MAXIMUM SIMILARITY BASED FEATURE MATCHING AND ADAPTIVE MULTIPLE KERNEL LEARNING FOR OBJECT RECOGNITION

by

Ziming Zhang

B.Eng., Northeastern University, 2005

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the School
of
Computing Science

APPROVAL

Name:	Ziming Zhang
Degree:	Master of Science
Title of thesis:	Maximum Similarity Based Feature Matching and Adaptive Multiple Kernel Learning for Object Recognition
Examining Committee:	Dr. Stella Atkins Chair
	Dr. Ze-Nian Li, Co-Senior Supervisor
	Dr. Mark S. Drew, Co-Senior Supervisor
	Dr. Greg Mori, SFU Examiner
Date Approved:	April ~~14~~ 26, 2010

Abstract

In this thesis, we perform object recognition using (i) maximum similarity based feature matching, and (ii) adaptive multiple kernel learning. Images are likely more similar if they contain objects within the same categories, so how to measure image similarities correctly and efficiently is one of the critical issues for object recognition. We first propose to match features between two images so that their similarity is maximized, and employ support vector machines (SVMs) for recognition based on the maximum similarity matrix. Secondly, given several similarity matrices (kernels) created by different visual information in images, we propose a novel adaptive multiple kernel learning technique to generate an optimal kernel from all the kernels based on biconvex optimization. These two new approaches are tested on the most recent image benchmark datasets and their results are impressive, equalling or bettering the state-of-the-art results.

Key words: Object recognition; Maximum similarity; Feature matching; Adaptive multiple kernel learning

for my parents

Acknowledgments

I would like to take this opportunity to thank my two supervisors, Professor Ze-Nian Li and Professor Mark Drew, for all their great help and advices during my studies in Simon Fraser University. Meanwhile, I would like to thank Professor Greg Mori to be my Examiner, my colleagues Dr. Yang Wang, Jiawei Huang, Weilong Yang, and many others at Vision and Media Lab, and all my friends for their suggestions and care.

Finally, I would like to thank my family for their understanding and support.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

Object recognition in computer vision aims to emulate one of the most essential functionalities of human vision to help machines recognize objects in the environments automatically. According to the data which contain objects, object recognition can be performed in 2D images, 3D videos, 4D medical data, etc. This thesis focuses on recognizing objects in images, specifically, recognizing the categories of objects rather than recognizing (or detecting) specific objects or identifying objects. For example, as long as there exists an instance of bike with a reasonable scale in an image, no matter where or at what scale the bike instance is, we can still recognize this image as a bike image.

According to the number of nouns in the human languages, there are at least 10,000 to 30,000 object categories in the world [9]. However, at present machines can only recognize a few of them. For instance, to our best knowledge, in the computer vision community, the biggest public dataset for object recognition is Caltech-256 [30], which consists of 256 object categories plus a background category. Fig. 1.1[1] shows 40 samples from the first 40 categories in Caltech-256, one sample per category. So far the best recognition rate on this dataset is only about 50% [24], whereas for humans these objects can be categorized very easily. Therefore, much more effort is still needed for object recognition, and its breakthroughs will bring great benefit to other relative research such as video surveillance, navigation, etc.

[1]These images are extracted from `http://www.vision.caltech.edu/Image_Datasets/Caltech256/images/`, where the samples of all the categories are shown.

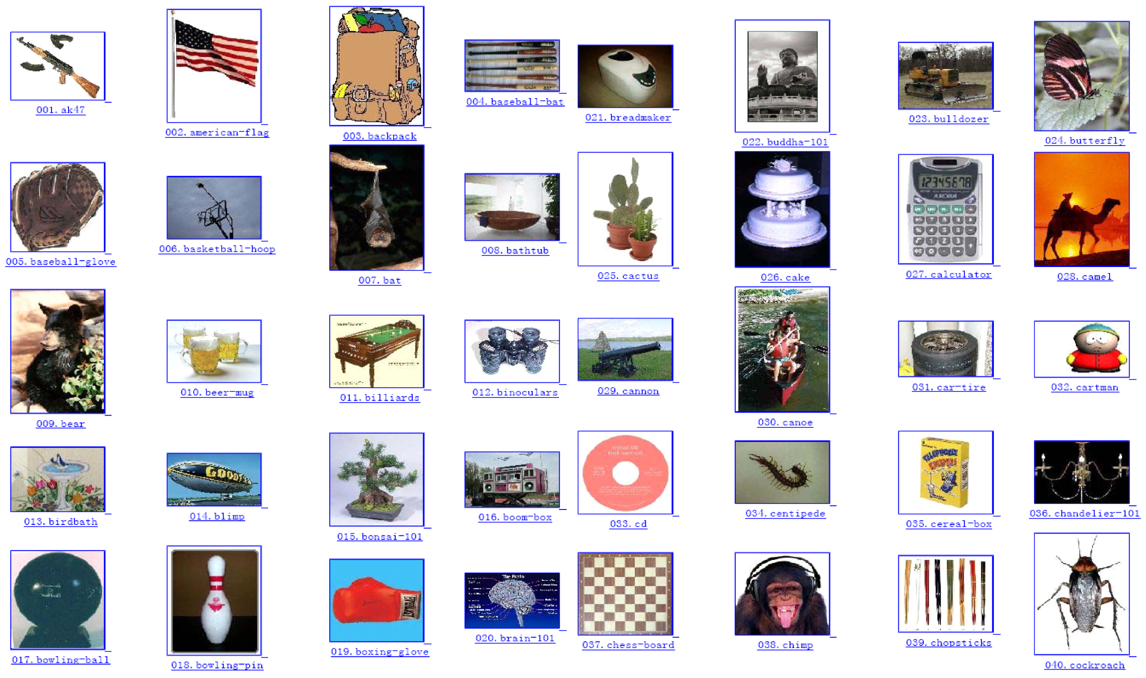


Figure 1.1: 40 samples from the first 40 categories in the Caltech-256, one sample per category.

Image similarity in this thesis is defined as an object based measure to show how likely two images contain the instances within the same object category, and a higher image similarity indicates a higher probability. Based on this measure, a perfect recognition can be made if the similarities of images which contain the same-category objects are higher than those of images which contain objects in different categories.

Considering that each object can be decomposed into smaller pieces, and each smaller piece gives an evidence that this object probably appears, we propose to match the *same-type* features in images so that the image similarities are maximized. Here, "same-type" means that the visual information in the features should be the same. For instance, all features come from Scale Invariant Feature Transform (SIFT) [45] descriptors. We believe that the maximization criterion is suitable to reflect the image similarities properly, because two more-similar images are inclined to have a higher maximization value. Specifically, with or without consideration for the spatial configurations of features, we further propose *constrained global feature correspondence* [70] and *probabilistic feature matching* approaches, respectively, for matching purpose.

With the increase of the number of visual sources from images, we face another problem, that is, how to combine these different visual information for object recognition. Given a set of pre-calculated kernels, each of which is generated by the same-type features, we propose another approach, called *Adaptive Multiple Kernel Learning*, to learn an optimal kernel using the max-margin criterion. Unlike other multiple kernel learning (MKL) approaches, our approach is based on a biconvex optimization technique involved with an arbitrary norm of kernel coefficients to search for local optima as the solutions, and its parameters can be learned for all the object categories either separately or jointly. Therefore, the efficiency of our approach is much higher than the traditional convex optimization based MKL approaches, especially with many object categories.

The rest of the thesis is organized as follows. Chapter 2 gives some background for object recognition in computer vision. Chapter 3 presents our maximum similarity based feature matching approaches. Chapter 4 explains our adaptive multiple kernel learning approach for information combination. Finally, we conclude the thesis in Chapter 5.

Chapter 2

Background

Again, the purpose of this thesis is only recognizing the object categories in images without localization or detection. In general, the object categorization process can be simply summarized as follows:

I **Representing objects**. An *object representation* is a type of organization of low level features in images, which are extracted from the image information of texture, color, shape, etc., to represent objects. Usually, different algorithms have their own object representations.

II **Learning object category models**. Based on the object representations in the training images, the model learning algorithms can generate a visual model for each object category. A *visual model* (*model* for short) of an object category can reflect some characteristics of the category defined by the model learning algorithms.

III **Recognizing object categories**. After learning the visual models, corresponding recognition algorithms are employed to calculate the similarities between the objects and the categories. A higher value means more likely the image contains the objects within the categories.

In the rest of this chapter, we will review some background information in each step. Specifically, Section 2.1 reviews some widely used low level features, Section 2.2 presents some image patch sampling techniques, Section 2.3 introduces some object representations, and Section 2.4 explains support vector machines (SVMs) for learning and recognition.

2.1 Low Level Features

Low level features (*descriptors* for short) are usually vectors describing the visual information (*e.g.* texture, color, edge, etc.) in images or smaller image parts (*patches* for short). A simple example of low level features is a vector of RGB pixel values, which describes the color information of the pixel. So far, many sophisticated local descriptors have been proposed, for instance, Scale Invariant Feature Transform (SIFT) [45], PCA-SIFT [34], Gradient Location-Orientation Histogram (GLOH) [48], Speeded Up Robust Features (SURF) [2], Shape Context (SC) [3] and Geometric Blur (GB) [6]. Since in our experiments, only SIFT, SC and GB are employed, we just introduce these three descriptors.

SIFT descriptors are highly distinctive, and its basic idea is to describe the patches in a high dimensional vector using gradient orientation histogram. Empirically in [45], to generate a SIFT descriptor for a patch, first this patch is resized to 16*16 pixels, then divided into 4*4 cells where in each cell there are 4*4 pixels. After this, a gradient orientation histogram with 8 bins is generated for each cell. Finally, an 8*4*4=128 dimensional feature vector is created for each SIFT descriptor.

Shape Context descriptors aim to locate the distribution of other shape points over relative positions in a region around a pre-defined reference point (*center* for short) by spatial quantization. When generating an SC descriptor for each point, a polar coordinate system is employed where one dimension is log-distance, and the other is relative angle.

Geometric Blur descriptors are highly dependent on the extracted edges in images, and the basic idea is to capture the relationship of oriented edges in a region around a pre-defined reference point (*center* for short) using a polar coordinate system, similar to SC descriptors. The difference is that a weighting distribution (usually a Gaussian distribution) is applied to the sample points so that each sample point has its own weight for counting purpose. The closer the sample point to the center is, the higher its weight is.

Additional properties (*e.g.* scale-invariance and rotation-invariance) of a descriptor are decided by how the patch is detected and how the descriptor is calculated. For example, SIFT descriptors are born to be scale-invariant and rotation-invariant, because (i) the interest point for each SIFT descriptor is located as a local optimum on multiple scales of an image using Difference-of-Gaussian (DoG) by comparing each sample point to its eight neighbors in the current DoG image (spatial space) and nine neighbors in the scale above and below (scale space), and then the whole patch is re-scaled to 16×16 pixels, which results in scale-invariance; (ii) to guarantee its rotation-invariance, the orientations of a SIFT descriptor are assigned relative to the interest point orientations. *Interest point orientations* are calculated based on the image gradient magnitudes and orientations at all the pixels, and assigned as the highest peak and the local peaks that are within 80% of the highest peak in a 36-bin weighted orientation histogram. For SC and GB descriptors, after locating the interest points and their orientations in images, the polar coordinate systems in both SC and GB descriptors can be centered at the interest points and aligned relative to the interest point orientations like SIFT descriptors so that both SC and GB descriptors can be scale-invariant and rotation-invariant.

2.2 Image Patch Sampling

An *image patch sampling* technique is an image patch extraction strategy before generating descriptors. In general, there exist several sampling techniques for object recognition, for example, dense sampling (*e.g.* [41]), random sampling (*e.g.* [43]), segment sampling (*e.g.* [31]), interest point sampling (*e.g.* [45]), etc.

Dense Sampling usually divides images into many same-size cells with specific structures. Its advantages are that it can control the total number, centers and scales of the patches, and it can utilize the information of each image sufficiently by covering it using the patches. The major drawback of this strategy is that the extracted patches may not be distinctive enough for object recognition.

Random Sampling decides the centers and scales of the patches randomly. Like dense sampling, it can control the total number of the patches. However, it cannot control the centers and scales of the patches, and it may not extract distinctive patches for objects.

Segment Sampling utilizes some segmentation techniques (*e.g.* normalized cut [57]) to extract patches. Its advantages are that it can control the total number of the patches to a certain degree, probably locate some distinctive patches for objects, and probably utilize the image information sufficiently. However, it cannot control the positions and scales of the patches.

Interest Point Sampling employs some interest point (or region) detectors to locate the centers and scales of the patches. An *interest point* is a pixel location around which a patch with a specific scale is extracted. Its advantage is that it can locate the distinctive object patches in the images as many as possible. However, it cannot extract an arbitrary number of patches like dense sampling or random sampling (*e.g.* no interest points at all in images), cannot control the centers and scales of the patches, and cannot utilize the image information sufficiently.

In [50], the authors did some research on the effect of different sampling strategies for image classification. It is found that the single most important factor governing performance is the number of patches sampled from the test image and ultimately interesting point detectors cannot provide enough patches to compete, so random sampling gives equal or better classifiers than interest point sampling. And in [43], the authors compared the dense, random, and interest point sampling strategies in their experiments, and addressed that dense sampling, which yields the most number of patches, is the best. So in our experiments, we only adopt dense sampling for patch extraction regardless of scale-invariance.

2.3 Object Representations

Using different algorithms, objects can be represented in different ways, for instance, Set-of-Features models (*SoF* for short, *e.g.* [10]), Bag-of-Words models (*BoW* for short, *e.g.* [15]), part based models (*e.g.* [20]), hierarchical models (*e.g.* [19]), etc.

Set-of-Features models represent each image as a set of same-type feature vectors, and assume that these features are independent and at least one feature comes from the target objects. Notice that features here could be either low level descriptors (*e.g.* SIFT) or high level vector representations (*e.g.* BoW). SoF is quite flexible, no limits on the number or dimensionality of features, and thus it is quite suitable for the feature matching purpose.

Bag-of-Words models represent each image as a histogram of occurrences of the codewords in a pre-defined codebook to capture their distribution. A *codebook* is generated from a collection of clusters of training descriptors, and a *codeword* is the center of a cluster. BoW maps each descriptor to an index in the codebook by searching for its nearest codeword, considers this codeword to appear once, and finally creates a histogram. Still, BoW assumes that the descriptors in each image are independent and at least one comes from the target objects. Impressively, the performances of this simple representation on many datasets for object recognition are very good (*e.g.* [60, 41, 44]).

Part based models take the spatial relationship of the patches in images into account, and represent objects in a structured way. Usually, descriptors are still quantized using a codebook like BoW, and based on this representation the learning algorithms can learn distinctive object part structures, which makes this representation quite successful in object recognition and detection. However, the number of learned object parts is restricted to a very small number, otherwise the computational time is very high.

Hierarchical models represent objects by organizing their information into tree-like trackable structures. Notice that the information here is not limited to the descriptors extracted directly from images, but it could be any high level information among the objects (*e.g.* semantics). Theoretically, more similar the images are, more likely they share more information in the hierarchy.

Due to the characteristics of our algorithms, in our experiments we employ SoF for maximum similarity based feature matching and BoW for adaptive multiple kernel learning.

2.4 Support Vector Machines

Support vector machines (SVMs) are a set of supervised learning algorithms for classification and regression based on the max-margin criterion. *Supervised learning* is used for generating mapping functions between inputs and their labels. Originally, an SVM is designed for binary-class cases, and its basic idea is to find a separating hyperplane which maximizes the scaled distance between its two parallel hyperplanes, that is, the *margin*. Notice that in an SVM, there is an assumption that the larger the margin is, the smaller the generalization error of the classifier will be.

