VISUAL LEARNING OF PAIRWISE SIMILARITY AND RELATIVE ORDER RELATIONSHIPS

WANG FAQIANG

Ph.D

The Hong Kong Polytechnic University

This programme is jointly offered by The Hong Kong Polytechnic University and Harbin Institute of Technology

2018

The Hong Kong Polytechnic University

Department of Computing

Harbin Institute of Technology

School of Computer Science and Technology

Visual Learning of Pairwise Similarity and Relative Order Relationships

Faqiang WANG

A thesis submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

January 2018

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____ (Signed)

Faqiang WANG (Name of student)

Abstract

Many computer vision problems can be viewed as image pairwise relationship learning tasks. They learn a model to predict whether a given image pair belongs to a particular pairwise relationship. Among the existing image pairwise relationships, the similarity relationship and relative order relationship are the two most common pairwise relationships in the computer vision tasks. The similarity learning methods aim to learn a proper similarity measure, with which the similarity between images can be more effectively evaluated for classification. It is widely applied in the computer vision applications such as face verification, person re-identification, etc. Different from similarity, the relative order is a kind of antisymmetric relationship. The goal of relative order relationship learning is to learn a prediction model to predict the relative order relationship between two images. It is applied into the ranking task, e.g. relative attributes, and the regression task, e.g. age estimation and camera pose estimation.

Although the similarity and relative order relationships learning has been widely and successfully applied into many computer vision tasks, there are still some issues to be further studied. The similarity learning can be divided by two categories, i.e. Mahalanobis distance metric learning and deep similarity learning. For the Mahalanobis distance metric learning methods, it is important to investigate the connections between metric learning and kernel classification and explore how to utilize the kernel classification resources in the research and development of new metric learning methods. It's thus interesting to investigate whether we can unify the similarity learning methods into a general framework, which can provide a good platform for developing new similarity learning algorithms. As the single im-

age representation (SIR) and pairwise image representation (PIR) are commonly utilized in deep learning methods, it's necessary to design a similarity function by fusing the SIR and PIR to exploit their advantages. For the relative order relationship learning methods, how to learn the relative order relationship to improve the performances of both ranking and regression methods is a crucial issue. As in some applications, there are multiple relative order relationship to be learned, it's also important to build a network architecture for better tradeoff between the variances and connections of different relative order relationships.

In this thesis, we aim to develop the distance metric learning, deep similarity learning, single relative order relationship learning and multiple relative order relationship learning models for image pairs.

In Chapter 2, we generalize several state-of-the-art metric learning methods, such as large margin nearest neighbor (LMNN) and information theoretic metric learning (ITML), into a kernel classification framework. First, doublets and triplets are constructed from the training samples, and a family of degree-2 polynomial kernel functions are proposed for pairs of doublets or triplets. Then, a kernel classification framework is established to generalize many popular metric learning methods such as LMNN and ITML. The proposed framework can also suggest new metric learning methods, which can be efficiently implemented, interestingly, by using the standard support vector machine (SVM) solvers. Two novel metric learning methods, namely doublet-SVM and triplet-SVM, are then developed under the proposed framework. Experimental results show that doublet-SVM and triplet-SVM achieve competitive classification accuracies with state-of-the-art metric learning methods but with significantly less training time.

In Chapter 3, we formulate metric learning as a kernel classification problem with the positive semidefinite constraint, and solve it by iterated training of SVMs. The new formulation is easy to implement and efficient in training with the off-the-shelf SVM solvers. Two novel metric learning models, namely Positive-semidefinite Constrained Metric Learning (PCML) and Nonnegative-coefficient Constrained Metric Learning (NCML), are developed. Both PCML and NCML can guarantee the global optimality of their solutions. Experiments are conducted on handwritten digit classification, face verification and person re-identification to evaluate our methods. Compared with the state-of-the-art approaches, our methods can achieve comparable classification accuracy and are efficient in training.

In Chapter 4, we analyze the connection between the SIR and PIR based approaches, and propose a novel similarity measure by fusing SIR and PIR to exploit their advantages and boost the matching performance. A convolutional neural network (CNN) based similarity learning approach is proposed to jointly learn the SIR and PIR to optimize the proposed similarity measure. Our CNN is composed of a sub-network shared by SIR and PIR, and followed by two concurrent sub-networks to extract the SIRs of given images and the PIRs of given image pairs, respectively. To reduce the computational cost, we adopt a shallow PIR sub-network which consists of only one convolutional layer, one pooling layer and one fully-connected layer. Therefore, both SIR and PIR can be jointly learned for pursuing better matching accuracy with moderate computational cost. Furthermore, the matching scores learned with pairwise comparison and triplet comparison objectives can be combined to improve the matching performance. Experiments on the CUHK03, CUHK01 and VIPeR datasets show that the proposed method can achieve favorable

accuracy with modest training time.

In Chapter 5, we study to extend the deep siamese network from similarity learning to relative order relationship learning. We formulate the second-order image representation and the relative order relationship prediction function. Then we propose an extended deep siamese CNN based method with relative order loss, mean square error (MSE) loss and softmax loss to learn the relative order relationship. Furthermore, we find that the proposed method can also be applied to the regression task, *e.g.* age estimation, although it is not aimed at predicting pairwise relationship. We conduct the experiments on relative attribute ranking and age estimation tasks. The results show that the proposed method achieves the state-of-the-art performance, and outperforms the competing methods.

In Chapter 6, we study the multiple relative order relationship learning problem for the camera pose estimation task. We consider the this task as an Multi-Task Learning (MTL) problem, in which the learning of each pose component is regarded as a learning task, and we propose a camera pose estimation method based on deep siamese networks. In our proposed method, we use the second-order representation of images to learn the relative order relationship, and adopt the relative order loss and mean square error (MSE) loss to make the predicted poses and their relative order to be consistent with the ground-truth. To jointly learn multiple relative order relationships of the camera pose, we propose a deep siamese network which consists of two shared branches. Each branch consists of the spatial sub-network and regression sub-network, which learn the spatial feature and the regressors, respectively. The spatial sub-network is shared across all the learning tasks, and it can capture the generality between different pose components. As the regressors

of the pose components are different, the regression sub-network of different pose components are separated. So it can capture the specificity of each pose component. The experimental results show that our proposed method has lower prediction error than PoseNet [67] and the nearest neighbor approaches.

To sum up, we developed a kernel classification learning framework for metric learning, and proposed a series of distance metric learning models, *i.e.* doublet-SVM, triplet-SVM, PCML and NCML, based on the framework. We also proposed a new similarity measure by fusing the SIR and PIR, and build a CNN to jointly learn these representations and the similarity measure. On the basis of the deep siamese network, we proposed a single relative order relationship learning model and applied it into the ranking and regression tasks. For the camera pose estimation task, we extended the single relative order relationship learning model into multiple relative order relationship learning, and developed a CNN to model the variances and connections between different relationships. In the future, we will study the new image pairwise relationship indicators and the new learning models. We will also investigate the new applications of similarity and relative order relationships learning.

List of Publications

1. **Faqiang Wang**, Wangmeng Zuo, Lei Zhang, Deyu Meng and David Zhang, "A Kernel Classification Framework for Metric Learning", IEEE Transactions on Neural Networks and Learning Systems 26(9): 1950-1962, 2015.

2. **Faqiang Wang**, Wangmeng Zuo, Liang Lin, David Zhang and Lei Zhang, "Joint learning of single-image and cross-image representations for person re-identification", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

3. Wangmeng Zuo, **Faqiang Wang**, David Zhang, Liang Lin, Yuchi Huang, Deyu Meng and Lei Zhang, "Distance Metric Learning via Iterated Support Vector Machines", IEEE Transactions on Image Processing 26(10): 4937-4950, 2017.

4. **Faqiang Wang**, Wangmeng Zuo, David Zhang, Liang Lin and Lei Zhang, "From Similarity to Relative Order: Extend the Deep Siamese Network for Learning Image Pairwise Relationship", (Submitted).

Acknowledgements

First of all, I would like to thank my chief supervisor, Prof. Lei Zhang, for his supervision and support in my PhD period. It is my great honor to work with him. He gives me important and useful guidance and suggestions of my research work, and always encourage me to do the valuable research. I'm very appreciated for his supervision.

I want to thank my co-supervisors Prof. David Zhang and Prof. Wangmeng Zuo. Prof. Zhang and Prof. Zuo gave me very valuable instruction, support and help during my PhD study. They often discuss the research works with me, and give me very valuable comments and suggestions. They also helped me overcoming the difficulties in my research work. Their support and supervision play an important role in my PhD study. I'm very grateful to be one of Prof. Zhang and Prof. Zuo's students.

I'd like to express my thankfulness to Prof. Deyu Meng and Prof. Liang Lin for their help in my PhD study. I'm thankful to their great effort in the revisions of my publications and their valuable suggestions to improve my work.

I also want to thank my colleagues Shuhang Gu, Kunai Zhang, Jun Xu, Sijia Cai, Kede Ma, Hui Li, Jianrui Cai, Lingxiao Yang, Mu Li, Jin Xiao and Jinxing Li. Their help makes my PhD study rich and colorful.

At last, I would like to give my appreciation to my parents. Their endless love and understanding makes my PhD study and life smoothly and happily.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview of image pairwise relationship

Image pairwise relationship is widely existed in the computer vision applications, *e.g.* face verification, person re-identification, relative attributes, age estimation, image quality assessment, camera pose estimation, *etc*. So far, there are many computer vision tasks aim to learn the pairwise relationship of images. Given an image pair, they learn a function to predict whether it belongs to a particular pairwise relationship or not.

In the computer vision applications, there are many kinds of image pairwise relationships. The most commonly used are the similarity relationship and the relative order relationship. The similarity relationship learning is mainly used in the image classification or matching task, *e.g.* face verification [112] and person re-identification [35], while the relative order relationship learning is mainly applied in the ranking task, *e.g.* relative attributes [101]. In this section, we give an intro-

duction of similarity and relative order relationship learning methods, respectively.

1.1.1 Similarity relationship learning

The similarity relationship is one of the most important and commonly used pairwise relationship in computer vision. How to measure the similarity between two images is a fundamental issue in image classification. The similarity learning method learns the appropriate similarity or distance function $s : X \times X \rightarrow \mathbb{R}$ from the image space X, which makes the similarity of the similar pair higher and the similarity of dissimilar pair lower, or reduce the distance of similar pair while enlarge the distance of dissimilar pair [8]. Given an image pair $(\mathbf{x}, \mathbf{y})$, we can predict whether they are similar or dissimilar by computing their learned similarity $s(\mathbf{x}, \mathbf{y})$.

The desired similarity measure can vary a lot in different applications due to the underlying data structures and distributions, as well as the specificity of the learning tasks. Learning the similarity from the given training images has been an active topic in the past decade [8], and it can substantially improve the performance of many clustering (e.g., k-means) and classification [e.g., k-nearest neighbors (NNs)] methods. Similarity learning has been successfully adopted in many real world applications, e.g., face identification [48], face verification [112], image retrieval [10, 55], and activity recognition [126].

Among all kinds of similarity functions, the most commonly used is Mahalanobis distance, which is formulated as

$$s_{\text{Mahal}}(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T \mathbf{M} (\mathbf{x} - \mathbf{y}) \tag{1.1}$$

where $\mathbf{M}$ is a positive semidefinite (PSD) matrix. We factorize the matrix $\mathbf{M}$ as

$\mathbf{M} = \mathbf{L}^T\mathbf{L}$, and Eq. (1.1) can be rewritten as the following equivalent form

$$s_{\text{Mahal}}(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T\mathbf{L}^T\mathbf{L}(\mathbf{x} - \mathbf{y}) = \|\mathbf{L}(\mathbf{x} - \mathbf{y})\|_2^2 \tag{1.2}$$

From Eq. (1.2), we can see that the Mahalanobis distance between $\mathbf{x}$ and $\mathbf{y}$ is equivalent to the Euclidean distance between the transformed samples $\mathbf{x}' = \mathbf{L}\mathbf{x}$ and $\mathbf{y}' = \mathbf{L}\mathbf{y}$, where $\mathbf{L}$ is regarded as a linear transformation matrix.

When Mahalanobis distance metric learning is applied on computer vision, most approaches extract the features of images in advance, and then learn the distance metric based on the image features. Therefore, the performance of Mahalanobis distance metric learning is largely relied on the image feature extraction process.

In recent years, the deep learning approaches have been successfully applied in computer vision applications. The core of deep learning is to learn the nonlinear representation to adapt the complex distribution of images. Many works combine the learning of similarity and deep neural networks together, and propose a series of deep similarity learning methods. They can not only learn the similarity measure of two images, but also learn the deep representations from raw images. Most of deep similarity learning methods formulate the similarity of image pairs as follows

$$s_{\text{Deep}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left\|\mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_j)\right\|_2^2 \tag{1.3}$$

where $\mathbf{f}(\mathbf{x}_i)$ and $\mathbf{f}(\mathbf{x}_j)$ are the deep representations of images $\mathbf{x}_i$ and $\mathbf{x}_j$.

1.1.2 Relative order relationship learning

It's clear that similarity is the symmetric relationship since $s(\mathbf{x}, \mathbf{y}) = s(\mathbf{y}, \mathbf{x})$. However, in some applications, the relationship of two images is not symmetric. For

example, the relative attribute ranking task aims to predict the relative order of attributes, *e.g.* natural, perspective, *etc.*, of two images. In this task, the prediction function is defined as $r(\mathbf{x}, \mathbf{y})$, which denotes the relative attribute value of $\mathbf{x}$ compared with $\mathbf{y}$. Different from the similarity, this relationship is antisymmetric, since $r(\mathbf{x}, \mathbf{y}) = -r(\mathbf{y}, \mathbf{x})$. Here we call this relationship $r(\mathbf{x}, \mathbf{y})$ as the *relative order relationship*.

To predict the relative order of two images, most of the relative order relationship learning approaches formulate the prediction function as the indicator of relative order [101, 121, 146]. Given two images $\mathbf{x}$ and $\mathbf{y}$, their relative order prediction function is formulated as

$$r(\mathbf{x}, \mathbf{y}) = h(\mathbf{f}(\mathbf{x})) - h(\mathbf{f}(\mathbf{y})) \tag{1.4}$$

where $\mathbf{f}(\mathbf{x})$ and $\mathbf{f}(\mathbf{y})$ are the image representations of $\mathbf{x}$ and $\mathbf{y}$, which can be extracted by the hand-crafted feature extractor or learned by the deep neural networks. h is the prediction function of image representation.

1.2 Related works

1.2.1 Similarity learning

As we have mentioned in Section 1.1.1, the most frequently used similarity measures are Mahalanobis distance and deep representation based similarities. So we review the related works about the similarity learning methods based on these two measures, respectively.

Mahalanobis distance metric learning

Compared with nonconvex metric learning models [45, 125], convex formulation of metric learning [44, 119, 140, 142, 149] has drawn increasing attentions due to its desired properties such as global optimality. Most convex models can be formulated as SDP or quadratic SDP problems. Standard SDP solvers, however, are inefficient for metric learning, especially when the size of training samples is big or the feature dimension is high. Therefore, customized optimizer is developed for each specific metric learning model. For LMNN, Weinberger *et al.* developed an efficient solver based on sub-gradient descent and active set [139]. In ITML, Davis *et al.* [33] suggested an iterative Bregman projection algorithm. Iterative projected gradient descent method [63, 142] has been widely employed for metric learning but it requires an eigenvalue decomposition in each iteration. Other algorithms such as block-coordinate descent [104], smooth optimization [148], and Frank-Wolfe [149] have also been studied for metric learning. Unlike the customized algorithms, we formulate metric learning as a kernel classification problem with PSD constraint and solve it using the off-the-shelf SVM solvers, which can guarantee the global optimality and the PSD property of the learned $\mathbf{M}$, and is easy to implement and efficient in training.

