Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University
Doctoral Dissertation

# Multiple Kernel Learning for Computer Vision Applications

Yen-Yu Lin

Advisors: Tyng-Luh Liu, Ph.D.
Chiou-Shann Fuh, Ph.D.

October 25, 2010

# Multiple Kernel Learning for Computer Vision Applications

Yen-Yu Lin

October 25, 2010

# Abstract

In this thesis, we aim to improve the performances of computer vision applications when multiple feature representations of data are available. To that end, we develop several *multiple kernel learning* (*MKL*) techniques to carry out feature combination and facilitate the accomplishment of the underlying vision tasks. The thesis consists of three parts. The proposed approaches in the three parts are self-contained but highly correlative. The common theme among them is that kernel matrices will serve as the unified feature representations for data under different descriptors. Based upon this setting, feature fusion can be carried out in the domain of kernel matrices, while the applications will have a direct connection to kernel machines, the best off-the-shelf machine learning methodologies.

In the first part of the thesis, we aim to provide a *unified* and *compact* view of data with multiple feature representations. It is motivated by the fact that the feature representations of data under various descriptors are typically high dimensional and assume diverse forms. Finding a way to transform them into an Euclidean space of lower dimension hence generally facilitates the underlying vision tasks, such as recognition or clustering. To this end, the proposed approach (termed as MKL-DR) generalizes the framework of *multiple kernel learning* for *dimensionality reduction*, and distinguishes itself with the following three main contributions. First, our method provides the convenience of using diverse image descriptors to describe useful characteristics of various aspects about the underlying data. Second, it extends a broad set of existing dimensionality reduction techniques to consider multiple kernel learning, and consequently improves their effec-

tiveness. Third, by focusing on the techniques pertaining to dimensionality reduction, the formulation introduces a new class of applications with the multiple kernel learning framework to address not only the supervised learning problems but also the unsupervised and semisupervised ones.

In the second part, we propose the local ensemble kernels for adaptive feature fusion. As data often display large interclass and intraclass variations in complex vision tasks, the optimal feature combination for recognition/clustering may vary form data class to class. Learning a global feature fusion may not make the most use of multiple features. We hence suggest to learn a set of local classifiers, and each of them is derived for one training sample and optimized in a way to give good classification performances for data falling within the corresponding local area. Since local classifiers as many as the training samples are required to learn in this setting, it is important to keep the computational cost feasible even if datasets of large sizes are considered. To this end, we cast the multiple, independent training processes of local classifiers as a correlative multi-task learning problem, and design a new boosting algorithm to accomplish these tasks simultaneously and with higher efficiency.

In the last part of this thesis, we illustrate how to integrate supervised MKL techniques for feature combination into the unsupervised/semi-supervised clustering tasks. The intrinsic difficulty of this part results from the unsupervised nature of clustering: We have no labeled data to guide the searching of the optimal ensemble kernel over a given convex set of base kernels. Our key idea for handling the difficulty is to cast the tasks of supervised feature selection and unsupervised clustering procedure into a joint optimization problem. Besides, we describe a clustering approach with the emphasis on detecting coherent structures in a complex dataset, and consider *cluster-dependent feature selection*. To that end, we associate each cluster with a boosting classifier derived from multiple kernel learning, and apply the cluster-specific classifier to performing feature selection cross various descriptors to best separate data of the cluster from the rest. We integrate

the multiple, correlative training tasks of the cluster-specific classifiers into the clustering procedure. Through iteratively solving the joint optimization problem, the cluster structure is gradually revealed by these classifiers, while their discriminant power to capture similar data would be progressively improved owing to better data labeling.

# Contents

# List of Figures

# List of Tables

Chapter 1

# Introduction

Most researchers agree that the feature representations of data critically affect the performance in many computer vision applications. Although current vision research achieves significant progress in designing more robust features, the general conclusion is still that no single feature is sufficient for handling nowadays object/image recognition or clustering problems. The performance of the state-of-the-art system is still easily humbled by human visual system, which can comfortably perform such a task with high accuracy. One major obstacle hindering the advance in developing recognition techniques has to do with the large intraclass feature variations caused by issues such as ambiguities from clutter background, various poses, different lighting conditions, possible occlusion, and so on. Another difficulty is that current applications often involve a large number of categories. The two unfavorable issues often occur simultaneously in nowadays benchmark datasets of computer vision researches, such as Caltech-101 [33], Caltech-256 [45], Pascal VOC (Visual Object Classes) Challenge $2006 \sim 2010$ [28–32]. For an illustration, we show the high intraclass variations of VOC 2008 dataset in Figure 1.1 and the diverse object categories of Caltech-101 dataset in Figure 1.2.

In this thesis, we aim to improve the performances of a set of computer vision applications when multiple feature representations of data are available. Employing multiple

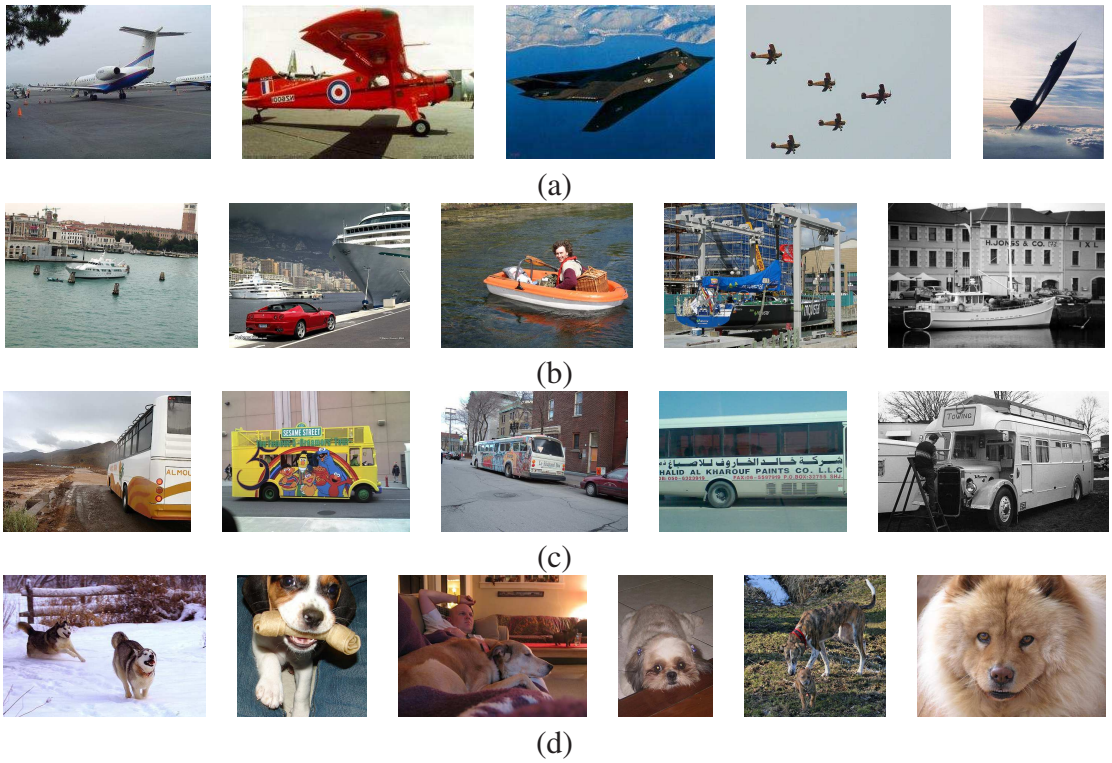



Figure 1.1: The high intraclass variations among data of categories (a) `aeroplane`, (b) `boat`, (c) `bus`, and (d) `dog` in VOC 2008 dataset.

features has been shown to be an effective way for performance boosting, since these features capture diverse characteristics of data to better discriminate among data samples of different classes while maintaining the desirable invariance to account for intraclass variations. This thesis will focus on how to combine heterogeneous features and how to complement them to result in the optimal combination for various vision applications. Specifically, we develop several *multiple kernel learning* (*MKL*) techniques to carry out feature fusion. The proposed MKL techniques are self-contained but highly correlative. The common theme among them is that kernel matrices are served as the unified feature representations for data under different descriptors. Based upon this setting, feature fusion will be carried out in the domain of kernel matrices, and the underlying applications will have a direct connection to kernel machines, e.g., SVMs (Support Vector Machines) [79, 97], GP (Gaussian processes) [75]), or KPCA (kernel principal component analysis)

Figure 1.2: The Caltech 101 dataset that consists of 101 object categories and one additional image class of background.

[80], the best off-the-shelf machine learning methodologies. We respectively specify the concept of multiple kernel learning, the overview of the proposed MKL techniques, and the thesis organization in the following.

## 1.1 Preliminary of Multiple Kernel Learning

MKL refers to the process of learning a kernel machine with multiple kernel functions or kernel matrices. Recent research efforts on MKL, e.g., [3, 43, 57, 74, 86] have shown that learning SVMs with multiple kernels not only increases the accuracy but also enhances the interpretability of the resulting classifiers. Most MKL researches aim to find an optimal way to linearly combine the given kernels. More precisely, suppose a set of base kernel functions $\{k_m\}_{m=1}^M$ (or base kernel matrices $\{K_m\}_{m=1}^M$) is given. An *ensemble kernel function* $k$ (or an *ensemble kernel matrix* $K$) is then defined by

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sum_{m=1}^{M} \beta_m k_m(\mathbf{x}_i, \mathbf{x}_j), \quad \beta_m \geq 0\,, \tag{1.1}$$

$$K = \sum_{m=1}^{M} \beta_m K_m, \quad \beta_m \geq 0\,. \tag{1.2}$$

Connecting the ensemble kernel with a kernel machine for binary-class data $\{(\mathbf{x}_i, y_i \in \pm 1)\}_{i=1}^N$, it results in the following formulation:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b \tag{1.3}$$

$$= \sum_{i=1}^{N} \alpha_i y_i \sum_{m=1}^{M} \beta_m k_m(\mathbf{x}_i, \mathbf{x}) + b. \tag{1.4}$$

Optimizing over both the coefficients $\{\alpha_i\}_{i=1}^N$ and $\{\beta_m\}_{m=1}^M$ is one particular form of the MKL problems.

Suppose data are described by $M$ kinds of features. When we record the pairwise relationships among data under each feature representation by a kernel matrix, determining a set of optimal ensemble coefficients $\{\beta_1, \beta_2, \ldots, \beta_M\}$ in MKL can be interpreted as finding appropriate weights for best fusing the $M$ feature representations.

Multiple kernel learning for feature fusion is indeed cornerstone of this thesis. We generalize MKL in three aspects, and apply them to facilitating a set of computer vision applications. The three aspects of generalization are described as follows:

- Diverse objective functions: Most MKL models are typically derived by minimizing the hinge loss. We provide a general framework in which MKL models can be obtained by using one of the objective functions of some existing dimensionality reduction methods, such as maximizing data variance in PCA or maximizing the ratio of between-class and within-class scatters in LDA. This generalization will expand the applicability of MKL in computer vision researches.

- Local ensemble kernels: Based upon the observation that the best feature combination for recognition often varies from object class to class, learning a *global* ensemble kernel for the whole dataset may be too general to make the optimal use of multiple features. In the situation, we show how to learn a set of *local* ensemble kernels for the given dataset, and how to perform recognition task by using the

resulting local classifiers, instead of the global one.

- Unsupervised/semi-supervised clustering tasks: Most MKL models are designed to address supervised classification/recognition tasks. In the aspect of generalization, we illustrate how to integrate supervised MKL techniques for feature combination into the unsupervised/semi-supervised clustering tasks.

## 1.2 Thesis Overview

This thesis consists of three parts. Each of them addresses one aspect of MKL generalizations mentioned in the previous section. We describe the three parts respectively in the following.

### 1.2.1 Multiple Kernel Learning for Dimensionality Reduction

Various descriptors have been developed to capture distinctive characteristics of data. The resulting data representations are typically high dimensional and assume diverse forms, such as vectors [67, 81], histograms [71, 99], high dimensional tensors [107, 111], bags of features [6, 63], pyramids [9, 58], and so on. The varieties of data representations complicate the process of feature combination. Thus finding a way to transform data under each descriptor into a unified representation generally facilitates the underlying learning tasks. In this proposal, we suggest to represent data under a specific descriptor by a kernel matrix, such that feature fusion can be carried out in the domain of kernel matrices through multiple kernel learning.

In this part, we propose an approach, called MKL-DR, that incorporates multiple kernel learning into the training process of dimensionality reduction. The main characteristics of the approach are: 1) the technique is flexible in the sense that it extends the existing dimensionality reduction methods to simultaneously tackle data under diverse de-

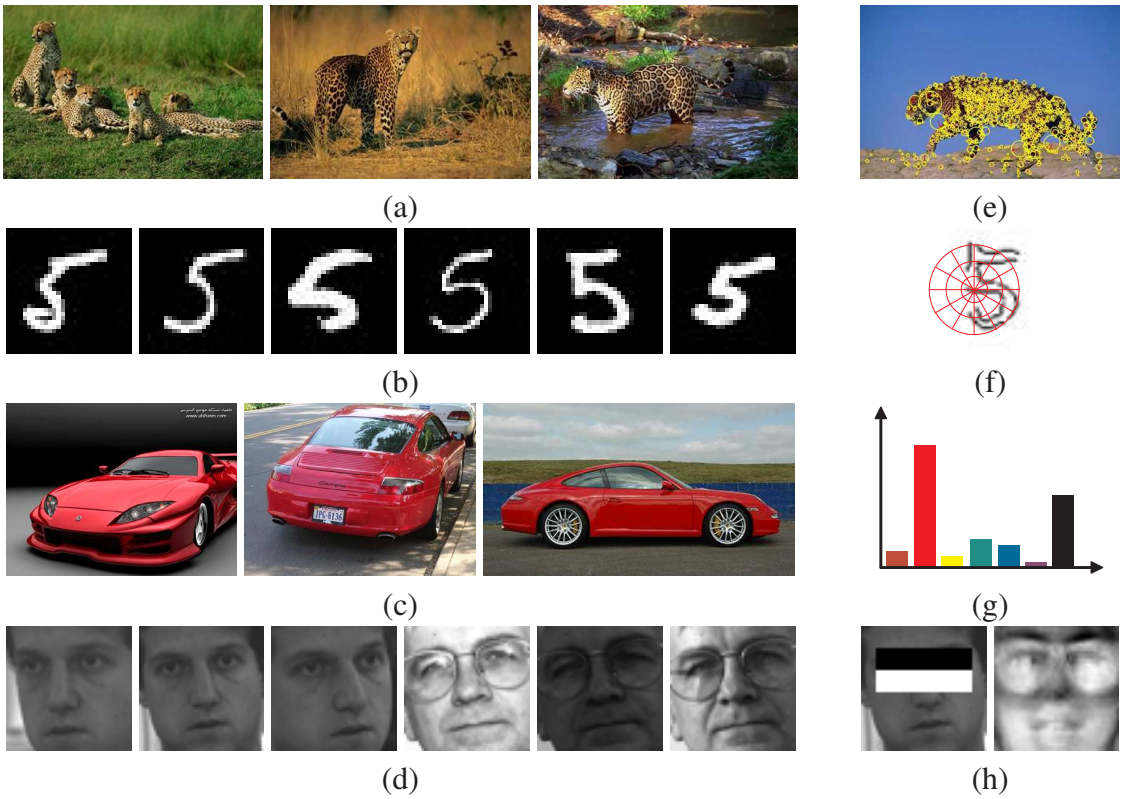



Figure 1.3: Images from four object categories: (a) `jaguar`, (b) `handwritten digit`, (c) `red car`, and (d) `human face`. Features that are respectively effective to the four categories are: (e) the DoG detector with SIFT descriptor [63], (f) shape context [4], (g) color histogram, and (h) a combination of the rectangle features [101] and eigenfaces [92].

scriptors; 2) the formulation is general as it is established upon *graph embedding* [108]. It follows that any dimensionality reduction methods explainable by graph embedding can be generalized to consider data in multiple feature representations; 3) via integrating with suitable dimensionality reduction methods, the multiple kernel learning framework can be extended to address not only the supervised learning problems but also the unsupervised and semisupervised ones. We demonstrate our approach on a broad range of computer vision applications, including supervised object categorization, unsupervised image clustering, and semi-supervised face recognition.

### 1.2.2 Multiple Kernel Learning for Local Learning

We consider the learned models by MKL-DR as *global*. The term, global, means that these models will be applied to all the testing data in prediction making. However, we think these global models may be too general to make optimal use of multiple descriptors. To have a better idea about the issue, images of four object categories are shown in Figure 1.3. As we could see, the keypoint-based descriptors, e.g. [63], well characterize the salient and distinctive skin in jaguars, despite their high intracalss variations in the aspects of poses and scales. Similarly, shape-based and color-based descriptors are effective for images in `handwritten digit` and `red car` categories respectively. As for the `human face` category, it may require some proper combination of several visual features, such as rectangle features [101] and eigenfaces [92], to yield a good representation. The example indicates the fact that the goodness of a descriptor often depends on what kinds of data it describes. It results in that the optimal feature or feature combination for recognition vary from category to category, or even from sample to sample if large intraclass variations are involved.

To consider the observation for a better utilization of multiple features, we relax the *responsibilities* of the learned models. That is, each model is not designed to give good performances for the whole training set, but for data falling within a local area of the feature space, such that *local learning* [2, 37] (locally adaptive feature selection in our case) can be carried out. Specifically, for each training sample, we learn a distinct ensemble kernel constructed in a way to best interpret the relationships among data within the corresponding neighborhood. We achieve this effect by aligning each ensemble kernel with a locally adapted target kernel, followed by smoothing out the discrepancies among kernels of nearby data. The entire training procedure, including localization and regularization of the ensemble kernel machines, could be reformulated as an energy minimization problem on the model of *Markov Random Fields* (MRFs).

Although object recognition methods based on local models can reasonably resolve the difficulties caused by the large variations in images from the same category, two unfavorable drawbacks may occur in local learning. First, each local model is learned based on a relatively small group of training data, the effect of overfitting is more likely to happen. Second, independently learning local models as many as training data is less efficient. The situations become worse in current recognition tasks, because databases that consist of thousands of data are used. Thus the applicability of local learning is restricted by the high risk of overfitting and the heavy computational cost.

To address these two unpleasant issues, we cast the multiple, independent training processes of local models as a correlative *multi-task learning* problem, and design a new boosting algorithm to accomplish it. As is pointed out from the literature of multi-task learning, e.g., [1, 13, 91], investigating related tasks *jointly* in most cases can achieve a considerable performance improvement than *independently*, since the extra knowledge from other tasks may convey useful information to the completion of the underlying task. On the other hand, as these local models in our case have overlapping training sets, learning them jointly generally can reduce information redundancy and lead to a substantial speed-up.

### 1.2.3 Multiple Kernel Learning for Unsupervised Clustering

In this part, we illustrate how to use supervised MKL techniques for feature combination in the unsupervised (or semi-supervised) clustering tasks. Besides, we describe a clustering approach with the emphasis on detecting coherent structures in a complex dataset, and show its effectiveness for object/face data clustering. By complex data, we mean the attribute variations among the data are too extensive such that clustering based on a single feature representation/descriptor is insufficient to faithfully divide the data into meaningful groups. The proposed method thus assumes the data are represented with various

feature representations, and aims to uncover the underlying cluster structure. To that end, we associate each cluster with a boosting classifier derived from multiple kernel learning, and apply the cluster-specific classifier to performing feature selection cross various descriptors to best separate data of the cluster from the rest. Specifically, we integrate the multiple, correlative training tasks of the cluster-specific classifiers into the clustering procedure, and cast them as a joint constrained optimization problem. Through the optimization iterations, the cluster structure is gradually revealed by these classifiers, while their discriminant power to capture similar data would be progressively improved owing to better data labeling.

## 1.3 Thesis Organization

The rest of this proposal is organized as follows. In Chapter 2, the MKL-DR framework is described, where multiple kernel learning is integrated into the learning process of dimensionality reduction methods. In Chapter 3, we carry out adaptive feature selection via deriving local ensemble kernels, and illustrate that the training procedure of sample-specific local classifiers can be completed by solving an energy minimization problem. In Chapter 4, we formulate the multiple, independent training processes of local models as a correlative multi-task learning problem, such that proper regularization and redundant elimination for local learning can be realized. In Chapter 5, the supervised MKL for feature fusion is integrated into the unsupervised clustering procedure. Finally, we make a conclusion and discuss the future work in Chapter 6.

Chapter 2

# Multiple Kernel Learning for Dimensionality Reduction

The fact that most computer vision problems deal with high dimensional data has made dimensionality reduction an inherent part of the current research. Besides having the potential for a more efficient approach, working with a new space of lower dimension often can gain the advantage of better analyzing the intrinsic structures in the data for various applications. For example, dimensionality reduction can be performed to compress data for a compact representation [50, 111], to visualize high dimensional data [77, 88], to exclude unfavorable data variations [15], or to improve the classification power of the nearest neighbor rule [16, 108].

Despite the great applicability, existing dimensionality reduction methods often suffer from two main restrictions. First, many of them, especially the linear ones, require data to be represented in the form of feature vectors. The limitation may eventually reduce the effectiveness of the overall algorithms when the data of interest could be more precisely characterized in other forms, e.g., bag-of-features [6, 63], matrices or high order tensors [108], [110]. Second, there seems to lack a systematic way of integrating multiple image features for dimensionality reduction. When addressing applications that no single de-

scriptor can appropriately depict the whole dataset, this shortcoming becomes even more evident. Alas, it is usually the case for addressing nowadays vision applications, such as the recognition task in *Caltech-101* dataset [33], or the classification and detection tasks in *Pascal VOC challenge* [29]. On the other hand, the advantage of using multiple features has been indeed consistently pointed out in a number of recent research efforts, e.g., [12, 41, 60, 98, 100].

Aiming to overcome the above-mentioned restrictions, we introduce a framework called *MKL-DR* that incorporates *multiple kernel learning* (MKL) into the training process of *dimensionality reduction* (DR) methods. It works with multiple *base kernels*, each of which is created based on a specific kind of data descriptor, and fuses the descriptors in the domain of kernel matrices. We will illustrate the formulation of MKL-DR with *graph embedding* [108], which provides a unified view for a large family of DR methods. Any DR techniques expressible by graph embedding can therefore be generalized by MKL-DR to boost their power by simultaneously taking account of data characteristics captured in different descriptors. It follows that the proposed approach can extend the MKL framework to address, as the corresponding DR methods would do, not only the *supervised* learning problems but also the *unsupervised* and *semisupervised* ones.

## 2.1 Literature Review

Since the relevant literature is quite extensive, our survey instead emphasizes the key concepts crucial to the establishment of the proposed framework.

### 2.1.1 Dimensionality Reduction

Techniques to perform dimensionality reduction for high dimensional data can vary considerably from each other due to, e.g., different assumptions about the data distribution or the availability of the data labeling. We categorize them as follows.

**Unsupervised DR:** *Principal component analysis* (PCA) [50] is the most well-known one that finds a linear mapping by maximizing the projected variances. For nonlinear DR techniques, *isometric feature mapping* (Isomap) [88] and *locally linear embedding* (LLE) [77] both exploit the manifold assumption to yield the embeddings. And, to resolve the out-of-sample problem in Isomap and LLE, *locality preserving projections* (LPP) [47] are proposed to uncover the data manifold by a linear relaxation.

**Supervised DR:** *Linear discriminant analysis* (LDA) assumes that the data of each class have a Gaussian distribution, and derives a projection from simultaneously maximizing the between-class scatter and minimizing the within-class scatter. Alternatively, *marginal Fisher analysis* (MFA) [108] and *local discriminant embedding* (LDE) [16] adopt the assumption that the data of each class spread as a submanifold, and seek a discriminant embedding over these submanifolds.

**Semisupervised DR:** If the observed data are partially labeled, dimensionality reduction can be performed by carrying out discriminant analysis over the labeled ones while preserving the intrinsic geometric structures of the remaining. Such techniques are useful, say, for vision applications where user interactions are involved, e.g., *semi-supervised discriminant analysis* (SDA) [11] for content-based image retrieval with relevance feedback.

**Kernelization:** It is possible to *kernelize* a certain type of linear DR techniques into nonlinear ones. As shown in [11], [16], [47], [65], [80], [108], the kernelized versions generally can achieve significant improvements. In addition, kernelization provides a convenient way for DR methods to handle data not in vector form by specifying an associated kernel, e.g., the *pyramid matching kernel* [44] for data in the form of bag-of-features, or the *dissimilarity kernel* [72] based on the pairwise distances.

### 2.1.2 Graph Embedding

A number of dimensionality reduction methods focus on modeling the pairwise relationships among data, and utilize graph-based structures. In particular, the framework of graph embedding [108] provides a unified formulation for a broad set of such DR algorithms. Let $\Omega = \{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^N$ be the dataset. A DR scheme accounted for by graph embedding involves a complete graph $G$ whose vertices are over $\Omega$. A corresponding affinity matrix $W = [w_{ij}] \in \mathbb{R}^{N\times N}$ is used to record the edge weights that characterize the similarity relationships between pairs of training samples. Then the optimal linear embedding $\mathbf{v}^* \in \mathbb{R}^d$ can be obtained by solving

$$\mathbf{v}^* = \underset{\substack{\mathbf{v}^\top XDX^\top\mathbf{v}=1, \text{ or} \\ \mathbf{v}^\top XL'X^\top\mathbf{v}=1}}{\arg\min} \mathbf{v}^\top XLX^\top\mathbf{v}, \tag{2.1}$$

where $X = [\mathbf{x}_1\ \mathbf{x}_2\ \cdots\ \mathbf{x}_N]$ is the data matrix, and $L = \mathrm{diag}(W \cdot \mathbf{1}) - W$ is the graph Laplacian of $G$. Depending on the property of a problem, one of the two constraints in (2.1) will be used in the optimization. If the first constraint is chosen, a diagonal matrix $D = [d_{ij}] \in \mathbb{R}^{N\times N}$ is included for scale normalization. Otherwise another complete graph $G'$ over $\Omega$ is required for the second constraint, where $L'$ and $W' = [w'_{ij}] \in \mathbb{R}^{N\times N}$ are respectively the graph Laplacian and affinity matrix of $G'$. The optimization problem (2.1) has an intuitive interpretation: $\mathbf{v}^\top X = [\mathbf{v}^\top\mathbf{x}_1\ \cdots\ \mathbf{v}^\top\mathbf{x}_N]$ represents the projected data; graph Laplacian $L$ (or $L'$) is to explore the *pairwise distances* of the projected data, while diagonal matrix $D$ is to weightedly combine their *distances to the origin*. More precisely, the meaning of (2.1) can be better understood with the following equivalent

problem:

$$\min_{\mathbf{v}} \sum_{i,j=1}^{N} ||\mathbf{v}^\top \mathbf{x}_i - \mathbf{v}^\top \mathbf{x}_j||^2 w_{ij} \tag{2.2}$$

$$\text{subject to} \sum_{i=1}^{N} ||\mathbf{v}^\top \mathbf{x}_i||^2 d_{ii} = 1, \text{ or} \tag{2.3}$$

$$\sum_{i,j=1}^{N} ||\mathbf{v}^\top \mathbf{x}_i - \mathbf{v}^\top \mathbf{x}_j||^2 w'_{ij} = 1. \tag{2.4}$$

The constrained optimization problem (2.2) implies that only distances to the origin or pairwise distances of projected data (in the form of $\mathbf{v}^\top \mathbf{x}$) are modeled by the framework. By specifying $W$ and $D$ (or $W$ and $W'$), Yan et al. [108] show that a set of dimensionality reduction methods, such as PCA [50], LPP [47], LDA, and MFA [108] can be expressed by (2.1). Clearly, LDE [16] and SDA [11] are also in the class of graph embedding.

### 2.1.3 Multiple Kernel Learning

MKL refers to the process of learning a kernel machine with multiple kernel functions or kernel matrices. Recent research efforts on MKL, e.g., [3, 43, 57, 74, 86] have shown that learning SVMs with multiple kernels not only increases the accuracy but also enhances the interpretability of the resulting classifiers. Our MKL formulation is to find an optimal way to linearly combine the given kernels. Suppose we have a set of base kernel functions $\{k_m\}_{m=1}^M$ (or base kernel matrices $\{K_m\}_{m=1}^M$). An *ensemble kernel function* $k$ (or an *ensemble kernel matrix* $K$) is then defined by

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sum_{m=1}^{M} \beta_m k_m(\mathbf{x}_i, \mathbf{x}_j), \quad \beta_m \geq 0\,, \tag{2.5}$$

$$K = \sum_{m=1}^{M} \beta_m K_m, \quad \beta_m \geq 0\,. \tag{2.6}$$

Consequently, an often-used MKL model from binary-class data $\{(\mathbf{x}_i, y_i \in \pm 1)\}_{i=1}^N$ is

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b \tag{2.7}$$

$$= \sum_{i=1}^{N} \alpha_i y_i \sum_{m=1}^{M} \beta_m k_m(\mathbf{x}_i, \mathbf{x}) + b. \tag{2.8}$$

Optimizing over both the coefficients $\{\alpha_i\}_{i=1}^N$ and $\{\beta_m\}_{m=1}^M$ is one particular form of the MKL problems. Our approach utilizes such an MKL optimization to yield more flexible dimensionality reduction schemes for data in different feature representations.

