Distance learning between image and class for object recognition

Wang, Zhengxiang

2013

Wang, Z. (2013). Distance learning between image and class for object recognition. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/54819>

<https://doi.org/10.32657/10356/54819>

DISTANCE LEARNING BETWEEN IMAGE AND CLASS FOR OBJECT RECOGNITION

A thesis submitted to
The School of Computer Engineering
Nanyang Technological University
Singapore

by

WANG ZHENGXIANG

in partial fulfilment of the requirement
for the degree of Doctor of Philosophy

2013

Abstract

Object recognition is an active research topic in the computer vision community. Recently a novel Image-to-Class (I2C) distance has been proposed to handle this problem, which classifies images using a simple Naive-Bayes based nearest-neighbor (NBNN) classifier but provides surprisingly excellent performance. This new distance provides a novel direction that avoids feature quantization and shows better generalization capability than the traditional Image-to-Image (I2I) distance. However, the computation cost of calculating this distance is too expensive since its performance relies heavily on searching the nearest neighbor (NN) from a large number of training features, and the label information of the training data is not fully used, which limits its recognition performance.

In this thesis, we aim to improve both the recognition performance and efficiency of this I2C distance as well as to extend its application field. **First of all**, we add a training phase to this distance for improving its recognition performance by learning a weighted I2C distance. A large margin optimization framework is proposed to learn the I2C distance function, which is modeled as a weighted combination of the distance from every local feature in an image to its NN in a candidate class. We learn these weights associated with local features in the training set by constraining the optimization such that the I2C distance from image to its belonging class should be less than that to any other class. To reduce the computation cost, we also propose two methods based on spatial division and hubness score to accelerate the NN search, which is able to largely reduce the on-line testing time while still preserving or even achieving a better classification accuracy. **Secondly**, we propose a distance metric learning method to further improve the performance of I2C distance by learning Per-Class Mahalanobis metrics. This Mahalanobis I2C distance is adaptive to different classes by combining with the learned metric for each class. These multiple Per-Class metrics are learned simultaneously by forming a convex

optimization problem and solved by an efficient subgradient descent method. For efficiency and scalability to large-scale problems, we also show how to simplify the method to learn a diagonal matrix for each class. **Thirdly**, we extend the object recognition to the multi-label problem and propose a Class-to-Image (C2I) distance, which shows better performance than the I2C distance for multi-label image classification. However, since the number of local features in a class is huge compared to that in an image, the calculation of the C2I distance is more expensive than the one of I2C distance. Moreover, the label information of training images can be used to help select relevant local features for each class and further improve the recognition performance. Therefore, to make the C2I distance faster and perform better, we propose an optimization algorithm using L_1 -norm regularization and large margin constraint to learn the C2I distance, which can not only reduce the number of local features in the class feature set, but also improve the performance of the C2I distance due to the use of label information. We also use this C2I distance for object localization, so that it can tell not only whether a candidate class appears in a test image, but also where it locates. With these three works, we are able to improve the recognition performance and efficiency of the I2C distance and make it applicable for the multi-label problem. Therefore, the learned distance between image and class would be more practical for real world object recognition applications.

Acknowledgments

First and foremost I would like to take this opportunity to express my sincere appreciation to my supervisor, Associate Professor Chia Liang-Tien, Clement, who has supported me in my four-year research with his patience and knowledge whilst allowing me the room to work in my own way. He gives valuable advice and inspiration on my research topic that makes research a joyful experience. This thesis would not have been possible without his guidance.

I must thank my colleagues Hu Yiqun for his great support on my research. He gives me a lot of suggestion for my research work, including but not limited to idea proposing and paper writing. I would also thank Professor Wu Fei, for his guidance at the beginning of my PhD study.

I would like to thank my colleagues Cheng xiangang, Gao Shenghua, Wang Huan and Ren Zhixiang, who have helped me with my study since the first day I came to NTU. Without their help, work in this thesis would be much harder for me. I want to give thanks to all my colleagues in CeMNet, Chen Lin, Duan Lixin, Duan Qi, Li Gang, Li Wen, Lu Haifeng, Teng Xiao, Wang Danqi, Xu Xinxing, Yang Ming, Yang Shengbo, Yan Shengye, Zhang Da, Zhang Juyong, Zhang Yu, Zhong Feng, etc. for their friendship and help, with which I have impressive years in CeMNet. I am especially grateful to Ms. Chua Poo Hua and Ms. Cindy Teo, who provide me well support of research environment. Last but not least, I must say thanks from bottom of my heart to my family, for their support for my PhD study.

Contents

Abstract	i
Acknowledgments	iii
List of Figures	ix
List of Tables	xiii
1 Introduction	1
1.1 Image-to-Class Distance	1
1.2 Motivation and Research Methodology	2
1.3 Contribution	5
1.4 Organization	6
2 Literature Review	9
2.1 Methods Based on Feature Quantization	9
2.2 Methods Based on the Set-to-Set Distance	13
2.2.1 Methods Based on the Image-to-Image Distance	14
2.2.2 Methods Based on the Image-to-Class Distance	16
2.2.3 Methods Based on the Class-to-Image Distance	19
2.3 Approximate NN for Acceleration	21
2.4 Multi-label Learning	22
2.5 Summary	23
3 Learning Weighted Image-to-Class Distance	25
3.1 Introduction	25
3.2 Optimization Problem	26
3.3 Problem Solver	30

3.4	Efficiency Improving on the NN Search	31
3.4.1	Feature Reduction by Spatial Division	31
3.4.2	Feature Reduction by Hubness Score	33
3.4.3	Discussion of the Two Efficiency Improving Methods	34
3.5	Multi-label Image Classification	35
3.6	Experiments	38
3.6.1	Classification Results of Weighted I2C Distance	38
3.6.2	Results of Accelerating the NN Search	43
3.6.3	Computation analysis	46
3.6.4	Comparing to Previous Results	48
3.6.5	Results on the Multi-label Problem	48
3.7	Summary	52
4	Image-to-Class Distance Metric Learning	53
4.1	Introduction	53
4.2	Related Work	55
4.3	Problem Formulation	56
4.4	An Efficient Subgradient Descent Solver	59
4.5	Learning Diagonal Matrices for Acceleration	62
4.6	Experiments	63
4.6.1	Experiments in Scene-15, Sports and Corel Datasets	63
4.6.2	Experiment in Caltech 101 and 256 Datasets	66
4.6.3	Class Imbalance Experiment	69
4.7	Summary	71
5	Learning Class-to-Image Distance for Multi-label Problems	73
5.1	Introduction	73
5.2	Problem Formulation and Solver	78
5.3	C2I Distance for Object Localization	82
5.4	Experiments	83
5.4.1	Dataset Setup	83
5.4.2	Classification Experiment Result	85

5.4.3	The Influence of Parameter C	88
5.4.4	Resulting Image Examples	89
5.4.5	Object Localization Result	89
5.5	Summary	92
6	Conclusion and Future Work	95
6.1	Conclusion	95
6.2	Future Work	96
	Publication	101
	References	103

List of Figures

1.1	The whole procedure for classifying a test image. Local features of this image are denoted as crosses with different colors. Their NN features in the classes are denoted with the same color (see the green arrow as example). Other unmatched features in the classes are in black color. The sum of Euclidean distance between features and their NN formulate the I2C distances to different classes, which are denoted as different length of blue bars for expressing the relative size. The class label with the shortest I2C distance is selected as the predicted class.	3
1.2	The structure of the thesis.	7
2.1	The procedure of BoW model for quantizing a set of feature vector to a single vector. Adapted from [1].	10
2.2	A toy example of the division of SPM from level 0 to level 2. Adapted from [2]. . . .	11
2.3	The large margin hyperplane for a SVM with samples from two classes.	12
2.4	The Set-to-Set distance and its sub-categories.	14
2.5	The two triplet formulations in the constraint of large margin optimization problem for [3] and [4] respectively. Weights are associated with local features in (a) reference images and (b) positive or negative images. Note the different I2I distance directions between reference image and positive or negative image in these two triplets.	15
2.6	The significance coefficients of two instances with respect to class “person” (left) and “horse” (right), adapted from [5].	21
3.1	The original unweighted I2C distance in NBNN (left) and our proposed weighted I2C distance (right).	27
3.2	The triplet constraint for distance learning. For an image X_i , its I2C distance to the positive class C_p should be less than that to the negative class C_n with a margin. . .	29

3.3	The I2C NN feature search with spatial division. We adopt the idea of spatial pyramid by restricting the NN of each feature in the same subregion at each spatial level. . . .	32
3.4	Per-Class classification accuracy in Scene-15, Sports and Corel datasets for comparing LWI2C with NBNN.	40
3.5	Distributions of local features with high weight (red) and low weight (blue) for classes of “owls”, “mountains” and “flowers” in Corel dataset.	41
3.6	Predicted examples in Corel dataset. Images in the top row are correctly classified by both LWI2C and NBNN. The middle row shows images that are correctly classified by LWI2C but wrongly classified by NBNN, and images in the bottom row are wrongly classified by either LWI2C or NBNN.	42
3.7	Comparing the classification accuracy of LWI2C to NBNN for Scene-15, Sports and Corel datasets with various percentages of preserved training features in the training set.	45
3.8	Comparing the classification accuracy of LWI2C to NBNN for Caltech 101 in spatial level 2 with various percentages of preserved training features in the training set. . .	46
3.9	F_1 score for measuring the performance of image annotation.	51
4.1	The prediction of image X_i from class C_1 to C_{N_C} , Where N_C is the total number of classes. The test image X_i is denoted as rectangle and each ellipse denotes a class with training images (rectangles) inside it.	54
4.2	The label ranks generated by both our method and NBNN for a test image of ak47 as example. Only the top 11 class labels are shown.	68
4.3	The recognition accuracy of our method compared to NBNN in 8 groups with different imbalanced levels.	70
5.1	The illustration of I2C distance (a) and C2I distance (b). Red and black rectangles represent the local features located at the region of “car” and “street” respectively. The NN search directions are represented by arrows from one side to the other side. Red and black arrows represent the NN match between same region and different regions respectively. Though region “street” may not be useful in the “car” class, the NN match from region “street” to “car” in the I2C distance (a) cannot be avoided, while in the C2I distance (b) it can be avoided.	74

5.2	The example of the C2I distance from training feature set of class “car” to some test images containing a car object.	77
5.3	The representative example of local features in the training feature set of classes selected by our learning method.	90
5.4	The representative example of local features in test images matched as NN feature by our learned C2I distance.	90
5.5	The good, median and bad localization examples according to the overlap area. Blue bounding box is predicted by our method and red bounding box is the ground truth. .	93

List of Tables

3.1	Classification accuracy (%) of LWI2C compared to NBNN for each dataset with its attributes.	39
3.2	Classification accuracy (%) of LWI2C compared to NBNN on different spatial levels as well as SPM for Scene-15, Sports, Corel, Caltech 101 and Caltech 256 datasets. L0 denotes level 0 of 1×1 subregions, L1 denotes level 1 of 2×2 subregions and L2 denotes level 2 of 4×4 subregions. SPM combines three spatial levels and achieves the best accuracy, while the accuracy of spatial levels at L1 and L2 is better than L0.	44
3.3	The average running time for weight learning and NN search of each image. (s=second, h=hour)	47
3.4	The average running time for the NN search of each image on different spatial division level and various percentages of preserved training features in the training set. (second)	47
3.5	Comparing to recently published results for classification accuracy (%).	49
3.6	Performance evaluation in natural scene dataset, ↓ means the smaller the better and ↑ means the larger the better.	50
3.7	Performance evaluation in MSRC dataset, ↓ means the smaller the better and ↑ means the larger the better.	51
4.1	Comparing our learned Mahalanobis I2C distance to Euclidean I2C distance, Mahalanobis I2I distance and Euclidean I2I distance in Scene-15, Sports and Corel datasets by recognition accuracy (%). Here, M and E stand for Mahalanobis and Euclidean distance, respectively.	64
4.2	Comparing the computation time and the number of triplets of our learning method to LMNN in Scene-15, Sports and Corel datasets (seconds).	65
4.3	Comparing learning diagonal matrices to full matrices in Scene-15, Sports and Corel datasets by recognition accuracy (%), memory space and computation time.	65

4.4	Comparison of our learned Mahalanobis I2C distance to Euclidean I2C distance, Mahalanobis I2I distance and Euclidean I2I distance in Caltech 101 by recognition accuracy (%). Here M and E stand for Mahalanobis and Euclidean distance respectively. . . .	66
4.5	Comparing learning diagonal matrices to full matrices in Caltech 101 by recognition accuracy (%), memory space and computation time.	67
4.6	Results in Caltech 256 by recognition accuracy (%).	67
4.7	The maximum, mean and variance of the number of training images per class in each group for class imbalance experiment in Caltech 101 dataset	70
5.1	Performance comparison using MAP	85
5.2	Sparsity comparison (Percentage of the number of local features compared to the original feature set)	87
5.3	Running time comparison (second)	87
5.4	Memory usage comparison (MB)	87
5.5	MAP and sparsity of our method over different C value on MSRC dataset	88
5.6	Localization accuracy of our method compared with other methods on VOC2007 dataset	91

Chapter 1

Introduction

1.1 Image-to-Class Distance

Successful object recognition would be the start of the quest to classify, retrieve or annotate images automatically by computers. This task is very useful in many applications, such as face detection [6], optical character recognition [7], pedestrian detection [8], fingerprint recognition [9], gesture recognition [10], etc. The technology of object recognition is also used in recent commercial products like Kinect ¹ and Google Image Search ². However, this task is still quite challenging in the computer vision community due to the large intra-class variances, ambiguities, occlusions, background clutters, viewpoints, scales and illumination conditions between different images.

In most studies, the first step of object recognition is to extract salient features from the original image for representation. These features are mainly divided into three categories according to [11]: global feature, region-based feature and patch-based feature (or local descriptors). Among them the region-based feature seems the most reasonable to represent images. However, this feature needs segmented regions for sampling, which is also a challenging problem since current best automatic image segmentation algorithms are far from satisfaction while manually segmentation is expensive. So in the literature most previous studies use patch-based local feature. In this case, each image is represented as a set of local features, which are extracted from patches around densely sampled keypoints and described by some popular local descriptors like SIFT [12] and

¹<http://en.wikipedia.org/wiki/Kinect>

²<http://images.google.com>

SURF [13]. Due to their robustness to occlusion, background clutter and other content changes, these patch-based features have been widely used in many previous studies.

These features are then used as input to classifiers for object recognition. Previous studies for designing such classifiers can roughly be divided into learning-based and non-parametric methods according to [14]. Learning-based methods (also known as parametric methods) include SVM [2, 15], Boosting [16], decision trees [17], etc, and they require a training phase for classification. Non-parametric methods directly classify test data without any training phase, and most of them rely on Nearest-Neighbor (NN) distance estimation. In most cases, learning-based methods can achieve better recognition performance than non-parametric methods as they have learned the model from the training set, which is useful for classifying test images. Nevertheless, recently Boiman *et al.* [14] proposed a novel non-parametric method named as NBNN (stand for Naive-Bayes based nearest-neighbor), which achieved comparable performance to those top learning-based methods even without any training phase. This method introduces an Image-to-Class (I2C) distance for classifying test images. They owe the success of NBNN to the use of I2C distance instead of the Image-to-Image (I2I) distance and the avoidance of feature quantization, and they have shown that the I2I distance leads to significant degradation for classification.

Specifically, the I2C distance is formulated as the sum of Euclidean (or L_2) distance from each patch-based feature in a test image to its NN in a set of features in a candidate class, where the feature set of this class is constructed by gathering local features from all training images belonging to this class, and the test image is also represented as a set of local features. These distances to all classes can generate a list of label rank and the top label is selected as the predicted label, which means the class with the shortest I2C distance is selected as the predicted class. Figure 1.1 gives an overview of such formulation and prediction.

1.2 Motivation and Research Methodology

One advantage of the I2C distance is that it avoids feature quantization, which is a standard step in most alternative schemes. Under patch-based feature representation,

Figure 1.1: The whole procedure for classifying a test image. Local features of this image are denoted as crosses with different colors. Their NN features in the classes are denoted with the same color (see the green arrow as example). Other unmatched features in the classes are in black color. The sum of Euclidean distance between features and their NN formulate the I2C distances to different classes, which are denoted as different length of blue bars for expressing the relative size. The class label with the shortest I2C distance is selected as the predicted class.

each image is represented by a set of features. So measuring the distance between two images usually requires feature quantization methods like Bag-of-Words (BoW) to convert these features into one vector, which may lose the internal discriminative information of the image and introduce the discretization errors. The I2C distance also has better generalization capability than the traditional I2I distance. When measuring the I2I distance from a test image to the training image, even if both images belong to the same class, the training image may contain other incompatible objects that makes them less similar and increases their I2I distance. However, the I2C distance may avoid such problem since each feature in the test image finds its NN (similar patches) through all training images in the class, and is less likely to compare with these incompatible objects but more likely to find the NN match from the compatible objects. In addition, the I2C distances from a test image to all the candidate classes directly reflect the probability of classifying this test image to each class, so the prediction using the I2C distance is straightforward. Therefore, this novel I2C distance is possibly providing a new way for measuring the distance in the problem of object recognition.

However, the performance of this distance relies heavily on the large number of local features in the training set. For example, the state-of-the-art performance [14] reported in Caltech 101 dataset is achieved by using densely sampled multi-scale local features, resulting in about 15000 to 20000 features per image. Such large feature set makes the NN search in the I2C distance calculation computation expensive during the testing phase, which limits its usage for real world applications. On the other hand, using a small number of local features results in poor performance, although it requires less computation cost for the NN search. Besides, since the current I2C distance does not have any training phase, the training data only served for NN match and its label information is not fully explored. When classifying images with multiple labels, this I2C distance also faces the problem that irrelevant local features in a test image cannot be removed when counting for the final I2C distance to a relevant class, which would influence the accuracy of the I2C distance.

Such problems motivate our effort to improve the I2C distance. First, we would like to learn this I2C distance by adding a training phase with some learning methods, so that the learned I2C distance can be more discriminative and provide better classification

performance. With this additional training phase, more running time in the on-line testing phase can be saved since the learned I2C distance can work well under a small number of local features. We also try to accelerate the NN search in the I2C distance calculation to further reduce the computation cost in the testing phase. Then we extend the object recognition to the multi-label problem and propose the Class-to-Image (C2I) distance, which shows better performance than the I2C distance for multi-label image classification.

1.3 Contribution

In this thesis, we aim to improve both recognition performance and efficiency of this I2C distance as well as to extend its application field. We summarize these contributions as follows:

- We add a training phase to learn a weighted I2C distance for improving its recognition performance. For each local feature in the training set, we associate it with a weight. Therefore the resulting I2C distance is modeled as a weighted combination of the distance from every local feature in an image to its NN in a candidate class. A large margin optimization framework is proposed to learn this I2C distance function with the constraint that the I2C distance from image to its belonging class should be less than that to any other class. Since the main computational bottleneck in the I2C distance calculation is the NN feature search due to the large number of local features in the training set, to improve the efficiency, we also propose two methods based on spatial division and hubness score to accelerate the NN search, which is able to largely reduce the on-line testing time while still preserving or even achieving a better classification accuracy.
- We also propose a distance metric learning method to further improve the performance of the I2C distance by learning Per-Class Mahalanobis metrics. For each class, we learn the class specific Mahalanobis metric to combine with the corresponding I2C distance. These multiple Per-Class metrics are learned simultaneously by forming a convex optimization problem. This optimization problem is

an instance of semi-definite programming (SDP) and can be solved using standard SDP solvers. However, the standard SDP solvers are usually computation expensive and less efficient. Since our optimization problem is convex, we use an efficient subgradient descent based method derived from [18] to solve this problem. For efficiency and scalability to large-scale problems, we simplify the method to learn only a diagonal matrix for each class instead of the full matrix, which can significantly reduce the computational time and memory usage while preserving the classification performance. Another contribution of this distance metric learning method is that it can correct the class imbalance problem, in which some classes contain much more training images than other classes.

- We extend the object recognition to the multi-label problem and propose the C2I distance to measure the distance between image and class for this problem. This distance is able to outperform the I2C distance when images belong to multiple labels. However, since the number of local features in a class is huge compared to that in an image, the calculation of the C2I distance is more expensive than the I2C distance. Moreover, the label information of training images can be used to help select relevant local features for each class and further improve the recognition performance. Therefore, to improve the performance and efficiency of the C2I distance, we propose an optimization algorithm using L_1 -norm regularization and large margin constraint to learn the C2I distance. With this optimization algorithm, we are able to get better recognition performance even though the size of feature set is much smaller. A gradient descent based solver is also proposed to efficiently solve this optimization problem. We also use this C2I distance for object localization, so that for a specific class, this distance can not only select test images belonging to this class, but also predict the bounding box of these test images.

1.4 Organization

The structure of this thesis is given in Figure 1.2. We organize this thesis as follows:

Chapter 2 Literature Review In this chapter, we will briefly review previous studies for object recognition, including feature quantization methods like BoW and I2C

Figure 1.2: The structure of the thesis.

related methods that directly measure distance without feature quantization. We will also discuss some approximate NN methods for accelerating NN search, since they serve as an important part in measuring the distance between image and class.

Chapter 3 Learning Weighted Image-to-Class Distance In this chapter we will describe the algorithm of learning weighted I2C distance to improve the recognition performance and two methods of accelerating the NN search to improve the efficiency of the I2C distance. With the learned weight, we will show that the learned I2C distance performs significantly better than the original I2C distance. The two methods of accelerating NN search will also be shown to largely reduce the computation cost while still preserving or even achieving a better classification accuracy.

Chapter 4 Image-to-Class Distance Metric Learning In this chapter we will present another I2C distance learning method that learns a Mahalanobis metric for each class. The learned Mahalanobis I2C distance will be more discriminative especially when there are a large number of classes. For efficiency, we also propose to learn a diagonal matrix for each class to reduce the computation cost and memory usage, which can still preserve the classification performance. We will also show that the distance metric learning method is able to correct the class imbalance problem.

Chapter 5 Learning Class-to-Image Distance for Multi-label Problems In this chapter, we extend the object recognition to the multi-label problem and propose a C2I distance learning method for multi-label image classification. We will also apply the learned C2I distance for the task of object localization and show that it can achieve excellent localization performance.

Chapter 6 Conclusions and Future Work In the final chapter, we will conclude this thesis and discuss some future works on this topic, which deserve future studies and investigations.

Chapter 2

Literature Review

In Chapter 1, we have mentioned that most of recent studies extract images using patch-based local features. Therefore, each image is represented by a set (or bag) of local features, which are extracted from patches around densely sampled keypoints and described by some popular local descriptors like SIFT [12] and SURF [13]. After that, there are two categories of measuring the distance between images: the first one uses some descriptor quantization methods like BoW to convert the set of local features to one single vector for representing an image, therefore kernel functions like linear kernel, RBF kernel, histogram intersection kernel [2, 19], etc. can be used to measure the distance or similarity between two images and kernel methods like support vector machine (SVM) can be used to classify these images. We will discuss such relevant methods in Section 2.1. The second category directly measures the distance between two sets using NN search without feature quantization, which can be further divided into three sub-categories: Image-to-Image distance, Image-to-Class distance and Class-to-Image distance. All of them are asymmetric and can be recognized as Set-to-Set (S2S) distance. We will discuss methods using such distances in Section 2.2. We will also discuss some approximate NN methods for accelerating NN search in Section 2.3, which serve as an important part in methods employing S2S distance. Previous study on multi-label learning will be discussed in Section 2.4, since our learning methods will be used under the multi-label problem.

2.1 Methods Based on Feature Quantization

BoW model is a popular feature quantization method due to its robustness to variance in scale, illumination, rotation and translation. The quantization in BoW is achieved

Figure 2.1: The procedure of BoW model for quantizing a set of feature vector to a single vector. Adapted from [1].

through a visual word vocabulary, which is usually generated using unsupervised clustering method like K-means. Therefore each cluster in feature space is recognized as a visual word in the vocabulary. This vocabulary is used to map local features to visual word by selecting the nearest cluster center, so that the original image can be represented as a bag of visual words. Finally the quantization to a single visual-word vector is achieved by counting the frequency of each visual word. Figure 2.1 gives an overview of the whole process.