Specifically, in *binary-class* cases where only two classes exist, one positive one negative, suppose the training data is $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_n, y_n)\}$ where $\mathbf{x}_i$ $(i = 1, 2, \cdots, n)$

denotes an input vector and its corresponding y_i ($i = 1, 2, \cdots, n$) denotes the class label ("1" if positive, otherwise "-1"). Then in an SVM the separating hyperplane is defined as $\langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b = 0$ and the two corresponding parallel hyperplanes are $\langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b = 1$ for the positive class and $\langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b = -1$ for the negative class, where $\mathbf{w}$ is the vector perpendicular to the separating hyperplane, b is a scalar, $\phi(\mathbf{x})$ denotes the pre-defined input vector mapping function of the SVM, and $\langle \cdot, \cdot \rangle$ denotes the inner product operator. For classification, if a test vector $\mathbf{x}_t$ satisfies $\langle \mathbf{w}, \phi(\mathbf{x}_t) \rangle + b > 0$, it will be classified as a positive instance; otherwise, if it satisfies $\langle \mathbf{w}, \phi(\mathbf{x}_t) \rangle + b < 0$, it will be classified as a negative instance. An SVM tries to find the optimal $\mathbf{w}$ and b to maximize the margin.

Since the margin in an SVM is always equal to $\frac{2}{||\mathbf{w}||_2}$ no matter what input vectors are, maximizing the margin is equivalent to minimizing $||\mathbf{w}||_2$, further, equivalent to minimizing $||\mathbf{w}||_2^2$. Therefore, an SVM primal formulation in the hard margin cases is defined in Eqn. 2.1. Here, *hard margin* means that the training data can be classified perfectly without any classification error.

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \parallel \mathbf{w} \parallel_2^2 \\ \text{s.t.} \quad & \forall i, \ y_i \Big[\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle + b \Big] \geq 1 \end{aligned} \tag{2.1}$$

In practice, training data is hardly ever classified properly without any error. To allow the classification errors by relaxing the margin condition from hard margin to *soft margin*, slack variables ξ are introduced into the SVM formulation above and a soft margin SVM is defined in Eqn. 2.2, where C is a pre-defined non-negative constant to control the trade-off between the margin and the classification errors.

$$\begin{aligned} \min_{\mathbf{w}, b, \xi} \quad & \frac{1}{2} \parallel \mathbf{w} \parallel_2^2 + C \sum_i \xi_i \\ \text{s.t.} \quad & \forall i, \ y_i \Big[\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle + b \Big] \geq 1 - \xi_i \\ & \xi_i \geq 0, \ C \geq 0 \end{aligned} \tag{2.2}$$

Based on the Lagrange multipliers α, Eqn. 2.2 can be rewritten as a dual formulation as shown in Eqn. 2.3, and $\mathbf{w}$ and b can be calculated using Eqn. 2.4 and 2.5, where N_S is the total number of support vectors S. A *support vector* is an input vector $\mathbf{x}_i$ with $\alpha_i \neq 0$.

Notice that $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$ is an element of a *kernel*, which is a definite semi-positive matrix. Eqn. 2.3 can be solved using quadratic programming (QP).

$$\begin{aligned} \max_{\alpha} \quad & \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \\ \text{s.t.} \quad & \forall i,\ 0 \le \alpha_i \le C, \quad \sum_i \alpha_i y_i = 0 \end{aligned} \tag{2.3}$$

$$\mathbf{w} = \sum_i \alpha_i y_i \phi(\mathbf{x}_i) \tag{2.4}$$

$$b = \frac{1}{N_S} \sum_{i \in S} \Big[y_i - \langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle \Big] = \frac{1}{N_S} \sum_{i \in S} \Big[y_i - \sum_{j \in S} \alpha_j y_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \Big] \tag{2.5}$$

In the *multi-class* cases, where more than two classes exist, in general there are two strategies to train an SVM based on the binary cases, that is, one-vs-one and one-vs-rest. Given N classes, the one-vs-one strategy trains $\frac{N(N-1)}{2}$ binary SVMs to distinguish any pair of classes, whereas the one-vs-rest strategy trains N binary SVMs, one for each class. For classification, different criteria can be used, *e.g.* max-voting [33], max-margin [15], etc. Also, multi-class SVMs can be trained for all the class models simultaneously (*all-in-one* for short). For instance, Weston and Watkins [67] proposed a multi-class SVM as defined in Eqn. 2.6, where $y_i = 1, 2, \cdots, N$ denotes the true class label for $\mathbf{x}_i$. Its basic idea is that the decision margin between an input and its true class should be bigger than that between it and any false class.

$$\begin{aligned} \min_{\mathbf{w}, \mathbf{b}, \xi} \quad & \frac{1}{2} \sum_{c=1}^{N} \| \mathbf{w}_c \|_2^2 + C \sum_{i, c \neq y_i} \xi_{i,c} \\ \text{s.t.} \quad & \forall i, c \neq y_i, \ \Big[\langle \mathbf{w}_{y_i}, \phi(\mathbf{x}_i) \rangle + b_{y_i} \Big] - \Big[\langle \mathbf{w}_c, \phi(\mathbf{x}_i) \rangle + b_c \Big] \ge 1 - \xi_{i,c} \\ & \xi_{i,c} \ge 0,\ C \ge 0 \end{aligned} \tag{2.6}$$

Hsu and Lin [33] compared the one-vs-one and one-vs-rest strategies, and addressed that their performances are highly dependent on the classification problem at hand, and in their experiments, one-vs-one is slightly better in most cases. Further, Demirkesen and Cherifi [17] compared the three strategies above for scene image classification, and drew a conclusion that the all-in-one strategy outperforms the other binary class based strategies based on their experiments.

Chapter 3

Maximum Similarity Based Feature Matching

In this chapter, we propose two feature matching approaches to maximize the similarities between images. We believe that the maximum image similarity is a good measure for object recognition because two images with similar objects in the same categories are incline to have a higher maximum similarity than two images with objects in different categories. Fig. 3.1 gives an example to explain the basic idea of our maximum similarity based feature matching approaches. By matching the patches in images, intuitively the maximum similarity between the two bike images should be higher than that between the upper bike image and the car image, because the two bike images contain similar bike instances, while the upper bike image and the car image do not. In this way, it can be inferred which images more likely contain the same-category objects by calculating pair-wise maximum image similarity, and object recognition can be carried out using some classifiers like K-nearest neighbors (KNN), support vector machines (SVMs), etc.

Accordingly we propose the following two feature matching criteria for maximizing image similarities: Section 3.1 explains a constrained global feature correspondence approach, Section 3.2 addresses a probabilistic feature matching approach. Finally Section 3.3 summarizes the chapter.

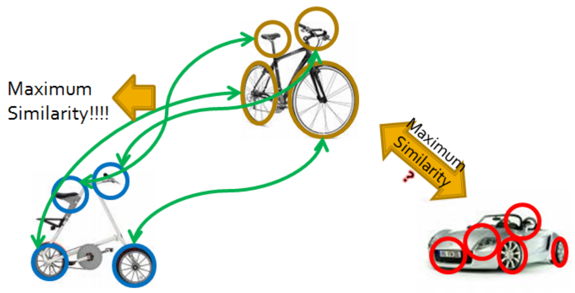


Figure 3.1: Illustration of the intuition of the maximum similarity measure.

3.1 Constrained Global Feature Correspondence

In this section, we consider the feature correspondence task as a graph matching problem. Our approach tries to maximize a similarity objective function, consisting of not only the feature vectors but also the constrained global spatial structures of two sets of feature points, by a new polynomial-time approximate optimization algorithm. This algorithm allows every node in a smaller graph to potentially be linked with any node in a larger graph, and thus it can handle one-to-one, many-to-one, and no match cases. We test this approach on the "hotel" and "house" video sequences for feature matching, and our results are quite impressive.

3.1.1 Introduction

Feature correspondence aims to find a sequence of feature matching pairs between two feature sets to satisfy certain conditions. Finding a correct feature correspondence is difficult, because in two sets of feature points (i) the descriptor for each point may be quite different, even for pairs of ground truth matching points; (ii) the locations of the points may be changed by translation, rotation, scaling, etc.; (iii) some points may appear in a set but disappear in the other set. In general, the computational complexity of the feature correspondence problem is NP-hard [26].

To deal with these difficulties, many approaches have been proposed. Koeser et al. [38]

employed affine features to estimate the parameters for conjugate rotation. Dellaert et al. [16] applied a Markov Chain Monte Carlo approach to estimate the posterior distribution over all possible feature correspondences instead of best-one matching scheme. Torresani et al. [59] formulated the feature correspondence task as a graph matching problem by minimizing an energy objective function based on the feature vectors and *local* spatial structures. Similarly, Caetano et al. [13] utilized graph learning algorithms to improve feature correspondence accuracy based on local structures of the graphs.

We also consider the feature correspondence task as a graph matching problem. Specifically, in order to maximize the number of the feature matching pairs, in this paper we define the feature correspondence task as follows:

Definition 1 (**Maximum Similarity Based Feature Correspondence**). *Given two sets of features* A=$\{a_1, a_2, \cdots, a_{|A|}\}$ *with* $|A|$ *features, and* B=$\{b_1, b_2, \cdots, b_{|B|}\}$ *with* $|B|$ *features (*$|A| \leq |B|$*), a maximum similarity based feature correspondence is defined as an asymmetric matching process from the smaller set* A *to the bigger set* B *so that (i) each feature in* A *appears at most once in the matching pairs to allow one-to-one, many-to-one, and no match cases, and (ii) the similarity between* A *and* B *is maximized.*

Naturally, minimizing an energy function is equivalent to maximizing a similarity function, and here we adopt similarity maximization, which involves not only the feature vectors but also their constrained global spatial structures (CGSS), to find the feature correspondence. For many feature correspondence cases, the object instances in images have their specific global spatial configurations, *e.g.* the configurations of their parts or the relative locations between them. In these cases, CGSS will be quite helpful because it is designed based on this type of global spatial structure information. Because of computational complexity, we propose a polynomial-time approximate algorithm using a greedy strategy to search for the maximum iteratively as well as handling the different matching situations.

3.1.2 Similarity Objective Function

Many research works suggest that for feature correspondence problems the best approach is to combine the feature vectors with structure information. However, most of them focus on the *local* structures of feature points (*e.g.* [59, 13]). Here, we introduce a new feature structure called *constrained global spatial structure* (CGSS). For each feature point, a CGSS is constructed by the normalized distances and the normalized angles between all the matched features in the matching sequence, and thus the CGSS is less sensitive to geometric transformations, *i.e.* translation, rotation and scaling. Further, we take the similarities between CGSS as a part of our similarity measure defined in Eqn. 3.1 for matching feature points between set A and set B, and the optimal feature correspondence is the one that maximizes the similarity between A and B, as defined in Eqn. 3.2.

$$\mathbb{S}(A,B) = \sum_{i=1}^{|A|} S_i = \sum_{i=1}^{|A|} k(v_i|f)^w k(s_i|f)^{1-w} p(f) \tag{3.1}$$

$$\mathbb{F}(A,B) = \arg\max_f \mathbb{S}(A,B) \tag{3.2}$$

Here, $\mathbb{S}(A,B) \in [0,1]$ and $\mathbb{F}(A,B)$ respectively denote the similarity and the optimal feature correspondence between set A and set B ($|A| \leq |B|$), f denotes any possible feature correspondence, v denotes a feature vector, s denotes a CGSS determined by f, $p(f)$ denotes the prior knowledge of occurrence of f modeled as a uniform distribution, $k(v_i|f)$ and $k(s_i|f)$ denote the feature similarity and structure similarity of the i^{th} matching pair in f, whose multiplication is proportional to the similarity score of the matching pair, S_i, and w ($0 \leq w \leq 1$) is a parameter controlling the trade-off between the two similarities for S_i. In general, structure s can be designed based on different purposes, and in our algorithm we design it as the normalized distances and the normalized angles between the feature points, as illustrated in Fig. 3.3 for a_1 and b_2.

The intuition of multiplication of the feature similarity, the CGSS similarity, and the feature correspondence occurrence prior is inspired by the graphical model shown in Fig. 3.2, where the feature matching pair v and their corresponding CGSS s can be considered as two attributes of f, and the probability of co-occurrence of f, s and v is the multiplication of $p(f)$, $k(v|f)$ and $k(s|f)$. Since in our formulation the values of the feature similarities and the structure similarities are between 0 and 1, we simply take similarity as the equivalence of probability.

In [5], Berg et al. proposed a similar formulation to ours for finding feature correspondence. However, the major difference between these two formulations is that in [5] the

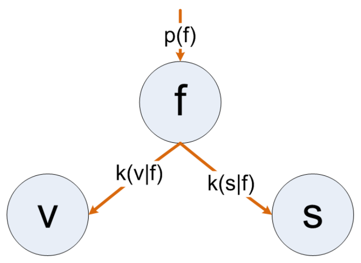


Figure 3.2: Intuition of multiplication of the feature similarity, the global feature structure similarity, and the feature correspondence occurrence prior.

feature structure cost is pair-wise, two features per set, and the mapping between these features from one set to the other defines the structure cost; whereas in our approach, the CGSS is defined based on all the previous matched features in a set, not limited to pair-wise. In other words, the definitions of feature structures in these two formulations are quite different, and to our best knowledge, introducing the *constrained global spatial structure (CGSS) similarity* into the matching process is novel.

3.1.3 Approximate Algorithm for Maximization

Notice that our similarity function is convex, and therefore its maximization is unique given set A and set B. However, this function is difficult to maximize because the CGSS s is related to all the matched feature points in the feature set. One way forward is to apply an exhaustive search to all possible feature correspondences, but its computational complexity is too high, given by $O(\frac{|B|!}{(|B|-|A|+1)!})$.

Inspired by the Markov Chain Monte Carlo (MCMC) technique, we propose a polynomial-time approximate algorithm to increase the similarity score monotonically and iteratively. Basically, there are two steps in our algorithm: initialization and re-matching.

Initialization

The goal of the initialization is to select a good candidate for f from all possible candidates available for searching for the maximum so that the computational time is reduced. In this

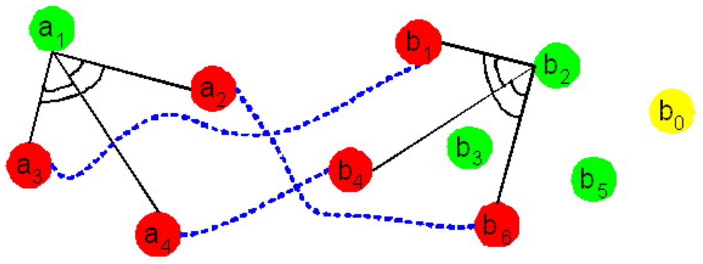


Figure 3.3: An example of feature correspondence between set A with 4 points and set B with 6 points. Here, green dots represent the unmatched points in each set, red dots represent the matched point pairs connected by blue dotted lines, and the yellow dot b_0 is for no match cases which does not belong to B. The constrained global spatial structure for point a_1 (reps. b_2) consists of the normalized distances and the normalized angles between a_1 (reps. b_2) and each matched point in the set.

step, we tentatively match the feature points in A and B such that each point in A has a match in B, since $|A| \leq |B|$. Here, we utilize the Iterated Conditional Modes (ICM) [7] criterion for initialization.

Specifically, ICM allows a point pair, which has the maximum similarity score S based on the matched pairs before, to be added into the current matching sequence in each iteration. We describe each selection process as follows:

$$f_i = \arg\max_{\hat{f}_i} k(v_i|\hat{f})^w k(s_i|\hat{f})^{1-w} p(\hat{f}) \tag{3.3}$$

where f_i ($i = 2, 3, \cdots, |A|$) denotes the i^{th} point pair which is added into the current matching sequence $\hat{f}$ consisting of $i-1$ point pairs, and $\hat{f}_i$ denotes any possible point pair candidate. Other variables have the same meanings as those in Eqn. 3.1. For the first matching pair in $\hat{f}$, we simply add the one with the maximum feature similarity into $\hat{f}$.

