.5 Example images from the scene category database for the HG representation.

5.1 Example images from the CMU PIE database. For each subject, there are 170 near frontal face images under varying pose, illumination, and expression.

5.2 The performance comparison of state-of-the-art face recognitions on LFW dataset.

5.3 Sample images of one person in the FG-NET database.

5.4 Cumulative scores of human age estimation results for QM, MLP, RUN [Yan 2007b], and RPK algorithms at error levels from 0 to 20 years on the two configurations of the YGA database.

5.5 Cumulative scores of age estimation for the QM, MLP, RUN [Yan 2007b], and RPK algorithms at error levels from 0 to 10 years on the FG-NET aging database.
5.6 Comparison confusion matrices on the YAMAHA-Female subset for the original RPK, RPK with free-patches, RPK without kernel regression, and RPK without inter-modality similarity synchronization (IMSS). For better viewing, please see the pdf file. .................................................. 57

5.7 MAEs of RPK with occlusions of different sizes. The blue dashed line denotes the best result reported in [Yan 2007b] without image occlusions. ..................... 58

5.8 Gender recognition result on YGA database. The horizontal axis stands for the percentage of data used for training. The vertical axis represents the average accuracy of 10 runs for gender recognition. As illustrated, the performance of our HG representation achieved more than 40% relative error reduction compared with GMM representation and SVM classifier based on holistic features. .............. 59

6.1 Visualization of SIFT-Bag in discriminating power and the capability of matching objects different in spatial positions, scales, and temporal positions. For better viewing, please see the color pdf file. ................................. 66

6.2 Comparison of confusion matrices for HG + Nearest Neighbor, HG + SVM, HG + DAP, and HG + fusion of WCCN and SVM. Note that the first value in the title is the mean average precision, and the second value is the overall recognition accuracy. For better viewing, please see the original pdf file. ......................... 67

6.3 The comparison of mean average precisions of different algorithms using randomly sampled 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, and 100% percentages of the frames within each test video clip. ........................................ 68

7.1 Sample images in the multi-scale car dataset. ........................................... 77

7.2 ROC curves for multi-scale car detection. (“G-n” denotes the result using n components in the Gaussianized vector representation. “Histogram” denotes the performance using the generic histogram-of-keywords approach by Lampert et al.) .... 78

7.3 Equal error rates for multi-scale car detection. ..................................... 79

7.4 Examples of good and bad localization based on Gaussianized vector representa-

7.4 Examples of good and bad localization based on Gaussianized vector representa-

7.5 Examples of good and bad localization based on Gaussianized vector representa-

8.1 Graph adaptive dynamic programming (GADP). Given (a) a test image, we group pixels into regions to form (b) a region graph according to adjacent relationship and apply GADP approach to get (c) multiple candidates, then we adopt more precise model and classifier to obtain (d) final output. ........................................ 84

8.2 Illustration of proposed GADP. The original image shown in Figure 1 is converted into region graph (a) where the center of each region is highlighted in red, and connections are highlighted in blue. Then GADP is proposed to solve graph structural decision problem (b), where the observed node is highlighted in green, and the current node is highlighted in star. The simple N-top approach gives almost identical solutions (c), while the proposed strategy gives very different results (d) that covers larger region in the search space. ................................. 86

8.3 Example segmentation results. ........................................ 91

8.4 Example detection results that improve most over [Everingham 2009]. .......... 93
8.5 Example detection results without significant improvement. . . . . . . . . . . 93
8.6 Examples of classification ranking. . . . . . . . . . . . . . . . . . . . . . . . . . . 94
Chapter 1

Introduction

1.1 Background

Image classification, which aims to categorize the whole or a part of the input image into one of the predefined categories, has been among the most important problems in computer vision. Based on the types of the input images and the predefined categories, image classification can be grouped into several applications, such as face recognition, object recognition, and scene category classification etc.

Despite decades of active research, the performance of image classification applications is far from perfect, especially under uncontrolled real-world conditions. One of the major difficulties comes from the huge variability of natural images that arises from viewpoint and lighting changes, movement and deformation of non-rigid or articulated objects, intraclass appearance variations, the presence of occlusion, and the quality and resolution of the images. Those variabilities make the distribution of each image class highly nonlinear and complicated, which increases the difficulty to obtain a compact image representation.

Another common difficulty that arises in the task of image classification is to find correspondences among multiple images. That is, how do we match the corresponding feature points between image pairs? Many dimensionality reduction techniques, including the global linear transformation methods such as Principal Component Analysis (PCA) [Jolliffe 2002] and Fisher’s Linear Discriminant Analysis (LDA) [Fisher 1938] as well as the manifold learning methods such as Locally Linear Embedding (LLE) [Roweis 2000] and Locality Preserving Projections (LPP) [He 2003a], require well-corresponded feature points between the images to seek a meaningful
low dimensional subspace. For various classifiers based on certain distance metrics in the feature space, such as the Nearest Neighbor (NN), correspondences are critical, too. For instance, it is meaningless to compute the distance between the nose tip point in one face image and the left eye corner point in another face image. The challenges of finding correspondences between images are at least twofold: First, in many cases the images undergo certain unknown transformations (e.g., rotation, affine, etc.) and the features extracted from these images are correspondingly distorted. Although such distortions can be somehow compensated by adopting features relatively invariant to the transformations, such as the Scale Invariant Feature Transform (SIFT) descriptor, it is unlikely to reverse the effect of the unknown transformations. Second, the order of the extracted feature vectors are partially, if not completely, unknown. This makes the task of finding two corresponding feature vectors in two different images extremely difficult.

Moreover, the efficiency of an algorithm becomes critical when the size of a dataset grows exponentially. Some newly developed image datasets, such as the imagenet, contain millions of images with hundreds of image categories. The number of images may explode when we aim to apply our algorithms to some real-world scenarios. For example, we may need to organize the billions of images on flickr or apply a face recognition algorithm to videos captured by a surveillance system, where we need to recognize in real-time the faces from millions of people.

Numerous research efforts have been proposed to solve the above problems. Many algorithms have achieved good performance at solving some specific variabilities for some particular types of images. However, it is not straightforward to apply them to other types of variabilities and other kinds of images. For example, the sparse representation [Wright 2008] can deal with the face recognition with illumination changes and partial occlusion quite well, but it has difficulties with the nonlinear variability, like mis-alignment and expression changes. Moreover, the alignment requirement prevents its usage on scene images, and furthermore, this approach is computationally expensive and does not scale well to large dataset problems.

Undoubtedly, an efficient and effective image representation plays an essential role in many image classification systems. It would be nice if we can develop an image representation with the
following properties:

- It is robust to most of the aforementioned variabilities
- It is able to give the correspondence across different images
- It is efficient enough for large scale and real-time applications
- It is a consistent representation of all kinds of images

In the following sections, we briefly review the popular methods for different image classification applications. We then introduce our novel image representation with the properties mentioned above.

1.2 Classification Tasks

According to the type of the input images, we group the classification tasks into three categories, i.e. classifications on: scene images, facial images and video clips.

1.2.1 Scene image

The important classification tasks on scene images include object recognition and natural scene classification. The object recognition aims to recognize the object categories in the scene images, such as car, person, chair, etc. The problem of natural scene classification is to recognize semantic scene categories such as beach, mountain, office, etc. The performance of both tasks are heavily affected by all kinds of variabilities and the correspondence on scene images is an extremely challenging problem.

Histogram representations, as the descriptions for orderless patch-based features, have been widely used in object recognition and natural scene classification [Schiele 2000, Swain 1991, Fei-Fei 2005]. Despite its popularity, histogram representation has some limitations. For example, it is sensitive to several factors such as outliers, the choice of bins, and the noise level in
data. Most importantly, encoding high-dimensional feature vectors by a relatively small codebook is vulnerable to large quantization errors and may lose discriminability [Boiman 2008]. Furthermore, histogram representation discards all the spatial configuration of image patches, which is an indispensable attribute for object and scene classification.

Several approaches have been proposed in the literature to overcome these limitations. Soft assignment, which allows each feature vector assigned to multiple histogram bins, has been suggested to capture partial similarities between images [Perronnin 2006, Yang 2008, van Gemert 2008, Agarwal 2006, Tuytelaars 2007, Philbin 2008]. To enhance the discriminating capability of histograms, Farquhar et al. [Farquhar 2005] and Perronnin et al. [Perronnin 2006] introduced several ways to construct category-specific histograms, Larlus et al. [Larlus 2006] and Yang et al. [Yang 2008] suggested to integrate histogram construction with classifier training, and Moosmann et al. [Moosmann 2007] proposed to use randomized forests to build discriminative histograms. As a flexible way to model a variety of distributions, GMM emerged as a better alternative to histograms in age estimation, object classification and video event analysis [Yan 2008b, Liu 2008, Zhou 2008a]. On the other hand, to alleviate the loss of spatial information in histogram representation, one of the most successful approaches by far is the spatial pyramid matching (SPM) technique proposed by Lazebnik et al. [Lazebnik 2006a].

1.2.2 Facial image classification

A facial image may encode many human characteristics, e.g., identity, expression, gender, ethnicity, age, and pose. Among them, the task of recognizing person identity, i.e., face recognition, has been one of the most classic tasks in computer vision for decades. Over the past 20 years, many studies have attempted to extract face image features which are robust to the aforementioned variabilities. Among appearance-based holistic approaches, eigenfaces [Turk 1991] and Fisherfaces [Belhumeur 1996] have proved to be effective in experiments using large databases. In [Wright 2008], sparse representation was applied to face recognition on well-aligned frontal face images and achieved good performance. However there are some limitations of this method. First,
this approach requires perfect face alignment, which is a challenging task not solved in practice. Second, like many conventional face recognition algorithms, this approach cannot handle cross-pose and cross-expression face recognition. Thirdly, this approach is computationally expensive and does not scale well to large dataset problems.

Recently, the study of recognizing other human characteristics has been more and more popular. Geng et al. [Geng 2006, Geng 2007] proposed to conduct age estimation by modeling the statistical properties of the aging pattern, namely a sequence of personal facial age images, based on the assumption that multiple images of different ages are available for each person. Recently, Yan et al. [Yan 2007b] proposed an algorithm based on semi-definite programming for age estimation, with allowance made for uncertainty in the reference age labels. Gender recognition from face images has attracted much research attention over last two decades. Early works were mostly based on k-nearest-neighbor and neural network [Tamura 1996, Gutta 2000], where satisfactory results were reported. Since 2000, SVMs [Moghaddam 2002, Lian 2005] and boosted classifiers (AdaBoost [Baluja 2007] or SODABoost [Xu 2007b]) were introduced to handle this problem. They are currently dominating because of their robustness and good generalization performance. The pose estimation problem has also attracted much attention [Fu 2006, Balasubramanian 2007, Raytchev 2004] in recent years owing to its great potential in practical systems.

Most previous algorithms for those applications are based on holistic image features, but holistic features are sensitive to illumination variations and image occlusions. Lucey et al. [Lucey 2004] demonstrated that face verification may benefit from the free-patch based representation, which has the potential to overcome these issues. The human age and head pose problems are, however, beyond the solution proposed in [Lucey 2004] for two reasons. First, the free-patch representation discards the coordinate information, which has been proved to be necessary for accurate pose estimation [Gee 1994]. Second, the algorithm in [Lucey 2004] is limited in addressing classification problems rather than regression problems, and its discriminating power may be greatly degraded if large within-class variations exist.
1.2.3 Video classification

Video based event recognition is an extremely challenging task due to all kinds of within-event variations, such as unconstrained motions, cluttered backgrounds, object occlusions, environmental illuminations and geometric deformations of objects. While there exist works attempting to detect unusual or abnormal events [Zhang 2005, Boiman 2007] in video clips, the research on event recognition in unconstrained real-life video is still at its preliminary stage [Efros 2003, Hauptmann 2006]. We define the scope of this study as recognition of pre-defined events based on the visual cues encoded in unconstrained video, e.g. broadcast news video, as in [Xu 2007a, Xu 2008].

Many statistical models, e.g., Hidden Markov Model (HMM) [Peursum 2003], coupled HMM [Brand 1997], and Dynamic Bayesian Network [Oliver 2000], were proposed to capture the spatial and temporal correlations of video events. The learned models are applied to pre-defined video event classification or abnormal event detection. On the other hand, appearance-based techniques were also widely used for video event detection and classification. Ke et al. [Ke 2005] applied the boosting procedure for choosing the volumetric features based on optical flow representations. Niebles et al. [Niebles 2006] adopted the spatio-temporal interest points [Dollar 2005] to extract the features, and other works [Dollar 2005, Laptev 2003, Schuldt 2004] extracted volumetric features from salient regions [Harris 1988, Laptev 2003]. There also exist works that used bag-of-words model to tackle the problem of object/event recognition [Sivic 2003, Quelhas 2007]. In addition, Bagdanov et al. [Bagdanov 2007] adopted bag-of-SIFTs to detect and recognize object appearances in videos.

Most previous research on video event analysis is limited to videos captured by fixed cameras in surveillance applications or greatly constrained live videos. More challenging is video event recognition in unconstrained domains such as broadcast news, which contains rich information about objects, people, activities, and events [Xu 2008]. For example, events in broadcast news video may involve small objects, large camera motion, and significant object occlusion. The reli-
able object tracking becomes very challenging.

Some recent research attempted to provide solutions for event analysis in news video. Ebadoollahi et al. [Ebadollahi 2006] proposed to treat each frame in a video clip as an observation and apply HMM to model the temporal patterns of event evolution in news video. Xu and Chang [Xu 2007a] proposed to encode a video clip as a bag of orderless descriptors obtained from mid-level semantic concept classifiers extracted from all of the constituent frames, along with the global features extracted within each video frame, and then apply the Earth Mover’s Distance (EMD) [Rubner 2000a] to integrate similarities among frames from two video clips. Multi-level temporal pyramid structure was adopted to integrate the information from different sub-clips with integer-value constrained EMD to explicitly align the sub-clips.

1.3 Novel Hierarchical Gaussianization Representation

In a typical image categorization/recognition system, it is common practice and advantageous to form a vector representation of an image so that subsequent steps in the categorization/recognition pipeline (e.g., metric learning, classification, etc.) can be conveniently performed. Recently, patch-based descriptors (or called local features) have been successfully used for image classification, indexing and retrieval application [Csurka 2004, Kadir 2001, Lowe 2004].

To take advantage of patch-based descriptors and vectorized representation, the “bag-of-features” (BoF) representation [Csurka 2004] is proposed, in which a number of possibly overlapping patches are extracted from an image and a histogram is computed to represent the distribution of these patches. However, it suffers from a serious problem — by forming the histogram of the free patches, the locality information in the image is completely lost. To overcome this, in our recent work [Yan 2008b], we propose a state-of-the-art image representation based on the “localized” patches. We augment the free patches with their location coordinates to form the coordinate patches (or “xy-patches”). The coordinate patches integrate both appearance and coordinate information to provide a localized representation that is informative about the holistic structure of the
image. In addition, we extend the BoF representation based on spatial pyramid matching (SPM) [Lazebnik 2006a]. The SPM preserves the spatial order of the local descriptors and increases the descriptive power of the image representation. However, it has come to our attention that the coordinate patches and SPM encode the locality information in an image in a “hard” way. Thus, the image representation may not deal with noise and outliers very well. A new, better image representation is desired.

Therefore, we aim at developing a novel image representation based on a conceptually new concept of hierarchical Gaussianization (HG). Specifically, in the HG image representation, a Gaussianized vector is formed for an image through a process of multi-level, coarse-to-fine summarization of the image features. The two core parts of the HG image representation are the Gaussian mixture model (GMM) for appearance modeling and Gaussian maps for locality modeling.

We estimate a GMM for the distribution of all feature vectors of each image. The reason to use a GMM to characterize the appearance features is twofold. First, the estimated GMM is a compact description of the underlying distribution of all appearance feature vectors. With the increasing number of components, the GMM can be arbitrarily accurate in describing such a complicated distribution. The estimated GMM is less prone to variabilities, compared with the feature vectors themselves. Second, although explicit correspondence between appearance feature vectors is not pursued in this framework, the Gaussian components in GMM impose an implicit multi-mode structure of the feature vector distribution in an image. The corresponding Gaussian components in two video clips may imply certain spatiotemporal correspondence, particularly when the GMMs for different images are adapted from the same global Gaussian Mixture Models as described afterwards.

A Gaussian map (GM) is a posterior probability map that emphasizes certain local structures of an image. Since the local structures of an image are revealed by unsupervised segmentation of the image by the HG, and the emphasis is expressed in terms of probabilities evaluated at the individual Gaussians, the Gaussian maps incorporate the locality information in the image into the image representation in a “soft” way. Compared to the hard encoding of the locality information in
the coordinate patches and SPM, this soft encoding method is a key factor to make the new image representation robust to noise and outliers.

To enhance the discriminating power of our representation, we propose to project our Gaussianized vector to a subspace that depresses the directions with high inter-category variabilities. The generative followed by the discriminative model structure balances the effectiveness and efficiency of our representation. We further introduce a random forest based algorithm for efficient computation of the Gaussianized vectors, which ensure the real-time usage of our new representation.

The HG image representation is generic and thus independent of the specific categorization/recognition task. A large variety of image features (e.g. pixel intensities, Gabor wavelet responses, SIFT, etc.) can be encoded by the HG image representation, and can be applied to a broad range of practical tasks including detection, recognition, regression, etc. Furthermore, the computational cost of the HG image representation can be dramatically reduced by either approximation algorithms such as random forest (without losing performance), or parallel implementations based on CPU or GPU clusters. In this way, the HG image representation scales very well to large datasets and is suitable for use in large-scale real-world applications such as image and video retrieval.

1.4 Contributions and Outline

The main advantages of our novel HG representation can be summarized as follows:

1. We apply the new representation on face images, scene images, and video clips, respectively, and further demonstrate the effectiveness of the new representation on several different applications. Experimental results show that our performance ranks among the top in all the applications.

2. Taking advantage of the local descriptors and GMM, the new representation is robust to most of the variabilities in the image appearance, such as illumination variations, pose changes, etc; we introduce the Gaussian maps to capture the locality information, which alleviates the bottleneck of the patch-based description.
3. We introduce a supervised learning method, Discriminant Adaptive Projection (DAP), to further enhance the discriminating power of our representation.

4. The computational cost of our new representation can be dramatically reduced by either using the approximation algorithms such as random forest (without losing performance), or by parallel implementations based on CPU or GPU clusters. In this way, our system scales very well to large datasets and is suitable for large-scale real-time applications.

5. The new representation is generic and may be used with a variety of features (e.g., pixel intensities, Gabor wavelet responses [Daugman 1988], SIFT features [Lowe 1999], etc.) in a broad range of tasks (detection, recognition, classification, regression, etc.).

The rest of this dissertation is organized as follows. Chapter 2 reviews existing literature on local image features and similarity measurement. Chapter 3 introduces the new image representation that incorporates both the appearance and locality information. From Chapter 4 to Chapter 6, we describe the feature, the classifier and the performance when applying the novel representation to scene image classification, facial image classification and video clip classification applications respectively. Chapter 6 introduces the use of our new representation on image detection tasks. Finally, Chapter 7 summarizes the contribution of the dissertation and discusses possible future research directions. The work described in this dissertation has been previously published in [Zhou 2008a, Yan 2008b, Zhou 2008b, Zhou 2009, Zhuang 2008, Zhuang 2009, Li 2009].
Chapter 2

Related Work

This chapter reviews the existing works on describing different image signatures and similarity measures.

2.1 Local Features

Using local features to describe images has several important advantages: first, as one image contains hundreds of local features and the dimensions of local features are relatively small, it is easier for us to estimate the distribution of local features than that of global features; secondly, local features can be preserved even when the input image is affected by clutters or occlusions as they are relatively small and compact. Thirdly, depending on the requirements of a particular application, we can choose to use local features invariant to different variabilities.

Therefore, the local features have become more and more popular in image classification applications in recent years and many different techniques for describing local image regions have been developed. Here we briefly summarize those descriptors; for detailed comparison, refer to [Mikalajczyk 2005].