The effectiveness of this quantization method attracted many recent work to focus on it and try to improve its performance. In [20, 21], they replaced the hard-assignment by soft-assignment, which adds some uncertainty to the visual word assigning so that each local feature is mapped to a weighted combination of visual words, rather than to a single visual word in hard-assignment. This soft-assignment can better preserve the loss in quantization and improve the performance of BoW, at the cost of more storage requirements for the indexing.

Another drawback of BoW is that the spatial information is lost during the quantization. To preserve this loss, Lazebnik *et al.* [2] proposed a spatial pyramid match (SPM)

Figure 2.2: A toy example of the division of SPM from level 0 to level 2. Adapted from [2].

for BoW, which is motivated by Grauman and Darrell’s pyramid matching [22] in the feature space. In SPM, each image is divided into increasingly finer subregions. The original image can be viewed as 1×1 region at level 0. Then this image is divided into 4 subregions of equal area, i.e., 2×2 at level 1. Such division is continued from upper level to lower level as shown in Figure 2.2. Local features located in each subregion at each level are quantized to a visual-word vector and these vectors in different subregions of each level are concatenated to form a long vector. So measuring two images is achieved by the weighted sum of matching the vectors at each level. This method provides an effective complement to the BoW model and improves its performance significantly for object recognition. The idea of such spatial increasingly division is widely used in recent works [15, 19, 23] and also used in this thesis.

Since most local features are formulated as histograms for patches around keypoints, e.g. SIFT, SURF and HOG, and it has been shown that the Euclidean distance is not as effective as histogram intersection for comparing two histograms, Wu and Rehg [19] proposed to replace the Euclidean distance by histogram intersection for measuring the distance between visual word and local feature, which is also an important improvement for BoW model.

Recently, a novel technique named sparse coding has been applied to object recognition problem [15]. This idea is originally used for signal processing to encode the signal

Figure 2.3: The large margin hyperplane for a SVM with samples from two classes.

in a sparse signal space. Similar to BoW model, sparse coding requires a vocabulary as basis set to reconstruct local features as sparse codes. However, different from the vocabulary generation method in BoW, unsupervised clustering method like K-means is replaced by sparse coding, which updates the vocabulary and coding vectors of training data iteratively and enforces the coding vector to be sparse. In this way, both the quality of vocabulary and the accuracy of coding are improved. Then max pooling strategy is applied to convert these sparse codes to one single vector for input to classifiers such as SVM. Compared to the quantization method in BoW, sparse coding is able to achieve much lower reconstruction error since it has less restrictive constraint. Recent studies [15, 24–27] based on sparse coding technique have shown excellent performance for image classification problem. This technique has also been widely used in other computer vision problem, such as face recognition [28–36], saliency detection [37], image annotation [11, 38], image super-resolution [39–42], motion and data segmentation [43, 44], etc.

All the above methods finally result in a single vector (a data point) for representing each image. Then comparing the distance or similarity of two images is easily achieved by measuring these two vectors using distance or similarity measuring functions like Euclidean distance, Chi-square distance, polynomial kernel, linear kernel, RBF kernel, histogram intersection, etc. For the task of image classification, classifiers like SVM can be used with the input of kernel matrix, which contains the pairwise similarity between training images and test images. This classifier simultaneously minimizes the empirical classification error and maximizes the geometric margin, with a large margin hyperplane

to create the largest separation between two classes. So the distance from this hyperplane to the nearest sample on each side is maximized as shown in Figure 2.3, and these samples on the margin are called support vectors. Such large margin idea leads to the success of SVM classifier for binary classification problem and is widely used by many subsequent studies [4, 5, 18, 45–48]. For multi-class classification problem, One-VS-One or One-VS-All SVM [2, 15, 23] is used to decompose the problem to many binary sub-problems, and One-VS-All SVM is more popular when classifying images from a large number of classes [49].

2.2 Methods Based on the Set-to-Set Distance

Under patch-based feature description, each image can be represented as a set of local features, where the dimensions of all local features are kept the same (e.g. when using SIFT feature, the dimension of every local feature is 128). However, the size of feature set for each image is decided by the image size and may not be the same among different images. In the I2C distance, a class can also be represented as a set of such local features by gathering local features from all images belonging to this class. Therefore, the feature set of either an image or a class can be denoted as a set $F_X = \{f_{X1}, f_{X2}, \dots, f_{Xm_X}\}$, where X represents an image or a class, m_X represents the total number of local features in X and each local feature in the set is denoted as $f_{Xi} \in R^d, \forall i \in \{1, \dots, m_X\}$, and d is the dimension of the local feature. The distance from one set F_X to another set F_Y is first proposed by Frome *et al.* [3, 4] to measure the I2I distance, which is then extended to the I2C distance by Boiman *et al.* [14] and can be generalized to all Set-to-Set distances. Specifically, the distance from set F_X to set F_Y is defined as the sum of distances from every local feature f_{Xi} in F_X to the set F_Y . Such distance from f_{Xi} in F_X to F_Y is named as Feature-to-Set distance and is defined as the Euclidean distance between f_{Xi} in F_X and its NN f_{Xi}^Y in F_Y , where the NN feature f_{Xi}^Y is searched over all local features in set F_Y with the shortest distance to f_{Xi} . Therefore, the distance from set F_X to set F_Y is formulated as:

$$Dist(X, Y) = \sum_{i=1}^{m_X} \|f_{Xi} - f_{Xi}^Y\|^2, \quad (2.1)$$

Figure 2.4: The Set-to-Set distance and its sub-categories.

where

$$f_{Xi}^Y = \arg \min_{f_{Yj} \in \{f_{Y1}, \dots, f_{Ym_Y}\}} \|f_{Xi} - f_{Yj}\|^2. \quad (2.2)$$

With this definition, the distance from F_X to F_Y is different from F_Y to F_X , therefore such distance is asymmetric. Since the feature set can either be an image or a class, recent studies in computer vision based on this type of distance can be divided into three sub-categories: Image-to-Image distance, Image-to-Class distance and Class-to-Image distance. We will briefly review recent works based on each of these three distances. Figure 2.4 shows an example of Set-to-Set distance and the three sub-categories.

2.2.1 Methods Based on the Image-to-Image Distance

The I2I distance function is first introduced in Frome *et al.* [3], which is formulated as a weighted combination of Feature-to-Set distances from every local feature in one image to the feature set of another image. The weight associated with each local feature aims to distinguish the relative importance of different local features in an image. A large margin framework is proposed to learn these weights. Similar to SVM, the error term

Figure 2.5: The two triplet formulations in the constraint of large margin optimization problem for [3] and [4] respectively. Weights are associated with local features in (a) reference images and (b) positive or negative images. Note the different I2I distance directions between reference image and positive or negative image in these two triplets.

in the optimization problem is constructed by a set of constraints with a large margin. Each constraint is formulated by a triplet consisting of a reference image, a positive image and a negative image, where a positive or negative image is decided by whether images are labeled with the same class to the reference image. The I2I distance from reference image to the positive image is constrained to be smaller than the I2I distance to the negative image with a large margin in each triplet. By changing the positive or negative image, multiple different triplets can be generated. Therefore, weights associated with local features in the reference image can be learned with these triplets in the large margin optimization problem as shown in Figure 2.5(a). However, since each triplet only involves the weights in the reference image, the whole optimization problem can be broken up into a separate sub-problem for each reference image and weights in different images are learned independently. Such formulation would result in an inconsistency between training and testing phases as the distances from different training images to a new test image cannot be directly comparable.

To solve the inconsistency problem, a new I2I distance learning method is proposed in [4], which learns all weights in the training set simultaneously and is able to keep global consistency during the training phase. Like [3], the weighted distance function is learned

via a large margin optimization problem, where each triplet constraint is formulated by a reference image, a positive image and a negative image. However, in this work the distance direction is reversed in each triplet as it constraints that the I2I distance from the positive image to the reference image should be smaller than from negative image to the same reference image with a large margin. With this reversion, weights associated with the positive and negative image are learned in each triplet as shown in Figure 2.5(b). Therefore each triplet can involve the weights in two images. By changing the positive or negative image in different triplets, weights in every training image are able to communicate with the weights in each other image. In this way all weights are able to be updated simultaneously during the learning procedure and make the whole learning procedure globally consistent, which ensures the I2I distances from different training images to a test image is directly comparable for the task of image retrieval or classification.

However, there is a problem in the triplet formulations in both methods: too many triplets are generated to update the weight. For a dataset with N_C classes and N_T training images per class, this formulation results in $O(N_C^2 \cdot N_T^3)$ triplets, which has much redundancy. Therefore it is necessary to reduce the number of triplets in the learning procedure whilst keep the learned weights effective.

2.2.2 Methods Based on the Image-to-Class Distance

The I2C distance is first proposed in the work of NBNN [14], which uses this I2C distance to approximate the optimal maximum-a-posteriori Naive-Bayes classifier and therefore avoids feature quantization in traditional methods. Specifically, the class label of an image X under Naive-Bayes assumption is given as:

$$C^* = \arg \max_C p(C|X). \quad (2.3)$$

If the class prior $p(C)$ is uniform, it can be reduced to the maximum likelihood classifier:

$$C^* = \arg \max_C p(X|C). \quad (2.4)$$

Since each image X is represented as a set of local features $F_X = \{f_{X1}, \dots, f_{Xi}, \dots, f_{Xm_X}\}$ and each local feature f_{Xi} could be assumed as independent of each other in the image,

$p(X|C)$ can be derived as:

$$p(X|C) = p(f_{X1}, \dots, f_{Xi}, \dots, f_{Xm_X}|C) = \prod_{i=1}^{m_X} p(f_{Xi}|C). \quad (2.5)$$

Integrate $p(X|C)$ and take the log probability in Equation (2.4) we can get:

$$C^* = \arg \max_C \log\left(\prod_{i=1}^{m_X} p(f_{Xi}|C)\right) = \arg \max_C \frac{1}{m_X} \sum_{i=1}^{m_X} \log(p(f_{Xi}|C)), \quad (2.6)$$

where $p(f_{Xi}|C)$ is the feature probability conditioned on class C and can be approximated by Parzen likelihood estimation. When using Gaussian kernel function, $p(f_{Xi}|C)$ can be expressed as:

$$p(f_{Xi}|C) = \sum_{j=1}^{m_C} \exp\left(-\frac{\|f_{Xi} - f_{Cj}\|^2}{2\sigma^2}\right), \quad (2.7)$$

where σ is a smoothing parameter, also called bandwidth.

However, this calculation is computational expensive since every local feature f_{Cj} in class C is involved in the calculation of $p(f_{Xi}|C)$. NBNN therefore only takes into account the nearest local feature of f_{Xi} in class C , denoted as f_{Xi}^C , for approximating $p(f_{Xi}|C)$:

$$p(f_{Xi}|C) = \exp\left(-\frac{\|f_{Xi} - f_{Xi}^C\|^2}{2\sigma^2}\right). \quad (2.8)$$

Then Equation (2.6) is simplified to:

$$\begin{aligned} C^* &= \arg \max_C \frac{1}{m_X} \sum_{i=1}^{m_X} \log\left(\exp\left(-\frac{\|f_{Xi} - f_{Xi}^C\|^2}{2\sigma^2}\right)\right) \\ &= \arg \min_C \sum_{i=1}^{m_X} \|f_{Xi} - f_{Xi}^C\|^2 = \arg \min_C \text{Dist}(X, C). \end{aligned} \quad (2.9)$$

Therefore, the I2C distance $\text{Dist}(X, C)$ is extracted from $\sum_{i=1}^{m_X} \|f_{Xi} - f_{Xi}^C\|^2$ and can be interpreted as the sum of Euclidean distance between every local feature f_{Xi} and its NN feature f_{Xi}^C in class C .

However, the performance of NBNN relies heavily on the large number of local features in the training feature set due to the limitation of the simplified approximation in Equation (2.8). To enhance the generalization ability, Behmo *et al.* [50] proposed an Optimal NBNN method, which is a parametric version of NBNN. In this work, the log-likelihood of local feature is approximated by an affine function of the nearest neighbor

distance, which involves two affine coefficients dependent on the properties of the training feature set. Specifically, the I2C distance from image X to class C is changed to:

$$Dist(X, C) = \alpha^c \cdot \sum_{i=1}^{m_X} \|f_{X_i} - f_{X_i}^C\|^2 + m_X \cdot \beta^c, \quad (2.10)$$

where α^C and β^C are the coefficients of the class C , which are solved by a linear problem with the hinge-loss minimization. The learned coefficients are also able to correct the class imbalance problem, while NBNN fails to perform well when some classes are more densely sampled than others and therefore contains more local features. Another contribution of this work is to reformulate the Optimal NBNN classifier to perform object detection and classifying images by the detection results, which is achieved by finding the image sub-window that maximizes a function that is linear in the image features. Due to this linearity, the optimal object location can be found by branch and bound sub-window search.

Another work based on the I2C distance is proposed by Tuytelaars *et al.* [51], where a kernelized version of NBNN is introduced, so that the I2C distance can be combined with other BoW based kernels and the combined kernel classified by SVM is able to achieve better classification performance. Specifically, the feature vector of each image X is constructed by concatenating the I2C distance to every class in the dataset:

$$\Phi(X) = [Dist(X, 1), \dots, Dist(X, C)]^T. \quad (2.11)$$

Then the kernel between image X and Y is calculated as:

$$K(X, Y) = \Phi(X)^T \Phi(Y). \quad (2.12)$$

The kernelized version of the I2C distance can be seen as complimentary to standard BoW based kernels and therefore generating a more discriminative classifier.

Recently, McCann and Lowe [52] proposed an alternate way to calculate the I2C distance without any additional training phase, which is named as Local NBNN and is able to accelerate the distance calculation. Instead of calculating the NN feature in every class for each local feature f_{X_i} in the image X as in the original NBNN, this method gathers the feature set of all classes together and only finds the K-NN for each local

feature f_{X_i} . If at least one local feature in these K-NN belongs to class C, the Euclidean distance between f_{X_i} and its nearest local feature in these K-NN belonging to class C is defined as the Feature-to-Set distance from feature f_{X_i} to class C. For classes without any local feature belonging to in these K-NN, the Feature-to-Set distances to these classes are defined as the Euclidean distance between f_{X_i} and its (K+1)-NN. The final I2C distance is a linear combination of these Feature-to-Set distances same as in the original NBNN. The main speed up in calculation is achieved by Approximate Nearest Neighbor (ANN) methods like KD-Tree [53]. Let N_T be the average number of training images per class, N_C the number of classes, and N_D the average number of local features per image. In the original NBNN, the feature set for each class requires to build a KD-tree, which contains $N_T \cdot N_D$ local features. Each of the N_D local feature in a test image requires an approximate search in each of the N_C KD-tree structures. The accuracy of the approximate search is controlled by the number of distance checks, denoted as c . Therefore the time complexity for processing one test image in the original NBNN is $O(c \cdot N_C \cdot N_D \cdot \log(N_T \cdot N_D))$. However, in Local NBNN, each local feature searches K-NN from the whole training feature set containing $N_C \cdot N_T \cdot N_D$ local features, as the distance checks are usually much larger than K, the time complexity is $O(c \cdot N_D \cdot \log(N_C \cdot N_T \cdot N_D))$. The term N_C is moved inside the log term and therefore Local NBNN needs much less computation cost for distance calculation compared to the original NBNN. More importantly, by adjusting an appropriate K, the performance of Local NBNN is able to outperform the original NBNN. The only problem in this method is the additional parameter K required to be adjusted before being adapted to a new dataset.

Besides these methods, there are other works based on the I2C distance. Huang *et al.* [54] applied this I2C distance in face and human gait recognition. Cai *et al.* [55] proposed a local learning method for the I2C distance to improve its performance. Zhang *et al.* [56] proposed a random sampling I2C distance to tackle the imbalanced problem, which enables the I2C distance comparable when the dataset is imbalanced.

2.2.3 Methods Based on the Class-to-Image Distance

The C2I distance is an inversed version of the I2C distance. The NN search direction is from local features in a class to the feature set of an image. For a class C with the

feature set $F_C = \{f_{C1}, f_{C2}, \dots, f_{Cm_C}\}$, the formulation of C2I distance to an image X is given as:

$$Dist(C, X) = \sum_{i=1}^{m_C} \|f_{Ci} - f_{Ci}^X\|^2, \quad (2.13)$$

where

$$f_{Ci}^X = \arg \min_{f_{Xj} \in \{f_{X1}, \dots, f_{Xm_X}\}} \|f_{Ci} - f_{Xj}\|^2 \quad (2.14)$$

is the NN feature searched in the feature set F_X of image X that has the shortest Euclidean distance to the local feature f_{Ci} in class C . The C2I distance is first proposed by Wang *et al.* [5, 57–59] for multi-instance learning. In this work, the feature set of a class or an image is a collection of instances rather than local features and each instance is assigned with a significance coefficient to reflect the relative importance of the instance with respect to its labeled classes, thereby providing a clear insight of a multi-instance data set. These significance coefficients are learned in a L_{0+} -norm problem in [57] and large margin approach in [5]. The examples of the learned significance coefficients for the same instances in two different classes are shown in Figure 2.6. The instance located at the horse learns a large significance coefficient in the class “horse” while a low value in the class “person”, meaning this instance contributes a lot in the C2I distance for class “horse” to a test image, while having less impact in the C2I distance from class “person” to a test image. Another recent work [59] learns a distance metric for C2I distance by solving $L_{2,1}$ -norm minimization and maximization (minmax) problem and used it for video classification problem.

In this thesis, we also use the C2I distance for the multi-label problem served as an alternative to the I2C distance. Different from the multi-instance learning C2I distance in the above works, the feature set of a class or an image is composed of local features. As the number of local features in a class is usually much more than that in an image, the computation cost of the C2I distance is more expensive than the I2C distance. We address this problem and use L_1 -norm regularization on the weight vector to generate a sparse solution, which is able to speed up the C2I distance calculation and outperform the I2C distance for the multi-label problem.

Figure 2.6: The significance coefficients of two instances with respect to class “person” (left) and “horse” (right), adapted from [5].

2.3 Approximate NN for Acceleration

In every sub-category of the Set-to-Set distance, NN search is required to find the shortest Euclidean distance for calculating the Feature-to-Set distance. If using brute force accurate NN search, the time complexity would be linear to the number of local features in the target set, which is unacceptable as the computation cost of distance calculation would be expensive when the size of set is very large. Therefore, methods for searching approximate nearest neighbor (ANN) is necessary to accelerate NN search and distance calculation when applied to any large scale dataset.

One of the popular ANN method is k-dimensional tree (KD-Tree) [53], which is able to reduce the NN search to logarithmic search time. A description of the construction of KD-Tree and how it works for NN search can be seen in [60]. Here we briefly summarize as follows: KD-Tree is constructed by recursively partitioning the feature space of data points into two subspaces. Each subspace is associated with an internal node, at which a partition hyperplane is generated to split the feature space. Such partition is recursively conducted from the root node to the leaf, where the leaf node may contain one or more data points. To find the NN of a query data point, the algorithm performs a top-down search. At each internal node, the algorithm will decide which side of the partition hyperplane to search down. Data points in the first leaf node are the initial candidates as NN, but not necessarily the accurate NN. Backtracking or iterative search is required to

check whether other leaf nodes have better candidates. The search procedure terminates when there are no more cells within the distance between the current best point and the query point. The effectiveness of KD-Tree for the NN search makes it widely used in many computer vision applications [60–64]. However, when the feature space is high dimensional, the time complexity would be increased to nearly linear to the size of the dataset and its efficiency is significantly reduced. This is because the time to backtrack through the tree to find the best NN point will be heavily increased with the increase in the dimensions.

Locality-Sensitive Hashing (LSH) [65] is another ANN method that performs more efficiently with high dimensional data. The basic idea is to hash the input data points so that similar points are mapped to the same buckets with high probability. Its performance of searching ANN is dependent on the choice of hash functions, the number of hash tables and the number of bits in the hash key, and the accuracy of the NN search is also a trade-off of the query time. Recent work using LSH for image search includes [66–71].

In this thesis, we decide to use KD-Tree for accelerating NN search in the experiment since it is used in the original I2C distance [14] and is reported to achieve excellent performance. Note that we only use it for C2I distance calculation since the original C2I distance is more computational expensive. However, in the experiment of the I2C distance, due to the less number of local features sampled in each image and the use of two efficiency improving methods proposed in Chapter 3, we do not use approximate NN but the accurate NN search for better measuring the improved recognition performance of our learning algorithms.

2.4 Multi-label Learning

Multi-label learning is an emerging and promising research topic in recent years. In multi-label learning, each instance is associated with multiple class labels simultaneously, so the traditional multi-class is its special case. The multi-label learning is originally developed for text classification. For example, Schapire and Singer [72] proposed a BoosTexter method extended from AdaBoost to classify multi-label text data. The success of multi-label learning for text data inspired researchers to apply it for dealing

with image data. Boutell *et al.* [73] first presented a framework to classify scene images by multi-label learning, which is also investigated by [74, 75]. Subsequently, Zhou *et al.* [76] developed a multi-instance multi-label (MIML) learning framework and applied it for scene classification. They decomposed MIML learning into multi-instance learning and multi-label learning tasks and thus proposed MIML-Boost and MIML-SVM algorithms respectively. To prevent decomposition when losing useful information, they then proposed a maximum margin method [77] for MIML learning and achieved improvement on scene classification. Moreover, Wang *et al.* [78] proposed a weighted KNN multi-label classification (KMLC) method for the image annotation problem. Li *et al.* [79] used contextual image decomposition for multi-label image annotation, which simultaneously maximizes inter-label difference and minimizes intra-label difference of the target label representations. In [11], sparse coding is involved for multi-label learning, which propagate label information from training images to test images. Lu *et al.* [80] also proposed a label propagation method by incorporating the context between visual keywords into the similarity between images. Label correlation information is utilized in [81] by integrating the correlated Green's function to annotate images into a graph. A recent survey [82] summarizes the recent works on multi-label learning.

2.5 Summary

In this chapter, we have briefly reviewed previous object recognition studies using feature quantization such as BoW related methods, as well as without using feature quantization like I2C related methods. It should be noted that besides these two categories, there are also many other methods to classify images using local descriptors. Therefore, not using feature quantization does not necessarily imply using S2S distance and vice versa. We also discussed some ANN methods for accelerating NN search and previous studies on multi-label learning.

Chapter 3

Learning Weighted Image-to-Class Distance

3.1 Introduction

In the formulation of the I2C distance, each local feature in the training set is given equal importance. This makes the I2C distance sensitive to irrelevant features, which are useless for measuring the distance and may hurt the classification accuracy. Besides, the performance of this I2C distance relies heavily on the large number of local features in the training set, which requires expensive computation cost during the NN search when classifying test images. On the other hand, a small training feature set may result in poor performance, although it requires less time for the NN search.

In this chapter, we propose a novel NN based classification method for learning a weighted I2C distance. For each local feature in the training set, we learn a weight associated with it in the training phase, thus decreasing the impacts of irrelevant features that are useless for measuring the distance. These weights are learned by formulating a weighted I2C distance function in the training phase, which is achieved by constraining the weighted I2C distance to the belonging class to be the shortest for each training image among its I2C distances to all classes. We adopt the large margin idea in Frome *et al.* [4] and formulate the triplet constraint in our optimization problem such that the I2C distance for each training image to its belonging class should be less than the distance to any other class with a large margin. Therefore, our method avoids the shortcomings of both non-parametric methods and most learning-based methods involving I2I distance

and descriptor quantization. This leads to a better classification accuracy than NBNN or those learning-based methods while requiring relatively smaller number of local features in the training set.

The main computational bottleneck in the I2C distance is the NN feature search due to the large number of features in the training feature set for each class. In most real world applications the training set is extremely large, which makes the NN search time expensive. Our training cost for learning the weight would be negligible compared to this heavy computation cost of the NN search. Therefore in this chapter we also propose two methods for accelerating the NN search, which make the I2C distance more practical for real world problems. These two methods reduce the candidate features set for the NN search from different aspects. The first method uses spatial division to split the image into several spatial subregions and restrict each feature to find its NN only in the same spatial subregion, while the second method ranks all local features in the training set by hubness score [83] and removes those features with low hubness score. Both methods are able to accelerate the NN search while maintaining or even achieving better classification accuracy.