Another line of work aims to develop metric learning algorithms by solving the Lagrange dual problems. Shen *et al.* derived the Lagrange dual of the exponential loss based metric learning model, and proposed a boosting-like approach, namely BoostMetric[118, 119]. MetricBoost [9] and FrobMetric [115, 117] were further proposed to improve BoostMetric. Liu and Vemuri incorporated two regularization terms in the duality for robust metric learning [85]. Note that BoostMetric [118,

119], MetricBoost [9], and FrobMetric [117] are proposed for metric learning with triplet constraints, whereas in many applications such as verification, only pairwise constraints are available in the training stage.

Studies have also given to connect SVM with metric learning [14, 36, 96]. Using SVM, Nguyen and Guo [96] formulated metric learning as a quadratic SDP, and adopted a projected gradient descent algorithm. They select the farthest neighbors for each sample to construct similar pairs, while we select the nearest neighbors in PCML and NCML. Moreover, the formulations and optimizers of our models are different from the model in [96]. Brunner *et al.* [14] proposed a pairwise SVM to learn a dissimilarity function. Their metric learning pairwise kernel is similar to that used in our models, but the PSD property is not considered in their model. Do *et al.* [36] analyzed the relation of LMNN and SVM, where LMNN is interpreted as the joint learning of multiple local SVM-like models. By studying SVM from a metric learning perspective, they presented an improved SVM for single sample classification. Different with [36], we explain metric learning as a SVM for sample pair classification with the PSD property, and propose two novel metric learning methods, *i.e.*, PCML and NCML, together with optimization algorithms.

Deep representation based similarity learning

Due to the achievement of deep CNNs in learning discriminative features from large-scale visual data, many methods have adopted the deep architecture to jointly learn the representation and the classifier [1, 28, 76, 112, 147]. Some of them focus on learning the single image representation (SIR) together with the similarity function. Schroff *et al.* proposed a FaceNet model for face verification [112], which

adopts a deep CNN to learn the Euclidean embedding per image by using the triplet comparison loss. Online triplet generation is also developed to gradually increase the difficulty of the triplets in training. Ding *et al.* proposed a deep SIR learning model based on relative distance comparison for person re-identification [35]. It first presents an effective triplet generation strategy to construct triplets, which contains one image with a matched image and a mismatched image. For each triplet, this model learns the SIR by maximizing the relative distance between the matched pair and the mismatched pair. Cheng *et al.* proposed a multi-channel part-based CNN method to jointly learn the features of the global and local human body images, and used an improved triplet loss for network training [29]. McLaughlin *et al.* developed a framework consisting of the CNN, recurrent neural network and temporal pooling layer to learn from the video sequence of a person [90]. It combines the pairwise contrastive loss and cross-entropy loss, which are designed for verification and identification tasks respectively, to train the whole network.

Despite learning SIR, some other methods are suggested to perform person re-identification based on pairwise image representation (PIR). Li *et al.* proposed a filter pairing neural network (FPNN) [76], which learns the PIRs by a patch matching layer followed by a maxout-grouping layer. In FPNN, the patch matching layer is used to model the displacement of each horizontal stripe in the images across views, the maxout-grouping layer improves the robustness of patch matching, and finally a softmax classifier is imposed on the learned PIR for person re-identification. The work in [1] shares the similar idea, but introduces a new layer to learn the pairwise image representation by computing the neighborhood difference between two input images. The work in [28] learns the PIR by formulating the person re-identification

task as a learning-to-rank problem. For each image pair, this model first stitchs its two images horizontally to form a holistic image, then feeds these images to a CNN to learn their representations. Finally the ranking loss is used to ensure that each sample is more similar to its positive matched image than its negative matched image. Liu *et al.* proposed a Matching CNN (M-CNN) architecture for human parsing [86], which learns the PIR of the image and a semantic region by a multi-layer cross image convolutional path to predict their matching confidence and displacements. Chen *et al.* learn multiple PIRs for an image pair with each corresponding to a local region, and combine the local and global similarities to match the image pairs [24].

1.2.2 Relative order relationship learning

The relative order relationship learning methods can be divided into two categories, *i.e.* single relative order relationship learning and multi relative order relationship learning. The single relative order relationship learning method is mainly applied in relative attribute ranking, and we will also apply it into age estimation in this thesis, while the general application of multi relative order relationship learning method is camera pose estimation. So we review the related works of these three tasks in this section.

Relative attribute ranking

The relative attribute ranking methods can be traced back to 2011, when Parikh and Grauman [101] learned a linear ranking function for relative attributes by an SVM-like problem. They extracted the gist [99] and Lab color histogram features of images, and then predicted the attribute values by the learned ranking function.

To utilize the correlation among different attributes, Chen *et al.* [27] proposed a multi-task learning method for relative attributes. Yang *et al.* [146] extended [101] to the deep CNN scheme to realize end-to-end learning of deep feature and ranking function.

These methods are based on global feature of images. Besides that, some methods learn the relative attribute based on image local parts. Sandeep *et al.* [109] used the part detector to detect the landmark points, then learn the significance coefficient and ranking model for relative attributes. Xiao and Lee [141] proposed a relative attribute ranking method which can discover the spatial extend of relative attributes without relying on the pre-trained detectors. Singh and Lee [121] developed a deep learning model for part-based relative attributes. It integrates the spatial transformer network [60] into CNN to localize and rank the relative attributes.

There are significant differences between these existing methods and our proposed method. First, we formulate the prediction function based on the second-order image representation, while the prediction functions of most existing methods are based on first-order image representation. Second, our proposed method adopt the MSE loss and softmax loss to employ the ground-truth label of each images, while most of the existing methods only use the relative orders of image pairs as the supervisory signal in training.

Age estimation

The earlier age estimation methods usually cast the age estimation as a classification problem by using the Active Appearance Models [32], which can integrate the texture and shape information of face images [73]. Fu and Huang [41] proposed a

multiple linear regression procedure on the discriminative aging manifold for age estimation. Guo *et al.* adopted some traditional regression methods, *e.g.* Support Vector Regression (SVR) [50, 52], Canonical Correlation Analysis (CCA) [53] and Partial Least Squares (PLS) [51], for age estimation.

Recently, some works about ordinal regression are proposed for age estimation. Chang *et al.* [19] proposed an ordinal hyperplanes ranker for age estimation. They used SVM to learn a series of functions to predict whether the age of input image is older than the various given ages, and combine the outputs of these functions to predict the age. Niu *et al.* [98] shared the similar idea of [19], but they used the CNN with cross-entropy loss to learn the prediction age.

Due to the advantage of deep CNN with large-scale data training, Rothe *et al.* [107] proposed an age estimation model based on the VGG-16 [120] architecture. They casted age estimation as a classification task and use the softmax loss to train the network. Antipov *et al.* [3] proposed a children-specialized age estimation method by integrate a particular network for children images.

Camera pose estimation

The camera pose estimation task, which is also known as metric localization, aims to estimate the camera position and orientation from the image. The earlier approaches usually uses the descriptor matching method to achieve this goal [30, 77, 78, 110, 122]. They build the 3D representation of scene images by Structure-from-Motion (SfM) technique, and then match the 2D query image and the 3D scene representation. However, this approach needs the 3D model which is relatively difficult to obtain.

In recent years, the deep learning approaches for camera pose estimation have been proposed. Kendall *et al.* [67] proposed the PoseNet method. It uses the deep CNN to learn the camera pose in an end-to-end manner. Melekhov *et al.* [91] proposed an end-to-end approach based Siamese CNN to estimate the relative pose of cameras. It makes the estimated relative pose and the ground-truth relative pose as close as possible.

1.3 Motivation

1.3.1 The framework of Mahalanobis distance metric learning

From the literature review, it can be seen that most of the existing Mahalanobis distance metric learning methods are formulated as convex programming problems of distance metric. They usually consists of a regularization term, a margin loss term and a series of doublet or triplet constraints. Besides, both of the Mahalanobis distance metric learning methods and most of the supervised learning methods learn the appropriate classifiers from the training data. So it is interesting to unify the distance metric learning methods into a general framework by casting it as a standard supervised learning problem. It is also highly expected that the new distance metric learning methods can be generated based on the framework.

1.3.2 The training efficiency of Mahalanobis distance metric learning

In recent years, the image data scale is rapidly increasing. So training on large-scale image data becomes a crucial task of computer vision, and the training efficiency is an important issue. For the Mahalanobis distance metric learning task, most of the existing methods formulate it as a convex positive semidefinite programming (SDP) problem. Many solvers have been proposed to obtain their global optimum solutions, *e.g.* projected gradient descent [142], iterative Bregman projection [33] and online solver [21, 69, 92, 116]. However, most of them are not efficient in training with large-scale image data. In this case, developing the efficient Mahalanobis distance metric learning method is fundamental and crucial in similarity learning.

1.3.3 The combination of single image and pairwise image representations in deep similarity learning

The existing deep learning methods usually learn the single image representation (SIR), which is the deep representation of a single image. Some other deep learning methods learn the pairwise image representation (PIR), which is the deep representation of an image pair, to predict whether the image pair is positive or negative. These two kinds of representations have their own advantage. The SIR is relatively efficient in matching, while the PIR is effective in capture the relationship of the image pair. Therefore, it's important to propose a new similarity measure and develop its learning algorithm to combine SIR and PIR, which takes the advantage of both representations.

1.3.4 The extension of deep siamese network for relative order relationship learning

The deep siamese network is widely and successfully used in deep similarity learning. As the relative order relationship is also a widely existing pairwise relationship, it is important to extend the deep siamese network from similarity learning to relative order relationship learning. In the ranking task and regression task, the image pairs satisfy the relative order relationship. Thus it's also necessary to investigate how to apply the relative order relationship learning method to the ranking and regression tasks.

1.3.5 The multiple relative order relationship learning

In some computer vision tasks such as camera pose estimation, we need to learn and predict the relative order relationships of multiple components. However, using the single relative order relationship learning method will loss the potential connections of different relative order relationships, and the traditional multiple regression methods haven't taken the relative order relationship of image pairs into consideration. Therefore, it is necessary to develop a multiple relative order relationship learning method which can model the generality and specificity of each relative order relationship learning tasks.

1.4 Contributions of the Thesis

This thesis focuses on proposing the learning models of similarity and relative order relationships of image pair. The framework of this thesis is illustrated in Fig. 1.1. The main contributions of this thesis are listed as follows.

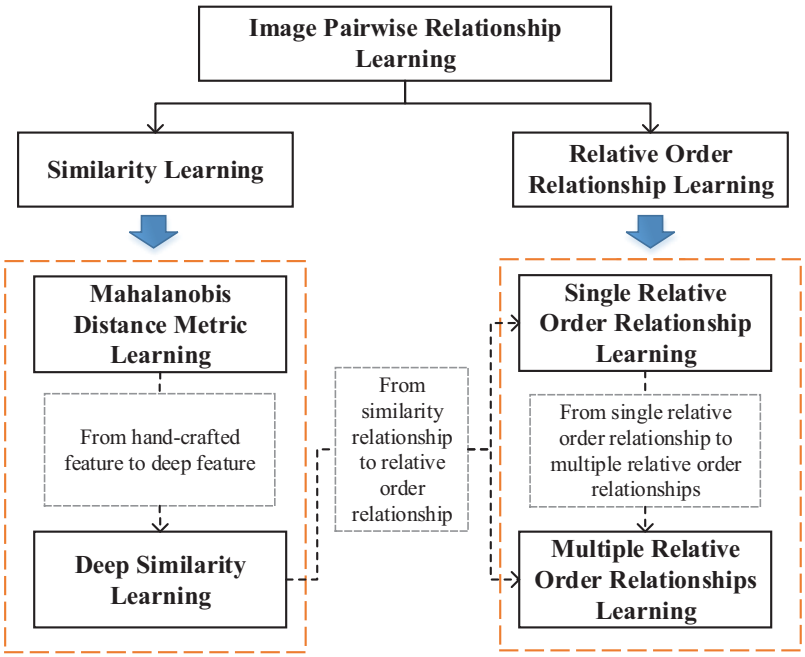


Figure 1.1 The framework of the thesis.

In Chapter 2, we develop a kernel classification framework for Mahalanobis distance metric learning by casting it as a standard supervised learning problem. This framework can unify many state-of-the-art distance metric learning methods, such as large margin nearest neighbor (LMNN) [138–140], information theoretic metric learning (ITML) [33], and logistic discriminative based metric learning (LDML) [48]. It also provides a platform for developing new distance metric learning meth-

ods. We developed two methods, i.e. doublet-SVM and triplet-SVM, based on the proposed framework. These two distance metric learning methods can be implemented using the off-the-shelf Support Vector Machine (SVM) solvers, and is efficient in training.

In Chapter 3, we proposed the Positive-semidefinite Constrained Metric Learning (PCML) method by formulating distance metric learning as a positive semidefinite programming problem, and it is solved by the iterated training of SVM and positive semidefinite projection. It can be easily implemented by the SVM solvers such as LibSVM [17]. By re-parameterizing the distance metric, we formulate the Nonnegative-coefficient Constrained Metric Learning (NCML) model, and it can be solved by iterated training of two SVMs. As PCML and NCML are convex models, their global optimal solutions can be obtained. The experiments on handwritten digit classification, face verification and person re-identification demonstrate that PCML and NCML can achieve favorable classification accuracy and efficient in training.

In Chapter 4, we formulate a new deep similarity measure by combining the SIR and PIR. To exploit the advantage of these two representations, we develop a framework to joint learn SIR and PIR with deep CNN. The network is trained based on pairwise comparison objective and triplet comparison objective, respectively. The pairwise comparison objective makes the deep similarities of the similar image pairs higher than a threshold, and those of the dissimilar image pairs lower than the threshold. The triplet comparison objective makes the deep similarity of each similar image pair is higher than that of the dissimilar pair. The similarities learned by the two objectives are combined to boost the performance. The experi-

mental results on person re-identification show that the proposed methods perform favorably compared with the state-of-the-art approaches.

In Chapter 5, we extend the deep siamese network from similarity learning to relative order relationship learning. We first design the second-order representation of images and formulate the relative order relationship prediction function. Then we design loss function of our extended deep siamese network which is composed of the relative order loss, mean square error (MSE) loss and softmax loss. They make the predicted relative order of each image pair to be consistent with the ground-truth and minimize the the error between the predicted value and the ground-truth label. We also demonstrate that the proposed method can be not only applied in the ranking task, but also applied in the regression task. The experiments on relative attributes and age estimation demonstrate the effectiveness of our proposed model in terms of prediction accuracy.