The distribution based descriptors use histograms to represent different characteristics of appearances or shapes. One simple descriptor is the distribution of the pixel intensities represented by a histogram. A more expressive representation was introduced by Johnson and Hebert [Johnson 1997] for 3D object recognition in the context of range data. Also an illumination invariant descriptor is developed by Zabih and Woodfill [Zabih 1994], which is suitable for texture representation. Lowe proposed a scale invariant feature transform (SIFT) based on the gradient distribution, which has
become the most popular local descriptor, and we will describe it in detail in the following section. The geometric histogram [Ashbrook 1995] and shape context [Belongie 2002] are similar to the SIFT descriptor, and also compute a histogram of location and orientation for edge points where all the edge points have equal contributions in the histogram.

Another kind of descriptor is the differential based descriptor, where a set of image derivatives computed up to a given order approximates a point neighborhood. Koenderink [Koenderink 1987] studied the properties of local derivatives (local jet), and Florack et al. [Florack 1991] derived differential invariants, which combine components of the local jet to obtain rotation invariance. Freeman and Adelson in [Freeman 1991] developed steerable filters, in which steer derivatives in a particular direction give the components of the local jet. A stable estimation of the derivatives is obtained by convolution with Gaussian derivatives.

The spatial-frequency descriptors, which are based on the frequency content of an image, have been popular for a long time. They usually use Fourier transform to decompose the image content into basis functions. One of the most popular spatial-frequency based descriptors is the Gabor transform [Gabor 1946].

No single feature can be the best image description for every task. We need to choose different descriptors for different applications. In the following section, we describe in detail two descriptors, Scale-Invariant Feature Transform (SIFT) and coordinate patches, which are the major descriptors used in the later experiments.

2.1.1 Scale-Invariant Feature Transform

Scale-Invariant Feature Transform (SIFT) [Lowe 1999] is a widely used algorithm to detect and describe salient local features within an image. The SIFT features are local and based on the appearance at particular interest points, and are invariant to image scale and rotation. They are also robust to changes in illumination, noise, minor changes in viewpoint, as well as occlusion. The SIFT features can be used for image matching, which is useful for object tracking and 3D scene reconstruction, and they are application-independent. In addition to these properties, they
are highly distinctive, relatively easy to extract, and allow for correct object identification with low probability of mismatch.

The extraction of SIFT features consists of four major steps: (1) scale-space extrema detection, (2) keypoint localization, (3) orientation assignment, and (4) keypoint descriptor. The first step identifies potential keypoints from all locations and scales of the image. In the second step, candidate keypoints are localized to sub-pixel accuracy and eliminated if found to be unstable. The third step identifies the dominant orientation(s) for each keypoint based on the histogram of gradient in its local image patch. The assigned orientation, scale and location for each keypoint enable SIFT to construct a canonical view for the keypoint, invariant to affine transforms. The final step builds a local image descriptor for each keypoint, based upon the histogram of gradients adjusted by the dominant orientation(s). For each keypoint, the SIFT descriptor divides a square patch into a $4 \times 4$ grid and computes a histogram of gradient orientations in each subregion. Eight gradient orientations are used, resulting in a 128-dimensional feature vector. Histogramming provides stability against deformations of the image pattern, while subdividing the support region offsets the potential loss of spatial information. In this way, a compromise is achieved between the conflicting requirements of greater geometric invariance on the one hand and greater discriminative power on the other. Intuitively, descriptors based on this compromise should be simultaneously richer and more robust than filter banks and differential invariants, which are functions of the entire region of support. Indeed, in a recent comparative evaluation [Mikolajczyk 2005], SIFT descriptors decisively outperform these more traditional methods.

For the scene image classification problem, the SIFT feature vectors are detected and extracted for each image, from which the so-called Bag of SIFT is constructed. Then our HG representation is used to transform length-variant orderless SIFT features into a fixed length supervector, and then the conventional machine learning algorithms can be applied based on this fixed length representation.
Figure 2.1: An illustration of the three image representations: (a) holistic image, where an appearance feature is assigned for each fixed coordinate; (b) coordinate patches, where certain appearance features may appear in a flexible area, and the attached ball for each local patch means that the coordinate of the patch is changeable; and (c) free patches, where coordinate information is discarded entirely [Lucey 2004].

2.1.2 Coordinate patches

Most previous algorithms for face image classification tasks are based on holistic image features, and hence are sensitive to illumination variations and image occlusions. In contrast to a holistic image representation, a patch-based image representation has the potential to overcome these limitations [Lucey 2004, Lucey 2006].

In this work, we introduce a local descriptor for image representation called the coordinate patch. Lucey et al. [Lucey 2004] proposed to encode each image as an ensemble of free patches, containing no information about patch coordinates. Unlike free patches, coordinate patches integrate both appearance information and coordinate information, in order to provide a local representation that is informative about the holistic structure of the image. We propose that coordinate patches are useful when the discriminative power of an image feature depends on its location in the image; e.g., humans may perform precise age estimation tasks not only by observing wrinkles, but by observing the location of wrinkles on the face; likewise, head pose may be estimated from the coordinate information of the nose-tip patch. An illustration of three types of image representations is shown in Figure 2.1.

For a position within the image plane, denoted as $q = (q_x, q_y)^T$, its corresponding coordinate
patch for a given image $x_i$ is defined as

$$Q(x_i, q) = \begin{bmatrix}
R(x_i, q) \\
q
\end{bmatrix},$$

(2.1)

where $R(x_i, q)$ denotes the feature vector extracted from the image $x_i$ within the rectangle centered at the position $q$. In this work, to compute $R(x_i, q)$, we first remove the mean of the intensity values within the rectangle, then normalize the intensities to unit variance, and finally use the 2D discrete cosine transform to extracting the final feature vector $R(x_i, q)$. Thus the coordinate patch is relatively robust to illumination variations.

### 2.2 Comparing Distributions of Local Features

After computing the descriptors of images as described in the previous section, we need to represent their distributions in the training and test images. One method for doing this is to cluster the set of local descriptors. Histogram representation has been widely used to describe the orderless local descriptors. The histogram bins are generated by unsupervised algorithms such as k-means. Then the histogram is calculated for each image and finally the distance between images can be computed by comparing the corresponding histograms. Another method to compare distributions of local features is using Earth Mover’s Distance (EMD) [Rubner 2000b], which has also been shown to be very suitable for measuring the similarity between image descriptors.

#### 2.2.1 Spatial pyramid matching for combining spatial information

Both histogram and the EMD representation based on orderless descriptors have not taken the spatial information into consideration. One of the successful approaches to alleviate the problem is the spatial pyramid matching proposed by Lazebnik et al. [Lazebnik 2006a].

Spatial pyramid matching attempts to compute the rough geometric correspondence on a global
scale. It repeatedly subdivides the image and computes histograms of local features at increasingly fine resolutions. Then it takes a weighted sum of the number of matches that occur at each level of resolution and, usually, matches found at finer resolutions are weighted more heavily than matches found at coarser resolutions.
Chapter 3

Hierarchical Gaussianization Representation

In this chapter, we introduce our novel image representation based on a new concept of hierarchical Gaussianization (HG). Specifically, the HG image representation, which captures both appearance information and locality information, forms a Gaussianized vector for each image through a process of multi-level, coarse-to-fine summarization of the image features. In the following sections, we first describe the flow chart of the new representation, and then introduce the components of the image representation respectively.

3.1 Flow Chart of Hierarchical Gaussianization Representation

We propose to develop a novel image representation to capture both the appearance and locality information. Figure 3.1 shows the flow chart of the new representation, where we adopt a hierarchical summarization process for feature vectors at difference levels: the whole corpus, each image and individual patches. First, each image is encoded as an ensemble of overlapped patches (Figure 3.1.a). The global distribution of the patches for the whole corpus is modeled by a Gaussian Mixture Model (GMM) (Figure 3.1.b). We learn an image-specific GMM through maximum a posterior criterion and then extract the appearance information from the parameters of image-specific GMM. Given an image-specific GMM, each patch of the image is softly assigned to a Gaussian component with respect to a posterior probability. Furthermore, for each Gaussian, the posterior probabilities of all the patches form a map over the patch locations, which we refer to
as a Gaussian map. We propose to extract the locality information from global and local statistics over Gaussian maps (Figure 3.1.c). Finally, we employ a supervised dimension reduction scheme, Discriminant Adaptive Projection (DAP), to further enhance the discriminating power of the representation (Figure 3.1.d) and get the Hierarchical Gaussianization (HG) image representation which we call a “supervector” (Figure 3.1.e).

Figure 3.1: (a) is an input image. (b) shows the patch features in the feature space. Each “+” denotes a feature vector, whose distribution is approximated by a GMM. (c) shows a set of Gaussian maps, each of which corresponds to one Gaussian component in (b). A supervised dimension reduction algorithm, DAP, is performed in (d) to form the final image representation, hierarchical Gaussianization vector.

The important components in our new representation include: GMMs for appearance representation, Gaussian maps for locality representation, image similarity calculating by computing model similarity, and Discriminant Adaptive Projection. We introduce a fast Gaussianization algorithm based on random forest to make our representation ready for large-scale real-time applications.

### 3.2 GMMs for Appearance Representation

While single mode distribution, like the Gaussian distribution, has some important analytical properties, it suffers from significant limitations when it comes to modeling real world data sets. To be able to model more complicated distributions which have multiple modes, we adopt the most
popular mixture model, GMM.

GMMs are widely used in data mining, pattern recognition, machine learning, and statistical analysis. As well as providing a framework for building more complex probability distributions, mixture models can also be used for data clustering. Usually, we can find maximum likelihood solutions for GMMs by expectation-maximization (EM) algorithm [Dempster 1977].

However, in our case, instead of separately estimating a GMM for each image, we estimate image-specific GMM by adapting from a global GMM. It is necessary and desirable, because (1) the number of the feature vectors extracted from one image is relatively small and insufficient for robust estimation of a GMM even in moderate scale; and (2) image-specific GMM adapted from the same global GMM tends to directly offer the correspondence between the Gaussian components of two GMMs.

Here we denote $z$ as a $p$-dimensional feature vector from the $I$-th image and model $z$ by a GMM, namely,

$$p(z|\Theta) = \sum_{k=1}^{K} w_{I}^I N(z; \mu_{I}^I, \Sigma_{I}^I),$$

(3.1)

where $K$ denotes the total number of Gaussian components, and $(w_{I}^I, \mu_{I}^I, \Sigma_{I}^I)$ are the image-specific weight, mean and covariance matrix of the $k$th Gaussian component, respectively. For computational efficiency, we restrict the covariance matrices $\Sigma_k^I$ to be a diagonal matrix $\Sigma_k$ shared by all images.

The number of model parameters $\Theta = \{w_{I}^I, \mu_{I}^I, \Sigma_k\}_{k=1:K, I=1:N}$ increases with respect to $N$, the number of training images. In practice the size of patches from one image is usually small and thus insufficient for a robust estimate of all parameters. To overcome this problem, we propose a hierarchical Bayesian framework to jointly estimate all the GMM parameters. We model the image-specific GMM parameters $w_{I}^I$'s and $\mu_{I}^I$'s by conjugate priors:

$$(w_{1}^I, \ldots, w_{K}^I) \sim \text{Dir}(Tw_1, \ldots, Tw_K),$$

$$\mu_{I}^I \sim N(\mu_k, \Sigma_k/r), k = 1:K.$$
The prior distribution over the weights $w_k^I$’s is a Dirichlet distribution with parameters $(T w_1, \ldots T w_K)$, which can be interpreted as adding total $T$ pseudo-counts with $w_k$ fraction of them from the $k$th component. The prior distribution for the mean $\mu_k^I$’s is a Gaussian centered at a global mean $\mu_k$ with a covariance matrix shrunk by a smoothing parameter $r$. Note that such a prior specification imposes dependence between images. And the rationale behind this is to “borrow” strength across similar images for estimation and therefore overcome the small sample size issue suffered in conventional learning processes.

We estimate the prior mean vector $\mu_k$, prior weights $w_k$ and covariance matrix $\Sigma_k$ by fitting a global GMM based on the whole corpus, and the remaining parameters by solving the following maximum a posteriori (MAP) loss,

$$\max_\Theta \left[ \ln p(z|\Theta) + \ln p(\Theta) \right].$$

The MAP estimates can be obtained via an EM algorithm: in the E-step, we compute

$$Pr(k|z_i) = \frac{w_k^I N(z_i; \mu_k^I, \Sigma_k)}{\sum_{j=1}^K w_j^I N(z_i; \mu_j^I, \Sigma_j)}, \quad (3.2)$$

$$n_k = \sum_{i=1}^N Pr(k|z_i), \quad (3.3)$$

$$E_k(z) = \frac{1}{n_k} \sum_{i=1}^N Pr(k|z_i) z_i \quad (3.4)$$

$$E_k(z^2) = \frac{1}{n_k} \sum_{i=1}^N Pr(k|z_i) z_i^2 \quad (3.5)$$

and in the M-step, we update

$$\hat{w}_k^I = \gamma_k n_k / N + (1 - \gamma_k) w_k, \quad (3.6)$$

$$\hat{\mu}_k^I = \alpha_k E_k(z) + (1 - \alpha_k) \mu_k, \quad (3.7)$$

$$\hat{\Sigma}_k^I = \alpha_k^v E_k(z^2) + (1 - \alpha_k^v)(\Sigma_k + \mu_k^2) - \mu_k^2, \quad (3.8)$$
where
\[ \alpha_k = \frac{n_k}{n_k + r}, \quad \alpha^v_k = \frac{n_k}{n_k + r^v}, \quad \gamma_k = \frac{N}{N + T}. \]

If a Gaussian component has a high probabilistic count, \( n_k \), then \( \alpha_k \) and \( \alpha^v_k \) approach 1 and the adapted parameters emphasize the new sufficient statistics \( m_k \); otherwise, the adapted parameters are determined by the global model \( \mu_k \). The tuning parameters \( r, r^v \) and \( T \) can also affect the MAP adaptation. In general, the larger \( r, r^v \) and \( T \), the larger the influence of the prior distribution on the adaptation. For example, when \( r \) goes to infinity, the MAP adaptation for \( \mu_k \) is fixed at the prior mean, similar for \( T \) and \( r^v \). In practice we adjust \( r, r^v \) and \( T \) empirically, based on the total number of feature vectors for each image.

### 3.2.1 Histogram as a special case of GMMs

It is easy to see that the histogram representation is a special case of GMMs, with only the weights \( w^I_k \) being adapted: If we set the hyper-parameters \( T = 0 \) and \( r = \infty \), from equations (3.6, 3.7), we have all the image-specific GMMs sharing the same mean vectors and covariance matrices, and therefore the only information captured by GMMs is the weight \( w^I_k \) which is proportional to the histogram counts.

Here we want to highlight three aspects in which the GMM-based approach extends histograms. First, histograms use the Euclidean distance as the clustering metric in constructing bins, while GMMs use the Mahahamalobis distance that takes into account the heterogeneity among features. Second, histograms use a hard decision rule in distributing feature vectors into bins and the resulting data summary is sensitive to noise, while GMMs use a soft decision rule in distributing feature vectors to Gaussian components and the resulting probabilistic summary of the data is more robust. The last and the most important advantage of GMMs over histograms is the gain of information. Histograms summarize the appearance information of an image (i.e., a bag of feature vectors) by the counts in each histogram bin, which correspond to the weights of Gaussian components in the adapted mixture model. In addition to weights, GMMs summarize each image by the
adapted mean vectors and covariance matrices, which provide richer information in constructing the super-vector and in calculating similarities between images.

### 3.3 Similarity of Distributions

After estimating the image-specific GMMs, we can compare two images by computing the similarity of their corresponding distributions.

Suppose we have two images \( x_a \) and \( x_b \), with the feature vectors \( Z_a \) and \( Z_b \) respectively. Then, from the GMM adaptation process in (3.2-3.7), we can obtain two adapted GMMs for them, denoted as \( g_a \) and \( g_b \). Consequently, each image is represented by a specific GMM distribution model, and a natural similarity measure between them is the Kullback-Leibler divergence. Here we consider two cases: in the first case, we assume the major information of image-specific GMMs is contained in the mean, so that the comparison is just based on the mean vectors; in the second case, we think both mean and variance play important roles in the image-specific GMMs, so that we jointly consider both mean and covariance vectors.

#### 3.3.1 GMM mean supervector

The formulation of the Kullback-Leibler divergence is shown as follows:

\[
D(g_a \| g_b) = \int g_a(z) \log \left( \frac{g_a(z)}{g_b(z)} \right) dz.
\]  

The Kullback-Leibler divergence itself does not satisfy the conditions for a kernel function, but there exists an upper bound from the log-sum inequality,

\[
D(g_a \| g_b) \leq \sum_{k=1}^{K} w_k D(\mathcal{N}(z; \mu_k^a, \Sigma_k) \| \mathcal{N}(z; \mu_k^b, \Sigma_k)),
\]  

where \( \mu_k^a \) denotes the adapted mean of the \( k \)th component from image \( x_a \), and likewise for \( \mu_k^b \).

Based on the assumption that the covariance matrices are unchanged during the MAP adaptation
process, the right side of the above inequality is equal to

\[ d(x_a, x_b) = \frac{1}{2} \sum_{k=1}^{K} w_k (\mu_a^k - \mu_b^k)^T \Sigma_k^{-1} (\mu_a^k - \mu_b^k). \] (3.11)

It is easy to prove that \( d(x_a, x_b) \) is a metric function, and therefore we can define the following kernel function:

\[ k(x_a, x_b) = e^{-d(x_a, x_b)/\delta_1^2}, \] (3.12)

where \( \delta_1 \) is a constant for controlling the final similarity. \( k(x_a, x_b) \) can be considered as a conventional Gaussian kernel defined on the so-called supervector,

\[ m(x_a) = [\sqrt{\frac{w_1}{2}} \Sigma_1^{-\frac{1}{2}} \mu_1^a; \cdots; \sqrt{\frac{w_K}{2}} \Sigma_K^{-\frac{1}{2}} \mu_K^a]. \] (3.13)

### 3.4 GMM Mean-Covariance Supervector

Recent studies have shown that covariance information contains rich information and demonstrated its effectiveness for image detection [Porikli 2006, Seo 2009] and language recognition [Campbell 2008]. Thus, instead of adopting mean-based image representation in HG, we propose a novel vectorized representation by jointly considering the information of mean and covariance of the GMM.