We also extend the weighted I2C distance to the multi-label problem. Firstly, we use the weighted I2C distance to generate a label rank by ranking the I2C distances to all classes, which prevents the disadvantage of decomposition into multiple binary classifiers in traditional multi-label learning. Then we make use of label correlation for label prediction given the label rank, which implies label co-occurrence and is shown to be very important in multi-label learning [81, 84–86]. These two contributions make our method effective for classifying multi-label images.

3.2 Optimization Problem

In this section, we propose a large margin optimization framework to construct a weighted I2C distance by learning the weight associated with each local feature in the training set. These features are extracted from patches around each keypoint and represented by some local descriptors such as SIFT [12]. Let $F_{X_i} = \{f_{X_i1}, f_{X_i2}, \dots, f_{X_im_i}\}$ denote local features belonging to an image X_i , where m_i represents the number of features in X_i and each

Figure 3.1: The original unweighted I2C distance in NBNN (left) and our proposed weighted I2C distance (right).

local feature is denoted as $f_{X_{ik}} \in R^d, \forall k \in \{1, \dots, m_i\}$. The feature set of each class C_j is composed of features from all training images belonging to that class and is denoted as $F_{C_i} = \{f_{C_{j1}}, f_{C_{j2}}, \dots, f_{C_{jm_j}}\}$. Similarly, m_j represents the number of features in class C_j . The original unweighted I2C distance from image X_i to a class C_j is formulated as the sum of L_2 distance (Euclidean distance) between every feature $f_{X_{ik}}$ in X_i and its NN feature in class C_j denoted as $f_{X_{ik}}^{C_j}$, which is shown in the left part of Figure 3.1. Here, the NN feature $f_{X_{ik}}^{C_j}$ is searched over the feature set of class C_j that has the shortest L_2 distance to feature $f_{X_{ik}}$ in image X_i and the distance between them is denoted as $d_{X_{ik}}^{C_j}$. This NN search is time consuming when the training feature set is large, so we will discuss some acceleration methods in Section 3.4. The formulation of this I2C distance, as a special case of the Set-to-Set distance in Equation (2.1), is represented as:

$$Dist(X_i, C_j) = \sum_{k=1}^{m_i} \|f_{X_{ik}} - f_{X_{ik}}^{C_j}\|^2 = \sum_{k=1}^{m_i} d_{X_{ik}}^{C_j}, \quad (3.1)$$

where

$$f_{X_{ik}}^{C_j} = \arg \min_{k'=\{1, \dots, m_j\}} \|f_{X_{ik}} - f_{C_{jk'}}\|^2. \quad (3.2)$$

However, in this formulation each local feature in the training set is given equal importance. This makes the I2C distance sensitive to irrelevant features that are useless for measuring the distance and may hurt the classification accuracy. To discriminate relevant features from irrelevant ones, we associate each feature in the training set with a weight, which is learned through the training phase. Therefore our new weighted I2C

distance (as shown in the right part of Figure 3.1) is represented as follows:

$$Dist(X_i, C_j) = \sum_{k=1}^{m_i} w_{X_{ik}}^{C_j} \cdot \|f_{X_{ik}} - f_{X_{ik}}^{C_j}\|^2 = \sum_{k=1}^{m_i} w_{X_{ik}}^{C_j} \cdot d_{X_{ik}}^{C_j}. \quad (3.3)$$

For each local feature $f_{X_{ik}}$, the L_2 distance $d_{X_{ik}}^{C_j}$ between this feature and its NN $f_{X_{ik}}^{C_j}$ is multiplied with the weight $w_{X_{ik}}^{C_j}$ learned for this NN feature $f_{c,k}$. In fact, the original I2C distance can be viewed as every weight equating to 1 in this formula. Since all these weights in the training set are globally consistent and can be learned simultaneously, we concatenate all the weights to a weight vector W during the learning procedure. For consistency, a distance vector $D(X_i, C_j)$ is also constructed from image X_i to class C_j with the same length as W . In this way the weighted I2C distance can be formulated as:

$$Dist(X_i, C_j) = \sum_{k=1}^{m_i} w_{X_{ik}}^{C_j} \cdot d_{X_{ik}}^{C_j} = W^T \cdot D(X_i, C_j). \quad (3.4)$$

We adopt the idea of large margin to learn these weights. This idea is popular due to its success in SVM classifier, which simultaneously minimizes the empirical classification error and maximizes the geometric margin. For the binary classification problem, a hyperplane is optimized to create the largest separation between two classes. In our large margin framework for learning the weight, we optimize the triplet constraint in a way different from [4]. In [4], for each input image a triplet is constructed by selecting an image from the same class and an image from a different class, and the constraint is formulated that the I2I distance between images in the same class should be less than that in different classes with a margin. Besides the limitation incurred by I2I distance as described in [14], this triplet formulating method will cause too many triplet constraints in the distance learning, especially when there are a large number of training images in each class. However, by using the I2C distance, we construct our triplet for each input image just by selecting two classes, one as positive class that the image belongs to, and the other from any other class as negative class, hereby reducing the number of triplet constraints. Our triplet constraint is therefore formulated by keeping the I2C distance to the positive class to be less than that to the negative class with a margin. This is illustrated in Figure 3.2. For each input image X_i , the triplet with positive class C_p and negative class C_n should be constrained as:

$$W^T \cdot [D(X_i, C_n) - D(X_i, C_p)] \geq 1 - \xi_{ipn}. \quad (3.5)$$

Figure 3.2: The triplet constraint for distance learning. For an image X_i , its I2C distance to the positive class C_p should be less than that to the negative class C_n with a margin.

Here the slack variable ξ_{ipn} is used for soft-margin as in the standard SVM form.

We formulate our large margin optimization problem in the form similar to SVM. Since the initial weight of each feature is 1 as in the original I2C distance, we regularize the learned weights according to a prior weight vector W_0 , whose elements are all equal to 1. This is to penalize those weights that are too far away from this prior in the optimization problem and keep the consistency between the training and testing phase. The optimization problem is given as:

$$\begin{aligned}
 & \arg \min_{W, \xi} \quad \frac{1}{2} \|W - W_0\|^2 + C \sum_{i,p,n} \xi_{ipn}, \\
 & s.t. \forall i, p, n : \quad W^T \cdot [D(X_i, C_n) - D(X_i, C_p)] \geq 1 - \xi_{ipn}, \\
 & \quad \quad \quad \xi_{ipn} \geq 0, \\
 & \quad \quad \quad \forall k : \quad W(k) \geq 0.
 \end{aligned} \tag{3.6}$$

Here, the parameter C controls the trade-off between the regularization and error terms as in SVM optimization problem. Each element of the optimizing weight vector is enforced to be non-negative as distances are always non-negative. For a dataset with N_C classes,

the number of triplet constraints for each training image is $N_C - 1$ since there are $N_C - 1$ different negative classes, and the total number of triplet constraints in the optimization problem should be $(N_C - 1) \times N_C \times N_T = O(N_C^2 \times N_T)$, where N_T stands for the average number of training images in each class. This is a large reduction compared to the number of triplets in I2I distance [4], which needs $O(N_C^2 \times N_T^3)$ triplets for learning the weight. Such reduction on the number of triplets will result in a faster speed during the weight updating procedure. It should also be noted that since the dimension of the weight vector W (which is the number of local features in the training set) is usually much larger than the number of triplet constraints in the optimization problem, the regularization over the prior vector W_0 will not cause over-fitting problem.

3.3 Problem Solver

We solve the optimization problem of Equation (3.6) in the dual form using the method in [4] as:

$$\begin{aligned}
 \arg \max_{\alpha, \mu} \quad & -\frac{1}{2} \left\| \sum_{i,p,n} \alpha_{ipn} \cdot Q_{ipn} + \mu \right\|^2 + \sum_{ipn} \alpha_{ipn} \\
 & - \left[\sum_{ipn} \alpha_{ijk} \cdot Q_{ipn} + \mu \right] \cdot W_0, \\
 \text{s.t.} \quad & \forall i, p, n : 0 \leq \alpha_{ipn} \leq C, \\
 & \forall j : \mu(j) \geq 0,
 \end{aligned} \tag{3.7}$$

where $Q_{ipn} = [D(X_i, C_n) - D(X_i, C_p)]$ for simplicity. This dual form is solved by iteratively updating the dual variable α and μ . We update the α variables that violate the Karush Kuhn Tucker (KKT) [87] conditions by taking the derivative of Equation (3.7) with respect to α and then update μ to ensure the positiveness of weight vector W in each iteration. The updated formula of α and μ is given as:

$$\begin{aligned}
 \alpha_{ipn} &= \left[1 - \sum_{\{i',p',n'\} \neq \{i,p,n\}} \alpha_{i'p'n'} \langle Q_{i'p'n'} \cdot Q_{ipn} \rangle - \right. \\
 & \quad \left. \langle (\mu + W_0) \cdot Q_{ipn} \rangle \right] / \|Q_{ipn}\|^2, \\
 \mu &= \max\{0, -\sum_{i,p,n} \alpha_{ipn} \cdot Q_{ipn} - W_0\},
 \end{aligned} \tag{3.8}$$

and the KKT conditions are:

$$\begin{aligned}
 \alpha_{ipn} = 0 &\Rightarrow W \cdot Q_{ipn} \geq 1, \\
 0 < \alpha_{ipn} < C &\Rightarrow W \cdot Q_{ipn} = 1, \\
 \alpha_{ipn} = C &\Rightarrow W \cdot Q_{ipn} \leq 1.
 \end{aligned} \tag{3.9}$$

The operation of $[f(x)]_{[0,C]}$ is to clip the value of $f(x)$ in the region $[0, C]$, and $\{i', p', n'\}$ represent any triplets in constraint set that are different from $\{i, p, n\}$. Since the optimization problem is convex, this objective function is guaranteed to reach the global minimum after iterative updating. The number of α variables is equal to the number of triplet constraints, which is much less than that in the I2I distance framework that was mentioned before. After convergence, the weight vector W in the primal problem can be calculated by:

$$W = \sum_{i,p,n} \alpha_{i,p,n} \cdot Q_{i,p,n} + \mu + W_0. \tag{3.10}$$

As all weights are optimized simultaneously in the weight vector, this consistency will ensure global ranking of the weighted I2C distances to all classes.

3.4 Efficiency Improving on the NN Search

The main computational bottleneck in the I2C distance calculation is the NN feature search, as the number of training features in each class is usually very large especially when using dense sampling strategy for feature extraction. The large amount of training features also requires large memory cost during the NN search. In this section, we propose two methods to reduce candidate feature set for the NN search in addition to these NN approximation algorithms. They can accelerate the NN search and thus make our I2C distance more practical for real world problems.

3.4.1 Feature Reduction by Spatial Division

The first method for reducing candidate feature set to accelerate the NN search is dividing the image into subregions of equal area. So the NN of each feature in each subregion is restricted to those features in the same subregion from candidate feature set of each

Figure 3.3: The I2C NN feature search with spatial division. We adopt the idea of spatial pyramid by restricting the NN of each feature in the same subregion at each spatial level.

class. Figure 3.3 illustrates this spatial division. At level 1, images are divided into 2×2 subregions, so the candidate feature set is reduced to $1/4$ and at level 2 it is reduced to $1/16$. This spatial division is inspired by Lazebnik *et al.* [2] who divided the image into subregions at increasingly fine resolutions for each level and then merged them together, which is known as spatial pyramid match (SPM). Since each feature is more likely to find its NN in the same subregion, using spatial division to reduce the candidate set will preserve the classification accuracy of weighted I2C distance while accelerating the NN search. For scene images that usually have geometric structures and classes that manifest common spatial layout, this spatial division will not only accelerate the NN search, but also improve classification accuracy, as it can reduce the rate of false NN match. Even for datasets with less spatial aligned images, this spatial division can still work. This is because the NN of each local feature in a test image is searched over every training image in the same subregion and it is less likely that none of the training images contain the object in the same subregion as in the test image.

In addition to accelerating the NN search for improving efficiency by a single level spatial division, it can also be used to further improve the classification accuracy by adding the weighted I2C distances from different spatial levels to formulate the I2C distance with SPM as shown in Figure 3.3, which is analogous to the spatial pyramid [2] and pyramid match [22]. Although using spatial division at a single spatial level can

accelerate the NN search, the SPM of combining multiple spatial levels for improving the classification accuracy requires more running time, and this is the trade-off between accuracy and efficiency. Note that the spatial subregions cannot be divided too finely, as it will make the NN search too sensitive to the spatial layout and increase the number of false matches. In the experiments of this chapter, we use spatial division for learning I2C distance only at spatial level 1 (2×2 subregions) and level 2 (4×4 subregions).

3.4.2 Feature Reduction by Hubness Score

Spatial division is a simple but effective method for reducing candidate feature set. However, this method does not remove any features in the training set and therefore still requires large memory cost during the NN search. For such situation, we develop another method which explores the hubness of features in candidate set for feature reduction. This is inspired by the work of [83], which measured data points using hubness score and explored its importance in many NN problems. Specifically, the hubness of a data point is reflecting the number of times this point is matched by other points as their NN. A hubness score is therefore introduced to measure the hubness of each data point. For those points with high occurrences as NN points by others, they should have high hubness score and thus are recognized as hub points.

In our problem where the weighted I2C distance is used for classification, we define a new hubness score for measuring the importance of each feature (here a data point is represented as a feature) in candidate set. Those unimportant features with low hubness scores are removed for improvement in speed. Inspired by the triplet in our learning framework which is composed of positive and negative classes, the counting of occurrence as matched by other features should also be split into positive and negative parts, since the occurrence of being matched by features in the belonging class plays a different role to that in other classes. Intuitively, we would remove those features frequently matched by features from other classes while less likely to be matched from the belonging class. Let $N_p(x)$ denote the number of times a feature is matched by other features from the belonging class, which we also call positive match. Similarly, we denote the negative match $N_n(x)$ as the number of times a feature is matched by features from other classes.

We define the hubness score of each feature x by:

$$h(x) = \frac{N_p(x)}{N_n(x)}. \quad (3.11)$$

For all features in the training set, we rank them in the order of descending hubness score, and only preserve features with higher hubness score. Other features are removed permanently and no longer put into the candidate feature set. We only learn the weights for these preserved features and use them for NN match in the testing phase.

3.4.3 Discussion of the Two Efficiency Improving Methods

We discuss the advantages and disadvantages of the two proposed methods for accelerating the NN search. The first method uses spatial division to reduce candidate set for each feature in a test image during the NN search. Since determining which subregion a feature in test image belongs to is simply calculated by its pixel location, the selection of reduced candidate feature set for the NN search will require little time in the testing phase. For geometric structured images, this method performs even better than the one without spatial division. However, this method does not remove any features in the training set, therefore it still requires large memory cost during the NN search. The number of weights for learning is not reduced either, which may result in a long running time during learning when the training set is very large.

The second method does not have the above mentioned problems. Since the features with low hubness scores are removed permanently after ranking all training features, it can reduce the memory cost during the NN search in the testing phase. It can also speed up the weight learning procedure, as lesser number of weights is required for learning. However, unlike spatial division that can improve classification accuracy for structured images, this method may drop in accuracy when most of the training features are reduced. In addition, every feature in the training set needs to count its positive and negative matches for calculating hubness score, which increases the computation cost in the training phase, although faster NN search speed in the testing phase is gained.

To achieve the best efficiency, both methods can be combined together for the task of NN search during the I2C distance calculation.

3.5 Multi-label Image Classification

We extend the learning weighted I2C distance for the multi-label learning problem. In this problem, the triplet constraint is also formulated with a reference image, a positive class and a negative class. However, the positive class is now not unique for each reference image. So the number of triplet constraint for each reference image is increased to be the number of positive classes times the number of negative classes. Then the large margin learning procedure for updating weight is the same as the learning algorithm in Section 3.2.

The learned I2C distance generates a label rank for a test image but do not provide the exact labels in prediction, so we need to split the label rank into positive and negative parts and thus predict the labels for test image. We use label correlation to do this task for predicting label, which is widely used for multi-label learning [81, 84–86] and shown to be very effective. The idea of exploring the correlation between pairwise labels comes from the fact that some labels are usually co-assigned to the same image while some other labels are unlikely to appear in one image. For example, an image containing “plane” is likely to also contain “sky”, while it is much less likely to contain “car”. By using the label correlation generated from training images to split the label rank for test image prediction, we are able to ensure the consistency of label correlation between training and test images.

We construct the label correlation from training set using the method in [84], which counts the combination of frequency for pairwise labels. A label correlation matrix M is built with its element M_{ij} representing the correlation between label l_i and l_j . The value of M_{ij} is defined in [84] as follows:

$$M_{ij} = \frac{AD - BC}{\sqrt{(A + B)(C + D)(A + C)(B + D)}}, \quad (3.12)$$

where A,B,C,D are the percentages of frequency count in training set for the combinations of the co-occurrence defined as follows:

$$\begin{aligned} A &= l_i \wedge l_j, \\ B &= l_i \wedge \neg l_j, \\ C &= \neg l_i \wedge l_j, \\ D &= \neg l_i \wedge \neg l_j. \end{aligned}$$

Then we use this label correlation matrix to split the label rank into positive part and negative part, where labels in the positive part are the predicted labels for a test image. The splitting point is initially set between the top rank and the second rank, so that the top label is always in the positive part to ensure at least one label is assigned to the test image. Then we shift the splitting point along the label rank by deciding whether the current label should be put into the positive part. This is done by checking whether there is any label “close” enough to the current label. A threshold θ is required to define the closeness of two labels so that if M_{ij} is larger than θ , label l_i and l_j is recognized as close to each other. If there is no label in the positive part “close” to the current label, the shifting is finished and the splitting point is fixed to separate the positive part and negative part. Note that the order of labels in the rank is determined by weighted I2C distance and keeps unchanged, while label correlation only determines the position of splitting point for label prediction, since the effort to put relevant labels at top of the rank is mainly achieved by label ranking.

The quality of label prediction is measured by *Hamming Loss*, which checks the difference between predicted labels and true labels using the XOR operation. Let P_i and Y_i denote the predicted label set and true labels for a test image X_i . The *Hamming Loss* of X_i is defined by:

$$HL(X_i) = \frac{P_i \Delta Y_i}{N_c}, \quad (3.13)$$

where Δ denotes the XOR operation and N_c is the number of total labels. We use *Hamming Loss* averaging over the whole training set to determine the best θ value for predicting test images. We also use it for measuring the performance of our label prediction method in experiment.

For measuring the quality of multi-label scene classification, we use five evaluation metrics commonly used in previous works for multi-label learning [72, 74–76]: *Hamming loss*, *One-Error*, *Coverage*, *Ranking Loss* and *Average Precision*. Note only under the *Average Precision* metric, larger values indicate better results, while for the other 4 metrics, it should be the smaller the better. We briefly describe the measurement of these five evaluation metrics:

- *Hamming loss*: It measures the quality of predicted label by checking the difference between predicted labels and true labels using the XOR operation. Formula is given in Equation (3.13).

- *One-Error*: This metric only evaluates whether the top label in the rank list is not in the true label set. Let $l_1(X_i)$ denote the top label ranked for image X_i and Y_i denote the true label set, for a test set with N_x images, the average one-error is:

$$OneError = \frac{1}{N_x} \sum_i [l_1(X_i) \notin Y_i].$$

Here, the $[f(x)]$ operation gives 1 when $f(x)$ is true and 0 otherwise.

- *Coverage*: This metric measures how far away the rank list is from “covering” all labels in the true label set. Let $rank(l, X_i)$ denote the position of label l in the rank list of X_i , then Coverage is given as:

$$Coverage = \frac{1}{N_x} \sum_i \max_{l \in Y_i} rank(l, X_i) - 1.$$

- *Ranking Loss*: This metric counts the fraction of label pairs between true labels and irrelevant labels that are in the reverse order in the rank list:

$$RankingLoss = \frac{1}{N_x} \sum_i \frac{F(X_i)}{|Y_i| \cdot |\bar{Y}_i|},$$

where $F(X_i) = |\{(l, l') | rank(l, X_i) > rank(l', X_i), l \in Y_i, l' \in \bar{Y}_i\}|$ is the number of reverse ordered label pairs in the rank list for image X_i .

- *Average Precision*: This measure is similar to Ranking Loss for measuring the quality of the whole rank list. It evaluates the average fraction of true labels ranked above a particular true label $l \in Y_i$:

$$Aver.Prec. = \frac{1}{N_x} \sum_i \frac{1}{|Y_i|} \sum_{l \in Y_i} \frac{G(X_i)}{rank(l, X_i)},$$

where $G(X_i) = |\{l' \in Y_i | rank(l', X_i) \leq rank(l, X_i)\}|$ is the number of true labels ranked above label l .

For the task of automatic image annotation, we also use F_1 score to evaluate our method and for comparing with previous methods. This evaluation metric measures the quality of retrieved relevant images with each keyword, which is treated as class label here. Given each keyword, the number of images annotated with this label in ground

truth is denoted as N_t , the number of images annotated with this label by the classifier is denoted as N_r , and the number of images correctly annotated with this label by the classifier is denoted as N_c . Then F_1 score is defined as:

$$\begin{aligned} Precision &= \frac{N_c}{N_r}, \\ Recall &= \frac{N_c}{N_t}, \\ F_1 &= \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}. \end{aligned}$$

3.6 Experiments

3.6.1 Classification Results of Weighted I2C Distance

We first evaluate our proposed method in five publicly available single-label datasets: Scene-15, Sports, Corel, Caltech 101 and Caltech 256 datasets. We describe them briefly as follows:

- **Scene-15:** Scene dataset consists of 15 scene categories, among which 8 were originally collected by Oliva *et al.* [88], 5 added by Li *et al.* [89] and 2 from Lazebnik *et al.* [2]. Each class has about 200 to 400 images, and the average image size is around 300×250 pixels. Following [2], we randomly select 100 images per class for training, and test on the rest. The average of each per-class accuracy is reported for evaluation.
- **Sports:** Sports event dataset is firstly introduced in [90], consisting of 8 sports event categories. The number of images in each class ranges from 137 to 250, so we follow [90] to select 70 and 60 images per class for training and test respectively. Since images in this dataset are usually very large, they are first resized such that the largest x/y dimension is 500.
- **Corel:** Corel dataset contains 10 scene categories published by Corel Corporation. Each class contains 100 images, and we follow [91] to separate them randomly into two subsets of equal size to form the training and test set. All the images are of the size 384×256 or 256×384 .

Table 3.1: Classification accuracy (%) of LWI2C compared to NBNN for each dataset with its attributes.

Dataset	Classes	Training Images	Test Images	Accuracy (%)	
				LWI2C	NBNN
Scene-15	15	1500	2985	80.0±0.4	72.8±0.7
Sports	8	560	480	82.0±1.2	67.6±1.1
Corel	10	500	500	91.1±1.1	85.7±0.9
Caltech 101	101	1515	1515	52.1±1.4	37.1±1.2
Caltech 256	256	3840	6400	24.0±1.1	18.3±1.3

- **Caltech 101:** Caltech 101 dataset is a large scale dataset containing 101 categories [92]. The number of images in each class varies from about 30 to 800. This dataset is more challenging due to the large number of classes and intra-class variance. Following the widely used measurement by the community we randomly select 15 images per class for training. For testing, we also select 15 images for each class and report the mean accuracy.
- **Caltech 256:** Caltech 256 dataset [93] is even larger than Caltech 101. Each class contains at least 80 images. The total of 256 categories make this dataset much more challenging than the other four datasets. We use this dataset to validate the scalability of our methods. In the experiment we randomly select 15 images per class for training and 25 different images per class for test.

Since the training and test set are selected randomly, we repeat each dataset for 5 times and report the average result. For feature extraction, we use dense sampling strategy and SIFT features [12] as our descriptor, which are extracted over every 8 pixels for all datasets. Although this feature extraction is simple and even discard color information, encouraging performance can be achieved. For the parameter C in our learning problem, we empirically fix it to 1 throughout the experiment. We name our method as LWI2C, short for Learning Weighted Image-to-Class distance.

To validate the effectiveness of our learned weight, we first compare LWI2C to NBNN [14], which uses unweighted I2C distance for classifying. Table 3.1 gives the classification accuracy of both methods, averaged over all classes. The number of classes in each dataset and the number of training and test images are also listed. Results in Table 3.1

Figure 3.4: Per-Class classification accuracy in Scene-15, Sports and Corel datasets for comparing LWI2C with NBNN.

show that in every dataset, our LWI2C outperforms NBNN significantly, which confirms our expectation that the weight learned in training set is able to enhance the discriminate of I2C distance for classifying test images.