Example 1 (Initialization). In Fig. 3.3, suppose we have matched 3 point pairs (red dots) between A and B, and would like to find a match for a_1 amongst b_2, b_3, and b_5 (green dots). Now we calculate the similarity score between a_1 and b_2. Their feature similarity $p(v_i|\hat{f})$ is calculated directly based on their feature vectors. The *constrained global spatial structure* (CGSS) of a_1 is (1) the normalized distances, which are the normalization of the distances between a_1 and a_2, a_3, a_4; and (2) the normalized angles, which are the normalization of the angles between a_1 and any pair of the matched point pairs. For b_2, its CGSS is defined in the same way. Then their structure similarity

$k(s_i|\hat{f})$ can also be calculated directly. Notice that for each point, the CGSS has its own order, and these orders must be consistent with each other. For instance, if the order of the normalized distances for a_1 is $\langle a_1, a_2 \rangle$, $\langle a_1, a_3 \rangle$ and $\langle a_1, a_4 \rangle$, then for b_2 the calculation order must be $\langle b_2, b_6 \rangle$, $\langle b_2, b_1 \rangle$ and $\langle b_2, b_4 \rangle$. It is similar for calculating the normalized angles. Thus, using Eqn. 3.3 the similarity score for the pair a_1 and b_2 can be calculated easily. After finishing the calculation of the similarity score for each candidate pair, we select the pair with the maximum as f_i. In sum, our approximate matching optimization allows including a measure of global spatial information without the penalty of an exhaustive or another slow search.

Re-matching

After initialization, we re-match each point in A with the points in B iteratively, using the exhaustive search while keeping the other pairs in the current feature correspondence f unchanged, so that the similarity defined by Eqn. 3.1 will increase. We maintain the current global structure by re-mapping a single node in turn whilst keeping the graph unchanged, and thus impose severe consistency constraints. The purpose of this step is to correct any "wrong" matching pair. In this process, we add a new point b_0 (yellow dot) into B for handling the no match cases, as illustrated in Fig. 3.3. When a point in A matches b_0, the similarity score of this pair is equal to 0, and if in this case the similarity of the matching sequence increases, then that point should have no matching point in B currently. In other cases, we calculate the similarity score for each pair of the point candidates, and when we find a new point correspondence which increases the similarity of the matching sequence, we update the matching sequence and its similarity score. In this way, our re-matching process can handle the one-to-one, many-to-one and no match cases.

Example 2 (Re-matching). In Fig. 3.3, suppose after initialization the feature correspondence is $\langle a_1, b_5 \rangle$, $\langle a_2, b_6 \rangle$, $\langle a_3, b_1 \rangle$ and $\langle a_4, b_4 \rangle$, and the corresponding similarity of this feature correspondence is s_0. Now we would like to re-match a_1 with any point in B plus b_0 so that s_0 can increase. Suppose we try to match a_1 with b_2; then to calculate the similarity between a_1 and b_2, we keep the other matched pairs $\langle a_2, b_6 \rangle$, $\langle a_3, b_1 \rangle$ and $\langle a_4, b_4 \rangle$ unchanged, and calculate the normalized distances and the normalized angles between a_1 and a_2, a_3, a_4 for a_1, and between b_2 and b_1, b_4, b_6 for b_2. This calculation is repeated for $\langle a_2, b_6 \rangle$, $\langle a_3, b_1 \rangle$ and $\langle a_4, b_4 \rangle$ respectively, and using Eqn. 3.1 we can calculate the similarity of the current feature correspondence $\langle a_1, b_2 \rangle$, $\langle a_2, b_6 \rangle$, $\langle a_3, b_1 \rangle$ and

$\langle a_4, b_4 \rangle$, denoted as s_1. If $s_1 > s_0$, then we update the feature correspondence to be the current one along with its similarity, and otherwise keep searching. Overall, this is a highly constrained and fast method.

As an approximation algorithm, we also enforce each increase amount to be no less than ϵ ($\epsilon > 0$), with our algorithm terminating if there is no such increase amount in $|A|(|B|+1)$ searches. Since our global feature structure similarities are highly dependent on the previous matched feature pairs, our approximation algorithm may be stuck in local solutions due to the greedy strategy. Therefore, a good feature correspondence candidate in the initialization stage will be quite useful, not only for avoiding local solutions but also for reducing computational time. Below is a proposition which proves that our approximate algorithm is polynomial-time.

Proposition 1. The computational complexity of our approximate optimization algorithm is $O(|A|^2|B|)$ ($|A| \leq |B|$).

Proof. Let S_{max} and S_{init} denote the maximum of $\mathbb{S}(A, B)$ and its value after initialization, and suppose that in T searches the maximum increase amount ΔS is found. Then because $S_{max} \leq |A|$, $S_{init} \geq 0, T \leq |A|(|B|+1)$, and $\Delta S \geq \epsilon$, we can estimate the computational complexity of our approximate algorithm as follows.

$$O\left(\frac{(S_{max} - S_{init})T}{\Delta S}\right) \leq O\left(\frac{|A| \cdot |A|(|B|+1)}{\epsilon}\right) = O(|A|^2|B|)$$

Thus, our approximate algorithm is polynomial-time. $\square$

To summarize our algorithm, we show the pseudo code of our algorithm in Alg. 3.1.

3.1.4 Experiments

We tested our approach on the "hotel" video sequence[1] used in [59, 13] consisting of 101 frames of a toy hotel, and the "house" video sequence[2] used in [13] consisting of 111 frames

[1]This sequence is downloaded from `http://vasc.ri.cmu.edu/idb/html/motion/hotel/index.html`.

[2]This sequence is downloaded from `http://vasc.ri.cmu.edu/idb/html/motion/house/index.html`.

```
Algorithm 3.1: Polynomial-time Approximate Algorithm for Maximization
Input: set $A$, set $B$
Output: feature correspondence $\mathbb{F}$
Initialize $\mathbb{F}$ based on Iterated Conditional Modes (ICM) using Eqn. 3.3 so that each feature
in $A$ appears once in the matching pairs;
repeat
    Calculate the similarity $S$ for a possible matching pair using Eqn. 3.1;
    if $\mathbb{S}(A, B)$ increased then Update $\mathbb{F}$;
    else Continue to search exhaustively;
until No increase of $\mathbb{S}(A, B)$ exists;
return $\mathbb{F}$
```

of a toy house. We employed the Shape Context[3] [4] descriptor to generate a feature vector for each point. To measure the similarity of two feature vectors or their corresponding CGSS, we used the RBF-kernel with the χ^2 distance. For instance, given two feature vectors v_1 and v_2 with d dimensions, their feature similarity $k(v_1, v_2)$ is defined in Eqn. 3.4.

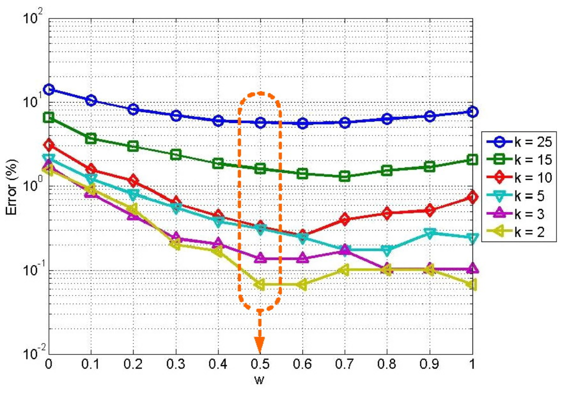
$$k(v_1, v_2) = \exp\left\{-\sum_{i=1}^{d} \frac{(v_{1,i} - v_{2,i})^2}{v_{1,i} + v_{2,i}}\right\} \tag{3.4}$$

In our experiments, for each sequence we take as ground truth[4] the same 30 point-pairs as in [59, 13]. Our goal is to find the feature correspondence between two frames with frame interval k, where $k \in \{25, 15, 10, 5, 3, 2\}$. That is, the frame-order distance between two frames in the sequence is equal to k. For example, if $k = 2$, we will find the feature correspondence for the frame pairs with order $(1, 3)$, $(2, 4)$, $\cdots$, $(N-3, N-1)$, $(N-2, N)$, where N is the total number of the frames in the sequence.

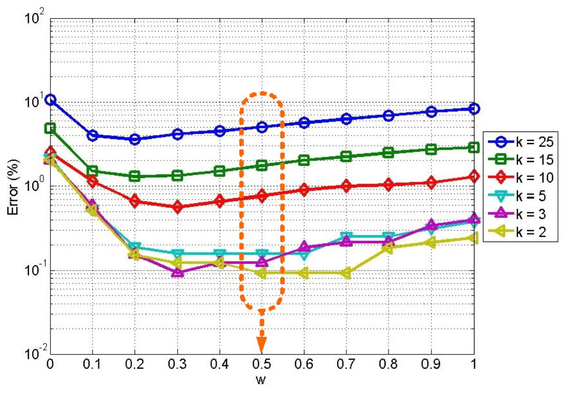
We fixed the parameter ϵ to 0.02 during all the experiments, and tested our approach using different values of w from 0 to 1 step by 0.1 on the two sequences, as shown in Fig. 3.4 for "hotel" (left) and "house" (right). Clearly (and unsurprisingly) our approach works better with smaller k: when $k = 2$ our best results are 0.07% for "hotel" and 0.09% for "house", truly excellent results. When $k = 25$ our best results are 5.53% and 3.57%,

[3]Since in [59, 13] Shape Context is employed, we also use it. Other descriptors can be also utilized here.

[4]We appreciate help from Drs. Lorenzo Torresani and Vladimir Kolmogorov in providing us with point-pair ground truth for the "hotel" and "house" sequences.



(a) Hotel



(b) House

Figure 3.4: Mismatching percentages (%) vs. parameter w using different k intervals for (a) "Hotel" and (b) "House" sequences.

Table 3.1: Performance comparison between $w = 0.5$ and the best w for different k intervals (%).

Sequence	hotel						house					
k =	2	3	5	10	15	25	2	3	5	10	15	25
Perf. with best w	0.07	0.10	0.17	0.26	1.28	5.53	0.09	0.09	0.16	0.56	1.28	3.57
Perf. with w=0.5	0.07	0.14	0.31	0.33	1.59	5.66	0.09	0.12	0.16	0.76	1.74	5.00

respectively, as listed in Table 3.1. This is quite reasonable, because when k is small, the relative positions of the points are little changed, leading to a better description of Shape Context and more similarity of the CGSS of points, compared to larger k. Even for $k = 25$, a very wide gap in time, the results are quite respectable.

Also we make another two observations: (i) The errors first decrease from $w = 0$ and then increase when w is close to 1, with the best results for w around 0.5. This indicates that both feature vectors and their corresponding CGSS contribute to finding the optimal feature correspondence; (ii) Around $w = 0.5$, all the lines are relatively flat, which means in that range the performance of our approach is less sensitive to the changes in w. Therefore, inferred from these two sequences, our approach needs no training data in order to select optimal value for w, and we can simply fix $w = 0.5$ when we do not have any prior knowledge about the data, since fixing w is the only learning aspect to the algorithm. For these two sequences, we compare performance with $w = 0.5$ to the best performance, in Table 3.1. From the table, we see that the differences are quite small between a fixed w

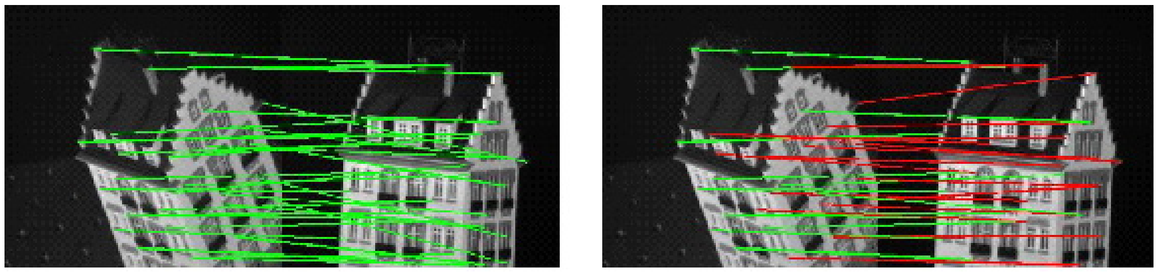


Figure 3.5: Our result for matching the first and last frames ($k = 100$) of the "hotel" sequence, where green lines indicate the correct feature correspondences, and the red lines indicate the wrong ones. *Left*: Ground truth of the feature correspondences predefined in the two images. *Right*: Our result (accuracy: $14/30 = 46.7\%$) with $w = 0.5$.

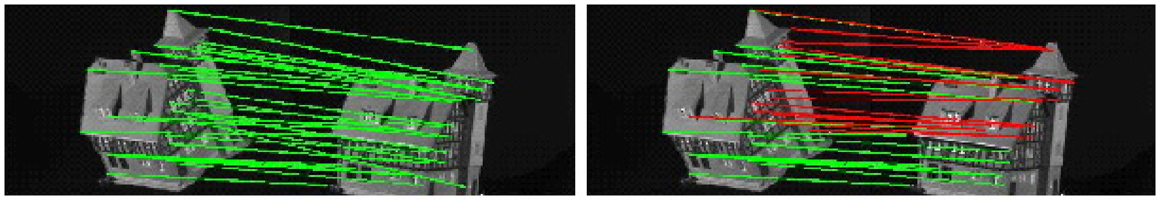


Figure 3.6: Our result for matching the first and last frames ($k = 110$) of the "house" sequence, where green and red lines have the same meanings as Fig. 3.5. *Left*: Ground truth of the feature correspondences predefined in the two images. *Right*: Our result (accuracy: $12/30 = 40.0\%$) with $w = 0.5$.

and the best w. Also, these results are comparable to those in [59, 13], although we cannot directly compare since the values of k for frame pairs are not stated there. Comparing the computational time, on average for matching a frame pair based on our Matlab implementation without any optimization, it takes 1.24 and 1.22 seconds on "hotel" and "house", respectively, on a 2.33GHz Core 2 Duo CPU, while in [59] their "DD" approach needs nearly 30 seconds.

We also tested our approach to find the feature correspondence between the *first* and *last* frames in each sequence — a very challenging test. Here we used a constant $w = 0.5$. The results are shown in Fig. 3.5 for "hotel" ($k = 100$) and Fig. 3.6 for "house" ($k = 110$), where the correspondence accuracies are $14/30 = 46.7\%$ and $12/30 = 40.0\%$ for "hotel" and "house", respectively. This is because the changes of the relative positions of the points make it difficult for the Shape Context descriptor to identify the same points. However, compared with the result of the linear assignment without learning in [13] for "house" (7/30), we have achieved significant improvement. Although the matching accuracy of the constrained global feature correspondence approach is respectable, some issues make it difficult to apply this approach to object recognition on large-size datasets, *e.g.* its asymmetric matching and scalability, and we do not perform any experiment on object recognition using this approach.

3.2 Probabilistic Feature Matching

In this section we propose another novel feature matching approach called *Probabilistic Feature Matching* (PFM). We consider the matching process as the bipartite graph matching problem, and define the image similarity as the inner product of the feature similarities and their corresponding matching probabilities, which are calculated by optimizing a quadratic formulation. Further, we prove that the image similarity and the sparsity of the feature matching probability distribution will decrease monotonically with the increase of parameter C in the quadratic formulation where $C \geq 0$ is a pre-defined data-dependent constant. Essentially, our approach is the generalization of a family of similarity measurement approaches. We test our approach on Graz datasets for object recognition, and achieve 89.4% on Graz-01 and 87.4% on Graz-02, respectively on average.

3.2.1 Introduction

We consider each image as an undirected graph and take the feature matching process as the bipartite graph matching problem as illustrated in Fig. 3.7, where any pair of features from two images could be possibly matched.