Here we adopt the symmetrized Kullback-Leibler divergence (KLD)

\[ D_s(g_a||g_b) = \int_{z \in \mathbb{R}^d} \left( g_a(z) \log \left( \frac{g_a(z)}{g_b(z)} \right) + g_b(z) \log \left( \frac{g_b(z)}{g_a(z)} \right) \right) dz. \] (3.14)
By using log-sum inequality, we can easily see that $D_s(g_a || g_b)$ is upper bounded by $U_s(g_a, g_b)$

$$D_s(g_a || g_b) \leq \sum_{i=1}^{M} \omega_{0,i} (D(N_{a,i} || N_{b,i}) + D(N_{b,i} || N_{a,i}))$$

$$= \sum_{i=1}^{M} \omega_{0,i} D_s(N_{a,i} || N_{b,i})$$

$$= U_s(g_a, g_b). \quad (3.15)$$

And a closed form formula for the symmetrized divergence between two Gaussian distributions can be given by

$$D_s(N_{a,i} || N_{b,i}) = \frac{1}{2} \text{tr}(\Sigma_{a,i}^{-1} \Sigma_{b,i}) + \frac{1}{2} \text{tr}(\Sigma_{b,i}^{-1} \Sigma_{a,i}) - d$$

$$+ \frac{1}{2} (m_{a,i} - m_{b,i})^T (\Sigma_{a,i}^{-1} + \Sigma_{b,i}^{-1}) (m_{a,i} - m_{b,i}). \quad (3.16)$$

In fact, $\Sigma_{a,i}$ and $\Sigma_{b,i}$ are diagonal in our system. Let $\sigma_{a,i,j}^2$, $\sigma_{b,i,j}^2$ and $\sigma_{0,i,j}^2$ ($j = 1, \ldots, d$) be the corresponding diagonal terms in the matrices $\Sigma_{a,i}$, $\Sigma_{b,i}$ and $\Sigma_{0,i}$, respectively. If we assume that $\sigma_{a,i,j}^2$ and $\sigma_{b,i,j}^2$ are around $\sigma_{0,i,j}^2$, by using the approximation $\frac{1}{1+z} \approx 1 - z + z^2$ while $z \to 0$,

$$\frac{1}{2} \left( \frac{\sigma_{a,i,j}^2}{\sigma_{b,i,j}^2} + \frac{\sigma_{b,i,j}^2}{\sigma_{a,i,j}^2} \right) = \frac{1}{2} \left( \frac{\sigma_{a,i,j}^2 / \sigma_{0,i,j}^2}{1 + \frac{\sigma_{b,i,j}^2 / \sigma_{0,i,j}^2}{\sigma_{a,i,j}^2 / \sigma_{0,i,j}^2}} + \frac{\sigma_{b,i,j}^2 / \sigma_{0,i,j}^2}{1 + \frac{\sigma_{a,i,j}^2 / \sigma_{0,i,j}^2}{\sigma_{b,i,j}^2 / \sigma_{0,i,j}^2}} \right)$$

$$\approx 1 + \frac{1}{2} \left( \frac{\sigma_{a,i,j}^2 / \sigma_{0,i,j}^2 - \sigma_{b,i,j}^2 / \sigma_{0,i,j}^2}{\sigma_{0,i,j}^2} \right)^2. \quad (3.17)$$

So the symmetrized KLD between two Gaussian is

$$D_s(N_{a,i} || N_{b,i}) = \frac{1}{2} \text{tr}[(\Sigma_{a,i} - \Sigma_{b,i}) \Sigma_{0,i}^{-2}(\Sigma_{a,i} - \Sigma_{b,i})]$$

$$+ (m_{a,i} - m_{b,i})^T (\frac{1}{2} \Sigma_{a,i}^{-1} + \frac{1}{2} \Sigma_{b,i}^{-1}) (m_{a,i} - m_{b,i}). \quad (3.18)$$
So the upper bound is

\[
U_s(g_a, g_b) = \sum_{i=1}^{M} \omega_{0,i} \frac{1}{2} tr\left[ (\Sigma_{a,i} - \Sigma_{b,i}) \Sigma_{0,i}^{-2} (\Sigma_{a,i} - \Sigma_{b,i}) \right] \\
+ \sum_{i=1}^{M} \omega_{0,i} (m_{a,i} - m_{b,i})^T \frac{1}{2} (\Sigma_{a,i}^{-1} + \Sigma_{b,i}^{-1}) (m_{a,i} - m_{b,i}).
\]  

(3.19)

By replacing \( \frac{1}{2} (\Sigma_{a,i}^{-1} + \Sigma_{b,i}^{-1}) \) with \( \Sigma_{0,i}^{-1} \), we can reach an approximation of the Eq.(3.19)

\[
U_s(g_a, g_b) \approx \sum_{i=1}^{M} \omega_{0,i} \frac{1}{2} tr\left[ (\Sigma_{a,i} - \Sigma_{b,i}) \Sigma_{0,i}^{-2} (\Sigma_{a,i} - \Sigma_{b,i}) \right] \\
+ \sum_{i=1}^{M} \omega_{0,i} (m_{a,i} - m_{b,i})^T \Sigma_{0,i}^{-1} (m_{a,i} - m_{b,i}).
\]  

(3.20)

From Eq.(3.20), we can see that the upper bound of the symmetrized KL divergence between two GMMs is composed of two parts. The first term on the right-hand side measures the dissimilarity between two GMMs from the view of the covariance information, while the second one is the difference of the means between two image distributions. And we form the following mean-covariance supervector to represent the image specific GMM:

\[
\phi_a(z) = [m_a(z); v_a(z)].
\]  

(3.21)

where

\[
m_a(z) = [\omega_1 \Sigma_{0,1}^{-1} m_{a,1}; \ldots; \omega_M \Sigma_{0,M}^{-1} m_{a,M}]
\]  

(3.22)

\[
v_a(z) = [\sqrt{\frac{1}{2} \omega_1 \Sigma_{0,1}^{-1}} \Sigma_{a,1}; \ldots; \sqrt{\frac{1}{2} \omega_M \Sigma_{0,M}^{-1}} \Sigma_{a,M}]
\]  

(3.23)

Here, \( m_a(z) \) is concatenated by the means of each Gaussian mixture component, while \( v_a(z) \) is concatenated by the covariances; both of them are normalized by global weights and global covariances. We name \( \phi_a(z) \) as the mean-covariance supervector (MCSup, for short), \( m_a(z) \) as the mean supervector (MSup), and \( v_a(z) \) as the covariance supervector (CSup).
If only means are considered, i.e. $\sum_{a,j} = \sum_{b,i} = \sum_{0,i}$, then only the second term of Eq.(3.20) is used, so it comes back to the first case, just considering mean vectors.

### 3.5 Gaussian Maps for Spatial Representation

According to equation (3.2), the feature vector at each patch is also modeled by a mixture of Gaussians with a mixture probability $Pr(k|z_i)$. For a fixed $k$, all such probabilities $Pr(k|z_i)$ form a map over the patch locations, which we refer to as the Gaussian map. While each Gaussian component represents some appearance structure in the feature space, the corresponding Gaussian map shows the geometric location of that structure on an image. For a GMM with $K$ components, we have $K$ Gaussian maps. We can learn the spatial information of an image by analyzing each of these Gaussian maps.

There are many possibilities to capture the locality information through those Gaussian maps. On one hand, we can summarize each Gaussian map respectively. In this way, we can obtain either global or local statistics of the particular Gaussian map, and use the combination of the statistics to represent the locality information of the corresponding appearance structure. On the other hand, we can jointly consider the Gaussian maps for locality summarization. In this way, we have the potential to capture the combination of different appearance structures, which is able to represent more complicated structures.

In general, we can use different ways to summarize the spatial information of an image as a vector. A natural way to summarize a Gaussian map is to use its mean location or normalized mean location. However, such global summary statistics do not work well for images. In Figure 3.2, we plot a subset of Gaussian maps for three images from Caltech 101 database, which is analyzed in Section 4. It is clear that local information is more important for the discriminant analysis than the global one.

Therefore we propose to hierarchically split a Gaussian map and extract summary statistics over local regions. Specifically, each of the $K$ Gaussian maps is divided into subregions based on
a sequence of increasingly coarser grids; assuming there are \( M \) subregions in total, we calculate some summary statistic \( \nu \) over each of the \( M \) regions. As a parallel form to (3.13), we define \( v(x^I) \), a vector expressing spatial information of image \( x^I \) as follows:

\[
v(x^I) = [\nu_{11}^I; \cdots; \nu_{M1}^I; \nu_{12}^I; \cdots; \nu_{M2}^I; \cdots; \nu_{MK}^I].
\] (3.24)

### 3.5.1 SPM as a special case of Gaussian maps

To avoid the loss of spatial information with histograms, Lazebnik et al. [Lazebnik 2006b] proposed the spatial pyramid matching (SPM). In SPM, images are repeatedly divided into subregions, similarity measures are repeatedly calculated for each subregions, and their weighted sum forms an overall similarity measure.

Since a histogram is a special case of GMM, SPM corresponds to a hierarchical spatial modeling over a degenerated Gaussian map where the posterior probabilities are either 0 or 1. The special similarity measure used by SPM, i.e. the histogram intersection function, corresponds to an intersection function defined over those posterior probabilities. So SPM can be viewed as a
special case of Gaussian map.

3.6 Discriminant Adaptive Projection

We concatenate the appearance vector \( m(x^I) \) and the spatial vector \( \nu(x^I) \) as a super-vector

\[
\phi(x^I) = [m(x^I); \sqrt{\eta} \nu(x^a)],
\]

where \( \eta \) is a tuning parameter to balance the information contribution from the two sources. However, directly employing such a high-dimensional vector for image classification may not lead to a good performance, because the super-vector is constructed without considering the inter-category or intra-category relationship.

To enhance the discriminating power of our representation, we propose to project \( \phi(x^I) \) to a subspace that depresses the directions with high inter-category variabilities. Let \( V \) denote the projection matrix toward the subspace with high inter-category variabilities; that is, \( (I - V)\phi(x^I) \) is the discriminant projection we are looking for. We solve \( V \) via the following objective function:

\[
V = \arg \max_{V^TV=I} \sum_{i \neq j} ||V^T \phi(x^i) - V^T \phi(x^j)||^2 W_{ij},
\]  
(3.25)

where \( W_{ij} = 1 \) when \( x^i \) and \( x^j \) belong to the same category, otherwise \( W_{ij} = 0 \). Let

\[
\Phi = [\phi(x^1), \phi(x^2), \cdots, \phi(x^N)],
\]

a matrix with \( N \) columns where \( N \) is the total number of training images. It can be shown that the optimal solution for \( V \) consists of the top eigenvectors corresponding to the largest eigenvalues of matrix \( \Phi(D - W)\Phi^T \), where \( D \) is a diagonal matrix with \( D_{ii} = \sum_{j=1}^{N} W_{ij}, \forall i \).

Suppose we use the dot product as a similarity measure between super-vectors. After applying discriminant adaptive projection (DAP), the similarity between two images, \( x^a \) and \( x^b \), is equal to

\[
D(x^a, x^b) = \phi(x^a)^T (I - VV^T) \phi(x^b).
\]  
(3.26)
That is, the projection toward $V$, which is irrelevant to the classification, is discarded in the similarity calculation.

In the DAP approach, each eigen-direction is either included or excluded for later analysis. An alternative is to adaptively shrink each direction of the subspace spanned by $V$: the one with larger eigen-values shrunk less and the one with smaller eigen-values shrunk more. Arrange all the shrinkage factors in a diagonal matrix $C$, then the similarity metric (3.26) can be reexpressed as

$$D(x^a, x^b) = \phi(x^a)^T(I - VCVT)\phi(x^b).$$

(3.27)

In our experiments, we set $C = I - \Lambda^{-1}$, where $\Lambda$ is a diagonal matrix with eigenvalues of matrix $\Phi(D - W)\Phi^T$.

### 3.7 Fast Gaussianization by Random Forest

The major computational cost of our Gaussianized image representation is to compute the likelihood of feature vectors given the global GMM. For the feature vector from each patch, we need to calculate:

$$p(z|\Theta) = \sum_{k=1}^{K} w_k N(z; \mu_k, \Sigma_k),$$

(3.28)

where $K$ denotes the total number of Gaussian components, and $(w_k, \mu_k, \Sigma_k)$ are their weights, mean and covariance matrix of the $k$th Gaussian component, respectively. In our experiments, we usually set $K = 512$ or $K = 1024$. Typically an image contains thousands of patches. Thus, likelihood computation becomes the bottleneck of our representation process.

There have been several approaches proposed to speed up the GMM computation in speech recognition community. However, it is difficult to directly use these approaches for our representation for two reasons. First, the number of the Gaussian components in a speech recognition system is usually hundreds of thousands, while we just have around a thousand Gaussian components in
our case. Thus, the Gaussian components clustering strategy for speech recognition works poorly in our case because after splitting, for each cluster, we do not have enough Gaussian components for describing the complicated distribution. Second, some fast computation approaches adopt the phoneme and context information over the Gaussian components in a speech system; however, as all the Gaussian components are obtained in an unsupervised way, there is no corresponding information in our GMM structure.

There is a well-known observation in GMM likelihood computation. Although all the Gaussian components contribute to the likelihood, for a given feature vector, only a few Gaussian components make major contributions. Usually, the top 5 components contribute more than 99% likelihood, which suggests we may reduce our computation by just computing on those top Gaussian components if we can learn in advance what is the set of the Gaussian that contributes most in a specific position in the feature space. We name the set of Gaussian as top Gaussians for a particular position. Assume the feature distribution is smooth enough, then a small region in the feature space around the specific position should share the same top Gaussians, and we can call the region a homogeneous region in the Gaussian computation.

To compute the likelihood efficiently, we should first be able to split the feature space into homogeneous regions. Also we should be able to quickly distribute a new feature vector into the corresponding region, and find the Gaussian components needed.

Thus, we propose to use random forest (RF) based approach for fast GMM computation. A RF is a classifier consisting of a collection of tree-structured classifiers [Breiman 2001]. It has been successfully used in many classification and regression tasks [Caruana 2008, Geurts 2006]. In [Caruana 2008], Caruana et al. compared the performance of RFs with neural nets, boosted trees, and support vector machines. RFs consistently achieved better performance than all the other classifiers across all dimensions. In this work, we use RF to split the feature space into small homogeneous regions; furthermore, the tree structure guarantees the speed of finding the corresponding region for a testing feature vector.
### 3.7.1 Building the random forest

The random forest is created by a set of decision trees. We first describe how to build a single decision tree and then go to a set of the trees.

The generalization error of a forest depends on the strength of the individual trees and the correlations between them. Ideally, the lowest error rate is achieved with uncorrelated and maximally strong trees. In practice, we need to find the best trade-off between strength and correlation. That means we need to achieve a large diversity across the forest of trees, while keeping the strength of each individual tree high. Although there are many ways of increasing the diversity of the forest by introducing randomness during training, we chose to randomize the question selection. In this method, the question for each node is chosen from a set of $K$ randomly generated questions. The questions are generated by choosing a feature dimension at random and asking a question about that dimension that evenly divides the data in the node. As $K$ increases, the strength of each tree increases, but so does the correlation among the trees of the forest.

**Building the decision tree**

We construct the hierarchical structure by growing a binary tree, as shown in Figure 3.3. Data travels from the top (root) node to a terminal (leaf) node along a path determined by questions asked by the nodes it traveled.

![Figure 3.3: A decision tree.](image-url)
By asking questions, the decision tree splits the feature space into small regions, each of which corresponds to a leaf node in the tree. At each node, data is partitioned into two parts such that each of them are as “pure” as possible. The objective is therefore to maximize the purity increase, or decrease of impurity,

\[ \Delta i(N) = i(N) - P_L \cdot i(N_L) - P_R \cdot i(N_R), \]  

(3.29)

where \( N_L \) and \( N_R \) are the left and right child nodes of node \( N \), and \( P_L \) and \( P_R \) are the probabilities of data reaching these two nodes from their parent respectively. Here we adopt the most widely used entropy impurity measure given by

\[ i(N) = \sum_j P(w_j|N) \log P(w_j|N), \]  

(3.30)

with \( P(w_j|N) \) being the probability of \( j^{th} \) class at node \( N \).

Various types of binary questions can be asked about feature vectors. For example, one can choose one dimension of the feature vector, and try different thresholds on it. This is equivalent to splitting the feature space by a coordinate-orthogonal plane. More generally, thresholds on arbitrary linear or non-linear function of the feature vector can be used as splitting questions. Apparently, the type of questions that can be asked defines the set of splits upon which the objective function in (3.29) is optimized.

There exists no theoretical solution for the splitting problem, though. While a large question set would generally yield a “better” split, a higher risk of over-fitting it may also suffer. Since the optimization is only done locally and greedily, the “best” split at current node would not guarantee that the successive locally optimal decisions lead to the global optimum. Moreover, by employing random forest, the pursuit of optimality of a single tree is largely alleviated. So here we simply ask questions based on a single dimension of the feature vector, and the “best” question is found by trying different thresholds on several of the dimensions.
As the purpose of the decision tree here is to split the feature space into small homogeneous regions, we ask questions according to the homogeneity for the tree splitting process. To compute the homogeneity, we need to first have the label for each training feature vector. So we calculate the posterior of feature vectors on the designed GMM in advance, then the posterior can be used as the soft labels for the feature vectors. Particularly, we can also assign the training feature to the Gaussian with the largest posterior, in that case, it becomes a hard assignment.

3.7.2 Obtaining the top Gaussians

It is straightforward to obtain the top Gaussians for a leaf node in a decision tree. As we already assigned the training feature vectors to Gaussian components, then for each leaf node, the top Gaussians correspond to the Gaussian component with maximum number of feature vectors. In the RF case, when a test feature vector comes, it will get its leaf node on each tree in the forest. Then each leaf node will contribute its own top Gaussian list and the final decision is made by voting across those lists.
Chapter 4
The Application of HG in Scene and Object Classification

In this chapter, we apply our HG representation on object recognition and scene categorization tasks. We first describe the classifiers adopted in the experiments in this chapter, and then report the performance of our approach on four diverse datasets: PASCAL VOC 2007 dataset [Everingham 2007], PASCAL VOC 2009 dataset, fifteen scene category dataset [Fei-Fei 2005] and Caltech101 database.

4.1 Classifiers

Our HG representation has already summarized the information of each image as a supervector. To perform classification tasks, we still need to pick the classifiers. In all the following experiments, we just use linear classifiers coupled with the supervectors. The reasons for using the linear classifier are: firstly, our HG representation has roughly linearized the bag-of-features, so that the linear classifiers are suitable for supervector, which has been demonstrated by the superior experiment results; secondly, the simplicity of linear classifier ensures the application of our representation in large-scale recognition tasks.

Here we introduce two linear classifiers: centroid based classifier and linear support vector machine (SVM).

4.1.1 Nearest centroid classifier

The first classifier we used is nearest centroid classifier. The image classification is directly based on the similarity between a testing image and the centroid of a category, where the centroid of a
category is denoted in the Super-Vector space, as $\bar{X}^s$, of the $s$-th category:

$$\phi(\bar{X}^s) = \frac{1}{N^s} \sum_{i \in \pi^s} \phi(x_i),$$

(4.1)

where $x_i$ is $i$-th image, $N^s$ is the number of images belonging to the $s$-th category, and $\pi^s$ denotes the index set of the images belonging to the $s$-th category. Then, the final image classification is based on normalized similarity vector as

$$C_1(X) = \left[ \frac{K(X, X^1)}{\sum_s K(X, X^s)}, \frac{K(X, X^2)}{\sum_s K(X, X^s)}, \cdots, \frac{K(X, X^S)}{\sum_s K(X, X^s)} \right],$$

where $S$ is the total number of categories, and $X$ refers to a test image.

### 4.1.2 Linear SVM

The second classifier we used is linear Support Vector Machine (SVM) [Platt 1999, Grauman 2005, Jing 2004, Lazebnik 2006b].

For a two-class problem, the decision function for a test image as $X$ has the following form:

$$g(X) = \sum_i \alpha_i y_i k(X, X_i) - b,$$

(4.2)

where $k(X, X_i)$ is the value of a kernel function for the training image $X_i$ and the test image $X$, $y_i$ is the class label of $X_i$ (+1 or −1), $\alpha_i$ is the learned weight of the training sample $X_i$ and $b$ is the constant bias. The training samples with weight $\alpha_i > 0$ are called support vectors. The support vectors and their corresponding weights are learned using the standard quadratic programming optimization process or other variations.

The multi-class SVM can also output a so-called confidence vector, denoted as

$$C_2(X) = [p_1(X), p_2(X), \cdots, p_S(X)],$$

(4.3)
where $p_s(X)$ can be approximately viewed as the probability of the image $X$ belonging to the $s$-th category. Then, the classification can be conducted based on the output values in $C_2(X)$.

### 4.1.3 Classifier fusion

Our experiments show that usually the outputs from nearest centroid and linear SVM are complementary to each other, which motivates us to fuse these two classifiers to further enhance the capability of the whole system. In this work, we use a simple criterion for the fusion of the outputs from these two classifiers. The vectors $C_1(X)$ and $C_2(X)$ both roughly measure the probabilities of a test image belonging to different categories, and hence we can simply average them for a more robust output as

$$C(X) = \frac{C_1(X) + C_2(X)}{2},$$

and then the classification can be done based on the averaged probability vector $C(X)$.

In the following sections, we investigate the effectiveness of various aspects of our representation and further compare our results with the existing works. In all experiments, we extract SIFT descriptors on the dense grid. We process in grayscale, even when color channels are available.

### 4.2 Object Classification on PASCAL VOC 2007

We first evaluate the proposed method using the widely adopted PASCAL VOC 2007 dataset [Everingham 2007], which contains objects from 20 classes. PASCAL VOC 2007 consists of 9,963 images which are divided into three subsets: training data (2501 images), validation data (2510 images), and test data (4952 images). These images range between indoor and outdoor scenes, close-ups and landscapes, and strange viewpoints. The dataset is an extremely challenging one because the appearances of objects and their poses vary significantly, with frequent occlusions. The sample images are shown in Figure 4.1.
All of the following experiment results are obtained on the testing datasets, except the comparison experiment for different Gaussian components $|C|$ (Table 4.1), which is performed on PASCAL VOC 2007 validation set.