In Chapter 6, we study the multiple relative order relationship learning problem for the camera pose estimation task. We consider the this task as an Multi-Task Learning (MTL) problem, in which the learning of each pose component is regarded as a learning task. We aim to learn these tasks jointly to discover the potential connection of pose components. Therefore, we propose a camera pose estimation method based on deep siamese networks. Similar to Chapter 5, we also use the second-order representation of images to learn the relative order relationship, and adopt the relative order loss and mean square error (MSE) loss to make the predicted poses and their relative order to be consistent with the ground-truth. Different from Chapter 5, the multiple relative order relationships are jointly learned using one deep siamese network. Our deep siamese network consists of two branches

which share the same parameters. Each branch consists of the spatial sub-network and regression sub-network, which learn the spatial feature and the regressors, respectively. The spatial sub-network is shared across all the learning tasks, and it can capture the generality between different pose components. As the regressors of the pose components are different, the regression sub-network of different pose components are separated. So it can capture the specificity of each pose component. The experimental results show that our proposed method has lower prediction error than PoseNet [67] and the nearest neighbor approaches.

Chapter 2

A Kernel Classification Framework for Metric Learning

2.1 Introduction

How to measure the distance (or similarity/dissimilarity) between two data points is a fundamental issue in unsupervised and supervised pattern recognition. The desired distance metrics can vary a lot in different applications due to the underlying data structures and distributions, as well as the specificity of the learning tasks. Learning a distance metric from the given training examples has been an active topic in the past decade [44, 142], and it can improve much the performance of many clustering (e.g., k-means) and classification (e.g., k-nearest neighbors) methods. Distance metric learning has been successfully adopted in many real world applications, e.g., face identification [48], face verification [149], image retrieval [10, 55], and activity recognition [126].

Generally speaking, the goal of distance metric learning is to learn a distance metric from a given collection of similar/dissimilar samples by punishing the large distances between similar pairs and the small distances between dissimilar pairs. So far, numerous methods have been proposed to learn distance metrics, similarity metrics, and even nonlinear distance metrics. Among them, learning the Mahalanobis distance metrics for k-nearest neighbor classification has been receiving considerable research interests [33, 45, 48, 59, 105, 116, 118, 133, 138]. The problem of similarity learning has been studied as learning correlation metrics and cosine similarity metrics [5, 7, 20, 42, 95]. Several methods have been proposed for nonlinear distance metric learning [61, 64, 132]. Extensions of metric learning have also been investigated for multiple kernel learning [132], semi-supervised learning [55, 97, 135], multiple instance learning [49], and multi-task learning [100, 145], etc.

Despite that many metric learning approaches have been proposed, there are still some issues to be further studied. First, since metric learning learns a distance metric from the given training dataset, it is interesting to investigate whether we can recast metric learning as a standard supervised learning problem. Second, most existing metric learning methods are motivated from specific convex programming or probabilistic models, and it is interesting to investigate whether we can unify them into a general framework. Third, it is highly demanded that the unified framework can provide a good platform for developing new metric learning algorithms, which can be easily solved by standard and efficient learning tools.

With the above considerations, in this chapter we present a kernel classification framework to learn a Mahalanobis distance metric in the original feature s-

pace, which can unify many state-of-the-art metric learning methods, such as large margin nearest neighbor (LMNN) [138–140], information theoretic metric learning (ITML) [33], and logistic discriminative based metric learning (LDML) [48]. This framework allows us to easily develop new metric learning methods by using existing kernel classifiers such as the support vector machine (SVM) [128]. Under the proposed framework, we consequently present two novel metric learning methods, namely doublet-SVM and triplet-SVM, by modeling metric learning as an SVM problem, which can be efficiently solved by the existing SVM solvers like LibSVM [17].

The remainder of this chapter is organized as follows. Section 2.2 presents the proposed kernel classification framework for metric learning. Section 2.3 introduces the doublet-SVM and triplet-SVM methods. Section 2.4 presents the experimental results, and Section 2.5 summarizes the chapter.

2.2 A Kernel Classification based Metric Learning Framework

Current metric learning models largely depend on convex or non-convex optimization techniques, some of which are very inefficient to solve large-scale problems. In this section, we present a kernel classification framework which can unify many state-of-the-art metric learning methods. It also provides a good platform for developing new metric learning algorithms, which can be easily solved by using the efficient kernel classification tools. The connections between the proposed framework and LMNN, ITML, and LDML will also be discussed in detail.

2.2.1 Doublets and Triplets

Unlike conventional supervised learning problems, metric learning usually considers a set of constraints imposed on the doublets or triplets of training samples to learn the desired distance metric. It is very interesting and useful to evaluate whether metric learning can be casted as a conventional supervised learning problem. To build a connection between the two problems, we model metric learning as a supervised learning problem operating on a set of doublets or triplets, as described below.

Let $\mathcal{D} = \{(\mathbf{x}_i, y_i) | i = 1, 2, \cdots, n\}$ be a training dataset, where vector $\mathbf{x}_i \in \mathbb{R}^d$ represents the ith training sample, and scalar y_i represents the class label of $\mathbf{x}_i$. Any two samples extracted from $\mathcal{D}$ can form a doublet $\left(\mathbf{x}_i, \mathbf{x}_j\right)$, and we assign a label h to this doublet as follows: $h = -1$ if $y_i = y_j$ and $h = 1$ if $y_i \neq y_j$. For each training sample $\mathbf{x}_i$, we find from $\mathcal{D}$ its nearest similar neighbor, denoted by $\mathbf{x}_i^s$, and its nearest dissimilar neighbor, denoted by $\mathbf{x}_i^d$, and then construct 2 doublets $\{(\mathbf{x}_i, \mathbf{x}_i^s), (\mathbf{x}_i, \mathbf{x}_i^d)\}$. By combining all such doublets constructed from all training samples, we build a doublet set, denoted by $\{\mathbf{z}_1, \cdots, \mathbf{z}_{N_d}\}$, where $\mathbf{z}_l = (\mathbf{x}_{l,1}, \mathbf{x}_{l,2})$, $l = 1, 2, \cdots, N_d$. The label of doublet $\mathbf{z}_l$ is denoted by h_l. Note that doublet based constraints are used in ITML [33] and LDML [48], but the details of the construction of doublets are not given.

We call $\left(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k\right)$ a triplet if three samples $\mathbf{x}_i$, $\mathbf{x}_j$ and $\mathbf{x}_k$ are from $\mathcal{D}$ and their class labels satisfy $y_i = y_j \neq y_k$. We adopt the following strategy to construct a triplet set. For each training sample $\mathbf{x}_i$, we find its nearest neighbor $\mathbf{x}_i^s$ which have the same class label as $\mathbf{x}_i$, and the nearest neighbors $\mathbf{x}_i^d$ which have different class labels from $\mathbf{x}_i$. We can thus construct a triplet $(\mathbf{x}_i, \mathbf{x}_i^s, \mathbf{x}_i^d)$ for each sample $\mathbf{x}_i$. By

grouping all the triplets, we form a triplet set $\{\mathbf{t}_1, \cdots, \mathbf{t}_{N_t}\}$, where $\mathbf{t}_l = (\mathbf{x}_{l,1}, \mathbf{x}_{l,2}, \mathbf{x}_{l,3})$, $l = 1, 2, \cdots, N_t$. Note that for the convenience of expression, here we remove the super-script "s" and "d" from $\mathbf{x}_{l,2}$ and $\mathbf{x}_{l,3}$, respectively. A similar way to construct the triplets was used in LMNN [138] based on the k-nearest neighbors of each sample.

2.2.2 A Family of Degree-2 Polynomial Kernels

We then introduce a family of degree-2 polynomial kernel functions which can operate on pairs of the doublets or triplets defined above. With the introduced degree-2 polynomial kernels, distance metric learning can be readily formulated as a kernel classification problem.

Given two samples $\mathbf{x}_i$ and $\mathbf{x}_j$, we define the following function:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathrm{tr}(\mathbf{x}_i \mathbf{x}_i^T \mathbf{x}_j \mathbf{x}_j^T), \tag{2.1}$$

where $\mathrm{tr}(\bullet)$ represents the trace operator of a matrix. One can easily see that $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j)^2$ is a degree-2 polynomial kernel, and $K(\mathbf{x}_i, \mathbf{x}_j)$ satisfies the Mercer's condition [114].

The kernel function defined in (2.1) can be extended to a pair of doublets or triplets. Given two doublets $\mathbf{z}_i = (\mathbf{x}_{i,1}, \mathbf{x}_{i,2})$ and $\mathbf{z}_j = (\mathbf{x}_{j,1}, \mathbf{x}_{j,2})$, we define the corresponding degree-2 polynomial kernel as

$$\begin{aligned} K_D(\mathbf{z}_i, \mathbf{z}_j) &= \mathrm{tr}\Big((\mathbf{x}_{i,1} - \mathbf{x}_{i,2})(\mathbf{x}_{i,1} - \mathbf{x}_{i,2})^T (\mathbf{x}_{j,1} - \mathbf{x}_{j,2})(\mathbf{x}_{j,1} - \mathbf{x}_{j,2})^T\Big) \\ &= \Big[(\mathbf{x}_{i,1} - \mathbf{x}_{i,2})^T (\mathbf{x}_{j,1} - \mathbf{x}_{j,2})\Big]^2 \end{aligned}. \tag{2.2}$$

The kernel function in (2.2) defines an inner product of two doublets. With this kernel function, we can learn a decision function to tell whether the two samples of

a doublet have the same class label. In Section 2.2.3 we will show the connection between metric learning and kernel decision function learning.

Given two triplets $\mathbf{t}_i = (\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \mathbf{x}_{i,3})$ and $\mathbf{t}_j = (\mathbf{x}_{j,1}, \mathbf{x}_{j,2}, \mathbf{x}_{j,3})$, we define the corresponding degree-2 polynomial kernel as

$$K_T(\mathbf{t}_i, \mathbf{t}_j) = \mathrm{tr}\left(\mathbf{T}_i \mathbf{T}_j\right), \tag{2.3}$$

where

$$\mathbf{T}_i = (\mathbf{x}_{i,1} - \mathbf{x}_{i,3})(\mathbf{x}_{i,1} - \mathbf{x}_{i,3})^T - (\mathbf{x}_{i,1} - \mathbf{x}_{i,2})(\mathbf{x}_{i,1} - \mathbf{x}_{i,2})^T, \tag{2.4}$$

$$\mathbf{T}_j = \left(\mathbf{x}_{j,1} - \mathbf{x}_{j,3}\right)\left(\mathbf{x}_{j,1} - \mathbf{x}_{j,3}\right)^T - \left(\mathbf{x}_{j,1} - \mathbf{x}_{j,2}\right)\left(\mathbf{x}_{j,1} - \mathbf{x}_{j,2}\right)^T. \tag{2.5}$$

The kernel function in (2.3) defines an inner product of two triplets. With this kernel, we can learn a decision function based on the inequality constraints imposed on the triplets. In Section 2.2.3 we will also show how to deduce the Mahalanobis metric from the decision function.

2.2.3 Metric Learning via Kernel Methods

With the degree-2 polynomial kernels defined in Section 2.2.2, the task of metric learning can be easily solved by kernel methods. More specifically, we can use any kernel classification method to learn a kernel classifier with one of the following two forms

$$g_d(\mathbf{z}) = \mathrm{sgn}\left(\sum\nolimits_l h_l \alpha_l K_D(\mathbf{z}_l, \mathbf{z}) - b\right), \tag{2.6}$$

$$g_t(\mathbf{t}) = \mathrm{sgn}\left(\sum\nolimits_l \alpha_l K_T(\mathbf{t}_l, \mathbf{t})\right), \tag{2.7}$$

where $\mathbf{z}_l$, $l = 1, 2, \cdots, N$, is the doublet constructed from the training dataset, h_l is the label of $\mathbf{z}_l$, $\mathbf{t}_l$ is the triplet constructed from the training dataset, $\mathbf{z} = \left(\mathbf{x}_{(i)}, \mathbf{x}_{(j)}\right)$ is the test doublet, $\mathbf{t}$ is the test triplet, α_l is the weight, and b is the bias.

For doublet, we have

$$\begin{aligned}&\sum_l h_l\alpha_l \operatorname{tr}\left((\mathbf{x}_{l,1}-\mathbf{x}_{l,2})(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T(\mathbf{x}_{(i)}-\mathbf{x}_{(j)})(\mathbf{x}_{(i)}-\mathbf{x}_{(j)})^T\right)-b\\ =&(\mathbf{x}_{(i)}-\mathbf{x}_{(j)})^T\mathbf{M}(\mathbf{x}_{(i)}-\mathbf{x}_{(j)})-b,\end{aligned} \tag{2.8}$$

where

$$\mathbf{M}=\sum_l h_l\alpha_l(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T \tag{2.9}$$

is the matrix $\mathbf{M}$ of the Mahalanobis distance metric. Thus, the kernel decision function $g_d(\mathbf{z})$ can be used to determine whether $\mathbf{x}_{(i)}$ and $\mathbf{x}_{(j)}$ are similar or dissimilar to each other.

For triplet, the matrix $\mathbf{M}$ can be derived as follows.

Theorem 2.2.1 *Denote by* $\mathbf{t}=\left(\mathbf{x}_{(i)},\mathbf{x}_{(j)},\mathbf{x}_{(k)}\right)$ *the test triplet and by* $\mathbf{t}_l=(\mathbf{x}_{l,1},\mathbf{x}_{l,2},\mathbf{x}_{l,3})$ *the* l^{th} *triplet in the training set. Let* $\mathbf{T}_l=(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})^T-(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T$, *and* $\mathbf{T}=(\mathbf{x}_{(i)}-\mathbf{x}_{(k)})(\mathbf{x}_{(i)}-\mathbf{x}_{(k)})^T-\left(\mathbf{x}_{(i)}-\mathbf{x}_{(j)}\right)\left(\mathbf{x}_{(i)}-\mathbf{x}_{(j)}\right)^T$. *For the decision function defined in Eq. (2.7), if we re-parameterize the Mahalanobis distance metric matrix* $\mathbf{M}$ *as*

$$\begin{aligned}\mathbf{M}&=\sum_l\alpha_l\mathbf{T}_l\\ &=\sum_l\alpha_l\left[(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})^T-(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T\right]\end{aligned}, \tag{2.10}$$

then there is

$$\sum_l\alpha_l K_T(\mathbf{t}_l,\mathbf{t})=(\mathbf{x}_{(i)}-\mathbf{x}_{(k)})^T\mathbf{M}(\mathbf{x}_{(i)}-\mathbf{x}_{(k)})-\left(\mathbf{x}_{(i)}-\mathbf{x}_{(j)}\right)^T\mathbf{M}\left(\mathbf{x}_{(i)}-\mathbf{x}_{(j)}\right). \tag{2.11}$$

Proof On the basis of the definition of $K_T(\mathbf{t}_l, \mathbf{t})$ in (2.3), we have

$$\begin{aligned}
&\sum_l \alpha_l K_T(\mathbf{t}_l, \mathbf{t}) = \sum_l \alpha_l \operatorname{tr}(\mathbf{T}_l \mathbf{T}) \\
&= \sum_l \alpha_l \operatorname{tr}\left(\mathbf{T}_l \left((\mathbf{x}_{(i)} - \mathbf{x}_{(k)})(\mathbf{x}_{(i)} - \mathbf{x}_{(k)})^T - \left(\mathbf{x}_{(i)} - \mathbf{x}_{(j)}\right)\left(\mathbf{x}_{(i)} - \mathbf{x}_{(j)}\right)^T\right)^T\right) \\
&= \sum_l \alpha_l \operatorname{tr}\left(\mathbf{T}_l (\mathbf{x}_{(i)} - \mathbf{x}_{(k)})(\mathbf{x}_{(i)} - \mathbf{x}_{(k)})^T\right) - \sum_l \alpha_l \operatorname{tr}\left(\mathbf{T}_l \left(\mathbf{x}_{(i)} - \mathbf{x}_{(j)}\right)\left(\mathbf{x}_{(i)} - \mathbf{x}_{(j)}\right)^T\right) \cdot \\
&= (\mathbf{x}_{(i)} - \mathbf{x}_{(k)})^T \left(\sum_l \alpha_l \mathbf{T}_l\right)(\mathbf{x}_{(i)} - \mathbf{x}_{(k)}) - \left(\mathbf{x}_{(i)} - \mathbf{x}_{(j)}\right)^T \left(\sum_l \alpha_l \mathbf{T}_l\right)\left(\mathbf{x}_{(i)} - \mathbf{x}_{(j)}\right) \\
&= (\mathbf{x}_{(i)} - \mathbf{x}_{(k)})^T \mathbf{M}(\mathbf{x}_{(i)} - \mathbf{x}_{(k)}) - \left(\mathbf{x}_{(i)} - \mathbf{x}_{(j)}\right)^T \mathbf{M}\left(\mathbf{x}_{(i)} - \mathbf{x}_{(j)}\right)
\end{aligned} \tag{2.12}$$

End of proof.