### 2.1.4 Dimensionality Reduction with Multiple Kernels

Our approach is related to the work of Kim et al. [52], where learning an optimal kernel over a given convex set of kernels is coupled with *kernel Fisher discriminant analysis* (KFDA) for binary-class data. Motivated by their idea of learning an optimal kernel for improving the KFDA performance, we instead consider establishing a general framework of dimensionality reduction for data in various feature representations via multiple kernel learning [61]. As we will show later, MKL-DR can be used to conveniently deal with image data depicted by different descriptors, and effectively tackle not only supervised but also semisupervised and unsupervised learning tasks. To the best of our knowledge, such a generalization of multiple kernel learning is novel.

## 2.2 The MKL-DR Framework

We first discuss the construction of base kernels from multiple descriptors, and then explain how to integrate them for dimensionality reduction. Finally, we present an optimization procedure to complete the framework.

### 2.2.1 Kernel as A Unified Feature Representation

Consider again a dataset $\Omega$ of $N$ samples, and $M$ kinds of descriptors to characterize each sample. Let $\Omega = \{\mathbf{x}_i\}_{i=1}^N$, $\mathbf{x}_i = \{\mathbf{x}_{i,m} \in \mathcal{X}_m\}_{m=1}^M$, and $d_m : \mathcal{X}_m \times \mathcal{X}_m \rightarrow 0 \cup \mathbb{R}^+$ be the distance function for data representation under the $m$th descriptor. In general, the domains resulting from distinct descriptors, e.g., feature vectors, histograms, or bags of features, are different. To eliminate such variances in representation, we express data under each descriptor as a kernel matrix. There are several ways to accomplish this goal, such as using the RBF kernel for data in the form of vector, or the pyramid match kernel [44] for data in the form of bag-of-features. We may also convert pairwise distances between data samples to a kernel matrix [98, 113]. By coupling each representation with its corresponding distance function, we obtain a set of $M$ *dissimilarity-based* kernel matrices $\{K_m\}_{m=1}^M$ where

$$K_m(i,j) = k_m(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-d_m^2(\mathbf{x}_{i,m}, \mathbf{x}_{j,m})}{\sigma_m^2}\right) \tag{2.9}$$

and $\sigma_m$ is a positive constant. Our use of dissimilarity-based kernels is convenient and advantageous in solving visual learning tasks, especially owing to that a number of well-designed descriptors and their associated distance function have been introduced over the years. However, $K_m$ in (2.9) is not always guaranteed to be positive semidefinite. Following [113], we resolve this issue by first computing the smallest eigenvalue of $K_m$. Then if it is negative, we add its absolute value to the diagonal of $K_m$. With (2.5), (2.6) and (2.9), determining a set of optimal ensemble coefficients $\{\beta_1, \beta_2, \ldots, \beta_M\}$ can now be interpreted as finding appropriate weights for best fusing the $M$ feature representations.

Note that in our formulation accessing the data is restricted to referencing the resulting $M$ kernels defined in (2.9). The main advantage of so doing is that it enables our approach to work with different descriptors and distance functions, without the need to explicitly handle the variations among the representations.

### 2.2.2 The MKL-DR Algorithm

Instead of designing a specific dimensionality reduction algorithm, we choose to describe MKL-DR upon graph embedding. This way we can emphasize the flexibility of the proposed approach: If a dimensionality reduction scheme is explained by graph embedding, then it will also be extendible by MKL-DR to handle data in multiple feature representations. Recall that there are two possible types of constraints in graph embedding. For the ease of presentation, we discuss how to develop MKL-DR subject to constraint (2.4). However, the derivation can be analogously applied when using constraint (2.3).

Kernelization in MKL-DR is accomplished in a similar way to that in kernel PCA [80], but with the key difference in using multiple kernels $\{K_m\}_{m=1}^M$. Suppose the ensemble kernel $K$ in MKL-DR is generated by linearly combining the base kernels $\{K_m\}_{m=1}^M$ as in (2.6). Let $\phi : \mathcal{X} \to \mathcal{F}$ denote the feature mapping induced by $K$. Via $\phi$, the training data can be implicitly mapped to a high dimensional Hilbert space, i.e.,

$$\mathbf{x}_i \mapsto \phi(\mathbf{x}_i), \text{ for } i = 1, 2, ..., N. \tag{2.10}$$

Since optimizing (2.1) or (2.2) can be reduced to solving the generalized eigenvalue problem $XLX^\top \mathbf{v} = \lambda XL'X^\top \mathbf{v}$, it implies that an optimal $\mathbf{v}$ lies in the span of training data, i.e.,

$$\mathbf{v} = \sum_{n=1}^{N} \alpha_n \phi(\mathbf{x}_n). \tag{2.11}$$

To show that the underlying algorithm can be reformulated in the form of inner product and accomplished in the new feature space $\mathcal{F}$, we observe that plugging into (2.2) each mapped sample $\phi(\mathbf{x}_i)$, projection $\mathbf{v}$ would appear exclusively in the form of $\mathbf{v}^\top \phi(\mathbf{x}_i)$. Hence, it suffices to show that in MKL-DR, $\mathbf{v}^\top \phi(\mathbf{x}_i)$ can be evaluated via the kernel

trick:

$$\mathbf{v}^\top \phi(\mathbf{x}_i) = \sum_{n=1}^{N}\sum_{m=1}^{M} \alpha_n \beta_m k_m(\mathbf{x}_n, \mathbf{x}_i) = \boldsymbol{\alpha}^\top \mathbb{K}^{(i)} \boldsymbol{\beta} \tag{2.12}$$

where

$$\boldsymbol{\alpha} = [\alpha_1 \ \cdots \ \alpha_N]^\top \in \mathbb{R}^N, \tag{2.13}$$

$$\boldsymbol{\beta} = [\beta_1 \ \cdots \ \beta_M]^\top \in \mathbb{R}^M, \tag{2.14}$$

$$\mathbb{K}^{(i)} = \begin{bmatrix} K_1(1,i) & \cdots & K_M(1,i) \\ \vdots & \ddots & \vdots \\ K_1(N,i) & \cdots & K_M(N,i) \end{bmatrix} \in \mathbb{R}^{N\times M}. \tag{2.15}$$

With (2.2) and (2.12), we define the constrained optimization problem for 1-*D* MKL-DR as follows:

$$\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \sum_{i,j=1}^{N} ||\boldsymbol{\alpha}^\top \mathbb{K}^{(i)} \boldsymbol{\beta} - \boldsymbol{\alpha}^\top \mathbb{K}^{(j)} \boldsymbol{\beta}||^2 w_{ij} \tag{2.16}$$

$$\text{subject to} \sum_{i,j=1}^{N} ||\boldsymbol{\alpha}^\top \mathbb{K}^{(i)} \boldsymbol{\beta} - \boldsymbol{\alpha}^\top \mathbb{K}^{(j)} \boldsymbol{\beta}||^2 w'_{ij} = 1, \tag{2.17}$$

$$\beta_m \geq 0, \ m = 1, 2, ..., M. \tag{2.18}$$

The additional constraints in (2.18) arise from the use of the ensemble kernel in (2.5) or (2.6), and are to ensure that the resulting kernel $K$ in MKL-DR is a non-negative combination of base kernels.

Observe from (2.12) that the one-dimensional projection $\mathbf{v}$ of MKL-DR is specified by a *sample coefficient vector* $\boldsymbol{\alpha}$ and a *kernel weight vector* $\boldsymbol{\beta}$. The two vectors respectively account for the relative importance among the samples and the base kernels in the construction of the projection. To generalize the formulation to uncover a multi-dimensional

projection, we consider a set of $P$ sample coefficient vectors, denoted by

$$A = [\boldsymbol{\alpha}_1 \ \boldsymbol{\alpha}_2 \ \cdots \ \boldsymbol{\alpha}_P]. \tag{2.19}$$

With $A$ and $\boldsymbol{\beta}$, each 1-D projection $\mathbf{v}_i$ is determined by a specific sample coefficient vector $\boldsymbol{\alpha}_i$ and the (shared) kernel weight vector $\boldsymbol{\beta}$. The resulting projection $V = [\mathbf{v}_1 \ \mathbf{v}_2 \ \cdots \ \mathbf{v}_P]$ will map samples to a $P$-dimensional Euclidean space. Analogous to the 1-D case, a projected sample $\mathbf{x}_i$ can be written as

$$V^\top \phi(\mathbf{x}_i) = A^\top \mathbb{K}^{(i)} \boldsymbol{\beta} \in \mathbb{R}^P. \tag{2.20}$$

The optimization problem (2.16) can now be extended to accommodate the multi-dimensional projection:

$$\min_{A,\boldsymbol{\beta}} \sum_{i,j=1}^{N} ||A^\top \mathbb{K}^{(i)} \boldsymbol{\beta} - A^\top \mathbb{K}^{(j)} \boldsymbol{\beta}||^2 w_{ij} \tag{2.21}$$

$$\text{subject to} \sum_{i,j=1}^{N} ||A^\top \mathbb{K}^{(i)} \boldsymbol{\beta} - A^\top \mathbb{K}^{(j)} \boldsymbol{\beta}||^2 w'_{ij} = 1, \tag{2.22}$$

$$\beta_m \geq 0, \ m = 1, 2, ..., M. \tag{2.23}$$

Before specifying the details of how to solve the constrained optimization problem (2.21) in the next section, we give an illustration of the four kinds of spaces related to MKL-DR and the connections among them in Figure 2.1. The four spaces in order are the input space of each feature representation, the reproducing kernel Hilbert space (RKHS) induced by each base kernel and the ensemble kernel, and the projected Euclidean space.

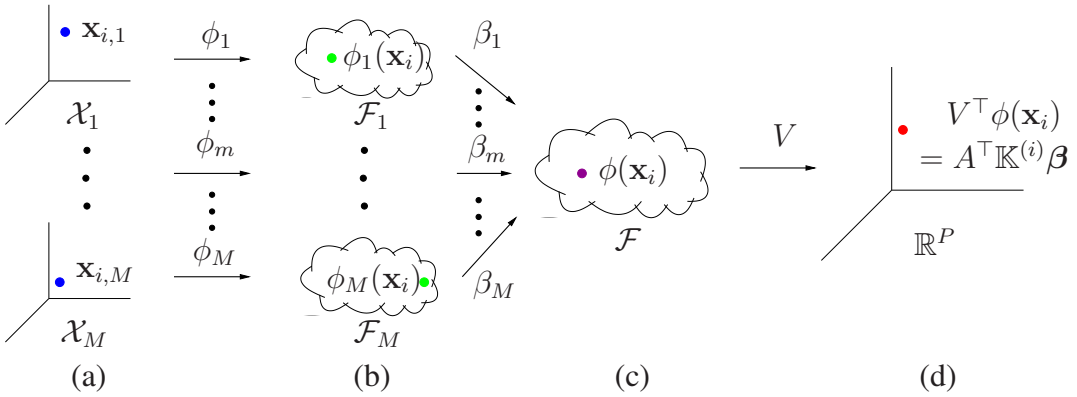



Figure 2.1: Four kinds of spaces in MKL-DR: (a) the input space of each feature representation, (b) the RKHS induced by each base kernel, (c) the RKHS by the ensemble kernel, and (d) the projected Euclidean space.

### 2.2.3 Optimization

Since direct optimization to (2.21) is difficult, we instead adopt an iterative, two-step strategy to alternately optimize $A$ and $\boldsymbol{\beta}$. At each iteration, one of $A$ and $\boldsymbol{\beta}$ is optimized while the other is fixed, and then the roles of $A$ and $\boldsymbol{\beta}$ are switched. Iterations are repeated until convergence or a maximum number of iterations is reached.

**On optimizing $A$:** By fixing $\boldsymbol{\beta}$ and using the property $||\mathbf{u}||^2 = \text{trace}(\mathbf{u}\mathbf{u}^\top)$ for a column vector $\mathbf{u}$, the optimization problem (2.21) is reduced to

$$\begin{aligned} \min_{A} \quad & \text{trace}(A^\top S_W^{\boldsymbol{\beta}} A) \\ \text{subject to} \quad & \text{trace}(A^\top S_{W'}^{\boldsymbol{\beta}} A) = 1 \end{aligned} \tag{2.24}$$

where

$$S_W^{\boldsymbol{\beta}} = \textstyle\sum_{i,j=1}^{N} w_{ij}(\mathbb{K}^{(i)} - \mathbb{K}^{(j)})\boldsymbol{\beta}\boldsymbol{\beta}^\top(\mathbb{K}^{(i)} - \mathbb{K}^{(j)})^\top, \tag{2.25}$$

$$S_{W'}^{\boldsymbol{\beta}} = \textstyle\sum_{i,j=1}^{N} w'_{ij}(\mathbb{K}^{(i)} - \mathbb{K}^{(j)})\boldsymbol{\beta}\boldsymbol{\beta}^\top(\mathbb{K}^{(i)} - \mathbb{K}^{(j)})^\top. \tag{2.26}$$

The optimization problem (2.24) is a *trace ratio* problem. Like [16] or what mentioned in [102], we obtain a closed form solution by transforming (2.24) into the corresponding *ratio trace* problem, i.e., $\min_A \text{trace}[(A^\top S_{W'}^{\boldsymbol{\beta}} A)^{-1}(A^\top S_W^{\boldsymbol{\beta}} A)]$. Consequently, the columns

**Algorithm 1** *The Training Procedure of MKL-DR*

**Input** : A DR method specified by two affinity matrices $W$ and $W'$ (cf. (2.2));
Data described by various visual features in form of base kernels $\{K_m\}_{m=1}^{M}$ (cf. (2.9));

**Output** : Sample coefficient vectors $A = [\boldsymbol{\alpha}_1\ \boldsymbol{\alpha}_2 \cdots \boldsymbol{\alpha}_P]$;
Kernel weight vector $\boldsymbol{\beta}$;

Make an initial guess for $A$ or $\boldsymbol{\beta}$;

**for** $t \leftarrow 1, 2, \ldots, T$ **do**

1. Compute $S_W^{\boldsymbol{\beta}}$ in (2.25) and $S_{W'}^{\boldsymbol{\beta}}$ in (2.26);
2. $A$ is optimized by solving the generalized eigenvalue problem (2.27);
3. Compute $S_W^A$ in (2.29) and $S_{W'}^A$ in (2.30);
4. $\boldsymbol{\beta}$ is optimized by solving optimization problem (2.31) via semidefinite programming;

**return** $A$ and $\boldsymbol{\beta}$;

of the optimal $A^* = [\boldsymbol{\alpha}_1\ \boldsymbol{\alpha}_2 \cdots \boldsymbol{\alpha}_P]$ are the eigenvectors corresponding to the first $P$ smallest eigenvalues in

$$S_W^{\boldsymbol{\beta}}\boldsymbol{\alpha} = \lambda S_{W'}^{\boldsymbol{\beta}}\boldsymbol{\alpha}. \tag{2.27}$$

**On optimizing $\boldsymbol{\beta}$:** By fixing $A$ and $||\mathbf{u}||^2 = \mathbf{u}^\top\mathbf{u}$, the optimization problem (2.21) becomes

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & \boldsymbol{\beta}^\top S_W^A \boldsymbol{\beta} \\ \text{subject to} \quad & \boldsymbol{\beta}^\top S_{W'}^A \boldsymbol{\beta} = 1 \text{ and } \boldsymbol{\beta} \geq \mathbf{0} \end{aligned} \tag{2.28}$$

where

$$S_W^A = \textstyle\sum_{i,j=1}^{N} w_{ij}(\mathbb{K}^{(i)} - \mathbb{K}^{(j)})^\top AA^\top(\mathbb{K}^{(i)} - \mathbb{K}^{(j)}), \tag{2.29}$$

$$S_{W'}^A = \textstyle\sum_{i,j=1}^{N} w'_{ij}(\mathbb{K}^{(i)} - \mathbb{K}^{(j)})^\top AA^\top(\mathbb{K}^{(i)} - \mathbb{K}^{(j)}). \tag{2.30}$$

The additional constraints $\boldsymbol{\beta} \geq \mathbf{0}$ cause that the optimization to (2.28) can no longer be formulated as a generalized eigenvalue problem. Indeed it now becomes a nonconvex *quadratically constrained quadratic programming* (QCQP) problem, and is known to be

hard to solve. We instead consider solving its convex relaxation by adding an auxiliary variable $B$ of size $M \times M$:

$$\min_{\boldsymbol{\beta},B} \quad \text{trace}(S_W^A B) \tag{2.31}$$

$$\text{subject to} \quad \text{trace}(S_{W'}^A B) = 1, \tag{2.32}$$

$$\mathbf{e}_m^\top \boldsymbol{\beta} \geq 0,\ m = 1, 2, ..., M, \tag{2.33}$$

$$\begin{bmatrix} 1 & \boldsymbol{\beta}^\top \\ \boldsymbol{\beta} & B \end{bmatrix} \succeq 0, \tag{2.34}$$

where $\mathbf{e}_m$ in (2.33) is a column vector whose elements are $0$ except that its $m$th element is $1$, and the constraint in (2.34) means that the square matrix is positive semidefinite. The optimization problem (2.31) is an *SDP relaxation* of the nonconvex QCQP problem (2.28), and can be efficiently solved by semidefinite programming (SDP). One can verify the equivalence between the two optimization problems (2.28) and (2.31) if we replace the constraint (2.34) with $B = \boldsymbol{\beta}\boldsymbol{\beta}^\top$. In view of that the constraint $B = \boldsymbol{\beta}\boldsymbol{\beta}^\top$ is nonconvex, it is relaxed to $B \succeq \boldsymbol{\beta}\boldsymbol{\beta}^\top$. Applying the Schur complement lemma, $B \succeq \boldsymbol{\beta}\boldsymbol{\beta}^\top$ can be equivalently expressed by the constraint in (2.34). (Refer to [96] for the details.) Pertaining to the computational complexity of MKL-DR, note that the numbers of constraints and variables in (2.31) are respectively linear and quadratic to $M$, the number of the adopted descriptors. In practice the value of $M$ is often small. ($M = 4 \sim 10$ in our experiments.) Thus like most of the other DR methods, the computational bottleneck of MKL-DR is still in solving the generalized eigenvalue problems, which is in $\mathcal{O}(N^3)$.

Listed in Algorithm 1, the procedure of MKL-DR requires an initial guess to either $A$ or $\boldsymbol{\beta}$ in the alternating optimization. We have tried two possibilities: 1) $\boldsymbol{\beta}$ is initialized by setting all of its elements as $1$ to equally weight base kernels; 2) $A$ is initialized by assuming $AA^\top = I$. In our empirical testing, the second initialization strategy gives more stable performances, and is thus adopted in the experiments. Pertaining to the con-

vergence of the optimization procedure, since SDP relaxation has been used, the values of the objective function are not guaranteed to monotonically decrease throughout the iterations. Still, the optimization procedures rapidly converge after only a few iterations in all our experiments.

### 2.2.4 Novel Sample Embedding

After accomplishing the training procedure of MKL-DR, we are ready to project a testing sample, say $\mathbf{z}$, into the learned space of lower dimension by

$$\mathbf{z} \mapsto A^\top \mathbb{K}^{(\mathbf{z})} \boldsymbol{\beta}, \text{ where} \tag{2.35}$$

$$\mathbb{K}^{(\mathbf{z})} \in \mathbb{R}^{N \times M} \text{ and } \mathbb{K}^{(\mathbf{z})}(n, m) = k_m(\mathbf{x}_n, \mathbf{z}). \tag{2.36}$$

Depending on the applications, some post-processing, such as the nearest neighbor rule for classification or $k$-means clustering for data grouping, is then applied to the projected sample(s) to complete the task. In the remaining of this paper, we specifically discuss three sets of experimental results to demonstrate the effectiveness of MKL-DR, including supervised learning for object categorization, unsupervised learning for image clustering, and semisupervised learning for face recognition.

## 2.3 Experimental Results I: Supervised Learning for Object Categorization

Applying MKL-DR to object categorization is appropriate as the complexity of the task often requires the use of multiple feature descriptors. And in our experiments the effectiveness of MKL-DR will be investigated through a supervised-learning formulation.

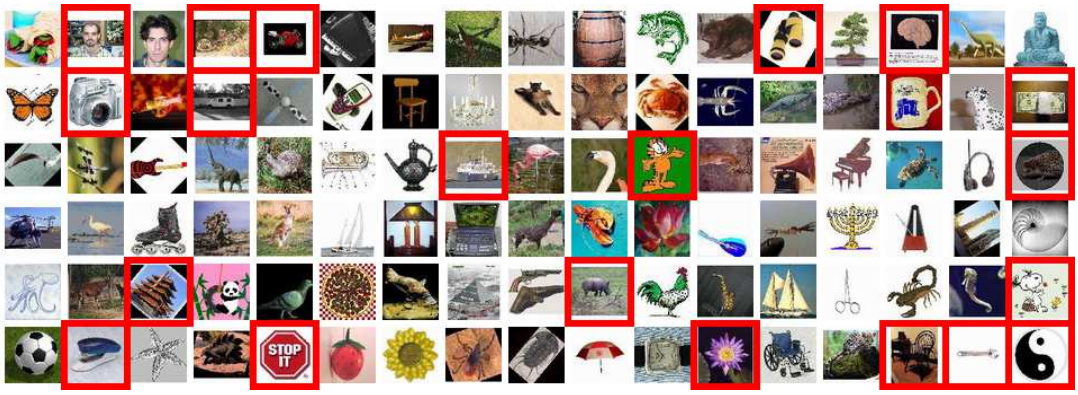



Figure 2.2: The Caltech-101 dataset. One example comes from each of the 102 categories. All the 102 categories are used in the experiments of supervised object recognition, while the 20 categories marked by the red bounding boxes are used in the following experiments of unsupervised image clustering.

### 2.3.1 Dataset

The Caltech-101 dataset [33], collected by Fei-Fei et al., is used in our experiments for object categorization. It consists of 101 object categories and one additional class of background images. The total number of categories is 102, and each category contains roughly 40 to 800 images. Although each target object often appears in the central region of an image, the large class number and substantial intraclass variations still make the dataset very challenging. Indeed the dataset provides a good test bed to demonstrate the advantage of using multiple image descriptors for complex recognition tasks. Note that as the images in Caltech-101 are not of the same size, we resize them to around 60,000 pixels, without changing their aspect ratio. Figure 2.2 shows an image example from each category of the dataset.

To implement Algorithm 1 for object recognition, we need to decide a set of descriptors for depicting the diverse objects and the underlying graph-based DR method to be generalized. Based on them, we can then derive a set of base kernels and a pair of affinity matrices respectively. The details are described as follows.

### 2.3.2 Image Descriptors and Base Kernels

Ten different image descriptors are considered and they respectively yield the following base kernels (denoted below in bold and in abbreviation):

- **GB-Dist**: For a given image, we randomly sample $400$ edge pixels, and apply *geometric blur* descriptor [5] to them. With these image features, we adopt the distance function, as is suggested in equation (2) of the work by Zhang et al. [113], to obtain the dissimilarity-based kernel.

- **GB**: The base kernel is constructed in the same way to that of GB-Dist, except the geometric distortion term is excluded in evaluating the distance.

- **SIFT-Dist**: The base kernel is analogously constructed as in GB-Dist, except now the *SIFT* descriptor [63] is used to extract features.

- **SIFT-SPM**: We apply the SIFT descriptor with three different scales to an evenly sampled grid of each image, and use $k$-means clustering to generate *visual words* from the resulting local features of all images. Then the base kernel is built by matching *spatial pyramids*, which is proposed in [58].

- **SS-Dist/SS-SPM**: The two base kernels are respectively constructed as in SIFT-Dist and SIFT-SPM, except that the SIFT descriptor is replaced with the *self-similarity* descriptor [82]. Note that for the latter descriptor, we set the size of each patch to $5 \times 5$, and the radius of the window to $40$.

- **C2-SWP/C2-ML**: Biologically inspired features are also adopted. Specifically, both the *C2* features derived by Serre et al. [81] and by Mutch and Lowe [67] have been considered. For each of the two kinds of C2 features, an RBF kernel is obtained.

- **PHOG**: We also adopt the *PHOG* descriptor [8] to capture image features, and limit the pyramid level up to 2. Together with the $\chi^2$ distance, it yields the resulting base kernel.

- **GIST**: The images are resized to $128 \times 128$ pixels prior to applying the *gist* descriptor [70]. Then an RBF kernel is established.

The parameters in the above descriptors and distance functions are tuned independently. Pairwise distance matrices yielded by the same descriptor but with different parameters are linearly combined in advance by using a greedy way. Namely, we separately and sequentially adjust the weight assigning to each distance matrix, such that the combined distance matrix can lead to better performance. The procedure is performed until no improvement can be achieved. Such a scheme is to ensure that the resulting kernels individually reach their best performances.

### 2.3.3 Dimensionality Reduction Methods

We investigate two supervised DR techniques, namely, linear discriminant analysis (LDA) and local discriminant embedding (LDE) [16], and show how MKL-DR can generalize them. Both LDA and LDE perform discriminant learning on a fully labeled dataset $\Omega = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, but make different assumptions about the data distribution: While in LDA data of each class are supposed to form a Gaussian, in LDE they are assumed to spread as a submanifold. Nevertheless, both techniques can be specified by a pair of affinity matrices to fit the formulation of graph embedding (2.2). For convenience, the resulting MKL dimensionality reduction schemes are respectively termed as *MKL-LDA* and *MKL-LDE*.

**Affinity matrices for LDA:** The two affinity matrices $W = [w_{ij}]$ and $W' = [w'_{ij}]$ are

defined as

$$w_{ij} = \begin{cases} 1/n_{y_i}, & \text{if } y_i = y_j, \\ 0, & \text{otherwise,} \end{cases} \tag{2.37}$$

$$w'_{ij} = \frac{1}{N}, \tag{2.38}$$

where $n_{y_i}$ is the number of data points that belong to class $y_i$. See [108] for the derivation.

**Affinity matrices for LDE:** In LDE, not only the data labels but also the neighborhood relationships are simultaneously considered, namely,

$$w_{ij} = \begin{cases} 1, & \text{if } y_i = y_j \wedge [i \in \mathcal{N}_k(j) \vee j \in \mathcal{N}_k(i)], \\ 0, & \text{otherwise,} \end{cases} \tag{2.39}$$

$$w'_{ij} = \begin{cases} 1, & \text{if } y_i \neq y_j \wedge [i \in \mathcal{N}_{k'}(j) \vee j \in \mathcal{N}_{k'}(i)], \\ 0, & \text{otherwise,} \end{cases} \tag{2.40}$$

where $i \in \mathcal{N}_k(j)$ means that sample $\mathbf{x}_i$ is one of the $k$ nearest neighbors of sample $\mathbf{x}_j$. The definitions of the affinity matrices are faithful to those in LDE [16]. For cases that there are multiple image descriptors, an affinity matrix is constructed for data under each descriptor. Because we have no idea about which affinity matrices are more important with respect to the given task in advance, thus we simply average the resulting affinity matrices from all the descriptors.