Different strategies can be utilized in the feature matching process. Lyu [46] introduced the Summation Kernel (SK) to measure the image similarities as follows:

$$K_{sum}(V_1, V_2) = \sum_{v_i \in V_1} \sum_{v_j \in V_2} k(v_i, v_j) \tag{3.5}$$

where V_1 (resp. V_2) denotes a feature set, $v_i \in V_1$ (resp. $v_j \in V_2$) denotes a feature vector in V_1 (resp. V_2), and $k(v_i, v_j)$ denotes an arbitrary feature similarity kernel. In the next

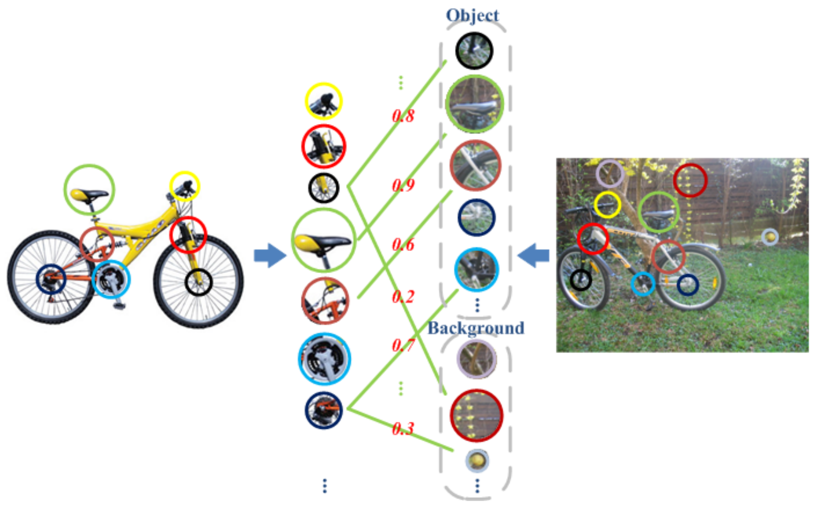


Figure 3.7: Illustration of matching two images. Each image is represented as a collection of features of the patches. Weights (red) on the edges (green) denote the matching probabilities between the feature pairs so that the similarity between the two images is obtained.

sections, we denote $k(v_i, v_j)$ as k_{ij} for short. Wallraven *et al.* [65] proposed the Max-selection Kernel (MK) as shown below:

$$K_{max}(V_1, V_2) = \frac{1}{2}\left\{\sum_{v_i \in V_1} \max_{v_j \in V_2} k_{ij} + \sum_{v_j \in V_2} \max_{v_i \in V_1} k_{ji}\right\} \tag{3.6}$$

Fröhlich *et al.* [22] proposed the Optimal Assignment Kernel (OAK) to maximize the similarity score between two structured objects by finding exactly one-to-one matches between the parts of these objects, defined as follows:

$$K_{OA}(x, y) = \begin{cases} \max_\pi \sum_{i=1}^{|x|} k(x_i, y_{\pi(i)}) & if\ |y| > |x| \\ \max_\pi \sum_{j=1}^{|y|} k(x_{\pi(j)}, y_j) & otherwise \end{cases} \tag{3.7}$$

where x (resp. y) denotes an object, x_i (resp. y_j) denotes a part of x (resp. y), $|x|$ (resp. $|y|$) denotes the total number of the parts of x (resp. y), and π denotes a permutation of parts.

In contrast, the novel contribution of this approach is that we introduce a probabilistic matching strategy into the matching process as illustrated in Fig. 3.7, and we show that our approach can be considered as the generalization of a family of similarity measurement approaches, including SK, MK, and OAK, so that the similarity can be decided adaptive to the data. In our approach, the similarity between two images is defined as the inner product of their feature similarities and the corresponding feature matching probabilities, which are calculated by optimizing a quadratic formulation.

3.2.2 Image Similarity Function

Given two images $X = \{x_1, \cdots, x_{|X|}\}$ and $Y = \{y_1, \cdots, y_{|Y|}\}$, where $x_i \in X$ (resp. $y_j \in Y$) denotes a feature in X (resp. Y) and $|X|$ (resp. $|Y|$) denotes the total number of features in X (resp. Y), according to the bipartite graph matching problem, their similarity can be defined as follows:

$$S(X, Y; \alpha, \mathbf{k}) = \sum_{i=1}^{|X|} \sum_{j=1}^{|Y|} \alpha_{ij} k_{ij} \tag{3.8}$$

where α_{ij} denotes the *feature matching probability* (FMP) between features x_i and y_j, k_{ij} denotes their similarity, and $S(X, Y; \alpha, \mathbf{k})$ denotes the similarity between X and Y given the feature matching probability function α (see Section 3.2.3 for details) and their feature similarity matrix $\mathbf{k}$.

3.2.3 Feature Matching Probability Function

Intuitively, an FMP α_{ij} can be utilized to describe how likely feature x_i and y_j are matched. As illustrated in Fig. 3.7, the axle with black circle on the left has 0.8 FMP with the axle with black circle on the right, while it has 0.2 FMP with the background feature, which is quite reasonable. Therefore, by considering the matching process as a function, we propose a very useful concept, called feature matching probability function, and give its definition as follows:

Definition 2 (**Feature Matching Probability Function**). *Given two images* $X = \{x_1, \cdots, x_{|X|}\}$ *and* $Y = \{y_1, \cdots, y_{|Y|}\}$*, a feature matching probability function (FMPF)* α *is defined as* $\alpha : X \times Y \rightarrow \left\{\{0\} \bigcup \mathbb{R}^+\right\}_{|X| \times |Y|}$*, where* $\mathbb{R}^+$ *denotes the field of positive real numbers. Letting* $\overrightarrow{x}$ *and* $\overrightarrow{y}$ *denote the two dimensions of* α*, and selecting an arbitrary dimension set*

$\mathcal{H} \subseteq \{\overrightarrow{x}, \overrightarrow{y}\}$ *from* α*, each FMPF will correspond to a point in the vector space covered by the following convex set:*

$$\left\{\alpha \,|\, \sum_{\forall h \in \mathcal{H}} \alpha \preceq \mathbf{1},\ \sum_{i,j} \alpha_{ij} = \min\left(|X|, |Y|\right),\ \mathbf{0} \preceq \alpha \preceq \mathbf{1}\right\}$$

where "$\preceq$*" denotes the element-wise operator of "*$\leq$*".*

Notice that if $\mathcal{H} = \emptyset$, the first constraint in the convex set above does not apply. In the constraint set of α, the first constraint guarantees that the total matching probability of *each* corresponding feature is no more than 1, the second constraint guarantees that the total matching probability of *all* the features is equal to the total number of features in the smaller set, and the last constraint guarantees that each element in α represents a probability.

Actually, we can use this concept to describe the matching processes in SK, MK, and OAK. Specifically, (i) in SK any pair of features could be matched and their matching weight is fixed as 1, so α can be modeled as a uniform distribution; (ii) in MK a unique feature is selected as a match for every feature in both feature sets and their matching weight is fixed as 1, others 0, totally $|X| + |Y|$ matched feature pairs; (iii) in OAK each feature in both sets can occur in the matched feature pairs at most once with the matching weight equal to 1, others 0, totally $\min\{|A|, |B|\}$ matched feature pairs. In these approaches, the weights are pre-defined and kept the same during the matching process, no matter what the data is. In our approach, α is calculated automatically by optimizing a quadratic formulation so that α can be adaptive to the different data, which could reflect the matching relationship between features better. As a result, our approach can be considered as the generalization of SK, MK, and OAK as explained in next section.

3.2.4 Probabilistic Feature Matching Optimization

We would like to perform the probabilistic feature matching between two images automatically. Therefore, we propose a quadratic optimization formulation [49] as defined in Eqn. 3.9 to calculate α, where $f(\alpha; C)$ denotes our objective function, α is the only variable,

C is a pre-defined non-negative constant, and $\mathbf{k}$ is the feature similarity matrix.

$$\begin{aligned} \max_{\alpha} \quad & f(\alpha; C) = \sum_{i,j} \alpha_{ij} k_{ij} - C \sum_{i,j} \alpha_{ij}^2 \\ \text{s.t.} \quad & \sum_{\forall h \in \mathcal{H}} \alpha \preceq \mathbf{1}, \sum_{i,j} \alpha_{ij} = \min\left(|X|, |Y|\right), \mathbf{0} \preceq \alpha \preceq \mathbf{1}, C \geq 0 \end{aligned} \tag{3.9}$$

In our objective function, the first part is to measure the image similarity, the second part is to measure the sparseness of the distribution of α, and $C \geq 0$ is a pre-defined data-dependent constant to control the trade-off between the two parts. Usually, *sparseness* is a measure from $\mathbb{R}^n$ to $\mathbb{R}$ for a vector to quantify how much energy is actually packed into only a few components [32]. In our case, since the total energy is fixed, we define the sparseness of a distribution as the variance of the distribution. Therefore, to minimize the sparseness of α, we have

$$\min\left\{SP(\alpha)\right\} = \min\left\{Var(\alpha)\right\} = \min\left\{E[\alpha^2] - E[\alpha]^2\right\} \Leftrightarrow \min\left\{E[\alpha^2]\right\} \tag{3.10}$$

where $SP(\cdot)$ denotes the sparseness of a distribution, and $E[\cdot]$ and $Var(\cdot)$ denote the mean and variance operators.

In order to see the relationship between our approach and some other similarity measurement approaches, we need the following important theorems on convexity [49]:

Theorem 1. *Consider* $\max f(x)$ *over* $x \in \mathcal{X}$, *where* $f(x)$ *is convex, and* $\mathcal{X}$ *is a closed convex set. If the optimum exists, a boundary point of* $\mathcal{X}$ *is the optimum.*

Theorem 2. *If a convex function* $f(x)$ *attains its maximum on a convex polyhedron* $\mathcal{X}$ *with some extreme points, then this maximum is attained at an extreme point of* $\mathcal{X}$.

Based on the theorems above, we can show that in certain cases our approach can be considered as equivalences to SK, MK and OAK by choosing different C and H in Eqn. 3.9.

- $C = +\infty$ and $\mathcal{H} = \{\overrightarrow{x}, \overrightarrow{y}\}$: According to Thm. 1, the optimized α in Eqn. 3.9 will be a uniform distribution, that is, $\alpha_{ij} = \frac{1}{\max(|X|,|Y|)}$, and by re-scaling α to $\mathbf{1}$, our approach can be considered as an equivalence to SK [46].

- $C = 0$ and $\mathcal{H} = \{\overrightarrow{x}, \overrightarrow{y}\}$: According to Thm. 2, the optimized α in Eqn. 3.9 will simulate a one-to-one matching process without feature re-occurrence, and our approach is equivalent to OAK [22].

- $C = 0$ and $\mathcal{H} = \{\overrightarrow{x}\}$: According to Thm. 2, the optimized α will simulate the matching process that selects the biggest similarity along the $\overrightarrow{x}$-dimension for each feature in the $\overrightarrow{y}$-dimension, and the corresponding similarity is equivalent to $\sum_j \max_i k_{ij}$ in Eqn. 3.6. Thus, by optimizing Eqn. 3.9 along the $\overrightarrow{x}$ and $\overrightarrow{y}$-dimension of α, respectively, our approach is equivalent to MK [65].

Moreover, our approach has the following property:

Proposition 2. For two images X and Y, both the sparseness of α and their similarity $S(X, Y; \alpha, \mathbf{k})$ will decrease monotonically with increasing C in Eqn. 3.9.

Proof. Considering $C_1 > C_2 \geq 0$ and their corresponding α_1 and α_2 calculated using Eqn. 3.9, we have $f(\alpha_1; C_1) \geq f(\alpha_2; C_1)$ and $f(\alpha_2; C_2) \geq f(\alpha_1; C_2)$. Putting them together, we have

$$C_1\alpha_2'\alpha_2 - C_1\alpha_1'\alpha_1 \geq \alpha_2'\mathbf{k} - \alpha_1'\mathbf{k} \geq C_2\alpha_2'\alpha_2 - C_2\alpha_1'\alpha_1 \tag{3.11}$$

where α_1, α_2 and $\mathbf{k}$ are vectorized, and $'$ denotes the transpose operator. Then we get

$$(C_1 - C_2)(\alpha_2'\alpha_2 - \alpha_1'\alpha_1) \geq 0 \tag{3.12}$$

Since $C_1 > C_2 \geq 0$, then $\alpha_1'\alpha_1 \leq \alpha_2'\alpha_2$, which indicates that a smaller C will lead to an α with larger sparseness. Besides, we have

$$S(X, Y; \alpha_2, \mathbf{k}) - S(X, Y; \alpha_1, \mathbf{k}) = \alpha_2'\mathbf{k} - \alpha_1'\mathbf{k} \geq C_2(\alpha_2'\alpha_2 - \alpha_1'\alpha_1) \geq 0 \tag{3.13}$$

Therefore, $S(X, Y; \alpha, \mathbf{k})$ will decrease monotonically with the increase of C. $\square$

This property simplifies the adjustment of C in the cross-validation for different data so that our approach can be adaptive to the data. For instance, based on this proposition and the optimization of Eqn. 3.9 in both $\overrightarrow{x}$ and $\overrightarrow{y}$-dimension of α, we know that when $C = +\infty$ the similarities between images are the smallest (equivalent to the normalization of SK) and when $C = 0$ their similarities are the biggest (equivalent to OAK). By increasing (or decreasing) the value of C monotonically using the cross-validation techniques, we can find a good parameter value for the recognition purpose.

3.2.5 Classification with Support Vector Machines

In general, there is no guarantee that the similarity matrix generated by our approach is a valid kernel, whereas theoretically support vector machines (SVMs) are utilized with kernels for classification. However, in practice, an arbitrary similarity matrix can be involved in an SVM by adding a small positive number to the entries along the diagonal when it is not valid (*e.g.* [69]), as did in Eqn. 3.14, where $|\lambda_{\min}|$ denotes the absolute value of the minimum eigenvalue of the similarity matrix K, and $\mathbf{I}$ denotes the identity matrix.

$$K' = K + |\lambda_{\min}|\mathbf{I}, \quad \text{if } \lambda_{\min} < 0 \tag{3.14}$$

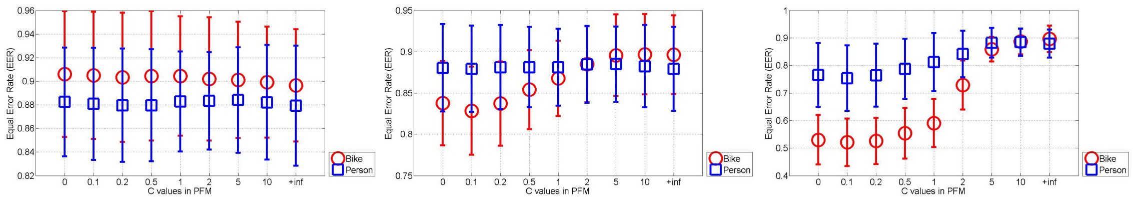
3.2.6 Experiments

We tested our approach on the Graz-01 [51] and Graz-02 [52] datasets to perform the "object & non-object" binary classification, with performance measured by Equal Error Rate (EER). Graz-01 is a challenging dataset with two object categories (bike: 373 images, person: 460 images) and a background category (270 images), because they vary greatly in object scale, pose and illumination. Compared to Graz-01, Graz-02 can be considered as an improved version with much more challenge, and comprises 3 object categories (bike: 365 images, person: 311 images, car: 420 images) and a background category (380 images). The size of each image in both datasets is either 640×480 or 480×640 pixels.