Clearly, equations (2.6)∼(2.12) provide us a new perspective to view and understand the distance metric matrix $\mathbf{M}$ under a kernel classification framework. Meanwhile, this perspective provides us new approaches for learning distance metric, which can be much easier and more efficient than the previous metric learning approaches. In the following, we introduce two kernel classification methods for metric learning: regularized kernel SVM and kernel logistic regression. Note that by modifying the construction of doublet or triplet set, using different kernel classifier models, or adopting different optimization algorithms, other new metric learning algorithms can also be developed under the proposed framework.

Kernel SVM-like Model

Given the doublet or triplet training set, an SVM-like model can be proposed to learn the distance metric:

$$\begin{aligned}
&\min_{\mathbf{M},b,\boldsymbol{\xi}} \; r(\mathbf{M}) + \rho(\boldsymbol{\xi}) \\
&\text{s.t.} \quad f_l^{(d)}\left((\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2}), b, \xi_l\right) \geq 0 \quad \text{(doublet set)}, \\
&\qquad \text{or } f_l^{(t)}\left((\mathbf{x}_{l,1}-\mathbf{x}_{l,3})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,3}) - (\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2}), \xi_l\right) \geq 0 \quad \text{(triplet set)}, \\
&\qquad \xi_l \geq 0,
\end{aligned} \tag{2.13}$$

where $r(\mathbf{M})$ is the regularization term, $\rho(\boldsymbol{\xi})$ is the margin loss term, the constraint $f_l^{(d)}$ can be any linear function of $(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})$, b, and ξ_l, and the constraint $f_l^{(t)}$ can be any linear function of $(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})-(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})$ and ξ_l. To guarantee that (2.13) is convex, we can simply choose convex regularizer $r(\mathbf{M})$ and convex margin loss $\rho(\boldsymbol{\xi})$. By plugging (2.9) or (2.10) in the model in (2.13), we can employ the SVM and kernel methods to learn all α_l to obtain the matrix $\mathbf{M}$.

If we adopt the Frobenius norm to regularize $\mathbf{M}$ and the hinge loss penalty on ξ_l, the model in (2.13) would become the standard SVM. SVM and its variants have been extensively studied [94, 111, 128] and various algorithms have been proposed for large-scale SVM training [31, 127]. Thus, the SVM-like model in (2.13) can allow us to learn good metrics efficiently from large-scale training data.

Kernel logistic regression

Under the kernel logistic regression model (KLR) [65], we let $h_l = 1$ if the samples of doublet $\mathbf{z}_l$ belong to the same class and let $h_l = 0$ if the samples of it belong to different classes. Meanwhile, suppose that the label of a doublet $\mathbf{z}_l$ is unknown, and we can calculate the probability that $\mathbf{z}_l$'s label is 1 as follows:

$$P(p_l = 1|\mathbf{z}_l) = \frac{1}{1 + \exp\left(\sum_i \alpha_i K_D\left(\mathbf{z}_i, \mathbf{z}_l\right) + b\right)}. \tag{2.14}$$

The coefficient vector $\boldsymbol{\alpha}$ and the bias b can be obtained by maximizing the following log-likelihood function:

$$(\boldsymbol{\alpha}, b) = \arg\max_{\boldsymbol{\alpha}, b} \left\{ l(\boldsymbol{\alpha}, b) = \sum\nolimits_l h_l \ln P(p_l = 1|\mathbf{z}_l) + (1 - h_l) \ln P(p_l = 0|\mathbf{z}_l) \right\}. \tag{2.15}$$

KLR is a powerful probabilistic approach for classification. By modeling metric learning as a KLR problem, we can easily use the existing KLR algorithms to learn the desired metric. Moreover, the variants and improvements of KLR, e.g., sparse KLR [68], can also be used to develop new metric learning methods.

2.2.4 Connections with LMNN, ITML, and LDML

The proposed kernel classification framework provides a unified explanation of many state-of-the-art metric learning methods. In this subsection, we show that LMNN and ITML can be considered as certain SVM models, while LDML is an example of the kernel logistic regression model.

LMNN

LMNN [138] learns a distance metric that penalizes both large distances between samples with the same label and small distances between samples with different labels. LMNN is operated on a set of triplets $\left\{\left(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k\right)\right\}$, where $\mathbf{x}_i$ has the same label as $\mathbf{x}_j$ but has different label from $\mathbf{x}_k$. The optimization problem of LMNN can be stated as follows:

$$\begin{aligned} \min_{\mathbf{M},\xi_{ijk}} \ & \sum\nolimits_{i,j} \left(\mathbf{x}_i - \mathbf{x}_j\right)^T \mathbf{M} \left(\mathbf{x}_i - \mathbf{x}_j\right) + C \sum\nolimits_{i,j,k} \xi_{ijk} \\ \text{s.t.} \quad & (\mathbf{x}_i - \mathbf{x}_k)^T \mathbf{M} (\mathbf{x}_i - \mathbf{x}_k) - \left(\mathbf{x}_i - \mathbf{x}_j\right)^T \mathbf{M} \left(\mathbf{x}_i - \mathbf{x}_j\right) \geq 1 - \xi_{ijk}, \\ & \xi_{ijl} \geq 0, \\ & \mathbf{M} \succcurlyeq 0. \end{aligned} \tag{2.16}$$

Since $\mathbf{M}$ is required to be positive semidefinite in LMNN, we introduce the following indicator function:

$$\iota_{\succcurlyeq} (\mathbf{M}) = \begin{cases} 0, & \text{if } \mathbf{M} \succcurlyeq 0, \\ +\infty, & \text{otherwise,} \end{cases} \tag{2.17}$$

and choose the following regularizer and margin loss:

$$r_{\text{LMNN}} (\mathbf{M}) = \sum\nolimits_{i,j} \left(\mathbf{x}_i - \mathbf{x}_j\right)^T \mathbf{M} \left(\mathbf{x}_i - \mathbf{x}_j\right) + \iota_{\succcurlyeq} (\mathbf{M}), \tag{2.18}$$

$$\rho_{\text{LMNN}} (\boldsymbol{\xi}) = C \sum\nolimits_{i,j,k} \xi_{ijk}. \tag{2.19}$$

Then we can define the following SVM-like model on the same triplet set:

$$\begin{aligned} \min_{\mathbf{M},\boldsymbol{\xi}} \ & r_{\text{LMNN}} (\mathbf{M}) + \rho_{\text{LMNN}} (\boldsymbol{\xi}) \\ \text{s.t.} \quad & (\mathbf{x}_i - \mathbf{x}_k)^T \mathbf{M} (\mathbf{x}_i - \mathbf{x}_k) - \left(\mathbf{x}_i - \mathbf{x}_j\right)^T \mathbf{M} \left(\mathbf{x}_i - \mathbf{x}_j\right) \geq 1 - \xi_{ijk}, \\ & \xi_{ijk} \geq 0. \end{aligned} \tag{2.20}$$

It is obvious that the SVM-like model in (2.20) is equivalent to the LMNN model in (2.16).

ITML

ITML [33] is operated on a set of doublets $\left\{\left(\mathbf{x}_i, \mathbf{x}_j\right)\right\}$ by solving the following minimization problem:

$$\begin{aligned} \min_{\mathbf{M},\boldsymbol{\xi}} \; & D_{ld}\left(\mathbf{M}, \mathbf{M}_0\right) + \gamma \cdot D_{ld}\left(\operatorname{diag}\left(\boldsymbol{\xi}\right), \operatorname{diag}\left(\boldsymbol{\xi}_0\right)\right) \\ \text{s.t. } & \left(\mathbf{x}_i - \mathbf{x}_j\right)^T \mathbf{M} \left(\mathbf{x}_i - \mathbf{x}_j\right) \leq \xi_{u(i,j)} \;\; (i,j) \in \mathcal{S}, \\ & \left(\mathbf{x}_i - \mathbf{x}_j\right)^T \mathbf{M} \left(\mathbf{x}_i - \mathbf{x}_j\right) \geq \xi_{l(i,j)} \;\; (i,j) \in \mathcal{D}, \\ & \mathbf{M} \succcurlyeq 0, \end{aligned} \tag{2.21}$$

where $\mathbf{M}_0$ is the given prior of the metric matrix, $\boldsymbol{\xi}_0$ is the given prior on $\boldsymbol{\xi}$, $\mathcal{S}$ is the set of doublets where $\mathbf{x}_i$ and $\mathbf{x}_j$ have the same label, $\mathcal{D}$ is the set of doublets where $\mathbf{x}_i$ and $\mathbf{x}_j$ have different labels, and $D_{ld}\left(\cdot, \cdot\right)$ is the LogDet divergence of two matrices defined as:

$$D_{ld}\left(\mathbf{M}, \mathbf{M}_0\right) = \operatorname{tr}\left(\mathbf{M}\mathbf{M}_0^{-1}\right) - \log\det\left(\mathbf{M}\mathbf{M}_0^{-1}\right) - n. \tag{2.22}$$

Davis *et al.* also proposed an iterative Bregman projection algorithm for ITML to avoid the positive semidefinite projection of the distance metric matrix $\mathbf{M}$ [33].

By introducing the following regularizer and margin loss:

$$r_{\text{ITML}}\left(\mathbf{M}\right) = D_{ld}\left(\mathbf{M}, \mathbf{M}_0\right) + \iota_{\succcurlyeq}\left(\mathbf{M}\right), \tag{2.23}$$

$$\rho_{\text{ITML}}\left(\boldsymbol{\xi}\right) = \gamma \cdot D_{ld}\left(\operatorname{diag}\left(\boldsymbol{\xi}\right), \operatorname{diag}\left(\boldsymbol{\xi}_0\right)\right), \tag{2.24}$$

we can then define the following SVM-like model on the same doublet set:

$$\begin{aligned} \min_{\mathbf{M},\boldsymbol{\xi}} \; & r_{\mathrm{ITML}}\left(\mathbf{M}\right) + \rho_{\mathrm{ITML}}\left(\boldsymbol{\xi}\right) \\ \text{s.t.} \quad & (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j) \leq \xi_{u(i,j)} \;\; (i,j) \in \mathcal{S}, \\ & (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j) \geq \xi_{l(i,j)} \;\; (i,j) \in \mathcal{D}, \\ & \xi_{ij} \geq 0, \end{aligned} \tag{2.25}$$

where $\mathbf{z}_{ij} = \left(\mathbf{x}_i, \mathbf{x}_j\right)$. One can easily see that the SVM-like model in (2.25) is equivalent to the ITML model in (2.21).

LDML

LDML [48] is a logistic discriminant based metric learning approach based on a set of doublets. Given a doublet $\mathbf{z}_l = \left(\mathbf{x}_{l(i)}, \mathbf{x}_{l(j)}\right)$ and its label h_l, LDML defines the probability that $y_{l(i)} = y_{l(j)}$ as follows:

$$\begin{aligned} p_l &= P(y_{l(i)} = y_{l(j)} | \mathbf{x}_{l(i)}, \mathbf{x}_{l(j)}, \mathbf{M}, b) \\ &= \sigma(b - d_{\mathbf{M}}(\mathbf{x}_{l(i)}, \mathbf{x}_{l(j)})), \end{aligned} \tag{2.26}$$

where $\sigma(z)$ is the sigmoid function, b is the bias, and $d_{\mathbf{M}}(\mathbf{x}_{l(i)}, \mathbf{x}_{l(j)}) = (\mathbf{x}_{l(i)} - \mathbf{x}_{l(j)})^T \mathbf{M} (\mathbf{x}_{l(i)} - \mathbf{x}_{l(j)})$. With the p_l defined in (2.26), LDML learns $\mathbf{M}$ and b by maximizing the following log-likelihood:

$$\max_{\mathbf{M},b} \left\{ l(\mathbf{M}, b) = \sum\nolimits_l h_l \ln p_l + (1 - h_l) \ln(1 - p_l) \right\}. \tag{2.27}$$

Note that $\mathbf{M}$ is not constrained to be positive semidefinite in LDML.

With the same doublet set, let $\boldsymbol{\alpha}$ be the solution obtained by the kernel logistic model in (2.15), and $\mathbf{M}$ be the solution of LDML in (2.27). It is easy to see that:

$$\mathbf{M} = \sum\nolimits_l \alpha_l (\mathbf{x}_{l(i)} - \mathbf{x}_{l(j)})(\mathbf{x}_{l(i)} - \mathbf{x}_{l(j)})^T. \tag{2.28}$$

Thus, LDML is equivalent to kernel logistic regression under the proposed kernel classification framework.

2.3 Metric Learning via SVM

The kernel classification framework proposed in Section 2.2 can not only generalize the existing metric learning models (as shown in Section 2.2.4), but also be able to suggest new metric learning models. Actually, for both ITML and LMNN, the positive semidefinite constraint is imposed on $\mathbf{M}$ to guarantee that the learned distance metric is a Mahalanobis metric, which makes the models unable to be solved using the efficient kernel learning toolbox. In this section, a two-step greedy strategy is adopted for metric learning. We first neglect the positive semidefinite constraint and use the SVM toolbox to learn a preliminary matrix $\mathbf{M}$, and then map $\mathbf{M}$ onto the space of positive semidefinite matrices. The projected sub-gradient algorithm used in many metric learning methods [140] share similar spirits with the two-step greedy strategy. As examples, we present two novel metric learning methods, namely doublet-SVM and triplet-SVM, based on the proposed framework. Like in conventional SVM, we adopt the Frobenius norm to regularize $\mathbf{M}$ and employ the hinge loss penalty, and hence the doublet-SVM and triplet-SVM can be efficiently solved by using the standard SVM toolbox.