For further exploring the improvement over the original I2C distance, we also compare the Per-Class classification accuracy of LWI2C to NBNN. We show the comparison for Scene-15, Sports and Corel datasets. The weighted I2C distance is able to outperform unweighted I2C distance in most classes for these three datasets as shown in Figure 3.4. We find that the improvement is more apparent in those challenging classes that NBNN is unable to classify well, e.g. the worst 3 classes in Scene-15, the worst 4 classes in Sports and the worst 2 classes in Corel. Therefore, we can conclude that our large margin framework for learning the weight is able to significantly improve the classification accuracy on those challenging classes, which are difficult to be classified well by the unweighted I2C distance.

For this conclusion to be more convincing, we also show some examples for the distributions of weighted local features. We choose two most challenging classes (“owls” and “mountains”) in Corel dataset that LWI2C improves significantly over NBNN and

Figure 3.5: Distributions of local features with high weight (red) and low weight (blue) for classes of “owls”, “mountains” and “flowers” in Corel dataset.

another easy class (“flowers”) that both methods can achieve high accuracy. For each class we select the top 1% high weighted features and top 1% low weighted features and show some example images in Figure 3.5. We can see that in the two challenging classes most high weighted features are located on the object, while the location of low weighted features are more arbitrary, i.e. either on the object or the background. As in LWI2C those features with high weight are given more importance, the weighted I2C distance is able to largely improve the classification accuracy over the original unweighted distance. However, in the class of “flowers”, both high and low weighted features are difficult to be distinguished from either the object or background. As both weighted and unweighted I2C distances can easily classify images in this class, this distribution shows that features located on both the object and background in this class are useful for the classification. Some predicted examples in this dataset are also shown in Figure 3.6. In the top row, both methods are able to correctly classify these images. In the middle row, images are wrongly classified by NBNN. However, after learning the weight, they can be correctly classified by our LWI2C. In the bottom row, we also show images that are unable to be

Ground Truth & LWI2C & NBNN	skiing	owls	mountains	flowers	beach	buildings	tigers

Ground Truth & LWI2C	skiing	owls	owls	owls	mountains	mountains	flowers	buildings
NBNN	building	tigers	flowers	food	skiing	elephants	food	elephants

Ground Truth	skiing	owls	mountains	beach	buildings	buildings	food
LWI2C	owls	tigers	buildings	buildings	food	owls	tigers
NBNN	food	tigers	food	buildings	food	food	tigers

Figure 3.6: Predicted examples in Corel dataset. Images in the top row are correctly classified by both LWI2C and NBNN. The middle row shows images that are correctly classified by LWI2C but wrongly classified by NBNN, and images in the bottom row are wrongly classified by either LWI2C or NBNN.

correctly classified even after learning the weight. We can see that some of these images are difficult to be classified as they have less connection to their ground truth labels, i.e., there is no sea and sands in this example “beach” image, and the two “building” images do not contain the contour of buildings compared to “building” images in the top and middle rows.

Compared to the published result of NBNN [14] in Caltech 101 (65.0%), our implementation of NBNN achieves lower result than their best reported performance. We notice that they have used multiple scales and more densely sampled features in their experiment, therefore the number of features is about 20 times more than the one in our experiment. This shows that the performance of the I2C distance without training phase relies heavily on the large number of complex features, which require heavy computation cost for the NN search. In contrast, we use simpler and a smaller number of features in our experiment and therefore achieve a much lower result for NBNN, which proves that

using a small number of features result in poor performance for the I2C distance without training phase. However, by learning the weight using our proposed method, the performance of the I2C distance can be significantly improved using the same, small number of features. In addition, even using a large number of local features in the training set same as that in [14], we are unable to reproduce the result reported in [14]. We guess this may be due to the different parameter settings in the SIFT feature extraction stage. Since both methods use the same feature descriptor as input in our experiment, the comparison of our method with NBNN is fair. This experiment also shows that our large margin framework for learning the weight is capable of dealing with large-scale datasets, while the computation cost of weight updating procedure in training is nearly negligible compared to the NN search in the I2C distance calculation as mentioned before.

3.6.2 Results of Accelerating the NN Search

We next analyze the performance of LWI2C as well as NBNN at different spatial levels. Table 3.2 shows the results of both methods from level 0 to level 2 as well as their combinations (SPM). Compared to level 0, the classification accuracies of both methods are improved at either spatial level 1 or level 2. This proves that spatial division is able to not only accelerate the NN search, but also improve the classification accuracy. Since images in the Sports dataset do not have such common spatial layout like that in scene constraint structures of the other datasets, but still gain improvement when using spatial division, we suggest that the spatial division is a useful strategy to help improve the I2C distance in most real-world images. We also notice that the largest improvement by spatial division is achieved in Caltech 101 dataset, where the performances of both methods at level 1 are improved over 6% compared to level 0 and further improved about 4% at level 2. This may be because that objects in most images of Caltech 101 dataset are often located at the center, so spatial division is able to maintain the common spatial layout in each subregion and reduce the rate of false NN match.

Since SPM combines the I2C distances of all spatial levels, it achieves the best classification accuracy in each dataset as shown in Table 3.2. However, SPM requires calculating the I2C distance and learning weight for LWI2C at each spatial level, so its computation cost is the sum of each spatial level and thus very expensive. Since the improvement

Table 3.2: Classification accuracy (%) of LWI2C compared to NBNN on different spatial levels as well as SPM for Scene-15, Sports, Corel, Caltech 101 and Caltech 256 datasets. L0 denotes level 0 of 1×1 subregions, L1 denotes level 1 of 2×2 subregions and L2 denotes level 2 of 4×4 subregions. SPM combines three spatial levels and achieves the best accuracy, while the accuracy of spatial levels at L1 and L2 is better than L0.

Dataset	LWI2C (%)				NBNN (%)			
	L0	L1	L2	SPM	L0	L1	L2	SPM
Scene-15	80.0	82.2	81.7	83.8	72.8	75.6	77.6	78.7
Sports	82.0	83.5	84.0	84.1	67.6	71.3	71.2	71.4
Corel	91.1	91.2	91.6	92.2	85.7	87.4	87.3	88.2
Caltech 101	52.1	58.1	62.1	66.7	37.1	43.6	48.1	50.4
Caltech 256	24.0	25.5	29.2	30.2	18.3	20.0	21.0	22.2

of SPM is limited compared to single spatial level in these datasets, using only spatial division at level 1 or level 2 for classification is a better strategy when efficiency is the primary issue. For example, the Caltech 256 dataset contains much more images and classes than the others, so using spatial division at level 2 to learn its I2C distance is faster while its accuracy is close to that using SPM.

Although spatial division and SPM is able to improve the classification performances of both LWI2C and NBNN, it should be emphasized that by learning the weight for each training feature, our LWI2C is more discriminative than the unweighted I2C distance in NBNN in every spatial level and its combination, which again validate the effectiveness of our large margin framework for learning the weight.

The result of spatial division is encouraging for reducing the candidate feature set during the NN search. However, a large number of features in the training set are still stored in memory during the testing phase, because features in different subregions are required to be matched by features in the corresponding subregions for each test image. Such a large number of training features not only occupy large memory space, but also require substantial computation cost for learning the weight associated with each feature. These problems can be overcome by our second method for improving the efficiency. This new method reduces the training features by the use of hubness score.

To evaluate the effectiveness of this method, we compare the performances of the training set with different sizes of preserved features. Here the amount of improvement on efficiency is represented by the percentage of preserved training features to the original,

Figure 3.7: Comparing the classification accuracy of LWI2C to NBNN for Scene-15, Sports and Corel datasets with various percentages of preserved training features in the training set.

since less number of preserved features need less computation cost for the NN search and achieve better improvement on efficiency. Figure 3.7 shows the results of both LWI2C and NBNN with percentages of preserved training features ranging from 1% to 100% for Scene-15, Sports and Corel datasets. In all the three datasets, the accuracy reduced by our LWI2C method is nearly negligible even when only 5% training features are preserved, but this reduction starts to become obvious when the preserved training features are reduced from 5% to 1%. However, massive reduction of training features will not allow NBNN to maintain satisfactory result, as its accuracy drops quickly in every dataset. This result indicates that the performance of NBNN relies heavily on the size of candidate

Figure 3.8: Comparing the classification accuracy of LWI2C to NBNN for Caltech 101 in spatial level 2 with various percentages of preserved training features in the training set.

training set, while our method is able to maintain the classification accuracy when there are only a small number of training features due to the effectiveness of learned weight. It should be emphasized that this method for feature reduction not only accelerates the NN search, but also speeds up weight learning and requires less memory space.

We also evaluate the performance of combining both spatial division and hubness score. We use Caltech 101 dataset in spatial level 2 for this experiment with percentages of preserved features ranging from 1% to 100%. Results on both LWI2C and NBNN are reported in Figure 3.8. Although only 15 images in each class are used for training in this dataset, which are much less than the training sets in the previous three datasets, we can see that the classification accuracy of our LWI2C method is still close to the original until the percentage of preserved training features drops to below 10%, whereas the accuracy of NBNN drops much faster compared to LWI2C.

3.6.3 Computation analysis

We also show the computation cost of our learning method as well as the accelerated NN search using our proposed two strategies. The average running time for updating the weight using our large margin optimization are shown in Table 3.3, where the running time for the NN search of each image without acceleration are also list as comparing. All of the experiments are running on **Intel x86 Xeon CPU E7320@2.13GHz**. We

Table 3.3: The average running time for weight learning and NN search of each image. (s=second, h=hour)

	Weight Learning	NN Search
Scene-15	381s	155s
Sports	171s	270s
Corel	74s	52s
Caltech 101	2151s	78s
Caltech 256	3.5h	239s

Table 3.4: The average running time for the NN search of each image on different spatial division level and various percentages of preserved training features in the training set. (second)

	Percentage of remaining training set										
	1%	2%	5%	10%	15%	20%	25%	30%	40%	50%	100%
L0	1.9	2.8	5.2	9.0	12.8	16.8	20.8	24.8	30.4	33.1	78.0
L1	1.4	1.8	2.7	3.6	4.5	5.5	6.5	7.6	9.4	11.4	24.7
L2	1.6	2.2	3.3	4.3	5.1	5.6	6.1	6.6	7.3	7.9	12.1

can see that in Scene-15, Sports and Corel datasets, due to the small number of classes, the weight learning procedure runs quickly and requires little computation cost. With the increased number of classes, more triplet constraints and more weights are required to be updated. Therefore the running time for updating the weight in Caltech 101 and 256 datasets are increased. However, the computation cost of such training procedure is not as demanding as the online NN search. As Table 3.3 shows, it requires about 1 to 4 minutes for the NN search of only one image. This means for a dataset like Caltech 101, which contains 1515 test images, the total running time for the NN search is about 32 hours. For the largest dataset of Caltech 256, which contains 6400 test images, the running time is increased to 17 days. Such large computation cost makes the I2C distance impractical for large scale dataset.

However, with our proposed acceleration method, the running time for the NN search can be significantly reduced. Taking Caltech 101 dataset as example, using spatial division at level 2, the running time is reduced to 12.1 seconds as shown in Table 3.4. By combining both spatial division and hubness score, the computation cost can be further reduced. As Figure 3.8 in previous subsection shows, the classification accuracy of preserving 10% training features on spatial level 2 is comparable to the original for our

LWI2C. However, the computation cost is reduced to 4.3 seconds per image as shown in Table 3.4. So the total running time for the NN search of the whole test set is only 1.8 hours, which would make the I2C distance practical for applying to large scale dataset.

3.6.4 Comparing to Previous Results

We also compare our method with recent published results in these datasets. All the results are summarized in Table 3.5. The best classification accuracy in our experiment is achieved by combining LWI2C with SPM in each dataset. Although we use single local feature in our experiment and do not even consider color information, our result is able to achieve comparative performance to the state-of-the-art results in most of these datasets. Specifically, in Scene-15 dataset, our result is able to outperform most of previous methods [2, 15, 21, 95], and is comparative to [19, 94, 96]. However, we find that the best result among these results reported in [19] use multi-scale and densely sampled keypoints for feature extraction as well as multiple feature combination, which is much too complex, while our method uses single local feature but is still comparable to most methods. Similar comparison is in Sports dataset, where our method significantly outperforms [90, 98] and is comparable to [19]. For Corel, Caltech 101 and Caltech 256 datasets, our method is also comparative to most other methods that report their results in these datasets recently.

3.6.5 Results on the Multi-label Problem

For the extension of our weighted I2C distances on the multi-label problem, we evaluate its performance on two multi-label datasets: natural scene dataset and MSRC dataset. Both datasets use the same feature generation method as in previous subsection. We denote the multi-label version of our learning method as **ML-LWI2C** and the unweighted I2C distance as ML-I2C.

The natural scene dataset is first used in [76], which consists of 2000 images with 5 class labels: *desert*, *mountains*, *sea*, *sunset* and *trees*. About 22% of the images contain more than 1 label and each image is associated with 1.24 labels in average. We split the whole dataset into 10 equal groups and perform a 10-fold cross validation experiments same as that in [74, 75] for comparing. Specifically, each time one group is selected for

Table 3.5: Comparing to recently published results for classification accuracy (%).

Dataset	Method	Accuracy (%)
Scene-15	Ours	83.8
	Lazebnik <i>et al.</i> [2]	81.4
	Liu <i>et al.</i> [94]	83.3
	Rasiwasia <i>et al.</i> [95]	72.2
	van Gemert <i>et al.</i> [21]	76.5
	Bosch <i>et al.</i> [96]	83.7
	Yang <i>et al.</i> [15]	80.3
	Wu <i>et al.</i> [19]	84.1
	Zhou <i>et al.</i> [97]	83.5
	Tuytelaars <i>et al.</i> [51]	85.0
Sports	Ours	84.1
	Li <i>et al.</i> [90]	73.4
	Wu <i>et al.</i> [98]	78.5
	Wu <i>et al.</i> [19]	84.2
Corel	Ours	92.2
	Lu <i>et al.</i> [99]	77.9
	Lu <i>et al.</i> [91]	90.0
	Gao <i>et al.</i> [28]	89.4
Caltech 101	Ours	66.7
	Yang <i>et al.</i> [15]	67.0
	Wang <i>et al.</i> [24]	65.4
	Wu <i>et al.</i> [19]	65.2
	Gu <i>et al.</i> [100]	65.0
	Tuytelaars <i>et al.</i> [51]	69.2
	McCann <i>et al.</i> [52]	66.1
Caltech 256	Ours	30.2
	Yang <i>et al.</i> [15]	27.7
	Griffin <i>et al.</i> [93]	28.3
	Gao <i>et al.</i> [28]	29.7
	McCann <i>et al.</i> [52]	33.5

Table 3.6: Performance evaluation in natural scene dataset, $\downarrow$ means the smaller the better and $\uparrow$ means the larger the better.

Method	Evaluation Metric				
	<i>Hamming Loss</i> $\downarrow$	<i>One-Error</i> $\downarrow$	<i>Coverage</i> $\downarrow$	<i>Ranking Loss</i> $\downarrow$	<i>Average Precision</i> $\uparrow$
ML-LWI2C	0.129	0.190	0.624	0.091	0.881
ML-I2C	0.159	0.311	0.883	0.156	0.804
InsDif [75]	0.152	0.259	0.834	0.140	0.830
ML-KNN [74]	0.169	0.300	0.939	0.168	0.803
BoosTexter [72]	0.179	0.311	0.939	0.168	0.798
Rank-SVM [101]	0.253	0.491	1.382	0.278	0.682
MIML-Boost [76]	0.189	0.351	0.989	0.181	0.777
MIML-SVM [76]	0.180	0.327	1.022	0.187	0.783

test and the rest are used for training, and the result is averaged over 10 runs. We compare our method with ML-I2C as well as state-of-the-art results of previous multi-label methods, including InsDif [75], ML-KNN [74], BoosTexter [72], Rank-SVM [101] and two MIML algorithms MIML-Boost and MIML-SVM in [76] since these methods have reported the results in this dataset.

Results are summarized in Table 3.6. We can see that for every evaluation metrics at each column, our method of ML-LWI2C performs the best. Since the evaluation on *One-Error*, *Coverage*, *Ranking Loss* and *Average Precision* do not require the classifier to output the exact predicted labels, the performance improved over these four metrics should be contributed by our large margin optimization framework of LWI2C for learning the effective weight and the use of the I2C distance. This is because the triplet constraints in the optimization problem try to push forward the truth labels to the top of the label rank during the learning procedure. Meanwhile, *Hamming loss* is used for measuring the quality of predicted labels, and we find under this evaluation our LWI2C is also able to achieve significantly better performance compared to previous methods. Even without the learned weight, using our label prediction method the unlearned I2C distance can also predict labels relatively well, as can be seen by comparing the *Hamming loss* of I2C to previous published results.

We also use MSRC dataset for evaluating our proposed methods. This dataset is widely used in recent years for image annotation problem [79–81, 102, 103]. This dataset

Table 3.7: Performance evaluation in MSRC dataset, $\downarrow$ means the smaller the better and $\uparrow$ means the larger the better.

Method	Evaluation Metric				
	Hamming Loss $\downarrow$	One-Error $\downarrow$	Coverage $\downarrow$	Ranking Loss $\downarrow$	Average Precision $\uparrow$
ML-LWI2C	0.071	0.200	3.254	0.066	0.809
ML-I2C	0.099	0.330	5.246	0.145	0.655

Figure 3.9: F_1 score for measuring the performance of image annotation.

consists of 591 images assigned to 21 predefined labels (As suggested by MSRC, the label “horse” and “mountain” are ignored due to few labeled images). Around 80% images are assigned to more than 1 label with about 3 labels per image in average. We randomly split the dataset into 391 training images and 200 test images, with additional constraint that each label contains at least 10 images in the training set. Results are averaged over 5 runs with different training and test sets.

We first report our result of ML-LWI2C and the unweighted ML-I2C on the previous five evaluation metrics. As shown in Table 3.7, the weighted I2C distance once again greatly improves the performance over the unweighted one in all metrics. Although there are more labels in this dataset compared to scene dataset, results on *Hamming Loss*, *Ranking Loss* and *Average Precision* is similar to that in scene dataset. As each image is annotated with more labels in average, the value of *Coverage* is also increased. But compared to ML-I2C, ML-LWI2C only requires about half of rank list for covering all

the true labels. For *One-Error*, we notice this error increased a little compared to that in scene dataset. Nevertheless, the better result on *Hamming Loss* indicates that true labels are still at the front of the rank list and is able to be predicted by our method.

In addition, we evaluate our method and compare with other methods by F_1 score, since this evaluation metric is commonly used in automatic image annotation studies and has been used by others in this dataset. Our methods are compared with MBRM [104] and SSK combined with CBKP [80]. Same as [80], the top 3 keywords in the rank list is taken as automatic annotation of a test image. All the results are summarized in Figure 3.9. Without learning the weight, ML-I2C is a little better than MBRM but lower than both ML-LWI2C and SSK+CBKP. However, our ML-LWI2C with weight learnt performs better than SSK+CBKP and has achieved state-of-the-art performance.

3.7 Summary

In this chapter, we introduced a novel method for nearest-neighbor classification by learning weighted Image-to-Class distance function. We reduced the effects of irrelevant patches e.g. cluttered background and enhanced relevant patches by learning their weights in a large margin framework. For efficiency, we presented two approaches to reduce the heavy computation cost of the NN search. We also extend this method to multi-label problem. Experiments in several prevalent image datasets show promising performance of our method compared with other methods.

Chapter 4

Image-to-Class Distance Metric Learning

4.1 Introduction

A good metric plays an important role in many real world problems. Both the original I2C distance and the weighted I2C distance proposed in Chapter 3 use Euclidean distance for measuring the Feature-to-Set distance, which may not be the optimal distance metric for measuring the distance between local features in real image classification problems, since it ignores the semantic gap between local features and class labels. Therefore, choosing a good distance metric is a critical problem for improving the performance of the I2C distance. In this chapter, we propose a distance metric learning method to learn a Mahalanobis I2C distance. For each class, we learn the class specific Mahalanobis metric to combine with the corresponding I2C distance. Compared to learning only a global metric, the class specific metric can better preserve the discriminative information for each class during the learning procedure and make the Mahalanobis I2C distance more discriminative especially when there are a large number of classes, since learning a global Mahalanobis distance metric may not be sufficient enough to improve the I2C distance. See Figure 4.1 as example. We learn a metric M_j for each class C_j , so the I2C distance $Dist(X_i, C_j)$ from image X_i to each class C_j is combined with the learned metric M_j . For prediction, the class with the shortest I2C distance for this image is selected as the predicted class.

To learn these Per-Class metrics, we also adopt the idea of large margin from SVM in our optimization problem. For each training image, we separate the I2C distance to

Figure 4.1: The prediction of image X_i from class C_1 to C_{N_C} , Where N_C is the total number of classes. The test image X_i is denoted as rectangle and each ellipse denotes a class with training images (rectangles) inside it.

its belonging class from the distance to any other class by a large margin, and form a large margin convex optimization problem as an instance of semi-definite programming (SDP). Then we apply an efficient subgradient descent method to solve this optimization problem. Since all these Per-Class metrics are learned simultaneously in the optimization problem, we ensure the learned Mahalanobis I2C distances to different classes are comparable during the prediction. To further improve the efficiency in the training phase, we also show how to simplify the method to learn a diagonal matrix for each class. This simplification can substantially improve the training efficiency especially when dealing with a large number of classes and enhance the scalability of our method.

Another contribution of this distance metric learning method is that it can correct the class imbalance problem, in which some classes contain much more training images than other classes. In such situation the original I2C distance has a bias towards these big classes like many other NN based methods. However, this proposed method is able to correct the bias by learning the Per-Class metrics since the discriminative information in each class is preserved by its associated metric.

4.2 Related Work

A good distance metric plays an important role in many real world problems. In recent years, a number of methods have been proposed for distance metric learning. We start reviewing these methods by first giving the formulation of the distance measure. Given two data points X_i and X_j , the square of their Mahalanobis distance is expressed as:

$$Dist(X_i, X_j) = (X_i - X_j)^T M (X_i - X_j), \quad (4.1)$$

where the symmetric matrix M is the Mahalanobis metric, which is valid only when it is positive semi-definite. When M is equal to the identity matrix I , this distance is reduced to Euclidean distance.

The goal of distance metric learning is focused on learning this Mahalanobis metric M . Previous studies on distance metric learning can be roughly divided into three categories according to the learning paradigm. The first category is unsupervised learning, which learns the distance metric without label information and is used for unsupervised learning task like clustering or dimension reduction. Representative methods include Principle Component Analysis (PCA), Multiple Dimension Scaling (MDS) [105], ISOMAP [106] and Locally Linear Embedding (LLE) [107]. The second category is supervised learning, which learns the distance metric from training data with class labels. By using these labels, most supervised distance metric learning methods try to keep data points in the same class similar while separate them far away if coming from different classes. Recent representative methods include Relevance Component Analysis (RCA) [108], Neighborhood Components Analysis (NCA) [109], Discriminative Component Analysis (DCA) [110], Large Margin Nearest Neighbor (LMNN) [18, 111], Local Distance Metric Learning (LDM) [112], Information-Theoretic Metric Learning (ITML) [113], Probabilistic Relevance Component Analysis (pRCA) [114], Semantics-Preserving Bag-of-Words (SPBoW) [115], etc. The third category is semi-supervised distance metric learning, which makes use of both labeled and unlabeled data for learning the distance metric. For the problem with only limited labeled data but plenty of unlabeled data, semi-supervised learning methods would be more useful. A recent example of semi-supervised distance metric learning is Hoi et al. [116] for the task of collaborative image retrieval.

Besides these three categories of distance metric learning, recently there is also another new category called transfer distance metric learning [117, 118], which not only utilizes labeled data from its own domain, but also uses the labeled data from other domains for learning the distance metric. Transfer distance metric learning would be suitable for the problem where different domains are relevant and the target domain contains less labeled data, but the source domains have plenty of labeled data to help learning the distance metric for the target domain.