In our experiments, all the images were converted into gray scale, and we utilized the dense sampling technique [60] to sample the images so that each patch consists of 10×10 pixels. For each patch, we employed the SIFT [45] descriptor to represent it, and then used K-means to generate a codebook with 200 codewords so that each descriptor can be represented by the closest codeword in the feature space. Finally, by counting the occurrence of each codeword in the cells of the 3×3 grid, we created 9 histograms to represent each image. The RBF-kernel with χ^2 distance was used to compare the similarity of two histograms, that is,

$$k_{ij} = \exp\left\{ -\sum_{n=1}^{d} \frac{(v_{i,n} - v_{j,n})^2}{v_{i,n} + v_{j,n}} \right\} \tag{3.15}$$

where d is the number of dimensions of histograms v_i and v_j. The regularization parameter in an SVM was fixed to 10^4. All the results here were averaged after 50 runs. To simplify



(a) PFM_1 (b) PFM_2 (c) PFM_3

Figure 3.8: Performance comparison on Graz-01 between different PFM with different C.

the notations, we use PFM_1, PFM_2 and PFM_3 to denote our approach with $\mathcal{H} = \{\overrightarrow{x}, \overrightarrow{y}\}$, $\mathcal{H} = \{\overrightarrow{x}\}$ or $\mathcal{H} = \{\overrightarrow{y}\}$, and $\mathcal{H} = \emptyset$, respectively.

Graz-01

Table 3.2: Comparison results between different approaches on Graz-01 (%)

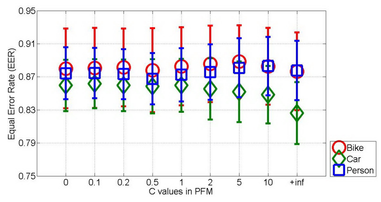
	Bike	Person	Ave.
SPK [41]	86.3±2.5	82.3±3.1	84.3
PDK [44]	90.2±2.6	87.2±3.8	88.7
PFM_1 (C=0)	***90.6±5.3***	88.2±4.6	***89.4***
PFM_2 (C=5)	89.6±4.9	***88.5±4.6***	89.0
PFM_3 (C=+∞)	89.6±4.8	87.9±5.1	88.8

For the training-test data selection, we followed the setup in [41]. Specifically, we randomly selected 100 images in the positive class and 50 in each negative class (including the background) as our training set, and performed the test on similarly distributed data sets consisting of half the number of the training images per category.

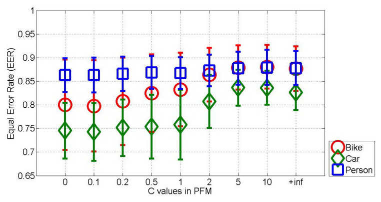
Fig. 3.8 shows our performance on Graz-01. In general, PFM_1 performs best, while PFM_3 performs worst, and PFM_1 is much more stable with the increase of C than the others. We also list the best performance of each PFM in Table 3.2 and compare them with some other published results. Clearly, all of our results outperform them.

Graz-02

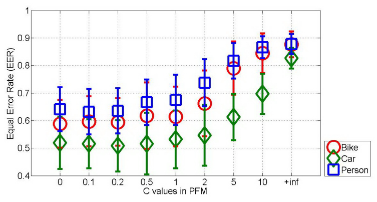
We followed the experimental setup in [52] for the training-test data selection. Specifically, for each object category, we randomly selected 150 positive and 150 negative (50 for each



(a) PFM$_1$



(b) PFM$_2$



(c) PFM$_3$

Figure 3.9: Performance comparison on Graz-02 between different PFM with different C.

Table 3.3: Comparison results between different approaches on Graz-02 (%)

	Bike	Person	Car	Ave.
Boost.+SIFT [52]	76.0	70.0	68.9	71.6
Boost.+Comb. [52]	77.8	81.2	70.5	76.5
PDK+SIFT [44]	86.7	86.7	74.7	82.7
PDK+hybrid [44]	86.0	87.3	74.7	82.7
PFM$_1$+SIFT (C=5)	***88.9***	***88.1***	***85.2***	***87.4***
PFM$_2$+SIFT (C=10)	88.0	87.9	83.6	86.5
PFM$_3$+SIFT (C=$+\infty$)	87.7	87.8	82.6	86.0

non-object class, including the background) images as the training data, and selected 75 positive and 75 negative (25 for each non-object class, including the background) with similar distribution of the training data as the test data, respectively.

Fig. 3.9 shows our performance on Graz-02. Compared to Fig. 3.8, similar observations can be made. Therefore, $\mathcal{H} = \{\overrightarrow{x}, \overrightarrow{y}\}$ seems the best choice among the three for our PFM. Also, Table 3.3 lists the best results using different PFM in comparison with some other published results, and all of ours outperform them significantly.

3.3 Summary

In this chapter, we propose two different criteria for matching the same-type features in images to maximize their similarities.

First, we formulate the feature correspondence task as a graph matching problem, and propose a similarity function involving both feature vector similarities as well as corresponding constrained global spatial structure similarities. That we can include this global structure information in optimal feature correspondence is due to the introduction of a new polynomial-time approximation maximization algorithm, which updates the similarity score iteratively. While allowing many potential matches, the update is strictly constrained by the global structures of features, making for a fast algorithm compared to some state-of-the-art graph matching algorithms with comparable accuracies as demonstrated by our experiments.

However, this approach defines an asymmetric matching process which limits its application in object recognition, particularly on large-size datasets. Therefore, we propose another novel feature matching approach called probabilistic feature matching (PFM). In this approach, the similarity between two images is defined as the inner product between the feature similarities and their corresponding matching probabilities, which are calculated data-dependently by solving a quadratic optimization problem. We also prove that the image similarity and the sparseness of the feature matching probability distribution will decrease monotonically with the increase of parameter C in the quadratic formulation. Essentially, our approach is the generalization of a family of similarity measurement approaches, including the Summation Kernel, the Max-selection Kernel, and the Optimal Assignment Kernel. In our experiments, we tested our approach on Graz datasets for object recognition. On average, we achieved 89.4% on Graz-01 and 87.4% on Graz-02, respectively.

Chapter 4

Adaptive Multiple Kernel Learning

Usually humans recognize objects based on their different sources of information rather than one [18]. Recently, impressive accuracies for object recognition in images have been achieved on several benchmark datasets by combining multiple sources of object information. In this chapter, we propose a novel multiple kernel learning (MKL) approach, called *Adaptive Multiple Kernel Learning* (AdaMKL)[1], for information combination based on the max-margin criterion. Unlike other traditional MKL approaches, our objective function is defined as a family of biconvex functions with an arbitrary ℓ_p-norm ($p \geq 1$) of kernel coefficients to learn the weights for support vectors and the kernel coefficients at a local optimum. In our learning algorithm, AdaMKL minimizes our objective function alternatively by updating some variables while fixing the others at a time, where only one convex formulation needs to be solved. Besides, AdaMKL is suitable for either binary-class learning or multi-class learning. These characteristics make the learning of AdaMKL much more efficient.

We test our approach on the Caltech datasets for object recognition. Our experiments demonstrate that training AdaMKL is much (at least 10 times) faster than training some convex optimization based MKL algorithms with better accuracies. On Caltech-101, our mean category accuracies are $(67.2 \pm 1.4)\%$ and $(74.4 \pm 0.5)\%$ using 15 and 30 training images per category, respectively, and on Caltech-256, ours are $(31.4 \pm 1.0)\%$ and $(36.4 \pm$

[1]Part of this work has been accepted by ICPR'10 [71].

$1.7)\%$, all of which are comparable to the best published results.

The rest of the chapter is organized as follows. Section 4.1 introduces our approach and reviews some related work. Section 4.2 explains the details of AdaMKL. Section 4.3 gives our experimental results on the Caltech datasets and compares AdaMKL with some other approaches. We finally summarize the chapter in Section 4.4.

4.1 Introduction

We propose a novel multiple kernel learning (MKL) approach to combine different information for object recognition in images, called Adaptive Multiple Kernel Learning (AdaMKL). To minimize our objective function, AdaMKL takes an alternative optimization strategy to learn the weights for support vectors and the kernel coefficients by updating some variables while fixing the others at a time, where only one convex formulation needs to be solved. At each iteration, the global optimum of the convex formulation is a local optimum of our biconvex objective formulation, which makes our approach "adaptive" to the maximum margin little by little.

Different from many other MKL approaches, our AdaMKL has the following characteristics, which make the learning process of AdaMKL much more efficient.

- Its objective function is defined as a family of biconvex, rather than convex, functions, and thus its optima are local instead of global. A function $f : X \times Y \to \mathbb{R}$ is called **biconvex** if f is convex both in $x \in X$ for fixed $y \in Y$ and in $y \in Y$ for fixed $x \in X$.
- An arbitrary ℓ_p-norm ($p \geq 1$) of kernel coefficients can be involved in the objective function, rather than the constraint set, for optimization. In this way, the traditional sparse (or non-sparse) learning due to the norm constraint is separated.
- Its kernel coefficients can be learned for either binary-class or multi-class cases. In binary-class learning, each object category has its own kernel coefficients, while in multi-class learning all the categories share the same coefficients.

4.1.1 Related Work

Many approaches have been proposed to combine different information of objects for recognition or detection. For example, Gu *et al.* [31] proposed an approach to learn the weights of regions in images for object recognition using their texture, color, and shape information. Another useful approach is multiple kernel learning (MKL), which has been demonstrated by many papers (*e.g.* [62, 39, 25, 63]). Varma and Ray [62][2] proposed an MKL approach to learn the optimal trade-off between different descriptors for a given classification task. Lampert and Blaschko [39] utilized a subsequent decision process that works jointly for all given object classes to combine the efficiency of single class localization using MKL. Gehler and Nowozin [25] extended MKL from finite kernels to the infinite kernels to learn kernel parameters for the pre-processing steps in classification. Vedaldi *et al.* [63] applied MKL to generate a three-stage classifier as an object detector.

Multiple Kernel Learning (MKL) [1] aims to generate an optimal kernel automatically by combining a set of basic kernels linearly as well as learning the weights for support vectors simultaneously. Given N labeled data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1,\cdots,N}$ where $\forall i, \mathbf{x}_i \in \mathcal{X}$ is an input feature and $y_i \in \{+1, -1\}$ is its class label for binary classification, and M feature mapping functions $\Phi = \{\phi_j\}_{j=1,\cdots,M}$, each of which maps an input feature into a Hilbert space $\mathcal{H}_j$, a reproducing kernel K_j is defined as an inner product of $\phi_j(\mathbf{x}_m)$ and $\phi_j(\mathbf{x}_n)$, that is, $K_j(\mathbf{x}_m, \mathbf{x}_n) = \langle \phi_j(\mathbf{x}_m), \phi_j(\mathbf{x}_n) \rangle_{\mathcal{H}_j}$, and the optimal kernel K_{opt} learned by MKL can be represented as Eqn. 4.1, where γ are *kernel coefficients*.

$$K_{opt} = \sum_{j=1}^{M} \gamma_j K_j, \quad \forall j,\ \gamma_j \geq 0 \tag{4.1}$$

By putting MKL into the context of max-margin classifiers, *e.g.* support vector machines (SVMs), several MKL approaches have been proposed [1, 40, 55, 58, 36, 37, 54]. A common existing framework in the MKL approaches is to minimize a convex optimization based objective function [47], and below Eqn. 4.2 shows a general case of MKL by

[2]We notice that the results on Caltech-101 in this paper are incorrect. For more information please refer to `http://research.microsoft.com/en-us/um/people/manik/projects/trade-off/trade-off.html`.

introducing an arbitrary norm of kernel coefficients into the constraint set [37].

$$\begin{aligned}
\min_{\gamma,\mathbf{w},b,\xi} \quad & \frac{1}{2}\sum_m \frac{\| \mathbf{w}_m \|_2^2}{\gamma_m} + C\sum_i \xi_i \\
\text{s.t.} \quad & \forall i, m,\ y_i\left[\sum_m \langle \mathbf{w}_m, \phi_m(\mathbf{x}_i)\rangle + b\right] \geq 1 - \xi_i, \\
& \gamma_m \geq 0,\ \| \gamma \|_p^p \leq 1,\ p \geq 1,\ \xi_i \geq 0,\ C \geq 0
\end{aligned} \tag{4.2}$$

Note that if $p = 1$, it results in sparse learning of γ like [1, 40, 55, 58, 54]; or if $p = 2$, it results in non-sparse learning like [36]. Also, different optimization approaches have been used in MKL, *e.g.* second-order cone programming [1], semi-definite programming [40], semi-infinite programming [58, 36], gradient-based approaches [55, 37, 54]. Many recent MKL solvers (*e.g.* [58, 37, 54, 62, 55]) adopt a two-step strategy to learn the parameters alternatively. For learning $\mathbf{w}$, the regular SVM formulation with a kernel is taken advantage of, whereas for learning γ their algorithms may be quite different to handle the norm constraint of kernel coefficients in the constraint set, especially for the arbitrary-norm cases. The convergence of the strategy above is not guaranteed in general [55], and how to learn the kernel coefficients γ is more crucial than how to learn the weights for support vectors.

One of the biggest issues of applying MKL to object recognition is how to learn the parameters in MKL efficiently due to the huge information in images. In [39], Lampert and Blaschko proposed to minimize the upper bound of the original formulation of MKL in order to gain the learning efficiency as well as keep good performance. This method actually can be considered as searching for a local optimum of the original MKL formulation instead of the global one as the solution.

Based on this consideration, by relaxing the convexity of the objective function, we propose our biconvex MKL approach based on the max-margin criterion, named *Adaptive Multiple Kernel Learning* (AdaMKL). The intuition of this approach is to make the learning of kernel coefficients much more easier by involving an arbitrary ℓ_p-norm ($p \geq 1$) of kernel coefficients with our objective function, which helps hide the norm constraint of kernel coefficients in the dual formulation when learning $\mathbf{w}$ without consideration explicitly. Besides, the learning of AdaMKL is suitable for either binary-class or multi-class cases. Accordingly, a two-step optimization algorithm is proposed to solve our problem locally.

4.2 Adaptive Multiple Kernel Learning

Adaptive Multiple Kernel Learning (AdaMKL) aims to learn an optimal kernel efficiently by combining multiple kernels as well as achieving good performance on recognition tasks. AdaMKL searches for the local optimal solutions rather than the global ones, which makes learning faster and easier.