Table 2.1: Recognition rates of LDA-based classifiers on Caltech-101. [mean $\pm$ std %]

| method | kernel(s) | number of training data per class ($N_{train}$) | | | | |
|---|---|---|---|---|---|---|
| | | 5 | 10 | 15 | 20 | 25 |
| Kernel LDA (KFD) | GB-Dist | $44.0 \pm 1.1$ % | $55.2 \pm 0.9$ % | $60.6 \pm 1.1$ % | $65.1 \pm 1.1$ % | $69.7 \pm 1.6$ % |
| | GB | $40.7 \pm 1.2$ % | $51.1 \pm 0.8$ % | $56.2 \pm 0.6$ % | $62.0 \pm 1.4$ % | $66.0 \pm 1.1$ % |
| | SIFT-Dist | $36.6 \pm 1.1$ % | $46.7 \pm 0.5$ % | $53.8 \pm 0.9$ % | $57.8 \pm 1.1$ % | $63.1 \pm 1.9$ % |
| | SIFT-SPM | $34.9 \pm 0.6$ % | $44.3 \pm 1.0$ % | $50.3 \pm 0.6$ % | $55.1 \pm 1.2$ % | $60.0 \pm 2.4$ % |
| | SS-Dist | $35.3 \pm 1.0$ % | $44.9 \pm 0.8$ % | $51.1 \pm 1.0$ % | $56.0 \pm 1.6$ % | $61.5 \pm 2.3$ % |
| | SS-SPM | $37.0 \pm 0.7$ % | $47.8 \pm 0.6$ % | $55.0 \pm 0.9$ % | $59.1 \pm 0.9$ % | $64.0 \pm 2.0$ % |
| | C2-SWP | $18.3 \pm 0.9$ % | $24.9 \pm 0.6$ % | $29.6 \pm 0.7$ % | $34.3 \pm 1.1$ % | $37.7 \pm 0.8$ % |
| | C2-ML | $30.4 \pm 0.9$ % | $39.2 \pm 0.6$ % | $45.1 \pm 0.6$ % | $48.1 \pm 0.7$ % | $52.8 \pm 2.0$ % |
| | PHOG | $27.6 \pm 0.8$ % | $34.4 \pm 0.8$ % | $40.7 \pm 0.6$ % | $42.5 \pm 0.8$ % | $46.5 \pm 1.5$ % |
| | GIST | $33.2 \pm 0.8$ % | $42.1 \pm 0.9$ % | $47.0 \pm 0.7$ % | $51.5 \pm 1.0$ % | $55.6 \pm 1.6$ % |
| KFD-*Voting* | - | $54.4 \pm 0.7$ % | $65.7 \pm 0.8$ % | $69.8 \pm 0.7$ % | $73.9 \pm 1.1$ % | $76.8 \pm 1.6$ % |
| KFD-*Concatenate* | - | $55.4 \pm 1.2$ % | $65.6 \pm 0.9$ % | $71.7 \pm 0.8$ % | $75.3 \pm 0.8$ % | $78.2 \pm 1.3$ % |
| KFD-*AvgKernel* | - | $55.9 \pm 1.0$ % | $66.1 \pm 0.6$ % | $72.0 \pm 1.0$ % | $75.6 \pm 0.7$ % | $78.2 \pm 1.5$ % |
| KFD-*SAMME* | - | $56.4 \pm 1.1$ % | $67.4 \pm 0.9$ % | $72.3 \pm 0.6$ % | $75.7 \pm 1.0$ % | $77.7 \pm 2.1$ % |
| MKL-LDA | All | **58.4 $\pm$ 1.0 %** | **68.8 $\pm$ 0.9 %** | **74.5 $\pm$ 1.0 %** | **77.5 $\pm$ 1.0 %** | **79.8 $\pm$ 1.7 %** |

### 2.3.4 Quantitative Results

Like in [6, 98, 113], we randomly pick $30$ images from each of the $102$ categories, and split them into two disjoint subsets: One contains $N_{train}$ images per category, and the other consists of the rest. The two subsets are respectively used as the training and testing data. Via MKL-DR, the data are projected to the learned space, and the recognition task is accomplished there by enforcing the *nearest-neighbor* rule. To relieve the effect of sampling, the whole process of performance evaluation are redone $20$ times by using different random splits between the training and testing subsets. The recognition rates are measured in the cases where the value of $N_{train}$ is respectively set as $5$, $10$, $15$, $20$, and $25$.

Coupling the ten base kernels with the affinity matrices of LDA and LDE, we can respectively derive MKL-LDA and MKL-LDE using Algorithm 1. Their effectiveness is investigated by comparing with KFD (kernel Fisher discriminant) [65] and KLDE (kernel LDE) [16]. Since KFD considers only one base kernel at a time, we implement four strategies to take account of the information from the ten resulting KFD classifiers, including 1) KFD-*Voting*: It is constructed based on the voting result of the ten KFD classifiers. If there is any ambiguity in the voting result, the next nearest neighbor in each KFD clas-

Table 2.2: Recognition rates of LDE-based classifiers on Caltech-101. [mean $\pm$ std %]

| method | kernel(s) | number of training data per class ($N_{train}$) | | | | |
|---|---|---|---|---|---|---|
| | | 5 | 10 | 15 | 20 | 25 |
| Kernel LDE (KLDE) | GB-Dist | $44.5 \pm 1.1$ % | $54.8 \pm 0.7$ % | $60.7 \pm 1.1$ % | $66.2 \pm 1.4$ % | $69.7 \pm 1.3$ % |
| | GB | $41.0 \pm 1.3$ % | $51.3 \pm 1.1$ % | $55.9 \pm 0.5$ % | $62.2 \pm 1.2$ % | $66.3 \pm 1.3$ % |
| | SIFT-Dist | $37.0 \pm 1.1$ % | $46.6 \pm 0.7$ % | $53.4 \pm 0.7$ % | $58.0 \pm 1.4$ % | $63.7 \pm 2.1$ % |
| | SIFT-SPM | $35.3 \pm 0.8$ % | $44.4 \pm 0.9$ % | $50.2 \pm 0.7$ % | $55.0 \pm 1.0$ % | $61.3 \pm 2.4$ % |
| | SS-Dist | $35.1 \pm 1.0$ % | $44.1 \pm 0.8$ % | $50.6 \pm 1.0$ % | $55.2 \pm 1.1$ % | $61.2 \pm 1.9$ % |
| | SS-SPM | $37.1 \pm 0.7$ % | $47.1 \pm 0.4$ % | $55.4 \pm 1.3$ % | $59.2 \pm 1.0$ % | $64.1 \pm 1.9$ % |
| | C2-SWP | $18.7 \pm 0.9$ % | $24.9 \pm 0.5$ % | $29.7 \pm 0.7$ % | $34.3 \pm 1.1$ % | $38.2 \pm 1.1$ % |
| | C2-ML | $30.7 \pm 0.9$ % | $39.5 \pm 0.8$ % | $45.9 \pm 0.8$ % | $48.3 \pm 0.6$ % | $54.2 \pm 2.2$ % |
| | PHOG | $28.0 \pm 1.0$ % | $34.8 \pm 0.9$ % | $39.5 \pm 0.8$ % | $42.3 \pm 1.0$ % | $46.1 \pm 1.0$ % |
| | GIST | $33.4 \pm 0.8$ % | $41.8 \pm 0.7$ % | $47.0 \pm 0.6$ % | $51.5 \pm 0.9$ % | $56.7 \pm 1.5$ % |
| KLDE-*Voting* | - | $53.9 \pm 1.2$ % | $64.3 \pm 0.8$ % | $70.3 \pm 1.0$ % | $74.3 \pm 0.7$ % | $76.6 \pm 1.6$ % |
| KLDE-*Concatenate* | - | $55.4 \pm 0.9$ % | $65.7 \pm 0.7$ % | $71.4 \pm 0.9$ % | $75.5 \pm 0.9$ % | $78.5 \pm 1.4$ % |
| KLDE-*AvgKernel* | - | $55.8 \pm 0.7$ % | $66.3 \pm 0.7$ % | $71.8 \pm 0.9$ % | $75.7 \pm 0.8$ % | $78.4 \pm 1.5$ % |
| KLDE-*SAMME* | - | $56.1 \pm 1.3$ % | $66.8 \pm 0.7$ % | $72.6 \pm 1.0$ % | $75.7 \pm 0.9$ % | $77.8 \pm 1.3$ % |
| MKL-LDE | All | **59.2 $\pm$ 1.5 %** | **68.9 $\pm$ 0.8 %** | **74.9 $\pm$ 1.1%** | **77.2 $\pm$ 0.9%** | **79.2 $\pm$ 1.6%** |

sifier will be considered, and the process is continued until a decision on the class label can be made; 2) KFD-*Concatenate*: We concatenate the separately-learned feature vectors of each sample. Note that each feature vector is normalized in advance by dividing the standard deviation of the pairwise distances among the projected training data; 3) KFD-*AvgKernel*: KFD is re-applied to the average kernel of the ten base ones; 4) KFD-*SAMME*: By viewing each KFD classifier as a multi-class weak learner, we boost them by *SAMME* [115], which is a multi-class generalization of AdaBoost. Analogously, the four strategies can be applied to the KLDE classifiers.

The values of hyperparameters, $\{\sigma_m\}$ in (2.9), are critical to the performance of MKL-DR. However, it is almost infeasible to tune these hyperparameters exhaustively. Thus we adopt the following procedure to adjust them: It can be observed that the larger $\sigma_m$ is, the more evenly the element values in $K_m$ distribute. Fixing some values of, say, $s$ and $t$, we adjust the value of $\sigma_m$ by *binary search* such that the largest $s$ elements in $K_m$ will take up $t\%$ of the sum of all elements. Once $s$ and $t$ are fixed, the values of $\{\sigma_m\}$ are also determined. We set $s$ as a constant and exhaustively seek the optimal $t$. The resulting $\{\sigma_m\}$ will serve as the initialization to the greedy search procedure described in Section 2.3.2.

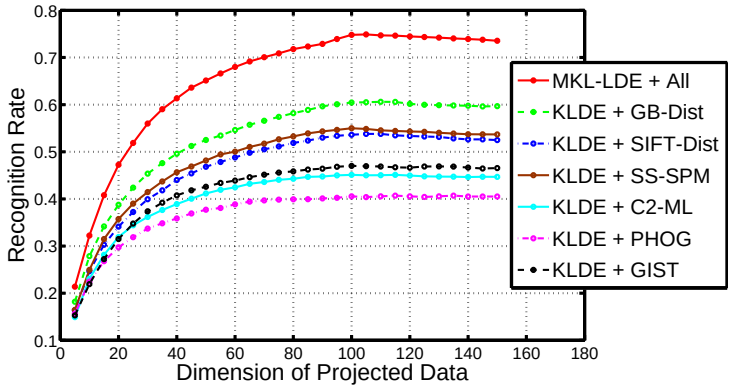



Figure 2.3: Recognition rates with different dimensions of the projected data when $N_{train} = 15$.

Table 2.1 summarizes the mean recognition rates and the standard deviations of KFD classifiers and MKL-LDA classifiers when different amounts of training data are available. By focusing on $N_{train} = 15$, we observe that MKL-LDA achieves a significant performance gain of $13.9\%$ $(= 74.5\% - 60.6\%)$ over the best recognition rate by the ten KFD classifiers. It suggests that the ten base kernels tend to complement each other, and our approach can effectively fuse them to result in a more powerful classifier. On the other hand, while KFD-*Voting*, KFD-*Concatenate* and KFD-*SAMME* try to combine the *separately* trained KFD classifiers, MKL-LDA *jointly* integrates the ten kernels into the learning process. The quantitative results show that MKL-LDA can make the most of fusing various feature descriptors, and improves the recognition rates from $69.8\%$, $71.7\%$ and $72.3\%$ to $74.5\%$. Besides, MKL-LDA outperforms KFD-*AvgKernel*, and it points out that the learned kernel weight vector $\boldsymbol{\beta}$ by MKL-DR leads to a more effective ensemble kernel than the average kernel. Similar improvements can be observed in cases where different numbers of training data per class are used.

The quantitative results of KLDE and MKL-LDE are reported in Table 2.2. Like MKL-LDA, MKL-LDE achieves similar degrees of improvements over the KLDE classifiers and their combinations. To discuss the dimensionality effect of MKL-DR in learning the unified feature space, we respectively evaluate the recognition rates of MKL-LDE and

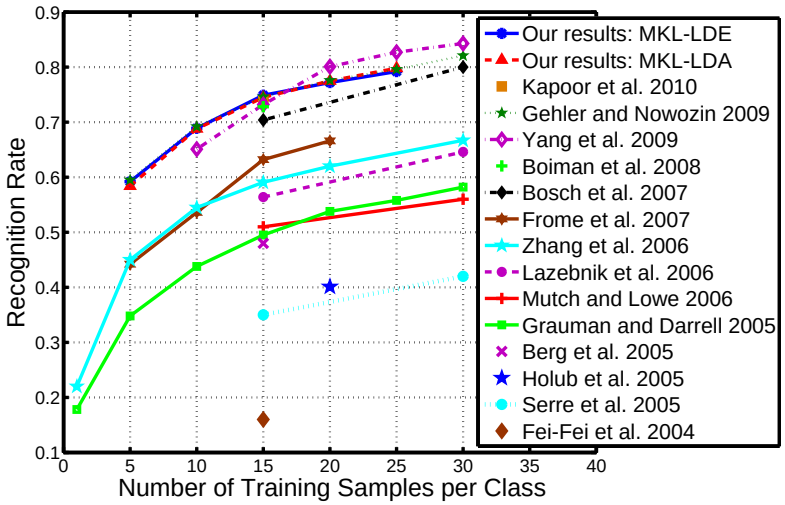
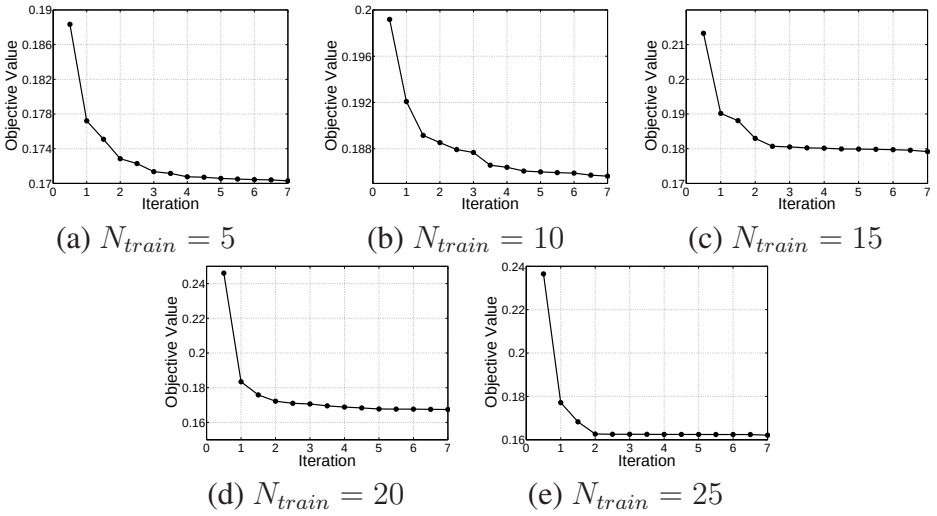



Figure 2.4: Recognition rates of several published systems on Caltech-101 with different amounts of training data.

KLDE over a range of embedding dimensions, i.e., $P$ in (2.20). The results are shown in Figure 2.3. It can be observed that MKL-LDE can achieve similar degrees of accuracy with fewer dimensions. In addition, both the recognition rates of MKL-LDE and KLDE converge around the dimensions of $90 \sim 110$.

When the number of training data per class from Caltech-101 is set as $15$, the recognition rate of $74.5\%$ by MKL-LDA and $74.9\%$ by MKL-LDE are favorably comparable to those by most existing approaches. In [6], Berg et al. report a recognition rate of $48\%$ based on deformable shape matching. Using the pyramid matching kernel over data in the bag-of-features representation, the recognition rate by Grauman and Darrell [44] is $50\%$. Subsequently, Lazebnik et al. [58] improves it to $56.4\%$ by considering the spatial pyramid matching kernel. In [113], Zhang et al. combine the geometric blur descriptor and spatial information to achieve $59.05\%$. Our related work [60] that performs adaptive feature fusing via locally combining kernel matrices has a recognition rate $59.8\%$, while merging twelve kernel matrices from the *support kernel machines* (SKMs) [3] by Kumar and Sminchisescu [55] yields $57.3\%$. Frome et al. [39] propose to learn a local distance for each training sample, and derive $60.3\%$. To tackle the weakly labeled attribute




(a) $N_{train} = 5$ (b) $N_{train} = 10$ (c) $N_{train} = 15$

(d) $N_{train} = 20$ (e) $N_{train} = 25$

Figure 2.5: The values of the objective function of MKL-LDA through the iterative optimization procedure when $N_{train}$ are (a) $5$, (b) $10$, (c) $15$, (d) $20$, and (e) $25$ respectively.

of Caltech-101, Bosch et al. [8] suggest to find the *ROIs* of images before performing recognition, and report an accuracy rate of $70.4\%$. By investigating the category features from *subcategories*, Todorovic and Ahuja [90] get a recognition rate around $73\%$. Christoudias et al. [18] learn the localized Gaussian processes with multiple kernels, and obtain $73\%$. Gehler and Nowozin [42] carry out multiple kernel learning in boosting ways, and achieve $74.6\%$. In Figure 2.4, we summarize the recognition rates of our approach and several published techniques, including [6–8, 33, 40, 42, 44, 49, 51, 58, 67, 81, 109, 113], under different sizes of training data.

We complete the section by discussing the convergence property of our algorithm. Take, for example, learning MKL-LDA with different sizes of training data per class. The values of the objective function (2.21) through the iterative optimization are respectively shown in Figs. 2.5a–2.5e. In each iteration, two such values are plotted to account for updating either $A$ or $\boldsymbol{\beta}$. It can be observed that all the optimization procedures rapidly converge after a few iterations. Also, increasing the size of training data tends to speed up the convergence, which is reasonable since sufficient information generally facilitates solving an optimization task.

## 2.4 Experimental Results II: Unsupervised Learning for Image Clustering

To explain the link between MKL-DR and unsupervised learning, we investigate the problem of image clustering. In this case, MKL-DR can be viewed as a *preprocessing* tool to enrich the capacity of an existing clustering technique. There are two main advantages for so doing. First, since MKL-DR can learn a unified space for image data in multiple representations, it enables the underlying clustering algorithm to simultaneously consider characteristics captured by distinct descriptors. Second, a majority of clustering algorithms, e.g., $k$-means, are designed to work only in the Euclidean space. With MKL-DR, no matter what the original spaces the data reside in, they all can be projected to the learned Euclidean space, and consequently our formulation can extend the applicability of such a clustering method.

### 2.4.1 Dataset

We follow the setting in [27], where *affinity propagation* [36] is used for unsupervised image categorization, and select the same twenty categories from Caltech-101 for the image clustering experiments. Examples from the twenty image categories are shown in Figure 2.2, and each is marked with a bold red bounding box. Due to the category-wise differences in the number of images, we randomly select $30$ images from each category to form a dataset of $600$ images.

### 2.4.2 Image Descriptors and Base Kernels

Since the dataset is now a subset of Caltech-101, it is convenient to use the same ten descriptors and distance functions that are discussed in Section 2.3.2 to establish the base kernels for MKL-DR.

### 2.4.3 Dimensionality Reduction Method

For image clustering, we consider implementing MKL-DR with locality preserving projections (LPP) [47], and denote it as MKL-LPP. The LPP technique is known to be an unsupervised DR scheme that can uncover a low dimensional subspace by preserving the neighborhood structures. The property is particularly useful since respecting the locality information often plays a key factor in the clustering outcomes. To carry out MKL-LPP, we need to reduce LPP to the formulation of graph embedding (2.2). This is accomplished by defining $W = [w_{ij}]$ and $D = [d_{ij}]$ as

$$w_{ij} = \begin{cases} 1, & \text{if } i \in \mathcal{N}_k(j) \vee j \in \mathcal{N}_k(i), \\ 0, & \text{otherwise}, \end{cases} \tag{2.41}$$

$$d_{ij} = \begin{cases} \sum_{n=1}^{N} w_{in}, & \text{if } i = j, \\ 0, & \text{otherwise}. \end{cases} \tag{2.42}$$

Note that LPP is specified by an affinity matrix $W$ and a diagonal matrix $D$, instead of a pair of affinity matrices $W$ and $W'$. In Section 2.2.3, although we only discuss how to derive MKL-DR with DR methods that can be expressed by a pair of $W$ and $W'$, the derivation for those by a pair of $W$ and $D$ is indeed analogous, and the details are omitted here for the sake of space.

### 2.4.4 Quantitative Results

Coupling LPP with the base kernels, MKL-LPP would project the given image data to a learned space, where clustering algorithms will be performed. In the experiments, we restrict the number of clusters to the number of classes in the dataset, i.e., $20$, for all the tested clustering algorithms, and evaluate their performances with the following two criteria: *normalized mutual information* (NMI) [87], and *clustering accuracy* (ACC). (Refer

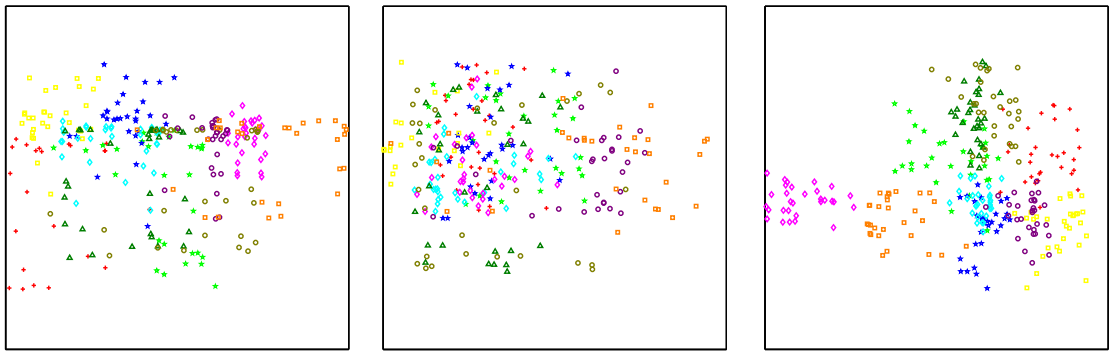
Table 2.3: Clustering performances on the 20-class image dataset. [NMI / ACC%]

| kernel(s) | preprocessing method | affinity propagation | | $k$-means clustering | |
|---|---|---|---|---|---|
| | | without data preprocessing | with data preprocessing | without data preprocessing | with data preprocessing |
| GB-Dist | Kernel LPP (KLPP) | 0.580 / 50.7% | 0.617 / 54.5% | – | 0.634 / 56.3% |
| GB | | 0.531 / 46.2% | 0.598 / 55.2% | – | 0.612 / 54.7% |
| SIFT-Dist | | 0.638 / 59.8% | 0.621 / 57.0% | – | 0.630 / 54.0% |
| SIFT-SPM | | 0.590 / 55.8% | 0.612 / 59.2% | – | 0.630 / 54.7% |
| SS-Dist | | 0.527 / 45.3% | 0.557 / 49.5% | – | 0.565 / 50.2% |
| SS-SPM | | 0.573 / 55.5% | 0.586 / 56.8% | – | 0.603 / 57.7% |
| C2-SWP | | 0.400 / 32.8% | 0.374 / 32.0% | 0.383 / 31.5% | 0.380 / 31.8% |
| C2-ML | | 0.494 / 44.2% | 0.490 / 44.0% | 0.525 / 47.0% | 0.509 / 44.7% |
| PHOG | | 0.455 / 42.7% | 0.490 / 46.3% | – | 0.504 / 47.2% |
| GIST | | 0.515 / 49.2% | 0.491 / 48.8% | 0.494 / 44.7% | 0.502 / 42.8% |
| - | KLPP-*Concatenate* | – | 0.626 / 68.8% | – | 0.667 / 62.3% |
| - | KLPP-*AvgKernel* | – | 0.702 / 77.7% | – | 0.708 / 62.5% |
| All | MKL-LPP | – | **0.737** / **78.3**% | – | **0.759** / **71.2**% |

to [104] for their definitions.)

For the purpose of comparison, we first consider affinity propagation [36] for clustering (without any data preprocessing). The clustering technique is devised to detect representative exemplars (clusters) by taking the similarities between data pairs as input. When image data are represented by each of the ten feature representations, the pairwise similarities are set to the negative distances measured by the corresponding distance function. The clustering results evaluated based on NMI and ACC are reported in the third column of Table 2.3.

We then respectively adopt kernel LPP and MKL-LPP to pre-process the data. The main difference between the two is that kernel LPP learns a projection by taking one base kernel into account at a time, while MKL-LPP considers the ten base kernels simultaneously. In the fourth column of Table 2.3, we show the clustering results of applying affinity propagation to the projected data by both schemes. It can be observed that with the advantage of exploring data characteristics from various aspects, MKL-LPP can achieve significant improvements in the clustering outcomes: NMI is increased from 0.621 to 0.737, and ACC is improved from 59.2% to 78.3%. Similar to object categorization, MKL-LPP makes better use of the complementary image descriptors, and outperforms



(a) KLPP + GB-Dist (b) KLPP + GIST (c) MKL-LPP + All kernels

Figure 2.6: The 2-D visualizations of the projected data. Each point represents a data sample, and its color indicates its class label. The projections are learned by (a) kernel LPP with base kernel GB-Dist, (b) kernel LPP with base kernel GIST, and (c) MKL-LPP with all the ten base kernels.

KLPP-*Concatenate* and KLPP-*AvgKernel*.

The same experiments are repeated by replacing affinity propagation with $k$-means, and the results are given in the last two columns of Table 2.3. Note that if no additional preprocessing is performed, $k$-means is only applicable to data under the representations of C2-SWP, C2-ML and GIST in that the others lead to non-Euclidean spaces. Again, considerable performance gains with MKL-LPP can be concluded from the clustering results.

Besides demonstrating the usefulness of MKL-LPP with the quantitative results, it would be insightful if the projected data can be visually compared. However, for kernel LPP and MKL-LPP, directly embedding the data into a 2-D space for visualization is not practical since both would not yield good clustering results in such a low-dimensional space. Instead, we first use kernel LPP and MKL-LPP to embed the data to low-dimensional spaces in which they respectively achieve their best clustering performance. Then we apply *multidimensional scaling* (MDS) [21] to find the 2-D projections. In Figure 2.6, we show 2-D visualizations of the projected data respectively obtained by kernel LPP with the base kernels GB-Dist and GIST, and by MKL-LPP with all the ten

base kernels. Each point in the figures represents a data sample, and its color indicates its class label. For the purpose of better illustration, only data from ten of the twenty classes (i.e., even-numbered classes) are plotted. Figure 2.6 reveals that MKL-LPP can effectively utilize the data characteristics extracted by different descriptors, and results in a more meaningful projection. That is, data of the same class tend to gather together, while data from different classes are kept apart. This property facilitates a clustering algorithm to identify representative clusters and achieve better performances.

## 2.5 Experimental Results III: Semisupervised Learning for Face Recognition

Our last set of experiments focuses on evaluating the performance gains of applying MKL-DR to semisupervised learning tasks. In solving such problems the given dataset is often partially labeled, and one is required to make use of both the class information of the labeled data and the intrinsic relationships of the unlabeled ones to accomplish the learning tasks. Specifically, we consider the face recognition problem to demonstrate the advantages of our approach, and exploit the property that the face images of an identity spread as a sub-manifold if they are sufficiently sampled to guide and regularize the optimization process in learning a more effective face recognition system.

### 2.5.1 Dataset

The CMU PIE database [84] is used in our experiments of face recognition. It comprises face images of $68$ subjects. For a practical setting, we divide the $68$ people into four equal-size disjoint groups, each of which contains face images from $17$ subjects characterizing by a certain kind of variations. (See Figure 2.7 for an overview.) Specifically, for each subject in the first group, we consider only the images of the frontal pose (C27) taken


Figure 2.7: Four kinds of intraclass variations caused by (a) different lighting conditions, (b) in-plane rotations, (c) partial occlusions, and (d) out-of-plane rotations.

in varying lighting conditions (those under the directory "`lights`"). For subjects in the second and third groups, the images with near frontal poses (C05, C07, C09, C27, C29) under the directory "`expression`" are used. While each image from the second group is rotated by a randomly sampled angle within $[-45°, 45°]$, each from the third group is instead occluded by a non-face patch, whose area is about ten percent of the face region. Finally, for subjects in the fourth group, the images with out-of-plane rotations are selected under the directory "`expression`" and with the poses (C05, C11, C27, C29, C37). All images are cropped and resized to $51 \times 51$ pixels.

Performing face recognition over the resulting dataset is challenging, because the distances among data of the same class (identity) could be even larger than those among data of distinct classes if improper descriptors are used. On the other hand, adding the aforementioned variations to the dataset is useful for emulating the practical situations, which are often caused, say, by imperfect face detectors or in uncontrolled environments.

Figure 2.8: Images obtained by applying the delighting algorithm [46] to the five images in Figure 2.7a. Clearly, variations caused by different lighting conditions are alleviated.

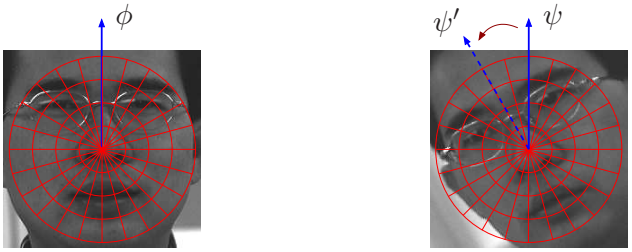



Figure 2.9: Each image is divided into $96$ regions. The distance between the two images is obtained when circularly shifting causes $\psi'$ to be the new starting radial axis.

### 2.5.2 Image Descriptors and Base Kernels

Again our objective is to select a set of visual features that can well capture subjects' characteristics as well as tolerate the large intraclass variations. Totally, we consider using four image descriptors and their respective distance function. Via (2.9), they result in four dissimilarity-based kernels described as follows.