When applying these distance metric learning methods for image classification, feature quantization is required to convert the local features into one vector for each image since these methods are only dealing with I2I distance. Such quantization may lose the discriminative information before the learning procedure and decrease the final classification performance. In this chapter, we propose to learn the distance metric for the I2C distance and avoid such quantization, which is expected to provide a more effective classifier for image classification.

4.3 Problem Formulation

The goal of learning the I2C distance metric is to learn a Mahalanobis distance between a set of local features in an image and their NN features in a candidate class. The dimension of learned matrix depends on the dimension of the local features. This is different from traditional distance metric learning methods that learn the distance between different images, where the dimension of learned matrix depends on the dimension of the only one vector that represents an image. Given an image X_i , the learned Mahalanobis I2C distance to a candidate class C_j is given as follows:

$$Dist(X_i, C_j) = \sum_{k=1}^{m_i} (f_{X_{ik}} - f_{X_{ik}}^{C_j})^T M_j (f_{X_{ik}} - f_{X_{ik}}^{C_j}), \quad (4.2)$$

where notations are consistent to Chapter 2 and Chapter 3. Compared to the original I2C distance, the only difference is that we change the Euclidean distance between local feature $f_{X_{ik}}$ and its NN feature $f_{X_{ik}}^{C_j}$ to a Mahalanobis distance with the matrix M_j learned for class C_j .

As there may be tens or hundreds of classes in many image classification problems, learning a global metric may not be sufficient enough to improve the I2C distance. Therefore, our method learns a different Mahalanobis matrix for every class. Such class specific metric can better preserve the discriminative information for different classes during the learning procedure, which is able to provide better performance than learning only a global metric as shown in [18]. Another contribution of the Per-Class metric learning is its ability to correct the class imbalance problems, where some classes contain more training images than other classes. Like many NN based methods, the original NBNN classifier has a bias towards these big classes. After learning a distance metric for each class, the discriminative information in each class is preserved by its associated metric, which is able to correct such bias. However, since the prediction for a test image requires comparing the learned I2C distances to different classes, all the metrics must be learned simultaneously to ensure the global consistency for prediction.

We adopt the idea of large margin again to learn these metrics. LMNN [18] also adopts this idea to keep the distance between differently labeled images far away from the distance between similar images with a large margin. In the case of the I2C distance, we try to keep the distance from the image to its belonging class less than the distance to any other class with a large margin, so that the distance to the belonging class can be the shortest among all classes. Specifically, for each training image X_i , a triplet constraint is formulated by selecting its belonging class C_p (named as positive class) and any other class C_n (named as negative class). We constrain in each triplet that the I2C distance to positive class C_p should be less than that to negative class C_n with a large margin, and add a nonnegative slack variable ξ for each triplet constraint. The constraint triplet is similar to that in our previous work of learning the weighted I2C distance in Chapter 3:

$$\begin{aligned} \text{Dist}(X_i, C_n) - \text{Dist}(X_i, C_p) &\geq 1 - \xi_{ipn}, \\ \text{s.t.} \quad \xi_{ipn} &\geq 0. \end{aligned} \tag{4.3}$$

The objective function in our optimization problem is composed of two terms: the regularization term and error term. This is analogous to the optimization problem of SVM. We minimize all the slack variables in the error term. For the regularization term, we use Frobenius norm of each metric to penalize its complexity. This regularizer is able

to prevent any elements within each matrix from being too large as shown in [118, 119]. The whole optimization problem is therefore formed as:

$$\begin{aligned}
\min_{M_1, M_2, \dots, M_C} O(M_1, M_2, \dots, M_{N_C}) = & \\
(1 - \lambda) \sum_{j=1}^{N_C} \|M_j\|_F^2 + \lambda \sum_{i,p,n} \xi_{ipn}, & \\
s.t. \forall i, p, n : Dist(X_i, C_n) - Dist(X_i, C_p) \geq 1 - \xi_{ipn}, & \\
\forall i, p, n : \xi_{ipn} \geq 0, & \\
\forall j : M_j \succeq 0. & \quad (4.4)
\end{aligned}$$

Here, the variable λ controls the trade-off between regularization term and error term and is usually empirically determined by cross-validation. The last constraint $M_j \succeq 0$ is to make sure that every learned matrix is positive semi-definite. This optimization problem is an instance of SDP, which can be solved using standard SDP solvers. However, the standard SDP solvers are usually computation expensive and less efficient. Since our optimization problem is convex, we use an efficient subgradient descent based method derived from [18] to solve this problem.

It should be noted that though our I2C distance metric learning method adopts the idea of large margin similar to that in LMNN, there are three major differences between these two works: 1) LMNN uses positive and negative instances from the k nearest neighbors of each training image to form the triplet, while in our method each triplet tries to keep the I2C distance to the belonging class less than to other classes by a large margin. So our triplet is generated at class level instead of image level for each training image, which reduces the number of constraints significantly. 2) LMNN uses Image-To-Image distance while we use Image-To-Class distance. Besides the advantages of I2C distance as described in [14], this distance with the Per-Class metrics can make the Mahalanobis I2C distance from the test image to its related class more discriminative, since the metric of the related class is able to learn the discriminative information for this class during the training phase. 3). The representation for each image is different. In LMNN each image must be represented by one feature vector, so local features must be quantized before using LMNN. However, our method directly applies to the origin representation and avoids feature quantization.

4.4 An Efficient Subgradient Descent Solver

To solve the optimization problem in Equation (4.4), we first rewrite the Mahalanobis I2C distance in Equation (4.2) by introducing a new term ΔX_{ij} , which is a $m_i \times d$ matrix representing the difference between all features in the image X_i and their nearest neighbors in the class C_j formed as:

$$\Delta X_{ij} = \begin{pmatrix} (f_{X_i1} - f_{X_{j1}}^{C_j})^T \\ (f_{X_i2} - f_{X_{j2}}^{C_j})^T \\ \dots \\ (f_{X_im_i} - f_{X_{jm_i}}^{C_j})^T \end{pmatrix}. \quad (4.5)$$

The learned I2C distance from image X_i to class C_j can then be reformulated as:

$$Dist(X_i, C_j) = Tr(\Delta X_{ij} M_j \Delta X_{ij}^T) = Tr(M_j \Delta X_{ij}^T \Delta X_{ij}), \quad (4.6)$$

and the constraint in optimization problem of Equation (4.4) is also reformulated as:

$$Tr(M_n \Delta X_{in}^T \Delta X_{in}) - Tr(M_p \Delta X_{ip}^T \Delta X_{ip}) \geq 1 - \xi_{ipn}. \quad (4.7)$$

The advantage of using this distance form for the Mahalanobis I2C distance is that the term $\Delta X_{ij}^T \Delta X_{ij}$ is a $d \times d$ matrix, which does not depend on the number of local features m_i in the image X_i and can be pre-calculated before the optimization problem.

We use a subgradient descent based solver to update each matrix since our optimization problem is convex w.r.t. all the matrices. At each step, every matrix is updated by taking a small step along the negative subgradient direction to reduce the objective function and then projected onto a feasible set to ensure the positive semi-definiteness. To calculate the subgradient of the objective function over each matrix, we need to incorporate each triplet constraint into the objective function. As the slack variable ξ is nonnegative, its value should be 0 if the corresponding triplet constraint is satisfied, and a positive value if the triplet constraint is unsatisfied. This can be expressed as:

$$\begin{aligned} \forall i, p, n, Tr(M_n \Delta X_{in}^T \Delta X_{in}) - Tr(M_p \Delta X_{ip}^T \Delta X_{ip}) \geq 1 &\Rightarrow \xi_{ipn} = 0, \\ Tr(M_n \Delta X_{in}^T \Delta X_{in}) - Tr(M_p \Delta X_{ip}^T \Delta X_{ip}) < 1 &\Rightarrow \xi_{ipn} > 0. \end{aligned} \quad (4.8)$$

Therefore, all the zero-value slack variables can be removed from the optimization problem and these positive values can be formulated as:

$$\xi_{ipn} = 1 - \text{Tr}(M_n \Delta X_{in}^T \Delta X_{in}) + \text{Tr}(M_p \Delta X_{ip}^T \Delta X_{ip}). \quad (4.9)$$

At each iteration t , we can first scan over all triplets to find a set of unsatisfied triplets. We denote N^t as the set of triplet indices such that $(i, p, n) \in N^t$ if $\xi_{ipn} > 0$. Then the objective function at t^{th} iteration can be reformulated by replacing those positive slack variables according to Equation (4.9):

$$\begin{aligned} O(M_1, M_2, \dots, M_{N_C}) &= \\ (1 - \lambda) \sum_{j=1}^{N_C} \|M_j\|_F^2 &+ \lambda \sum_{(i,p,n) \in N^t} 1 - \text{Tr}(M_n \Delta X_{in}^T \Delta X_{in}) + \text{Tr}(M_p \Delta X_{ip}^T \Delta X_{ip}). \end{aligned} \quad (4.10)$$

For each matrix M_j^t at t^{th} iteration, its subgradient $G(M_j^t)$ can be calculated by taking the derivative of the above objective function w.r.t. M_j^t :

$$G(M_j^t) = 2(1 - \lambda)M_j^{t-1} + \lambda \sum_{(i,p,n) \in N^t, j=p} \Delta X_{ij}^T \Delta X_{ij} - \lambda \sum_{(i,p,n) \in N^t, j=n} \Delta X_{ij}^T \Delta X_{ij}. \quad (4.11)$$

Directly calculating the subgradient at each iteration using this formula would be computation expensive. As the changes in the subgradient from one iteration to the next are only determined by the differences between the sets N^t and N^{t+1} , we use $G(M_j^t)$ to calculate the subgradient $G(M_j^{t+1})$ for the next iteration, which is more efficient:

$$\begin{aligned} G(M_j^{t+1}) &= G(M_j^t) + 2(1 - \lambda)(M_j^t - M_j^{t-1}) \\ &+ \lambda \left(\sum_{(i,p,n) \in (N^{t+1} - N^t), j=p} \Delta X_{ij}^T \Delta X_{ij} - \sum_{(i,p,n) \in (N^{t+1} - N^t), j=n} \Delta X_{ij}^T \Delta X_{ij} \right) \\ &- \lambda \left(\sum_{(i,p,n) \in (N^t - N^{t+1}), j=p} \Delta X_{ij}^T \Delta X_{ij} - \sum_{(i,p,n) \in (N^t - N^{t+1}), j=n} \Delta X_{ij}^T \Delta X_{ij} \right). \end{aligned} \quad (4.12)$$

Here, $(N^{t+1} - N^t)$ denotes the set of triplets that do not exist at t^{th} iteration but are added at $(t+1)^{\text{th}}$ iteration, and $(N^t - N^{t+1})$ denotes the set of triplets that exist at t^{th} iteration but are removed at $(t+1)^{\text{th}}$ iteration.

Since the term $(\Delta X_{ij}^T \Delta X_{ij})$ is pre-calculated before the first iteration, it is treated as a constant during the iterations. This term occupies the main memory cost in this

algorithm, because it is a $d \times d$ matrix and is required between every training image X_i and every class C_j . Therefore, the total memory complexity is $O(N_C \cdot N_T \cdot d^2)$ for a dataset with N_C classes and N_T training images per class. The matrix is updated by taking a small step along the negative subgradient direction for each iteration. To enforce the positive semi-definiteness, the updated matrix needs to be projected onto a feasible set. This projection is done by eigen-decomposition of the matrix and truncating all the negative eigenvalues to zeros. As the optimization problem is convex, this solver is guaranteed to converge to the global optimum. During the classification process, the learned Mahalanobis I2C distances to all classes can be directly compared for prediction since all the metrics are updated simultaneously in the learning procedure, therefore guarantee the consistency for prediction. We summarize the whole work flow in Algorithm 1.

Algorithm 1 A Subgradient Descent Method for Solving Our Optimization Problem

Input: step size α , parameter λ and pre-calculated data $(\Delta X_{ij}^T \Delta X_{ij}), i \in \{1, \dots, N_X\}, j \in \{1, \dots, N_C\}$
for $j := 1$ to N_C **do**
 $M_j^0 := I$
 $G(M_j^0) := 2(1 - \lambda)M_j^0$
end for{Initialize M and subgradient for each class}
 Set $t := 0$
repeat
 Compute N^t by checking each error term ξ_{ipn}
 for $j = 1$ to N_C **do**
 Update $G(M_j^{t+1})$ using Equation (4.12)
 $M_j^{t+1} := M_j^t + \alpha G(M_j^{t+1})$
 Project M_j^{t+1} for keeping positive semi-definite
 end for
 Calculate new objective function
 $t := t + 1$
until Objective function converged
 Output: each matrix $M_1, \dots, M_{N_C}$

4.5 Learning Diagonal Matrices for Acceleration

To further improve the efficiency during the training phase, we can learn a diagonal matrix for each class instead of a full matrix. This simplification can substantially improve the training efficiency especially when dealing with a large number of classes and enhance the scalability of our method. The method for learning diagonal matrix is simply derived from our method for learning the full matrix in the previous subsection. Recall that the Mahalanobis I2C distance from image X_i to a candidate class C_j is:

$$Dist(X_i, C_j) = Tr(M_j \Delta X_{ij}^T \Delta X_{ij}). \quad (4.13)$$

Since M_j is a diagonal matrix here, only diagonal elements of $\Delta X_{ij}^T \Delta X_{ij}$ are used. So this I2C distance can be simplified as:

$$Dist(X_i, C_j) = Tr(M_j Diag(\Delta X_{ij}^T \Delta X_{ij})) = Tr(M_j D(X_{ij})). \quad (4.14)$$

Here, $Diag(\Delta X_{ij}^T \Delta X_{ij})$ indicates the diagonal matrix which only preserves the diagonal elements of $\Delta X_{ij}^T \Delta X_{ij}$ and is denoted as $D(X_{ij})$ for simplicity. We use this new notation of the I2C distance in the optimization problem and rewrite Equation (4.4) as:

$$\begin{aligned} \min_{M_1, M_2, \dots, M_{N_C}} O(M_1, M_2, \dots, M_{N_C}) &= (1 - \lambda) \sum_{j=1}^{N_C} \|M_j\|_F^2 + \lambda \sum_{i, p \rightarrow i, n \rightarrow i} \xi_{ipn}, \\ s.t. \forall i, p, n : Tr(M_n D(X_{in})) - Tr(M_p D(X_{ip})) &\geq 1 - \xi_{ipn}, \\ \forall i, p, n : \xi_{ipn} &\geq 0, \\ \forall j : M_j &\succeq 0. \end{aligned} \quad (4.15)$$

These diagonal matrices can be solved by the subgradient descent solver introduced previously. The new equation of subgradient $G(M_j^t)$ is formulated simply by replacing $\Delta X_{ij}^T \Delta X_{ij}$ in Equation (4.11) to its diagonal matrix $D(X_{ij})$:

$$G(M_j^t) = 2(1 - \lambda) M_j^{t-1} + \lambda \sum_{(i, p, n) \in N^t, j=p} D(X_{ij}) - \lambda \sum_{(i, p, n) \in N^t, j=n} D(X_{ij}). \quad (4.16)$$

Equation (4.12) is re-formulated in the same way to calculate $G(M_j^{t+1})$ from $G(M_j^t)$ at each iteration:

$$\begin{aligned}
G(M_j^{t+1}) &= G(M_j^t) + 2(1 - \lambda)(M_j^t - M_j^{t-1}) \\
&+ \lambda \left(\sum_{(i,p,n) \in (N^{t+1}-N^t), j=p} D(X_{ij}) - \sum_{(i,p,n) \in (N^{t+1}-N^t), j=n} D(X_{ij}) \right) \\
&- \lambda \left(\sum_{(i,p,n) \in (N^t-N^{t+1}), j=p} D(X_{ij}) - \sum_{(i,p,n) \in (N^t-N^{t+1}), j=n} D(X_{ij}) \right). \tag{4.17}
\end{aligned}$$

The work flow of learning diagonal matrices is similar to Algorithm 1. Note that eigen-decomposition is not required here after updating a new M_j^t since it is a diagonal matrix. The positive semi-definiteness of every updated matrix can be preserved by simply truncating all the negative elements to zeros. This simplification also improves the efficiency of our method. Moreover, learning diagonal matrices also reduces the memory cost from $O(N_C \cdot N_T \cdot d^2)$ to $O(N_C \cdot N_T \cdot d)$ for a dataset with N_C classes and N_T training images per class, since $\Delta X_{ij}^T \Delta X_{ij}$ is replaced by $D(X_{ij})$. Such reduction makes it possible to load all triplets into the memory even when the training set is very large.

4.6 Experiments

We evaluate our proposed method in the same five single-label datasets as in Chapter 3: Scene-15, Sports, Corel, Caltech 101 and 256 datasets. The training and test set splitting way and the local feature extraction method is also the same as in Chapter 3. We denote our method as I2CDML (Image-To-Class Distance Metric Learning), where the value of λ is determined via a separate validation set in each dataset and ranging from 0.85 to 0.95, and the learning rate α is set to 0.005. The updating procedure stops when the objective function converges within 0.05 between two consecutive iterations or meets the maximum number of iterations, which is set to 100 in the experiment.

4.6.1 Experiments in Scene-15, Sports and Corel Datasets

We first show the experiment results in three small datasets: Scene-15, Sports and Corel. We compare our learned Mahalanobis I2C distance to the original Euclidean I2C distance (NBNN [14]), Mahalanobis I2I distance and Euclidean I2I distance to evaluate the

Table 4.1: Comparing our learned Mahalanobis I2C distance to Euclidean I2C distance, Mahalanobis I2I distance and Euclidean I2I distance in Scene-15, Sports and Corel datasets by recognition accuracy (%). Here, M and E stand for Mahalanobis and Euclidean distance, respectively.

Method	Scene-15	Sports	Corel
I2C-M (I2CDML)	77.0 $\pm$ 0.60	78.5 $\pm$ 1.63	88.8 $\pm$ 0.93
I2C-E (NBNN)	72.3 $\pm$ 0.93	67.6 $\pm$ 1.10	85.7 $\pm$ 1.20
I2I-M (LMNN)	66.8 $\pm$ 0.62	74.8 $\pm$ 0.89	81.2 $\pm$ 1.39
I2I-M (ITML)	68.9 $\pm$ 0.44	72.9 $\pm$ 0.87	80.8 $\pm$ 1.44
I2I-M (DCA)	61.4 $\pm$ 0.65	62.9 $\pm$ 2.61	75.4 $\pm$ 1.40
I2I-M (RCA)	52.8 $\pm$ 0.38	66.9 $\pm$ 2.35	69.0 $\pm$ 1.29
I2I-M (NCA)	57.5 $\pm$ 0.62	69.5 $\pm$ 2.23	76.0 $\pm$ 1.14
I2I-M (LDA)	51.3 $\pm$ 1.09	64.0 $\pm$ 2.23	69.8 $\pm$ 0.72
I2I-E	49.8 $\pm$ 1.09	63.5 $\pm$ 2.48	66.4 $\pm$ 0.68

effectiveness of our method. For the I2I distance, we need to first convert the local features into a fixed-length vector for representing each image. In this experiment, we use the BoW model for such feature quantization due to its effectiveness in recent research. The codebook size is set to 400 in every dataset generated by K-means clustering, so each image is represented as a 400-dimension vector and each component in the vector represents the frequency of the corresponding visual word. For the prediction of I2I distance, 3-NN strategy is used as in [18]. We also tried other values for the number of NN in prediction but did not find much difference in the classification result. For the Mahalanobis I2I distance we implemented several distance metric learning methods including: LMNN [18], ITML [113], DCA [110], RCA [108], NCA [109] and LDA. Table 4.1 summarizes the results in these three datasets. We can see that our Mahalanobis I2C distance significantly outperforms the all other distances in every dataset, which illustrates the effectiveness of our learning method. We also find that even using Euclidean distance, NBNN can still beat most Mahalanobis I2I distances, while the Euclidean I2I distance performs the worst in all three datasets. This comparison shows that the I2C distance is more discriminative than the traditional I2I distance due to the avoidance of feature quantization, which may lose the discriminative information that would not be compensated even after learning the distance metric.

We also compare the computation cost of our learning procedure to LMNN since both methods are learned in a large margin optimization framework. Table 4.2 lists the

Table 4.2: Comparing the computation time and the number of triplets of our learning method to LMNN in Scene-15, Sports and Corel datasets (seconds).

Method	Scene-15		Sports		Corel	
	time	# triplets	time	# triplets	time	# triplets
I2C-M (I2CDML)	3448s	21000	537s	3920	758s	4500
I2I-M (LMNN)	17992s	950000	2659s	137200	1776s	112500

Table 4.3: Comparing learning diagonal matrices to full matrices in Scene-15, Sports and Corel datasets by recognition accuracy (%), memory space and computation time.

Method	Scene-15		Sports		Corel	
	Diagonal	Full	Diagonal	Full	Diagonal	Full
I2CDML	76.1	77.0	77.4	78.5	87.2	88.8
Memory Cost	21MB	2260MB	4MB	460MB	5MB	512MB
Computation Time	44s	3448s	25s	537s	27s	758s

running time of learning procedure required for each dataset and the number of triplets used in the two methods. As our method formulates the triplet constraints by selecting one positive class and one negative class at each time, and each training image has one positive class and $(N_C - 1)$ negative classes, there are totally $(N_C - 1) \cdot N_C \cdot N_T$ triplets for a dataset with N_C classes and N_T training images per class. However, in LMNN, the triplet for each training image is constructed by selecting a positive image in its kNN and a negative image. Since the number of images is usually much more than the number of classes, the triplets required for LMNN are much more than those for our method as shown in Table 4.2. Therefore, our method needs much less computation cost during the learning phase, while the classification performance of our method is significantly better than LMNN due to the effectiveness of the I2C distance.

To alleviate the computation cost, we simplify our method to learn only a diagonal matrix for each class and compare it to the original I2CML which learns full matrices as shown in Table 4.3. The space complexity is reduced from $O(N_C \cdot N_T \cdot d^2)$ to $O(N_C \cdot N_T \cdot d)$ for a dataset with N_C classes and N_T training images. Therefore, the memory space and computation time are significantly reduced for learning diagonal matrices, while it achieves a comparable classification performance.

Table 4.4: Comparison of our learned Mahalanobis I2C distance to Euclidean I2C distance, Mahalanobis I2I distance and Euclidean I2I distance in Caltech 101 by recognition accuracy (%). Here M and E stand for Mahalanobis and Euclidean distance respectively.

Method	Caltech 101
I2C-M (I2CDML 4×4)	58.1 ± 0.83
I2C-E (NBNN 4×4)	47.8 ± 1.35
I2C-E (NBNN 1×1)	37.4 ± 1.22
I2I-M (LMNN)	27.3 ± 1.14
I2I-M (ITML)	26.5 ± 1.22
I2I-M (DCA)	23.3 ± 1.41
I2I-M (RCA)	20.4 ± 1.13
I2I-M (NCA)	20.4 ± 1.11
I2I-M (LDA)	20.7 ± 1.34
I2I-E	20.0 ± 1.52

4.6.2 Experiment in Caltech 101 and 256 Datasets

Caltech 101 and 256 datasets contain more classes than the previous three datasets. We evaluate our method in these two datasets to demonstrate its scalability in large-scale datasets. Again we compare our method with Euclidean I2C distance (NBNN [14]), Mahalanobis I2I distance (LMNN [18], ITML [113], DCA [110], RCA [108], NCA [109] and LDA) and Euclidean I2I distance. For the I2I distance we use the same implementation in the previous three datasets. We also use a single spatial level of 4×4 subregions to further reduce the size of feature set and improve the performance of I2C distances (See Section 3.4 for more detail of this search space reduction).

Table 4.4 summarizes the results in Caltech 101. Three conclusions can be drawn from this table: First, due to the increased number of classes, both I2I and I2C Euclidean distances do not perform well in this dataset. Second, after learning the Per-Class metric, our method improves the performance of the I2C distance for 10.3%, which is better than LMNN and ITML over the original Euclidean I2I distance, while other distance metric learning algorithms improve little over the I2I Euclidean distance. This may be because our method learns the distance metric for every class rather than learning one global distance metric. Third, this table again shows the I2C distance is much better than I2I distance, although some improvement should be contributed to the use of spatial division. A fair comparison is also available by checking the performance of Euclidean I2C distance

Table 4.5: Comparing learning diagonal matrices to full matrices in Caltech 101 by recognition accuracy (%), memory space and computation time.