In all of the experiments, the 128-dimensional SIFT vectors are extracted over a grid with spacing of 4 pixels on three patch scales (16x16, 25x25 and 31x31). The dimension of descriptors is reduced to 80 by applying principal component analysis (PCA). The codebooks $C$ are trained on one million randomly sampled descriptors.

The classification performance is evaluated using the Average Precision (AP) measure, which is a standard metric used by PASCAL challenge. It computes the size of the area under the Precision/Recall curve, and the higher the score, the better the performance.

Table 4.2 compares our approach with the top ranked systems in the PASCAL VOC 2007 evaluation. Our approach outperforms the best system in most of the categories (18/20). Our approach achieves 64.0% mean AP over 20 categories, which achieves a 4.6% absolute improvement against the best system in the evaluation. Furthermore, the best system in the evaluation, “INRIA(GA)”, has already used multiple detectors and multiple descriptors and further combined all the subspace by genetic algorithm, whereas our approach just adopts single detector and descriptor, and just
uses equal weights for the components of the spatial pyramid, which shows the potential of our new representation.

Table 4.1: The influence of codebook sizes $|C|$, on PASCAL VOC 2007 validation set.

| AP (%) | $|C| = 256$ | $|C| = 512$ | $|C| = 1024$ | $|C| = 2048$ |
|--------|-------------|-------------|-------------|-------------|
| aeroplane | 77.7 | 77.9 | 77.9 | 78.7 |
| bicycle | 55.6 | 57.2 | 58.2 | 58.7 |
| bird | 51.0 | 53.5 | 54.4 | 54.0 |
| boat | 66.3 | 66.9 | 67.1 | 68.9 |
| bottle | 25.5 | 29.8 | 31.5 | 31.9 |
| bus | 56.2 | 59.7 | 60.9 | 60.0 |
| car | 78.8 | 79.6 | 79.8 | 80.5 |
| cat | 59.5 | 61.4 | 62.3 | 62.4 |
| chair | 56.4 | 56.6 | 56.8 | 58.0 |
| cow | 40.0 | 43.6 | 45.6 | 44.3 |
| dining_table | 52.7 | 58.8 | 61.1 | 60.7 |
| dog | 42.3 | 46.5 | 48.7 | 47.1 |
| horse | 72.5 | 72.1 | 72.2 | 74.4 |
| motorbike | 65.7 | 68.7 | 70.1 | 70.5 |
| person | 79.8 | 81.0 | 81.6 | 81.7 |
| potted_plant | 23.3 | 22.9 | 22.5 | 23.2 |
| sheep | 30.2 | 33.9 | 35.5 | 32.0 |
| sofa | 52.2 | 54.7 | 55.9 | 57.3 |
| train | 80.2 | 81.2 | 81.4 | 82.5 |
| tv/monitor | 55.0 | 56.4 | 57.2 | 57.9 |
| average | 56.0 | 58.1 | 59.0 | 59.2 |

4.2.1 The effect of codebook size

In this section we report experimental results on PASCAL VOC 2007 validation set to show the effect of codebook size $|C|$ on classification performance. As shown in Table 4.1, as we increase $|C|$ from 256, to 512, 1024, and 2048, the classification accuracy increases as well. But the improvement speed drops after $|C|$ goes over 1024.

4.3 Object Classification on PASCAL VOC 2009

We also evaluate our method using the widely adopted PASCAL VOC 2009 dataset. PASCAL VOC 2009 consists of 14,743 images of 20 object categories aforementioned and correspondingly
Table 4.2: Comparison of our method with top performers in PASCAL VOC 2007.

<table>
<thead>
<tr>
<th></th>
<th>QMUL</th>
<th>TKK</th>
<th>XRCE</th>
<th>INRIA(flat)</th>
<th>INRIA(GA)</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>aeroplane</td>
<td>71.6</td>
<td>71.4</td>
<td>72.3</td>
<td>74.8</td>
<td>77.5</td>
<td><strong>79.4</strong></td>
</tr>
<tr>
<td>bicycle</td>
<td>55.0</td>
<td>51.7</td>
<td>57.5</td>
<td>62.5</td>
<td>63.6</td>
<td><strong>72.5</strong></td>
</tr>
<tr>
<td>bird</td>
<td>41.1</td>
<td>48.5</td>
<td>53.2</td>
<td>51.2</td>
<td><strong>56.1</strong></td>
<td>55.6</td>
</tr>
<tr>
<td>boat</td>
<td>65.5</td>
<td>63.4</td>
<td>68.9</td>
<td>69.4</td>
<td>71.9</td>
<td><strong>73.8</strong></td>
</tr>
<tr>
<td>bottle</td>
<td>27.2</td>
<td>27.3</td>
<td>28.5</td>
<td>29.2</td>
<td>33.1</td>
<td><strong>34.0</strong></td>
</tr>
<tr>
<td>bus</td>
<td>51.1</td>
<td>49.9</td>
<td>57.5</td>
<td>60.4</td>
<td>60.6</td>
<td><strong>72.4</strong></td>
</tr>
<tr>
<td>car</td>
<td>72.2</td>
<td>70.1</td>
<td>75.4</td>
<td>76.3</td>
<td>78.0</td>
<td><strong>83.4</strong></td>
</tr>
<tr>
<td>cat</td>
<td>55.1</td>
<td>51.2</td>
<td>50.3</td>
<td>57.6</td>
<td>58.8</td>
<td><strong>63.6</strong></td>
</tr>
<tr>
<td>chair</td>
<td>47.4</td>
<td>51.7</td>
<td>52.2</td>
<td>53.1</td>
<td>53.5</td>
<td><strong>56.6</strong></td>
</tr>
<tr>
<td>cow</td>
<td>35.9</td>
<td>32.3</td>
<td>39.0</td>
<td>41.1</td>
<td>42.6</td>
<td><strong>52.8</strong></td>
</tr>
<tr>
<td>dining_table</td>
<td>37.4</td>
<td>46.3</td>
<td>46.8</td>
<td>54.9</td>
<td>54.9</td>
<td><strong>63.2</strong></td>
</tr>
<tr>
<td>dog</td>
<td>41.5</td>
<td>41.5</td>
<td>45.3</td>
<td>42.8</td>
<td>45.8</td>
<td><strong>49.5</strong></td>
</tr>
<tr>
<td>horse</td>
<td>71.5</td>
<td>72.6</td>
<td>75.7</td>
<td>76.5</td>
<td>77.5</td>
<td><strong>80.9</strong></td>
</tr>
<tr>
<td>motorbike</td>
<td>57.9</td>
<td>60.2</td>
<td>58.5</td>
<td>62.3</td>
<td>64.0</td>
<td><strong>71.9</strong></td>
</tr>
<tr>
<td>person</td>
<td>80.8</td>
<td>82.2</td>
<td>84.0</td>
<td>84.5</td>
<td><strong>85.9</strong></td>
<td>85.1</td>
</tr>
<tr>
<td>potted_plant</td>
<td>15.6</td>
<td>31.7</td>
<td>32.6</td>
<td>36.3</td>
<td>36.3</td>
<td><strong>36.4</strong></td>
</tr>
<tr>
<td>sheep</td>
<td>33.3</td>
<td>30.1</td>
<td>39.7</td>
<td>41.3</td>
<td>44.7</td>
<td><strong>46.5</strong></td>
</tr>
<tr>
<td>sofa</td>
<td>41.9</td>
<td>39.2</td>
<td>50.9</td>
<td>50.1</td>
<td>50.6</td>
<td><strong>59.8</strong></td>
</tr>
<tr>
<td>train</td>
<td>76.5</td>
<td>71.1</td>
<td>75.1</td>
<td>77.6</td>
<td>79.2</td>
<td><strong>83.3</strong></td>
</tr>
<tr>
<td>tv/monitor</td>
<td>45.9</td>
<td>41.0</td>
<td>49.5</td>
<td>49.3</td>
<td>53.2</td>
<td><strong>58.9</strong></td>
</tr>
<tr>
<td>average</td>
<td>51.2</td>
<td>51.7</td>
<td>55.6</td>
<td>57.5</td>
<td>59.4</td>
<td><strong>64.0</strong></td>
</tr>
</tbody>
</table>

are divided into three subsets: training data (3473 images), validation data (3581 images), and testing data (7689 images).

We use the same setting as in used in PASCAL VOC 2007. From Table 4.3, we can see that our approach significantly outperforms the best system in the 2009 evaluation.

4.3.1 Visualization of the learned patch-level function

Because the supervector is a linear combination of feature vectors and linear classifiers are used in the next step, the final classification score of an image can be viewed as a linear combination of contributions from all the feature vectors of the images. Therefore, we can visualize that different patches have different contributions to a particular category.

In Figure 4.2, we show the response map (with kernel smoothing) on a set of random images from the PASCAL VOC 2009 test set. In most of the cases, the results are quite meaningful – the target objects are mostly covered by high-valued responses of $g(x)$. This observation suggests a
Table 4.3: Comparison of our method with top performers in PASCAL VOC 2009.

<table>
<thead>
<tr>
<th></th>
<th>AP (%)</th>
<th>LEOBEN</th>
<th>LIP6</th>
<th>LEAR</th>
<th>FIRSTNIKON</th>
<th>CVC</th>
<th>UVASURREY</th>
<th>OURS</th>
</tr>
</thead>
<tbody>
<tr>
<td>aeroplane</td>
<td>79.5</td>
<td>80.9</td>
<td>79.5</td>
<td>83.3</td>
<td>86.3</td>
<td>84.7</td>
<td></td>
<td>87.1</td>
</tr>
<tr>
<td>bicycle</td>
<td>52.1</td>
<td>52.3</td>
<td>55.5</td>
<td>59.3</td>
<td>60.7</td>
<td>63.9</td>
<td></td>
<td>67.4</td>
</tr>
<tr>
<td>bird</td>
<td>57.2</td>
<td>53.8</td>
<td>54.5</td>
<td>62.7</td>
<td>66.4</td>
<td>66.1</td>
<td>65.8</td>
<td></td>
</tr>
<tr>
<td>boat</td>
<td>59.9</td>
<td>60.8</td>
<td>63.9</td>
<td>65.3</td>
<td>65.3</td>
<td>67.3</td>
<td></td>
<td>72.3</td>
</tr>
<tr>
<td>bottle</td>
<td>29.3</td>
<td>29.1</td>
<td>43.7</td>
<td>30.2</td>
<td>41.0</td>
<td>37.9</td>
<td>40.9</td>
<td></td>
</tr>
<tr>
<td>bus</td>
<td>63.5</td>
<td>66.2</td>
<td>70.3</td>
<td>71.6</td>
<td>71.7</td>
<td>74.1</td>
<td></td>
<td>78.3</td>
</tr>
<tr>
<td>car</td>
<td>55.1</td>
<td>53.4</td>
<td>66.4</td>
<td>58.2</td>
<td>64.7</td>
<td>63.2</td>
<td></td>
<td>69.7</td>
</tr>
<tr>
<td>cat</td>
<td>53.9</td>
<td>55.9</td>
<td>56.5</td>
<td>62.2</td>
<td>63.9</td>
<td>64.0</td>
<td></td>
<td>69.7</td>
</tr>
<tr>
<td>chair</td>
<td>51.1</td>
<td>50.7</td>
<td>54.4</td>
<td>54.3</td>
<td>55.5</td>
<td>57.1</td>
<td></td>
<td>58.5</td>
</tr>
<tr>
<td>cow</td>
<td>31.3</td>
<td>33.8</td>
<td>38.8</td>
<td>40.7</td>
<td>40.1</td>
<td>46.2</td>
<td></td>
<td>50.1</td>
</tr>
<tr>
<td>dining table</td>
<td>42.9</td>
<td>43.9</td>
<td>44.1</td>
<td>49.2</td>
<td>51.3</td>
<td>54.7</td>
<td></td>
<td>55.1</td>
</tr>
<tr>
<td>dog</td>
<td>44.1</td>
<td>44.6</td>
<td>46.2</td>
<td>50.0</td>
<td>45.9</td>
<td>53.5</td>
<td></td>
<td>56.3</td>
</tr>
<tr>
<td>horse</td>
<td>54.8</td>
<td>59.4</td>
<td>58.5</td>
<td>66.6</td>
<td>65.2</td>
<td>68.1</td>
<td></td>
<td>71.8</td>
</tr>
<tr>
<td>motorbike</td>
<td>58.4</td>
<td>58</td>
<td>64.2</td>
<td>62.9</td>
<td>68.9</td>
<td>70.6</td>
<td></td>
<td>70.8</td>
</tr>
<tr>
<td>person</td>
<td>81.1</td>
<td>80.0</td>
<td>82.2</td>
<td>83.3</td>
<td>85.0</td>
<td>85.2</td>
<td></td>
<td>84.1</td>
</tr>
<tr>
<td>potted plant</td>
<td>30.0</td>
<td>25.3</td>
<td>39.1</td>
<td>34.2</td>
<td>40.8</td>
<td>38.5</td>
<td></td>
<td>31.4</td>
</tr>
<tr>
<td>sheep</td>
<td>40.2</td>
<td>41.9</td>
<td>41.3</td>
<td>48.2</td>
<td>49</td>
<td>47.2</td>
<td></td>
<td>51.5</td>
</tr>
<tr>
<td>sofa</td>
<td>44.2</td>
<td>42.5</td>
<td>39.8</td>
<td>46.1</td>
<td>49.1</td>
<td>49.3</td>
<td></td>
<td>55.1</td>
</tr>
<tr>
<td>train</td>
<td>74.9</td>
<td>78.4</td>
<td>73.6</td>
<td>83.4</td>
<td>81.8</td>
<td>83.2</td>
<td></td>
<td>84.7</td>
</tr>
<tr>
<td>tv/monitor</td>
<td>58.2</td>
<td>60.1</td>
<td>66.2</td>
<td>65.5</td>
<td>68.6</td>
<td>68.1</td>
<td></td>
<td>65.2</td>
</tr>
<tr>
<td>average</td>
<td>53.1</td>
<td>53.6</td>
<td>56.9</td>
<td>58.9</td>
<td>61.1</td>
<td>62.1</td>
<td></td>
<td>64.3</td>
</tr>
</tbody>
</table>

potential to extend the current framework toward joint classification and detection.

4.4 Object Recognition on Caltech 101

Our third set of experiments are conducted on the Caltech101 database. This database consists of 101 object classes with high intra-class appearance and shape variability. The number of images in each class varies from 31 to 800, and most images are of medium resolution (about 300 × 300 pixels). This database is one of the most diverse and thoroughly studied databases for object recognition, and significant progress has been made on it for state-of-the-art algorithms. There exist several drawbacks for this database, though. For example, most objects are located at the center of an image with clean background. And many classes are devoid of pose and scale variability. Moreover, the presence of rotation artifacts tends to make some classes (e.g. minaret) much easier to identify.

In this experiment, the representation step is the same as in scene recognition: first extract SIFT
Figure 4.2: Visualization of the learned patch-level function on image examples from PASCAL-09.
Table 4.4: Performance comparison on Caltech101 (single descriptor).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>15 Train</th>
<th>30 Train</th>
</tr>
</thead>
<tbody>
<tr>
<td>HS+LS [Zhang 2008]</td>
<td>–</td>
<td>53.9</td>
</tr>
<tr>
<td>SPM [Lazebnik 2006b]</td>
<td>56.4</td>
<td>64.6</td>
</tr>
<tr>
<td>SVM-KNN [Zhang 2006]</td>
<td>59.1</td>
<td>66.2</td>
</tr>
<tr>
<td>GBDist-SVM [Varma 2007]</td>
<td>59.3</td>
<td>–</td>
</tr>
<tr>
<td>GBDist-NN [Varma 2007]</td>
<td>45.2</td>
<td>–</td>
</tr>
<tr>
<td>Griffin SPM [Griffin 2007]</td>
<td>59.0</td>
<td>67.6</td>
</tr>
<tr>
<td>LearnDist [Frome 2007]</td>
<td>63.2</td>
<td>–</td>
</tr>
<tr>
<td>ML+CORR [Jain 2008]</td>
<td>61.0</td>
<td>69.6</td>
</tr>
<tr>
<td>NBNN [Boiman 2008]</td>
<td>65.0</td>
<td>70.4</td>
</tr>
<tr>
<td>HG</td>
<td><strong>65.5</strong></td>
<td><strong>73.1</strong></td>
</tr>
</tbody>
</table>

descriptor within a $20 \times 20$ sliding window, and then learn a 512-mixture GMM and Gaussian map for each image. For experiment setup, we follow the standard procedure, namely we randomly select 15 and 30 training images per class and 50 for testing. The recognition rate is then computed as the average of per-class accuracies. Similar to the previous experiments, the entire procedure is repeated ten times, and the average performance and its standard deviation are reported.

Table 4.4 shows a performance comparison of HG representation with several recently reported methods, all based on a single descriptor. For the two different training/testing settings, HG representation achieves the best result, i.e., 65.5% for 15 training images and 73.1% for 30 training images. It is worth mentioning that most of previous methods used computing-extensive classifiers, such as support vector machine (SVM) and nearest neighbor (NN), or a hybrid of them [Lazebnik 2006b, Varma 2007, Griffin 2007, Zhang 2006]. Especially for NBNN [Boiman 2008], although it achieved a comparable performance at 15 training samples, it involves finding the nearest patch among all patches in each class, which is extremely time-consuming at the testing phase. While it is true that the computational burden of NBNN can be somehow alleviated by approximated-k-nearest-neighbor algorithm, there is still a big issue when the number of labeled samples increases, where this is often the case in many real-world applications. In our framework, however, the classification step becomes trivial after the representation is obtained. We only need
to compute the distance from each class centroid (image-to-class distance), of which the cost is constant for a given number of classes.

![Example images from the scene category database.](image)

**Figure 4.3:** Example images from the scene category database.

### 4.5 Scene Category Recognition

The scene database is composed of fifteen scene categories, thirteen provided by Fei-Fei *et al.* in [Fei-Fei 2005] and the other two collected by Lazebnik *et al.* in [Lazebnik 2006b]. Each scene category contains 200 to 400 images. The average size of the images is around $300 \times 250$ pixels. This database is one of the most comprehensive scene category databases used in the literature. Example images of different scene categories of this database are illustrated in Figure 4.3.
Here, the experiment settings are set the same as the ones used in [Lazebnik 2006b] and [Fei-Fei 2005] to guarantee the fairness of performance comparison. Specifically, all experiments are repeated ten times with 100 randomly selected images per class for training and the rest for testing. We first extract 128 dimensional SIFT [Lowe 2004] vectors from $16 \times 16$ and $24 \times 24$ patches with overlapping windows shifted by 4 pixels. The dimension of SIFT descriptor is reduced to 64 by Principal Component Analysis (PCA). The GMM contains 512 Gaussian components, while the histogram contains 512 bins.

Table 4.5: Performance comparison on scene category database.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Histogram [Fei-Fei 2005]</td>
<td>65.2</td>
</tr>
<tr>
<td>SPM [Lazebnik 2006b]</td>
<td>81.4</td>
</tr>
<tr>
<td><strong>HG</strong></td>
<td><strong>88.4</strong></td>
</tr>
</tbody>
</table>

Table 4.5 compares our approach with several existing systems on the scene classification task. The result in [Fei-Fei 2005] is 65.2%, which is based on histogram representation without any spatial information. In [Lazebnik 2006b], Lazebnik et al. introduced spatial pyramid matching (SPM) to incorporate the spatial information with histogram representation and reported an accuracy of 81.4% using SVM with nonlinear histogram intersection kernel. In the experiment, by a simple nearest centroid (NC) classifier, HG representation achieves a superior performance of 88.4% in accuracy. The results are consistent with our analysis in the previous sections: HG is more general than both histogram and SPM.