- **RsL2**: Each sample is represented by its pixel intensities in raster scan order. Also, the Euclidean ($L^2$) distance is used to correlate two images. This is a widely used representation for face images.

- **RsLTS**: The base kernel is similar to RsL2, except that the distance function is now based on the *least trimmed squares* (LTS) with $20\%$ outliers allowed. It is designed to take account of the partial occlusions in a face image.

- **DeLight**: The underlying feature representation is obtained from the delighting algorithm [46], and the corresponding distance function is set as $1 - \cos\theta$, where $\theta$ is the angle between a pair of samples under the representation. Some delighting

results are shown in Figure 2.8. It can be seen that variations caused by different lighting conditions are significantly alleviated under the representation.

- **LBP**: As is illustrated in Figure 2.9, we divide each image into $96 = 24 \times 4$ regions, and use a rotation-invariant *local binary pattern* (LBP) operator [69] (with operator setting $LBP_{8,1}^{riu2}$) to detect $10$ distinct binary patterns. Thus an image can be represented by a $960$-dimensional vector, where each dimension records the number of occurrences that a specific pattern is detected in the corresponding region. To achieve rotation invariant, the distance between two such vectors, say, $\mathbf{x}_i$ and $\mathbf{x}_j$, is the minimal one among the $24$ values computed from the distance function $1-\text{sum}(\min(\mathbf{x}_i, \mathbf{x}_j))/\text{sum}(\max(\mathbf{x}_i, \mathbf{x}_j))$ by circularly shifting the starting radial axis for $\mathbf{x}_j$. Clearly, the base kernel is constructed to deal with variations resulting from rotations.

### 2.5.3 Dimensionality Reduction Method

We adopt SDA (semi-supervised discriminant analysis) [11] as the semisupervised DR technique to be generalized by MKL-DR. SDA carries out discriminant learning over labeled data while preserving the geometric structure of unlabeled data. Analogous to LDA and LDE in Section 2.3.3, SDA can be specified by two affinity matrices $W = [w_{ij}]$ and $W' = [w'_{ij}]$, when a partially labeled dataset, $\Omega = \{\mathbf{x}_p, y_p\}_{p=1}^{N_\ell} \cup \{\mathbf{x}_q\}_{q=N_\ell+1}^{N_\ell+N_u}$, is

available:

$$w_{ij} = \begin{cases} 1/n_{y_i} + \alpha \cdot s_{ij}, & \text{if } y_i = y_j, \\ \alpha \cdot s_{ij}, & \text{otherwise,} \end{cases} \tag{2.43}$$

$$w'_{ij} = \begin{cases} 1/N_\ell, & \text{if both } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are labeled,} \\ 0, & \text{otherwise,} \end{cases} \tag{2.44}$$

where

$$s_{ij} = \begin{cases} 1, & \text{if } i \in \mathcal{N}_k(j) \vee j \in \mathcal{N}_k(i), \\ 0, & \text{otherwise,} \end{cases} \tag{2.45}$$

and $\alpha$ is a positive parameter to adjust the relative importance between the label information and the neighborhood relationships.

### 2.5.4 Quantitative Results

In the experiments, we randomly select $12$ images from each subject. Three of them serve as the labeled training data, while the other nine as the unlabeled ones. Totally, we have $816$ (i.e., $12 \times 68$) images for training ($204$ of them are labeled), and the remaining for testing. Since the numbers of images for subjects in different groups may not be the same, an accuracy rate is first computed for each subject. And the reported recognition rate is the average of them. The overall procedure is repeated eight times to reduce the effect of sampling.

We report the mean recognition rates and the standard deviation in the third column of Table 2.4. It can be observed that MKL-SDA achieves a significant performance gain of $16.8\%$ ($= 78.8\% - 62.0\%$) over the best recognition rate by the four kernel SDA

Table 2.4: Recognition rates of several classifiers on CMU PIE dataset. [mean $\pm$ std %]

| method | kernel(s) | dataset (number of classes) | | | | |
|---|---|---|---|---|---|---|
| | | All (68) | Lighting (17) | Rotation (17) | Occlusion (17) | Profile (17) |
| Kernel SDA (KSDA) | RsL2 | $37.8 \pm 2.0$ % | $56.1 \pm 8.0$ % | $29.9 \pm 3.3$ % | $25.5 \pm 2.6$ % | $39.7 \pm 6.4$ % |
| | RsLTS | $54.5 \pm 2.0$ % | $46.8 \pm 5.0$ % | $48.5 \pm 9.7$ % | $68.9 \pm 5.9$ % | $53.7 \pm 6.6$ % |
| | DeLight | $44.7 \pm 1.5$ % | $98.8 \pm 2.3$ % | $15.9 \pm 4.0$ % | $42.6 \pm 5.3$ % | $21.3 \pm 3.7$ % |
| | LBP | $62.0 \pm 2.1$ % | $88.0 \pm 6.3$ % | $52.2 \pm 7.8$ % | $58.1 \pm 8.3$ % | $49.8 \pm 8.6$ % |
| KSDA-*Voting* | - | $65.7 \pm 1.5$ % | $94.1 \pm 4.0$ % | $54.9 \pm 5.2$ % | $66.7 \pm 3.7$ % | $47.2 \pm 6.3$ % |
| KSDA-*Concatenate* | - | $68.1 \pm 2.1$ % | $94.4 \pm 6.2$ % | $47.8 \pm 8.4$ % | $73.0 \pm 5.7$ % | $57.1 \pm 8.6$ % |
| KSDA-*AvgKernel* | - | $70.8 \pm 2.4$ % | $93.6 \pm 4.2$ % | $55.1 \pm 5.7$ % | $79.4 \pm 5.0$ % | $55.1 \pm 8.9$ % |
| KSDA-*SAMME* | - | $67.2 \pm 1.2$ % | $96.1 \pm 3.3$ % | $52.9 \pm 5.9$ % | $68.6 \pm 5.0$ % | $51.1 \pm 5.3$ % |
| MKL-LDA | All | $69.7 \pm 1.5$ % | $97.5 \pm 2.9$ % | $48.0 \pm 7.4$ % | $72.3 \pm 6.9$ % | $61.0 \pm 7.1$ % |
| MKL-LDE | All | $68.1 \pm 2.6$ % | **$99.8 \pm 0.7$ %** | $45.6 \pm 6.3$ % | $71.3 \pm 7.3$ % | $55.6 \pm 10.2$ % |
| MKL-SDA | All | **$78.8 \pm 2.5$ %** | $97.8 \pm 2.4$ % | **$65.4 \pm 8.0$ %** | **$82.8 \pm 3.3$ %** | **$69.1 \pm 5.8$ %** |

(or KSDA for short) classifiers. It also outperforms KSDA-*Voting*, KSDA-*Concatenate*, KSDA-*AvgKernel*, and KSDA-*SAMME*, four mechanisms of merging information from four base kernels, and improves the recognition rates from $65.7\% \sim 70.8\%$ to $78.8\%$.

To evaluate the effect of using unlabeled training data in SDA, we compare MKL-SDA with MKL-LDA and MKL-LDE. The main difference among them is that MKL-SDA considers both labeled and unlabeled training data, while MKL-LDA and MKL-LDE use only the labeled ones. The quantitative results in Table 2.4 show that MKL-SDA can boost the recognition rate about $10\%$ by making use of the additional information from the unlabeled training data.

We also provide the recognition rates with respect to each of the four groups in the last four columns of Table 2.4. (We name each group according to the type of its intraclass variation.) Note that each of such recognition rates is computed by considering only the data in a particular group. And no new classifiers are trained. As expected, the four base kernels generally result in classifiers that produce good performances in dealing with some specific kinds of intraclass variations. For example, the base kernel DeLight achieves a near perfect result for subjects in the Lighting group, and RsLTS yields satisfactory results in the Occlusion group. However, none of them is good enough for dealing with the whole dataset. On the other hand, MKL-DR can effectively combine the four

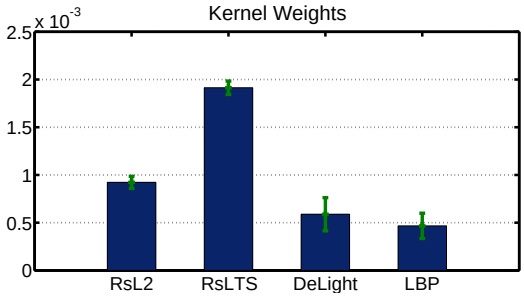



Figure 2.10: The learned kernel weights by MKL-SDA.

base kernels to complement them, and leads to a remarkable increase in accuracy.

Finally, we discuss the learned combination weights over the four base kernels, i.e., $\beta$ in Algorithm 1. We plot the average and the standard deviation of the learned $\beta$ by MKL-SDA in Figure 2.10. Observe that the weights are not directly in proportion to the individual performances of the base kernels. This is due to the fact that the scales of the four base kernels are different. Further the degrees of information complement and redundance among these base kernels should be considered, since they are learned jointly. From the quantitative results where MKL-SDA significantly outperforms KSDA and SDA-*AvgKernel*, it points out that the yielded ensemble kernel by MKL-SDA more effectively embeds the image data with respect to the underlying task. Besides, the low standard deviation highlights the stability of MKL-DR.

## 2.6 Summary

The proposed MKL-DR technique is useful as it has the advantage of learning a unified space of low dimension for data in multiple feature representations. Our approach is general and applicable to most of the graph-based DR methods, and improves their performance. Such flexibilities allow one to make use of more prior knowledge for effectively analyzing a given dataset, including choosing a proper set of visual features to better characterize the data, and adopting a graph-based DR method to appropriately model the

relationship among the data points. On the other hand, via integrating with a suitable DR scheme, MKL-DR can extend the multiple kernel learning framework to address not just the supervised learning problems but also the unsupervised and the semisupervised ones.

Chapter 3

# Localized Multiple Kernel Learning for Object Category Recognition

As each learned model in the previous chapter consists of an ensemble kernel and is applied to all the testing data, we regard it as a *global ensemble* model. We find that the global ensemble model indeed achieves better performance than any others that are constructed by considering only a single kernel. Still, it will be discussed in this chapter, the classification power can be further improved by learning a set of *local ensemble kernels*. This is based upon the observed phenomenon that *the goodness of a descriptor is class-dependent or even sample-dependent*.

## 3.1 Motivation

We start the section by considering a conceptual example shown in Fig. 3.1. There we have a dataset of three classes (indicated by their colors) in three feature representations $a$, $b$, and $c$. The pairwise distances among data points in the three representations, measured by distance functions $d^a$, $d^b$, and $d^c$, are plotted in Fig. 3.1a, 3.1b, and 3.1c respectively. For a global consideration, a new distance function $d_1$ is created by uniformly combining

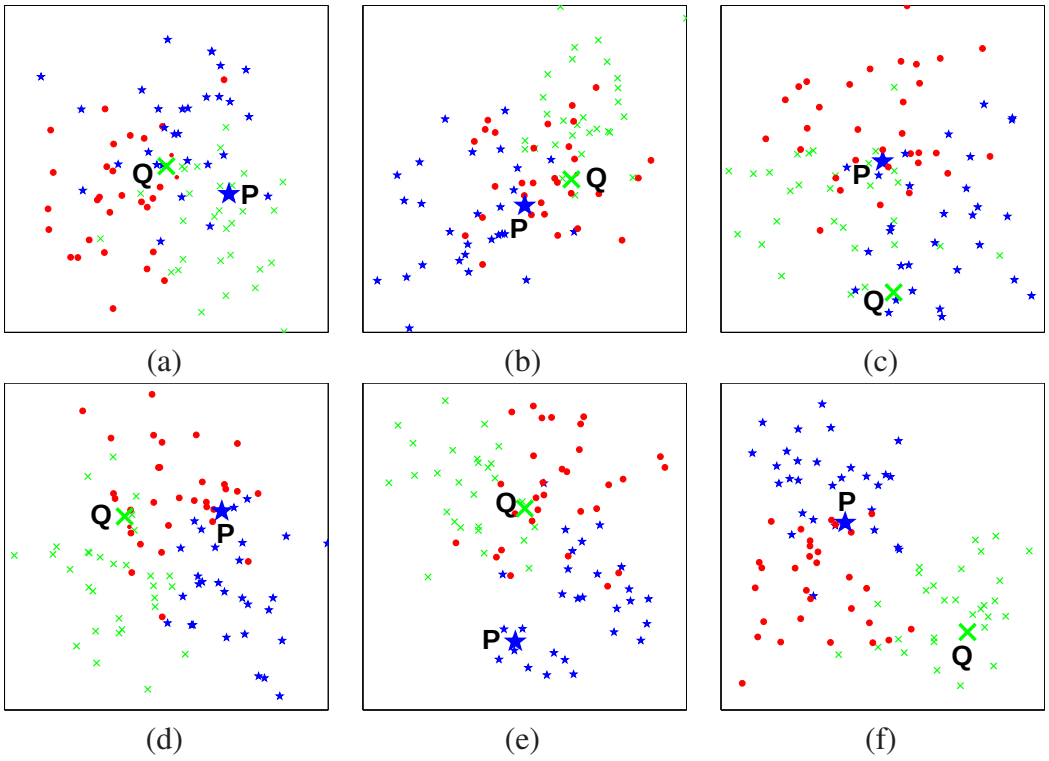



Figure 3.1: The pairwise distances among data of three classes, measured by distance functions (a) $d^a$, (b) $d^b$, (c) $d^c$, (d) $d_1 = \frac{1}{3}(d^a + d^b + d^c)$, (e) $d_2 = \frac{1}{2}(d^a + d^b)$, and (f) $d_3 = \frac{1}{2}(d^b + d^c)$. See the text for details.

the three base ones, i.e., $d_1 = \frac{1}{3}(d^a + d^b + d^c)$, and the resulting pairwise distances among data are displayed in Fig. 3.1d. While $d_1$ can cause a better separation among data of different classes, it tends to misclassify samples $P$ and $Q$. Thus it appears that a local learning scheme may be beneficial. In Fig. 3.1e, the local distance function $d_2 = \frac{1}{2}(d^a + d^b)$ should be effective for performing classification for samples around $P$. Similar effect can be found for $d_3 = \frac{1}{2}(d^b + d^c)$ around $Q$, as shown in Fig. 3.1f.

The example in Fig. 3.1 clearly illustrates not only the importance of using multiple features (distance functions) but also the fact that the goodness of a feature is often dependent on what kinds of data it describes. It results in that the optimal feature or feature combination for recognition vary from category to category, or even from sample to sample if large intraclass variations are involved. To carry out locally adaptive feature selection for reflecting the fact, the learning of each ensemble kernel machine in our for-

mulation is done in a *sample-dependent* fashion. That is, there are as many number of *local ensemble kernels* as the number of training samples. Each local ensemble kernel is optimized not for the whole dataset but for the neighborhood of a specific training sample.

## 3.2 Literature Review

Local learning, e.g., [2, 37] is an extensively explored topic. The motivation of adopting local learning is that a set of local models (e.g., classifiers), each of which is responsible for a subset of samples, typically achieves a better performance than the use of a global model. The improvement mainly comes from the investigation of local properties in learning these local models. For example, to enhance the nearest neighbor rule for classification, Domeniconi and Gunopulos [26] propose a *local flexible metric* by adaptively reweighting dimensions of the feature space according to the sample location.

An important issue in local learning is how to deploy the local models among data points. In [53], Kim and Kittler use $k$-means clustering to partition data into several clusters, and train a linear discriminant analysis (LDA) classifier for each cluster. Dai et al. [23], motivated by that more local models should be located near data with high risks of being misclassified, propose the *responsibility mixture model*, in which an EM algorithm is adopted to place local classifiers according to the distribution of data uncertainty. However, for approaches of this kind, it is hard to pre-determine the optimal number of local models. After all, the Gaussian assumptions for the data distribution or uncertainty distribution are not always valid. Their method alleviates the problem caused by that data are not linearly separable, and increases the performance. An interesting example of local learning is the *SVM-KNN* by Zhang et al. [113], in which an SVM classifier is dynamically yielded in the testing phase by considering training data near the test/query sample. It shows an effective way for accurately performing classification.

Alternatively, in [39, 40, 60, 64], a local model is specifically designed for each train-

ing sample. Thus, no assumptions about the data distribution are required. It makes sense to couple the concepts of local learning and feature fusion. In this manner, visual features could be adaptively combined to best interpret the relationships among data in a local area of the input space. Carrying out the idea, Frome et al. [39], [40] have established a framework in which a local distance function is derived for each training sample by combining several visual features in the level of patches. In the previous work [60], we suggest that various features could be merged in the domain of kernel matrices, and learn a *local ensemble kernel*, a linear combination of the base kernels, for each training sample. The entire training procedure, including localization and regularization of the ensemble kernel machines, could be reformulated as an energy minimization problem on the model of *Markov random fields* (MRFs).

Generally speaking, local learning has demonstrated its attraction in the aspect of accuracy. However, the main drawback is the inefficiency of the training process, because the time complexity grows linearly to the number of the learned local classifiers or distance functions. The situations become worse when dealing with large-sized datasets. However, it is often the case for complex visual learning tasks.

## 3.3 Information Fusion via Kernel Alignment

Within a kernel-based classification framework such as SVMs, the underlying kernel matrix plays the role of information bottleneck: It records the inner product of each pair of training data in some high-dimensional feature space, and has a critical bearing on the resulting decision boundary. From Mercer kernel theory [97], any symmetric and positive semi-definite matrix is a valid kernel matrix, in which there exists one corresponding embedding space for the data, and vice versa. In the section, the kernel alignment, used to measure the degree of agreement between a pair of kernel matrices, is first introduced. Then we describe how to fuse various descriptor in the domain of kernel matrices via

alignment.

### 3.3.1 Kernel Alignment

The kernel alignment, proposed by Cristianini et al. [22], is a quantitative measure of agreement between two kernel matrices, and can be further used to estimate the goodness of a kernel matrix to a specific learning task. To begin with, consider now a two-class dataset $S = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$ with $y_n \in \{+1, -1\}$. The alignment between two kernel matrices $K_1$ and $K_2$ over $S$ (i.e., $K_1, K_2 \in \mathbb{R}^{N \times N}$) is defined by

$$\hat{A}(S, K_1, K_2) = \frac{\langle K_1, K_2 \rangle_F}{\sqrt{\langle K_1, K_1 \rangle_F \langle K_2, K_2 \rangle_F}}, \tag{3.1}$$

where $\langle K_p, K_q \rangle_F = \sum_{i,j=1}^{N} K_p(i,j) K_q(i,j)$.

With (3.1), the goodness of a kernel matrix $K$ with respect to the learning task $S$ can be estimated by the alignment score, denoted as $\hat{A}(S, K, G)$, with a task-specific target kernel $G = \mathbf{y}\mathbf{y}^T$ where $\mathbf{y} = [y_1\ y_2\ \cdots\ y_N]^T$.

Based on the principle of alignment to a target kernel, Lanckriet et al. [56] propose the following procedure to learn a kernel matrix for classification. First, a set of kernel matrices are generated by using different kernel functions or different parameter values. Then, maximizing the alignment score is integrated into the learning process of the kernel machine. The optimized kernel is a convex combination of the pre-generated kernel matrices. The resulting kernel machine empirically achieves good performances in their experiments.

### 3.3.2 Feature Fusion via Kernel Alignment

Like what we have described in (2.9), a base kernel can be constructed for each visual feature. Let $\Omega = \{K^1, K^2, \ldots, K^M\}$ be the *kernel bank* yielded by applying the proce-

dure to all the $M$ kinds of features. For multi-class classification problems, we redefine the target kernel matrix $G$ as

$$G(i,j) = \begin{cases} \lambda, & \text{if } y_i = y_j, \\ -1, & \text{otherwise}, \end{cases} \tag{3.2}$$

where $\lambda$ is a positive constant and its value to set to make the resulting $G$ have a zero mean.

Now feature fusion over the kernel bank $\Omega$ can be achieved by kernel alignment with respect to target kernel $G$. In particular, we follow the formulation of Lanckriet et al. [56] to solve

$$\begin{aligned} \max_{\boldsymbol{\beta}} \quad & \hat{A}(S,K,G) \\ \text{subject to} \quad & K = \sum_{m=1}^{M} \beta^m K^m, \\ & trace(K) = 1, \\ & \beta^m \geq 0,\ m = 1,2,...,M. \end{aligned} \tag{3.3}$$

*Semidefinite programming* can be applied to solved the optimization problem in (3.3). Alternatively, Hoi et al. [48] show that the optimization problem can be more efficiently solved using *quadratic programming* after reducing (3.3) into the following equivalent formulation:

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & \boldsymbol{\beta} D^T D \boldsymbol{\beta} \\ \text{subject to} \quad & vec(G)^T D \boldsymbol{\beta} = 1, \\ & \beta^m \geq 0,\ m = 1,2,...,M, \end{aligned} \tag{3.4}$$

where $vec(A)$ is the *vectorization* of matrix $A$, and $D = [vec(K^1)\ vec(K^2)\ \cdots\ vec(K^M)]$. Note that all kernel matrices in $\Omega$ will be normalized before solving (3.3) or (3.4) as suggested in [56].

The optimized variable $\boldsymbol{\beta} = [\beta^1\ \beta^2\ \cdots\ \beta^M]$ in (3.3) or (3.4) specifies how the ensemble kernel $K$ is constructed from kernel bank $\Omega$. It can also be interpreted as the weights for combining the $M$ visual features with respect to the learning task.

We can now derive the kernel machine (SVMs are chosen here) by using the ensemble kernel matrix $K = \sum_{m=1}^{M} \beta^m K^m$ as input. For handling data of multiple classes, *one-against-one*, *one-against-rest*, or *decision directed acyclic graph* (DDAG) [73] strategies can be chosen and used. For instance, if one-against-one rule is used, the learned binary SVM classifier $sign(f)$ for separating data of classes $p$ and $q$ would be in the following form:

$$\begin{aligned} f(\mathbf{x}) &= \sum_{i \in \mathbf{i_p}} \alpha_i k(\mathbf{x}_i, \mathbf{x}) - \sum_{j \in \mathbf{i_q}} \alpha_j k(\mathbf{x}_j, \mathbf{x}) + b, \\ &= \sum_{i \in \mathbf{i_p}} \alpha_i \sum_{m=1}^{M} \beta^m k^m(\mathbf{x}_i, \mathbf{x}) - \sum_{j \in \mathbf{i_q}} \alpha_j \sum_{m=1}^{M} \beta^m k^m(\mathbf{x}_j, \mathbf{x}) + b, \end{aligned} \tag{3.5}$$

where $\mathbf{i_p}$ and $\mathbf{i_q}$ are the sets of indices of training data whose class labels are $p$ and $q$ respectively.

## 3.4 Local Ensemble Kernel Learning

The previous section introduces a general principle for fusing features. Although it does provide a unified way of globally combining different feature representations, the approach is too general to account for the interclass and intraclass variations in a complex object recognition problem.

### 3.4.1 Initialization via Localized Kernel Alignment

In learning local ensemble kernels, intuitively one can try to generalize the idea of kernel alignment to *localized* kernel alignment. As it turns out, the approach would give satisfactory results. Nevertheless we only use them as the initial observations to our proposed optimization framework for reasons that will become evident later.

We first introduce the notion of *neighborhood* for each sample $\mathbf{x}_i$. Recall that the $r$th representation of $\mathbf{x}_i$ is $\mathbf{x}_i^r$, and the distance function is $d^r$. The neighborhood of $\mathbf{x}_i$ can be specified by a normalized weight vector $\mathbf{w}_i = [w_{i,1}, \ldots, w_{i,\ell}]$, where

$$w_{i,j} = \frac{1}{M}(w_{i,j}^1 + w_{i,j}^2 + \cdots + w_{i,j}^M) \tag{3.6}$$

$$w_{i,j}^r = \frac{\exp\left(\frac{-[d^r(\mathbf{x}_i^r, \mathbf{x}_j^r)]^2}{\sigma^r}\right)}{\sum_{k=1}^{\ell} \exp\left(\frac{-[d^r(\mathbf{x}_i^r, \mathbf{x}_k^r)]^2}{\sigma^r}\right)}. \tag{3.7}$$

We then define the *local target kernel* of $\mathbf{x}_i$ according to $\mathbf{w}_i$ as follows:

$$G_i(p, q) = w_{i,p} \times w_{i,q} \times G(p, q), \text{ for } 1 \leq p, q \leq \ell. \tag{3.8}$$

By replacing $G$ with $G_i$ in (3.3) and (3.4), we complete the formulation of localized kernel alignment. The new constrained optimization now yields an optimal vector $\boldsymbol{\alpha}_i$ and therefore a local ensemble kernel $K_i$ for each sample $\mathbf{x}_i$.

Note that in (3.8) a weight distribution is dispersed over the local target kernel $G_i$ such that whenever an element of $G_i$ relates to more relevant samples in the neighborhood of $\mathbf{x}_i$, it will have a larger weight (according to (3.6) and (3.7)). This property enables the resulting kernel $K_i = \sum \alpha_i^r K^r$ to be formed by emphasizing those $K^r \in \Omega$ that their corresponding visual cues can more appropriately describe the relations between $\mathbf{x}_i$ and its neighbors. Meanwhile, $\sigma^r$ in (3.7) can be used to control the extent of locality around $\mathbf{x}_i$. To ensure a consistent way of specifying a neighborhood for each representation $\mathbf{x}_i^r$, we adopt the following scheme: Fixing some values of, say, $s$ and $t$, we adjust the value

of $\sigma^r$ by binary search such that the nearest $s$ neighbors of $\mathbf{x}_i$, using distance $d^r$, will take up $t\%$ of the total weight of $\mathbf{w}_i$.

### 3.4.2 Local Ensemble Kernel Optimization

There are two main reasons to look beyond the local ensemble kernels generated by kernel alignment. First, in most of the local learning approaches, e.g., [23, 39, 53] the resulting classifiers are determined by a relatively limited number of training samples from the neighborhood or some local group. They are sometimes sensitive to curse of dimensionality, or at the risk of overfitting caused by noisy data. To ease such unfavorable effects, we prefer a local solution with proper regularization. Second, although the kernel alignment technique in (3.1) has its own theoretical merit [22], we find that it does not always precisely reflect the *goodness* of a kernel matrix. In our empirical testing, kernel matrices that better align to the target kernel may fail to achieve better classification performance.

To address the above two issues, we construct an MRF graphical model $(V, E)$ for learning local ensemble kernels. For each sample $\mathbf{x}_i$, we create an observation node $\mathbf{o}_i$ and a state node $\boldsymbol{\beta}_i$. And for each pair of $\mathbf{o}_i$ and $\boldsymbol{\beta}_i$, an edge is connected. In addition if $\mathbf{x}_j$ is one of the $c$ nearest neighbors of $\mathbf{x}_i$ according to (3.6), an edge will be created between $\boldsymbol{\beta}_i$ and $\boldsymbol{\beta}_j$. Hence $|V| = 2\ell$ and $|E| \leq (c+1)\ell$. Thus there are two types of edges: $E_1$ includes edges connecting a state node to its observation node, and $E_2$ comprises those linking two state nodes. Clearly $E = E_1 \cup E_2$. With the MRF so defined, we consider the following energy function:

$$E(\{\boldsymbol{\beta}_i\}) = \sum_{(i,i)\in E_1} V_d(\boldsymbol{\beta}_i, \mathbf{o}_i) + \sum_{(i,j)\in E_2} V_s(\boldsymbol{\beta}_i, \boldsymbol{\beta}_j). \tag{3.9}$$

**On designing $V_d(\boldsymbol{\beta}_i, \mathbf{o}_i)$.** In general the data term $V_d$ should incorporate observation evidence at $\mathbf{x}_i$. Specifically, we consider the neighborhood relation of $\mathbf{x}_i$ specified by (3.6) and the $\boldsymbol{\alpha}_i$ derived by localized kernel alignment. For the ease of algorithm design,

the total number of possible states should be controlled within a manageable range. We therefore run $k$-medoids clustering (based on the alignment score) to divide $\{\boldsymbol{\alpha}_i\}_{i=1}^{\ell}$ into $n$ clusters, denoted as $\{\hat{\boldsymbol{\alpha}}_p\}_{p=1}^{n}$. ($n = 50$ in all our experiments.) The mapping $\boldsymbol{\alpha}_i \mapsto \hat{\boldsymbol{\alpha}}_{p(i)}$ is used to describe that $\boldsymbol{\alpha}_i$ is assigned to the $p(i)$th cluster, and its vector value is approximated by $\hat{\boldsymbol{\alpha}}_{p(i)}$. Consequently, $K_i = \sum_{r=1}^{M} \alpha_i^r K^r$ is replaced by $\hat{K}_{p(i)} = \sum_{r=1}^{M} \hat{\alpha}_{p(i)}^r K^r$. In the MRF graph each observation node $\mathbf{o}_i$ can now be set to $\hat{\boldsymbol{\alpha}}_{p(i)}$. With these modifications, we are ready to define $V_d$ as follows:

$$V_d(\boldsymbol{\beta}_i = \hat{\boldsymbol{\alpha}}_q, \mathbf{o}_i = \hat{\boldsymbol{\alpha}}_{p(i)}) = \sum_{j=1}^{\ell} w_{i,j} \times 1_{\{f_q(\mathbf{x}_j) \neq y_j\}} \tag{3.10}$$

where $1_{\{\cdot\}}$ is an indicator function, and $f_q(\mathbf{x}_j)$ is the result of leave-one-out (LOO) SVM using kernel $\hat{K}_q$ (removing the $j$th row and column) on $\mathbf{x}_j$. In practice, implementing the LOO setting in (3.10) is too time-consuming. Nevertheless, it can be reasonably approximated by the following scheme: Each LOO testing on $\mathbf{x}_j$ can be carried out by applying $f_q$ to $\mathbf{x}_j$ with the constraint that $\mathbf{x}_j$ cannot be one of the support vectors. (If $\mathbf{x}_j$ is a support vector, then remove it from $f_q$.)