Method	Diagonal	Full
I2CDML	56.0	58.1
Memory Cost	156MB	20GB
Computation Time	271s	8.3h

Table 4.6: Results in Caltech 256 by recognition accuracy (%).

Method	Caltech 256
I2C-M (I2CDML 4×4)	25.5 ± 0.92
I2C-E (NBNN 4×4)	20.0 ± 1.02
I2C-E (NBNN 1×1)	18.5 ± 0.95
I2I-M (LMNN)	8.4 ± 0.42
I2I-M (ITML)	6.3 ± 0.32
I2I-M (DCA)	5.6 ± 0.51
I2I-M (RCA)	6.9 ± 0.78
I2I-M (NCA)	7.1 ± 1.24
I2I-M (LDA)	7.3 ± 1.01
I2I-E	7.0 ± 0.35

without using spatial division (37.4%), which is still better than all Mahalanobis I2I distances.

Due to the large number of classes in Caltech 101, efficiency is also an important factor for distance learning methods. Our method requires learning 101 metrics in this dataset, which makes it more time consuming compared to the previous three datasets. To alleviate the computation cost, we simplify our method to learn only a diagonal matrix for each class and compare it to learning full matrices as shown in Table 4.5. Only 156MB space is required to be loaded into the memory for learning diagonal matrices, while learning full matrices requires about 20GB space, which makes it difficult to load all data into the memory at the same time. The computation cost of learning full matrices is also much more than that in learning the diagonal matrices. Meanwhile, learning diagonal matrices achieves a comparable classification performance to learning full matrices. Therefore, learning a diagonal matrix instead of full matrices is especially useful for large-scale image classification problem where efficiency is more important.

I2CDML		NBNN	
Rank	Class	Rank	Class
1	skateboard	1	cockroach
2	ak47	2	blimp
3	cockroach	3	fighter-jet
4	elephant	4	dog
5	bulldozer	5	skateboard
6	picnic-table	6	bulldozer
7	ostrich	7	helicopter
8	blimp	8	ostrich
9	dog	9	picnic-table
10	fighter-jet	10	sword
11	golf-ball	11	ak47
.	.	.	.
.	.	.	.
.	.	.	.

Figure 4.2: The label ranks generated by both our method and NBNN for a test image of ak47 as example. Only the top 11 class labels are shown.

Caltech 256 contains more classes than Caltech 101 dataset. Even using 15 images per class for training, the memory cost of learning full matrices is nearly $128GB$, which is too large to be loaded into the memory. Therefore, we only learn diagonal matrices in this dataset, which needs only about $1GB$ memory space and can be loaded into the memory during the learning procedure. The computation time of the learning procedure is less than 1 hour, which is nearly negligible compared to the computation cost of the feature extraction procedure and NN search. Such small overload is able to improve the classification compared to the original Euclidean I2C distance. We show the results of our method and other methods as compared in the previous four datasets in Table 4.6. Similar conclusions from the results in Caltech 101 can be drawn again in this dataset. Due to the large number of classes, I2I distance metric learning methods that learn only one global metric gains little improvement over the Euclidean I2I distance, while our method can still improve the I2C distance because of its Per-Class metric learning. The performance gap between I2C distance and the traditional I2I distance becomes more prominent in this dataset as it contains larger number of classes than the previous four datasets.

In all the previous experiments we use recognition accuracy to evaluate the performance of each method. However, for Caltech 256 dataset with such a large number of classes, this evaluation method cannot fully reflect the improvement of our method compared to NBNN. The I2C distance to all classes can produce a label rank, while only the top class is selected for prediction and evaluated by the recognition accuracy. It may be possible the position of the ground truth class is improved in the label rank but still does not reach the top position, which can not be reflected using the previous evaluation method. Taking the test image in Figure 4.2 as example. Both our method and NBNN can produce the label ranks by sorting the I2C distances to all classes. The ground truth class for this test image is ranked at 11th position by NBNN, while our method improves it to the 2nd position in the rank. However, such improvement is not reflected by the previous evaluation method since both algorithms predict incorrectly. Therefore in order to provide more detailed improvement, we also compare the position of the ground truth class in the label rank generated by our method to NBNN. On average, the label rank position of the ground truth class is improved by 8.77 compared to NBNN. There are 51.3% test images from the whole test set that our method improves their positions of the ground truth class compared to NBNN, while another 21.7% are the same as that in NBNN. This is a much larger improvement since it covers more than half of the test set, while using previous evaluation method we only find small improvement of the recognition accuracy since only 5.5% more test images are improved to the top position in the label rank by our method. Therefore this new evaluation method is able to provide more detailed improvement and validate the effectiveness of our distance learning method.

4.6.3 Class Imbalance Experiment

In this experiment we show that the proposed method can solve the class imbalance problem. We use Caltech 101 dataset for this experiment since the number of training images per class varies largely in this dataset, ranging from 31 to 800. 8 groups with different training sizes are formed for evaluating the performance of our method compared to NBNN. The test sets for all groups are kept the same by selecting 15 images per class. For the training set, we first select 15 different images per class as a balanced group in Group 1. Then we increase the maximum number of training images per class by 15 for

Table 4.7: The maximum, mean and variance of the number of training images per class in each group for class imbalance experiment in Caltech 101 dataset

	Group1	Group2	Group3	Group4	Group5	Group6	Group7	Group8
Max	15	30	45	60	75	90	105	120
Mean	15.0	28.3	37.4	42.8	46.1	47.9	49.2	50.2
Variance	0	13.2	89.3	217.4	363.0	487.7	615.3	741.3

Figure 4.3: The recognition accuracy of our method compared to NBNN in 8 groups with different imbalanced levels.

each group. Since some classes have limited images in this dataset and their training sizes cannot be increased, while other classes have plenty images in the dataset and their training sizes can be increased, the size of training set in each class will be different and result in an imbalanced group. With the increased maximum number of training images per class, the variance of the training set per class is also increased and leads the group to be more imbalanced, as shown in Table 4.7.

We compare our distance metric learning method with NBNN in each group. We use the diagonal metric learning in our method while other settings are kept the same as the previous experiment. The recognition accuracies of our method and NBNN are shown in Figure 4.3. We can see that the performance of NBNN is firstly increased in Group 2 but then decreased in other groups. The reason for the improvement from Group 1 to Group 2 is due to the largely increased training set with less variance. However, though

the training set is continually increased, the increase of variance and imbalance is more prominent from Group 3 to Group 8 and causes the performance of NBNN to decrease. Therefore we can see that NBNN cannot work well for imbalanced classes. However, with the increased imbalance our method can still keep the same recognition accuracy, which validates that the learned Per-Class metrics are able to correct the class imbalance problem.

4.7 Summary

In this chapter, we proposed a distance metric learning method to learn a Mahalanobis I2C distance. Instead of learning one global distance metric, we learned the class specific Mahalanobis metric for each class, which makes the learned Mahalanobis I2C distance more discriminative especially when there are a large number of classes. A large margin framework has been formulated to simultaneously learn the Per-Class Mahalanobis distance metrics, with the constraint that the I2C distance for each training image to the belonging class should be less than the distance to any other class with a large margin. We proposed a subgradient descent method to efficiently solve this optimization problem. For large scale classification problems, a simplified diagonal version of mahalanobis I2C distance is also proposed. By only learning class specific diagonal matrix for every class, the computation cost and memory usage can be significantly reduced while the classification performance is preserved. The experiment results in five image datasets of Scene-15, Sports, Corel, Caltech 101 and 256 verified that our I2CDML method can significantly outperform the original Euclidean I2C distance as well as other distance metric learning methods that learn a Mahalanobis I2I distance. We also show that the learned Per-Class metrics are able to correct the class imbalance problem.

However, we notice that the performance of this distance metric learning method is slightly lower than the weight learning method proposed in Chapter 3. This may be because the number of variables to be learned in a full or diagonal matrix is much less than the number of weights, which are equal to the number of local features in the training set extracted by dense sampling. With more variables to be learned, the triplet constraints in Chapter 3 are more likely to be satisfied and therefore providing

better performance. Nevertheless, both methods try to improve the performance of I2C distance from different aspects and therefore can be combined together to generate a better performance.

Chapter 5

Learning Class-to-Image Distance for Multi-label Problems

5.1 Introduction

In Chapter 3, we have discussed learning the weighted I2C distance for the multi-label problem. This method learns the weight to distinguish the different importance of local features in the class. However, in the multi-label problem, a test image may also contain several regions belonging to different classes. The resulting feature set for this test image will contain local features belonging to different classes, while all local features have to be counted in the I2C distance calculation, since it is unknown which local feature belongs to which class before the I2C distance calculation. Consider an example test image in Figure 5.1 (a), which mainly contains two regions: “car” and “street”. When measuring the distance to each class, local features in every region of the test image are counted in the final I2C distance. In this example, local features located at “street” region are also counted in the I2C distance to class “car”, while distances between these features and their NNs in the class “car” (marked as black arrows) are useless and will make the I2C distance to class “car” inaccurate. Such problem also exists when measuring the distance between this image and class “street”. Since it is even more difficult to first detect the region of “car” and “street” in this test image before the I2C distance calculation, the I2C distance may not be suitable for the multi-label problem.

In this chapter, we propose to use the Class-to-Image (C2I) distance to measure the distance between image and class for the multi-label problem. We could assume all local

Figure 5.1: The illustration of I2C distance (a) and C2I distance (b). Red and black rectangles represent the local features located at the region of “car” and “street” respectively. The NN search directions are represented by arrows from one side to the other side. Red and black arrows represent the NN match between same region and different regions respectively. Though region “street” may not be useful in the “car” class, the NN match from region “street” to “car” in the I2C distance (a) cannot be avoided, while in the C2I distance (b) it can be avoided.

features in each class are relevant to this class, either by manually removing irrelevant local features or using some feature selection technologies. Therefore, we define the C2I distance as an inverse version of the I2C distance: the Euclidean distance between each local feature in the class feature set and its NN in the feature set of the test image is summed up to form its C2I distance. Such an inversion can solve the problem that I2C distance encounters in the multi-label problem. We explain that by taking the same test image as example. Figure 5.1 (b) shows the formation of the C2I distance from class “car” to the test image. The feature set of class “car” is constructed during the training phase and we should assume most of its elements are located at the regions of “car”. If a test image contains a car as in this example, the local features in class “car” will have better opportunities to find their NNs located at the regions of car in this test image and thus a smaller C2I distance than other images without a car. Irrelevant local features in the region of other objects such as “street” in this test image is less likely to be matched as NN features from class “car” and therefore less likely to influence the accuracy of the C2I distance compared to the I2C distance.

There are two problems in the C2I distance. First, the success of the C2I distance in the previous examples relies on the assumption that most of the local features in the feature set of class “car” are located at the regions of “car”. Since the training images also contain multiple regions, we do not know which local features belong to class “car” and which do not. Manually labeling every region to its class label can solve this problem but this is very expensive and most datasets only provide a list of class labels for each training image without pixel-wise or bounding box label information. Second, the number of local features in a class is usually much more than that in an image, resulting in a time-consuming C2I distance calculation when compared to the I2C distance.

To solve these problems, in this chapter, we propose an optimization algorithm using large margin constraint and L_1 -norm regularization to learn the C2I distance. The feature set of each class is first constructed by gathering all local features in the training images belonging to this class, same as that in I2C distance. Then we associate a weight to every local feature to distinguish the different importance of each local feature, and formulate an optimization problem to learn these weights. We try to use the label information to select relevant local features based on their weights and reduce the size of feature set by

L_1 -norm regularization. Therefore, the objective function of our optimization problem is comprised of two parts: the error term and the regularization term. In the error term, label information is utilized by constraining in each triplet that the C2I distance to the relevant image should be smaller than that to irrelevant image with a large margin. In the regularization term, we use L_1 -norm on the weight vector to generate a sparse solution as in [120, 121]. To solve this optimization problem, we also propose an efficient gradient descent based solver, which can update the weights and converge to the global optimum quickly. After the learning procedure, a significant large amount of the weights get zero values due to the L_1 -norm regularization and those local features with zero weight can be removed from the feature set, while the remaining local features with non-zero weight are preserved to make up the feature set. This reduction not only speeds up the C2I distance calculation, but also decreases the memory usage. Using this optimization algorithm, we are able to get better recognition performance with the learned C2I distance even though the size of the resulting feature set is much smaller.

We show in Figure 5.2 the example of our learned C2I distance from the training feature set of the class “car” for consistent description, to some test images containing the car. Local features are densely sampled from training images having a label of class “car”. After the learning algorithm of our optimization framework, only those local features with non-zero weight (Marked with red rectangles in the Figure 5.2) are preserved to make up the final feature set. The local features of test image are also densely sampled to form the image feature set. To evaluate the C2I distance from the class “car” to each test image, every local feature in class “car” needs to find its NN in the feature set of each test image (Those local features matched as NN features in the test images are also marked with red rectangles in Figure 5.2, and we can see that most of them are correctly located at the region of car). The sum of Euclidean distance from every local feature in class “car” to its NN feature in a test image multiplied by the learned weight is defined as the C2I distance to this image.

As most of the NN features in the test image of Figure 5.2 are correctly located at the region of car, we find the location of these NN feature is able to predict the location of the object class. Therefore, we also extend the C2I distance to the task of object localization problem by utilizing both the location of the NN matched feature and the

Figure 5.2: The example of the C2I distance from training feature set of class "car" to some test images containing a car object.

Feature-to-Set distance in the calculation of the C2I distance. Our learned C2I distance is able to not only predict whether a candidate class appears in a given test image, but also where the object class is located at the image.

5.2 Problem Formulation and Solver

We briefly describe the notations before describing the objective function of our optimization problem. Notations used in this chapter are consistent with that in previous chapters. For a class C_i , all the local features extracted from the training images belonging to this class are gathered together to make up the initial feature set of this class, and are denoted as $F_{C_i} = \{f_{C_i1}, f_{C_i2}, \dots, f_{C_im_i}\}$, where m_i represents the total number of local features in C_i and each local feature is denoted as $f_{C_ik} \in R^d, \forall k \in \{1, \dots, m_i\}$. Similarly, the feature set of an image X_j is denoted as $F_{X_j} = \{f_{X_j1}, f_{X_j2}, \dots, f_{X_jm_j}\}$, where m_j represents the total number of local features in image X_j . To calculate the C2I distance from class C_i to image X_j , every local feature f_{C_ik} in F_{C_i} needs to find its NN feature in F_{X_j} , which we denote as $f_{C_ik}^{X_j}$. The sum of Euclidean distance between every local feature f_{C_ik} in F_{C_i} and its NN feature $f_{C_ik}^{X_j}$ is defined as the C2I distance from class C_i to image X_j and is denoted as $Dist(C_i, X_j)$, which is similar to what was described in Chapter 2:

$$Dist(C_i, X_j) = \sum_{k=1}^{m_i} \|f_{C_ik} - f_{C_ik}^{X_j}\|^2, \quad (5.1)$$

where

$$f_{C_ik}^{X_j} = \arg \min_{k'=\{1, \dots, m_j\}} \|f_{C_ik} - f_{X_jk'}\|^2. \quad (5.2)$$

We associate a weight W_{C_ik} to every local feature f_{C_ik} in class C_i to distinguish the different importance of each local feature and learn a weighted C2I distance. All the weights can be put into a long vector W and a distance vector $D(C_i, X_j)$ is constructed as well with the same dimension to W , so the weighted C2I distance is formulated as:

$$W^T \cdot D(C_i, X_j) = \sum_{k=1}^{m_i} W_{C_ik} \cdot \|f_{C_ik} - f_{C_ik}^{X_j}\|^2. \quad (5.3)$$

The objective function of our optimization problem is composed of two terms: the regularization term and the error term. In the error term, we form the triplet constraints

by selecting one class with one relevant image and one irrelevant image for each triplet and constrain that the C2I distance from class C_i to the relevant image X_p should be smaller than the distance to the irrelevant image X_n with a large margin (This triplet is similar but slightly different from the one in Chapter 3 and 4, which select a positive or relevant class C_p and a negative or irrelevant class C_n for each image X_i as the triplet constraint):

$$W^T \cdot D(C_i, X_n) - W^T \cdot D(C_i, X_p) \geq 1 - \xi_{ipn}, \quad (5.4)$$

where ξ_{ipn} is a slack variable in order to allow soft-margin. With these triplet constraints, the learned weights are able to discriminate the different importance of local features in the class. Higher important local features are likely to get higher value weight and less important local features will get smaller weight, while irrelevant local features will get zero value weight and are removed from the feature set.

In the regularization term, if we impose L_2 -norm regularization on the weight vector W like standard SVM framework and our previous weight learning method in Chapter 3, the sparseness of the weight vector cannot be guaranteed. Since the number of local features in a class is usually very large, the sparsity of the learned weight is very important to make the C2I distance practical. Therefore, in this thesis we use L_1 -norm regularization in the objective function and try to learn a more sparse weight vector. The objective function of our optimization framework with L_1 -norm regularization is formulated as follows:

$$\begin{aligned} \min_W O(W) &= \|W\|_1 + C \sum_{i,p,n} \xi_{ipn}, \\ s.t. \forall i, p, n : \quad & W^T \cdot (D(C_i, X_n) - D(C_i, X_p)) \geq 1 - \xi_{ipn}, \\ & \xi_{ipn} \geq 0, \\ \forall k : \quad & W(k) \geq 0, \end{aligned} \quad (5.5)$$

where C is used to control the trade-off between regularization term and error term. With this objective function, less important and irrelevant local features will be more likely to receive a zero weight and be removed from the feature set, making the size of final feature set smaller and therefore accelerating the C2I distance calculation. To solve

this new optimization problem, we propose a gradient descent based method, which is able to effectively and efficiently update the weight vector.

Since we guarantee the non-negativeness of weight vector, the L_1 -norm $\|W\|_1$ can be reformulated as $W^T \cdot e$ where e is a constant vector with all entries to be 1 and same dimension to W , and this optimization problem is convex w.r.t. W . Therefore, the gradient descent based method can be used to iteratively update the weight vector and is guaranteed to converge to the global optimum. At each iteration, the weight vector is updated by taking a small step along the negative gradient direction to reduce the objective function and then negative components are truncated to zero to ensure the non-negativeness. To calculate the gradient of the objective function over the weight vector W , we first rewrite this objective function by replacing the slack variable ξ_{ipn} using the triplet constraints. To simplify, we denote $D(C_i, X_n) - D(C_i, X_p)$ by X_{ipn} . Since

$$\xi_{ipn} = 0 \quad \text{if} \quad W^T \cdot X_{ipn} \geq 1, \quad (5.6)$$

and

$$\begin{aligned} \xi_{ipn} &> 0 \quad \text{if} \quad W^T \cdot X_{ipn} < 1, \\ \text{and} \quad W^T \cdot X_{ipn} + \xi_{ipn} &= 1, \end{aligned} \quad (5.7)$$

at each iteration t , we can first scan over all triplets to find a set of unsatisfied triplets where $\xi_{ipn} > 0$. We denote N^t as the set of triplet indices such that $(i, p, n) \in N^t$ if $\xi_{ipn} > 0$. Then the objective function at t^{th} iteration can be rewritten as:

$$O(W) = W^T \cdot e + C \sum_{(i,p,n) \in N^t} (1 - W^T \cdot X_{ipn}). \quad (5.8)$$

The gradient $G(W^t)$ at t^{th} iteration can be calculated by taking the derivative of the above objective function w.r.t. W :

$$G(W^t) = e - C \cdot (e^T \cdot \sum_{(i,p,n) \in N^t} X_{ipn}). \quad (5.9)$$

Therefore, the weight vector W^t at t^{th} iteration is calculated as:

$$W^t = W^{t-1} - \alpha \cdot G(W^t), \quad (5.10)$$

where α is the step size, which is tuned during the iterations using the method in [18]. To satisfy the non-negativeness, all negative components of the weight vector are truncated to zero after each updating. Since the whole optimization problem is convex w.r.t. W , it is guaranteed to converge to the global optimum after iterative updating.

The main computation bottleneck in this solving method is the calculation of N^t in each iteration t , since every triplet constraint needs to be checked by calculating $W^{tT} \cdot X_{ipn}$ (If $W^{tT} \cdot X_{ipn} < 1$, then $\xi_{ipn} > 0$ and $(i, p, n) \in N^t$). This calculation is time-consuming as the dimension of W and X_{ipn} depends on the initial feature set size and is usually very huge, and the number of triplets is also large. To accelerate this calculation, we can keep an active set for triplets that have been violating the constraints and scanning only the triplets in the active set during each iteration. The full scanning over all triplets is only made at the beginning of the algorithm and every 10-20 iterations. Meanwhile, since the weight vector is usually very sparse after a few iterations, the calculation of $W^{tT} \cdot X_{ipn}$ can be accelerated as only the non-negative components are involved in the calculation. The whole work flow of updating the weight vector for each class is summarized in Algorithm 2.

Algorithm 2 The Algorithm for Solving Our Optimization Problem

Input: step size α , parameter C and pre-calculated data X_{ipn}
 $W^0 := e$ {Initialize the weight vector with all entries equal to 1}
 Set $t := 0$
repeat
 Compute N^t by checking each error term ξ_{ipn}
 Update $G(W^t)$:
 $G(W^t) := e - C \cdot (e^T \cdot \sum_{(i,p,n) \in N^t} X_{ipn})$
 Update W^t :
 $W^t := W^{t-1} - \alpha \cdot G(W^t)$
 Truncate negative components of W^t to 0
 Calculate new objective function
 $t := t + 1$
until Objective function converged
 Output: Weight Vector W

5.3 C2I Distance for Object Localization

Section 5.2 of learning the C2I distance has solved the problem of predicting whether a candidate class appears in a given test image. Here, we show the C2I distance can also tell where the object class located by predicting its bounding box in the test image. Recall that the image feature set of a test image is constructed by gathering all local features densely sampled in regular grids, while the class feature set for this class is constructed in training phase by previous work. During the C2I distance calculation, each local feature in the class will find its NN feature in the test image, and the location of these NN feature can help detecting the position of the object class, which innovates this work for the task of object localization. In the image feature set, some local features may be matched as NN by many local features from the class, while some local features may not be matched by any one from the class. We distinguish them by assigning a score for each local feature in the image. Assuming $N(f_{X_{jl}})$ is the set for local feature $f_{X_{jl}}$ in a test image X_j , which contains the local features in the class feature set F_{C_i} that finds $f_{X_{jl}}$ as its NN. We define the score for $f_{X_{jl}}$ as the summation of similarity between $f_{X_{jl}}$ and every local feature f_k in the set $N(f_{X_{jl}})$, weighted by the weight w_k associated with f_k learned by our previous C2I learning algorithm. We can recognize this score as a variant of page rank for each local feature in the test image. So higher score indicates the associated local feature may be matched as NN by many ones from the class. Local features without being matched as NN by any one from the class get a zero score. Therefore, the definition of the score is given as:

$$score(f_{X_{jl}}) = \sum_{f_k \in N(f_{X_{jl}})} w_k \cdot \exp(-\gamma \cdot \|f_{X_{jl}} - f_k\|^2), \quad (5.11)$$

where γ is a pre-defined constant parameter.

Given the score of each local feature in the test image, we try to find the bounding box with the largest density over these scores. This means on one hand we want to maximize the sum of scores within the bounding box, on the other hand we want to minimize the area of the bounding box. Therefore, we search for the predicted bounding box W_j in image X_j by the following formula:

$$W_j = \arg \max_W \left[\frac{\sum_{f_{X_{jl}} \in W} score(f_{X_{jl}})}{\sum_{f_{X_{jl}} \in X_j} score(f_{X_{jl}})} + C_L \cdot \left(1 - \frac{area(W)}{area(X_j)}\right) \right]. \quad (5.12)$$

The summation of scores for different candidate bounding boxes can be efficiently calculated by efficient subwindow search (ESS) algorithm [122]. C_L is a parameter to control the trade-off between score and area.

5.4 Experiments

5.4.1 Dataset Setup

We evaluate the performance of our method and compare it with other related methods on three multi-label image datasets: MSRC ¹, Pascal VOC 2011 [123] and MirFlickr [124]. We briefly describe the setup for each of the three datasets as follows:

MSRC dataset contains 591 images and 21 class labels, and the average number of class labels per image is 2.46. We use 10-fold cross validation on this dataset. Specifically, the dataset is divided into 10 parts and each time 9 parts are used for training and the rest one is used for test, and the average result over 10 runs is reported.