2.3.1 Doublet-SVM

In doublet-SVM, we set the Frobenius norm regularizer as $r_{\mathrm{SVM}}(\mathbf{M}) = \frac{1}{2}\|\mathbf{M}\|_F^2$, and set $\rho_{\mathrm{SVM}}(\boldsymbol{\xi}) = C\sum_l \xi_l$ as the margin loss term, resulting in the following model:

$$\begin{aligned} \min_{\mathbf{M},b,\boldsymbol{\xi}} \ & \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum\nolimits_l \xi_l \\ \text{s.t.} \ & h_l\left((\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})+b\right) \geq 1-\xi_l, \\ & \xi_l \geq 0, \ \forall l, \end{aligned} \tag{2.29}$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

We solve this problem by its Lagrangian dual problem. According to the original problem of doublet-SVM in (2.29), its Lagrangian can be defined as follows:

$$\begin{aligned} L(\mathbf{M},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta}) &= \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum\nolimits_l \xi_l - \sum\nolimits_l \beta_l\xi_l \\ &- \sum\nolimits_l \alpha_l\left[h_l\left((\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})+b\right)-1+\xi_l\right], \end{aligned} \tag{2.30}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are the Lagrange multipliers which satisfy $\alpha_l \geq 0$ and $\beta_l \geq 0, \ \forall l$. To convert the original problem to its dual, we let the derivative of the Lagrangian with respect to $\mathbf{M}$, b and $\boldsymbol{\xi}$ to be $\mathbf{0}$:

$$\frac{\partial L(\mathbf{M},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta})}{\partial \mathbf{M}} = \mathbf{0} \Rightarrow \mathbf{M} - \sum\nolimits_l \alpha_l h_l (\mathbf{x}_{l,1}-\mathbf{x}_{l,2})(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T = \mathbf{0}, \tag{2.31}$$

$$\frac{\partial L(\mathbf{M},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta})}{\partial b} = 0 \Rightarrow \sum\nolimits_l \alpha_l h_l = 0, \tag{2.32}$$

$$\frac{\partial L(\mathbf{M},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta})}{\partial \xi_l} = 0 \Rightarrow C-\alpha_l-\beta_l = 0 \Rightarrow 0 < \alpha_l < C, \ \forall l. \tag{2.33}$$

Equation (2.31) implies the relationship between $\mathbf{M}$ and $\boldsymbol{\alpha}$ as follows:

$$\mathbf{M} = \sum\nolimits_l \alpha_l h_l (\mathbf{x}_{l,1}-\mathbf{x}_{l,2})(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T. \tag{2.34}$$

Substituting (2.31)∼(2.33) back into the Lagrangian, we get the Lagrange dual problem of doublet-SVM as follows:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & -\frac{1}{2}\sum\nolimits_{i,j}\alpha_i\alpha_j h_i h_j K_D\left(\mathbf{z}_i,\mathbf{z}_j\right)+\sum\nolimits_i \alpha_i \\ \text{s.t.} \quad & 0\le\alpha_l\le C, \quad \forall l, \\ & \sum\nolimits_l \alpha_l h_l = 0. \end{aligned} \tag{2.35}$$

which can be easily solved by many existing SVM solvers such as LibSVM [17].

2.3.2 Triplet-SVM

In triplet-SVM, we also choose $r_{\mathrm{SVM}}(\mathbf{M}) = \frac{1}{2}\|\mathbf{M}\|_F^2$ as the regularization term, and choose $\rho_{\mathrm{SVM}}(\boldsymbol{\xi}) = C\sum_l \xi_l$ as the margin loss term. Since the triplets do not have label information, we choose the linear inequality constraints which are adopted in LMNN, resulting in the following triplet-SVM model:

$$\begin{aligned} \min_{\mathbf{M},\boldsymbol{\xi}} \quad & \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum\nolimits_l \xi_l \\ \text{s.t.} \quad & (\mathbf{x}_{l,1}-\mathbf{x}_{l,3})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,3}) - (\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2}) \ge 1-\xi_l, \\ & \xi_l \ge 0, \ \forall l. \end{aligned} \tag{2.36}$$

Actually, the proposed triplet-SVM can be regarded as a one-class SVM model, and the formulation of triplet-SVM is similar to the one-class SVM in [111]. We also attempt to solve it by its Lagrangian dual problem. According to the original problem of triplet-SVM in (2.36), its Lagrangian can be defined as follows:

$$\begin{aligned} L(\mathbf{M},\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta}) = & \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum\nolimits_l \xi_l \\ & - \sum\nolimits_l \alpha_l\left[(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,3}) - (\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^T\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})\right] \\ & + \sum\nolimits_l \alpha_l - \sum\nolimits_l \alpha_l\xi_l - \sum\nolimits_l \beta_l\xi_l, \end{aligned} \tag{2.37}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are the Lagrange multipliers, which satisfy $\alpha_l \geq 0$ and $\beta_l \geq 0,\ \forall l$. To convert the original problem to its dual, we let the derivative of the Lagrangian with respect to $\mathbf{M}$ and $\boldsymbol{\xi}$ to be $\mathbf{0}$:

$$\begin{aligned} &\frac{\partial L\left(\mathbf{M}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}\right)}{\partial \mathbf{M}} = \mathbf{0} \Rightarrow \\ &\mathbf{M} - \sum\nolimits_l \alpha_l \left[\left(\mathbf{x}_{l,1} - \mathbf{x}_{l,3}\right) \left(\mathbf{x}_{l,1} - \mathbf{x}_{l,3}\right)^T - \left(\mathbf{x}_{l,1} - \mathbf{x}_{l,2}\right) \left(\mathbf{x}_{l,1} - \mathbf{x}_{l,2}\right)^T \right] = \mathbf{0}, \end{aligned} \tag{2.38}$$

$$\frac{\partial L\left(\mathbf{M}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}\right)}{\partial \xi_l} = 0 \Rightarrow C - \alpha_l - \beta_l = 0 \Rightarrow 0 < \alpha_l < C,\ \forall l. \tag{2.39}$$

Equation (2.38) implies the relationship between $\mathbf{M}$ and $\boldsymbol{\alpha}$ as follows:

$$\mathbf{M} = \sum\nolimits_l \alpha_l \left[\left(\mathbf{x}_{l,1} - \mathbf{x}_{l,3}\right) \left(\mathbf{x}_{l,1} - \mathbf{x}_{l,3}\right)^T - \left(\mathbf{x}_{l,1} - \mathbf{x}_{l,2}\right) \left(\mathbf{x}_{l,1} - \mathbf{x}_{l,2}\right)^T \right]. \tag{2.40}$$

Substituting (2.38) and (2.39) back into the Lagrangian, we get the Lagrange dual problem of triplet-SVM as follows:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & -\frac{1}{2} \sum\nolimits_{i,j} \alpha_i \alpha_j K_T\left(\mathbf{t}_i, \mathbf{t}_j\right) + \sum\nolimits_i \alpha_i \\ \text{s.t.} \quad & 0 \leq \alpha_l \leq C, \quad \forall l. \end{aligned} \tag{2.41}$$

which can also be efficiently solved by existing SVM solvers [17].

2.3.3 Discussions

The matrix $\mathbf{M}$ learned by doublet-SVM and triplet-SVM may not be positive semidefinite. To learn a Mahalanobis distance metric, which requires $\mathbf{M}$ to be positive semidefinite, we can compute the singular value decomposition of $\mathbf{M} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}$, where $\boldsymbol{\Lambda}$ is the diagonal matrix of eigenvalues, and then preserve only the positive eigenvalues in $\boldsymbol{\Lambda}$ to form another diagonal matrix $\boldsymbol{\Lambda}_+$. Finally, we let $\mathbf{M}_+ = \mathbf{U}\boldsymbol{\Lambda}_+\mathbf{V}$ be the Mahalanobis metric matrix.

The proposed doublet-SVM and triplet-SVM are easy to implement since the use of Frobenius norm regularizer and hinge loss penalty allows us to readily employ the available SVM toolbox to solve them. A number of efficient algorithms, e.g., sequential minimal optimization [103], have been proposed for SVM training, making doublet-SVM and triplet-SVM very efficient to optimize. Moreover, using the large-scale SVM training algorithms [12, 31, 37, 127], we can easily extend doublet-SVM and triplet-SVM to deal with large-scale metric learning problems.

A number of kernel methods have been proposed for supervised learning [114]. With the proposed framework, we can easily couple them with the degree-2 polynomial kernel to develop new metric learning approaches. Semi-supervised, multiple instance, and multi-task metric learning have been investigated in [4, 49, 55, 100]. Fortunately, the proposed kernel classification framework can also allow us to develop such kind of metric learning approaches based on the recent progress of kernel methods for semi-supervised, multiple instance, and multitask learning [2, 6, 38, 43]. Taking semi-supervised metric learning as an example, based on Laplacian SVM [6] and doublet-SVM, we can readily extend the kernel classification framework for semi-supervised metric learning.

Let $\{(\mathbf{z}_i, h_i)\}_{i=1}^{L}$ be a set of L labeled doublets, and $\{\mathbf{z}_i\}_{i=L+1}^{i=L+U}$ be a set of U unlabeled doublets. With the degree-2 polynomial kernel $K_D\left(\mathbf{z}_i, \mathbf{z}_j\right)$, the decision function can be expressed as:

$$f(\mathbf{z}) = \sum\nolimits_{i=1}^{L} \alpha_i h_i K_D(\mathbf{z}, \mathbf{z}_i) + \sum\nolimits_{i=L+1}^{L+U} \alpha_i K_D(\mathbf{z}, \mathbf{z}_i), \tag{2.42}$$

where $\mathbf{z} = \left(\mathbf{x}_{(j)}, \mathbf{x}_{(k)}\right)$, $\mathbf{z}_i = (\mathbf{x}_{i,1}, \mathbf{x}_{i,2})$. Analogous to Laplacian SVM, one can com-

bine the Frobenius norm regularizer and the manifold regularizer:

$$r(f) = \gamma_A \|f\|_K^2 + \frac{\gamma_I}{(U+L)^2}\mathbf{f}^T(\mathbf{D}-\mathbf{W})\mathbf{f}, \tag{2.43}$$

where $\|f\|_K$ denotes the norm in the kernel feature space, $f_i = \sum_{j=1}^{L+U} \alpha_j K_D\left(\mathbf{z}_i, \mathbf{z}_j\right)$, $\mathbf{f} = (f_1, \cdots, f_{L+U})^T$, $\mathbf{W}$ is introduced to model the adjacency between doublets with $W_{ij} = \exp\left(\left(-K_D(\mathbf{z}_i, \mathbf{z}_i) + 2K_D\left(\mathbf{z}_i, \mathbf{z}_j\right) - K_D\left(\mathbf{z}_j, \mathbf{z}_j\right)\right)/4t\right)$ (t is the constant parameter), where $\mathbf{D}$ is a diagonal matrix with $D_{ii} = \sum_{j=1}^{L+U} W_{ij}$. By using hinge loss as the margin loss term $\rho(\boldsymbol{\xi})$ and introducing the Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{W}$, semi-supervised metric learning can then be formulated as Laplacian SVM:

$$\begin{aligned} \min_f \ & \gamma_A \|f\|_K^2 + \frac{\gamma_I}{(u+l)^2}\mathbf{f}^T\mathbf{L}\mathbf{f} + C\sum\nolimits_i \xi_i \\ \text{s.t.} \quad & h_i\left(f(\mathbf{z}_i) + b\right) \geq 1 - \xi_i, \\ & \xi_i \geq 0,\ i = 1, \cdots, L. \end{aligned} \tag{2.44}$$

The Lagrange dual problem of Laplacian SVM can be represented as

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \ & \sum\nolimits_{i=1}^{L} \alpha_i - \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{Q}\boldsymbol{\alpha} \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C,\ i = 1, \cdots, L, \\ & \sum\nolimits_{i=1}^{L} \alpha_i h_i = 0, \end{aligned} \tag{2.45}$$

where $\mathbf{Q} = \mathbf{YJK}\left(2\gamma_A\mathbf{I} + 2\frac{\gamma_I}{(L+U)^2}\mathbf{LK}\right)^{-1}\mathbf{J}^T\mathbf{Y}$, $\mathbf{K}$ is the kernel Gram matrix with $K_{ij} = K_D\left(\mathbf{z}_i, \mathbf{z}_j\right)$, $\mathbf{Y}$ is an $(L+U)\times(L+U)$ diagonal matrix with $Y_{ii} = h_i$ when $i \leq L$ and 0 otherwise, J is an $(L+U)\times(L+U)$ diagonal matrix with $J_{ii} = 1$ when $i \leq L$ and 0 otherwise.

The above Laplacian SVM problem can be solved by the standard SVM solver [6]. Given the optimal solution on $\boldsymbol{\alpha}$, the positive semidefinite matrix $\mathbf{M}$ can be

obtained by

$$\begin{aligned}\mathbf{M} = &\sum_{i=1}^{L} \alpha_i h_i \left(\mathbf{x}_{i,1} - \mathbf{x}_{i,2}\right)\left(\mathbf{x}_{i,1} - \mathbf{x}_{i,2}\right)^T \\ &+ \sum_{i=L+1}^{L+U} \alpha_i \left(\mathbf{x}_{i,1} - \mathbf{x}_{i,2}\right)\left(\mathbf{x}_{i,1} - \mathbf{x}_{i,2}\right)^T.\end{aligned} \tag{2.46}$$

Similarly, one can extend the kernel classification framework for multiple instance and multi-task metric learning based on the multiple instance and multi-task kernel learning methods [2, 38, 43].

2.4 Experimental Results

In the experiments, we evaluate the proposed doublet-SVM and triplet-SVM for k-NN classification with $k = 1$ on the handwritten digit classification and person re-identification tasks. We implemented doublet-SVM and triplet-SVM based on the popular SVM toolbox LibSVM[1]. In the training stage, the doublet set used in doublet-SVM is exactly the same as that used in ITML, but is different from that used in the other models. The triplet set used in triplet-SVM is different from that used in LMNN. The reason that we do not use the same doublet or triplet sets as the other methods is that the released codes of these approaches either include inherent default doublet or triplet sets, or dynamically tune the doublet or triplet sets during the training stage.

2.4.1 Handwritten Digit Recognition

We perform the experiments on three widely used large scale handwritten digit sets, *i.e.*, MNIST, USPS, and Semeion, to evaluate the performances of doublet-SVM

[1] `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

and triplet-SVM. On the MNIST and USPS datasets, we use the defined training and test sets to train the models and calculate the classification error rates. On the Semeion datasets, we use 10-fold cross validation to evaluate the metric learning methods, and the error rate and training time are obtained by averaging over the 10 runs. Table 2.1 summarizes the basic information of the three handwritten digit datasets.

Table 2.1 The handwritten digits datasets used in the experiments

Dataset	**# of training samples**	**# of test samples**	**Feature dimension**	**PCA dimension**	**# of classes**
MNIST	60,000	10,000	784	100	10
USPS	7,291	2,007	256	100	10
Semeion	1,434	159	256	100	10

Table 2.2 The classification error rates (%) and average ranks of the competing methods on the handwritten digit datasets

Dataset	**Doublet-SVM**	**Triplet-SVM**	**NCA**	**LMNN**	**ITML**	**MCML**	**LDML**
MNIST	3.19	2.92	5.46	**2.28**	2.89	-	6.05
USPS	5.03	**4.93**	5.68	5.38	6.63	5.08	8.77
Semeion	5.21	**4.46**	8.60	6.09	5.71	11.23	11.98
Average Rank	*2.67*	***1.67***	*5.00*	*3.00*	*3.67*	-	*6.67*

As the dimensions of digit images are relatively high, PCA is utilized to reduce the feature dimension. The metric learning models are trained in the PCA subspace.