Table 4.6: The classification results on scene category database.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Histogram</td>
<td>42.8</td>
</tr>
<tr>
<td>GMM</td>
<td>76.8</td>
</tr>
<tr>
<td>GMM+GMap</td>
<td>81.8</td>
</tr>
<tr>
<td>GMM+DAP</td>
<td>83.5</td>
</tr>
<tr>
<td>HG with MSup</td>
<td>86.6</td>
</tr>
<tr>
<td>HG with MCSup</td>
<td>88.4</td>
</tr>
</tbody>
</table>

Table 4.6 gives an in-depth analysis of the effectiveness of each aspect of our representation.
Figure 4.4: Confusion matrix on scene category database for the HG representation. The average classification accuracy is 88.4%. The entry in the $i^{th}$ row and $j^{th}$ column is the percentage of images from class $i$ that were misidentified as class $j$. For better viewing, please see the pdf file.

Here all the results are obtained by nearest centroid (NC) classifier. The table demonstrates the performance when adding the components of our representation one by one. It is evident that the three components - GMM for appearance representation, Gaussian maps (GM) for spatial layout encoding and DAP for discriminant dimension reduction - jointly improve the recognition accuracy. Also, combining the covariance information with mean information when forming the supervector (MCSup) achieves better performance than just consider the mean information in supervector (MSup). Note that [Lazebnik 2006b] reported an accuracy of 74.2% based on histogram representation, which is higher than 42.8% here. This is because [Lazebnik 2006b] employed a nonlinear histogram intersection kernel for SVM. This indicated that the performance of histogram representation is sensitive to choice of kernel metrics, and relies heavily on the classifier.

Figure 4.4 shows the confusion matrix of the fifteen scene categories for the HG representation. And the highest recognition rate is 100% for “CALsuburb” while the lowest accuracy is 77% for “livingroom”.

Chapter 5

The Application of HG in Facial Biometrics

5.1 Implementation Detail

We apply the HG representation in facial biometrics in this chapter. For all of the experiments in this chapter, we use coordinate patch as the input features. Particularly, some applications, such as the age estimation and the pose estimation, are regression tasks rather than classification tasks. Therefore, we should modify our HG representation to fit the regression tasks. We first introduce two major modifications, (1) Synchronizing Inter-Modality Similarity, which is the modified DAP in regression case; and (2) kernel regression, which is used as a back-end process for regression tasks.

Following the discussion on modifications of HG, we systematically evaluate the effectiveness of our new HG representation, and compare HG with the state-of-the-art algorithms for human face recognition, human gender recognition, human age estimation, and head pose estimation. The human face recognition experiments are conducted on the CMU PIE database. The human gender recognition is conducted on the YGA database. The human age estimation experiments are conducted on the YGA database and the FG-NET aging database. The head pose estimation experiments are conducted on the CHIL data used for CLEAR07 evaluation.

5.1.1 The synchronization of inter-modality similarities

In this subsection, we present a weak learning process for enhancing the kernel discriminating power for regression case. We take age and pose estimation as the applications as discussed below.
More specifically, we want the distance kernel computed using a pair of images with similar ages or poses to have a small value, while the distance kernel computed using images with greatly different ages or poses should have a large value. In this way the similarities measured in the feature modality and label modality are synchronized; hence we call this process Inter-Modality Similarity Synchronization. A natural way to achieve this goal is to remove any directions in which supervectors corresponding to similar labels (similar ages or similar poses) spread out over a wide range of values (high-variability). These directions are assumed in this work to be characterized by a subspace spanned by the projection matrix $V$. In order to identify $V$, we first define the label-similarity matrix $W$ as

$$W_{ij} = e^{-||l_i - l_j||^2/\delta_2^2},$$  \hspace{1cm} (5.1)$$

which measures the label similarity between image $x_i$ and image $x_j$, using hyper-parameter $\delta_2$ to control the scale over which label similarities are distinguished.

The goal of inter-modality similarity synchronization is to identify the subspace, $V$, that has maximum inter-image distance (maximum $\|V^T \phi(x_i) - V^T \phi(x_j)\|^2$) for image pair with high label similarity (large $W_{ij}$). Expressing this goal in the form of an optimality criterion, we find that

$$V = \arg \max_{V^TV=I} \sum_{i \neq j} ||V^T \phi(x_i) - V^T \phi(x_j)||^2 W_{ij}. \hspace{1cm} (5.2)$$

Denote $\hat{X} = [\phi(x_1), \phi(x_2), \cdots, \phi(x_N)]$, then the optimal $V$ consists of the eigenvectors corresponding to the top few largest eigenvalues of the matrix $\hat{X}(D - W)\hat{X}^T$, where $D$ is a diagonal matrix with $D_{ii} = \sum_{j=1}^N W_{ij}$, $\forall i$.

$V$ identifies the major directions in which feature similarity and label similarity are out of synchronization (high label similarity corresponds to low feature similarity, and vice versa). In order to achieve inter-modality similarity synchronization, we must discard the components $V^\phi(x_i)$ prior to computing the similarity between any two images. It is possible to define a similarity-
synchronized distance metric, \( d(x_a, x_b) \), as

\[
d(x_a, x_b) = \frac{1}{2} \sum_{k=1}^{K} w_k (\mu^a_k - \mu^b_k)^T \Sigma_k^{-\frac{1}{2}} (I - V V^T) \Sigma_k^{-\frac{1}{2}} (\mu^a_k - \mu^b_k),
\]

by noticing the equality \((I - V V^T)(I - V V^T) = (I - V V^T)\).

Note that the kernel is applicable not only to image pairs. If any object can be characterized by a feature set \( Z \), then we can adapt the global GMM to a new one by the process shown in (3.2-3.7); thus we can compute the kernel similarity between an image and an image set.

5.2 Kernel Regression

Kernel regression [Takeda 2006] is a non-parametric technique in statistics to estimate the conditional expectation of a random variable. In this work, we use it for age estimation and pose estimation. We generalize this model and set the expected values of the reference points as model parameters to be determined. In kernel regression, a set of reference points is required for learning the model. We evenly divide the label field into multiple subsets, and then for each subset with training images denoted as \( X_m \), we can compute the similarity, denoted as \( k(x, X_m) \), between an image \( x \) and the image set \( X_m \). Then the kernel regression model is expressed as

\[
F(x) = \frac{\sum_{m=1}^{M} \beta_m k(x, X_m)}{\sum_{m=1}^{M} k(x, X_m)}, \tag{5.3}
\]

where \( M \) is the number of reference subsets, and the parameters \( \beta_m \) can be easily derived by using the Least Squares Error method based on the training images. For a new image, its age or pose label can be directly computed from (5.3).

Discussion: There exist many popular algorithms for regression, e.g., linear regression and neural networks [Rumelhart 2002]. In this work, we choose kernel regression because the patch-kernel itself provides reasonably good similarity measurement, and then the term \( k(x, X_m) \) will have a large value if \( x \) is within the age or pose range in which \( X_m \) lies. This property coin-
cides with the philosophy of kernel regression. The result of learning is a set of kernel regression coefficients that approximately equal the label means of the reference image sets $X_m$.

We use RPK in experiments to represent our algorithms when adopting kernel regression.

### 5.3 Face Recognition

We investigate the performance of our representation on face recognition on the CMU PIE database and we Labeled Face in the Wild (LFW) database.

#### 5.3.1 The experiments on CMU PIE database

The CMU PIE database contains 41,368 face images from 68 individuals. For each individual, face images of varying pose, illumination, and expression are captured by 13 synchronized cameras under 21 flashes. Sample images from this database are shown in Figure 5.1. We choose the five near frontal poses (C05, C07, C09, C27, C29) and use all such images under different illuminations, lighting and expressions, which leaves us 170 near frontal face images for each individual. A random subset with $l = \{10, 20, 30\}$ images per individual is taken to form the training set, while the rest of the database are used as the testing set. For each given $l$, we average the results over 10 random splits. All the configuration is the same as in [Cai 2007].

![Figure 5.1: Example images from the CMU PIE database. For each subject, there are 170 near frontal face images under varying pose, illumination, and expression.](image)

In the experiments, original images are manually aligned (two eyes are aligned at the same position), cropped, and then re-sized to $32 \times 32$ pixels, with 256 gray levels per pixel. The patch
size is set to $6 \times 6$ pixels, and the patches are densely sampled pixel by pixel. As suggested in [Lucey 2004], we extract feature of each patch by 2-D Discrete Cosine Transform (DCT). The GMM contains 256 Gaussian components, while histogram contains 256 bins.

In Table 5.1, we compare the performance of our HG based algorithms to the state-of-the-art face recognition algorithms on the database, including the Eigen-Faces (PCA) [Turk 1991], Fisher-Face (LDA) [Bellhumeur 1996], Laplacian-Face (LPP) [He 2005], regularized Fisher-Face (RLDA) [Cai 2007a], spatially smooth Fisher-Face (SLDA) [Cai 2007b], and the Elastic and Partial Matching Metric (EPMM) [Hua 2009]. The recognition error rates (i.e. 1- recognition rate) of all the methods are summarized in Table 5.1. For all the results in the table, a random subset with 30 images per individual is taken to form the training set. All the learned metrics are trained on the gallery faces. The proposed HGV based face recognition algorithm performed significantly better than all the existing methods. Comparing with the second best method, EPMM [Hua 2009], we get 83.3% relative error reduction.

Table 5.1: The recognition error rates of the proposed HG representation on the PIE dataset compared with the state-of-the-art.

<table>
<thead>
<tr>
<th>Method</th>
<th>Recognition Error rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>28.0 ± 0.61</td>
</tr>
<tr>
<td>LDA</td>
<td>7.9 ± 0.37</td>
</tr>
<tr>
<td>LPP</td>
<td>7.5 ± 0.31</td>
</tr>
<tr>
<td>RLDA</td>
<td>4.3 ± 0.27</td>
</tr>
<tr>
<td>SLDA</td>
<td>3.6 ± 0.27</td>
</tr>
<tr>
<td>EPMM</td>
<td>2.4 ± 0.20</td>
</tr>
<tr>
<td><strong>HG</strong></td>
<td><strong>0.4 ± 0.09</strong></td>
</tr>
</tbody>
</table>

Table 5.2: Face recognition error rate (%) on PIE database.

<table>
<thead>
<tr>
<th>Method</th>
<th>10 Train</th>
<th>20 Train</th>
<th>30 Train</th>
</tr>
</thead>
<tbody>
<tr>
<td>Histogram</td>
<td>31.2 ± 0.69</td>
<td>21.5 ± 0.63</td>
<td>17.8 ± 0.53</td>
</tr>
<tr>
<td>GMM+DAP</td>
<td>3.9 ± 0.52</td>
<td>1.1 ± 0.21</td>
<td>0.5 ± 0.08</td>
</tr>
<tr>
<td><strong>HG</strong></td>
<td><strong>3.7 ± 0.45</strong></td>
<td><strong>0.9 ± 0.18</strong></td>
<td><strong>0.4 ± 0.09</strong></td>
</tr>
</tbody>
</table>

Table 5.2 further analyzes the recognition results by our HG representation for different setting.
Note that:

1. The accuracies of histogram representation are much worse than the performance by HG. The results indicate that the details of appearance are quite important for face recognition, while histogram representations have difficulties in capturing these details.

2. Unlike scene/object recognition task, the spatial layout does not show much benefit in performance improvement for face recognition. This seems reasonable since the difference between face images is mostly due to detailed appearance rather than the spatial configuration.

5.3.2 The experiments on LFW database

The Labeled Face in the Wild (LFW) dataset contains 13,233 images of faces (from 5749 people) collected from the web. Each face is labeled with the name of the person pictured. 1680 of the people pictured have two or more distinct photos in the data set. The evaluation of the quality of a face recognition algorithm on LFW is to classify a pair of faces as either match or nonmatch based on the distance between them. From different threshold settings, a ROC curve is generated. In the experiments on LFW dataset, we are able to achieve far better results than all existing algorithms. The ROC curves are shown in Figure 5.2. It can be observed that, based on single descriptor/feature (DCT2), our algorithm performs much better than the state-of-the-art algorithm (“combine b/g”), which is quite computational expensive by combining 5 features and more than 10 kernels.

5.4 Human Age Estimation

5.4.1 Human aging databases

The YGA database contains 8000 Japanese facial images of 1600 persons with ages ranging from 0 to 93. Each person has 5 images and the YGA database is divided into two subsets with 4000
images from 800 males and another 4000 images from 800 females. Our experiments are carried out separately on female and male subsets. For each subset, 1000 images are randomly selected for model training while the remaining 3000 samples are used for testing, and the configurations of the training and testing sets are the same as in [Yan 2007b]. The FG-NET aging database contains 1002 face images of 82 persons with ages ranging from 0 to 69. For both databases, the image is cropped and scaled to 32-by-32 pixels, and some example images of one person from the FG-NET database are depicted in Figure 5.3. For comparison, the results from the traditional regression

Figure 5.2: The performance comparison of state-of-the-art face recognitions on LFW dataset.

Figure 5.3: Sample images of one person in the FG-NET database.
algorithms, Quadratic Models (QM) [Lanitis 2004], Neural Networks [Rumelhart 2002], and the Nonlinear Regression with Uncertain Nonnegative Labels (RUN) algorithm [Yan 2007b], are used as baselines to evaluate the performance of our RPK framework. For the YGA database, the latest results are from the RUN algorithm as reported in [Yan 2007b], and for the FG-NET database the best results are reported in [Yan 2007a], where the evaluation protocol is Leave-One-Person-Out.

5.4.2 Experiments setup

In this work, we used two measures to evaluate algorithmic performance. The first one is the Mean Absolute Error (MAE) criterion as used in [Geng 2006, Lanitis 2004]. MAE is defined as the average of the absolute errors between the estimated labels and ground truth labels, i.e.,

\[ MAE = \frac{1}{N_t} \sum_{i=1}^{N_t} |\hat{a}_i - a_i| \]

where \( \hat{a}_i \) is the estimated age or pose for the \( i \)th testing sample, \( a_i \) is the corresponding ground truth, and \( N_t \) is the total number of the testing samples. Another popular measure is the cumulative score [Geng 2006] defined as:

\[ CumScore(\theta) = \frac{N_{e\leq\theta}}{N_t} \times 100\% \]

where \( N_{e\leq\theta} \) is the number of samples on which the absolute errors are not higher than \( \theta \).

For these two aging databases, the patch size is set as 6-by-6 pixels, and for each image, the patches are densely sampled pixel by pixel within the image plane. The GMM contains 512 Gaussian components.

For the CHIL head pose database, there exist four images for each sample; hence, the CHIL database is larger than the other two databases. To speed up the process, we train four GMMs for these four images respectively, and finally combine them to compute the supervectors. The patch size is set as 5-by-5 pixels, and each GMM contains 256 components. For all experiments, the subspace matrix \( V \) contains 50 columns, and the parameters \( \delta_1 \) and \( \delta_2 \) are selected empirically.

5.4.3 Human age estimation results on YGA database

Figure 5.4 depicts the cumulative scores from RPK and the other three comparison algorithms, and Table 5.3 lists the detailed MAEs of these algorithms. Notice that:
1. The MAE of human age estimation is substantially reduced from 9.78 years (best reported result [Yan 2007b]) to 4.94 years for the female subset, and 10.36 years (best reported result [Yan 2007b]) to 4.38 years for the male subset. On average, an MAE reduction of more than 50% is achieved compared with the best results ever reported.

2. Our proposed patch-kernel based regression framework performs perfectly in the age range of [0, 9]. This is quite different from the behavior of the other three comparison algorithms, which perform particularly poorly in this age range.

Table 5.3: MAEs of different algorithms on the YGA database.

<table>
<thead>
<tr>
<th></th>
<th>Female@YGA</th>
<th></th>
<th>Male@YGA</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Range</td>
<td>RPK</td>
<td>RUN</td>
<td>QM</td>
</tr>
<tr>
<td>0-9</td>
<td>1.83</td>
<td>11.21</td>
<td>11.97</td>
<td>14.33</td>
</tr>
<tr>
<td>20-29</td>
<td>5.27</td>
<td>7.95</td>
<td>9.29</td>
<td>9.70</td>
</tr>
<tr>
<td>40-49</td>
<td>6.73</td>
<td>8.64</td>
<td>10.45</td>
<td>8.78</td>
</tr>
<tr>
<td>50-59</td>
<td>5.37</td>
<td>9.43</td>
<td>10.15</td>
<td>9.53</td>
</tr>
<tr>
<td>70-93</td>
<td>5.22</td>
<td>15.56</td>
<td>19.66</td>
<td>16.52</td>
</tr>
<tr>
<td>Average</td>
<td>4.94</td>
<td>9.79</td>
<td>11.80</td>
<td>11.03</td>
</tr>
</tbody>
</table>

5.4.4 Human age estimation results on FG-NET database

For the FG-NET database, warped appearance features are conventionally used for image representation. First, 68 key facial points are labeled for each image, and then the shape, texture, and appearance models are trained based on all the samples. Finally the first 200 appearance parameters [Geng 2007] from the appearance model are used to represent each face image. For detailed information on shape, texture, and appearance models, please refer to [Cootes 2001]. In practical systems, face alignment is still a tough problem, especially for the cases with pose and expression variations as shown in Figure 5.3.

RPK works directly on original raw image features without the requirement of face alignment. Figure 5.5 shows the cumulative scores from RPK and the other comparison algorithms, and Ta-
ble 5.4 lists the detailed MAEs of these algorithms. From these results, we have the following observations: (1) even without face alignment, RPK still outperforms state-of-the-art algorithms that require precise face alignment; and (2) the age range of [0, 9] is again the one in which RPK has its best age estimation accuracy.

Table 5.4: MAEs of different algorithms on the FG-NET aging database. Note that BM below signifies the bilinear model used in [Yan 2007a].

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0-9</td>
<td>2.30</td>
<td>2.51</td>
<td>6.26</td>
<td>11.63</td>
</tr>
<tr>
<td>10-19</td>
<td>4.86</td>
<td>3.76</td>
<td>5.85</td>
<td>3.33</td>
</tr>
<tr>
<td>20-29</td>
<td>4.02</td>
<td>6.38</td>
<td>7.10</td>
<td>8.81</td>
</tr>
<tr>
<td>30-39</td>
<td>7.32</td>
<td>12.51</td>
<td>11.56</td>
<td>18.46</td>
</tr>
<tr>
<td>40-49</td>
<td>15.24</td>
<td>20.09</td>
<td>14.80</td>
<td>27.98</td>
</tr>
<tr>
<td>50-59</td>
<td>22.20</td>
<td>28.07</td>
<td>24.27</td>
<td>37.20</td>
</tr>
<tr>
<td>60-69</td>
<td>33.15</td>
<td>42.50</td>
<td>37.38</td>
<td>49.13</td>
</tr>
<tr>
<td>Average</td>
<td>4.95</td>
<td>5.78</td>
<td>7.57</td>
<td>10.39</td>
</tr>
</tbody>
</table>


5.4.5 Algorithmic analysis

In this subsection, we give an in-depth analysis of the effectiveness of the three components of the RPK framework, namely coordinate patch representation, inter-modality similarity synchronization, and kernel regression. Then we evaluate the algorithm’s robustness to image occlusions.
Figure 5.5: Cumulative scores of age estimation for the QM, MLP, RUN [Yan 2007b], and RPK algorithms at error levels from 0 to 10 years on the FG-NET aging database.

**Effectiveness of individual components of RPK**

In this subsection, we evaluate the effectiveness of the individual components of RPK on the YAMAHA-Female subset. For each experiment, we remove one component of RPK, and conduct the regression based on the other two components. More specifically, when the coordinate patch is not used, we use the free-patch instead; and when the kernel regression component is removed, we predict the label of a new datum as the label mean of the nearest $X_m$. Detailed comparison results are shown in Figure 5.6 as confusion matrices, from which we can observe that: (1) the removal of any component degrades the overall performance; and (2) the inter-modality similarity synchronization component proves to be the most important in the whole framework. Note that RPK without kernel regression achieves better group-based classification accuracy, but at the expense of higher variance of the regression output, hence the overall regression performance is much worse than that of RPK. Apparently the kernel regression component smooths the outputs from different reference sets.
Figure 5.6: Comparison confusion matrices on the YAMAHA-Female subset for the original RPK, RPK with free-patches, RPK without kernel regression, and RPK without inter-modality similarity synchronization (IMSS). For better viewing, please see the pdf file.