VOC2011 dataset is a much larger dataset, which contains a total of 28,952 images annotated into 20 classes. However, the majority of images in this dataset are single-labeled, on average about 1.44 labels per image. To better evaluate our method for the multi-label problem, we select a subset where most of images have more than one label. Specifically, We randomly select images with multiple labels such that each class contains at least 50 training and 50 test images, which leads to 1777 images used in our experiment and the average number of class labels per image is increased to 2.12. This random selection is conducted 5 times and the average result is reported.

MirFlickr dataset consists of 25000 images downloaded from Flickr.com. There are 1386 tags which occur in at least 20 images. We use the most common 30 content based tags as suggested by [124] in our experiment and select at least 100 images for each class, resulting in average 2.47 class labels per image. The selected image collections are equally divided into the training and test sets and we run 5 random partitions to report the average result.

In each dataset, we use SIFT feature [12] as our descriptor densely sampled at every 10 pixels to make up the feature set for training and test images. We evaluate our method

¹<http://research.microsoft.com/en-us/projects/objectclassrecognition/>

of learning the C2I distance with weight learned using L_1 -norm regularization, which is denoted as “**LC2I-L1**”, and compare it to:

- “**LC2I-L1-FS**”, which only uses the local features with non-zero weight to calculate the C2I distance but does not multiply with the learned weight, to show the performance of our feature selection strategy;
- “**C2I**”, the original C2I distance summed up by Euclidean distance from all local features without feature selection and weighting;
- “**C2I-DVW-FS**”, C2I distance with the feature selection method of selecting discriminative visual words used in [125];
- “**LC2I-L2**”, the learned C2I distance with weight learned using L_2 -norm regularization, which is solved using the method in [4].

We also compare with other related methods:

- “**I2C**”, the original NBNN method in [14];
- “**LWI2C**”, the learning of weighted I2C distance in Chapter 3;
- “**Local NBNN**”, an improved I2C distance (improving on both accuracy and efficiency) [52], see Section 2.2.2 for the detailed description;
- “**NBNN Kernel**” [51], I2C distance as input of One-vs-All SVM with linear kernel;
- “**BoW**”, Bag-of-Words model [126] for image representation with One-vs-All SVM of histogram intersection kernel.

We use mean average precision (MAP) to evaluate and compare different methods in our experiment since this evaluation metric is more popular in these datasets. The definition of MAP is given in [127] and is different from Average Precision in Section 3.5. Due to the heavy computation cost of the original C2I distance calculation, we use KD-Tree to approximate the NN search in our experiment implemented by VLFeat [128], with 5 kd-trees in each feature set and a maximum of 100 comparisons for each NN search, which does not show much difference on the final MAP compared to that of accurate NN search.

Table 5.1: Performance comparison using MAP

Method	MSRC	VOC2011	MirFlickr
LC2I-L1	0.6379 ± 0.0020	0.3522 ± 0.0036	0.2561 ± 0.0029
LC2I-L1-FS	0.5679 ± 0.0037	0.3238 ± 0.0035	0.2306 ± 0.0061
C2I	0.3438 ± 0.0007	0.2180 ± 0.0016	0.1591 ± 0.0014
C2I-DVW-FS [125]	0.3387 ± 0.0041	0.2035 ± 0.0007	0.1534 ± 0.0012
LC2I-L2 [4]	0.5864 ± 0.0010	0.3016 ± 0.0001	0.2078 ± 0.0009
I2C (NBNN) [14]	0.3549 ± 0.0016	0.1508 ± 0.0011	0.1548 ± 0.0023
LWI2C	0.5231 ± 0.0017	0.1690 ± 0.0003	0.1619 ± 0.0016
Local NBNN [52]	0.6019 ± 0.0052	0.2669 ± 0.0018	0.1979 ± 0.0021
NBNN Kernel [51]	0.6324 ± 0.0051	0.2716 ± 0.0038	0.1964 ± 0.0024
BoW [126]	0.6110 ± 0.0024	0.3096 ± 0.0014	0.2299 ± 0.0033

5.4.2 Classification Experiment Result

Table 5.1 shows the result of different methods on the three datasets, and we can see that the proposed method, “LC2I-L1” significantly outperforms all other methods. In particular, the improvement over the original C2I distance can be divided into two parts: feature selection by removing local features with zero value weight and the discrimination of different importance for the preserved local features by their associated non-zero weight.

The improvement over feature selection by removing local features with zero value weight can be seen by comparing the performance of “LC2I-L1-FS” to “C2I”. “LC2I-L1-FS” only uses the weight to select relevant local features without weighting in the C2I distance calculation, and its improvement over the original C2I distance shows that this feature selection is able to remove the irrelevant local features that deteriorate the performance of the C2I distance. These irrelevant local features have to be put into the initial feature set of each relevant class because the training images may also belong to multiple classes but the information that which region in the image belongs to which class is not provided since this pixel-wise or bounding box human labeling is expensive. Due to the inclusion of these irrelevant local features, the original C2I distance does not show better performance compared to I2C distance on MSRC and MirFlickr datasets as shown in Table 5.1. However, with the feature selection to remove these irrelevant local

features, the performance of the C2I distance can be largely improved, as seen when comparing “LC2I-L1-FS” to “C2I”.

The improvement over the second part of non-zero weighting can be recognized by comparing the performance of “LC2I-L1” to “LC2I-L1-FS”. This improvement shows the learned weight not only provides a feature selection strategy to reduce the size of feature set and remove irrelevant local features, but also discriminates the different importance of the preserved local features in the final feature set by their associated non-zero weight to further improve the C2I distance, which validates the effectiveness of our learning algorithm.

Compared to I2C distance related methods, our learned C2I distance is able to achieve better recognition performance under the multi-label problem. Though I2C distance of non-learning methods like NBNN and Local NBNN may benefit from a larger feature set sampled from denser grids, their performances are still worse than our method. To verify this, we have tried sampling at every 4 pixels and the results show that Local NBNN is only slightly improved to 0.2846 in VOC2011 dataset. It should be noted the improvement over I2C distance methods is based on the multi-label problem. When more images are single-labeled, e.g. in the full VOC dataset, such large improvement is not expected. Due to the computation limitation of our working machine for the heavy NN search in distance calculation, we only try class “aeroplane”, “bicycle”, “car”, “chair” and “person” for the full VOC 2007 dataset, and find the performance of our method is worse than I2C distance (0.502 vs 0.649 [129]), while in our multi-label subset version of these classes, our method significantly outperform NBNN (0.467 vs 0.247).

We also compare the performance of learning weight using L_1 -norm regularization to L_2 -norm. By comparing “LC2I-L1” to “LC2I-L2” in Table 5.1, we can see that even using L_1 -norm to learn a sparser solution, the recognition performance is still better than using L_2 -norm, which again shows the effectiveness of our learning algorithm. Meanwhile, the resulting feature set learned by L_1 -norm is much sparser than L_2 -norm. This is shown in Table 5.2, where the sparsity is represented by the percentage of the number of preserved local features compared to the original feature set. With L_1 -norm regularization, only 3.35% local features are preserved in MSRC dataset and in VOC2011 and MirFlickr datasets, this is reduced to only 0.26% and 0.31%, respectively. This is a much larger

Table 5.2: Sparsity comparison (Percentage of the number of local features compared to the original feature set)

Method	MSRC	VOC2011	MirFlickr
LC2I-L1	3.35% $\pm$ 0.14	0.26% $\pm$ 0.01	0.31% $\pm$ 0.02
LC2I-L2 [4]	12.50% $\pm$ 0.10	5.86% $\pm$ 0.16	1.58% $\pm$ 0.32
C2I-DVW-FS [125]	18.45% $\pm$ 0.15	16.20% $\pm$ 0.09	16.85% $\pm$ 0.12
C2I,I2C [14]	100%	100%	100%

Table 5.3: Running time comparison (second)

Method	MSRC	VOC2011	MirFlickr
LC2I-L1	0.61 $\pm$ 0.03	0.63 $\pm$ 0.05	0.61 $\pm$ 0.04
LC2I-L2 [4]	3.02 $\pm$ 0.22	5.64 $\pm$ 0.53	1.86 $\pm$ 0.15
C2I-DVW-FS [125]	9.12 $\pm$ 0.41	21.08 $\pm$ 1.53	22.20 $\pm$ 1.01
C2I	51.95 $\pm$ 1.07	132.88 $\pm$ 5.03	129.72 $\pm$ 3.05
I2C [14]	5.08 $\pm$ 0.13	7.44 $\pm$ 0.61	14.29 $\pm$ 0.33
Local NBN [52]	1.18 $\pm$ 0.09	3.04 $\pm$ 0.22	3.57 $\pm$ 0.15

Table 5.4: Memory usage comparison (MB)

Method	MSRC	VOC2011	MirFlickr
LC2I-L1	5.91 $\pm$ 0.04	6.12 $\pm$ 0.05	5.38 $\pm$ 0.08
LC2I-L2 [4]	31.19 $\pm$ 0.51	65.38 $\pm$ 2.44	20.34 $\pm$ 0.40
C2I-DVW-FS [125]	101.88 $\pm$ 1.49	240.25 $\pm$ 3.72	254.08 $\pm$ 4.13
C2I,I2C [14]	547.88 $\pm$ 3.26	1508.94 $\pm$ 10.28	1512.23 $\pm$ 15.11

reduction on both computation cost of the C2I distance calculation and memory usage, since their complexities are linear to the size of class feature set. To verify this reduction, we show in Table 5.3 the average running time of distance calculation for a test image over all classes and Table 5.4 shows the memory usage for loading the training feature sets of all classes. All of the experiments are running on a single core of **Intel x86 Xeon CPU E7320@2.13GHz** and **16GB** memory.

Compared to I2C distance, the original C2I distance needs much more expensive computation cost since the number of local features in a class is much more than that in an image, while both methods need the same amount of memory usage to store the whole feature sets of classes for distance calculation. However, with our learning method using L_1 -norm regularization, only a very small number of local features are preserved in the final feature set for each class. Therefore, the computation cost of our C2I distance can be even lower than I2C distance, and the memory usage is greatly reduced. Even compared

Table 5.5: MAP and sparsity of our method over different C value on MSRC dataset

C	MAP	Sparsity
10^{-3}	0.6027 ± 0.0031	$1.73\% \pm 0.03$
10^{-2}	0.6379 ± 0.0020	$3.35\% \pm 0.14$
10^{-1}	0.6054 ± 0.0007	$8.71\% \pm 0.23$
10^0	0.5727 ± 0.0020	$14.97\% \pm 0.51$
10^1	0.5669 ± 0.0007	$20.55\% \pm 0.95$

to Local NBN, which is the most efficient method in I2C distance computation so far, our method still needs less running time for distance calculation.

The second best result in MSRC dataset is NBN Kernel, and BoW ranks the second best in VOC2011 and MirFlickr datasets (excluding the L_2 -norm version of our method), as shown in Table 5.1. The commonality of both methods is that they both use One-vs-All SVM. We guess this would be the reason that makes these two methods perform well and One-vs-All SVM would be very effective in dealing with multi-label problem.

The method of “C2I-DVW-FS” uses the idea of selecting discriminative visual words in [125] to perform the feature selection task, and is done for each class separately. Local features in each class are clustered to 1000 visual words by K-means clustering, and then the top 64 most discriminative (with the most term frequency) visual words are selected. Local features belonging to these visual words are selected to make up the final feature set for each class, while the rest local features are removed. For the C2I distance calculation, only these preserved local features in the final class feature set are used. However, the result in Table 5.1 and Table 5.2 shows that its recognition performance is even worse than the original C2I distance, probably because label information is not utilized in the clustering and selection procedure. Therefore, this method cannot distinguish relevant local features used in the C2I distance. Meanwhile, its sparsity is not as good as our learning method. In summary, our learned C2I distance not only achieves better recognition performance, but also greatly reduces the computation cost and memory usage.

5.4.3 The Influence of Parameter C

We also analyze the impact of the trade-off parameter C in the objective function. Table 5.5 shows the MAP and sparsity under different C values ranging from 10^{-3} to 10^1 on

MSRC dataset. With the increase of C , the size of the feature set is always increased, meaning more local features are preserved and more computation cost and memory usage are required. This can be recognized as the increase of C makes the objective function pay less emphasis on the sparsity of the weight vector and therefore resulting in more local features preserved. However, when C is increasing, the recognition performance is first increased but then decreased, as a large value of C might put too much emphasis on the error term to cause over-fitting. The best MAP is achieved when C is equal to 10^{-2} on MSRC dataset. If we further increase C after this peak point, both the recognition performance and efficiency are getting worse. For our results shown in Table 5.1, the C value is determined by selecting the one with the best MAP, i.e., for MSRC dataset C is set to 10^{-2} and for the other two datasets it is 10^{-3} .

5.4.4 Resulting Image Examples

We have already shown in Figure 5.2 an example of our learned local features in the training feature set of class “car” and their NN features in some test images containing the car. In this section we show more classes as examples. Figure 5.3 shows the distribution of the local features in the training feature set of class “sky”, “dog”, “flower”, “tree” and “sea”, marked by red rectangles. These local features are selected by our learned weight from the original large size feature sets constructed by dense sampling, and we can see that most of them are correctly located at the regions of these classes. This can significantly validate the effectiveness of our learning algorithm, which uses the training label information to select relevant local features to make up the feature set for each class, with the L_1 -norm regularization to ensure a small size for the final feature set. Figure 5.4 shows some representative examples of test images belonging to these classes. The local features matched as NN features by our learned C2I distance are marked by red rectangles and most of them are located at the region of these classes as well.

5.4.5 Object Localization Result

We use Pascal VOC2007 [127] for evaluating the performance on object localization as the ground truth of bounding box data of the test set in VOC2011 and other datasets are not public available. For a specific class, we use the C2I distance to detect the bounding

Figure 5.3: The representative example of local features in the training feature set of classes selected by our learning method.

Figure 5.4: The representative example of local features in test images matched as NN feature by our learned C2I distance.

Table 5.6: Localization accuracy of our method compared with other methods on VOC2007 dataset

	Ours	Blaschko et al. [132]	Everingham et al. [127]	Vedaldi et al. [133]	Galleguillos et al. [134]	Galleguillos et al. [130]
aeroplane	0.45	0.11	0.26	0.38	0.63	0.33
bicycle	0.36	0.12	0.41	0.48	0.22	0.24
bird	0.32	0.09	0.10	0.15	0.14	0.47
boat	0.27	0.06	0.09	0.15	0.42	0.69
bottle	0.06	0.00	0.21	0.22	0.43	0.22
bus	0.50	0.25	0.39	0.51	0.50	0.37
car	0.44	0.14	0.43	0.51	0.62	0.71
cat	0.56	0.36	0.24	0.30	0.32	0.33
chair	0.12	0.09	0.13	0.17	0.37	0.07
cow	0.34	0.14	0.14	0.33	0.19	0.15
diningtable	0.45	0.24	0.10	0.23	0.30	0.74
dog	0.48	0.32	0.16	0.22	0.29	0.21
horse	0.50	0.27	0.34	0.51	0.15	0.26
motorbike	0.54	0.34	0.38	0.46	0.31	0.55
person	0.25	0.03	0.22	0.23	0.43	0.33
pottedplant	0.17	0.02	0.12	0.12	0.33	0.29
sheep	0.30	0.09	0.18	0.24	0.41	0.38
sofa	0.44	0.30	0.15	0.29	0.37	0.23
train	0.51	0.30	0.33	0.45	0.29	0.51
tvmonitor	0.14	0.08	0.29	0.49	0.62	0.57
Mean	0.36	0.17	0.23	0.32	0.37	0.39

box of objects only in test images that actually belonging to this class, so as to measure the performance of pure localization task as in [122]. We use the overlap area provided by Pascal VOC for evaluation, which is calculated by the intersection area of the predicted bounding box and the ground truth bounding box divided by their union area. The overlap area above 50% is considered as a correct localization. The average localization accuracy for all test images in each class is reported in Table 5.6 and compared with other localization methods. Though our method only uses one single SIFT feature descriptor as consistent with all our other experiments in this thesis, our method achieves a comparable localization accuracy averaged over all classes with Galleguillos *et al.* [130], where they used multiple different descriptors (SIFT, SSIM [131], PHOG [23], etc.) and incorporated different levels of contextual interactions.

We also show some good, median and bad examples for some classes according to

the overlap area. For each category of the examples, images in the left column visualize the scores of local features by the radius of blue circles, so that high score is drawn by larger radius and low score with small radius, while score of zero is without drawing a circle. Therefore, though the local features in each image are densely sampled in regular grids, only part of them with non-zero score are shown. Images in the right column show the predicted bounding box (blue) and the ground truth bounding box (red). The predicted bounding box is generated by scores of local features trying to have the largest density over these scores. We can see that in “good” category the predicted bounding boxes match ground truth very well, even when there contain other object classes in these multi-label test images. In “median” category, the background and other object classes make the localization task more challenging. In such situation the predicted bounding box has a lower overlap area with the ground truth compared to “good” category, while our method can still find a correct position of the localized object class. The worst performance in “bad” category is because the object class appears too small in the test image to be localized, and the local features from the background and other related objects are visually similar to the learned local features in the training set, therefore our method fails to find their bounding boxes.

5.5 Summary

In this chapter, we have discussed the drawbacks of I2C distance in classifying multi-label images and proposed to use the C2I distance for dealing with multi-label image classification. To select relevant local features for each class and speed up the C2I distance calculation, we associated a weight for each local feature densely sampled from images, and proposed an optimization algorithm using large margin constraint and L_1 -norm regularization to learn these weights. A gradient descent based solver is proposed to efficiently and effectively update these weights in the learning procedure. Experiments on three multi-label datasets have shown that our learning algorithm can not only improve the performance of the C2I distance and outperform other related methods, but also significantly reduce the number of local features in the class feature set and greatly improve the efficiency. We also used the learned C2I distance for the task of object localization and achieved excellent localization performance.

Figure 5.5: The good, median and bad localization examples according to the overlap area. Blue bounding box is predicted by our method and red bounding box is the ground truth.

However, it should be noted such performance and efficiency improvements in the test phase is a trade-off of the additional training phase for learning the C2I distance. The original C2I distance calculation in the training phase is computation expensive since this is done before the learning procedure and the whole feature set is used in the calculation. Its running time is similar to the row of “C2I” in Table 5.3, which also uses the whole feature set for C2I distance calculation. If there are a large number of local features in the original feature set, an initial local feature selection before the learning procedure may be required to alleviate the computation cost of distance calculation in the training phase, which would be a possible work in the future.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we have tried to learn the distance between image and class for the problem of object recognition. Our work can be divided into three main parts:

- We proposed a large margin optimization framework to learn a weighted I2C distance, which is modeled as a weighted combination of the distance from every local feature in an image to its NN in a candidate class. These weights associated with local features in the training set are learned by constraining the optimization such that the I2C distance from image to its belonging class should be less than that to any other class. Experiment result showed that the performance of the I2C distance for classification can significantly be improved by learning a weighted I2C distance function. For efficiency, we also proposed two methods based on spatial division and hubness score to accelerate the NN search, which is able to largely reduce the on-line testing time while still preserving or even achieving a better classification accuracy.
- Both the original I2C distance and the weighted I2C distance use Euclidean distance for measuring the Feature-to-Set distance. From experience, we know that choosing a good distance metric is a critical problem for object recognition, thus we proposed a distance metric learning method to learn a Mahalanobis I2C distance to further improve the performance of the I2C distance. Instead of learning one global distance metric, we learned the class specific Mahalanobis metric for

each class, which makes the learned Mahalanobis I2C distance more discriminative especially when there are a large number of classes. A large margin framework has been formulated to simultaneously learn the Per-Class Mahalanobis distance metrics, and a subgradient descent method is proposed to efficiently solve this optimization problem. For efficiency and scalability to large-scale problems, we simplified the method to learn only a diagonal matrix for each class instead of the full matrix, which can significantly reduce the computation cost and memory usage while preserving the classification performance. The experiment results verified that our I2CDML method can significantly outperform the original Euclidean I2C distance as well as other distance metric learning methods that learn a Mahalanobis I2I distance. We also showed that the learned Per-Class metrics are able to correct the class imbalance problem.

- Due to the drawback of the I2C distance in classifying multi-label images, we proposed to use the C2I distance for measuring the distance between image and class for the multi-label problem. This distance is able to outperform the I2C distance when images belong to multiple labels. To select relevant local features for each class and speed up the C2I distance calculation, we associated a weight for each local feature densely sampled from images, and proposed an optimization algorithm using large margin constraint and L_1 -norm regularization to learn these weights. With this optimization algorithm, we are able to get better recognition performance even though the size of the resulting feature set is much smaller. We also used this C2I distance for the task of object localization, by using the location information of the NN matched local feature in a given test image to predict its bounding box.

6.2 Future Work

In future, we can extend our work to the following directions:

First, our current distance metric learning algorithm needs to learn a Mahalanobis metric (either full or diagonal matrix) for each class, which may be too complex when there are thousands of classes. A decomposition of our learning algorithm would be useful

for dealing with such large number of classes. We could construct a class hierarchy to split these classes into sub-problems. In each sub-problem the number of classes is limited and our current distance metric learning algorithm could be used. Then a global merging method should be developed to make the learned distance metrics in each sub-problem comparable for the final classification. The class hierarchy can either be learned from the training data as in [135–137] or using the semantic structure of classes as in ImageNet dataset [138]. However, though efficiency improving is gained, such decomposition to class hierarchy would possibly lead to performance degradation since a wrong prediction in any sub-problem would result in an incorrect classification in final. The construction of the class hierarchy is an important issue to reduce the performance drop and therefore it requires further study.

Second, the original C2I distance calculation in the training phase is computation expensive since the number of local features in a class is much more than that in an image before our learning algorithm for feature selection, which limits its usage for training in a large scale dataset. We will try some initial local feature selection methods to alleviate the computation cost of distance calculation in the training phase and improve the training efficiency. One possible direction is to use the method of local feature reduction by hubness score as discussed in Chapter 3, which ranks local features according to how frequent it is matched as NN by local features in images of the same class and how less frequent it is matched from local features in different classes. In this way, common pattern local features in each class are likely to be preserved for calculating the C2I distance before learning the weight.

Third, we can learn the I2C distance for target domain from source domain. This would be useful when target domain has less labeled instances while source domain contains plentiful label information. The learned model in source domain is expected to help learning the I2C distance function in target domain. How to bridge the content gap between source domain and target domain for each class is the critical issue for the I2C distance transfer learning while still yet to be explored.

Fourth, the I2C distance would be benefit from active learning, which produces the most uncertain images for labeling from a set of unlabeled images. In the I2C distance framework, the uncertainty of an image can be represented by the closeness between two

(or more) shortest I2C distances. With active learning, the most useful triplet constraints can be selected from these uncertain images, and help learning the weights or distance metric for the I2C distance. However, how to compare the uncertainty between different images in order to select the most uncertain images remains a problem, which requires further study for actively learning the I2C distance.

Fifth, we are more interested in how to further improve the I2C distance. The current learning framework mainly exploits the triplet constraint satisfying that the I2C distance from each image to its belonging class should be less than the distance to any other class. This type of triplet constraint is able to help improve the recognition performance of the original I2C distance to some extent. However, there would be other types of constraints to be added in the optimization problem as complementary to our current method and further improve the I2C distance, which should be studied in the future work.

Sixth, the type of Set-to-Set distance (either I2C or C2I distance) described in this thesis may be used to the image set classification, which is popularly used in recent face recognition studies [139, 140]. Our current work only targets single image classification, where each image is represented by a set of feature vectors (local descriptors) with equal dimension and varying set size. However, our work also supports other problems where each object instance is represented by a set of feature vectors. In the image set classification, each training and test instance is an image set rather than a single image, and classifying an image set is proved to be better in recognition performance than classifying a single image [139, 140]. Each image set is composed of a set of images with varying set size, and each image is represented by a feature vector with equal dimension, either extracted by some local descriptor or directly concatenated by the original pixel values. The critical problem in the image set classification is how to effectively measure the distance between two image sets. Since the representation of an image set is similar to the single image representation in this thesis, our distance learning framework may be used for measuring the distance between image sets.