Both doublet-SVM and triplet-SVM involve the parameter C. Using the USPS dataset as an example, we analyze the sensitivity of classification error rate to

this parameter. Fig. 2.1 shows the curves of classification error rate versus C for doublet-SVM and triplet-SVM. One can see that the error rate is insensitive to C in a wide range, but it jumps when C is no less than 10^3 for doublet-SVM and no less than 10^0 for triplet-SVM. Thus, we set $C = 10^{-2}$ for doublet-SVM and $C = 10^{-4}$ for triplet-SVM in our experiments.

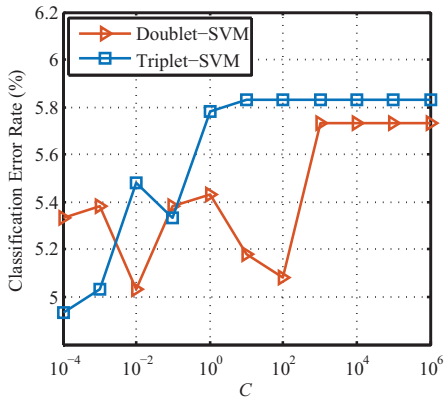


Figure 2.1 Classification error rate (%) versus C for doublet-SVM and triplet-SVM.

We compare the proposed methods with five representative and state-of-the-art metric learning models, i.e., LMNN [138], ITML [33], LDML [48], neighbourhood component analysis (NCA) [45] and maximally collapsing metric learning (MCML) [44], in terms of classification error rate and training time (in seconds). The source codes of LMNN[2], ITML[3], LDML[4], NCA[5] and MCML[6] are online avail-

[2] `http://www.cse.wustl.edu/~kilian/code/code.html`

[3] `http://www.cs.utexas.edu/~pjain/itml/`

[4] `http://lear.inrialpes.fr/people/guillaumin/code.php`

[5] `http://www.cs.berkeley.edu/~fowlkes/software/nca/`

[6] `http://homepage.tudelft.nl/19j49/Matlab_Toolbox_for_Dimensionality_Reduction.html`

able. Table 2.2 lists the classification error rates on the handwritten digit datasets. On the MNIST dataset, LMNN achieves the lowest error rate; on the USPS dataset, doublet-SVM achieves the lowest error rate; and on the Semeion dataset, triplet-SVM obtains the lowest error rate. We do not report the error rate of MCML on the MNIST dataset because MCML requires too large memory space (more than 30 GB) on this dataset and cannot be run in our PC.

The last row of Table 2.2 lists the average ranks of the seven metric learning models. We can see that triplet-SVM can achieve the best average rank, and doublet-SVM achieves the second best average rank.

We then compare the training time of these metric learning methods. All the experiments are executed in a PC with 4 Intel Core i5-2410 CPUs (2.30 GHz) and 16 GB RAM. We compare the five doublet-based metric learning methods and the two triplet-based methods, respectively. Fig. 2.2 shows the training time of doublet-SVM, ITML, LDML, MCML, and NCA. We can see that doublet-SVM is much faster than the other four methods. In average it is 2,000 times faster than the second fastest algorithm, ITML. Fig. 2.3 shows the training time of triplet-SVM and LMNN. One can see that triplet-SVM is about 100 times faster than LMNN on the three datasets.

2.4.2 Doublets/Triplets Construction

Let's first compare the classification performance by using different strategies to construct the doublet set. Using Doublet-SVM as an example, we consider the following two strategies to construct the doublet set:

1. *Nearest neighbor (NN) selection*: As described in Section 2.2.1, for each

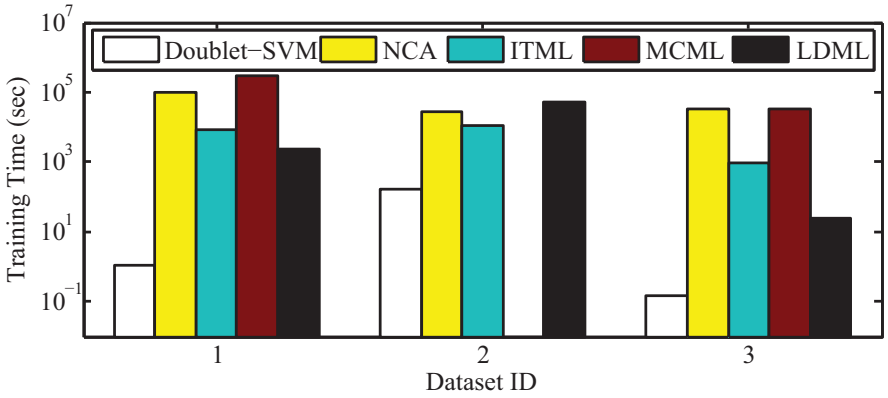


Figure 2.2 Training time (sec.) of doublet-SVM, NCA, ITML, MCML and LDML. From 1 to 3, the Dataset ID represents USPS, MNIST and Semeion.

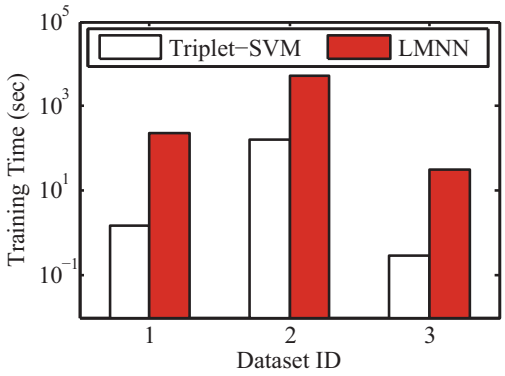


Figure 2.3 Training time (sec.) of triplet-SVM and LMNN. From 1 to 3, the Dataset ID represents USPS, MNIST and Semeion.

training sample $\mathbf{x}_i$, we construct 2 doublets $\{(\mathbf{x}_i, \mathbf{x}_i^s), (\mathbf{x}_i, \mathbf{x}_i^d)\}$, where $\mathbf{x}_i^s$ denotes the similar nearest neighbor of $\mathbf{x}_i$, and $\mathbf{x}_i^d$ denotes the dissimilar nearest

neighbor of $\mathbf{x}_i$. By constructing all such doublets from the training samples, we build a doublet set using the NN strategy.

2. *Random selection*: Given a training set of n samples, we randomly select $2n$ doublets from all the $n(n-1)/2$ possible doublets.

Table 2.3 lists the classification error rates of doublet-SVM by using the NN and the random selection strategies to construct the doublet set. The NN selection outperforms the random selection on all the three handwritten digit datasets. One can conclude that for doublet-SVM, the NN selection is better than the random selection to construct doublet set.

We then compare the classification performance by using different strategies to construct the triplet set. Using Triplet-SVM as an example, we also consider the NN selection and random selection strategies to construct triplet set:

1. *Nearest neighbor (NN) selection*: For each training sample $\mathbf{x}_i$, we construct a triplet $(\mathbf{x}_i, \mathbf{x}_i^s, \mathbf{x}_i^d)$, where $\mathbf{x}_i^s$ denotes the similar nearest neighbor of $\mathbf{x}_i$, and $\mathbf{x}_i^d$ denotes the dissimilar nearest neighbor of $\mathbf{x}_i$. By constructing all such triplets from the training samples, we build a triplet set using the NN strategy.

2. *Random selection*: Given a training set of n samples, we randomly select n triplets from all the possible triplets.

Table 2.3 lists the classification error rates of doublet-SVM and triplet-SVM by using the NN and the random selection strategies. The NN selection outperforms the random selection on 2 out of the 3 handwritten digit datasets. One can conclude that the NN selection strategy is also a better choice than the random selection strategy for triplet-SVM to construct triplet sets.

Table 2.3 The classification error rates (%) by using the random selection strategy and the NN selection strategy to select doublets/triplets on the handwritten digit datasets

Method	**Doublet-SVM (Random)**	**Doublet-SVM (NN)**	**Triplet-SVM (Random)**	**Triplet-SVM (NN)**
MNIST	3.41	**3.19**	3.84	**2.92**
USPS	5.49	**5.43**	**5.65**	5.78
Semeion	6.43	**5.09**	6.96	**4.71**

2.4.3 Person Re-identification

We evaluate our proposed doublet-SVM and triplet-SVM on the person re-identification task. We use the CUHK03 dataset in our evaluation. The CUHK03 dataset consists of 14,096 images taken from 1,467 pedestrians by two disjoint cameras [76]. We randomly select the images of 1,367 pedestrians as the training set, and use the rest images of 100 pedestrians as the testing set. By this strategy, 20 partitions of training and testing sets are constructed, and the reported performances are averaged over all the partitions. In the testing stage, we adopt the single-shot setting for evaluation, which randomly select one image from each pedestrian taken by one camera as the probe set, and select one image from each pedestrian taken by another camera as the gallery set.

We report the rank-1 accuracies and training time of doublet-SVM, triplet-SVM and the competing methods, *i.e.* ITML [33], DML-eig [149], LMNN [140], RANK [89], LDML [48], symmetry-driven accumulation of local features (SDALF) [39], eSDC [155], KISSME [69], XQDA [81], filter pairing neural network (FPNN) [76] and Zhang *et al.* [151], on CUHK03 database with manually labeled and detect-

ed bounding boxes in Table 2.4 and Table 2.5. For the methods with the rank-1 accuracy higher than 30%, we also report their CMC curves in Fig. 2.4. For our proposed doublet-SVM method, we select 5 nearest positive neighbors and 10 nearest negative neighbors of each sample to construct the doublet set. For our proposed triplet-SVM method, we select 3 nearest positive neighbors and 5 nearest negative neighbors of each sample to construct the triplet set. For FPNN [76], RANK [89], SDALF [39] and eSDC [155], we use the results in their original papers. As to the other methods, the results are obtained by using an effective feature representation named Local Maximal Occurrence (LOMO) [81].

Table 2.4 Rank-1 accuracies (%) on the CUHK03 database

Methods	**CUHK03-Labeled**	**CUHK03-Detected**
Doublet-SVM (LOMO)	51.25	45.05
Triplet-SVM (LOMO)	51.15	45.15
ITML (LOMO) [33]	46.40	43.25
DML-eig (LOMO) [149]	17.70	13.90
KISSME (LOMO) [69]	45.95	38.25
XQDA (LOMO) [81]	52.20	46.25
Zhang *et al.* (LOMO) [151]	58.90	53.70
LDML (LOMO) [48]	51.20	45.40
LMNN (LOMO) [140]	51.08	44.64
FPNN [76]	20.65	19.89
Euclidean (LOMO)	11.05	10.95
RANK [89]	10.42	8.52
SDALF [39]	5.60	4.87
eSDC [155]	8.76	7.68

We can see that the recognition accuracies of doublet-SVM and triplet-SVM

Table 2.5 Training time (s) on the CUHK03 database with LOMO feature [81]

Methods	Training Time (s)
Doublet-SVM	227.54
Triplet-SVM	190.72
ITML [33]	1228.50
DML-eig [149]	523.33
KISSME [69]	0.85
XQDA [81]	902.35
Zhang *et al.* [151]	1954.60
LDML [48]	794.38
LMNN [140]	8383.60

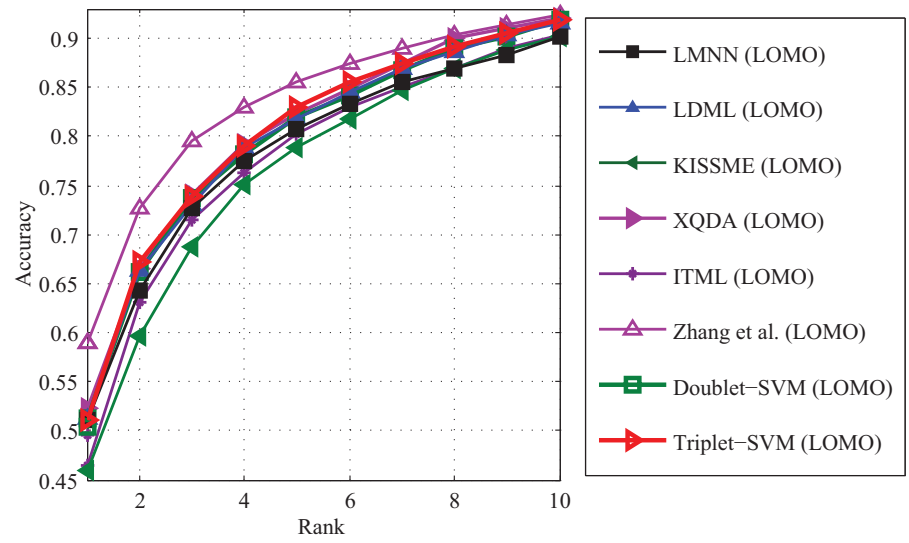
are higher than most of the other state-of-the-art methods, comparable to LDML [48], XQDA [81], and lower than the method by Zhang *et al.* [151]. The possible reason is that the method in [151] learns a discriminative null space by the kernel trick, while doublet-SVM and triplet-SVM learns the Mahalanobis distance metric or the linear subspace. We can also see that the training time of doublet-SVM and triplet-SVM are shorter than most of the other comparison methods except KISSME [69], because KISSME has its analytical solution, while the solutions of doublet-SVM and triplet-SVM should be obtained by SVM training and singular value decomposition.

2.5 Summary

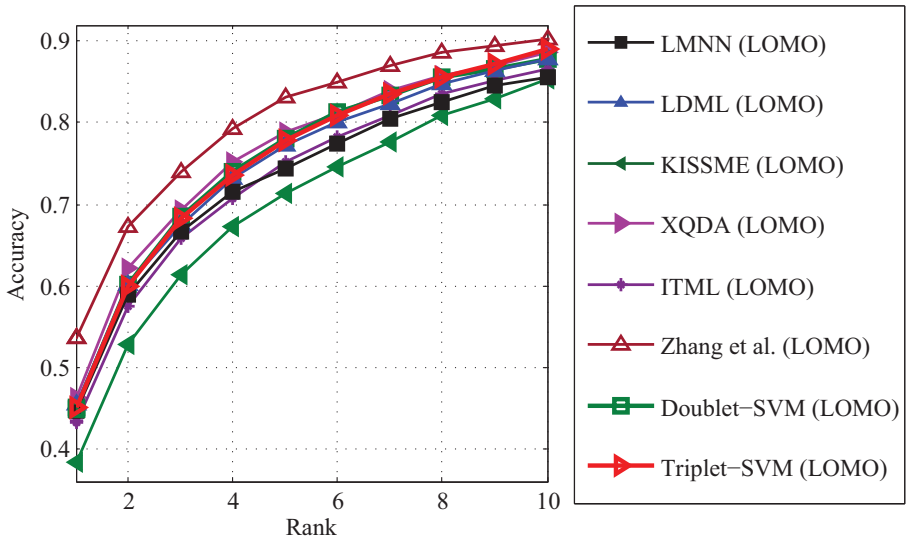
In this chapter, we proposed a general kernel classification framework for distance metric learning. By coupling a degree-2 polynomial kernel with some kernel

methods, the proposed framework can unify many representative and state-of-the-art metric learning approaches such as LMNN, ITML and LDML. The proposed framework also provides a good platform for developing new metric learning algorithms. Two metric learning methods, *i.e.*, doublet-SVM and triplet-SVM, were developed and they can be efficiently implemented by the standard SVM solvers. Our experimental results on the handwritten digit classification and person re-identification tasks showed that doublet-SVM and triplet-SVM are much faster than most of the state-of-the-art methods in terms of training time, while they achieve very competitive results in terms of classification error rate.