Robustness to image occlusions

To demonstrate the algorithm’s robustness to image occlusions, we systematically evaluate the performance of RPK on testing images with manually synthesized occluded patches of different sizes. Results are shown in Figure 5.7. When the size of the occluded patch is not larger than 6-by-6 pixels, RPK is quite robust to image occlusions. When the patch size is 12-by-12 pixels, the performance of RPK is still much better than the best result reported in [Yan 2007b] for images without occlusion.
5.5 Human Gender Recognition

5.5.1 Data sets and experimental setups

The gender recognition experiments are also conducted on YGA databased. We apply the OpenCV face detector on each of the 8000 images. 7620 face images were detected after we manually remove the false alarms. These detected face images are used directly in our experiments without any alignment process. In each experiment below, we randomly pick a certain ratio of data for training, and the remaining part is used for testing. We split the dataset in a rigorous manner to ensure that all faces of the same person reside in either training or testing set. We also vary the training ratio from 0.1 to 0.9 in order to examine how performance increases with more training data. The accuracy of gender recognition in each experiment is reported as the average of 10 individual runs.

Each detected face image is first converted into a $64 \times 64$ grayscale image. We then extract local descriptors within $6 \times 6$ windows with 1 pixel overlap in both horizontal and vertical direction. The local descriptor is here calculated as 2D Discrete Cosine Transform (DCT) of the 36-dimensional vector of concatenated pixels in the sliding window. The final feature vector is therefore 38 dimensions after $(x, y)$ coordinates appended.
5.5.2 Gender recognition results

Figure 5.8 shows the recognition accuracy of the HG representation (red curve) compared with the GMM for appearance representation only. To compare the patch-based feature and holistic feature, we also report results from SVM classifier on raw pixel vectors (blue curve). It is obvious that HG significantly outperforms both GMM with appearance information only and SVM on holistic features, with about 40% and 50% relative error reduction respectively. It is also indicated in Figure 5.8 that 50% of training data is enough to achieve a promising performance.

![Graph showing gender recognition results](image)

Figure 5.8: Gender recognition result on YGA database. The horizontal axis stands for the percentage of data used for training. The vertical axis represents the average accuracy of 10 runs for gender recognition. As illustrated, the performance of our HG representation achieved more than 40% relative error reduction compared with GMM representation and SVM classifier based on holistic features.

5.6 Head Pose Estimation

5.6.1 CHIL head pose database

For the CHIL data in the CLEAR07 evaluation, each sample consists of four images captured by four cameras. In our experiments, we use the experimental configuration proposed by the evaluation committee. For training, 10 videos are provided with the annotations of the head bounding
Table 5.5: MAEs of the algorithms PCA, LEA, SSE and RPK on the CHIL data from the CLEAR07 evaluation.

<table>
<thead>
<tr>
<th>Pan Angle</th>
<th>Subject-1</th>
<th>Subject-2</th>
<th>Subject-3</th>
<th>Subject-4</th>
<th>Subject-5</th>
<th>Total Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>8.54</td>
<td>8.19</td>
<td>6.91</td>
<td>4.53</td>
<td>4.78</td>
<td>6.94</td>
</tr>
<tr>
<td>LEA</td>
<td>7.60</td>
<td>8.77</td>
<td>6.33</td>
<td>4.50</td>
<td>4.51</td>
<td>6.72</td>
</tr>
<tr>
<td>SSE</td>
<td>8.45</td>
<td>7.27</td>
<td>6.22</td>
<td>4.33</td>
<td>3.94</td>
<td>6.60</td>
</tr>
<tr>
<td>RPK</td>
<td>7.08</td>
<td>4.80</td>
<td>4.89</td>
<td>3.95</td>
<td>3.23</td>
<td><strong>4.96</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tilt Angle</th>
<th>Subject-1</th>
<th>Subject-2</th>
<th>Subject-3</th>
<th>Subject-4</th>
<th>Subject-5</th>
<th>Total Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>8.49</td>
<td>5.97</td>
<td>11.59</td>
<td>5.25</td>
<td>12.53</td>
<td>10.86</td>
</tr>
<tr>
<td>LEA</td>
<td>7.88</td>
<td>5.74</td>
<td>12.29</td>
<td>5.29</td>
<td>12.23</td>
<td>10.87</td>
</tr>
<tr>
<td>SSE</td>
<td>8.61</td>
<td>6.28</td>
<td>9.08</td>
<td>4.92</td>
<td>9.64</td>
<td>8.25</td>
</tr>
<tr>
<td>RPK</td>
<td>6.14</td>
<td>4.99</td>
<td>7.72</td>
<td>4.08</td>
<td>14.09</td>
<td><strong>6.66</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Roll Angle</th>
<th>Subject-1</th>
<th>Subject-2</th>
<th>Subject-3</th>
<th>Subject-4</th>
<th>Subject-5</th>
<th>Total Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>4.66</td>
<td>2.59</td>
<td>4.20</td>
<td>2.86</td>
<td>3.30</td>
<td>4.01</td>
</tr>
<tr>
<td>LEA</td>
<td>5.41</td>
<td>2.59</td>
<td>4.06</td>
<td>2.90</td>
<td>2.91</td>
<td>4.07</td>
</tr>
<tr>
<td>SSE</td>
<td>5.55</td>
<td>2.22</td>
<td>3.72</td>
<td>2.38</td>
<td>2.34</td>
<td>3.42</td>
</tr>
<tr>
<td>RPK</td>
<td>4.51</td>
<td>2.38</td>
<td>3.24</td>
<td>2.16</td>
<td>2.57</td>
<td><strong>3.02</strong></td>
</tr>
</tbody>
</table>

boxes and the original ground truth information on three pose angles, namely, pan, tilt, and roll.

For evaluation, 5 videos from 5 subjects are used. In total, the training set contains 5348 samples (each consists of four images), and the testing set contains 2402 samples. Each image is cropped and scaled to 24-by-24 pixels.

5.6.2 Head pose estimation results

For comparison, we implemented Principal Component Analysis (PCA), Locally Embedded Analysis (LEA) [Fu 2006], and Submanifold Synchronized Embedding (SSE) [Yan 2008a] which produced the best results as reported in the CLEAR07 evaluation. The detailed results on the three angles of head pose are listed in Table 5.5, from which we can observe that our framework performs the best among all the algorithms evaluated. Note that we carefully tuned the parameters for SSE, and hence the results reported here for SSE are a little better than those originally reported in the CLEAR07 evaluation.
Chapter 6
The Application of HG in Video Categorization

6.1 Implementation Details

Video based event recognition is an extremely challenging task due to huge within-event variations, such as unconstrained motions, cluttered backgrounds, object occlusions, environmental illuminations and geometric deformations of objects.

Our HG representation can be applied to video event recognition with the following observations. First, low-level global features, e.g. grid color moments, Gabor texture histogram, and edge direction histogram, are not sufficient to characterize the local details. Second, without the motion information, the accuracy of event recognition is still reasonably good as reported in [Xu 2007a]. Thirdly, the video mismatch may exist in both spatial and temporal domains; that is, a sub-cube of one video clip may correspond to a sub-cube of another video clip belonging to the same event, but their positions and scales may be greatly different in both spatial and temporal domains. The third observation suggests that video matching should be conducted based on smaller elements rather than whole frames or video clips.

Similar as in scene images, we extract SIFT features from each frame of a video clip. However, unlike the scene image case, where we use dense grid as the detector, here we use SIFT as both detector and descriptor. Then we use all the feature vectors of the video clip to learn a video-specific GMM and represent each video clip as a super-vector according to HG representation. Then we also use two linear classifiers, nearest centroid and linear SVM in the experiments.
6.2 Experiments

Our experiments are conducted over the large TRECVID 2005 video corpus as used in [Xu 2008] and we compare different configurations of our framework with one of the state-of-the-art algorithms, Temporally Aligned Pyramid Matching (TAPM) [Xu 2007a, Xu 2008].

6.2.1 Corpus and metric

As in [Xu 2007a], the following ten events are chosen from the LSCOM lexicon [Xu 2008, Naphade 2006, Yanagawa 2007]: Car Crash, Demonstration Or Protest, Election Campaign Greeting, Exiting Car, Ground Combat, People Marching, Riot, Running, Shooting, and Walking. They are chosen because these events are relatively frequent in the TRECVID data set [Naphade 2006] and are intuitively recognizable from visual cues. The number of video clips for each event class ranges from 54 to 877. When training the SVM, we use the video clips from the other nine events as the negative samples. We randomly choose 60\% of the data for training and use the remaining 40\% for testing, with the same configurations as in [Xu 2007a, Xu 2008].

It is computationally prohibitive to compute the similarities among video clips and train multiple SVMs with cross-validation over multiple random training and testing splits. Therefore, we reported the results from the split used in [Xu 2007a, Xu 2008]. In the experiments, the feature extraction for all video clips costs about four hours for a 15-node computer cluster with a dual-core 2.8GHz CPU and 1G memory for each node; the global GMM training costs about one hour; the MAP adaptation for all video clips costs about 80 minutes; the WCCN and SVM step, along with the final classification, is very fast, and can be finished within few minutes.

We use non-interpolated Average Precision (AP) [Smeaton 2006] as the performance metric, which is the official performance metric in TRECVID. It reflects the performance on multiple average precision values along a precision-recall curve. The effect of recall is also incorporated when AP is computed over the entire classification result set. Mean Average Precision (MAP) is defined as the mean of APs over all ten events.
6.2.2 Comparison with TAPM

TAPM is the state-of-the-art algorithm for video event recognition in unconstrained news video. We also got the result by Bag-of-Words quantization with SVM classification. Table 6.1 summarizes the comparison experimental results for different algorithms. From all these results, we have some interesting observations:

1. The mean average precision is boosted from the best reported 38.2% in [Xu 2008] to 60.4% based on our new framework with the fusing stage.

2. For the video event Election Campaign Greeting, the average precision is dramatically increased from the 13.9% to 94.8%.

3. The fusion of the two classifiers can further improve the average precision compared with the single classifier.

4. The centroid-based algorithm, namely HG+NC, is comparable to the margin-based algorithm, namely HG+SVM.

5. The proposed framework does not achieve the performance of the TAPM algorithm for the video event Exiting Car. A possible explanation is that our framework does not explicitly pursue temporal information, and the video event Exiting Car depends on the temporal contextual information heavily.

6.2.3 Extensive study

In this subsection, we present an extensive study of our new video analysis framework in three aspects discussed below.
Table 6.1: Comparison of Average Precision (%) using different algorithms. Note that: (1) TAPM-1 is the TAPM algorithm with same weights for all the three levels; [Xu 2008] (2) TAPM-2 refers to the TAPM algorithm with different weights for the three levels; [Xu 2008] (3) Hist+SVM refers to Bag-of-Words quantization with SVM classification; (4) HG+NN is the algorithm based on HG representation and Nearest Neighbor classifier; (5) HG+SVM means HG representation with SVM classification; (6) HG+NC refers to the centroid-based algorithm using DAP; and (7) NC+SVM refers to the algorithm based on the fusion of two classifiers. The last row, referred to as Mean AP, is the mean of APs over ten events.

<table>
<thead>
<tr>
<th>Event Name</th>
<th>TAPM-1</th>
<th>TAPM-2</th>
<th>Hist+SVM</th>
<th>HG+NN</th>
<th>HG+SVM</th>
<th>HG+NC</th>
<th>NC+SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Car Crash</td>
<td>51.1</td>
<td>51.0</td>
<td>33.0</td>
<td>33.5</td>
<td>39.7</td>
<td>46.5</td>
<td>53.3</td>
</tr>
<tr>
<td>Demonstration</td>
<td>23.6</td>
<td>23.6</td>
<td>38.2</td>
<td>38.3</td>
<td>49.3</td>
<td>48.5</td>
<td>50.1</td>
</tr>
<tr>
<td>Election Campaign</td>
<td>13.9</td>
<td>13.7</td>
<td>82.5</td>
<td>79.2</td>
<td>92.6</td>
<td>94.8</td>
<td>94.4</td>
</tr>
<tr>
<td>Exiting Car</td>
<td>50.7</td>
<td>50.1</td>
<td>22.1</td>
<td>31.5</td>
<td>35.2</td>
<td>33.9</td>
<td>38.1</td>
</tr>
<tr>
<td>Ground Combat</td>
<td>44.2</td>
<td>44.1</td>
<td>68.1</td>
<td>58.2</td>
<td>71.4</td>
<td>72.8</td>
<td>73.4</td>
</tr>
<tr>
<td>People Marching</td>
<td>25.8</td>
<td>25.8</td>
<td>70.0</td>
<td>67.7</td>
<td>75.8</td>
<td>76.9</td>
<td>78.7</td>
</tr>
<tr>
<td>Riot</td>
<td>22.7</td>
<td>22.9</td>
<td>16.9</td>
<td>30.9</td>
<td>24.9</td>
<td>25.4</td>
<td>27.7</td>
</tr>
<tr>
<td>Running</td>
<td>86.7</td>
<td>86.6</td>
<td>88.1</td>
<td>89.3</td>
<td>91.4</td>
<td>89.9</td>
<td>91.9</td>
</tr>
<tr>
<td>Shooting</td>
<td>10.4</td>
<td>9.9</td>
<td>18.0</td>
<td>20.0</td>
<td>21.9</td>
<td>22.7</td>
<td>23.1</td>
</tr>
<tr>
<td>Walking</td>
<td>52.4</td>
<td>52.8</td>
<td>52.6</td>
<td>59.3</td>
<td>73.3</td>
<td>66.5</td>
<td>73.8</td>
</tr>
<tr>
<td>Mean AP</td>
<td>38.2</td>
<td>38.1</td>
<td>49.0</td>
<td>50.8</td>
<td>57.6</td>
<td>57.8</td>
<td>60.4</td>
</tr>
</tbody>
</table>
SIFT-Bag Visualization

A SIFT-Bag consists of the ensemble of SIFT feature vectors extracted from a video clip. We present a visualization to show that by modeling the SIFT feature vector distribution of each SIFT-Bag using a GMM, we implicitly establish the correspondence between the variant numbers of SIFT feature vectors in two video clips.

First, we project the SIFT feature vector into a 2D feature space using dimensionality reduction techniques, e.g. Locality Preserving Projection [He 2003b]. All the component means of the global GMM are mapped to this 2D space. For each SIFT feature vector, its coordinates in this 2-D space are the sums of the coordinates of the component means of the global GMM, weighted by the posteriors of the components for the given SIFT feature vector.

Figure 6.1 shows the 2D distributions of the SIFT-Bags from three video clips, two of which belong to the same video event Election Campaign Greeting, and the other to the video event Running. We can see that the SIFT feature vector distributions in the 2D space are characterized by distribution within different components, as indicated by the different colors in Figure 6.1. These components implicitly establish the correspondence between feature vectors in different SIFT Bags, which shows that SIFT-Bag Kernel offers the capability to match the patches from two video clips, similar in content yet different in spatial positions, scales, and temporal positions. For the video clips from the same event, we can see that the feature vector distributions within the corresponding components tend to share a similar structure, while they are relatively more different for those from different events.

Evaluation Using Confusion Matrices

Besides comparing our framework with the TAPM using average precision shown in Table 6.1, we present more detailed performance comparison using confusion matrices shown in Figure 6.2.

From these confusion matrices, we observe that: (1) when evaluated by the confusion matrices, the fusion of classifiers improves the recognition accuracy; (2) the better the overall recognition
accuracy is, the more possible the video event *Shooting* is misclassified. A possible explanation is that the event of *Shooting* is visually very similar to the event of *Ground Combat*, and cannot benefit from the improved discriminating power for most general events.

**Algorithmic Robustness**

For video event recognition, the boundaries of video clips are usually ambiguous, and frame rates of video clips vary a lot. A good algorithm should be robust to these factors. We therefore present a set of experiments to evaluate the algorithmic robustness to these factors. In these experiments only a random portion of the frames of each video clip are used to construct the SIFT-Bag. Other setups of the video event recognition framework remain unchanged.

The detailed experimental results are shown in Figure 6.3, with nine configurations using por-
Figure 6.2: Comparison of confusion matrices for HG + Nearest Neighbor, HG + SVM, HG + DAP, and HG + fusion of WCCN and SVM. Note that the first value in the title is the mean average precision, and the second value is the overall recognition accuracy. For better viewing, please see the original pdf file.

From these results, we can see that our framework is robust to the variation of boundaries and the frame rates of video clips. In particular, even when only 20% of the frames are used, our result (55.3%) still outperforms the best result (38.2%) reported in [Xu 2008].
Figure 6.3: The comparison of mean average precisions of different algorithms using randomly sampled 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, and 100% percentages of the frames within each test video clip.
Chapter 7

The Application of HG in Object Detection

Object detection predicts the bounding box of a specific object class within the image. Effective object detection relies on an efficient and effective searching method, and a robust image representation and learning method. The task remains challenging due to intra-class variations and the large search space for candidate bounding boxes.

While the HG representation proves effective in the above visual recognition tasks, all these are classification or regression problems working on the whole images. In contrast, the object detection or localization problem finds the rectangle bounding boxes for instances of a particular object with varying locations, widths and heights. However, it is not clear how to use the HG representation to capture localized information besides global information in an image. In this chapter, we apply our HG representation on the object detection problem.

A straightforward way to carry out detection is the sliding window approach [Rowley 1996], which applies learned classifiers over all candidate bounding boxes. However, an exhaustive search in an \( n \times n \) image needs to evaluate \( O(n^4) \) candidate bounding boxes, and is not affordable with complex classifiers. Tricky heuristics about possible bounding box locations, widths and heights, or local optimization methods would have to be used, resulting in false estimates. This intrinsic tradeoff between performance and efficiency is not desirable, particularly for applications in consumer electronics that are highly efficiency sensitive. In recent years, the most popular technique in the sliding window approach is the cascade [Viola 2001], which decomposes a strong object/non-object classifier into a series of simpler classifiers arranged in a cascade. However, the cascade is slow to train and involves many empirical decisions. Moreover, it always reduces the performance compared with the original strong classifier. As an alternative to the sliding window
approach, Lampert et al. introduced a branch-and-bound search scheme [Lampert 2008], which finds the globally optimal bounding box efficiently without the above problems.

Therefore, we present an object detection approach combining the efficient branch-and-bound searching method with our HG representation. The branch-and-bound search scheme is adopted to perform fast hierarchical search for the optimal bounding boxes, leveraging a quality bound for rectangle sets. We demonstrate that the quality function based on the HG representation can be written as the sum of contributions from each feature vector in the bounding box. Moreover, a quality bound can be obtained for any rectangle set in the image, with little computational cost, in addition to calculating the HG representation for the whole image.

We carry out an object detection experiment on a multi-scale car dataset. The results show the proposed object localization approach based on the HG representation outperforms a similar system using the branch-and-bound search based on the histogram-of-keywords representation. This suggests the HG representation can be effective for the detection problem in addition to the classification and regression problems.

### 7.1 Detection with HG Representation

#### 7.1.1 Branch-and-bound search

Detection of an object is essentially to find the subarea in the image on which a quality function $f$ achieves its maximum, over all possible subareas. One way to define these subareas is the bounding box, which encodes the location, width and height of an object with four parameters, i.e., the top, bottom, left and right coordinates $(t, b, l, r)$.

The sliding window approach is most widely used in object detection with bounding boxes [Rowley 1996, Dalal 2005]. To find the bounding box where the quality function $f$ reaches its global maximum, we need evaluate the function on all possible rectangles in the image, whose number is on the order of $O(n^4)$ for an $n \times n$ image. To reduce the computational cost, usually
only rectangles at a coarse location grid and of a small number of possible widths and heights are considered. On the other hand, different approaches can be adopted to use a local optimum to approximate the global one, when the quality function $f$ has certain properties, such as smoothness. All these approaches make detection tractable at the risk of missing the global optimum, and with demand for well informed heuristics about the possible location and sizes of the object.

In recent years, the most popular technique in the sliding window approach is the cascade [Viola 2001]. The cascade technique decomposes a strong object/non-object classifier into a series of simpler classifiers. These classifiers are arranged in a cascade, so that the simpler and weaker classifiers will eliminate most of the candidate bounding boxes, before the more powerful and complicated classifiers will make finer selection. However, the cascade of classifiers is slow to train. Moreover, it unfortunately involves many empirical decisions, e.g., choosing the false alarm rate and missing rate at each stage of the cascade. The cascade technique always reduces the performance compared with the original strong classifier.