Finally, our I2C distance learning framework may have the potential to be applied for video recognition. A video composing a set of frames can also be represented by a set of feature vectors and each frame in a video could be seen as a local feature in our current framework. The time series information of each frame in the video could be treated as

the location information of the local feature in the image. Since the time series in the video is one dimension while location information in the image is two dimensions, a slight modification would be required by simplifying the two dimensional spatial division in image classification to one dimensional time series division for video recognition. However, uniform division would not be a good solution for this problem since the content of the video may not necessarily be distributed equally in the video. Therefore further study is required to learn the time series division according to the visual content of the video.

Publication

- (i) **Zhengxiang Wang**, Yiqun Hu and Liang-Tien Chia “Learning Instance-To-Class Distance for Human Action Recognition” Proceedings of 16th IEEE International Conference on Image Processing 2009
- (ii) **Zhengxiang Wang**, Yiqun Hu and Liang-Tien Chia “Multi-Label Learning by Image-To-Class Distance for Scene Classification and Image Annotation” ACM International Conference on Image and Video Retrieval 2010
- (iii) **Zhengxiang Wang**, Yiqun Hu and Liang-Tien Chia “Image-To-Class Distance Metric Learning for Image Classification” European Conference on Computer Vision 2010
- (iv) Shenghua Gao, **Zhengxiang Wang**, Liang-Tien Chia and Ivor Wai-Hung Tsang “Automatic Image Tagging via Category Label and Web Data” ACM Multimedia 2010
- (v) **Zhengxiang Wang**, Yiqun Hu and Liang-Tien Chia “Improving the Learning of Image-To-Class Distance and Accelerating the NN Search for Better Image Classification” Pattern Recognition, Volume 44 Issue 10-11, October, 2011
- (vi) **Zhengxiang Wang**, Shenghua Gao and Liang-Tien Chia ”Learning Class-to-Image Distance via Large Margin and L1-norm Regularization ” European Conference on Computer Vision 2012
- (vii) **Zhengxiang Wang**, Yiqun Hu and Liang-Tien Chia “Learning Image-To-Class Distance Metric for Image Classification ”, ACM Transactions on Intelligent Systems and Technology, Volume 4 Issue 2, 2013

References

- [1] J. Yang, Y.-G. Jiang, A. G. Hauptmann, and C.-W. Ngo, “Evaluating bag-of-visual-words representations in scene classification,” in *Proceedings of the international workshop on multimedia information retrieval*, 2007.
- [2] S. Lazebnik, C. Schmid, and J. Ponce, “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2169–2178, 2006.
- [3] A. Frome, Y. Singer, and J. Malik, “Image retrieval and classification using local distance functions,” in *Advances in Neural Information Processing Systems 19*, 2006.
- [4] A. Frome, Y. Singer, F. Sha, and J. Malik, “Learning globally-consistent local distance functions for shape-based image retrieval and classification,” in *Proceedings of IEEE International Conference on Computer Vision*, 2007.
- [5] H. Wang, H. Huang, F. Kamangar, F. Nie, and C. Ding, “Maximum margin multi-instance learning,” in *Advances in Neural Information Processing Systems 24*, 2011.
- [6] P. Viola and M. J. Jones, “Robust real-time face detection,” *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.
- [7] S. Mori, H. Nishida, and H. Yamada, *Optical character recognition*. John Wiley & Sons, Inc., 1999.
- [8] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

- [9] A. K. Hrechak and J. A. McHugh, “Automated fingerprint recognition using structural matching,” *Pattern Recognition*, vol. 23, no. 8, pp. 893–904, 1990.
- [10] S. Mitra and T. Acharya, “Gesture recognition: A survey,” *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 37, no. 3, pp. 311–324, 2007.
- [11] C. Wang, S. Yan, L. Zhang, and H.-J. Zhang, “Multi-label sparse coding for automatic image annotation,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1643–1650, June 2009.
- [12] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, 2004.
- [13] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, “SURF: Speeded up robust features,” *Computer Vision and Image Understanding*, vol. 110, no. 3, 2008.
- [14] O. Boiman, E. Shechtman, and M. Irani, “In defense of nearest-neighbor based image classification,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [15] J. Yang, K. Yu, Y. Gong, and T. Huang, “Linear spatial pyramid matching using sparse coding for image classification,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009.
- [16] A. Opelt, M. Fussenegger, A. Pinz, and P. Auer, “Weak hypotheses and boosting for generic object detection and recognition,” in *Proceedings of European Conference on Computer Vision*, 2004.
- [17] A. Bosch, A. Zisserman, and X. Munoz, “Image classification using random forests and ferns,” in *Proceedings of IEEE International Conference on Computer Vision*, 2007.
- [18] K. Q. Weinberger and L. K. Saul, “Distance metric learning for large margin nearest neighbor classification,” *Journal of Machine Learning Research*, vol. 10, pp. 207–244, June 2009.

- [19] J. Wu and J. M. Rehg, “Beyond the euclidean distance: Creating effective visual codebooks using the histogram intersection kernel,” in *Proceedings of IEEE International Conference on Computer Vision*, 2009.
- [20] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, “Lost in quantization: improving particular object retrieval in large scale image databases,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [21] J. C. van Gemert, J.-M. Geusebroek, and C. J. Veenman, “Kernel codebooks for scene categorization,” in *Proceedings of European Conference on Computer Vision*, 2008.
- [22] K. Grauman and T. Darrell, “The pyramid match kernel: Discriminative classification with sets of image features,” in *Proceedings of IEEE International Conference on Computer Vision*, 2005.
- [23] A. Bosch, A. Zisserman, and X. Munoz, “Representing shape with a spatial pyramid kernel,” in *Proceeding of the ACM International Conference on Image and Video Retrieval*, 2007.
- [24] J. Wang, J. Yang, K. Yu, F. Lv, and Y. Gong, “Locality-constrained linear coding for image classification,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.
- [25] J. Yang, K. Yu, and T. S. Huang, “Supervised translation-invariant sparse coding,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.
- [26] S. Gao, I. W.-H. Tsang, L.-T. Chia, and P. Zhao, “Local features are not lonely c laplacian sparse coding for image classification,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3555 – 3561, 2010.

- [27] X.-T. Yuan and S. Yan, “Visual classification with multi-task joint sparse representation,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.
- [28] S. Gao, I. W.-H. Tsang, and L.-T. Chia, “Kernel sparse representation for image classification and face recognition,” in *Proceedings of European Conference on Computer Vision*, 2010.
- [29] M. Yang and L. Zhang, “Gabor feature based sparse representation for face recognition with gabor occlusion dictionary,” in *Proceedings of European Conference on Computer Vision*, 2010.
- [30] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, “Robust face recognition via sparse representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.
- [31] M. Yang and L. Zhang, “Robust sparse coding for face recognition,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2011.
- [32] H. Zhang, J. Yang, Y. Zhang, N. Nasrabadi, and T. Huang, “Close the loop: Joint blind image restoration and recognition with sparse representation prior,” in *Proceedings of IEEE International Conference on Computer Vision*, 2011.
- [33] I. Theodorakopoulos, I. Rigas, G. Economou, and S. Fotopoulos, “Face recognition via local sparse coding,” in *Proceedings of IEEE International Conference on Computer Vision*, 2011.
- [34] L. Zhang, P. Zhu, Q. Hu, and D. Zhang, “A linear subspace learning approach via sparse coding,” in *Proceedings of IEEE International Conference on Computer Vision*, 2011.
- [35] L. Ma, C. Wang, B. Xiao, and W. Zhou, “Sparse representation for face recognition based on discriminative low-rank dictionary learning,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012.

- [36] Z. Jiang, G. Zhang, and L. S. Davis, "Submodular dictionary learning for sparse coding," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012.
- [37] Y. Li, Y. Zhou, L. Xu, X. Yang, and J. Yang, "Incremental sparse saliency detection," in *Proceedings of the IEEE International Conference on Image Processing*, 2009.
- [38] S. Gao, L.-T. Chia, and I. W.-H. Tsang, "Multi-layer group sparse coding - for concurrent image classification and annotation," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2011.
- [39] J. Yang, J. Wright, and T. Huang, "Image super-resolution as sparse representation of raw image patches," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [40] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010.
- [41] Y. Zhang, G. Wu, P.-T. Yap, Q. Feng, J. Lian, W. Chen, and D. Shen, "Reconstruction of super-resolution lung 4d-ct using a patch-based sparse representation," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012.
- [42] X. Lu, H. Yuan, P. Yan, Y. Yuan, and X. Li, "Geometry constrained sparse coding for single image super-resolution," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012.
- [43] V. Cevher, A. Sankaranarayanan, M. F. Duarte, D. Reddy, R. G. Baraniuk, and R. Chellappa, "Compressive sensing for background subtraction," in *Proceedings of European Conference on Computer Vision*, 2008.
- [44] M. Dikmen and T. S. Huang, "Robust estimation of foreground in surveillance videos by sparse error estimation," in *Proceedings of International Conference on Pattern Recognition*, 2008.

- [45] Y. Xiang, X. Zhou, Z. Liu, T.-S. Chua, and C.-W. Ngo, “Semantic context modeling with maximal margin conditional random fields for automatic image annotation,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.
- [46] B. Ghanem and N. Ahuja, “Maximum margin distance learning for dynamic texture recognition,” in *Proceedings of European Conference on Computer Vision*, 2010.
- [47] X.-C. Lian, Z. Li, B.-L. Lu, and L. Zhang, “Max-margin dictionary learning for multiclass image categorization,” in *Proceedings of European Conference on Computer Vision*, 2010.
- [48] C. H. Lampert, “Maximum margin multi-label structured prediction,” in *Advances in Neural Information Processing Systems 24*, 2011.
- [49] J. Deng, A. C. Berg, K. Li, and L. Fei-Fei, “What does classifying more than 10,000 image categories tell us?,” in *Proceedings of European Conference on Computer Vision*, 2010.
- [50] R. Behmo, P. Marcombes, A. Dalalyan, and V. Prinet, “Towards optimal naive bayes nearest neighbor,” in *Proceedings of European Conference on Computer Vision*, 2010.
- [51] T. Tuytelaars, M. Fritz, K. Saenko, and T. Darrell, “The nbnn kernel,” in *Proceedings of IEEE International Conference on Computer Vision*, 2011.
- [52] S. McCann and D. G. Lowe, “Local naive bayes nearest neighbor for image classification,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012.
- [53] J. H. Friedman, J. L. Bentley, and R. A. Finkel, “An algorithm for finding best matches in logarithmic expected time,” *ACM Transactions on Mathematical Software*, vol. 3, no. 3, pp. 209–226, 1977.
- [54] Y. Huang, D. Xu, and T.-J. Cham, “Face and human gait recognition using image-to-class distance,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 3, pp. 431–438, 2010.

- [55] X. Cai, B. Xiao, C. Wang, and R. Zhang, “A local learning based image-to-class distance for image classification,” in *Proceedings of Asian Conference on Pattern Recognition*, 2011.
- [56] D. Zhang, B. Liu, C. Sun, and X. Wang, “Making image to class distance comparable,” in *International Conference on Neural Information Processing*, 2011.
- [57] H. Wang, F. Nie, and H. Huang, “Learning instance specific distance for multi-instance classification,” in *AAAI-11: Proceedings of the 25th AAAI conference on Artificial intelligence*, 2011.
- [58] H. Wang, F. Nie, H. Huang, and Y. Yang, “Learning frame relevance for video classification,” in *MM ’11 Proceedings of the 19th ACM international conference on Multimedia*, 2011.
- [59] H. Wang, F. Nie, and H. Huang, “Robust and discriminative distance for multi-instance learning,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012.
- [60] Y. Jia, J. Wang, G. Zeng, H. Zha, and X.-S. Hua, “Optimizing kd-trees for scalable visual descriptor indexing,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.
- [61] J. Beis and D. G. Lowe, “Shape indexing using approximate nearest-neighbour search in high-dimensional spaces,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997.
- [62] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, “Object retrieval with large vocabularies and fast spatial matching,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.
- [63] K. Mikolajczyk and J. Matas, “Improving descriptors for fast tree matching by optimal linear projection,” in *Proceedings of IEEE International Conference on Computer Vision*, 2007.

- [64] C. Silpa-Anan and R. Hartley, “Optimised kd-trees for fast image descriptor matching,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [65] P. S. Bradley and O. L. Mangasarian, “Similarity search in high dimensions via hashing,” in *Proceedings of the International Conference on Very Large Data Bases*, 1999.
- [66] G. Shakhnarovich, P. Viola, and T. Darrell, “Fast pose estimation with parameter-sensitive hashing,” in *Proceedings of IEEE International Conference on Computer Vision*, 2003.
- [67] O. Chum, J. Philbin, and A. Zisserman, “Near duplicate image detection: min-hash and tf-idf weighting,” in *Proceedings of British Machine Vision Conference*, 2008.
- [68] Y. Liang, J. Li, and B. Zhang, “Vocabulary-based hashing for image search,” in *MM ’09 Proceedings of the 17th ACM international conference on Multimedia*, 2009.
- [69] P. Jain, B. Kulis, and K. Grauman, “Fast image search for learned metrics,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [70] B. Kulis and K. Grauman, “Kernelized locality-sensitive hashing for scalable image search,” in *Proceedings of IEEE International Conference on Computer Vision*, 2009.
- [71] J. Deng, A. Berg, and L. Fei-Fei, “Hierarchical semantic indexing for large scale image retrieval,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2011.
- [72] R. E. Schapire and Y. Singer, “BoosTexter: a boosting-based system for text categorization,” *Machine Learning*, vol. 39, no. 2-3, pp. 135–168, 2000.
- [73] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, “Learning multi-label scene classification,” *Pattern Recognition*, vol. 37, no. 9, pp. 1751–1771, 2004.

- [74] M.-L. Zhang and Z.-H. Zhou, “Ml-knn: A Lazy learning approach to multi-label learning,” *Pattern Recognition*, vol. 40, no. 7, 2007.
- [75] M.-L. Zhang and Z.-H. Zhou, “Multi-label learning by instance differentiation,” in *AAAI’07: Proceedings of the 22nd AAAI conference on Artificial intelligence*, pp. 669–674, 2007.
- [76] Z.-H. Zhou and M.-L. Zhang, “Multi-instance multi-label learning with application to scene classification,” in *Advances in Neural Information Processing Systems 19*, (Cambridge, MA), pp. 1634–1641, MIT Press, 2006.
- [77] M.-L. Zhang and Z.-H. Zhou, “M3MIML: a maximum margin method for multi-instance multi-label learning,” in *Proceedings of IEEE International Conference on Data Mining*, pp. 688–697, December 2008.
- [78] M. Wang, X. Zhou, and T.-S. Chua, “Automatic image annotation via local multi-label classification,” in *Proceeding of the ACM International Conference on Image and Video Retrieval*, 2008.
- [79] T. Li, T. Mei, S. Yan, I.-S. Kweon, and C. Lee, “Contextual decomposition of multi-label images,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2270–2277, June 2009.
- [80] Z. Lu, H. H. Ip, and Q. He, “Context-based multi-label image annotation,” in *Proceeding of the ACM International Conference on Image and Video Retrieval*, July 2009.
- [81] H. Wang, H. Huang, and C. Ding, “Image annotation using multi-label correlated green’s function,” in *Proceedings of IEEE International Conference on Computer Vision*, 2009.
- [82] M.-L. Zhang and Z.-H. Zhou, “A review on multi-label learning algorithms,” *IEEE Transactions on Knowledge and Data Engineering*, In press.
- [83] M. Radovanovic, A. Nanopoulos, and M. Ivanovic, “Nearest neighbors in high-dimensional data: The emergence and influence of hubs,” in *Proceedings of International Conference on Machine Learning*, 2009.

- [84] G. Tsoumakas, A. Dimou, E. Spyromitros, V. Mezaris, I. Kompatsiaris, and I. Vlahavas, “Correlation-based pruning of stacked binary relevance models for multi-label learning,” in *ECML PKDD Workshop on Learning From Multi-Label Data*, pp. 101–116, September 2009.
- [85] Z.-J. Zha, T. Mei, J. Wang, Z. Wang, and X.-S. Hua, “Graph-based semi-supervised learning with multi-label,” in *IEEE International Conference on Multimedia and Expo*, pp. 1321–1324, 2008.
- [86] F. Kang, R. Jin, and R. Sukthankar, “Correlated label propagation with application to multi-label learning,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1719–1726, 2006.
- [87] H. W. Kuhn and A. W. Tucker, “Nonlinear programming,” in *Proceedings of 2nd Berkeley Symposium*, pp. 481 – 492, Berkeley: University of California Press, 1951.
- [88] A. Oliva and A. Torralba, “Modeling the shape of the scene: A holistic representation of the spatial envelope,” *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.
- [89] L. Fei-Fei and P. Perona, “A bayesian hierarchical model for learning natural scene categories,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 524–531, 2005.
- [90] J. Li and L. Fei-Fei, “What, where and who? Classifying events by scene and object recognition,” in *Proceedings of IEEE International Conference on Computer Vision*, 2007.
- [91] Z. Lu and H. H. Ip, “Image categorization with spatial mismatch kernels,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009.
- [92] L. Fei-Fei, R. Fergus, and P. Perona, “Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories,” in *CVPR Workshop on Generative-Model Based Vision*, 2004.

- [93] G. Griffin, A. Holub, and P. Perona, “Caltech-256 object category dataset,” Tech. Rep. 7694, California Institute of Technology, 2007.
- [94] J. Liu and M. Shah, “Scene modeling using co-clustering,” in *Proceedings of IEEE International Conference on Computer Vision*, 2007.
- [95] N. Rasiwasia and N. Vasconcelos, “Scene classification with low-dimensional semantic spaces and weak supervision,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [96] A. Bosch, A. Zisserman, and X. Munoz, “Scene classification using a hybrid generative/discriminative approach,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, pp. 712–727, 2008.
- [97] X. Zhou, X. Zhuang, H. Tang, M. Hasegawa-Johnson, and T. S. Huang, “Novel gaussianized vector representation for improved natural scene categorization,” *Pattern Recognition Letters*, vol. 31, no. 8, pp. 702–708, 2010.
- [98] J. Wu and J. M. Rehg, “Where am I: Place instance and category recognition using spatial PACT,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [99] Z. Lu and H. H. Ip, “Image categorization by learning with context and consistency,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009.
- [100] C. Gu, J. Lim, P. Arbelaez, and J. Malik, “Recognition using regions,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009.
- [101] J. W. A. Elisseeff, “A kernel methods for multi-labelled classification,” in *Advances in Neural Information Processing Systems 14*, (Cambridge, MA), pp. 681–687, MIT Press, 2001.

- [102] S. Vijayanarasimhan and K. Grauman, “What’s it going to cost you?: predicting effort vs. informativeness for multi-label image annotations,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2262–2269, June 2009.
- [103] Z.-J. Zha, X.-S. Hua, T. Mei, J. Wang, G.-J. Qi, and Z. Wang, “Joint multi-label multi-instance learning for image classification,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2008.
- [104] S. L. Feng, R. Manmatha, and V. Lavrenko, “Multiple bernoulli relevance models for image and video annotation,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1002–1009, 2004.
- [105] T. Cox and M. Cox, *Multidimensional scaling*. London: Chapman & Hall, 1994.
- [106] J. Tenenbaum, V. de Silva, and J. C. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.
- [107] S. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
- [108] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall, “Learning a mahalanobis metric from equivalence constraints,” *Journal of Machine Learning Research*, vol. 6, pp. 937–965, 2005.
- [109] G. H. J. Goldberger, S. Roweis, and R. Salakhutdinov, “Neighbourhood components analysis,” in *Advances in Neural Information Processing Systems 17*, 2004.
- [110] S. Hoi, W. Liu, M. R. Lyu, and W.-Y. Ma, “Learning distance metrics with contextual constraints for image retrieval,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2072–2078, 2006.
- [111] K. Q. Weinberger, J. Blitzer, and L. K. Saul, “Distance metric learning for large margin nearest neighbor classification,” in *Advances in Neural Information Processing Systems 18*, 2005.

- [112] L. Yang, R. Jin, R. Sukthankar, and Y. Liu, “An efficient algorithm for local distance metric learning,” in *AAAI’06: Proceedings of the 21st AAAI conference on Artificial intelligence*, 2006.
- [113] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, “Information-theoretic metric learning,” in *Proceedings of the 24th International Conference on Machine Learning*, pp. 209–216, 2007.
- [114] L. Wu, S. Hoi, R. Jin, J. Zhu, and N. Yu, “Distance metric learning from uncertain side information with application to automated photo tagging,” in *MM ’09 Proceedings of the 17th ACM international conference on Multimedia*, 2009.
- [115] L. Wu, S. Hoi, and N. Yu, “Semantics-preserving bag-of-words models and applications,” *IEEE Transactions on Image Processing*, vol. 19, no. 7, pp. 1908 – 1920, 2010.
- [116] S. Hoi, W. Liu, and S.-F. Chang, “Semi-supervised distance metric learning for collaborative image retrieval,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [117] Z. Zha, T. Mei, M. Wang, Z. Wang, and X. Hua, “Robust distance metric learning with auxiliary knowledge,” in *Proceedings of the 21st international joint conference on Artificial intelligence*, 2009.
- [118] Y. Zhang and D.-Y. Yeung, “Transfer metric learning by learning task relationships,” in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010.
- [119] L. Si, R. Jin, S. C. H. Hoi, and M. R. Lyu, “Collaborative image retrieval via regularized metric learning,” *Multimedia Systems*, vol. 12, no. 1, pp. 34–44, 2006.
- [120] P. S. Bradley and O. L. Mangasarian, “Feature selection via concave minimization and support vector machines,” in *Proceedings of the 24th International Conference on Machine Learning*, 1998.

- [121] G.-X. Yuan, K.-W. Chang, C.-J. Hsieh, and C.-J. Lin, “A comparison of optimization methods and software for large-scale L1-regularized linear classification,” *Journal of Machine Learning Research*, vol. 11, no. 52, 2010.
- [122] C. Lampert, M. Blaschko, and T. Hofmann, “Beyond sliding windows: Object localization by efficient subwindow search,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [123] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results,” 2011.
- [124] M. J. Huiskes, B. Thomee, and M. S. Lew, “New trends and ideas in visual concept detection: The mir flickr retrieval evaluation initiative,” in *Proceedings of the 11th ACM SIGMM International Conference on Multimedia Information Retrieval*, pp. 527–536, 2010.
- [125] O. Chum and A. Zisserman, “An exemplar model for learning object classes,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.
- [126] J. Sivic and A. Zisserman, “Video Google: a text retrieval approach to object matching in videos,” in *Proceedings of IEEE International Conference on Computer Vision*, 2003.
- [127] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results,”
- [128] A. Vedaldi and B. Fulkerson, “VLFeat: An open and portable library of computer vision algorithms,” 2008.
- [129] R. Timofte and L. V. Gool, “Iterative nearest neighbors for classification and dimensionality reduction,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012.
- [130] C. Galleguillos, B. McFee, S. Belongie, and G. Lanckriet, “Multi-class object localization by combining local contextual interactions,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.

- [131] E. Shechtman and M. Irani, “Matching local self-similarities across images and videos,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.
- [132] M. B. Blaschko and C. H. Lampert, “Object localization with global and local context kernels,” in *Proceedings of British Machine Vision Conference*, 2009.
- [133] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman, “Multiple kernels for object detection,” in *Proceedings of IEEE International Conference on Computer Vision*, 2009.
- [134] C. Galleguillos, A. Rabinovich, and S. Belongie, “Object categorization using co-occurrence, location and appearance,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [135] G. Griffin and P. Perona, “Learning and using taxonomies for fast visual categorization,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [136] M. Marszalek and C. Schmid, “Constructing category hierarchies for visual recognition,” in *Proceedings of European Conference on Computer Vision*, 2008.
- [137] S. Fidler, M. Boben, and A. Leonardis, “A coarse-to-fine taxonomy of constellations for fast multi-class object detection,” in *Proceedings of European Conference on Computer Vision*, 2010.
- [138] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009.
- [139] R. Wang, S. Shan, X. Chen, and W. Gao, “Manifold-manifold distance with application to face recognition based on image set,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.
- [140] R. Wang and X. Chen, “Manifold discriminant analysis,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009.