**On designing $V_s(\boldsymbol{\beta}_i, \boldsymbol{\beta}_j)$.** We adopt the Potts model to make $V_s$ a smoothness prior in (3.9) on the free variable space. The regularization would enrich our kernel learning formulation against noisy data. Specifically, we have

$$V_s(\boldsymbol{\beta}_i = \hat{\boldsymbol{\alpha}}_{q_i}, \boldsymbol{\beta}_j = \hat{\boldsymbol{\alpha}}_{q_j}) = \begin{cases} 0, & \text{if } q_i = q_j, \\ t, & \text{else if } y_i \neq y_j, \\ P \times t, & \text{otherwise,} \end{cases} \tag{3.11}$$

where $t$ is a constant penalty, and constant $P \geq 1$ is used to increase penalty between state nodes $\boldsymbol{\beta}_i$ and $\boldsymbol{\beta}_j$ whose samples belong to the same object category.

With (3.10) and (3.11), the energy function in (3.9) is fully specified. Since finding the exact solution to minimizing (3.9) is NP-hard, we apply *graph cuts* [10] to approximating the optimal solution. Let $\{\boldsymbol{\beta}_i^* = \hat{\boldsymbol{\alpha}}_{p^*(i)}\}$ be the outcome derived by graph cuts. Then the *optimal* local ensemble kernel of $\mathbf{x}_i$ learned with (3.9) is $K_i^* = \sum_{r=1}^{M} \hat{\alpha}_{p^*(i)}^r K^r$. In testing, given a test sample $\mathbf{z}$, we can readily locate the nearest $\mathbf{x}_i$ to $\mathbf{z}$ by referencing (3.6), and use SVM with the local ensemble kernel $K_i^*$ to perform classification.

## 3.5 Experimental Results

A real-world recognition application often involves objects from diverse and broad categories. Even for objects from the same category, their appearances and characteristics can still vary due to different poses, scales, or lighting conditions. Nonetheless, a problem like this serves as a good test bed for evaluating the proposed local ensemble kernel learning. In our implementation, we formulate the recognition task as a multiclass classification problem, and use LIBSVM [14], in which one-against-one rule is adopted, to learn the classifiers with the kernel matrices produced by our method. We carry out experiments on two sets of images. The first one is Caltech-101 collected by Fei-Fei et al. [33], and the second set is a mixture of images from Corel, CUReT [24] and Caltech-101. Detailed experimental results and discussions are given below.

### 3.5.1 Image Descriptors and Base Kernels

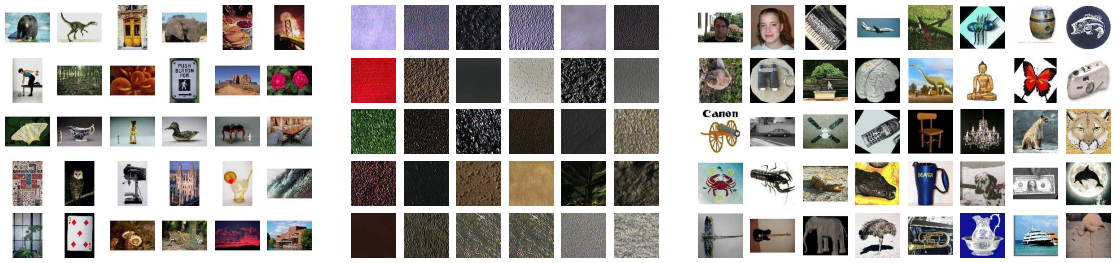
We briefly describe the image representations and their corresponding distance functions used in our experiments. These features are chosen to capture diverse characteristics of objects, such as shape, color, and texture. And our expectation is to use them to construct a kernel bank $\Omega$ that is rich enough for generating good ensemble kernels for recognizing objects of various categories. Note that hereafter a term marked in **bold font** is to denote a pair of an image representation and its distance measure.

- Shape-based features can provide strong evidence for object recognition, e.g., [6, 39, 63]. We adopt the geometric blur descriptor proposed by Berg et al. [6]. The descriptor summarizes the edge responses within an image patch, and is relatively robust (via employing a spatially varying kernel) to shape deformation and affine transformation. Since the optimal kernel scale can depend on several factors, e.g., the object size, we implement geometric blur descriptors with two kernel scales, and denote them as **GB-L** and **GB-S** respectively. Our implementation of geometric blur follows (2) of [113], in which spatial information is used.

- Roughly speaking, texture feature refers to those image patterns that display homogeneity. To capture such visual cues, we consider the setting of [99], where $99$ filters from three filter banks are used to generate the **textons** (the vocabularies of texture prototypes [59]). An image can then be represented by a histogram that records its probability distribution over all the generated textons. Like in [59, 99], the $\chi^2$ distance is selected as the similarity measure.

- We use the SIFT (Scale Invariant Feature Transform) detector [63] to find interest points in an image. To measure the distance between two images, vector quantization, as suggested in [85], is applied to transform a bag-of-features representation to a single feature vector via clustering. Since the number of clusters can critically affect the performance, we implement two different settings: The numbers of clusters are set to 2000 (**SIFT-2000**) and 500 (**SIFT-500**) respectively. We also normalize the feature vector of each image to a distribution, and again use the $\chi^2$ distance as the similarity measure.

- Serre et al. [81] propose a set of features that emulates the visual system mechanism. Through a four-layer (known as S1, C1, S2, and C2) hierarchical processing, an image can be expressed by a set of scale and translation-invariant **C2** features. Motivated by their good performance for recognition, we use the C2 features as one



(a) Corel (b) CUReT (c) Caltech-101

Figure 3.2: 100 image categories. (a) 30 categories from Corel, (b) 30 categories from CUReT, and (c) 40 categories from Caltech-101.

of our image representations in the experiments. For this representation, Euclidean distance is applied to measuring the dissimilarity between a pair of images.

- The visual features described so far catch characteristics only based on gray-level information. We thus further consider color-related image features, especially those which have compact representations and match human intuition. Specifically, we use a 125-bin color histogram (**CH**) extracted from the HSV color space to represent an image. To include spatial information, the 250-bin color coherence vector (**CCV**) [71] is also implemented. After normalizing a CH or CCV of an image to a distribution, *Jeffrey divergence* is used as the distance function.

### 3.5.2 Caltech-101 Dataset

The Caltech-101 [33] dataset consists of 101 object categories and an additional class of background images. Each category contains about 40 to 800 images. Although objects in these images often locate in the central regions, the total number of categories (i.e., 102) and large intraclass variation still make this set a very challenging one. Some examples are shown in Figure 3.2c.

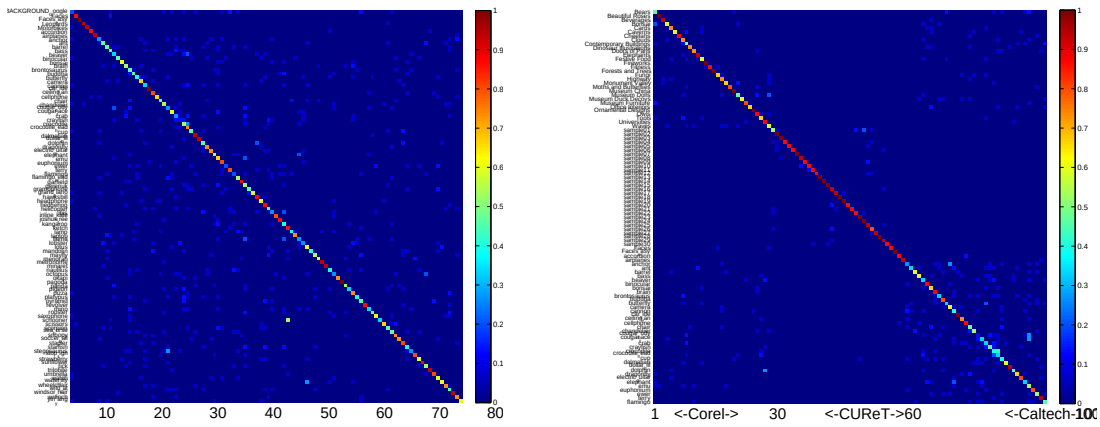
As the sizes of images in this set are different and some of our adopted image representations are sensitive to this issue, we resize each image into a resolution around $300 \times 200$. (The aspect ratio is maintained.) For the sake of comparison, our experiment setup is sim-

Table 3.1: Recognition rates for Caltech-101.

| Method | Rep.-$r$ | Caltech-101 dataset | |
|---|---|---|---|
| | | With-background | Without-background |
| $K^r$ | GB-L | *51.57 %* | *51.95 %* |
| | GB-S | **53.40 %** | **53.86 %** |
| | Texton | 20.73 % | 20.99 % |
| | SIFT-2000 | 28.76 % | 29.31 % |
| | SIFT-500 | 23.86 % | 24.29 % |
| | C2 | 31.37 % | 31.68 % |
| | CH | 13.66 % | 13.80 % |
| | CCV | 15.10 % | 15.25 % |
| $K$ | All | 54.38 % | 54.92 % |
| $K_i$ | All | *57.25 %* | *57.95 %* |
| $K_i^*$ | All | **59.80 %** | **61.25 %** |

ilar to the one in Berg et al. [6] and Zhang et al. [113]. Namely, we randomly select $30$ images from each category: $15$ of them are used for training and the rest are used for testing. However, in [39], the class of background images (i.e., `BACKGROUND_Google`) is excluded from the experiments. We thus include both the two settings in our experiments, and denote them as *with-background* and *without-background*. The quantitative results and the confusion table of testing Caltech-101 are reported in Table 3.1 and Figure 3.3a, respectively. In Table 3.1, the exact meanings of the abbreviations for the eight kinds of image representations, listed in the `Rep.`-$r$ column, have been described in Section 4. In the `Method` column, we specify what kind of kernel matrix is used with SVM to form a kernel machine: $K^r$ means the kernel (in $\Omega$) with respect to the representation $\mathbf{x}^r$ is used, and analogously $K$, $K_i$ and $K_i^*$ indicate the use of an ensemble kernel derived by solving global kernel alignment (3.4), localized kernel alignment (3.8), and MRF optimization (3.9), respectively.

The performance gain by our technique is significant. Despite our implementation of geometric blur is less effective (the **GB-L** and **GB-S** entries in Table 3.1) than those reported in [113], the proposed ensemble kernel learning can still achieve state-of-the-art




(a) (b)

Figure 3.3: The confusion tables by our method on two different datasets. (a) Caltech-101. (b) Corel + CUReT + Caltech-101.

recognition rates.

### 3.5.3 Corel + CUReT + Caltech-101 Dataset

In testing with Caltech-101 dataset, we observe that the performance of geometric blur descriptor [6] is noticeably dominant. This phenomenon generally causes the resulting ensemble kernel is not uniformly combined, and therefore the advantage of our method may not be fully exploited. We thus construct a second dataset by collecting images from different sources to further increase the data variations.

We first select $30$ image categories from the Corel image database, which is widely used in the research of image retrieval. Images in the same category share the same semantic concept, but still have their individual variety. To illustrate, images from each of the $30$ categories are shown in Figure 3.2a. We then collect the first $30$ texture categories from CUReT [24] database. Texture images within a category are pictured for the same material under various viewing angles and illumination conditions. Similar to [99], we crop the central $200 \times 200$ texture region of each image, and convert it to grayscale. However unlike in [99], we do not normalize the intensity distribution to zero mean and unit standard deviation in that this pre-processing is not applied to images from the other two

sources. Figure 3.2b gives an overview of the $30$ categories selected from CUReT. Finally, we choose $40$ object categories from Caltech-101 dataset according to their alphabetical order. (Note that the background category is excluded.) These categories are indeed those shown in Figure 3.2c. In total the resulting dataset from the three sources has $100$ image categories.

Similar to the setting in Section 5.1, we randomly select $30$ images from each category. Half of them are used for training, and the rest are used for testing. The quantitative results and the confusion table are shown in Table 3.2 and Figure 3.3b, respectively. In Table 3.2, there are four recognition rates recorded for each scheme. The first three are evaluated by considering only samples within specific datasets, and the last is computed by taking all samples into account.

From Table 3.2 and Figure 3.3b, we have the following observations: 1) In the Corel database, several image representations can achieve comparable performance, and tend to complement each other. Thus all the three schemes $K$, $K_i$ and $K_i^*$ that combine various features achieve substantial improvements. 2) Unlike testing with Caltech-101, the optimal feature combinations for classifying objects in the combined dataset are more diversely distributed. Thus the accuracy improvement of the scheme $K$ that globally learns a single fusion of visual features for all samples is relatively limited, compared with those by the two local schemes $K_i$ and $K_i^*$. 3) No matter using which dataset, the performance of our approach is better than the best performance obtained from using a single image representation. This means that our method can effectively select good feature combination for each sample, and thus improve the recognition rates.

### 3.5.4 Complexity Analysis

Most of the time complexity in training is consumed by the construction of the kernel bank $\Omega$. It involves the pairwise distance calculations, and some of our chosen distances

Table 3.2: Recognition rates for Corel + CUReT + Caltech-101.

| Method | Rep.-$r$ | Corel + CUReT + Caltech-101 dataset | | | |
|---|---|---|---|---|---|
| | | Corel (30 classes) | CUReT (30 classes) | Caltech (40 classes) | ALL (100 classes) |
| $K^r$ | GB-L | *59.91 %* | 70.36 % | *52.30 %* | *60.00 %* |
| | GB-S | **62.14 %** | 75.47 % | **53.30 %** | **62.60 %** |
| | Texton | 53.33 % | **88.89 %** | 23.67 % | 52.13 % |
| | SIFT-2000 | 45.56 % | *83.56 %* | 30.00 % | 50.73 % |
| | SIFT-500 | 42.00 % | *83.56 %* | 25.83 % | 48.00 % |
| | C2 | 48.22 % | 59.78 % | 30.00 % | 44.40 % |
| | CH | 55.33 % | 24.22 % | 16.17 % | 30.33 % |
| | CCV | 56.44 % | 33.78 % | 17.47 % | 34.13 % |
| $K$ | All | 75.78 % | 86.00 % | 46.17 % | 67.00 % |
| $K_i$ | All | *76.67 %* | **92.89 %** | *55.83 %* | *73.20 %* |
| $K_i^*$ | All | **77.33 %** | *92.67 %* | **59.33 %** | **74.73 %** |

require extensive computation time. For $1500$ training samples, this step would take several hours to complete. In addition, after kernels are grouped as clusters in the MRF optimization, learning the sample-dependent SVM classifiers can be done in less than $2$ minutes. In testing a novel sample, our method carries out nearest neighbor search to locate the appropriate local classifier and then performs the classification. Totally, it would take about $5\frac{1}{2}$ minutes for testing a new sample. Finally, we remark that although a local kernel is learned for each training sample, we do not need to store the whole kernel matrix but the ensemble weights. Hence there is no extra space requirements due to our proposed method.

## 3.6 Summary

Motivated by that the best visual feature combination for classification could vary from object to object, we have introduced a *sample-dependent* learning method to construct ensemble kernel machines for recognizing objects over broad categories. Overall, we have strived to demonstrate such advantages with the proposed optimization framework over an MRF model, of which we are able to use kernel alignment to give good initializations, and also with the promising experimental results on two extensive datasets.

Chapter 4

# Efficient Localized Multiple Kernel Learning by Multi-task Boosting

In the previous chapter, we have demonstrated that local learning can reasonably resolve the difficulties of object recognition caused by large intra-class variations. However, the *high risk of overfitting* and the *heavy computational cost* in training numerous local classifiers often limit its applicability. In this chapter, we provide a solution to dealing with the two unpleasant issues of local learning based upon the observation that sample-specific local classifiers will lie in some parametric space and spread as a manifold-like structure.

The soul of our approach is to *cast the multiple, independent training processes of local classifiers as a correlative multi-task learning problem*. Specifically, we establish a parametric space (i.e., classifier space) where these local classifiers lie and spread as a manifold-like structure, and develop a new boosting algorithm, called *multi-task boosting* to perform local model training by completing the manifold embedding. Via sharing the common embedding space, the learning of each local classifier can be properly regularized by the extra knowledge from other models, while the training time is also significantly reduced.

## 4.1 Motivation

Local learning is effective in tackling large intraclass variations in data. Unlike the global approach, it considers multiple local models, each of which is learned to account for only a subset of data. For instance, to detect both frontal and profile faces, Schneiderman and Kanade [78] use multiple *view-based* detectors to specifically uncover faces within a certain range of poses. Similar strategies are also adopted in recent researches for image classification and recognition, e.g., [39, 40, 60, 62, 64]. However, except for certain cases such as the face images that can be reasonably aligned and divided into subset by their pose, it is generally a difficult task to partition image data into meaningful subsets, each of which is covered by one local model. In the aforementioned approaches, the learning is performed for each sample or its neighborhood, and results in local models as many as the number of training data. In [39, 40, 64], the local model corresponds to a distance function, while in [43, 60] it relates to a kernel matrix.

Even for a modest size of training data, learning the sample-specific local models may already not scale well. Take, for example, the two popular benchmark datasets for object recognition, Caltech-101 [33] and Pascal VOC 2007 [29]. Each comprises thousands of images, and learning all the local models often takes time in the order of hours or even days. It almost makes parameter tuning, a key step for obtaining satisfactory classification results, infeasible. Further, local learning may suffer from the risk of overfitting in that it often considers a relatively small group of training data. Addressing these two unfavorable issues of local learning will be among the main focuses of this work.

## 4.2 Literature Review

This section describes some of the key concepts related to the establishment of the proposed approach, including local learning and multi-task learning.

### 4.2.1 Local Learning

Local learning has been an active research topic in machine learning [2]. It draws on the idea that a local model should most likely better characterize the distinctive properties shared by a small subset of data than a global one does for the whole data. Consequently adopting multiple locally adaptive models would achieve better performances. For example, to enhance the nearest neighbor rule for classification, Domeniconi and Gunopulos [26] propose a *local flexible metric* by adaptively reweighting dimensions of the feature space according to the sample location.

An important issue in local learning is how to deploy the local models among data points. In [53], Kim and Kittler use $k$-means clustering to partition data into several clusters, and train a *linear discriminant analysis* (LDA) classifier for each cluster. Dai et al. [23], motivated by that more local models should be located near data with high risks of being misclassified, propose the *responsibility mixture model*, in which an EM algorithm is adopted to place local classifiers according to the distribution of data uncertainty. However, for approaches of this kind, it is hard to pre-determine the optimal number of local models. After all, the Gaussian assumptions for the data distribution or uncertainty distribution are not always valid.

Alternatively, in [39, 40, 64], a local model is specifically designed for each training sample. Thus, no assumptions about the data distribution are required. In addition, local learning in these works is often coupled with feature fusion. That is, not only the discriminant functions but also the discriminant visual features are jointly selected to enhance the power of each local model. Nevertheless, training the sample-specific models is time-consuming.

### 4.2.2 Multi-task Learning

Pertaining to object recognition, the tasks of learning a local model from a given dataset are typically correlated. As is pointed out from the literature of multi-task learning, e.g., [1, 13, 91], investigating related tasks *jointly* in most cases can achieve a considerable performance improvement than *independently*, since the extra knowledge from other tasks may convey useful information to the completion of the underlying task.

The proposed approach is related to *JointBoost* [91] in the sense that multiple boosted classifiers are simultaneously generated under a specific form of multi-task learning. In contrast to JointBoost, our method can accomplish it more efficiently. We also note that our use of *dyadic hypercuts*, introduced in Moghaddam and Shakhnarovich [66], as the weak learners for boosting is pivotal to the formulation in that they can effectively capture useful information in a kernel matrix, and elegantly connect multiple kernel learning with boosting.

## 4.3 The Proposed Framework

We begin by specifying the notations used in this work, defining the problem of local learning for object recognition, and describing its link to the multi-task learning.

### 4.3.1 Notations

Since the multi-class object recognition can always be reduced to an array of two-class problems by adopting *one-against-one* or *one-against-all* rules, we assume a binary dataset $S = \{\mathbf{x}_n \in \mathcal{X}, y_n \in \pm 1\}_{n=1}^{N}$, where $N$ is the data size and $\mathcal{X}$ denotes the input space.

In view of the complexity of visual object recognition, it is hard to find a universal descriptor to well characterize the whole dataset. Instead, we consider representing each $\mathbf{x}_n$ with totally $M$ kinds of different descriptors, i.e., $\mathbf{x}_n = \{\mathbf{x}_{n,m} \in \mathcal{X}_m\}_{m=1}^{M}$, and each

descriptor is associated with a distance measure $d_m : \mathcal{X}_m \times \mathcal{X}_m \to \mathbb{R}$.

The useful data representations are often of high dimensional and in diverse forms, such as vectors [67], histograms [9], bags of features [113], or pyramids [58]. To avoid the difficulties caused by working with these varieties, we represent data under each descriptor by a kernel matrix. And it leads to $M$ kernel functions $\{k_m\}_{m=1}^M$ together with the corresponding kernel matrices $\{K_m\}_{m=1}^M$:

$$K_m(n, n') = k_m(\mathbf{x}_n, \mathbf{x}_{n'}) = \exp(-\gamma_m d_m^2(\mathbf{x}_n, \mathbf{x}_{n'})) \tag{4.1}$$

where $\gamma_m$ is a positive constant.

### 4.3.2 Problem Definition

Our goal is to carry out local learning by deriving a classifier $f_i$ for each training sample $\mathbf{x}_i$ such that $f_i$ is expected to give good performances for testing samples falling around $\mathbf{x}_i$. To this end, we specify the *neighborhood* of $\mathbf{x}_i$ with a weight distribution $\mathbf{w}_i = \{w_{i,n}\}_{n=1}^N$ over $S$ by

$$w_{i,n} = \begin{cases} 1/C, & \text{if } \mathbf{x}_n \in C\text{-NN of } \mathbf{x}_i, \\ 0, & \text{otherwise,} \end{cases} \tag{4.2}$$

where $C$-NN of $\mathbf{x}_i$ denotes the $C$ nearest neighbors of $\mathbf{x}_i$ (including $\mathbf{x}_i$ itself). In cases that multiple descriptors are used, we compute $\mathbf{w}_i$ for data under each representation and average the outcomes. Then each local classifier $f_i$ can be obtained by coupling an optimal function with suitable features to best discriminate the weighted dataset $S_i = \{\mathbf{x}_n, y_n, w_{i,n}\}_{n=1}^N$. The purpose of our local learning is to learn such classifiers $\{f_i\}_{i=1}^N$.

Specifically, each $f_i$ is resulted from the boosting algorithm, and consists of a set of weak learners. Let $\mathcal{H}$ denote the domain of weak hypotheses. Our formulation assumes that each weak learner $h \in \mathcal{H}$ is generated by referencing only the $M$ kernels defined

in (4.1). The main advantage of so doing is that it uses these kernels as the unified information bottleneck, and enjoys the convenience of working with different descriptors and distance measures. In our discussions below, we will treat the training of each local classifier as a particular *task*. It follows that the numbers of tasks and training samples are both $N$. For sake of clarity, hereafter we will use subscript $i$ as the index to tasks or classifiers, and $n$ to the training samples.

### 4.3.3 Formulation

While local learning is effective for addressing complicated vision problems like object recognition, care must be taken to ensure that the learning procedure has been properly done for a given dataset. In particular, we pinpoint the following two critical issues related to learning local classifiers $\{f_i\}_{i=1}^N$. First, each $f_i$ is learned with a small portion of training data. When the number of weak learner candidates $|\mathcal{H}|$ is large or infinite, $f_i$ is at the high risk of overfitting. Second, learning a local classifier for each training sample is indeed an inefficient procedure. And the situation gets worse when dealing with a large dataset, but it is often the case for vision applications.

To alleviate the two above-mentioned unfavorable effects, we view completing the independent training processes of $\{f_i\}_{i=1}^N$ as a correlative multi-task learning problem. This is accomplished by assuming these local classifiers share the same weak learners, but have their respective ensemble coefficients, i.e.,

$$f_i(\mathbf{x}) = \sum_{t=1}^{T} \alpha_{i,t} h_t(\mathbf{x}), \text{ for } i = 1, 2, ..., N. \tag{4.3}$$

With (4.3), all the classifiers will be learned jointly. We will later describe a systematic way for constructing $\{f_i\}_{i=1}^N$ with multi-task learning in the next section. For now we give justifications on why the two unfavorable effects can be eased by setting local classifiers in the form of (4.3).

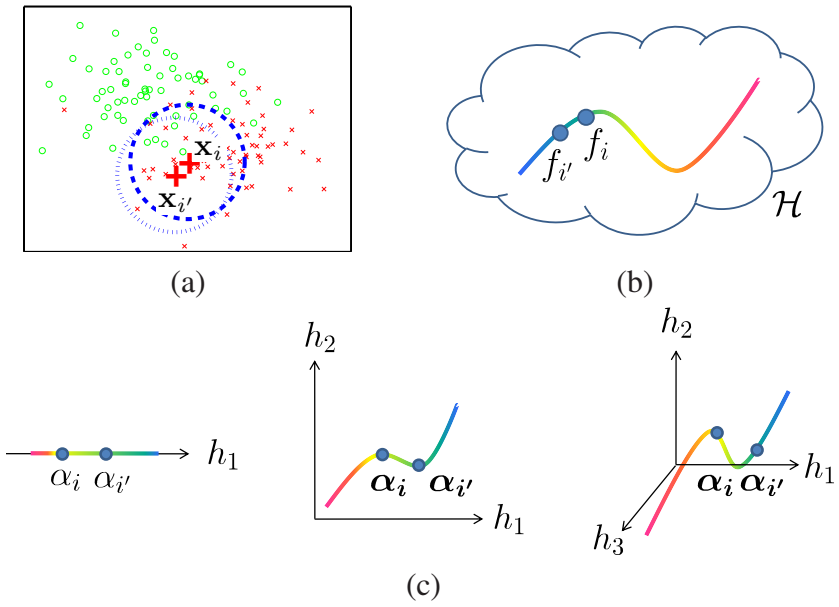



Figure 4.1: (a) Two training samples, $\mathbf{x}_i$ and $\mathbf{x}_{i'}$, and their respective neighborhoods (denoted by blue dotted circles). (b) All the local classifiers, including $f_i$ and $f_{i'}$, spread as a manifold-like structure in the high-dimensional classifier space induced by $\mathcal{H}$. (c) These classifiers can be obtained by incrementally completing the manifold embedding. The embedding space is spanned by weak learners $\{h_t\}$, and the new coordinates of $f_i$ and $f_{i'}$ in the embedding space are their respective ensemble coefficients $\boldsymbol{\alpha}_i$ and $\boldsymbol{\alpha}_{i'}$.

**Proper regularization.** It is instructive to think as if all the local classifiers lie in a parametric space induced by $\mathcal{H}$. The space dimension is $|\mathcal{H}|$, and the coordinates of a classifier (i.e., a point) in the space are its ensemble coefficients over the weak learners. Consider now two local classifiers $f_i$ and $f_{i'}$ corresponding to two nearby samples $\mathbf{x}_i$ and $\mathbf{x}_{i'}$. According to (4.2), the highly overlapping neighborhoods, $\mathbf{w}_i$ and $\mathbf{w}_{i'}$, should lead to the high similarity between $f_i$ and $f_{i'}$. Hence, $f_i$ and $f_{i'}$ are expected to be close in the classifier space. Extending the concept to all the classifiers, it suggests that they would spread as a manifold-like structure. We instill this property into regularizing the training process of each classifier. In other words, we learn all the local classifiers simultaneously by respecting the underlying manifold structure. This could lessen the instability (overfitting) problem when otherwise independently learning each classifier with insufficient training data would cause. Observe that constructing the local classifiers of the

form in (4.3) can capture the idea faithfully. As we will show later, while the ensemble coefficients $\{\alpha_{i,t}\}_{t=1}^{T}$ are to represent the embedding coordinates of $f_i$, the shared weak learners $\{h_t\}_{t=1}^{T}$ can be optimized to span the embedding space. An illustration of the regularization is given in Figure 4.1.