4.2.1 Motivation

Eqn. 4.3 shows one type of the decision functions in MKL (*e.g.* [27]), where θ are the coefficients *w.r.t.* kernels, $\mathbf{w}$ are the normal vectors for an SVM, and b is the bias term.

$$f(\mathbf{x}) = \sum_{m=1}^{M} \theta_m \langle \mathbf{w}_m, \phi_m(\mathbf{x}) \rangle + b \tag{4.3}$$

Suppose $\theta_m \geq 0$ and $\| \theta \|_p^p \leq 1$, then we make some modifications to Eqn. 4.3, that is, letting $\mathbf{w}'_m = \sqrt{\theta_m}\mathbf{w}_m$ and $\phi'_m(\mathbf{x}_i) = \sqrt{\theta_m}\phi_m(\mathbf{x}_i)$. Further, we fit $\mathbf{w}'$ and $\phi'_m(\mathbf{x})$ with Eqn. 4.3 into the binary-class SVM formulation (see Section 2.4 for details). It turns out that the corresponding formulation is exactly the same as Eqn. 4.2 [37]. However, by fitting Eqn. 4.3 directly into the original binary-class SVM formulation to minimize the max-margins between positive and negative instances, we can obtain Eqn. 4.4, where C is a constant and ξ are errors.

$$\begin{aligned} \min_{\theta,\mathbf{w},b,\xi} \quad & \frac{1}{2}\sum_m \theta_m^2 \| \mathbf{w}_m \|_2^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & \forall i,\ y_i \left[\sum_m \theta_m \langle \mathbf{w}_m, \phi_m(\mathbf{x}_i) \rangle + b \right] \geq 1 - \xi_i \\ & \xi_i \geq 0,\ C \geq 0 \end{aligned} \tag{4.4}$$

Essentially Eqn. 4.4 defines a biconvex optimization problem (OP), and inspired by [39] its objective function can be easily relaxed to one of its upper bounds as shown in Eqn. 4.5, which also defines a biconvex OP. Next, we will show that these upper bound formulations allow us to introduce an arbitrary ℓ_p-norm ($p \geq 1$) of kernel coefficients into AdaMKL.

$$\sum_m \theta_m^2 \| \mathbf{w}_m \|_2^2 \leq \max_m \theta_m^2 \sum_m \| \mathbf{w}_m \|_2^2 \leq \| \theta^2 \|_p \| \mathbf{w} \|_2^2 \quad \leq \quad \| \theta^2 \|_1 \| \mathbf{w} \|_2^2 \tag{4.5}$$

4.2.2 Binary-class AdaMKL

A binary-class AdaMKL (B-AdaMKL) aims to learn parameters $\mathbf{w}$, θ and b for each individual class without sharing any information between them. Using the upper bounds of Eqn. 4.4, we define the primal of B-AdaMKL as Eqn. 4.6 where $p \geq 1$ and c denotes a specific class. Apparently, our objective formulation contains a family of biconvex objective functions due to different p, where parameters are learned alternatively.

$$\begin{aligned} \min_{\theta_c, \mathbf{w}_c, b_c, \xi} \quad & \frac{1}{2} \| \theta_c^2 \|_p \| \mathbf{w}_c \|_2^2 + C \sum_i \xi_i \\ \text{s.t.} \quad & \forall i,\ y_i \left[\sum_m \theta_{c,m} \langle \mathbf{w}_{c,m}, \phi_m(\mathbf{x}_i) \rangle + b_c \right] \geq 1 - \xi_i \\ & \xi_i \geq 0,\ C \geq 0 \end{aligned} \tag{4.6}$$

When learning parameters $\mathbf{w}_c$, b_c, and ξ_i, parameter θ_c is fixed. Based on the Lagrange multipliers, we rewrite our primal in Eqn. 4.6 and obtain its dual in Eqn. 4.7, where α_c is the set of the Lagrange multipliers. Accordingly, we can calculate each $\mathbf{w}_{c,m}$ and b_c using Eqn. 4.8 and 4.9, respectively, where N_S is the total number of support vectors S.

$$\begin{aligned} \max_{\alpha_c} \quad & \sum_i \alpha_{c,i} - \frac{1}{2} \sum_{i,j} \alpha_{c,i} \alpha_{c,j} y_i y_j \left[\sum_m \frac{\theta_{c,m}^2}{\| \theta_c^2 \|_p} K_m(\mathbf{x}_i, \mathbf{x}_j) \right] \\ \text{s.t.} \quad & \forall i,\ 0 \leq \alpha_{c,i} \leq C, \quad \sum_i \alpha_{c,i} y_i = 0 \end{aligned} \tag{4.7}$$

$$\mathbf{w}_{c,m} = \sum_i \alpha_{c,i} y_i \frac{\theta_{c,m}}{\| \theta_c^2 \|_p} \phi_m(\mathbf{x}_i) \tag{4.8}$$

$$b_c = \frac{1}{N_S} \sum_{i \in S} \left\{ y_i - \sum_{j \in S} \alpha_{c,j} y_j \left[\sum_m \frac{\theta_{c,m}^2}{\| \theta_c^2 \|_p} K_m(\mathbf{x}_i, \mathbf{x}_j) \right] \right\} \tag{4.9}$$

Notice that if we denote $\gamma_{c,m} = \frac{\theta_{c,m}^2}{\|\theta_c^2\|_p}$, then we actually obtain $\| \gamma_c \|_p = 1$ ($p \geq 1$). In this way, an arbitrary ℓ_p-norm of kernel coefficients is hidden perfectly in the dual formulation of our B-AdaMKL when learning $\mathbf{w}$, and it needs no consideration in the constraint set as usual.

Next, we learn parameters θ_c, b_c, and ξ_i while keeping parameter $\mathbf{w}_c$ unchanged, and the corresponding primal formulations are shown in Eqn. 4.10, where $\phi_m^{\mathbf{w}_c}(\mathbf{x}_i) = \langle \mathbf{w}_{c,m}, \phi_m(\mathbf{x}_i) \rangle$.

Methods	L$_p$-norm MKL [37]	B-AdaMKL
Primal formulations	$\min_{\theta,w,b,\xi} \frac{1}{2}\sum_m \frac{\lVert w_m \rVert_2^2}{\theta_m} + C\sum_i \xi_i$ $s.t. \quad \forall i, y_i\left[\sum_m \langle w_m, \phi_m(x_i)\rangle + b\right] \geq 1-\xi_i$ $\theta_m \geq 0, \lVert \theta \rVert_p^p \leq 1$ $\xi_i \geq 0, C \geq 0$	$\min_{\theta,w,b,\xi} \frac{1}{2}\lVert \theta^2 \rVert_p \lVert w \rVert_2^2 + C\sum_i \xi_i$ $s.t. \quad \forall i, y_i\left[\sum_m \theta_m \langle w_m, \phi_m(x_i)\rangle + b\right] \geq 1-\xi_i$ $\xi_i \geq 0, C \geq 0$
Dual formulations when learning w	$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2}\left(\sum_m \left(\sum_{i,j} \alpha_i\alpha_j y_i y_j K_m(x_i,x_j)\right)^{\frac{p}{p-1}}\right)^{\frac{p-1}{p}}$ $s.t. \quad \forall i, 0 \leq \alpha_i \leq C, \sum_i \alpha_i y_i = 0$	$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i\alpha_j y_i y_j \left[\sum_m \frac{\theta_m^2}{\lVert \theta^2 \rVert_p} K_m(x_i,x_j)\right]$ $s.t. \quad \forall i, 0 \leq \alpha_i \leq C, \sum_i \alpha_i y_i = 0$
Convexity	Convex optimization	Biconvex optimization
Norm constraint of kernel coefficients	List in the constraint set in the primal formulation	Hide in the objective function of the dual formulation
Kernel coefficient learning methods	Second order optimization (Newton Descent, cutting planes)	Quadratic programming

Figure 4.1: Comparison between a typical general MKL approach proposed in [37] and B-AdaMKL.

Notice that here we minimize the upper bound of Eqn. 4.6. Using similar equations to Eqn. 4.7, 4.8, and 4.9, we can calculate θ_c and b_c easily.

$$\begin{aligned} \min_{\theta_c,b_c,\xi} \quad & \frac{1}{2} \parallel \theta_c^2 \parallel_1 \parallel \mathbf{w}_c \parallel_2^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & \forall i,\ y_i\left[\sum_m \theta_{c,m}\phi_m^{\mathbf{w}_c}(\mathbf{x}_i) + b_c\right] \geq 1-\xi_i \\ & \xi_i \geq 0,\ C \geq 0 \end{aligned} \tag{4.10}$$

We summarize our learning algorithm for B-AdaMKL in Alg. 4.1, and show the comparison in Fig. 4.1 with a typical general approach proposed in [37]. Clearly, due to the introduction of the biconvex optimization, we do not need to handle the arbitrary norm constraint of the kernel coefficients directly. Actually, we hide this constraint into the dual formulation when learning parameter $\mathbf{w}$. As a result, our biconvex optimization problem can be easily solved only using quadratic programming.

Algorithm 4.1: Alternative Parameter Updating Algorithm for B-AdaMKL

```
Input: K,y
Output: $\alpha_c$,$\theta_c$,$b_c$
Initialize $\theta_c$;
Calculate $\alpha_c$ and $b_c$ using Eqn. 4.7 and Eqn. 4.9;
repeat
    Update $\theta_c$ using Eqn. 4.10;
    Update $\alpha_c$ and $b_c$ using Eqn. 4.7 and Eqn. 4.9;
until Satisfied some conditions;
return $\alpha_c$,$\theta_c$,$b_c$
```

4.2.3 Multi-class AdaMKL

Different from B-AdaMKL, a multi-class AdaMKL (M-AdaMKL) learns θ jointly with all the classes. In this way, each class information can be shared by the others, and it is helpful for improving performance as well as accelerating the learning speed. As shown in Eqn. 4.11, the primal formulation of M-AdaMKL is quite similar to that of B-AdaMKL in Eqn. 4.6, but contains $\mathbf{w}_c$ of all the classes and only one shared θ. Here, y_i denotes the true class label of $\mathbf{x}_i$ rather than a binary label, $\mathbb{I}(y_i = c)$ is an indicator, and $\mathbb{I}(y_i = c) = 1$ if $y_i = c$, otherwise $\mathbb{I}(y_i = c) = -1$.

$$\begin{aligned} \min_{\theta,\mathbf{w}_c,b_c,\xi} \quad & \frac{1}{2} \parallel \theta^2 \parallel_p \sum_c \parallel \mathbf{w}_c \parallel_2^2 + C \sum_{c,i} \xi_{c,i} \\ \text{s.t.} \quad & \forall c, i,\ \mathbb{I}(y_i = c) \left[\sum_m \theta_m \langle \mathbf{w}_{c,m}, \phi_m(\mathbf{x}_i) \rangle + b_c \right] \geq 1 - \xi_{c,i} \\ & \xi_{c,i} \geq 0,\ C \geq 0 \end{aligned} \tag{4.11}$$

We use the same learning strategy in B-AdaMKL to learn parameters $\mathbf{w}_c$, θ and b_c alternatively for M-AdaMKL. However, there will be $|\mathbf{x}| \times |c|$ variables in the dual of Eqn. 4.11, where $|\mathbf{x}|$ and $|c|$ denote the numbers of training data and classes, respectively. So solving this dual problem will be quite memory- and time-consuming for large image datasets.

Again, we prefer to minimize an upper bound of the objective function in Eqn. 4.11 as shown in Eqn. 4.12, and thus learn $\mathbf{w}_c$, b_c, and $\xi_{c,i}$ for each class separately using

Eqn. 4.7, 4.8 and 4.9 while fixing θ.

$$\min\left\{\frac{1}{2}\parallel\theta^2\parallel_p\sum_c\parallel\mathbf{w}_c\parallel_2^2+C\sum_{c,i}\xi_{c,i}\right\}\leq\sum_c\min\left\{\frac{1}{2}\parallel\theta^2\parallel_p\parallel\mathbf{w}_c\parallel_2^2+C\sum_i\xi_{c,i}\right\} \tag{4.12}$$

In order to learn parameter θ, we choose to minimize Eqn. 4.11 with $p = 1$ as did in Eqn. 4.10 while keeping all $\mathbf{w}_c$ unchanged.

4.2.4 Discussion

It is easy to see that AdaMKL shares the same computational complexity of quadratic programming. Since we use biconvex optimization, the solutions for our optimization problems will be the local optima rather than the global ones. For the convergence of AdaMKL, which could be an important termination criterion, we cannot guarantee that every formulation of AdaMKL will converge to its local minimum finally in general [55]. However, letting $g_1(\theta, \mathbf{w}, \xi_1)$ and $g_2(\theta, \mathbf{w}, \xi_2)$ (g_1 and g_2 for short) be the primal objective functions for learning $\mathbf{w}$ and θ in AdaMKL, respectively, we have the following propositions for its convergence.

Proposition 3. If $g_1 = g_2$, then it is guaranteed that AdaMKL will converge to a local minimum monotonically.

Proof. At the t^{th} and s^{th} updates for θ and $\mathbf{w}$, respectively, we have $g_1(\theta^{(t)}, \mathbf{w}^{(s)}, \xi_1^{(s)}) \geq g_1(\theta^{(t)}, \mathbf{w}^{(s+1)}, \xi_1^{(s+1)}) = g_2(\theta^{(t)}, \mathbf{w}^{(s+1)}, \xi_1^{(s+1)}) \geq g_2(\theta^{(t+1)}, \mathbf{w}^{(s+1)}, \xi_2^{(t+1)}) = g_1(\theta^{(t+1)}, \mathbf{w}^{(s+1)}, \xi_2^{(t+1)})$. Similar analysis can be made for g_2. Besides, $\forall\theta, \mathbf{w}, \xi_1, \xi_2$, $g_1 \geq 0$, $g_2 \geq 0$, AdaMKL is guaranteed to converge to a local minimum monotonically if $g_1 = g_2$. □

Proposition 4. For hard-margin cases ($C = +\infty$), if g_1 and g_2 can be both optimized (which means $\xi_1 = \xi_2 = \mathbf{0}$) at the initialization stage, then AdaMKL will converge to a local minimum monotonically.

Proof. Notice that both θ and $\mathbf{w}$ parts in g_1 and g_2 are non-negative and have the same monotonicity. So at the t^{th} and s^{th} updates for θ and $\mathbf{w}$, respectively, we have $g_1(\theta^{(t)}, \mathbf{w}^{(s)},$

$\xi_1^{(s)}) \geq g_1(\theta^{(t)}, \mathbf{w}^{(s+1)}, \xi_1^{(s+1)}) \geq g_1(\theta^{(t+1)}, \mathbf{w}^{(s+1)}, \xi_1^{(s+1)})$ and $g_2(\theta^{(t)}, \mathbf{w}^{(s)}, \xi_2^{(t)}) \geq g_2(\theta^{(t)}, \mathbf{w}^{(s+1)}, \xi_2^{(t)}) \geq g_2(\theta^{(t+1)}, \mathbf{w}^{(s+1)}, \xi_2^{(t+1)})$. Besides, $\forall \theta, \mathbf{w}, \xi_1, \xi_2, g_1 \geq 0, g_2 \geq 0$, so AdaMKL will converge to a local minimum monotonically. □

Proposition 5. If g_1 or g_2 converged, then AdaMKL reached a local minimum.

Proof. Without losing generality, suppose at the t^{th} and s^{th} updates for θ and $\mathbf{w}$, respectively, g_2 converged, then $\theta^{(t)} = \theta^{(t+1)}$ and $\xi_1^{(s)} = \xi_2^{(t)} = \xi_2^{(t+1)} = \xi_1^{(s+1)}$. Due to the local convergence, we have $g_1(\theta^{(t)}, \mathbf{w}^{(s)}, \xi_1^{(s)}) = g_1(\theta^{(t+1)}, \mathbf{w}^{(s)}, \xi_2^{(t+1)}) = g_1(\theta^{(t+1)}, \mathbf{w}^{(s+1)}, \xi_1^{(s+1)})$, which indicates AdaMKL has reached a local minimum. Similar analysis can be made if g_1 converged. □

4.2.5 Implementation

In our unoptimized Matlab implementation, we set the initial value of θ as $\mathbf{1}$, like some other convex optimization based MKL approaches did (*e.g.* [54]). Certainly random initialization can also be used for θ in our algorithm, but we prefer the same start point, which always gives us consistent results. All the SVM formulations are solved using LIBSVM [14].

To improve the learning efficiency further for M-AdaMKL, we simplify the θ learning process by setting all b_c to a same value b, as shown in Eqn. 4.13. Still, $\phi_m^{\mathbf{w}_c}(\mathbf{x}_i) = \langle \mathbf{w}_{c,m}, \phi_m(\mathbf{x}_i) \rangle$. In this way, θ can still be learned jointly and shared by all the classes, but only one common SVM solver (LIBSVM) is needed.

$$\begin{aligned} \min_{\theta,b,\xi} \quad & \frac{1}{2} \parallel \theta^2 \parallel_1 \sum_c \parallel \mathbf{w}_c \parallel_2^2 + C \sum_{c,i} \xi_{c,i} \\ \text{s.t.} \quad & \forall c, i,\ \mathbb{I}(y_i = c) \left[\sum_m \theta_m \phi_m^{\mathbf{w}_c}(\mathbf{x}_i) + b \right] \geq 1 - \xi_{c,i} \\ & \xi_{c,i} \geq 0,\ C \geq 0 \end{aligned} \tag{4.13}$$

4.3 Experiments

We test our approach on the Caltech-101 [42] and Caltech-256 [30] datasets for object recognition. In our experiments, every image is resized so that its larger dimension contains

300 pixels but its aspect ratio is maintained. Then we employ GB[3], color GB (C-GB), SIFT, and OpponentSIFT (*O-SIFT* for short, a type of color SIFT) as our descriptors. We predefine 5 scales — 10, 20, 30, 40, 50 pixels — for all the descriptors. For GB and C-GB, the simple Canny edge detector (an inherent function in the Matlab toolbox) is applied to detect the edges in images, and at each scale at most 500 descriptors are extracted, each of which has 204 dimensions. C-GB is a modified version of GB where GB is applied in R, G, B channels, respectively, to extract a collection of 204-dim descriptors. For SIFT and O-SIFT, we employ the software [60][4] to extract descriptors densely spaced by 10 pixels. After this, we utilize K-means to generate 300 codewords for each type of descriptors, and construct spatial pyramid BoW histograms as did in [41] using $L = 0, 1, 2$ for each scale. To create kernels, we apply the histogram intersection kernel [28] at each layer and scale for each type of descriptors, totally $3 \times 5 \times 4 = 60$ kernels. Notice that all the histograms are normalized using l_1-norm before creating kernels. We set parameters $C = 10^3$ in all the learning algorithms used in the experiments without tuning. Each query image is labeled based on the maximum margin using a one-vs-rests strategy, and our classification results shown here are the average mean recognition accuracies across different categories after 3 runs.