The branch-and-bound search scheme was recently introduced [Lampert 2008] to find the globally optimal bounding box without the heuristics and assumptions about the property of the quality function. It hierarchically splits the parameter space of all the rectangles in an image, and discards large parts if their upper bounds fall lower than an examined rectangle.

For localization based on bounding boxes, a set of rectangles is encoded with $[T, B, L, R]$, each indicating a continual interval for the corresponding parameter in $(t, b, l, r)$. The approach starts with a rectangle set containing all the rectangles in the image, and terminates when one rectangle is found that has a quality function no worse than the bounds $\hat{f}$ of any other rectangle set.

At every iteration, the parameter space $[T, B, L, R]$ is split along the largest of the four dimension, resulting in two rectangle sets both pushed into a queue together with their upper bounds. The rectangle set with the highest upper bound is retrieved from the queue for the next iteration.

The steps of the branch-and-bound search scheme can be summarized as follows:

1. Initialize an empty queue $Q$ of rectangle sets. Initialize a rectangle set $R$ to be all the rectangles: $T$ and $B$ are both set to be the complete span from zero to the height of the
image. $L$ and $R$ are both set to be the complete span from zero to the width of the image.

2. Obtain two rectangle sets by splitting the parameter space $[T, B, L, R]$ along the largest of the four dimension.

3. Push the two rectangle sets in Step 2 into queue $Q$ with their respective quality bound.

4. Update $R$ with the rectangle set with the highest quality bound in $Q$.

5. Stop and return $R$ if $R$ contains only one rectangle $R$. Otherwise go to Step 2.

The quality bound $\hat{f}$ for a rectangle set $R$ should satisfy the following conditions:

1. $\hat{f}(R) \geq \max_{R' \in R} f(R')$

2. $\hat{f}(R) = f(R)$, if $R$ is the only element in $R$

Critical for the branch-and-bound scheme is to find the quality bound $\hat{f}$. Given the proven performance of the HG representation in classification tasks shown in previous work [Yan 2008b, Zhou 2008a, Zhuang 2008, Zhou 2008b], we are motivated to design a quality bound based on this representation, to enable localization based on this representation.

### 7.1.2 Quality function

For the HG representation, the binary classification score informs the confidence that the evaluated image subarea contains the object instead of pure background. Therefore, we can use this score as the quality function for the HG representation.

In particular, the quality function $f$ can be defined as follows:

$$f(Z) = g(Z) = \sum_t \alpha_t \phi(Z) \cdot \phi(Z_t) - b,$$

(7.1)
which can be expanded as

\[
 f(Z) = \sum_t \alpha_t \sum_{k=1}^K \sqrt{\frac{w_c}{2} \Sigma_c^{-\frac{1}{2}} \mu_k} \cdot \sqrt{\frac{w_k}{2} \Sigma_k^{-\frac{1}{2}} \mu_k^t} - b
 = \sum_t \alpha_t \sum_{k=1}^K \frac{w_k}{2} \Sigma_k^{-1} \mu_k \bullet \mu_k^t - b.
\]

(7.2)

According to Equation 3.7, the adapted mean of an image-specific GMM is the sum of the feature vectors in the image, weighted by the corresponding posterior. Therefore,

\[
 f(Z) = \sum_t \alpha_t \sum_{k=1}^K \frac{w_k}{2} \Sigma_k^{-1} \frac{1}{n_k} \sum_{j=1}^H Pr(k|z_j) z_j \bullet \mu_k^t - b.
 = \sum_{j=1}^H \left\{ \sum_{k=1}^K \frac{1}{n_k} Pr(k|z_j) z_j \bullet \frac{w_k}{2} \Sigma_k^{-1} \sum_t \alpha_t \mu_k^t \right\} - b.
\]

(7.3)

7.1.3 Quality bound

We define the “per feature vector contribution” as the contribution of each feature vector in a subarea to the confidence that this subarea is the concerned object. In particular, the “per feature vector contribution” is defined as in Equation 7.4.

\[
 W_j = \sum_{k=1}^K \frac{1}{n_k} Pr(k|z_j) z_j \bullet \frac{w_k}{2} \Sigma_k^{-1} \sum_t \alpha_t \mu_k^t.
\]

(7.4)

Therefore, Equation 7.3 can be rewritten as Equation 7.5, showing that the quality function can
be viewed as the sum of contribution from all involved feature vectors.

\[
f(Z) = \sum_j W_j - b. \tag{7.5}
\]

Given a test image, if we approximate the term \( n_k \) with their values calculated on the whole image, the per feature vector contributions \( W_j, j \in 1, \ldots, H \) are independent from the bounding box within the test image. This means that we can precompute \( W_j \) and evaluate the quality function on different rectangles by summing up those \( W_j \) that fall into the concerned rectangle.

We design a quality bound for the Gaussianized vector representation in a way similar to the quality bound for histogram of keywords proposed in [Lampert 2008]. For a set of rectangles, the quality bound is the sum of all positive contributions from the feature vectors in the largest rectangle and all negative contributions from the feature vectors in the smallest rectangle. This can be formulated as

\[
\hat{f}(R) = \sum_{W_{j_1} \in R_{max}} W_{j_1} \times (W_{j_1} > 0) + \sum_{W_{j_2} \in R_{min}} W_{j_2} \times (W_{j_2} < 0), \tag{7.6}
\]

where \([T, B, L, R]\) are the intervals of \( t, b, l, r \) and \( R_{max} \) and \( R_{min} \) are the largest and the smallest rectangles.

We demonstrate that Equation 7.6 satisfies the conditions of a qualify bound for the branch-and-bound search scheme defined in Section 7.1.1.

First, the proposed \( \hat{f}(R) \) is an upper bound for all rectangles in the set \( R \). In particular, the qualify function evaluated on any rectangle \( R \) can be written as the sum of postive contributions
and negative contributions from feature vectors in this rectangle,

\[ f(R) = \sum_{W_j \in R} W_j \times (W_j > 0) + \sum_{W_j \in R} W_j \times (W_j < 0). \quad (7.7) \]

Obviously, given a rectangle set R, the first term in Equation 7.7 is maximized by taking all the positive contributions from the largest rectangle in the set. The second term in Equation 7.7 is negative and its absolute value can be minimized by taking all the negative contributions in the smallest rectangle.

Second, when the rectangle set R contains only one rectangle, \( R_{\text{min}} = R_{\text{max}} = R \). Equation 7.6 equals Equation 7.7,

\[ \hat{f}(R) = f(R). \quad (7.8) \]

This quality bound defined by Equation 7.6 is used in the branch-and-bound scheme discussed in Section 7.1.1 to achieve fast and effective detection and localization. Note that since the bound is based on sum of per feature vector contributions, the approach can be repeated to find multiple bounding boxes in an image, after removing those features claimed by the previously found boxes. This avoids the problem of finding multiple non-optimal boxes near a previously found box as in the sliding window approach.

Note that estimating \( W_j \) in Equation 7.4 involves no more computation than the calculation in a binary classifier using the Gaussianized vector representation of the whole image. To further expedite the localization, we can use two integral images [Viola 2001] to speed up the two summations in Equation 7.6 respectively. This makes the calculation of \( \hat{f}(R) \) independent from the number of rectangles in the set R.
7.2 Experiment

In this chapter, we carry out an object detection experiment using the proposed efficient object localization approach based on the HG representation. We compare the detection performance with a similar object detection system based on the generic histogram of keywords.

7.2.1 Dataset

We use a multi-scale car dataset [Agarwal 2004] for the detection experiment. There are 1050 training images of fixed size $100 \times 40$ pixels, half of which exactly show a car while the other half show other scenes or objects. Since the proposed detection approach has the benefit of requiring no heuristics about the possible locations and sizes of the bounding boxes, we use a test set consisting of 107 images with varying resolution containing 139 cars in sizes between $89 \times 36$ and $212 \times 85$. This dataset also includes ground truth annotation for the test images in the form of bounding rectangles for all the cars. The training set and the multi-scale test set are consistent with the setup used in [Lampert 2008].

A few sample test images of the dataset is shown in Figure 7.1. Note that some test images contain multiple cars and partial occlusion may exist between different cars as well as between a car and a “noise” object, such as a bicyclist, a pedestrian or a tree.

7.2.2 Metric

The detection performance is measured by recall, precision and F-measure, the same way as in [Agarwal 2004] and [Lampert 2008]. A hypothesized bounding box is counted as a correct detection if its location coordinates and size lie within an ellipsoid centered at the true coordinates and size. The axes of the ellipsoid are 25% of the true object dimensions in each direction. For multiple detected bounding boxes satisfying the above criteria for the same object, only one is counted as correct and the others are counted as false detections.
7.2.3 Results

To keep the setting the same as in [Lampert 2008], we search each test image for the three best bounding boxes, each affiliated with the quality function score. In particular, the branch-and-bound search scheme is applied to each test image three times. After each time, those features claimed by the found boxes are removed as discussed in Section 7.1.1.

The ROC curves, characterizing precision vs. recall, are obtained by changing the threshold on the quality function score for the found boxes. The equal error rate (EER) equals $1 - F$-measure when precision equals recall.

The ROC curves and the EER are presented in Figure 7.2 and Figure 7.3 respectively. We compare the results with a localization system using the same branch-and-bound scheme, but based on the generic histogram of keywords with 1000 entry codebook generated from SURF descriptors.
at different scales on a dense pixel grid [Lampert 2008].

Figure 7.2: ROC curves for multi-scale car detection. (“G-n” denotes the result using n components in the Gaussianized vector representation. “Histogram” denotes the performance using the generic histogram-of-keywords approach by Lampert et al.)

We can see that the HG representation outperforms the histogram of keywords in this multi-scale object detection task. In particular, using 64 Gaussian components gives the best performance.

In Figure 7.4, we present a few examples of correct detection and erroneous detection in the best setting in Figure 7.3. Each test image is accompanied by a “per-feature-contribution” map. Negative and positive contributions are denoted by blue and red, with the color saturation reflecting absolute values. The quality function evaluated on a bounding box is the sum of all the per-feature-contributions, as discussed in Section 7.1.

The examples of correct detection demonstrate that the system can effectively localize one or
7.3 Conclusion and Discussion

In this chapter, we discuss effective object localization leveraging an efficient and effective searching method, and a robust image representation and learning method. In particular, we present an
efficient object localization approach based on the HG representation. We design a quality bound for rectangle sets characterized by the HG representation. This bound can be obtained for any rectangle set within the image boundaries, with little extra computational cost, in addition to calculating the HG representation for the whole image classification problem. Adopting the branch-and-bound search scheme, we leverage the proposed quality bound for fast hierarchical search. The proposed object localization approach based on the HG representation outperforms a similar localization system based on the generic histogram-of-keywords representation on a multi-scale car dataset. This is the first work using the HG representation in object detection and localization.
Chapter 8

The Application of HG in Image Parsing

8.1 Introduction

Typical natural images contain multiple regions with each image region being a set of pixels grouped based on homogeneity in terms of location, appearance and smoothness constraint. The image parsing process gives object labels to image regions so that the most probable interpretation of the input image can be achieved. It also provides information such as shape (where is its boundary), semantics (what is the probability of the region belonging to each object class) and context (who are neighboring regions). Image parsing functionality is one of the most important features in the human visual system (HVS) because it provides necessary support to higher-level understanding of the physical world by human brain.

In computer vision research, abundant progress has been achieved in several well-defined tasks, including three most fundamental ones: classification [Lazebnik 2006b, Perronnin, van Gemert, Chum 2007] which focuses on predicting presence/absence of a specific object class in the given image, detection [Viola 2001][Lampert 2008, Dalai 2005, Ferrari 2008] which predicts the bounding box of certain objects, and segmentation [Winn 2006, Heitz] which further provides a pixel-wise boundary of each presented object. These three tasks can all be regarded as the image parsing problem with differences in the granularity of information to provide.

One of the major reasons for separating these related tasks into different problems is that, since these tasks provide levels of information about the image, the adopted algorithm requires significantly different amount of computation, which usually renders a method in one task infeasible for another tasks. For instance, in the segmentation task, pixel-level granularity is required, and
hence the popular method, such as Normalized Cut [Shi 2000], is very computationally demanding. When only rectangle bounding box is required as in the detection task, lighter searching methods such as sliding-window [Lampert 2008] dominate. In the image classification task where only a binary decision of the presence of a certain object class is required, many low-latency image concept detection engineers are readily available.

Such a situation has limited the possibility to extend existing techniques in classification, detection and/or segmentation to solve the image parsing problem, not to mention how to incorporate even more precise and complex models during the parsing process. One existing strategy to handle such a computation issue is to apply cascade structure [Viola 2001], where simpler scorers were firstly applied to filter out candidates that are unlikely to contain the object of interest. Then more complex models are used in the next step. The cascade structure has been proven effective in many object detection tasks such as face detection [Viola 2001].

However, it is difficult to directly apply cascade structure to the image parsing problem for two reasons: First, the simple scorer used to filter out low-quality candidate still requires huge computation in image parsing. To illustrate, given an image with 100 regions and 20 concept classes to parse, then there are $20^{100}$ possible image parsing results. Even with a simple linear classifier as scorer, the searching process will take $20^{100}$ vector multiplications to go through all these candidates, which is infeasible for modern computers. Second, the decision of whether a candidate is of high/low-quality requires one suitable threshold that is tuned based on certain global criteria. However, one cannot guarantee that such global optimum based on the simple scorer also applies to the more complex model in the cascade structure. In fact, the image parsing problem might exist in multiple modes, such that the global threshold selected from a simple scorer may falsely reject many high-quality candidates in different modes. Simply decreasing the threshold to allow more candidates for the complex model would increase computation.

In general, there are the following challenges in the image parsing problem:

1. The search model must be efficient to support large search space. The search space for image parsing is far beyond that for detection task as it supports unconstrained shapes and
deals with multiple categories labeling simultaneously.

2. The search algorithm must be able to handle the multi-mode issue in the image parsing problem. The search space is not necessarily linear, such that a local minimum exists. To search for the joint interpretation of multiple image regions, the parsing algorithm should give good global or near global solution.

3. The search method should impose less constraints on modeling the images. For example, although Efficient Subwindow Search [Lampert 2008] is very efficient, its linear classifier requirement constrains the usage of more powerful non-linear features and classifiers.

While the HG representation has proved effective in several whole image based recognition tasks, as well as object detection based on partial image, in this chapter, we introduce a novel framework, which can cooperate with HG to support image parsing. As presented in Figure 8.1, we first group pixels into regions to reduce the number of processing elements from millions of pixels to only a few hundreds of regions. Then we introduce a graph adaptive dynamic programming (GADP) algorithm to generate high-quality parsing hypotheses that satisfy all the three above-mentioned requirements. GADP takes into account not only regions themselves but also the contextual relationship among regions to significantly reduce the search space. The goal of GADP is to assign object label to each region and generate multiple hypotheses of region label assignment. GADP is extremely efficient and robust because it only searches for truly promising solutions while maintaining sufficient variety to avoid local optima. In this way, we are able to incorporate more precise and complex object models to achieve superior performance. In addition, since we handle the image parsing problem directly, the results can be utilized by the classification, detection and segmentation tasks in an uniform way.

As shown in our experiment, the proposed image parsing approach based on the HG representation is able to outperform state-of-the-art techniques in all three major tasks in the Pascal VOC challenge [Everingham 2009].
Figure 8.1: Graph adaptive dynamic programming (GADP). Given (a) a test image, we group pixels into regions to form (b) a region graph according to adjacent relationship and apply GADP approach to get (c) multiple candidates, then we adopt more precise model and classifier to obtain (d) final output.

8.2 Graph Adaptive Dynamic Programming for Multi-Hypotheses Generation

As explained in Section 8.1, although there is a very large number of candidate image representations we need to search for, most of them can be easily excluded. Hence we can borrow the idea from cascade structure where we need an approach to efficiently reduce the search space, and keep the variety of remaining candidates to deal with the multi-mode problem. However, most existing sampling based methods, such as Markov chain Monte Carlo (MCMC) [Tu 2005] and conditional random field (CRF) [Lafferty 2001], cannot satisfy both requirements. Hence this chapter introduces a novel Graph Adaptive Dynamic Programming (GADP) algorithm.

To begin with, Dynamic Programming (DP) is a well studied tool for solving sequential decision problems in an efficient way. It performs global optimization by locally optimizing a sub-problem. One typical scenario where DP has been extensively used is to find the optimal sequence of a fix number of moves, starting from point $i$ and ending at point $j$, with associated cost $\varphi(i, j)$. However, applying DP algorithm to the searching problem in image parsing opens two challenges: First, due to the nature of image, the topology of image units is more complicated than a sequential connection. Second, due to the existence of multi-mode, simply taking the top-N solutions by DP as the hypotheses is not feasible, because these hypotheses are usually too similar to each other as
observed in our experiment, which means that they fall into the same mode. Hence in this dissertation, we extend the classical DP to optimize with graph structure and to create multiple hypotheses with adjustable variety.

### 8.2.1 Graph adaptive dynamic programming

Let $\vec{X} = [x_1, x_2, \ldots, x_N]$ be the set of $N$ image regions and $\vec{S} = [s_1, s_2, \ldots, s_N]$ be the state array of the corresponding region. Define a state space $s_i \in S = \{0, 1, \ldots, K\}$ where $s_i = 0$ indicates the background category, and the rest indicates the $K$ foreground categories. Then the image parsing task strives to find the optimal interpretation $\hat{S}$ that maximizes $p(\vec{S}|\vec{X})$. Under the Bayesian framework, we want to solve

$$
\hat{S} = \arg \max_{\vec{S} \in \Omega} p(\vec{S}|\vec{X}) = \arg \max_{\vec{S} \in \Omega} p(\vec{X}|\vec{S})p(\vec{S}) ,
$$

where $p(\vec{X}|\vec{S})$ specifies the generative likelihood from $\vec{S}$ to $\vec{X}$, and $p(\vec{S})$ refers to the prior probability of the states array.

As demonstrated in Figure 8.2a, the regions obtained from the low-level modules are converted into a graph in the following way: Each node in the graph represents an image region $x_i$, and two nodes are connected if their corresponding regions are adjacent. Assuming the total number of nodes is $N$, we number each node according to the area of its corresponding region in descending order, i.e., larger region gets smaller index number. Next, we denote the set of nodes connected with node $i$ as $\vec{S}_i$, $i = 1, \ldots, N$. We further split the set into two parts: the subset of nodes whose numbers are less than $i$ is denoted as $\vec{S}_i^p$, and the subset of nodes with numbers larger than $i$ as $\vec{S}_i^f$.

According to chain rule, Eq.(8.1) expands as

$$
\hat{S} = \arg \max_{\vec{S} \in \Omega} \prod_{i=1}^{N} p(x_i|s_i)p(s_i|s_1, \ldots, s_{i-1})
$$

$$
= \arg \max_{\vec{S} \in \Omega} \prod_{i=1}^{N} p(x_i|s_i)p(s_i|\vec{S}_i^p) .
$$

(8.2)
The original image shown in Figure 1 is converted into region graph (a) where the center of each region is highlighted in red, and connections are highlighted in blue. Then GADP is proposed to solve graph structural decision problem (b), where the observed node is highlighted in green, and the current node is highlighted in star. The simple N-top approach gives almost identical solutions (c), while the proposed strategy gives very different results (d) that covers larger region in the search space.

The likelihood \( p(x_i|s_i) \) forms a \( K \) by \( N \) matrix, where the \( m^{th} \) column represents the likelihoods of the \( m^{th} \) node given different states. Then, taking the negative log-likelihood

\[
\zeta(s_i|S^p_i) = -\log (p(x_i|s_i)p(s_i|S^p_i))
\]

as the cost, the optimization aims at minimizing the following overall objective function:

\[
\hat{S} = \arg \min_{S \in \Omega} \sum_{i=1}^{N} \zeta(s_i|S^p_i).
\]

Similar to classic DP, to solve Eq.(8.4), in GADP we recursively optimize a sub-problem from node 1 to \( m \). In each iteration we increase \( m \), and the whole process stops when \( m = N \). For
each sub-problem, let $\varphi_m(k) = \sum_{i=1}^{m} \zeta(s_i = k | S^p_i)$ be the cost from node 1 to node $m$ on state $k$, $m = 1, \ldots, N$, the GADP algorithm is illustrated in Alg. 1.