**Redundancy elimination.** If $\{f_i\}_{i=1}^{N}$ are learned independently, redundancy can easily become an unpleasant concern. The phenomenon can be understood from both the aspects of weak learners and training data. Since a weak learner generally yields similar performances in the related tasks, learning each of them separately is to overlook such relatedness. On the other hand, as a training sample can be accessed by multiple tasks, inefficiency can occur when measuring the loss induced by the sample is evaluated independently throughout the relevant tasks. Our strategy is to learn all local classifiers of (4.3) jointly so that information redundancy among them can be reasonably circumvented with a substantial speed-up in the training process.

## 4.4 Multi-task Boosting

The main theme of this section is to detail the steps of the proposed multi-task boosting algorithm, and to discuss its justifications and useful properties.

### 4.4.1 Design of Weak Learners

We consider dyadic hypercuts [66] as the weak learners in that they can achieve good classification performances, and be generated by referencing only the kernel functions $\{k_m\}_{m=1}^{M}$ or matrices $\{K_m\}_{m=1}^{M}$ of (4.1). A dyadic hypercut $h$ is specified by a kernel and a pair of training samples of opposite labels. Specifically, $h$ is parameterized by positive sample $\mathbf{x}_n$, negative sample $\mathbf{x}_{n'}$, and kernel function $k_m$, and can be expressed by

$$h(\mathbf{x}) = sign(k_m(\mathbf{x}_n, \mathbf{x}) - k_m(\mathbf{x}_{n'}, \mathbf{x}) - \theta), \tag{4.4}$$

**Algorithm 2** *Multi-task Boosting for Local Learning*

**Input** : $N$ tasks: task $i$ involves weighted dataset
$$S_i = \{\mathbf{x}_n, y_n, w_{i,n}\}_{n=1}^N.$$
**Output** : Local classifiers $\{f_i\}_{i=1}^N$, where
$$f_i(\mathbf{x}) = \sum_{t=1}^T \alpha_{i,t} h_t(\mathbf{x}).$$
**Initialize:** $w_{i,n}^{(1)} = w_{i,n}$, for $i, n = 1, 2, ..., N$.
**for** $t \leftarrow 1, 2, \ldots, T$ **do**
1. Compute the cross-task data weights $\{\tilde{w}_n^{(t)}\}_{n=1}^N$:
$$\tilde{w}_n^{(t)} = \sum_{i=1}^N w_{i,n}^{(t)}.$$
2. Select the optimal dyadic hypercut $h_t$:
$$h_t = \arg\min_h \sum_{n=1}^N \tilde{w}_n^{(t)} \cdot 1_{[h(\mathbf{x}_n) \neq y_n]}.$$
3. Compute task-wise weighted errors $\{\epsilon_{i,t}\}_{i=1}^N$:
$$\epsilon_{i,t} = \sum_{n=1}^N w_{i,n}^{(t)} \cdot 1_{[h_t(\mathbf{x}_n) \neq y_n]}.$$
4. Compute task-wise weighted accuracies $\{c_{i,t}\}_{i=1}^N$:
$$c_{i,t} = \sum_{n=1}^N w_{i,n}^{(t)} \cdot 1_{[h_t(\mathbf{x}_n) = y_n]}.$$
5. Set task-wise ensemble coefficients $\{\alpha_{i,t}\}_{i=1}^N$:
$$\alpha_{i,t} = \max(0, \tfrac{1}{2} \ln \tfrac{c_{i,t}}{\epsilon_{i,t}}).$$
6. Update data weights $\{w_{i,n}^{(t+1)}\}_{i,n=1}^N$:
$$w_{i,n}^{(t+1)} = w_{i,n}^{(t)} \exp(-y_n \alpha_{i,t} h_t(\mathbf{x}_n)).$$

where $\theta$ is for thresholding. The size of the resulting pool of weak learners is $|\mathcal{H}| = N^+ \times N^- \times M$, where $N^+$ and $N^-$ are the numbers of positive and negative training data. To have an efficient boosting process, we randomly sample a subset of weak learners from the pool at each iteration.

### 4.4.2 The Boosting Algorithm

The multi-task setting of local learning is accomplished via a boosting algorithm, in which task $i$ is to learn classifier $f_i$ as in (4.3) with the weighted dataset $S_i = \{\mathbf{x}_n, y_n, w_{i,n}\}_{n=1}^N$. Steps of the proposed multi-task boosting are listed in Algorithm 2. Note that the algorithm maintains a two-dimensional weight array $\{w_{i,n}^{(t)}\}_{i,n=1}^N$ to link successive iterations,

where $w_{i,n}^{(t)}$ denotes the weight of $\mathbf{x}_n$ in task $i$ at iteration $t$. In what follows, we describe the details of running iteration $t$.

We start by defining the loss function of task $i$. At iteration $t$, $f_i = \sum_{\tau=1}^{t-1} \alpha_{i,\tau} h_\tau$ is a linear combination of the $(t-1)$ selected weak learners. With the *exponential loss* model [35], the loss of $f_i$ with respect to $S_i$ is

$$\mathcal{L}(f_i, S_i) = \sum_{n=1}^{N} w_{i,n} \exp(-y_n f_i(\mathbf{x}_n)). \tag{4.5}$$

Since all the classifiers are trained jointly, a reasonable choice of the joint objective function is the total loss induced by these classifiers in all the tasks, i.e.,

$$\text{Loss} = \sum_{i=1}^{N} \mathcal{L}(f_i, S_i). \tag{4.6}$$

To decide the best weak learner $h_t$ shared by all classifiers at iteration $t$, we minimize the total loss as in (4.6):

$$h_t = \arg\min_h \sum_{i=1}^{N} \mathcal{L}(f_i + h, S_i) \tag{4.7}$$

$$= \arg\min_h \sum_{i=1}^{N} \sum_{n=1}^{N} w_{i,n} \exp(-y_n (f_i(\mathbf{x}_n) + h(\mathbf{x}_n))) \tag{4.8}$$

$$= \arg\min_h \sum_{i=1}^{N} \sum_{n=1}^{N} w_{i,n}^{(t)} \exp(-y_n h(\mathbf{x}_n)) \tag{4.9}$$

$$= \arg\min_h \sum_{n=1}^{N} \tilde{w}_n^{(t)} \exp(-y_n h(\mathbf{x}_n)) \tag{4.10}$$

In (4.9), we have $w_{i,n}^{(t)} = w_{i,n} \exp(-y_n f_i(\mathbf{x}_n))$ from the initialization and step 6 of Algorithm 2. The *cross-task* data weight of $\mathbf{x}_n$ is defined to be its total weight in all tasks,and is denoted by $\tilde{w}_n^{(t)} = \sum_{i=1}^{N} w_{i,n}^{(t)}$. Thus, (4.10) implies that the optimal discrete $h$ is the

one with the minimal cross-task weighted error, i.e.,

$$h_t = \arg\min_h \sum_{n=1}^{N} \tilde{w}_n^{(t)} \cdot 1_{[h(\mathbf{x}_n) \neq y_n]}. \tag{4.11}$$

Once we have $h_t$, the remaining work at iteration $t$ is to determine its task-specific ensemble coefficients $\{\alpha_{i,t}\}_{i=1}^N$. Analogously, the optimal value of $\alpha_{i,t}$ is set to minimize (4.6). We observe that $\alpha_{i,t}$ has influence only on $\mathcal{L}(f_i, S_i)$, and is irrelevant to the loss of other tasks. Thus the optimal value of $\alpha_{i,t}$ can be obtained by setting the first derivative of $\mathcal{L}$ with respect to $\alpha_{i,t}$ to zero. It follows that

$$\alpha_{i,t} = \frac{1}{2} \ln \frac{c_{i,t}}{\epsilon_{i,t}}, \text{ where} \tag{4.12}$$

$$\epsilon_{i,t} = \sum_{n=1}^{N} w_{i,n}^{(t)} \cdot 1_{[h_t(\mathbf{x}_n) \neq y_n]}, \quad c_{i,t} = \sum_{n=1}^{N} w_{i,n}^{(t)} \cdot 1_{[h_t(\mathbf{x}_n) = y_n]}.$$

Note that the resulting $\alpha_{i,t}$ is not guaranteed to be positive. However, we allow only nonnegative combinations of weak learners. Thus $\alpha_{i,t}$ will be set to zero if it is negative. Finally, iteration $t$ is completed by updating the data weights as in step 6 of Algorithm 2.

**Novel sample prediction.** In the testing phase, given a new sample $\mathbf{z}$, the proposed technique will first find its nearest training sample, say, $\mathbf{x}_i$, and then use local classifier $f_i$ to predict the label of $\mathbf{z}$. Since $\mathbf{z}$ belongs to the neighborhood of $\mathbf{x}_i$, such a tactic is justified by that $f_i$ is optimized to give good classification performances around $\mathbf{x}_i$.

### 4.4.3 Useful Properties

The proposed multi-task boosting is simple, easy to implement, and has theoretic merit. At each iteration, it picks the optimal weak learner and computes its ensemble coefficients by directly minimizing the total exponential loss as in (4.6). Although the selected weak learners are shared cross tasks, it can be readily verified that the exponential loss of each

task is monotonically decreased in training. This property guarantees the convergence of our algorithm.

Regarding the gain efficiency of leaning all the local classifiers, we elucidate this point with two aspects of considerations. First, observe that according to (4.11), no matter how many tasks sample $\mathbf{x}_n$ is associated with, evaluating whether $\mathbf{x}_n$ is misclassified by some weak learner $h$ is performed only once. Second, each selected weak learner is shared cross all the tasks, and the degree of sharing in these tasks is adaptively controlled by the ensemble coefficients. It is in this aspect that the proposed approach has the advantage over other related work, e.g., *JointBoost* [91], where jointly training would quickly become infeasible due to an exponential growth in the number of search hypotheses as the size of training dataset increases.

As is noted before, the algorithm maintains a 2-D weight array $\{w_{i,n}\}$ throughout the boosting iterations. It records not only the difficulties of training data but also those of the tasks. With iterative re-weighting, high weights will be gradually distributed to the difficult data and tasks, and results in more emphases on them. The strategy ensures that all learning tasks will be appropriately addressed.

One final remark about the multi-task boosting is that it iteratively learns classifiers by fusing information from multiple kernels and thus inherently leads to an on-line and incremental way to perform multiple kernel learning (MKL). In addition, compared with conventional optimization techniques like *semi-definite programming* (SDP) [96], it does scale better when the size of data is increased.

## 4.5 Experimental results

To evaluate the performances of the proposed method, we carry out experiments of object recognition on two benchmark databases, Caltech-101 [33] and VOC 2007 [29]. Both the two datasets contain images with multiple class labels, and are complicated by large intr-

aclass variations. Therefore, they serve as good test beds for justifying the effectiveness of our approach to local learning.

### 4.5.1 Caltech-101

We implement the *one-against-all* rule for our approach to handle multi-class recognition on Caltech-101. In the following, we respectively describe the image data, the adopted descriptors, the experiment setting and the results.

#### Dataset

The Caltech-101 dataset contains 101 object categories and one additional category of background images. Each category consists of $31 \sim 800$ images. Although the object is positioned near the central region of each image, the large class number and intraclass variations still result in a challenging recognition task. Since the resolutions of images in the dataset are different, we resize them to around $60,000$ pixels before performing feature extraction.

#### Descriptors and Kernels

To enrich the set of weak learners, several image descriptors and their associated distance function are used to extract diverse image features. From (4.1), the resulting kernels are

- **GB-Dist**: For a given image, the *geometric blur* descriptor [6] is applied to each of 400 randomly sampled edge pixel. Coupling with the distance function suggested in (2) of [113], the kernel is constructed.
- **GB**: The same as GB-Dist, except the geometric distortion term is excluded in evaluating the distance.
- **SIFT-Dist**: The same as GB-Dist, except *SIFT* descriptor [63] is used instead.

- **SIFT-SPM**: With a densely sampled grid, SIFT features are extracted and quantized into visual words. The kernel is built by matching *spatial pyramids* [58].
- **SS-SPM / SS-Dist**: The same as SIFT-SPM and SIFT-Dist respectively, except the *self-similarity* descriptor [82] is used to replace the SIFT descriptor.
- **C2-SWP / C2-ML**: We adopt biologically inspired features to depict images. Both the *C2* features suggested by Serre et al. [81] and by Mutch and Lowe [67] are respectively used to establish the two RBF kernels.
- **PHOG**: The *PHOG* descriptor [9] is used to summarize the distributions of edge orientations. Together with the $\chi^2$ distance, the kernel is established.
- **GIST**: The images are resized to $128 \times 128$ pixels prior to applying the *gist* descriptor [70]. The RBF kernel is then constructed with the $L^2$ distance.

The distance matrices result from different parameter values of each descriptor are combined in advance. It is used for ensuring that the resulting kernels individually reach their best performances.

### Quantitative Results

Like in [6, 98, 113], we randomly pick $30$ images from each of the $102$ categories, and split them into two disjoint subsets: one contains $N_{train}$ images per category, and the other consists of the rest. The two subsets are respectively used for training and testing. The whole evaluation process are repeated $20$ times by using different splits between the training and testing subsets. Recognition rates are measured in cases that $N_{train}$ is set to $5$, $10$, $15$, $20$, and $25$.

We first investigate the effectiveness of each kernel by comparing our method with the *one-nearest-neighbor* (1-NN) classifier, SVM, and AdaBoost. When the value of $N_{train}$ is $15$, the recognition rates are reported in Table 4.1a. The 1-NN serves as the baseline

Table 4.1: Recognition rates [mean $\pm$ std %] of several approaches on the Caltech-101 dataset. (a) A kernel is taken into account at a time. (b) All kernels are jointly considered.

| | 1-NN | SVM | AdaBoost | Ours |
|---|---|---|---|---|
| GB-Dist | $42.4 \pm 1.3$ | $61.4 \pm 0.8$ | $61.5 \pm 0.7$ | **63.2 ± 0.8** |
| GB | $37.4 \pm 0.9$ | $57.6 \pm 1.0$ | $58.5 \pm 0.7$ | **59.7 ± 1.2** |
| SIFT-Dist | $49.6 \pm 0.8$ | **58.7 ± 0.9** | $57.5 \pm 1.0$ | $57.8 \pm 0.7$ |
| SIFT-SPM | $48.8 \pm 0.7$ | **56.1 ± 0.9** | $53.3 \pm 0.8$ | $55.2 \pm 0.7$ |
| SS-Dist | $31.7 \pm 1.4$ | $53.4 \pm 1.0$ | $56.1 \pm 0.7$ | **56.3 ± 0.9** |
| SS-SPM | $41.7 \pm 0.9$ | $53.9 \pm 0.9$ | $55.5 \pm 0.7$ | **56.9 ± 1.3** |
| C2-SWP | $22.0 \pm 0.8$ | **28.3 ± 0.8** | $26.0 \pm 0.9$ | $26.9 \pm 1.0$ |
| C2-ML | $37.7 \pm 0.7$ | $46.3 \pm 1.0$ | $44.8 \pm 0.9$ | **46.6 ± 0.6** |
| PHOG | $27.7 \pm 1.0$ | **43.9 ± 0.8** | $41.3 \pm 1.0$ | $42.9 \pm 0.8$ |
| GIST | $36.8 \pm 1.1$ | $48.7 \pm 0.8$ | $49.1 \pm 1.0$ | **51.2 ± 0.8** |

(a)

| | SimpleMKL | AdaBoost (local) | Ours |
|---|---|---|---|
| All | $74.3 \pm 1.2$ | $74.6 \pm 1.3$ | **75.8 ± 1.1** |
| | $3.26 \times 10^2$ sec. | $1.87 \times 10^5$ sec. | $3.92 \times 10^3$ sec. |

(b)

for the experiment. By transforming a kernel to a set of dyadic hypercuts (weak learners), AdaBoost and our approach can fuse these hypercuts into a global classifier and a set of local classifiers, respectively. Observe that AdaBoost achieves comparable performance to that of SVM. It implies that the hypercuts can capture useful information in a kernel. Overall, the proposed local learning consistently outperforms AdaBoost, and gives the best recognition rates in most cases.

Focusing now on the case that multiple kernels are considered simultaneously, we evaluate the performances of *SimpleMKL* [74], AdaBoost for local learning, and our method. SimpleMKL is a well-known software for multiple kernel learning, and can learn an optimal ensemble kernel by searching the convex combinations of kernels. With SimpleMKL, a global classifier is obtained with training time $326$ seconds, and achieves recognition rate $74.3\%$. We then compare AdaBoost and our approach in realizing local learning. While AdaBoost carries out local learning for a total of $N = 1530$ classifiers one by one, our method constructs them via a joint training. The recognition rates and

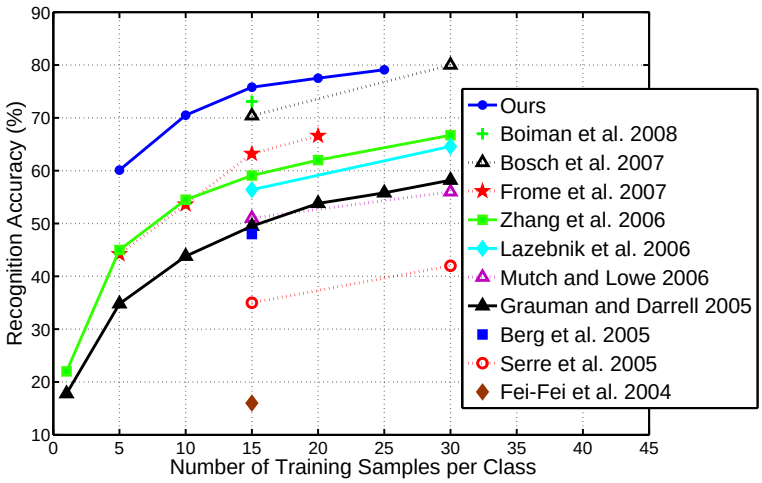



Figure 4.2: The recognition rates of several approaches on Caltech-101 dataset with different numbers of training data per class.

computational costs are shown in Table 4.1b. The results demonstrate that our approach achieves not only a better performance but also gives a two order speed-up.

In Figure 4.2, the accuracy rates of several recent techniques, including ours, on Caltech-101 are plotted with respect to different numbers of training data. The outcomes of ours are $60.1 \pm 1.1\%$, $70.5 \pm 0.7\%$, $77.5 \pm 0.7\%$, and $79.1 \pm 2.1\%$ when $N_{train}$ is set to 5, 10, 20, and 25 respectively. For the case that $N_{train}$ is 15, the recognition rate $75.8\%$ by our method is either better or comparable to those by other published systems, e.g., [7, 9, 40, 44, 58, 113].

### 4.5.2 Pascal VOC 2007

The second set of our experiments is performed on the dataset provided by the VOC 2007 classification challenge, which serves as a benchmark for comparing image classification methods in dealing with realistic scenes.

### Dataset

The VOC 2007 dataset contains 20 object categories, including indoor objects, vehicles, animals, and people. Each category consists of $195 \sim 4192$ images, and most of them are of similar sizes. Unlike Caltech-101, objects in the images are neither centrally located nor with similar scales. Besides, more than one kind of objects can be present in an image. The overall learning task contains 20 binary classification problems, each of which is to predict the presence/abcense of objects from a certain category.

### Descriptors and Kernels

We use 18 distance matrices generated from the combinations of six kinds of descriptors and three kinds of spatial pyramids. The image descriptors are

- **SIFT / GB / SS**: We extract the SIFT features from a densely sampled grid over multiple scales, and quantize them into 4000 visual words. The $\chi^2$ distance is adopted as the distance function. Following the same procedure, we build the other two using the geometric blur and self-similarity features.
- **TC-SIFT**: The same as SIFT, except that SIFT features are computed over three normalized RGB channels separately [94].
- **GIST**: All images are resized to resolution $128 \times 128$ before performing feature extraction. Then the $L^2$ distance is used as the distance function.
- **C2-ML**: This is constructed with the C2 feature by Mutch and Lowe [67], and the $L^2$ distance function.

Three kinds of spatial pyramids, $1 \times 1$ (whole image), $2 \times 2$ (image quarters) [58], and $1 \times 3$ (horizontal bars), are considered to exploit the spatial information. Coupling the descriptors and the spatial pyramids yields 18 distance matrices. However, using all the distance matrices will lead to large memory consumption, and it can be resolved by using

Table 4.2: Average APs (%) of several approaches on Train+Val set of the VOC 2007 dataset.

| | Train | | Train+Val | |
|---|---|---|---|---|
| | SVM | Ours | SVM | Ours |
| SIFT | 46.3 | **47.5** | 50.7 | **51.4** |
| TC-SIFT | 46.8 | **46.9** | 51.1 | 51.1 |
| SS | 48.4 | **48.8** | **52.4** | 52.1 |
| GB | **45.4** | 42.8 | **47.5** | 46.7 |
| Gist | 38.6 | **39.4** | 43.1 | **44.3** |
| C2-ML | 34.6 | **36.2** | **39.8** | 39.6 |
| | SimpleMKL | Ours | SimpleMKL | Ours |
| All | 51.4 | **55.2** | 57.3 | **59.3** |

only one average distance matrix for each descriptor. To address the possibly large variations due to using different distance measures, the distance matrices are normalized by dividing their respective standard deviation. After generating the distance matrices, the hyper-parameters $\gamma_m$ in (4.1) are tuned to form kernels that achieve best performances.

### Quantitative Results

The VOC 2007 dataset provides two training sets: the larger one (Train+Val) contains 5011 images, and the other (Train) has 2501 images. We experiment on both training sets and use the same test set, which consists of 4952 images, for performance evaluation.

For each category, one needs to predict if there exists at least one object of that class in a test image. Each prediction should be a real value to reflect the confidence of object presence. The precision-recall curve is built on the prediction values of all test images, and the performance is measured by the *average precision* (AP), which is proportional to the area under the precision-recall curve. Finally the average of 20 APs is used for the comparison.

In Table 4.2, we report the results of comparing our method with SVM and SimpleMKL. When training with one individual kernel at a time, our method performs slightly better than SVM on the Train set, and comparably as SVM on the Train+Val set. It shows

Table 4.3: Average APs (%) for the VOC 2007 dataset.

| | avg. | Aero. | Bicy. | Bird | Boat | Bott. | Bus | Car | Cat | Chai. | Cow | Tabl. | Dog | Hors. | Moto. | Pers. | Plan. | Shee. | Sofa | Trai. | Tv |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INRIA | 59.4 | 77.5 | 63.6 | 56.1 | **71.9** | **33.1** | 60.6 | 78.0 | **58.8** | 53.5 | 42.6 | 54.9 | 45.8 | 77.5 | 64.0 | 85.9 | 36.3 | 44.7 | 50.6 | 79.2 | 53.2 |
| XRCE | 57.5 | 72.3 | 57.5 | 53.2 | 68.9 | 28.5 | 57.5 | 75.4 | 50.3 | 52.2 | 39.0 | 46.8 | 45.3 | 75.7 | 58.5 | 84.0 | 32.6 | 39.7 | 50.9 | 75.1 | 49.5 |
| TKK | 51.7 | 71.4 | 51.7 | 48.5 | 63.4 | 27.3 | 49.9 | 70.1 | 51.2 | 51.7 | 32.3 | 46.3 | 41.5 | 72.6 | 60.2 | 82.2 | 31.7 | 30.1 | 39.2 | 71.1 | 41.0 |
| van Gemert et al. [95] | **60.5** | **80.4** | **64.9** | **57.0** | 69.1 | 24.6 | **65.8** | **78.2** | 54.3 | **56.9** | 42.4 | 53.7 | **47.0** | **81.5** | 65.6 | **87.9** | **38.3** | **52.3** | 53.9 | **83.2** | 53.3 |
| SimpleMKL | 57.3 | 74.1 | 62.7 | 48.7 | 66.9 | 29.1 | 62.6 | 75.0 | 56.9 | 54.5 | 42.7 | 54.8 | 44.2 | 76.3 | **65.8** | 83.6 | 28.7 | 42.5 | 51.5 | 74.7 | 50.9 |
| Ours | 59.3 | 76.5 | 64.6 | 51.8 | 68.3 | 32.2 | 61.3 | 77.5 | 57.8 | 56.3 | **43.5** | **58.8** | 44.8 | 78.4 | 65.2 | 85.4 | 30.4 | 47.7 | **54.6** | 76.4 | **54.6** |

that the learned local models give the same or better results than a global one does. When learning with multiple kernels, our method achieves average AP $59.3\%$ on the Train+Val set, and yields significant improvements to those learned on each individual kernel. The performance gain in the classification accuracy supports that the use of various distances can complement each other and the concept of local models can better capture the structures of complex data. When SimpleMKL is applied to the same kernels to learn a global model, the resulting average AP is $57.3\%$.

Table 4.3 summarizes the per-class results on the Train+Val set of our method, SimpleMKL, the top three (denoted respectively as INRIA, XRCE, and TKK) in VOC Challenge $2007$ [29], and van Gemert et al. [95], which has produced by far the best results for the dataset. Our method achieves average AP $59.3\%$, and performs the best in $4$ out of $20$ categories. The performance by the proposed approach is consistently better than that of SimpleMKL, while it is also comparable with that by INRIA and meanwhile falls slightly behind the average AP $60.5\%$ reported in [95].

## 4.6 Summary

We have introduced an efficient local learning approach to resolving the difficulties in object recognition caused by large intraclass variations. We cast multiple, independent training processes of local classifiers to a correlative multi-task learning problem, and develop a boosting-based algorithm to accomplish it. The proposed technique is comprehensively evaluated with two benchmark datasets. The recognition rates in both datasets are com-

parable to those yielded by the respective state-of-the-art approaches. Our method can be considered as a general multi-task learning tool for vision applications where multiple correlative classifiers are required, such as multi-view face detection, multi-part object tracking, or user-dependent media ranking. The framework also provides a new way to carry out multiple kernel learning in an incremental manner.

Chapter 5

# Group-dependent Multiple Kernel Learning for Image Clustering

In this chapter, we address the problem of clustering data with multiple data representations and consider *cluster-dependent feature selection* via multiple kernel learning. That is, for each cluster we select features across heterogeneous data descriptors such that data of the cluster are coherent to each other, and distinct from the rest. The major difficulty of this work results from the unsupervised nature of clustering: We have no labeled data to guide the searching of the optimal ensemble kernel over a given convex set of base kernels. Our key idea for handling the difficulty is to cast the tasks of supervised feature selection and unsupervised clustering procedure into a joint optimization problem, and the two tasks can then be carried out collaboratively. Specifically, each cluster is associated with a classifier, which implements multiple kernel learning in a boosting way. We integrate the supervised classifier training processes into the unsupervised or semi-supervised clustering procedure. The proposed approach is applied to visual object categorization and face image grouping. The promising experimental results demonstrate the advantages of our approach over the existing, classic clustering algorithms.

## 5.1 Motivation

Clustering is a technique to partition the data into groups so that *similar* (or *coherent*) objects and their properties can be readily identified and exploited. While such a goal is explicit and clear, the notion of *similarity* is often not well defined, partly due to the lack of a universally applicable *similarity measure*. As a result, previous research efforts on developing clustering algorithms mostly focus on dealing with different scenarios or specific applications. In the field of vision research, performing data clustering is essential in addressing various tasks such as object categorization [27, 93] or image segmentation [20, 83]. Despite the great applicability, a fundamental difficulty hindering the advance of clustering techniques is that the intrinsic cluster structure is not evidently revealed in the feature representation of complex data. Namely, the resulting similarities among data points do not faithfully reflect their true relationships.

We are thus motivated to consider establishing a clustering framework with the flexibility of allowing the data to be characterized by multiple descriptors. The generalization aims to bridge the gap between the resulting data similarities and their underlying relationships. Take, for example,the images shown in Figure 5.1. Without any ambiguities, one can easily divide them into three clusters. Nevertheless, say, in an object recognition system, color related features are required to separate the images in category `sunset` from the others. Analogously, shape and texture based features are respectively needed for describing categories `bicycle` and `jaguar`. This example not only illustrates the importance of using multiple features but also points out that the optimal features for ensuring each cluster coherent often vary from cluster to cluster.