4.3.1 Caltech-101

Caltech-101 consists of over 9000 images in 101 object categories plus a background category. In our experiments, we used all 102 categories. The numbers of randomly selected training images per category are 1, 5, 10, 15, 20, 25, 30, respectively, and the rest are taken as query images.

[3]We downloaded the Matlab code from `http://www.cs.berkeley.edu/~aberg/demos/gb_demo.tar.gz`, and used it to extract GB descriptors directly.

[4]This software can be downloaded from `http://staff.science.uva.nl/~ksande/research/colordescriptors/`.

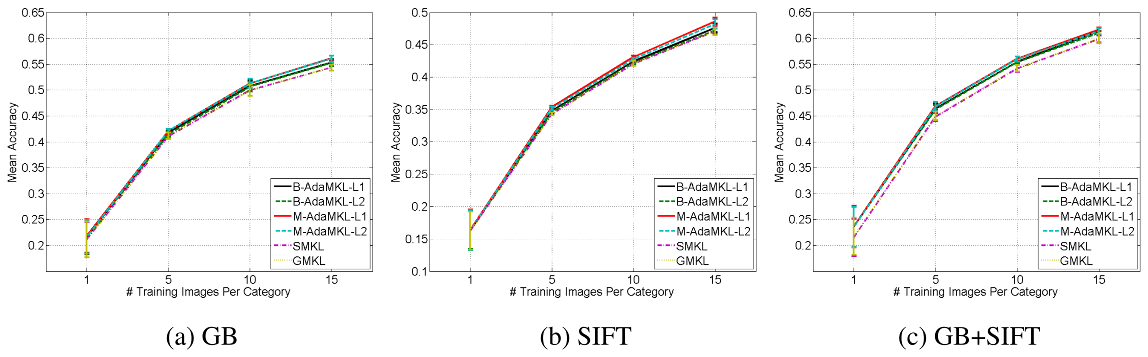


Figure 4.2: MKL comparison on Caltech-101 using descriptors of (a) GB, (b) SIFT, and (c) GB+SIFT. ℓ_1 and ℓ_2 in the figures denote ℓ_1-norm and ℓ_2-norm of kernel coefficients. Clearly, our biconvex optimization based AdaMKL approaches are consistently comparable to SMKL and GMKL, which are both convex optimization based MKL approaches.

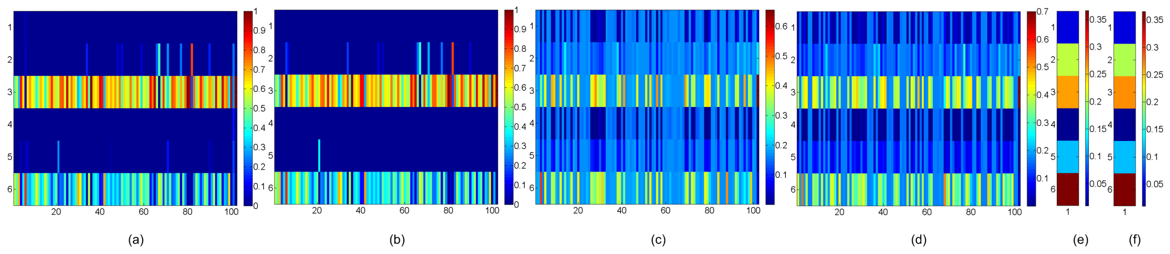
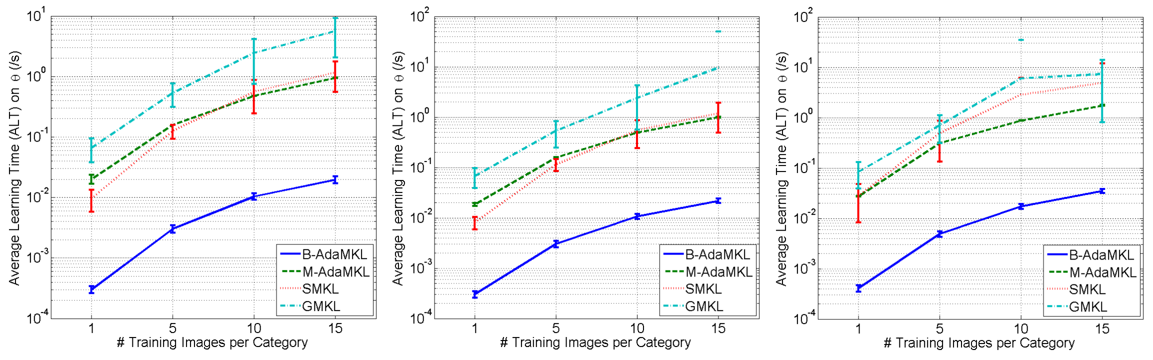


Figure 4.3: An example of comparison of the learned kernel coefficients between (a) SMKL, (b) GMKL, (c) ℓ_1-norm B-AdaMKL, (d) ℓ_2-norm B-AdaMKL, (e) ℓ_1-norm M-AdaMKL, and (f) ℓ_2-norm M-AdaMKL after ℓ_1 normalization using GB+SIFT and 15 training images per category. In (a), (b), (c), and (d), the x-axis denotes the index of each object category in Caltech-101, and the y-axis denotes 10-scale, 20-scale, and 30-scale GB descriptors plus 10-scale, 20-scale, and 30-scale SIFT descriptors in order from top to bottom. In (e) and (f), the figures show the shared kernel coefficients for all the categories.

Comparison with convex optimization based MKL approaches

First of all, let us compare AdaMKL with some other MKL approaches on this dataset. Considering the computational time, here we only use GB and SIFT descriptors at the first scale, and the numbers of training images per category are 1, 5, 10 and 15. In this way, there are only $2 \times 3 = 6$ kernels used for recognition, 3 for GB and SIFT, respectively. We set $p = 1, 2$ in Eqn. 4.6 and 4.11 for B-AdaMKL and M-AdaMKL, called ℓ_1*-norm B-AdaMKL*, ℓ_2*-norm B-AdaMKL*, ℓ_1*-norm M-AdaMKL*, and ℓ_2*-norm M-AdaMKL*, respectively, and employ two recent convex optimization based MKL approaches for comparison



(a) GB (b) SIFT (c) GB+SIFT

Figure 4.4: MKL comparison on Average Learning Time (ALT) in seconds on kernel coefficients on Caltech-101 using descriptors of (a) GB, (b) SIFT, and (c) GB+SIFT. On the curves, the lower parts of standard deviation of GMKL using 15 training images per category in (b) and using 10 training images in (c) are missing, because the values are negative.

purpose: SimpleMKL[5] (SMKL) [54], and General MKL[6] (GMKL) [62, 55].

The comparison results are shown in Fig. 4.2. With the increase of the number of training images, in all three cases our AdaMKL is consistently comparable to SMKL and GMKL. In terms of numbers, our ℓ_1-norm M-AdaMKL performs the best. We also show an example of comparison of the learned kernel coefficients between different approaches in Fig. 4.3. As we see, the learned kernel coefficients using SMKL and GMKL are quite sparse since they used the ℓ_1-norm constraint on kernel coefficients which leads to sparse learning, and most energy concentrates on the 30-scale GB descriptors and 30-scale SIFT descriptors. Compared with these two approaches, our approaches learned the kernel coefficients more smoothly, and the energy is also mainly located on the 30-scale GB descriptors and 30-scale SIFT descriptors. The similar distributions of the learned kernel coefficients between our approaches and SMKL and GMKL indicate that our AdaMKL can learn the kernel coefficients reasonably.

For the computational time, we list the comparison in Table 4.1. All the results are run

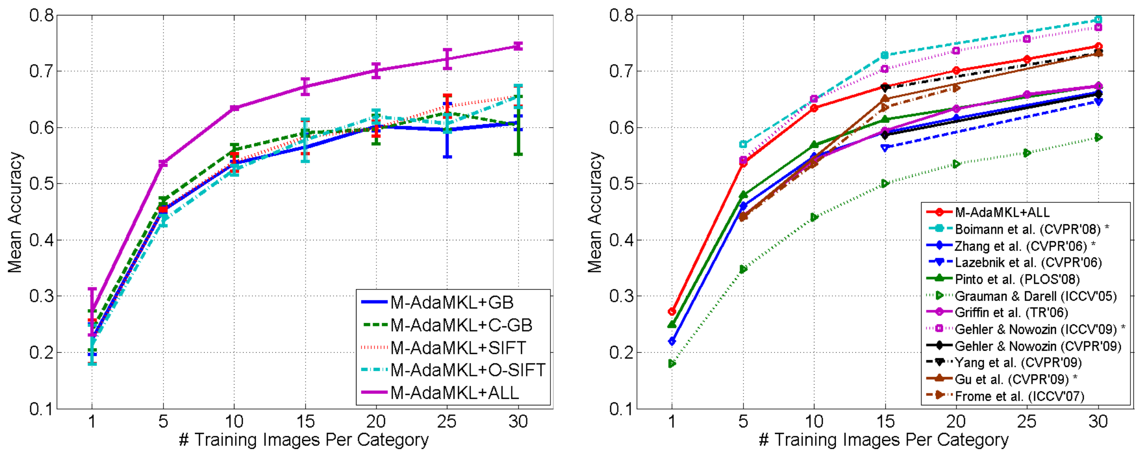
[5]Its Matlab code can be downloaded from `http://asi.insa-rouen.fr/enseignants/~arakotom/code/mklindex.html`.

[6]Its Matlab code can be downloaded from `http://research.microsoft.com/en-us/um/people/manik/code/GMKL/download.html`.

Table 4.1: Comparison between the computational time (/s) on training different MKL approaches using Caltech-101 in the form of (mean ± standard deviation).

Method	GB			
	#1	#5	#10	#15
B-AdaMKL-ℓ_1	0.71±0.01	9.37±0.63	25.64±1.72	42.59±1.15
B-AdaMKL-ℓ_2	0.57±0.01	8.97±0.38	25.98±1.87	43.19±0.12
M-AdaMKL-ℓ_1	0.24±0.01	***3.26±1.14***	10.97±0.03	28.30±5.04
M-AdaMKL-ℓ_2	***0.18±0.03***	3.26±1.15	***10.97±0.01***	***25.30±0.05***
SMKL	6.18±1.60	82.88±1.77	(3.29±1.20)*10^2	(1.30±0.46)*10^3
GMKL	(3.34±0.02)*10^2	(1.67±0.01)*10^3	(5.70±0.73)*10^3	(1.21±0.02)*10^4
	SIFT			
	#1	#5	#10	#15
B-AdaMKL-ℓ_1	0.75±0.03	12.18±0.31	40.59±5.35	72.16±2.39
B-AdaMKL-ℓ_2	0.57±0.03	10.90±0.43	38.87±5.52	73.16±1.79
M-AdaMKL-ℓ_1	0.24±0.03	***4.49±0.01***	15.22±0.03	***32.05±5.18***
M-AdaMKL-ℓ_2	***0.23±0.01***	4.49±0.02	***15.22±0.02***	32.06±5.18
SMKL	5.29±1.43	78.00±24.00	(3.34±1.12)*10^2	(6.15±2.01)*10^2
GMKL	(3.25±0.12)*10^2	(1.62±0.11)*10^3	(5.13±0.24)*10^3	(1.07±0.03)*10^4
	GB+SIFT			
	#1	#5	#10	#15
B-AdaMKL-ℓ_1	0.82±0.05	14.45±0.39	50.00±3.30	84.34±11.07
B-AdaMKL-ℓ_2	0.79±0.05	16.22±0.20	59.81±3.19	(1.07±0.08)*10^2
M-AdaMKL-ℓ_1	0.27±0.00	5.04±0.04	***14.92±2.44***	***37.14±0.07***
M-AdaMKL-ℓ_2	***0.26±0.00***	***5.03±0.04***	14.93±2.47	37.17±0.08
SMKL	9.43±0.48	(2.24±0.15)*10^2	((8.38±0.18)*10^2	(2.96±0.57)*10^3
GMKL	(4.67±0.05)*10^2	(3.30±0.06)*10^3	(1.36±0.04)*10^4	(3.17±0.07)*10^4

on a 2.33GHz Core 2 Duo CPU. Clearly, in each case our AdaMKL is much faster than both SMKL and GMKL over an order of magnitude speedup. As we expect, M-AdaMKL is a little faster than B-AdaMKL because it learns a set of shared kernel coefficients for all the categories simultaneously while the others learn a set for each category one by one. Further, we compare the Average Learning Time (ALT) in seconds on kernel coefficients θ using different descriptors in Fig. 4.4. *Average learning time* is the mean of the learning time per iteration. Some observations can be made from this figure: (i) The learning time on θ in B-AdaMKL is the least; (ii) From the standard deviation at each point, our AdaMKL is much more stable than both SMKL and GMKL; (iii) The curves for B-AdaMKL and



(a) (b)

Figure 4.5: Our results on Caltech-101 using ℓ_1-norm M-AdaMKL. (a) Result comparison based on different descriptors. (b) Result comparison between ours and some other published approaches. From top to bottom, the corresponding references are [11, 69, 41, 53, 28, 30, 24, 25, 68, 31, 23]. Notice that the approaches with the mark "*" used different training and test methods from ours. Precisely, using all the descriptors, ℓ_1-norm M-AdaMKL achieves the mean accuracies of $(27.3\pm4.1)\%$, $(53.6\pm0.4)\%$, $(63.4\pm0.2)\%$, $(67.2 \pm 1.4)\%$, $(70.1 \pm 1.3)\%$, $(72.1 \pm 1.7)\%$, and $(74.4 \pm 0.5)\%$, respectively in order.

M-AdaMKL seem parallel to each other because the only difference between them is the number of instances for learning θ; (iv) With the increase of the numbers of training images and kernels, the learning time is raised as well, but the rising speed of AdaMKL is much slower than the others'. We believe that the main reason for reducing the training time of MKL is that our AdaMKL simplifies the learning process reasonably so that the learning efficiency is improved greatly as well as keeping the comparable recognition accuracy. Considering both accuracies and computational time, ℓ_1-norm M-AdaMKL seems our good choice for the following experiments.