**Algorithm 1 GADP**

| input: $p(x_i|s_i), p(s_i|s_{i-1})$ |
| output: Optimal path $(i_1, i_2, \ldots, i_M)$ |
| 1: for $k = 0$ to $K$ do {Initialization} |
| 2: $\varphi_0(k) \leftarrow 0$, |
| 3: end for {Recursion} |
| 4: for $k = 0$ to $K$ do |
| 5: for $m = 0$ to $N - 1$ do |
| 6: $\varphi_{m+1}(k) \leftarrow \min_{1 \leq l \leq K} [\varphi_m(l) + \zeta_m(l, k)]$ |
| 7: $\xi_{m+1}(k) \leftarrow \arg \min_{1 \leq l \leq K} [\varphi_m(l) + \zeta_m(l, k)]$ |
| 8: end for |
| 9: end for {Path backtracking} |
| 10: $i_M \leftarrow \arg \min_{1 \leq k \leq K} \xi_M(k)$ |
| 11: for $m = M - 1$ to 1 do |
| 12: $i_m \leftarrow \xi_{m+1}(i_{m+1})$ |
| 13: end for |

As explained above, in addition to generating a hypothesis efficiently, we further require multiple hypotheses with certain variation between each other to cover enough search space. It is possible to simply keep multiple costs for each node in DP to get several paths from Alg. 1. However, we noticed that with such a simple approach, the difference between the obtained hypotheses is negligible, as shown in the example in Figure 8.2c. Hence in order to generate multiple hypotheses with significant difference, we run GADP for several turns. In each turn, we take the top best path and adaptively update the likelihood matrix $p(x_i|s_i)$ according to current and all previous paths. The idea is that by changing $p(x_i|s_i)$, GADP will be forced to consider different paths, and thus to give multiple distinguishing hypotheses in an adjustable way. To depict, for each node, we set the likelihood of the currently selected states to zero so as to force the DP process to reconsider another path in successive runs. By controlling how many nodes to be changed, we can adjust the diversity of the solved paths by GADP. The more nodes being reset, the more diverse the finally generated hypothesis will be. This strategy significantly improves the quality of generated object
hypotheses. As can be seen from Figure 8.2d, the five obtained results differ from each other, and the best results can be observed in third hypothesis.

The cost function used in Eq. (8.2) contains two terms. The first term $p(x_i|s_i)$ indicates the likelihood of feature vectors given current state, and the second term $p(s_i|\vec{S}_i^p)$ represents the transition probability of current state given the seen state array. The first term is calculated by

$$p(x_i|s_i) = \frac{p(s_i|x_i)p(x_i)}{p(s_i)}$$

(8.5)

where $p(s_i|x_i)$ is a pseudo-posterior obtained by “region scoring” module as described in Section 3.

The computation for the second term, $p(s_i|\vec{S}_i^p)$, is not so straightforward. $\vec{S}_i^p$ varies when we change the order of the node sequence for an image, and simply sorting the index of the nodes according to the size of their corresponding regions does not make it a complete set.

On the other hand, let $\vec{S}_i$ represent the full surrounding of $s_i$ including both $\vec{S}_i^p$ and $\vec{S}_i^f$. Hence we propose to compute $p(s_i|\vec{S}_i^p)$ through $p(\vec{S}_i|s_i)$ as follows:

The second term $p(s_i|\vec{S}_i^p)$ is calculated by

$$p(s_i|\vec{S}_i^p) = \frac{p(\vec{S}_i^p|s_i)p(s_i)}{\sum_{s_i} p(\vec{S}_i^p|s_i)p(s_i)}$$

$$= \frac{p(s_i) \sum_{\vec{S}_i^f} p(\vec{S}_i^p, \vec{S}_i^f|s_i)}{\sum_{s_i} \sum_{\vec{S}_i^f} p(\vec{S}_i^p, \vec{S}_i^f|s_i)}.$$ 

(8.6)

Since the combinations of $\vec{S}_i^f$ are infinite, strictly computing Eq.(8.6) is impossible. Neither can we simply use a first order Markov assumption as in many sequential learning problems, because the distances between node $i$ to all its connected neighbors are the same according to our topology.

To solve this problem, we propose to simplify the computation by the following strategy: defining the relationship between $s_i$ and its surrounding states, $\vec{S}_i^f|s_i$ and $\vec{S}_i^p|s_i$, $\vec{S}_i|s_i$, has the following types:
**Same** All the nodes in set $\vec{S}_i$ take the same state value as $s_i$

**Different** None of the nodes in set $\vec{S}_i$ takes the same state value as $s_i$

**Partial** Part of the nodes in set $\vec{S}_i$ take the same state value as $s_i$

**None** The set $S_i$ is empty

Since $\vec{S}_i$ is the combination of $\vec{S}_i^f$ and $\vec{S}_i^p$, it is easy to see that the value of $\vec{S}_i|s_i$ is determined by $\vec{S}_i^f|s_i$ and $\vec{S}_i^p|s_i$. For example, if both $\vec{S}_i^f|s_i$ and $\vec{S}_i^p|s_i$ take “Same”, then $\vec{S}_i|s_i$ takes “Same” also; and if either $\vec{S}_i^f|s_i$ or $\vec{S}_i^p|s_i$ takes “Different”, then $\vec{S}_i|s_i$ takes “Different”.

Note that $p(\vec{S}_i|s_i)$ can be learned from the training set by simply counting. For example, $p(\vec{S}_i = \text{“Same”}|s_i = \text{“Car”})$ can be learned by counting how many times in the training set a region with label “Car” with all surrounding regions having label “Car” also.

Figure 8.2b illustrates our graph-based DP process, where the observed nodes are highlighted in green and the unobserved ones are in red. The current node is highlighted in star which connects with five nodes, one observed in green and four unobserved in red.

### 8.3 Implementation Detail

This section introduces the modules in our system that pre-process the original input image to generate input for the GADP algorithm, as well as modules that post-process the output of GADP to obtain the final image parsing results.

**Region Generation**: Since pixels in neighboring regions are very alike, it is highly redundant to perform parsing at the pixel level. In this module, we instead group similar pixels together to form a region which is the basic processing unit in our work. We simply use a commercial software called VectorMagic for the purpose. For an image with size of $500 \times 500$, this typically generates about 100 regions.

**Region Scoring**: This module obtains a probability of every region into one of the $K$ pre-defined categories. The module only considers local observation from each region. We adopt the HG
image representation to describe the appearance information of each region. Then a multi-class linear SVM is used to get the possibilities of the region being either of the $K$ categories. This becomes the input for the GADP algorithm for hypotheses generation.

**Hypotheses Verification:** In this module, the hypotheses generated by GADP is tested to get the final parsing result. We first combine regions with the same hypothesized labels into a single object hypothesis. Then any single object hypothesis or a set of several disjoint hypotheses are re-scored by another multi-class linear SVM. Similar to “Region scoring”, we also adopt the HG representation to describe the appearance information of each combination. Finally, the most probably parsing of the image is obtained.

**Global Level Analysis:** It is possible to go one step further than the inter-object level analysis in the “Hypotheses Verification” module. This eventually comes to a “Global level” analysis where the whole image is interpreted together. In this module, the HG representation is again adopted to describe the appearance information of the whole image. Similarly, a third multi-class linear SVM is used to calculate the confidence score of the image being each of the $K$ categories.

### 8.4 Experimental Results

We evaluated our proposed framework using the widely adopted PASCAL 2007 dataset [Everingham 2009]. It consists of 9,963 images with 24,640 annotated objects from 20 classes. These images range between indoor and outdoor scenes, close-ups and landscapes, and strange viewpoints. The dataset is an extremely challenging one because the appearances of objects and their poses vary significantly, with frequent occlusions. There are three major tasks in PASCAL 2007, specifically the “Segmentation”, “Detection” and “Classification”. Our intention to use the PASCAL 2007 dataset is to validate that the proposed technique focuses on solving the image parsing problem directly, hence its results can be used for all related tasks. In addition, due to the efficiency of the proposed GADP algorithm, we are able to apply more precise and complex models to improve performance. The following subsections discuss the results respectively.
8.4.1 Object segmentation performance

The segmentation task in PASCAL 2007 dataset uses a subset of images for training/evaluation. Each image in this subset has pixel-wise labeling of being one of the original 20 classes or background. The segmentation performance is measured by averaging segmentation accuracy across the 21 classes, where accuracy for a single class is the number of correctly labeled pixels of that class divided by the total number of pixels of that class in the ground truth.

Before presenting our results, we want to highlight that, while most participants in the original challenge used either the pixel-wise annotations in the subset images only, or together with the object bounding boxes available for the detection task, as the training label, in our method we only used the object bounding boxes as labels in training. Such a setup obviously adds to the practical value of the technique because otherwise obtaining pixel-wise annotation is extremely labor-intensive.

Table 8.1 lists the segmentation results. We also compared our results with the best performance of the original challenge [Everingham 2009], as well as that by [Shotton 2008] which used the same dataset and won “Best Demo” in CVPR 2008. As can be seen from Table 8.1, even when used without the global-level analysis (Section 8.3), our method achieves 44.5% accuracy, 2.5% higher than the best of the comparison results. This validates the effectiveness of the proposed hierarchical framework. In addition, when combined with the global-level analysis, our results further increase to 53.4%. Some example images are listed in Figure 8.3.

![Example segmentation results](image-url)

Figure 8.3: Example segmentation results.
Table 8.1: Object segmentation results.

<table>
<thead>
<tr>
<th>method</th>
<th>average accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best of PASCAL’07 [Everingham 2009]</td>
<td>30.4</td>
</tr>
<tr>
<td>STF [Shotton 2008]</td>
<td>42</td>
</tr>
<tr>
<td>Ours</td>
<td>44.5</td>
</tr>
<tr>
<td>Ours + global</td>
<td><strong>53.4</strong></td>
</tr>
</tbody>
</table>

### 8.4.2 Object detection performance

The detection performance is evaluated using the Average Precision (AP) measure which is a standard metric used by PASCAL challenge. It computes the size of the area under the Precision/Recall curve, and the higher the score, the better the performance. We report the AP scores for all 20 object classes as shown in Table 8.2. We also list the best performance of the original challenge [Everingham 2009] and that by [Felzenszwalb 2008] whose implementation was made publicly available.

As can be seen from Table 8.2, our method achieved the best AP score for 11 out of 20 classes. The mean AP improved 5.1% over [Felzenszwalb 2008], and 3.1% over the original challenge [Everingham 2009]. To give some insight of our detection results, Figure 8.4 illustrates some example result that our method improves most over [Everingham 2009] or [Felzenszwalb 2008]. Particularly, in contrast to [Felzenszwalb 2008] which works best for rigid object classes such as bicycle or TV that only present viewing angle change, our proposed method gives promising results for both rigid objects, such as aeroplane, and non-rigid objects, such as cat. We would also like to investigate those classes where our method could not outperform existing techniques. As can be noticed from Figure 8.5, in most of these classes, there are multiple neighboring objects of the same class. This posed a challenging situation to our candidate generation procedure. In fact, this is a common difficulty in object detection tasks, and one possibility is to adopt stronger features or scoring models, and/or incorporate domain-specific knowledge to effectively alleviate such limitations.
Table 8.2: Object detection results.

<table>
<thead>
<tr>
<th>object class</th>
<th>plane</th>
<th>bike</th>
<th>bird</th>
<th>boat</th>
<th>bottle</th>
<th>bus</th>
<th>car</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM [Felzenszwalb 2008]</td>
<td>18.0</td>
<td><strong>41.1</strong></td>
<td>9.2</td>
<td>9.8</td>
<td><strong>24.9</strong></td>
<td>34.9</td>
<td>39.6</td>
</tr>
<tr>
<td>Best PASCAL'07 [Everingham 2009]</td>
<td>26.2</td>
<td>40.9</td>
<td>9.8</td>
<td>9.4</td>
<td>21.4</td>
<td>39.3</td>
<td>43.2</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>34.8</strong></td>
<td>34.1</td>
<td><strong>14.5</strong></td>
<td><strong>13.1</strong></td>
<td>9.1</td>
<td><strong>43.7</strong></td>
<td><strong>39.9</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>object class</th>
<th>cat</th>
<th>chair</th>
<th>cow</th>
<th>table</th>
<th>dog</th>
<th>horse</th>
<th>mbike</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM [Felzenszwalb 2008]</td>
<td>11.0</td>
<td><strong>15.5</strong></td>
<td>16.5</td>
<td>11.0</td>
<td>6.2</td>
<td>30.1</td>
<td>33.7</td>
</tr>
<tr>
<td>Best of PASCAL'07 [Everingham 2009]</td>
<td>24.0</td>
<td>12.8</td>
<td>14.0</td>
<td>9.8</td>
<td>16.2</td>
<td>33.5</td>
<td>37.5</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>35.2</strong></td>
<td>9.7</td>
<td><strong>25.7</strong></td>
<td><strong>32.9</strong></td>
<td><strong>22.4</strong></td>
<td><strong>37.3</strong></td>
<td><strong>38.4</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>object class</th>
<th>person</th>
<th>plant</th>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tv</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM [Felzenszwalb 2008]</td>
<td><strong>26.7</strong></td>
<td><strong>14.0</strong></td>
<td>14.1</td>
<td>15.6</td>
<td>20.6</td>
<td><strong>33.6</strong></td>
<td>21.3</td>
</tr>
<tr>
<td>Best of PASCAL'07 [Everingham 2009]</td>
<td>22.1</td>
<td>12.0</td>
<td>17.5</td>
<td>14.7</td>
<td>33.4</td>
<td>28.9</td>
<td>23.3</td>
</tr>
<tr>
<td>Ours</td>
<td>18.7</td>
<td>11.0</td>
<td><strong>26.9</strong></td>
<td><strong>24.5</strong></td>
<td><strong>36.9</strong></td>
<td>19.1</td>
<td><strong>26.4</strong></td>
</tr>
</tbody>
</table>

Figure 8.4: Example detection results that improve most over [Everingham 2009].

8.4.3 Image classification performance

The classification performance is also evaluated using the AP metric as in subsection 8.4.2. The AP scores for all 20 object classes are shown in Table 8.3. We also compared our results with the best performance of the original challenge [Everingham 2009], as well as another recent results in [Uijlings 2009]. As can be seen, our final results that combine both verified objection

Figure 8.5: Example detection results without significant improvement.
hypotheses and global-level analysis outperform existing methods by 6.5% which is very impressive. Revisiting other promising results reported in previous subsections, the effectiveness of the introduced joint framework in modeling certain intrinsic image characterizations can be justified.

Note that the results of our global-level analysis can be applied for image classification already. Hence to give more insight about the effectiveness of combining both the high-level object detection results and global-level analysis, we further list the global-level analysis results, as well as the improvement after combination. As shown in Figure 8.6, although the global-level results are already remarkable, combination gains additional 0.1% to 11.4% improvement. This shows that the information obtained from image parsing captures different yet complementary patterns from the image. Figure 8.6 also lists the top ten images for the “sheep” and the “plant” class that benefit most from combining. As can be seen, those non-class images that get top ranked (highlighted in red) in the classification result are removed after combining.

Figure 8.6: Examples of classification ranking.
*top row: top-ten images ranked by global analysis only; bottom row: top-ten images ranked by global analysis and detection; non-class images are highlighted in red*
Table 8.3: Object classification results.

<table>
<thead>
<tr>
<th>object class</th>
<th>plane</th>
<th>bike</th>
<th>bird</th>
<th>boat</th>
<th>bottle</th>
<th>bus</th>
<th>car</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obj.+Contex [Uijlings 2009]</td>
<td>80.2</td>
<td>61.0</td>
<td>49.8</td>
<td>69.6</td>
<td>21.0</td>
<td>66.8</td>
<td>80.7</td>
</tr>
<tr>
<td>Best PASCAL’07 [Everingham 2009]</td>
<td>77.5</td>
<td>63.6</td>
<td>56.1</td>
<td>71.9</td>
<td>33.1</td>
<td>60.6</td>
<td>78.0</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>80.9</strong></td>
<td><strong>74.3</strong></td>
<td><strong>56.7</strong></td>
<td><strong>75.2</strong></td>
<td><strong>33.6</strong></td>
<td><strong>73.6</strong></td>
<td><strong>84.3</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>object class</th>
<th>cat</th>
<th>chair</th>
<th>cow</th>
<th>table</th>
<th>dog</th>
<th>horse</th>
<th>mbike</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obj.+Contex [Uijlings 2009]</td>
<td>51.1</td>
<td>51.4</td>
<td>35.9</td>
<td>62.0</td>
<td>38.6</td>
<td>69.0</td>
<td>61.4</td>
</tr>
<tr>
<td>Best of PASCAL’07 [Everingham 2009]</td>
<td>58.8</td>
<td>53.5</td>
<td>42.6</td>
<td>54.9</td>
<td>45.8</td>
<td>77.5</td>
<td>64.0</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>63.1</strong></td>
<td><strong>58.0</strong></td>
<td><strong>56.9</strong></td>
<td><strong>65.8</strong></td>
<td><strong>50.7</strong></td>
<td><strong>81.4</strong></td>
<td><strong>73.2</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>object class</th>
<th>person</th>
<th>plant</th>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tv</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obj.+Contex [Uijlings 2009]</td>
<td>84.6</td>
<td>28.7</td>
<td>53.5</td>
<td>61.9</td>
<td>81.7</td>
<td>59.5</td>
<td>58.4</td>
</tr>
<tr>
<td>Best of PASCAL’07 [Everingham 2009]</td>
<td>85.9</td>
<td>36.3</td>
<td>44.7</td>
<td>50.9</td>
<td>79.2</td>
<td>53.2</td>
<td>59.4</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>86.3</strong></td>
<td><strong>42.0</strong></td>
<td><strong>56.5</strong></td>
<td><strong>62.1</strong></td>
<td><strong>83.6</strong></td>
<td><strong>60.6</strong></td>
<td><strong>65.9</strong></td>
</tr>
</tbody>
</table>
Chapter 9

Conclusion

In the final chapter of the dissertation, we briefly summarize our contributions and discuss the potential directions for future work.

The motivation of the study is to seek a novel image representation, which is (1) robust to most of the variabilities; (2) able to give the correspondence across different images; (3) efficient enough for large scale real-time usage; (4) unified for images of various types and different applications.

The proposed HG image representation meets all of the aforementioned requirements. First, we model the feature vectors, from the whole corpus, from each image and at each individual patch, in a Bayesian hierarchical framework using mixtures of Gaussians. After this hierarchical Gaussianization, each image is represented as a Gaussian mixture model (GMM) for its appearance, and several Gaussian maps for its spatial layout. Then we extract the appearance information from the GMM parameters, and the locality information from global and local statistics over Gaussian maps. Finally, we employ a supervised dimension reduction technique called DAP (discriminant adaptive projection) to remove the noise directions and to further enhance the discriminating power of our representation. Also, we propose a random forest based fast computation algorithm to make the HG representation applicable to large scale real-time scenarios.

The advantages of HG representation are as follows:

- Taking advantage of local descriptors and GMM, the new representation is robust to most of the variabilities in image appearance, such as illumination variation, pose changes, etc; Gaussian maps are introduced to capture the locality information, which alleviates the bottleneck of the patch-based description.
• We introduce a supervised learning method, Discriminant Adaptive Projection (DAP), to further enhance the discriminating power of our representation.

• The computational cost of the proposed representation can be dramatically reduced by either using approximation algorithms such as random forest (without performance degeneration), or by parallel implementations based on CPU or GPU clusters. In this way, our system scales very well to large datasets and is suitable for use in large-scale real-time applications.

• The new representation is generic in nature and may be used with a variety of features (e.g., pixel intensities, Gabor wavelet responses [Daugman 1988], SIFT features [Lowe 1999], etc.) in a wide range of tasks (detection, recognition, classification, regression, etc.).

We also apply the new representation on several tasks, including object recognition, scene category classification, face recognition, age estimation, pose estimation, gender recognition, video event recognition, object detection and image parsing. All of the applications demonstrate the effectiveness and efficiency of the new representation. Experimental results show that our performance ranks among the top in all the applications.
References