The other concept critical to our approach is *unsupervised feature selection*. It is challenging due to the absence of data labels to guide the relevance search, e.g., [76, 112]. To take account of the use of multiple descriptors for clustering, our formulation generalizes unsupervised feature selection to its *cluster/group-dependent* and *cross feature space*

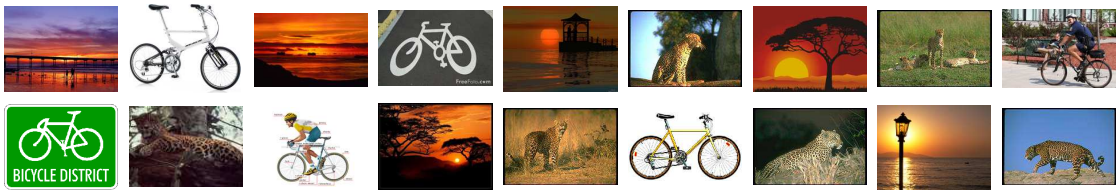



Figure 5.1: Images from three different categories: `sunset`, `bicycle` and `jaguar`

extensions. To that end, each cluster is associated with a classifier learned with multiple kernels to give a good separation between data inside and outside the cluster, and data are dynamically assigned to appropriate clusters through the progressive learning processes of these cluster-specific classifiers. Iteratively, the learned classifiers are expected to facilitate the revealing of the intrinsic cluster structure, while the progressively improved clustering results would provide more reliable data labels in learning the classifiers.

Specifically, we integrate the multiple, correlative training processes of the cluster-specific classifiers into the clustering procedure, and realize the unified formulation by 1) proposing a general constrained optimization problem that can accommodate both fully unlabeled and partially labeled datasets; and 2) implementing *multiple kernel learning* [57] in a *boosting* way to construct the cluster-specific classifiers. Prior knowledge can thus be conveniently exploited in choosing a proper set of visual features of diverse forms to more precisely depict the data. Indeed our approach provides a new perspective of applying multiple kernel learning, which typically addresses supervised applications, to both unsupervised and semisupervised ones. Such a generalization is novel in the field. Different from other existing clustering techniques, our method can not only achieve better clustering results but also have access to the information regarding the commonly shared features in each cluster.

## 5.2 Literature Review

As the relevant literature about clustering is quite extensive, our survey instead emphasizes clustering algorithms and techniques closely related to the proposed framework.

### 5.2.1 Clustering

Techniques on clustering can vary considerably in many aspects, including assuming particular principles for data grouping, making different assumptions about cluster shapes or structures, and using various optimization techniques for problem solving. Such variations however do not devalue the importance of clustering being a fundamental tool for unsupervised learning. Instead, clustering methods such as $k$-*means*, *spectral clustering* [68, 83], *mean shift* [20] or *affinity propagation* [36] are constantly applied in more effectively solving a broad range of computer vision problems.

It is possible to overcome the unsupervised nature of clustering by incorporating a small amount of labeled data in the procedure so that satisfactory results can be achieved, especially in complex tasks. For example, Xing et al. [105] impose *side information* for metric learning to facilitate clustering, while Kulis et al. [54] and Tuzel et al. [93] utilize pairwise constraints to perform semisupervised clustering in a kernel induced feature space.

### 5.2.2 Clustering with Feature Selection

Although most clustering algorithms are developed with theoretic support, their performances still depend critically on the feature representation of data. Previous approaches, e.g., [76, 112], concerning the limitation have thus suggested to perform clustering and feature selection simultaneously such that relevant features are emphasized. Due to the inherent difficulty of unsupervised feature selection, methods of this category often proceed in an iterative manner, namely, the steps of feature selection and clustering are carried out

alternately.

Feature selection can also be done cluster-wise, say, by imposing the *Gaussian mixture models* on the data distribution, or by learning a distance function for each cluster via re-weighting feature dimensions such as in [17, 25]. However, these methods typically assume that the underlying data are in a single feature space and in form of vectors. The restriction may reduce the overall effectiveness when the data of interest can be more precisely characterized by considering multiple descriptors and diverse forms, e.g., bag-of-features [5, 63] or pyramids [8, 58].

Xu et al. [106] instead consider the large margin principle for measuring how good a data partitioning is. Their method first maps the data into the kernel-induced space, and seeks the data labeling (clustering) with which the maximum margin can be obtained by applying SVMs to the then labeled data. Subsequently, Zhao et al. [114] introduce a cutting-plane algorithm to generalize the framework of maximum margin clustering from binary-class to multi-class.

### 5.2.3 Ensemble Clustering

The technique of *cluster ensembles* by Strehl and Ghosh [87] is most relevant to our approach. It provides a useful mechanism for combining multiple clustering results. The ensemble partitioning is optimized such that it shares as much information with each of the elementary ones as possible. Fred and Jain [34] introduce the concept of *evidence accumulation* to merge various clusterings to a single one via a voting scheme. These methods generally achieve better clustering performances. Implicitly, they also provide a way for clustering data with multiple feature representations: One could generate an elementary clustering result for each data representation, and combine them into an ensemble one. However, the obtained partitioning is optimized in a global fashion, neglecting the fact the optimal features are often cluster-dependent.

## 5.3 Problem Definition

We formalize and justify the proposed clustering technique in this section. Prior to that, we need to specify the notations adopted in the formulation.

### 5.3.1 Notations

Given a dataset $D = \{\mathbf{x}_i\}_{i=1}^{N}$, our goal is to partition $D$ into $C$ clusters. We shall use a *partition matrix*, $Y = [y_{ic}] \in \{0,1\}^{N\times C}$, to represent the clustering result, where $y_{ic} = 1$ indicates that $\mathbf{x}_i$ belongs to the $c$th cluster, otherwise $y_{ic} = 0$. Besides, let $\mathbf{y}_{i,:}$ and $\mathbf{y}_{:,c}$ denote the $i$th row and $c$th column of $Y$ respectively.

To tackle complex clustering tasks, we consider the use of multiple descriptors to more precisely characterize the data. These descriptors may result in diverse forms of feature representations, such as vectors [67], bags of features [113], or pyramids [58]. To avoid directly working with these varieties, we adopt a strategy similar to that in [8, 98], where kernel matrices are used to provide a uniform representation for data under various descriptors. Specifically, suppose $M$ kinds of descriptors are employed to depict each sample, i.e., $\mathbf{x}_i = \{\mathbf{x}_{i,m} \in \mathcal{X}_m\}_{m=1}^{M}$. For each descriptor $m$, a non-negative distance function $d_m : \mathcal{X}_m \times \mathcal{X}_m \to \mathbb{R}$ is associated. The corresponding kernel matrix $K_m$ and kernel function $k_m$ can be established by

$$K_m(i,j) = k_m(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma_m d_m^2(\mathbf{x}_{i,m}, \mathbf{x}_{j,m})\right), \tag{5.1}$$

where $\gamma_m$ is a positive constant. By applying the procedure to each descriptor, a kernel bank of size $M$ is obtained, i.e., $\Omega = \{K_m\}_{m=1}^{M}$. The kernel bank will serve as the information bottleneck in the sense that data access is restricted to referencing only the $M$ kernels. This way our method can conveniently work with various descriptors without worrying about their diversities.

### 5.3.2 Formulation

The idea of improving clustering performances for complex data is motivated by the observation that the optimal features for grouping are often cluster-dependent. Our formulation associates each cluster with a classifier to *best* interpret the relationships among data and the cluster. Specifically, a cluster-specific classifier is designed to divide the data so that its members would share certain common features, which are generally distinct from the rest. Furthermore, the *goodness* of the clustering quality about a resulting cluster can be explicitly measured by the induced loss (namely, the *degree* of difficulty) in learning the specific classifier. It follows that the proposed clustering seeks an optimal data partitioning with the minimal total loss in jointly learning all the $C$ cluster-specific classifiers.

As one may notice that our discussion so far would lead to a cause-and-effect dilemma: While the data labels are required in learning the cluster-specific classifiers, they in turn can only be determined through the clustering results implied by these classifiers. We resolve this difficulty by incorporating the learning processes of the cluster-specific classifiers into the clustering procedure, and cast the task as the following constrained optimization problem:

$$\min_{Y,\{f_c\}_{c=1}^{C}} \quad \sum_{c=1}^{C} \text{Loss}(f_c, \{\mathbf{x}_i, y_{ic}\}_{i=1}^{N}) \tag{5.2}$$

$$\text{subject to} \quad Y \in \{0,1\}^{N\times C}, \tag{5.3}$$

$$\mathbf{y}_{i,:}\mathbf{e}_C = 1, \text{ for } i = 1, 2, ..., N, \tag{5.4}$$

$$\ell \leq \mathbf{e}_N^{\top}\mathbf{y}_{:,c} \leq u, \text{ for } c = 1, 2, ..., C, \tag{5.5}$$

$$\mathbf{y}_{i,:} = \mathbf{y}_{j,:}, \text{ if } (i,j) \in S, \tag{5.6}$$

$$\mathbf{y}_{i,:} \neq \mathbf{y}_{j,:}, \text{ if } (i,j) \in S', \tag{5.7}$$

where $\{f_c\}_{c=1}^{C}$ are the cluster-specific classifiers. $\mathbf{e}_C$ and $\mathbf{e}_N$ are column vectors, whose

elements are all one, of dimensions $C$ and $N$ respectively.

We now give justifications for the above constrained optimization problem. Our discussions focus first on the part of constraints. With (5.3) and (5.4), $Y$ is guaranteed to be a valid partition matrix. Since in practical applications most clusters are rarely of extreme sizes, we impose the desired upper bound $u$ and lower bound $\ell$ of the cluster size in (5.5). The remaining constraints (5.6) and (5.7) are optional so that our method can be extended to address semisupervised learning. In that case, (5.6) and (5.7) would provide a set of pairwise instance-level constraints, each of which specifies either a pair of data points must reside in the same cluster or not. $S$ in (5.6) and $S'$ in (5.7) are respectively used to denote the collections of these *must-links* and *cannot-links*.

Assuming that all the constraints are satisfied, the formulation would look for optimal data partitioning $Y^*$ such that, according to (5.2), the total induced loss of all the cluster-specific classifiers is minimized. That is, the proposed clustering approach would prefer that data residing in each cluster are well separated from the rest by the cluster-specific classifier (and hence yields a small loss), which is derived by coupling a discriminant function with an optimal feature selection to achieve the desired property. This implies that most of the data in an arbitrary cluster $c$ would share some coherent characteristics implicitly defined by the optimal feature selection in forming $f_c^*$. The proposed optimization elegantly connects the unsupervised clustering procedure with the supervised learning of the specific classifiers. By jointly addressing the two tasks, our method can uncover a reasonable cluster structure even for a complex dataset.

## 5.4 Optimization Procedure

To deal with the cause-and-effect factor in (5.2), we consider an iterative strategy to solve the constrained optimization problem. At each iteration, the cluster-specific classifiers $\{f_c\}_{c=1}^C$ and the partition matrix $Y$ are alternately optimized. More specifically, $\{f_c\}$ are

first optimized while $Y$ is fixed, and then their roles are switched. The iterations are repeated until the loss cannot be further reduced.

### 5.4.1 On Learning Cluster-specific Classifiers

Notice that in the constrained optimization problem (5.2) the cluster-specific classifiers $\{f_c\}$ only appear in the objective function, and there is no correlation among them once $Y$ is fixed. Thus, these classifiers can be optimized independently by minimizing their corresponding loss function. That is, $f_c$ can be derived by considering only the binary labeled data $\{\mathbf{x}_i, y_{ic}\}_{i=1}^N$.

Our choice of selecting a suitable supervised learning methodology for constructing the classifiers is based on two key requirements stemming from the properties related to the classifiers and the iterative training process. First, the cluster-specific classifiers should be generated by using information from multiple kernels, i.e., via multiple kernel learning [57, 98]. Second, the *degree of data fitting* in the classifiers can be conveniently controlled. The latter requirement arises due to the expected phenomenon that the data labels $\{y_{ic}\}_{i=1}^N$ would be noisy during the earlier iterations, and then progressively become more accurate through the iterative optimization. By addressing this effect, we can significantly alleviate the possibility of overfitting or underfitting in learning the classifiers. Having taken the two into account, we consider each $f_c$ a boosting classifier. In what follows, we describe the two main elements in learning such classifiers.

#### The Pool of Weak Learners

We adopt a similar strategy proposed in [62]. To begin with, the discriminant power of each kernel is transferred into a set of weak learners, called *dyadic hypercuts* [66]. We then construct the pool of weak learners by including the dyadic hypercuts generated from all the kernels in $\Omega$. The procedure naturally enables a boosting algorithm to learn

classifiers that inherit the discriminant power from the multiple kernels.

A dyadic hypercut $h$ is specified by three parameters: a positive sample $\mathbf{x}_p$, a negative sample $\mathbf{x}_n$, and a kernel function $k_m$. (Note that the positive and negative samples here depend on labels $\{y_{ic}\}_{i=1}^N$.) The model for prediction is

$$h(\mathbf{x}) = a \cdot \text{sign}(k_m(\mathbf{x}_p, \mathbf{x}) - k_m(\mathbf{x}_n, \mathbf{x}) - \theta) + b, \tag{5.8}$$

where $a$ and $b$ are real values, and $\theta$ is used for thresholding. The size of the set of weak learners is $|\mathcal{H}| = N^+ \times N^- \times M$, where $N^+$ ($N^-$) is the number of positive (negative) training data, and $M$ is the number of kernels.

**Loss Function for Boosting**

Among the many choices of loss function for learning boosting classifiers, we have implemented two of the most popular ones, i.e., *ExpLoss* and *LogLoss* [19, 38], to test our method. In our experiments, *LogLoss* leads to better performances, and is thus adopted. It follows that in (5.2) we have

$$\text{Loss}(f_c, \{\mathbf{x}_i, y_{ic}\}_{i=1}^N) = \sum_{i=1}^N \ln\left(1 + \exp\left(-\tilde{y}_{ic} f_c(\mathbf{x}_i)\right)\right), \tag{5.9}$$

where $\tilde{y}_{ic} = 2y_{ic} - 1$ is to convert a binary label $y_{ic} \in \{0, 1\}$ in partition matrices to $\tilde{y}_{ic} \in \{-1, 1\}$ for boosting models. With the pool of weak learners generated from the kernel bank $\Omega$ and the loss function (5.9), all cluster-specific classifiers $\{f_c\}$ can be learned one by one via *LogitBoost* [38].

### 5.4.2 On Assigning Data into Clusters

Once the cluster-specific classifiers are fixed, we illustrate that how the partition matrix $Y$ in (5.2) can be optimized by *binary integer programming* (BIP) [103]. For the ease of

our discussion, the canonical form of a BIP problem is given below

$$\min_{\mathbf{z}} \quad \mathbf{d}^\top \mathbf{z} \tag{5.10}$$

$$\text{subject to} \quad A\mathbf{z} \leq \mathbf{b} \text{ and } A_{eq}\mathbf{z} = \mathbf{b}_{eq}, \tag{5.11}$$

$$z_i \in \{0, 1\}. \tag{5.12}$$

It suffices to show the proposed constrained optimization can be transformed to the above form. To rewrite the objective function (5.2) as the inner product (5.10), we let $\mathbf{z} \equiv vec(Y) = [y_{11} \ \cdots y_{1C} \cdots \ y_{ic} \ \cdots \ y_{NC}]^\top$, the *vectorization* of partition matrix $Y$ and set the column vector $\mathbf{d} = [d_{ic}]$ as

$$d_{ic} = \ln\left(1 + \exp\left(-f_c(\mathbf{x}_i)\right)\right) + \sum_{c'=1 \ \& \ c' \neq c}^{C} \ln\left(1 + \exp\left(f_{c'}(\mathbf{x}_i)\right)\right). \tag{5.13}$$

The definitions of $\mathbf{d}$ and $\mathbf{z}$ would lead to

$$\mathbf{d}^\top \mathbf{z} = \mathbf{d}^\top vec(Y) = \sum_{c=1}^{C} \sum_{i=1}^{N} \ln\left(1 + \exp\left(-\tilde{y}_{ic} f_c(\mathbf{x}_i)\right)\right). \tag{5.14}$$

Indeed the derivation of (5.14) is based on (5.4). For each sample $\mathbf{x}_i$, there is one and only one element whose value is 1 in the vector $\mathbf{y}_{i,:} = [y_{i1} \ \cdots \ y_{iC}]$. And no matter which element equals to 1, we have

$$\sum_{c=1}^{C} d_{ic} y_{ic} = \sum_{c=1}^{C} \ln\left(1 + \exp\left(-\tilde{y}_{ic} f_c(\mathbf{x}_i)\right)\right). \tag{5.15}$$

Now, summing over all the data on the both sides of (5.15) gives (5.14). We are left to express the constraints (5.3)–(5.7) into (5.11) and (5.12). Since the derivations related to (5.3)–(5.6) are straightforward, we focus on the reduction of constraint (5.7). To represent $\mathbf{y}_{i,:} \neq \mathbf{y}_{j,:}$, we consider additional auxiliary variables, $\mathbf{p} \in \{0, 1\}^{C \times 1}$ and $\mathbf{q} \in \{0, 1\}^{C \times 1}$,

and the following three constraints

$$\mathbf{y}_{i,:} - \mathbf{y}_{j,:} = \mathbf{p} - \mathbf{q},\ \mathbf{p} + \mathbf{q} \leq \mathbf{e}_C,\ \text{and}\ \mathbf{e}_C^\top \mathbf{p} + \mathbf{e}_C^\top \mathbf{q} = 2. \tag{5.16}$$

It can be verified that $\mathbf{y}_{i,:} \neq \mathbf{y}_{j,:}$ if and only if the constraints in (5.16), which are all conformed to (5.11), hold. Thus, our discussion justifies that when $\{f_c\}$ are fixed, the constrained optimization problem (5.2) can be effectively solved by BIP to obtain a new data partitioning $Y$.

### 5.4.3 Implementation Details

In solving the constrained optimization, we begin by providing an initial $Y$ derived by randomly splitting the data into clusters of similar sizes. As it turns out the proposed optimization procedure is robust against different initializations, and converges fast. (Further details will be discussed in the next section.)

It is useful to progressively adjust the data fitting power in learning the classifiers, since the reliability of the data labeling is expected to improve through the iterations. Specifically, say, at iteration $t$, we set the number of weak learners in $f_c$ as `base` + $t$*`step_size`, where the value of `step_size` is decided by the tradeoff between the convergence speed and the risk of overfitting. In all our experiments, we have `base` $= 5$ and `step_size` $= 2$. Also note that the boosting classifiers tend to perfectly classify the training data, and underestimate the LogLoss (5.9). This can be resolved by *leave-one-out* estimation: The induced loss of sample $\mathbf{x}_i$ in (5.9) is evaluated by the classifier learned with the rest of the data. (For computational issue, we implement ten-fold cross-validation.)

Being a special case of integer programming, BIP is still *NP-hard*. A practical implementation of an appropriate methodology such as *branch-and-bound* or *cutting plane* would require a feasible initialization to reduce BIP into a series of linear programs,

and thus speed up the underlying optimization process. In our case, we design a greedy scheme to find an initial set of data labels. We first assume an upper bound on the cluster size. Then, among those undecided samples we identify the next possible sample labeling such that the assignment yields the smallest loss and would not cause the size of the target cluster to exceed the upper bound. The process is repeated until the data labeling is completed. Given the initialization, we apply MOSEK [89] to efficiently solving the BIP problems. For example, it takes less than one second when $(N, C) = (600, 20)$.

## 5.5 Experimental Results

We carry out two sets of experiments: visual object categorization and face image grouping. The image data used in the experiments are complex and display rich variations caused by various factors. They nevertheless provide a good test bed to demonstrate the importance of using multiple feature representations. In the first experiment, we compare our approach with the state-of-the-art clustering algorithms and discuss the convergence issues. In the second experiment, we show the advantages of our approach in the aspects of performing cluster-dependent, cross-space feature selection and incorporating partially labeled data.

### 5.5.1 Visual Object Categorization

#### Dataset

The Caltech-$101$ dataset [33], collected by Fei-Fei et al., is used in our experiments of object categorization. Following the setting in [27], we select the same twenty object categories from the Caltech-101 dataset, and randomly pick $30$ images from each category to form a dataset of $600$ images. The large and diverse intraclass variations make clustering on the dataset very challenging.

### Descriptors, Distances and Kernels

For the dataset, we consider five different image descriptors and their corresponding distance function. They yield the following kernels (denoted below in bold and in abbreviation) via (5.1):

- **GB**. For a given image, we randomly sample $400$ edge pixels, and apply *geometric blur* descriptor [5] to them. With these image features, we adopt the distance function, as is suggested in equation (2) of the work by Zhang et al. [113], to obtain the kernel.

- **SIFT**. The kernel is analogously constructed as kernel GB, except now the *SIFT* descriptor [63] is used to extract features.

- **SS**. We apply the *self-similarity* descriptor [82] to an evenly sampled grid of each image, and use $k$-means clustering to generate *visual words* from the resulting local features of all images. Then the kernel is built by matching *spatial pyramids*, which is proposed in [58].

- **C2**. Mutch and Lowe [67] propose a set of features that emulates the visual system mechanism. We adopt these biologically inspired features to characterize images. An RBF kernel is constructed for the data representation.

- **PHOG**. We adopt the *PHOG* descriptor [8] to capture image features. Together with the $\chi^2$ distance, the kernel is established.

### Quantitative Results

In all the experiments, we set the number of clusters to the number of classes in ground truth, and evaluate clustering performances with the two criteria: *clustering accuracy* (ACC) [112], and *normalized mutual information* (NMI) [87]. The output ranges of the

Table 5.1: The performances in form of [ACC (%) / NMI] of different clustering methods. **Top**: each kernel is considered individually. **Bottom**: all kernels are used jointly.

| kernel | $k$-means | Affinity Prop. | Spectral Clus. | Ours |
|---|---|---|---|---|
| GB | 68.0 / 0.732 | 52.5 / 0.578 | 69.5 / 0.704 | **75.0 / 0.742** |
| SIFT | 62.5 / 0.680 | 59.8 / 0.638 | 62.5 / 0.668 | **69.6 / 0.706** |
| SS | **65.7 / 0.659** | 55.7 / 0.574 | 63.3 / 0.655 | 62.1 / 0.639 |
| C2 | 37.8 / 0.417 | 47.5 / 0.517 | **57.7 / 0.585** | 51.2 / 0.550 |
| PHOG | 53.3 / 0.547 | 43.3 / 0.464 | **61.0 / 0.624** | 55.2 / 0.569 |

| kernels | CE + $k$-means | CE + Affinity Prop. | CE + Spectral Clus. | Ours |
|---|---|---|---|---|
| All | 73.8 / 0.737 | 63.3 / 0.654 | 77.3 / 0.758 | **85.7 / 0.833** |

two criteria are both $[0, 1]$. The larger the values, the better the clustering results are. Our approach starts from a random initialization of data partitioning $Y$. We run our approach $20$ times with different random partitionings, and report the average performance. Besides, we respectively set $\ell$ and $u$ in (5.5) as $\lfloor 0.8k_1 \rfloor$ and $\lceil 1.2k_2 \rceil$, where $k_1$ and $k_2$ are the minimal and the maximal cluster sizes in the dataset respectively.

We first evaluate our approach in the cases that each descriptor is used *individually*, and compare it with three popular clustering methodologies: $k$-means, affinity propagation [36], and spectral clustering [68]. The implementations for the three techniques are as follows. $k$-means works on data in Euclidean space, so we use *multidimensional scaling* [21] to recover the feature vectors of data from their pairwise distances. Affinity propagation detects representative exemplars (clusters) by considering similarities among data. We set the pairwise similarities as the negative distances. Spectral clustering and our approach both take a kernel matrix as input. The outcomes by the four clustering algorithms are shown in Table 5.1 (**top**). In this setting, the proposed approach outperforms $k$-means and affinity propagation, and is competitive with spectral clustering.

When multiple kernels are used simultaneously, we compare our approach with cluster ensembles (CE) [87]. Our approach jointly considers the five kernels in the clustering procedure, while cluster ensembles combines the five separately generated clustering results by one of the three methods: $k$-means, affinity propagation and spectral clustering. We report the results in Table 5.1 (**bottom**). First of all, our approach achieves significant

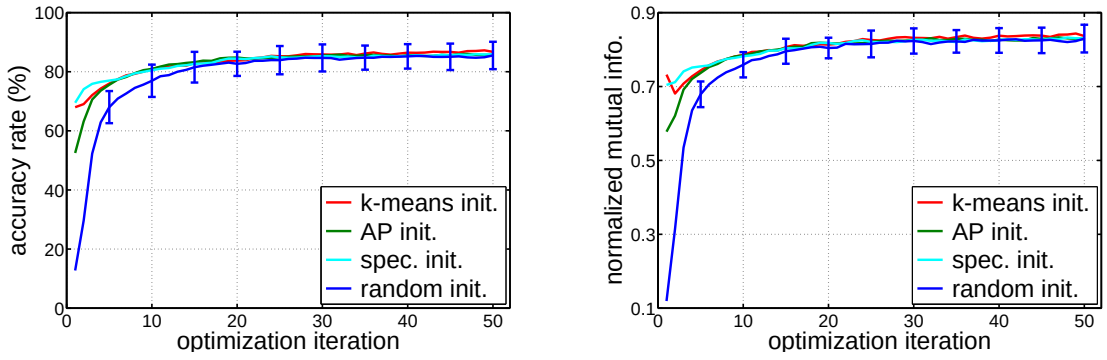



Figure 5.2: With different initializations, the clustering accuracy (**left**) and normalized mutual information (**right**) of our approach along the iterative optimization

improvements of $10.7\%$ $(= 85.7\% - 75.0\%)$ in ACC and $0.091$ $(= 0.833 - 0.742)$ in NMI over the best result obtained with a single kernel. It suggests that these kernels tend to complement, and our approach can exploit them to yield a better clustering. On the other hand, while cluster ensembles merges multiple clustering results in a global fashion, our approach instead performs cluster-dependent feature selection over multiple descriptors to recover the cluster structure. The quantitative results show that our method can make the most of multiple kernels, and improves the performance from $77.3\%$ to $85.7\%$ in ACC and from $0.758$ to $0.833$ in NMI.

Pertaining to the convergence issues, we evaluate our approach with $23$ different initializations, including $20$ random data partitionings and three meaningful clustering results by applying $k$-means, affinity propagation and spectral clustering to kernel GB, respectively. The clustering performances through the iterative optimization procedure are plotted in Figure 5.2. It can be observed that the proposed optimization algorithm is efficient and robust: It converges within a few iterations and yields similar performances with diverse initializations.

### 5.5.2 Face Image Grouping

#### Dataset

The CMU PIE database [84] is used in our experiments of face image grouping. It comprises face images of $68$ subjects. To evaluate our approach in cluster-dependent feature selection, we divide the $68$ people into four equal-size disjoint groups, each of which contains face images from $17$ subjects characterizing by a certain kind of variations. See Figure 2.7 for an overview.

Specifically, for each subject in the first group, we consider only the images of the frontal pose (C27) taken in varying lighting conditions (those under the directory "`lights`"). For subjects in the second and third groups, the images with near frontal poses (C05, C07, C09, C27, C29) under the directory "`expression`" are used. While each image from the second group is rotated by a randomly sampled angle within $[-45°, 45°]$, each from the third group is instead occluded by a non-face patch, whose area is about ten percent of the face region. Finally, for subjects in the fourth group, the images with out-of-plane rotations are selected under the directory "`expression`" and with the poses (C05, C11, C27, C29, C37). All images are cropped and resized to $51 \times 51$ pixels.

#### Descriptors, Distances and Kernels

With the dataset, we design and adopt a set of visual features, and establish the following four kernels.

- **DeLight**. The data representation is obtained from the delighting algorithm [46], and the corresponding distance function is set as $1 - \cos\theta$, where $\theta$ is the angle between a pair of samples under the representation. Some delighting results are shown in Figure 2.8. It can be seen that variations caused by different lighting conditions are significantly alleviated under the representation.