The proposed kernel classification framework provides a new perspective on developing metric learning methods via kernel classifiers. By incorporating the kernel learning methods for semi-supervised learning, multiple instance learning, etc., the proposed framework can be adopted to develop metric learning approaches for many other applications. By replacing the degree-2 polynomial kernel with nonlinear kernel functions which satisfy the Mercer's condition [114], the proposed framework can also be extended to nonlinear metric learning.



(a)



(b)

Figure 2.4 The CMC curves of different methods on the CUHK03 database with (a) manually labeled bounding box and (b) automatically detected bounding box.

Chapter 3

Distance Metric Learning via Iterated Support Vector Machines

3.1 Introduction

Distance metric learning aims to train a valid distance metric which can enlarge the distances between samples of different classes while reducing the distances between samples of the same class [8, 158]. Metric learning is closely related to other learning problems, including k-Nearest Neighbor (k-NN) classification [140] and clustering [142], and has also been widely applied in many image classification tasks, *e.g.*, face recognition [48] and person re-identification [69, 79]. One popular metric learning approach is Mahalanobis distance metric learning, which is to learn a linear transformation matrix $\mathbf{L}$ or a matrix $\mathbf{M} = \mathbf{L}^T\mathbf{L}$ from the training data. Given two samples $\mathbf{x}_i$ and $\mathbf{x}_j$, their Mahalanobis distance is defined as:

$$d^2_{\mathbf{M}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left\|\mathbf{L}(\mathbf{x}_i - \mathbf{x}_j)\right\|_2^2 = \left(\mathbf{x}_i - \mathbf{x}_j\right)^T \mathbf{M}\left(\mathbf{x}_i - \mathbf{x}_j\right). \tag{3.1}$$

To satisfy the nonnegative property of a distance metric, $\mathbf{M}$ should be positive semidefinite (PSD). According to which one of $\mathbf{M}$ and $\mathbf{L}$ is learned, Mahalanobis distance metric learning methods can be grouped into two categories. Methods that learn $\mathbf{L}$, including neighborhood components analysis (NCA) [45], large margin components analysis (LMCA) [125] and neighborhood repulsed metric learning (NRML) [87], are mostly formulated as nonconvex optimization problems, which are solved by gradient descent optimizers. Taking the PSD constraint into account, methods that learn $\mathbf{M}$, including large margin nearest neighbor (LMNN) [138] and maximally collapsing metric learning (MCML) [44], are mostly formulated as convex semidefinite programming (SDP), which can be optimized by standard SDP solvers [138], projected gradient [142], Boosting-like [119], or Frank-Wolfe [149] algorithms. Davis *et al.* [33] proposed an information-theoretic metric learning (ITML) model with an iterative Bregman projection algorithm to avoid the projections onto the PSD cone. Besides, the use of online solvers has been discussed in [21, 69, 92, 116].

On the other hand, the Mahalanobis distance in (3.1) can be equivalently written as:

$$d^2_{\mathbf{M}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \mathrm{tr}\left(\mathbf{M}^T(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T\right) = \left\langle \mathbf{M}, \mathbf{X}_{ij} \right\rangle, \tag{3.2}$$

where $\mathbf{M}$ is a PSD matrix, $\mathbf{X}_{ij} = (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$, $\langle \mathbf{A}, \mathbf{B} \rangle = \mathrm{tr}\left(\mathbf{A}^T\mathbf{B}\right)$ is defined as the Frobenius inner product of two matrices $\mathbf{A}$ and $\mathbf{B}$, and tr($\bullet$) stands for the matrix trace operator. By defining the following kernel function

$$K\left(\left(\mathbf{x}_i, \mathbf{x}_j\right), (\mathbf{x}_k, \mathbf{x}_l)\right) = \left\langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \right\rangle = \left(\left(\mathbf{x}_i - \mathbf{x}_j\right)^T (\mathbf{x}_k - \mathbf{x}_l)\right)^2, \tag{3.3}$$

we can cast the Mahalanobis distance in (3.2) as a kernel classifier. For convenience,

we rewrite $K\left(\left(\mathbf{x}_i, \mathbf{x}_j\right), (\mathbf{x}_k, \mathbf{x}_l)\right)$ as K_{ijkl} in the following sections.

As kernel methods [2, 6] have been widely studied in many learning tasks, e.g., semi-supervised learning, multiple instance learning, multitask learning, etc. Kernel learning methods, such as support vector machine (SVM), exhibit good generalization performance. There are many open resources on kernel classification methods, and a variety of toolboxes and libraries have been released [11, 17, 113, 124, 127]. It is thus important to investigate the connections between metric learning and kernel classification and explore how to utilize the kernel classification resources in the research and development of new metric learning methods. In Chapter 2, we made an attempt on developing a kernel classification framework for metric learning. However, in their heuristic two-step greedy scheme, the PSD constraint is ignored in the first step, and then they simply project the learned matrix onto the PSD cone to obtain the final valid distance metric.

In this chapter, we propose a novel formulation of metric learning by casting it as a kernel classification problem with PSD constraint, which allows us to effectively and efficiently learn valid distance metrics by iterated training of SVM. The off-the-shelf SVM solvers such as LibSVM [17] can be employed to solve the metric learning problem. Specifically, we propose two novel methods to bridge metric learning with the well-developed SVM techniques, and they are easy to implement. First, we propose a Positive-semidefinite Constrained Metric Learning (PCML) model, which can be solved via iterating between PSD projection and dual SVM learning. Second, by re-parameterizing the matrix $\mathbf{M}$, we propose a Nonnegative-coefficient Constrained Metric Learning (NCML) model, which can be solved by iterated learning of two SVMs. Both PCML and NCML have globally

optimal solutions. Compared with [131], our PCML and NCML provide principled schemes to exploit SVM solver for metric learning with guarantee on global optimum. Our experiments on handwritten digit recognition, face verification and person re-identification tasks clearly demonstrate the effectiveness of our methods. The contribution of this chapter is three-fold:

1. Two models, *i.e.*, PCML and NCML, are proposed by formulating metric learning as kernel classification problem with PSD constraint. Both PCML and NCML models are convex, and can guarantee the PSD property of the learned distance metric.

2. An optimization algorithm is developed for solving PCML by iterating between SVM training and PSD projection. It has the computational complexity of $O(d^3)$ per iteration w.r.t the feature dimension d, and can converge to global optimum.

3. An optimization algorithms is developed for NCML by iterating between the training of two SVMs. It has the computational complexity of $O(d)$ per iteration w.r.t d, and can guarantee the global optimality of the solution.

3.2 Positive-semidefinite Constrained Metric Learning (PCML)

In this section, we formulate metric learning as a convex SDP, and propose the PCML model. We then develop a learning algorithm by alternatively iterating between SVM training and PSD projection, and discuss the convergence of PCML.

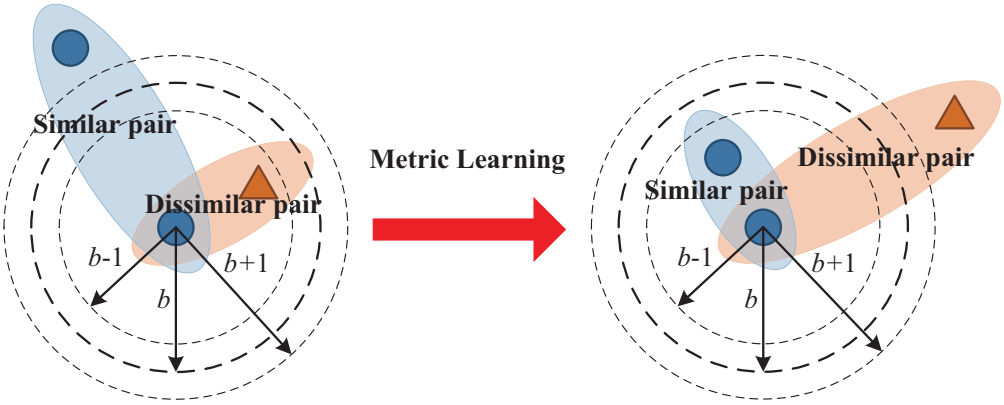


Figure 3.1 Schematic illustration of the constraints of similar and dissimilar pairs.

3.2.1 PCML Problem

Denote by $\{(\mathbf{x}_i, y_i)|\, i = 1, 2, ..., N\}$ a training set, where $\mathbf{x}_i \in \mathbb{R}^d$ is the ith training sample, and y_i is the class label of $\mathbf{x}_i$. Let $\mathcal{S} = \{\left(\mathbf{x}_i, \mathbf{x}_j\right) : \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ have the same class label }\}$ be the set of similar pairs, $\mathcal{D} = \{\left(\mathbf{x}_i, \mathbf{x}_j\right) : \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ have different class labels}\}$ be the set of dissimilar pairs, and b is the distance threshold. We hope the Mahalanobis distance of a similar pair should be lower than $b-1$, and that of a dissimilar pair should be higher than $b+1$ (see Fig. 3.1). By introducing an indicator variable h_{ij},

$$h_{ij} = \begin{cases} 1, \text{ if } (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{D} \\ -1, \text{ if } (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{S}, \end{cases} \tag{3.4}$$

the PCML model can be formulated as:

$$\begin{aligned} \min_{\mathbf{M},b,\xi} \quad & \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum\nolimits_{i,j}\xi_{ij} \\ \text{s.t.} \quad & h_{ij}\left(\left\langle \mathbf{M}, \mathbf{X}_{ij}\right\rangle - b\right) \geq 1 - \xi_{ij}, \xi_{ij} \geq 0, \ \forall i, j \\ & \mathbf{M} \succcurlyeq 0, \end{aligned} \tag{3.5}$$

where ξ_{ij} denotes the slack variables, and $\|\cdot\|_F$ denotes the Frobenius norm.

3.2.2 PCML Dual Problem

The PCML model is convex and can be solved by standard SDP solvers. However, the high complexity of general-purpose interior-point SDP solver makes it only suitable for small-scale problems. In order to improve the efficiency, we first analyze the Lagrange duality of the PCML model, and then propose an algorithm to iterate between SVM training and PSD projection to learn the distance metric.

From the PCML model in Eq. (3.5), we derive the Lagrangian of PCML as follows,

$$L(\boldsymbol{\lambda}, \boldsymbol{\kappa}, \mathbf{Y}, \mathbf{M}, b, \boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum\nolimits_{i,j}\xi_{ij} - \sum\nolimits_{i,j}\lambda_{ij}\left[h_{ij}\left(\left\langle \mathbf{M}, \mathbf{X}_{ij}\right\rangle - b\right) - 1 + \xi_{ij}\right] - \sum\nolimits_{i,j}\kappa_{ij}\xi_{ij} - \langle \mathbf{Y}, \mathbf{M}\rangle, \tag{3.6}$$

where $\lambda_{ij} \geq 0$, $\kappa_{ij} \geq 0, \forall i, j$, and $\mathbf{Y} \succcurlyeq 0$ are the Lagrange multipliers. Converting the primal problem to its dual problem needs the following KKT conditions:

$$\frac{\partial L(\boldsymbol{\lambda}, \boldsymbol{\kappa}, \mathbf{Y}, \mathbf{M}, b, \boldsymbol{\xi})}{\partial \mathbf{M}} = \mathbf{0} \Rightarrow \mathbf{M} - \sum\nolimits_{i,j}\lambda_{ij}h_{ij}\mathbf{X}_{ij} - \mathbf{Y} = \mathbf{0}, \tag{3.7}$$

$$\frac{\partial L(\boldsymbol{\lambda}, \boldsymbol{\kappa}, \mathbf{Y}, \mathbf{M}, b, \boldsymbol{\xi})}{\partial b} = 0 \Rightarrow \sum\nolimits_{i,j}\lambda_{ij}h_{ij} = 0, \tag{3.8}$$

$$\frac{\partial L(\boldsymbol{\lambda}, \boldsymbol{\kappa}, \mathbf{Y}, \mathbf{M}, b, \boldsymbol{\xi})}{\partial \xi_{ij}} = C - \lambda_{ij} - \kappa_{ij} = 0 \Rightarrow 0 \leq \lambda_{ij} \leq C, \ \forall i, j, \tag{3.9}$$

$$h_{ij}\left(\left\langle \mathbf{M}, \mathbf{X}_{ij}\right\rangle - b\right) - 1 + \xi_{ij} \geq 0, \ \xi_{ij} \geq 0, \tag{3.10}$$

$$\lambda_{ij} \geq 0, \ \kappa_{ij} \geq 0, \ \mathbf{Y} \succcurlyeq 0, \tag{3.11}$$

$$\lambda_{ij}\left[h_{ij}\left(\left\langle \mathbf{M}, \mathbf{X}_{ij}\right\rangle - b\right) - 1 + \xi_{ij}\right] = 0, \ \kappa_{ij}\xi_{ij} = 0. \tag{3.12}$$

(3.7) implies the following relationship:

$$\mathbf{M} = \sum\nolimits_{i,j}\lambda_{ij}h_{ij}\mathbf{X}_{ij} + \mathbf{Y}. \tag{3.13}$$

Substituting (3.7)~(3.9) back into the Lagrangian, we get the Lagrange dual problem of PCML:

$$\begin{aligned} \max_{\lambda,\mathbf{Y}} \quad & -\frac{1}{2}\left\|\sum\nolimits_{i,j}\lambda_{ij}h_{ij}\mathbf{X}_{ij}+\mathbf{Y}\right\|_F^2+\sum\nolimits_{i,j}\lambda_{ij} \\ \text{s.t.} \quad & \sum\nolimits_{i,j}\lambda_{ij}h_{ij}=0, 0\le\lambda_{ij}\le C, \forall i,j, \quad \mathbf{Y}\succcurlyeq 0. \end{aligned} \tag{3.14}$$

From (3.13) and (3.14), $\mathbf{M}$ is explicitly determined by the training procedure, but b is not. Nevertheless, b can be found by using the KKT condition in (3.9) and (3.12), and we can take any training point, for which $0<\lambda_{ij}<C$, to compute b by

$$b=\left\langle \mathbf{M},\mathbf{X}_{ij}\right\rangle -1/h_{ij}, \quad \text{for all } 0<\lambda_{ij}<C. \tag{3.15}$$

After b is computed, we can compute ξ_{ij} by

$$\xi_{ij}=\begin{cases} 0, & \text{for all } \lambda_{ij}<C \\ \left[1-h_{ij}\left(\left\langle \mathbf{M},\mathbf{X}_{ij}\right\rangle -b\right)\right]_+, & \text{for all } \lambda_{ij}=C, \end{cases} \tag{3.16}$$

where $[z]_+=\max(z,0)$ denotes the hinge loss.