Comparison with some published results

For this purpose, we feed all 60 kernels into our AdaMKL. Since it is difficult to store all the kernels in the limited memory, we organize these kernels in a hierarchy to reduce the memory requirement. Specifically, we first combine the 3 kernels at each scale for each type of descriptors into a new kernel, totally $4 \times 5 = 20$ new kernels, and further combine

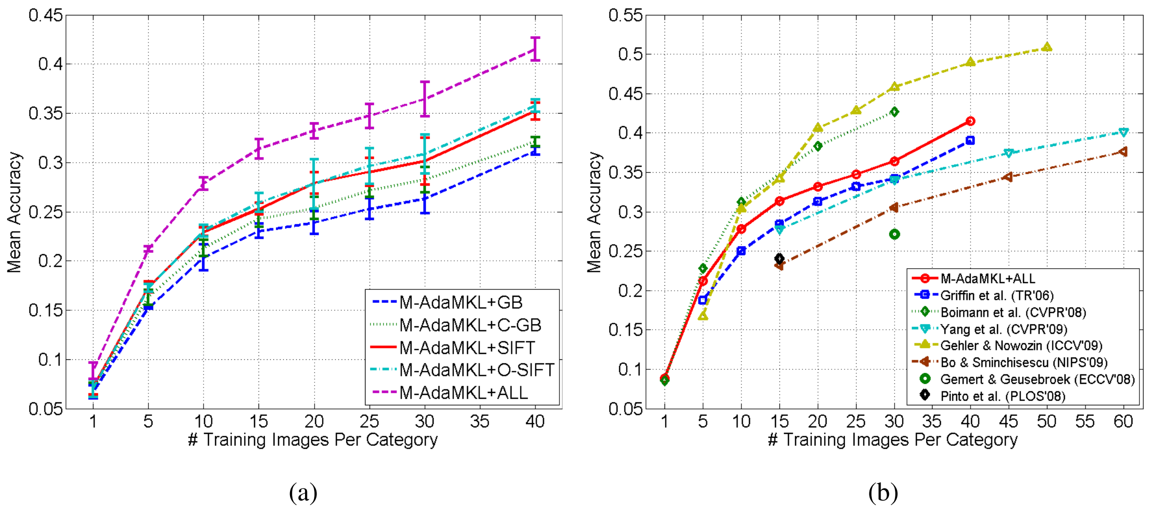


Figure 4.6: Our results on Caltech-256 using ℓ_1-norm M-AdaMKL. (a) Result comparison based on different descriptors. (b) Result comparison between ours and some other published approaches. From top to bottom, the corresponding references are [30, 11, 68, 24, 10, 61, 53]. Precisely, using all the descriptors, ℓ_1-norm M-AdaMKL achieves the mean accuracies of $(8.8\pm0.8)\%$, $(21.2\pm0.3)\%$, $(27.9\pm0.6)\%$, $(31.4\pm1.0)\%$, $(33.2 \pm 0.8)\%$, $(34.7 \pm 1.2)\%$, $(36.4 \pm 1.7)\%$, and $(41.5 \pm 1.2)\%$ respectively in order.

every 5 new kernels with the same type of descriptors to generate 4 kernels, and finally combine these 4 kernels into one kernel which is utilized for recognition. To reduce the computational time, we only apply ℓ_1-norm M-AdaMKL.

Fig. 4.5 (a) shows our results on Caltech-101 using different descriptors, and Fig. 4.5 (b) compares our method with some other published approaches. From Fig. 4.5 (a), we can see that the performances of the four types of descriptors are quite similar. However, by combining them together, the performance is boosted significantly, especially for the case of 30 training images per category. From Fig. 4.5 (b), it is clear that our results are comparable to the best. Interestingly, using fewer training images (≤ 10), our results are quite close to the best two [11, 24], even though their training and test strategies are different from ours, but with the increase of the number of training images the performance is not improved so much. The reason we believe is that the accuracy per category is nearly saturated using 10 training images per category.

4.3.2 Caltech-256

Caltech-256 consists of 30607 images in 256 object categories plus a background category. In our experiments, we use all the categories, and randomly select 1, 5, 10, 15, 20, 25, 30, 40 training images and 25 query images per category as training data and test data, respectively. Still we only utilize ℓ_1-norm M-AdaMKL and feed all 60 kernels to it. Like on Caltech-101, first we show the result comparison based on different descriptors in Fig. 4.6 (a), and compare our best results, which are generated by combining all four types of descriptors, with some other published results in Fig. 4.6 (b). Clearly, our performance is also comparable to the best.

4.4 Summary

In this chapter, Adaptive Multiple Kernel Learning (AdaMKL) is proposed for information combination based on the max-margin criterion with a family of biconvex objective functions which are optimized by updating the parameters alternatively. Instead of searching for global optima, AdaMKL is designed to search for local optima for learning efficiency due to the introduction of an arbitrary ℓ_p-norm ($p \geq 1$) of kernel coefficients into our formulation rather than the constraint set where a traditional MKL puts the norm. Also, AdaMKL is suitable for both binary-class learning (B-AdaMKL) and multi-class learning (M-AdaMKL) cases.

We mainly discuss the convergence issue of AdaMKL, and further compare four specific AdaMKL with two recent convex optimization based MKL approaches in our experiments, which demonstrate that our approach outperforms the two convex optimization based MKL slightly in terms of recognition accuracy with huge gain on training time over an order of magnitude speedup. Finally, the results of ℓ_1-norm M-AdaMKL on Caltech-101 and Caltech-256 for object recognition show that our approach is comparable to some recent published approaches. On Caltech-101, our results are $(67.2 \pm 1.4)\%$ and $(74.4 \pm 0.5)\%$ using 15 and 30 training images per category, respectively, and on Caltech-256, our results are $(31.4 \pm 1.0)\%$ and $(36.4 \pm 1.7)\%$.

Chapter 5

Conclusion

In this thesis, we perform object recognition using maximum similarity based feature matching and adaptive multiple kernel learning. First, in Chapter 1 we introduced the problem of object recognition in images, and proposed two different approaches based on the image similarities for recognizing objects. Then in Chapter 2 we reviewed some background knowledge about object recognition, including low level features, image patch sampling techniques, object representations, and support vector machines for recognition. We explained the details of our maximum similarity based feature matching approaches in Chapter 3, and the adaptive multiple kernel learning approach in Chapter 4.

The contributions of this thesis are:

- We introduced the constrained global spatial structures between low level features into the feature matching process, and accordingly proposed a polynomial-time approximate maximization algorithm to exhaustively search for the optimal feature correspondences which maximize the image similarities.

- We proposed a novel probabilistic feature matching algorithm to maximize the image similarities, where a matching probability is calculated by optimizing a quadratic formulation and then assigned to each pair of matching features. It is guaranteed that with the increase of parameter C in the formulation, the maximum similarity between two images and the sparseness of the matching probability distribution will decrease monotonically. In addition, several proposed feature matching approaches can be considered as special cases of our approach.

- We proposed an efficient multiple kernel learning algorithm, called Adaptive Multiple Kernel Learning (AdaMKL), to combine different visual information for object recognition by optimizing our biconvex objective function locally using an alternative parameter updating strategy. Moreover, an arbitrary norm of kernel coefficients can be easily involved with our objective function. In addition, AdaMKL is suitable for both binary-class and multi-class learning.

Since our approaches are based on the Set-of-Features and Bag-of-Words models, they inherit the drawbacks of these object representations. We achieved good accuracies in the probabilistic feature matching approach for object recognition at the cost of discarding the spatial structures between features. However, sometimes distinctive feature structures are important as recognition clues, and thus they should not be ignored. It will be a part of our future work to introduce structure learning into the probabilistic matching process as well as keeping good accuracy.

For AdaMKL, its termination criterion is a crucial issue because in general there is no guarantee that it will converge to a local optimum monotonically, and different termination criteria may lead to quite different recognition performances. Analyzing what types of local optima are good for recognition performance will be another part of our future work.

Object recognition in images is a fundamental and challenging task in computer vision because there is a huge gap between the low level features of objects and the high level object categories. Our object-based image similarity provides a good intermedium to fill in this gap because it is obtained based on the low level features of objects and also it is the basis for learning and recognizing object categories. In this thesis, we only explore two different approaches based on image similarity, and we believe that they can be improved if more visual information of objects can be utilized. In the future, we will research more along this direction, and hope that one day our techniques could change the world.

Bibliography

[1] Francis R. Bach, Gert R. G. Lanckriet, and Michael I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *ICML'04*, 2004.

[2] H. Bay, T. Tuytelaars, and L.J. Van Gool. SURF: Speeded up robust features. In *ECCV'06*, pages I: 404–417, 2006.

[3] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape context: A new descriptor for shape matching and object recognition. In *NIPS'00*, pages 831–837, 2000.

[4] S.J. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *PAMI*, 24(4):509–522, April 2002.

[5] A.C. Berg, T.L. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *CVPR'05*, pages I: 26–33, 2005.

[6] A.C. Berg and J. Malik. Geometric blur for template matching. In *CVPR'01*, pages I:607–614, 2001.

[7] J. Besag. On the statistical analysis of dirty pictures. *RoyalStat*, B-48(3):259–302, 1986.

[8] David Beymer. Feature correspondence by interleaving shape and texture computations. In *CVPR'96*, pages 921–928, 1996.

[9] Irving Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94:115–147, 1987.

[10] L. Bo and C. Sminchisescu. Efficient match kernel between sets of features for visual recognition. In *NIPS'09*, December 2009.

[11] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *CVPR'08*, 2008.

[12] A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. In *CIVR'07*, 2007.

[13] T. Caetano, L. Cheng, Q. Le, and A. Smola. Learning graph matching. In *ICCV'07*, 2007.

[14] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[15] C. Dance, J. Willamowski, L. Fan, C. Bray, and G. Csurka. Visual categorization with bags of keypoints. In *ECCV'04 Workshop on Statistical Learning in Computer Vision*, 2004.

[16] F. Dellaert, S. Seitz, C. Thorpe, and S. Thrun. Feature correspondence: A markov chain monte carlo approach. In *NIPS'01*, 2001.

[17] C. Demirkesen and H. Cherifi. A comparison of multiclass svm methods for real world natural scenes. In *ACIVS'08*, 2008.

[18] Sven J. Dickinson. Object representation and recognition. In *WHAT IS COGNITIVE SCIENCE*, pages 172–207. Basil Blackwell Publishers, 1999.

[19] B. Epshtein and S. Ullman. Feature hierarchies for object classification. In *ICCV'05*, 2005.

[20] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR'03*, volume 2, pages 264–271, June 2003.

[21] F. Fraundorfer, H. Bischof, and S. Ober. Natural, salient image patches for robot localization. In *ICPR'04*, pages 881–884, 2004.

[22] Holger Fröhlich, Jörg K. Wegner, Florian Sieker, and Andreas Zell. Optimal assignment kernels for attributed molecular graphs. In *ICML'05*, pages 225–232, 2005.

[23] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *ICCV'07*, 2007.

[24] P. V. Gehler and S. Nowozin. On feature combination for multiclass object classification. In *ICCV'09*, October 2009.

[25] P.V. Gehler and S. Nowozin. Let the kernel figure it out; principled learning of pre-processing for kernel classifiers. In *CVPR'09*, pages 2836–2843, 2009.

[26] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *PAMI*, 18(4):377–388, April 1996.

[27] Mehmet Gönen and Ethem Alpaydin. Localized multiple kernel learning. In *ICML'08*, pages 352–359, 2008.

[28] K. Grauman and T.J. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *ICCV'05*, pages II: 1458–1465, 2005.

[29] Kristen Grauman and Trevor Darrell. The pyramid match kernel: discriminative classification with sets of image features (version 2). Technical report, MIT, March 2006.

[30] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[31] Chunhui Gu, Joseph J. Lim, Pablo Arbelaez, and Jitendra Malik. Recognition using regions. In *CVPR'09*, pages 1030–1037, 2009.

[32] Patrik O. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–1469, 2004.

[33] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, August 2002.

[34] Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *CVPR'04*, pages II: 506–513, 2004.

[35] A. Kembhavi, B. Siddiquie, R. Miezianko, S. McCloskey, and L.S. Davis. Incremental multiple kernel learning for object recognition. In *ICCV'09*, 2009.

[36] Marius Kloft, Ulf Brefeld, Pavel Laskov, and Sören Sonnenburg. Non-sparse multiple kernel learning. In *NIPS'08 Workshop on Kernel Learning: Automatic Selection of Optimal Kernels*, 2008.

[37] Marius Kloft, Ulf Brefeld, Soeren Sonnenburg, Pavel Laskov, Klaus-Robert Müller, and Alexander Zien. Efficient and accurate ℓ_p-norm multiple kernel learning. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *NIPS'09*, pages 997–1005. 2009.

[38] K. Koeser, C. Beder, and R. Koch. Conjugate rotation: Parameterization and estimation from an affine feature correspondence. In *CVPR'08*, 2008.

[39] C.H. Lampert and M.B. Blaschko. A multiple kernel learning approach to joint multiclass object detection. In *DAGM'08*, 2008.

[40] Gert Lanckriet, Nello Cristianini, Peter Bartlett, and Laurent E. Ghaoui. Learning the kernel matrix with semidefinite programming. *JMLR*, 5:27–72, 2004.

[41] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR'06*, pages 2169–2178, 2006.

[42] F.F. Li, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR'04 Workshop on Generative-Model Based Vision*, 2004.

[43] F.F. Li and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *CVPR'05*, pages II: 524–531, 2005.

[44] H.B. Ling and S. Soatto. Proximity distribution kernels for geometric context in category recognition. In *ICCV'07*, 2007.

[45] David G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[46] S.W. Lyu. Mercer kernels for object recognition with local features. In *CVPR'05*, pages II: 223–229, 2005.

[47] Charles A. Micchelli and Massimiliano Pontil. Learning the kernel function via regularization. *J. Mach. Learn. Res.*, 6:1099–1125, 2005.

[48] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *PAMI*, 27(10):1615–1630, 2005.

[49] K.G. Murty. *Linear Complementarity, Linear and Nonlinear Programming*. Helderman-Verlag, Berlin, 1988.

[50] E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *ECCV'06*, pages IV: 490–503, 2006.

[51] A. Opelt, M. Fussenegger, A. Pinz, and P. Auer. Weak hypotheses and boosting for generic object detection and recognition. In *ECCV'04*, pages Vol II: 71–84, 2004.

[52] A. Opelt, A. Pinz, M. Fussenegger, and P. Auer. Generic object recognition with boosting. *PAMI*, 28(3):416–431, March 2006.

[53] Nicolas Pinto, David D Cox, and James J DiCarlo. Why is real-world visual object recognition hard? *PLoS Comput Biol*, 4(1):27, 01 2008.

[54] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *JMLR*, 9:2491–2521, 2008.

[55] Alain Rakotomamonjy, Francis Bach, Stéphane Canu, and Yves Grandvalet. More efficiency in multiple kernel learning. In *ICML'07*, pages 775–782, 2007.

[56] G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320, March 2001.

[57] J.B. Shi and J. Malik. Normalized cuts and image segmentation. In *PAMI*, volume 22, pages 888–905, August 2000.

[58] Sören Sonnenburg, Bernhard S. Bernhard, P. Bennett, and Emilio Parrado-Hernández. Large scale multiple kernel learning. *JMLR*, 7, 2006.

[59] L. Torresani, V. Kolmogorov, and C. Rother. Feature correspondence via graph matching: Models and global optimization. In *ECCV'08*, pages II: 596–609, 2008.

[60] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. *PAMI*, (in press), 2010.

[61] J.C. van Gemert, J.M. Geusebroek, C.J. Veenman, and A.W.M. Smeulders. Kernel codebooks for scene categorization. In *ECCV'08*, pages III: 696–709, 2008.

[62] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *ICCV'07*, October 2007.

[63] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV'09*, September 2009.

[64] Olga Veksler. Dense features for semi-dense stereo correspondence. *IJCV*, 47(1-3):247–260, 2002.

[65] C. Wallraven, B. Caputo, and A. Graf. Recognition with local features: the kernel recipe. In *ICCV'03*, pages 257–264, 2003.

[66] G. Wang, Y. Zhang, and L. Fei-Fei. Using dependent regions for object categorization in a generative framework. In *CVPR'06*, 2006.

[67] J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *ESANN'99*, 1999.

[68] J.C. Yang, K. Yu, Y.H. Gong, and T.S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR'09*, pages 1794–1801, 2009.

[69] H. Zhang, A.C. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *CVPR'06*, pages II: 2126–2136, 2006.

[70] Z.M. Zhang, Z.N. Li, and M.S. Drew. Feature correspondence with constrained global spatial structures. In *ICIP'09*, pages 177–180, 2009.

[71] Z.M. Zhang, Z.N. Li, and M.S. Drew. AdaMKL: A novel biconvex multiple kernel learning approach. In *ICPR'10*, 2010. Accepted.