Table 5.2: The performances of cluster ensembles and our approach in different settings.

| method | kernel(s) | dataset (number of classes) | | | | |
|---|---|---|---|---|---|---|
| | | All (68) | Lighting (17) | Rotation (17) | Occlusion (17) | Profile (17) |
| Ours | DeLight | 40.2 / 0.628 | **91.4 / 0.974** | 21.0 / 0.435 | 25.5 / 0.508 | 23.0 / 0.487 |
| | LBP | **47.3 / 0.672** | 71.1 / 0.886 | **59.9 / 0.744** | 30.0 / 0.500 | **28.2 / 0.512** |
| | RsLTS | 39.3 / 0.647 | 35.4 / 0.518 | 32.9 / 0.495 | **61.4 / 0.757** | 27.6 / 0.492 |
| | RsL2 | 31.6 / 0.628 | 50.9 / 0.685 | 27.6 / 0.464 | 19.5 / 0.352 | 28.4 / 0.509 |
| CE | All | 55.4 / 0.746 | 92.6 / 0.975 | 43.8 / 0.657 | 55.4 / 0.695 | 29.8 / 0.535 |
| Ours | All | **61.9 / 0.822** | **93.6 / 0.985** | **57.8 / 0.730** | **64.8 / 0.781** | **31.6 / 0.554** |

- **LBP**. As is illustrated in Figure 2.9, we divide each image into $96 = 24 \times 4$ regions, and use a rotation-invariant *local binary pattern* (LBP) operator [69] (with operator setting $LBP^{riu2}_{8,1}$) to detect $10$ distinct binary patterns. Thus an image can be represented by a $960$-dimensional vector, where each dimension records the number of occurrences that a specific pattern is detected in the corresponding region. To achieve rotation invariant, the distance between two such vectors, say, $\mathbf{x}_i$ and $\mathbf{x}_j$, is the minimal one among the $24$ values computed from the distance function $1-\text{sum}(\min(\mathbf{x}_i, \mathbf{x}_j))/\text{sum}(\max(\mathbf{x}_i, \mathbf{x}_j))$ by circularly shifting the starting radial axis for $\mathbf{x}_j$. Clearly, the base kernel is constructed to deal with variations resulting from rotations.

- **RsL2**. Each sample is represented by its pixel intensities in raster scan order. Euclidean ($L^2$) distance is used to correlate two images.

- **RsLTS**. The same as RsL2 except that the distance function is now based on the *least trimmed squares* (LTS) with $20\%$ outliers allowed. The kernel is designed to take account of the partial occlusions in face images.

### Quantitative Results

We report the performances of applying our approach to the four kernels one by one in the third column of Table 5.2 (**top**). Besides, we also record the performances with respect to each of the four groups in the last four columns of the same table. (Each group is named

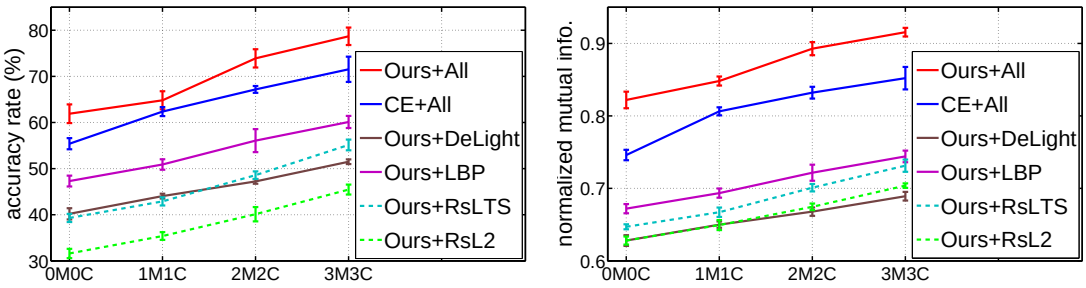



Figure 5.3: Coupling different amounts of **m**ust-links and **c**annot-links per subject with different settings of kernel(s), the performances of cluster ensembles and our approach

according to the type of its intraclass variation.) Note that each quantitative result in the last four columns is computed by considering only data in the corresponding group. No additional clustering is performed. As expected, the four kernels generally yield good performances in dealing with some specific kinds of intraclass variations. For example, the kernel DeLight achieves a satisfactory result for subjects in the Lighting group, while LBP and RsLTS yield acceptable outcomes in Rotation and Occlusion groups respectively. However, none of them is good enough for dealing with the whole dataset. Still the results reveal that if we could choose proper features for each subject, it would lead to a substantial improvement.

To verify the point, we apply the proposed clustering technique to the four kernels simultaneously, and compare it with cluster ensembles, which is used to merge the four clustering results derived by implementing our approach with single kernel. In Table 5.2 (**bottom**), it shows that using multiple kernels in our approach can achieve remarkable improvements over the best result obtained from using a single kernel (i.e. LBP), and also substantially outperforms the foregoing setting for cluster ensembles.

Indeed performing clustering with this dataset is hard, due to the large subject number and the extensive intraclass variations. We thus randomly generate one **m**ust-link and one **c**annot-link for each subject, and denote the setting of semisupervised clustering as `1M1C`. Analogously, we also have `0M0C` (i.e. unsupervised), `2M2C` and `3M3C`. Com-

bining different amounts of pairwise constraints and different settings of kernel(s), the performances with respect to ACC and NMI of our approach are shown in Figure 5.3. It is clear that by introducing only a few constraints, our approach can achieve considerable gains in performance.

## 5.6 Summary

We have presented an effective approach that carries out cluster-dependent feature selection for clustering complex data with multiple feature representations. In the approach, we incorporate the supervised training processes of cluster-specific classifiers into the unsupervised clustering procedure, cast them as a joint optimization problem, and develop an efficient technique to accomplish it. The proposed approach is comprehensively evaluated by two challenging vision applications coupled with various feature representations of data. The promising experimental results consolidate the usefulness of our approach. In addition, the introduced formulation of our approach provides a new way of extending the multiple kernel learning framework, which is typically used in supervised problems, to address unsupervised and semisupervised applications. This aspect of generalization introduces a new frontier in applying multiple kernel learning to handling increasingly complex vision tasks.

Chapter 6

# Conclusions and Future Work

We conclude the thesis and discuss the future work in this chapter.

## 6.1 Conclusions

In this thesis, we have developed several multiple kernel learning techniques, and applied them to computer vision applications. The thesis consists of three parts. The three parts are self-contained but highly correlative. They share a common theme: Multiple kernels, each of which is established from data under a particular descriptor, are jointly taken into account to provide a unified representation for heterogeneous data features, enhance the analysis of the given dataset, and boost the performances of the underlying vision applications. Learning with multiple kernels not only alleviates the problem that no single descriptor can catch up the complexity of data in nowadays vision applications but also connect various data descriptors to one class of the best off-the-shelf classification methodologies – kernel machines. In the following, we respectively summarize the three main parts of the thesis, and discuss their contributions to advancing the related fields of computer vision.

In the first part, the proposed MKL-DR framework transforms data lying in multiple spaces induced by various descriptors into an Euclidean space of lower dimension. It thus

provides a compact and unified view of these data for a set of vision applications, such as recognition and clustering. On the one hand, MKL-DR introduces a new paradigm to generalize a broad scope of existing dimensionality reduction techniques to consider multiple kernels. On the other hand, it riches the kernel selection process in the MKL framework through the diverse objective functions of different DR methods. The work results in two important impacts: 1) It allows us to explore more prior knowledge for data analysis via DR, including choosing a proper set of visual features to characterize data and adopting a DR method to appropriately model the relationships among data points; 2) It provides a new perspective of applying multiple kernel learning, which typically addresses supervised applications, to both unsupervised and semi-supervised ones. This aspect of generalization explores a new frontier in applying MKL to solving vision and learning applications.

The second part of the thesis focuses on local learning for supervised object recognition. For a large dataset to be recognized, we observe that the optimal data descriptor for recognition is often category-dependent or even sample-dependent. We improve the recognition accuracies by taking the observation into account. Our idea is to represent data under each descriptor by a kernel matrix, and by this way information fusion from various descriptors can be carried out in the domain of kernel matrices. We learn an ensemble kernel for each training sample, and each ensemble kernel is derived by selecting the descriptors to best interpret the relationships among data near the corresponding sample. Recognition is then performed by a set of sample-dependent classifiers rather than a global classifier. To our knowledge, this is the first approach that adopts multiple kernel learning to deal with object recognition.

Besides, we aim to eliminate the two unpleasant effects, inefficiency and the high risk of overfitting, arising in local learning. Specifically, we cast multiple, independent training processes of local classifiers to a correlative multi-task learning problem, and develop a boosting-based algorithm to accomplish it. The proposed technique is comprehensively

evaluated with two benchmark datasets, i.e., Caltech-101 and VOC 2007. The recognition rates in both datasets are comparable to those yielded by the respective state-of-the-art approaches. Our method can be considered as a general multi-task learning tool for vision applications where multiple correlative classifiers are required, such as multi-view face detection, multi-part object tracking, or user-dependent media ranking. The framework also provides a new way to carry out multiple kernel learning in an incremental manner.

The last part of this thesis illustrate how to use supervised MKL techniques to deal with unsupervised data clustering. The major difficulty in this part results from the unsupervised nature of clustering: We have no labeled data to guide the searching of the optimal ensemble kernel over a given convex set of base kernels. The key idea for handling the difficulty is to formulate the tasks of supervised feature selection and unsupervised clustering procedure into a joint optimization problem, and the two tasks can then be carried out collaboratively. The proposed method assumes the data are represented with various feature representations, and aims to uncover the underlying cluster structure. To that end, we associate each cluster with a boosting classifier derived from MKL, and apply the cluster-specific classifier to performing feature selection cross various descriptors to best separate data of the cluster from the rest. Through solving the joint optimization iterations, the cluster structure is gradually revealed by these classifiers, while their discriminant power to capture similar data would be progressively improved owing to better data labeling. The introduced formulation of our approach provides a new way of extending the multiple kernel learning framework to address both unsupervised and semisupervised clustering problems. This aspect of generalization introduces a frontier in applying multiple kernel learning to handling increasingly complex clustering tasks.

## 6.2 Future Work

In this thesis, several multiple kernel learning techniques are developed to combine various data feature representations and improve the performances of some computer vision applications. Along this line, I identify two focal points of my future research efforts in the following. The first one is to explore and extend the applicability of multiple kernel learning to practical learning scenarios/settings, such as multiple instance learning, multi-task learning, or on-line learning. The second one is to design more efficient and robust algorithms of multiple kernel learning for manipulating large-scale and/or noisy data.

### 6.2.1 New Learning Scenarios for Multiple Kernel Learning

Despite the significant progress in descriptor design, the general conclusion remains that no single descriptor is sufficient for handling complex data of broad vision applications. Multiple kernel learning has been demonstrated to be an effective way for feature fusion in recent computer vision research. Therefore MKL becomes a core technique for performance improvement in many vision applications. In this thesis, we have extended MKL from supervised learning to semi-supervised and unsupervised learning. My future research effort is to establish a framework that generalizes MKL to various learning scenarios, such as *multi-task learning*, *multiple instance learning*, or *on-line learning*. Such generalizations will make an impact on the community of computer vision, because many vision applications build on these learning scenarios and can be advanced by feature fusion. For example, multi-task learning is widely adopted in visual learning tasks with multiple correlative classifiers, multiple instance learning can handle tasks with data of imperfect labeling, and on-line learning deals with data changing over time.

### 6.2.2 Multiple Kernel Learning for Large-scale Data Analysis

Many successful computer vision applications rely upon large numbers of training data. Further, various complementary data descriptors are often needed to capture diverse features of data. In these applications, we require more efficient and compact MKL techniques which scale well to both the size of training data and the number of the resulting kernels.

Besides, manually collecting training data is an expensive and time-consuming job. One of recent research trends in the field of computer vision focuses on automatic or semi-automatic data collection from the Internet, e.g., videos from YouTube or images from Flickr. Although it seems to be a practical substitute for manual collection, problems arising when obtaining data in this way, such as noisy or irrelevant data, should be properly addressed. Our future research along this track would emphasize on designing robust MKL techniques, which can detect irrelevant data and extract knowledge from the remaining ones with feasible computational costs.

# Bibliography


[1] R. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Machine Learning Research*, 6:1817–1853, 2005. 8, 68

[2] C. Atkeson, A. Moore, and S. Schaal. Locally weighted learning. *Artificial Intelligence Review*, 11(1-5):11–73, 1997. 7, 49, 67

[3] F. Bach, G. Lanckriet, and M. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *Proc. Int'l Conf. Machine Learning*, 2004. 3, 15, 32

[4] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24 (4):509–522, 2002. xi, 6

[5] A. Berg and J. Malik. Geometric blur for template matching. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 607–614, 2001. 26, 89, 98

[6] A. Berg, T. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 26–33, 2005. 5, 11, 29, 32, 33, 58, 60, 61, 77, 78

[7] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *Proc. Conf. Computer Vision and Pattern Recognition*, 2008. 80

[8] A. Bosch, A. Zisserman, and X. Muñoz. Image classification using random forests and ferns. In *Proc. Int'l Conf. Computer Vision*, 2007. 27, 33, 89, 90, 98

[9] A. Bosch, A. Zisserman, and X. Muñoz. Representing shape with a spatial pyramid kernel. In *Proc. ACM Conf. Image and Video Retrieval*, pages 401–408, 2007. 5, 69, 78, 80

[10] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001. 57

[11] D. Cai, X. He, and J. Han. Semi-supervised discriminant analysis. In *Proc. Int'l Conf. Computer Vision*, 2007. 13, 15, 41

[12] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *Proc. Conf. Computer Vision and Pattern Recognition*, 2010. 12

[13] R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997. 8, 68

[14] C.-C. Chang and C.-J. Lin. *LIBSVM: A Library for Support Vector Machines*, 2001. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm. 57

[15] C.-P. Chen and C.-S. Chen. Lighting normalization with generic intrinsic illumination subspace for face recognition. In *Proc. Int'l Conf. Computer Vision*, pages 1089–1096, 2005. 11

[16] H.-T. Chen, H.-W. Chang, and T.-L. Liu. Local discriminant embedding and its variants. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 846–853, 2005. 11, 13, 15, 21, 27, 28, 29

[17] H. Cheng, K. Hua, and K. Vu. Constrained locally weighted clustering. In *Proc. Int'l Conf. Very Large Data Bases*, pages 90–101, 2008. 89

[18] M. Christoudias, R. Urtasun, and T. Darrell. Bayesian localized multiple kernel learning. Technical report, EECS Dept., University of California, Berkeley, 2009. 33

[19] M. Collins, R. Schapire, and Y. Singer. Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, pages 158–169, 2002. 94

[20] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002. 86, 88

[21] T. Cox and M. Cox. *Multidimentional Scaling*. Chapman & Hall, London, 1994. 37, 99

[22] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola. On kernel-target alignment. In *Advances in Neural Information Processing Systems*, 2001. 51, 55

[23] J. Dai, S. Yan, X. Tang, and J. Kwok. Locally adaptive classification piloted by uncertainty. In *Proc. Int'l Conf. Machine Learning*, pages 225–232, 2006. 49, 55, 67

[24] K. Dana, B. Van-Ginneken, S. Nayar, and J. Koenderink. Reflectance and texture of real world surfaces. *ACM Trans. on Graphics*, 18(1):1–34, 1999. 57, 61

[25] C. Domeniconi and M. Al-Razgan. Weighted cluster ensembles: Methods and analysis. *ACM Trans. Knowledge Discovery from Data*, 2(4), 2009. 89

[26] C. Domeniconi and D. Gunopulos. Adaptive nearest neighbor classification using support vector machines. In *Advances in Neural Information Processing Systems*, 2001. 49, 67

[27] D. Dueck and B. Frey. Non-metric affinity propagation for unsupervised image categorization. In *Proc. Int'l Conf. Computer Vision*, 2007. 34, 86, 97

[28] M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool. The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results. http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf, 2006. 1

[29] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html, 2007. 12, 66, 76, 83

[30] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html, 2008.

[31] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results. http://www.pascal-network.org/challenges/VOC/voc2009/workshop/index.html, 2009.

[32] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html, 2010. 1

[33] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR Workshop on Generative-Model Based Vision*, 2004. 1, 12, 25, 33, 57, 59, 66, 76, 97

[34] A. Fred and A. Jain. Combining multiple clusterings using evidence accumulation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(6):835–850, 2005. 89

[35] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory*, 1995. 74

[36] B. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315:972–976, 2007. 34, 36, 88, 99

[37] J. Friedman. Local learning based on recursive covering. Technical report, Dept. of Statistics, Stanford University, 2005. 7, 49

[38] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 28(2):337–374, 2000. 94

[39] A. Frome, Y. Singer, and J. Malik. Image retrieval and classification using local distance functions. In *Advances in Neural Information Processing Systems*, pages 417–424, 2006. 32, 49, 50, 55, 58, 60, 66, 67

[40] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *Proc. Int'l Conf. Computer Vision*, 2007. 33, 49, 50, 66, 67, 80

[41] C. Galleguillos, B. McFee, S. Belongie, and G. Lanckriet. Multi-class object localization by combining local contextual interactions. In *Proc. Conf. Computer Vision and Pattern Recognition*, 2010. 12

[42] P. Gehler and S. Nowozin. On feature combination for multiclass object classification. In *Proc. Int'l Conf. Computer Vision*, 2009. 33

[43] M. Gönen and E. Alpaydin. Localized multiple kernel learning. In *Proc. Int'l Conf. Machine Learning*, pages 352–359, 2008. 3, 15, 66

[44] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proc. Int'l Conf. Computer Vision*, pages 1458–1465, 2005. 13, 17, 32, 33, 80

[45] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007. http://authors.library.caltech.edu/7694. 1

[46] R. Gross and V. Brajovic. An image preprocessing algorithm for illumination invariant face recognition. In *Audio-and Video-Based Biometrie Person Authentication*, pages 10–18, 2003. xii, 40, 101

[47] X. He and P. Niyogi. Locality preserving projections. In *Advances in Neural Information Processing Systems*, 2003. 13, 15, 35

[48] S. Hoi, M. Lyu, and E. Chang. Learning the unified kernel machines for classification. In *Proc. ACM Conf. Knowledge Discovery and Data Mining*, pages 187–196, 2006. 52

[49] A. Holub, M. Welling, and P. Perona. Combining generative models and fisher kernels for object recognition. In *Proc. Int'l Conf. Computer Vision*, pages 136–143, 2005. 33

[50] I. Joliffe. *Principal Component Analysis*. Springer-Verlag, 1986. 11, 13, 15

[51] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell. Gaussian processes for object categorization. *Int'l J. Computer Vision*, 88(2):169–188, 2010. 33

[52] S.-J. Kim, A. Magnani, and S. Boyd. Optimal kernel selection in kernel fisher discriminant analysis. In *Proc. Int'l Conf. Machine Learning*, pages 465–472, 2006. 16

[53] T.-K. Kim and J. Kittler. Locally linear discriminant analysis for multimodally distributed classes for face recognition with a single model image. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(3):318–327, 2005. 49, 55, 67

[54] B. Kulis, S. Basu, I. Dhillon, and R. Mooney. Semi-supervised graph clustering: A kernel approach. In *Proc. Int'l Conf. Machine Learning*, pages 457–464, 2005. 88

[55] A. Kumar and C. Sminchisescu. Support kernel machines for object recognition. In *Proc. Int'l Conf. Computer Vision*, 2007. 32

[56] G. Lanckriet, N. Cristianini, P. Bartlett, L. Ghaoui, and M. Jordan. Learning the kernel matrix with semi-definite programming. In *Proc. Int'l Conf. Machine Learning*, pages 323–330, 2002. 51, 52, 53

[57] G. Lanckriet, N. Cristianini, P. Bartlett, L. Ghaoui, and M. Jordan. Learning the kernel matrix with semidefinite programming. *J. Machine Learning Research*, 5: 27–72, 2004. 3, 15, 87, 93

[58] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 2169–2178, 2006. 5, 26, 32, 33, 69, 78, 80, 81, 89, 90, 98

[59] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *Int'l J. Computer Vision*, 43(1):29–44, 2001. 58

[60] Y.-Y. Lin, T.-L. Liu, and C.-S. Fuh. Local ensemble kernel learning for object category recognition. In *Proc. Conf. Computer Vision and Pattern Recognition*, 2007. 12, 32, 49, 50, 66

[61] Y.-Y. Lin, T.-L. Liu, and C.-S. Fuh. Dimensionality reduction for data in multiple feature representations. In *Advances in Neural Information Processing Systems*, pages 961–968, 2008. 16

[62] Y.-Y. Lin, J.-F. Tsai, and T.-L. Liu. Efficient discriminative local learning for object recognition. In *Proc. Int'l Conf. Computer Vision*, 2009. 66, 93

[63] D. Lowe. Distinctive image features from scale-invariant keypoints. *Int'l J. Computer Vision*, 60(2):91–110, 2004. xi, 5, 6, 7, 11, 26, 58, 77, 89, 98

[64] T. Malisiewicz and A. Efros. Recognition by association via learning per-exemplar distances. In *Proc. Conf. Computer Vision and Pattern Recognition*, 2008. 49, 66, 67

[65] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing*, pages 41–48, 1999. 13, 29

[66] B. Moghaddam and G. Shakhnarovich. Boosted dyadic kernel discriminants. In *Advances in Neural Information Processing Systems*, 2002. 68, 72, 93

[67] J. Mutch and D. Lowe. Multiclass object recognition with sparse, localized features. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 11–18, 2006. 5, 26, 33, 69, 78, 81, 90, 98

[68] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering. In *Advances in Neural Information Processing Systems*, 2001. 88, 99

[69] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002. 41, 102

[70] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *Int'l J. Computer Vision*, 42(3):145–175, 2001. 27, 78

[71] G. Pass, R. Zabih, and J. Miller. Comparing images using color coherence vectors. In *Proc. ACM Conf. Multimedia*, pages 65–73, 1996. 5, 59

[72] E. Pekalska, P. Paclik, and R. Duin. A generalized kernel approach to dissimilarity-based classification. *J. Machine Learning Research*, 2(2):175–211, 2002. 13

[73] J. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin dags for multiclass classification. In *Advances in Neural Information Processing Systems*, pages 547–553, 1999. 53

[74] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. More efficiency in multiple kernel learning. In *Proc. Int'l Conf. Machine Learning*, pages 775–782, 2007. 3, 15, 79

[75] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. 2

[76] V. Roth and T. Lange. Feature selection in clustering problems. In *Advances in Neural Information Processing Systems*, 2003. 86, 88

[77] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000. 11, 13

[78] H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 1746–1759, 2000. 66

[79] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002. 2

[80] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998. 3, 13, 18

[81] T. Serre, L. Wolf, and T. Poggio. Object recognition with features inspired by visual cortex. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 994–1000, 2005. 5, 26, 33, 58, 78

[82] E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *Proc. Conf. Computer Vision and Pattern Recognition*, 2007. 26, 78, 98

[83] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. 86, 88

[84] T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination, and expression database. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 25(12):1615–1618, 2005. 38, 101

[85] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proc. Int'l Conf. Computer Vision*, pages 1470–1477, 2003. 58

[86] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *J. Machine Learning Research*, 7:1531–1565, 2006. 3, 15

[87] A. Strehl and J. Ghosh. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *J. Machine Learning Research*, 3:583–617, 2002. 35, 89, 98, 99

[88] J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000. 11, 13

[89] The MOSEK Optimization Software. http://www.mosek.com/index.html. 97

[90] S. Todorovic and N. Ahuja. Learning subcategory relevances for category recognition. In *Proc. Conf. Computer Vision and Pattern Recognition*, 2008. 33

[91] A. Torralba, K. Murphy, and W. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(5):854–869, 2007. 8, 68, 76

[92] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991. xi, 6, 7

[93] O. Tuzel, F. Porikli, and P. Meer. Kernel methods forweakly supervised mean shift clustering. In *Proc. Int'l Conf. Computer Vision*, 2009. 86, 88

[94] K. van de Sande, T. Gevers, and C. Snoek. Evaluation of color descriptors for object and scene recognition. In *Proc. Conf. Computer Vision and Pattern Recognition*, 2008. 81

[95] J. C. van Gemert, C. J. Veenman, A. W. M. Smeulders, and J. M. Geusebroek. Visual word ambiguity. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 32(7):1271–1283, 2009. 83

[96] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38: 49–95, 1996. 23, 76

[97] V. Vapnik. *Statistical Learning Theory*. Wiley, 1998. 2, 50

[98] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *Proc. Int'l Conf. Computer Vision*, 2007. 12, 17, 29, 78, 90, 93

[99] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *Int'l J. Computer Vision*, 62(1-2):61–81, 2005. 5, 58, 61

[100] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *Proc. Int'l Conf. Computer Vision*, 2009. 12

[101] P. Viola and M. Jones. Robust real-time face detection. *Int'l J. Computer Vision*, 57(2):137–154, 2004. xi, 6, 7

[102] H. Wang, S. Yan, D. Xu, X. Tang, and T. Huang. Trace ratio vs. ratio trace for dimensionality reduction. In *Proc. Conf. Computer Vision and Pattern Recognition*, 2007. 21

[103] L. Wolsey. *Integer Programming*. John Wiley & Sons, 1998. 94

[104] M. Wu and B. Schölkopf. A local learning approach for clustering. In *Advances in Neural Information Processing Systems*, pages 1529–1536, 2006. 36

[105] E. Xing, A. Ng, M. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In *Advances in Neural Information Processing Systems*, 2002. 88

[106] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In *Advances in Neural Information Processing Systems*, 2004. 89

[107] S. Yan, D. Xu, Q. Yang, L. Zhang, X. Tang, and H. Zhang. Discriminant analysis with tensor representation. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 526–532, 2005. 5

[108] S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, and S. Lin. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(1):40–51, 2007. 6, 11, 12, 13, 14, 15, 28

[109] J. Yang, Y. Li, Y. Tian, L. Duan, and W. Gao. Group-sensitive multiple kernel learning for object categorization. In *Proc. Int'l Conf. Computer Vision*, 2009. 33

[110] J. Ye, R. Janardan, and Q. Li. Two-dimensional linear discriminant analysis. In *Advances in Neural Information Processing Systems*, 2004. 11

[111] J. Ye, R. Janardan, and Q. Li. Gpca: An efficient dimension reduction scheme for image compression and retrieval. In *Proc. ACM Conf. Knowledge Discovery and Data Mining*, pages 354–363, 2004. 5, 11

[112] J. Ye, Z. Zhao, and M. Wu. Discriminative k-means for clustering. In *Advances in Neural Information Processing Systems*, 2007. 86, 88, 98

[113] H. Zhang, A. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 2126–2136, 2006. 17, 26, 29, 32, 33, 49, 58, 60, 69, 77, 78, 80, 90, 98

[114] B. Zhao, F. Wang, and C. Zhang. Efficient multiclass maximum margin clustering. In *Proc. Int'l Conf. Machine Learning*, pages 1248–1255, 2008. 89

[115] J. Zhu, S. Rosset, H. Zou, and T. Hastie. Multi-class adaboost. Technical report, Dept. of Statistics, University of Michigan, 2005. 30

# Publications

1. **Yen-Yu Lin**, Tyng-Luh Liu, and Chiou-Shann Fuh. "Multiple Kernel Learning for Dimensionality Reduction," *IEEE Transactions on Pattern Analysis and Machine Intelligence* (***TPAMI***), To appear as a regular paper.

2. **Yen-Yu Lin**, Tyng-Luh Liu, and Chiou-Shann Fuh. "Clustering Complex Data with Group-dependent Feature Selection," In *European Conference on Computer Vision* (***ECCV***), volume 6316, pages 84-97, Crete, Greece, 2010.

3. **Yen-Yu Lin**, Jyun-Fan Tsai, and Tyng-Luh Liu. "Efficient Discriminative Local Learning for Object Recognition," In *IEEE International Conference on Computer Vision* (***ICCV***), Kyoto, Japan, 2009.

4. **Yen-Yu Lin**, Tyng-Luh Liu, and Chiou-Shann Fuh. "Dimensionality Reduction for Data in Multiple Feature Representations," In *Advances in Neural Information Processing Systems* (***NIPS***), pages 961-968, Vancouver, Canada, 2008.

5. **Yen-Yu Lin**, Tyng-Luh Liu, and Chiou-Shann Fuh. "Local Ensemble Kernel Learning for Object Category Recognition," In *IEEE International Conference on Computer Vision and Pattern Recognition* (***CVPR***), Minneapolis, USA, 2007. (Oral)

6. **Yen-Yu Lin**, Tyng-Luh Liu, and Hwann-Tzong Chen. "Semantic Manifold Learning for Image Retrieval," In *ACM International Conference on Multimedia* (***ACM-MM***), pages 249-258, Singapore, 2005. (Nominated for Best Student Paper Award)

7. **Yen-Yu Lin**, Tyng-Luh Liu. "Shape Recognition Using Fast Boosted Filtering," In *IEEE International Conference on Image Processing* (***ICIP***), pages 554-557, Genova, Italy, 2005.

8. **Yen-Yu Lin**, Tyng-Luh Liu. "Robust Face Detection with Multi-Class Boosting," In *IEEE International Conference on Computer Vision and Pattern Recognition* (***CVPR***), pages 679-686, San Diego, USA, 2005.

9. **Yen-Yu Lin**, Tyng-Luh Liu, and Chiou-Shann Fuh. "Fast Object Detection with Occlusions," In *European Conference on Computer Vision* (***ECCV***), volume 3021, pages 402-413, Prague, Czech, 2004.

# Patents

1. Tyng-Luh Liu and **Yen-Yu Lin**. "Object-detection Method Multi-class Bhattacharyya Boost Algorithm Used Therein," U.S. Patent #7286707, Taiwan Patent #I284862, 2007.