3.2.3 Alternating Optimization Algorithm

To solve the dual problem efficiently, we propose an optimization approach by updating $\boldsymbol{\lambda}$ and $\mathbf{Y}$ alternatively. Given $\mathbf{Y}$, we introduce a new variable $\boldsymbol{\eta}$ with $\eta_{ij}=1-h_{ij}\left\langle \mathbf{X}_{ij},\mathbf{Y}\right\rangle =1-h_{ij}\left(\mathbf{x}_i-\mathbf{x}_j\right)^T\mathbf{Y}\left(\mathbf{x}_i-\mathbf{x}_j\right)$. With the kernel function in (3.3), the subproblem on $\boldsymbol{\lambda}$ can be formulated as the following QP problem:

$$\begin{aligned} \max_{\lambda} \quad & -\frac{1}{2}\sum\nolimits_{i,j,k,l}\lambda_{ij}\lambda_{kl}h_{ij}h_{kl}K_{ijkl}+\sum\nolimits_{i,j}\eta_{ij}\lambda_{ij} \\ \text{s.t.} \quad & \sum\nolimits_{i,j}\lambda_{ij}h_{ij}=0, 0\le\lambda_{ij}\le C, \quad \forall i,j. \end{aligned} \tag{3.17}$$

Algorithm 1 Algorithm of PCML

Input: $\mathcal{S} = \{\left(\mathbf{x}_i, \mathbf{x}_j\right) : \text{the class labels of } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are the same}\}$, $\mathcal{D} = \{\left(\mathbf{x}_i, \mathbf{x}_j\right) : \text{the class labels of } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are different}\}$, and h_{ij}.

Output: $\mathbf{M}$.

Initialize $\mathbf{Y}^{(0)}$, $t \leftarrow 0$.

repeat

1. Update $\eta^{(t+1)}$ with $\eta_{ij}^{(t+1)} = 1 - h_{ij} \left\langle \mathbf{X}_{ij}, \mathbf{Y}^{(t)} \right\rangle$.
2. Update $\boldsymbol{\lambda}^{(t+1)}$ by solving the subproblem (7) using an SVM solver.
3. Update $\mathbf{Y}_0^{(t+1)} = -\sum_{i,j} \lambda_{ij}^{(t+1)} h_{ij} \mathbf{X}_{ij}$.
4. Update $\mathbf{Y}^{(t+1)} = \mathbf{U}^{(t+1)} \boldsymbol{\Lambda}_+^{(t+1)} \mathbf{U}^{(t+1)^T}$, where $\mathbf{Y}_0^{(t+1)} = \mathbf{U}^{(t+1)} \boldsymbol{\Lambda}^{(t+1)} \mathbf{U}^{(t+1)^T}$ and $\boldsymbol{\Lambda}_+^{(t+1)} = \max\left(\boldsymbol{\Lambda}^{(t+1)}, \mathbf{0}\right)$.
5. $t \leftarrow t + 1$.

until convergence

$\mathbf{M} = \sum_{i,j} \lambda_{ij}^{(t-1)} h_{ij} \mathbf{X}_{ij} + \mathbf{Y}^{(t-1)}$.

return M

This subproblem on $\boldsymbol{\lambda}$ is a kernel-based classification problem, and can be efficiently solved by using the existing SVM solvers [17]. Given $\boldsymbol{\lambda}$, the subproblem on $\mathbf{Y}$ can be formulated as the projection onto the convex cone of PSD matrices:

$$\min_{\mathbf{Y}} \quad \|\mathbf{Y} - \mathbf{Y}_0\|_F^2, \quad \text{s.t.} \quad \mathbf{Y} \succcurlyeq 0, \tag{3.18}$$

where $\mathbf{Y}_0 = -\sum_{i,j} \lambda_{ij} h_{ij} \mathbf{X}_{ij}$. Through the eigen-decomposition of $\mathbf{Y}_0$, i.e., $\mathbf{Y}_0 = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ and $\boldsymbol{\Lambda}$ is the diagonal matrix of eigenvalues, the solution to (3.18) can be expressed as $\mathbf{Y} = \mathbf{U}\boldsymbol{\Lambda}_+\mathbf{U}^T$, where $\boldsymbol{\Lambda}_+ = \max(\boldsymbol{\Lambda}, \mathbf{0})$. Finally, the PCML algorithm is summarized in **Algorithm 1**.

3.2.4 Optimality Condition

Our algorithms can be treated as an implementation of generalized block coordinate descent (GBCD) [144] with two blocks. In our algorithms, the optimal solution to each subproblem is obtained. As stated in [144], when the objective function is strongly convex, GBCD can converge to the global optimal solution. Therefore, the proposed algorithm can reach the global optimum of the problems in (3.5) and (3.14).

Moreover, the optimality condition of our algorithm can be checked by the duality gap in each iteration, which is defined as the difference between the primal and dual objective values:

$$\text{DualGap}_{\text{PCML}}^{(n)} = \frac{1}{2}\left\|\mathbf{M}^{(n)}\right\|_F^2 + C\sum\nolimits_{i,j}\xi_{ij}^{(n)} - \sum\nolimits_{i,j}\lambda_{ij}^{(n)} + \frac{1}{2}\left\|\sum\nolimits_{i,j}\lambda_{ij}^{(n)}h_{ij}\mathbf{X}_{ij} + \mathbf{Y}^{(n)}\right\|_F^2, \tag{3.19}$$

where $\mathbf{M}^{(n)}$, $\boldsymbol{\xi}^{(n)}$, $\boldsymbol{\lambda}^{(n)}$, and $\mathbf{Y}^{(n)}$ are feasible primal and dual variables, and $\text{DualGap}_{\text{PCML}}^{(n)}$ is the duality gap in the nth iteration. According to (3.13), we can derive that

$$\mathbf{M}^{(n)} = \sum\nolimits_{i,j}\lambda_{ij}^{(n)}h_{ij}\mathbf{X}_{ij} + \mathbf{Y}^{(n)} = \mathbf{Y}^{(n)} - \mathbf{Y}_0^{(n)}. \tag{3.20}$$

As shown in Section 3.2.3, $\mathbf{Y}_0^{(n)} = \mathbf{U}^{(n)}\boldsymbol{\Lambda}^{(n)}\mathbf{U}^{(n)^T}$, $\mathbf{Y}^{(n)} = \mathbf{U}^{(n)}\boldsymbol{\Lambda}_+^{(n)}\mathbf{U}^{(n)^T}$, and hence $\mathbf{M}^{(n)} = \mathbf{U}^{(n)}\boldsymbol{\Lambda}_-^{(n)}\mathbf{U}^{(n)^T}$, where $\boldsymbol{\Lambda}_-^{(n)} = \boldsymbol{\Lambda}_+^{(n)} - \boldsymbol{\Lambda}^{(n)}$. Thus, $\left\|\mathbf{M}^{(n)}\right\|_F^2$ can be computed by

$$\begin{aligned}\left\|\mathbf{M}^{(n)}\right\|_F^2 &= \text{tr}\left(\mathbf{M}^{(n)^T}\mathbf{M}^{(n)}\right) = \text{tr}\left(\mathbf{U}^{(n)}\boldsymbol{\Lambda}_-^{(n)}\mathbf{U}^{(n)^T}\mathbf{U}^{(n)}\boldsymbol{\Lambda}_-^{(n)}\mathbf{U}^{(n)^T}\right)\\ &= \text{tr}\left(\mathbf{U}^{(n)}\boldsymbol{\Lambda}_-^{(n)2}\mathbf{U}^{(n)^T}\right) = \text{tr}\left(\boldsymbol{\Lambda}_-^{(n)2}\right).\end{aligned} \tag{3.21}$$

Substituting (3.20) and (3.21) into (3.19), the duality gap of PCML can be obtained as follows

$$\text{DualGap}_{\text{PCML}}^{(n)} = C\sum\nolimits_{i,j}\xi_{ij}^{(n)} - \sum\nolimits_{i,j}\lambda_{ij}^{(n)} + \text{tr}\left(\boldsymbol{\Lambda}_-^{(n)^2}\right). \tag{3.22}$$

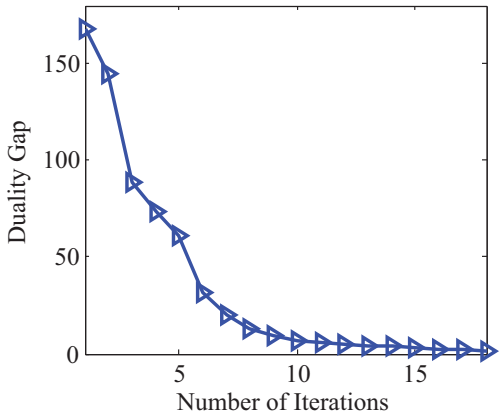


Figure 3.2 Duality gap vs. number of iterations on the *PenDigits* database for PCML.

Based on the KKT conditions of the PCML dual problem in (3.14), $\xi_{ij}^{(n)}$ and b can be obtained by Eqns. (3.16) and (3.15), respectively. The duality gap is always nonnegative and approaches to zero when the primal problem is convex. Thus, it can be used as the termination condition of the algorithm. Fig. 3.2 plots the curve of duality gap versus the number of iterations on the *PenDigits* database by PCML. The duality gap approaches to zero in less than 20 iterations and our algorithm will reach the global optimum (Chapter 5, [13]). In **Algorithm 1**, we adopt the following termination condition:

$$\text{DualGap}_{\text{PCML}}^{(t)} < \varepsilon \cdot \text{DualGap}_{\text{PCML}}^{(1)}, \tag{3.23}$$

where ε is a small constant.

3.2.5 Remarks

Construction of pairwise constraints: Based on the training set, N^2 pairwise con-

straints can be introduced in total. However, in practice we only need to choose a subset of pairwise constraints to reduce the computational cost. For each sample, we find its k nearest neighbors with the same label to construct similar pairs and its k nearest neighbors with different labels to construct dissimilar pairs. Thus, we only need $2kN$ pairwise constraints, and we can reduce the scale of pairwise constraints from $O\left(N^2\right)$ to $O\left(kN\right)$. Since k is usually small, the computational cost of metric learning is much reduced. Similar strategy for constructing pairwise or triplet constraints can be found in [55, 140].

Computational Complexity: We use the LibSVM library for SVM training. The computational complexity of SMO-type algorithms [103] is $O(k^2N^2d)$. For PSD projection, the complexity of conventional SVD algorithms is $O(d^3)$.

3.3 Nonnegative-coefficient Constrained Metric Learning (NCML)

In PCML, the computational complexity of the PSD projection is $O(d^3)$, which limits the training efficiency for data with high dimension. Therefore, we propose a NCML model, in which we re-parameterize the matrix $\mathbf{M}$ as the linear combination of a series of rank-1 matrices, and let the coefficients to be nonnegative to guarantee the PSD property of the matrix $\mathbf{M}$. NCML does not need any PSD projection in training and has low computational complexity w.r.t. d.

Given a set of rank-1 PSD matrices $\mathbf{M}_t = \mathbf{m}_t\mathbf{m}_t^T$ $(t = 1, \cdots, T)$, a linear combination of $\mathbf{M}_t$ is defined as $\mathbf{M} = \sum_t \alpha_t\mathbf{M}_t$, where α_t is the scalar coefficient. One can easily prove the following theorem.

Theorem 3.3.1 *If the scalar coefficient $\alpha_t \geq 0,\ \forall t$, the matrix $\mathbf{M} = \sum_t \alpha_t \mathbf{M}_t$ is PSD, where $\mathbf{M}_t = \mathbf{m}_t \mathbf{m}_t^T$ is a rank-1 PSD matrix.*

Proof Denote by $\mathbf{u} \in \mathbb{R}^d$ a random vector. Based on the expression of $\mathbf{M}$, we have:

$$\mathbf{u}^T \mathbf{M} \mathbf{u} = \mathbf{u}^T \left(\sum\nolimits_t \alpha_t \mathbf{m}_t \mathbf{m}_t^T \right) \mathbf{u} = \sum\nolimits_t \alpha_t \mathbf{u}^T \mathbf{m}_t \mathbf{m}_t^T \mathbf{u} = \sum\nolimits_t \alpha_t \left(\mathbf{u}^T \mathbf{m}_t \right)^2.$$

Since $\left(\mathbf{u}^T \mathbf{m}_t \right)^2 \geq 0$ and $\alpha_t \geq 0,\ \forall t$, we have $\mathbf{u}^T \mathbf{M} \mathbf{u} \geq 0$. Therefore, $\mathbf{M}$ is a PSD matrix.

3.3.1 NCML Problem

Motivated by **Theorem 3.3.1**, we impose the PSD constraint by re-parameterizing the distance metric $\mathbf{M}$, and develop a nonnegative-coefficient constrained metric learning (NCML) method to learn the PSD matrix $\mathbf{M}$. Given the training data $\mathcal{S}$ and $\mathcal{D}$, a rank-1 PSD matrix $\mathbf{X}_{ij}$ can be constructed for each pair $\left(\mathbf{x}_i, \mathbf{x}_j \right)$. By assuming that the learned matrix should be the linear combination of $\mathbf{X}_{ij}$ with the nonnegative coefficient constraint, the NCML model is formulated as:

$$\begin{aligned} \min_{\mathbf{M}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}} \quad & \frac{1}{2} \|\mathbf{M}\|_F^2 + C \sum\nolimits_{i,j} \xi_{ij} \\ \text{s.t.} \quad & h_{ij} \left(\left\langle \mathbf{M}, \mathbf{X}_{ij} \right\rangle - b \right) \geq 1 - \xi_{ij}, \alpha_{ij} \geq 0, \xi_{ij} \geq 0, \forall i, j \\ & \mathbf{M} = \sum\nolimits_{i,j} \alpha_{ij} \mathbf{X}_{ij}. \end{aligned} \tag{3.24}$$

By substituting $\mathbf{M}$ with $\sum_{i,j} \alpha_{ij} \mathbf{X}_{ij}$, we reformulate NCML as:

$$\begin{aligned} \min_{\boldsymbol{\alpha}, b, \boldsymbol{\xi}} \quad & \frac{1}{2} \sum\nolimits_{i,j} \sum\nolimits_{k,l} \alpha_{ij} \alpha_{kl} K_{ijkl} + C \sum\nolimits_{i,j} \xi_{ij} \\ \text{s.t.} \quad & h_{ij} \left(\sum\nolimits_{k,l} \alpha_{kl} K_{ijkl} - b \right) \geq 1 - \xi_{ij} \\ & \alpha_{ij} \geq 0,\ \xi_{ij} \geq 0,\ \forall i, j. \end{aligned} \tag{3.25}$$

3.3.2 NCML Dual Problem

According to the NCML problem in Eq. (3.25), its Lagrangian can be defined as:

$$L(\boldsymbol{\beta},\boldsymbol{\sigma},\boldsymbol{\nu},\boldsymbol{\alpha},b,\boldsymbol{\xi}) = \frac{1}{2}\sum_{i,j,k,l}\alpha_{ij}\alpha_{kl}K_{ijkl} + C\sum_{i,j}\xi_{ij} - \sum_{i,j}\beta_{ij}\left[h_{ij}\left(\sum_{k,l}\alpha_{kl}K_{ijkl} - b\right) - 1 + \xi_{ij}\right] - \sum_{i,j}\nu_{ij}\xi_{ij} - \sum_{i,j}\sigma_{ij}\alpha_{ij}, \tag{3.26}$$

where $\beta_{ij} \geq 0$, $\sigma_{ij} \geq 0$ and $\nu_{ij} \geq 0$, $\forall i, j$ are the Lagrange multipliers. Converting the original problem to its dual problem needs the following KKT conditions:

$$\frac{\partial L(\boldsymbol{\beta},\boldsymbol{\sigma},\boldsymbol{\nu},\boldsymbol{\alpha},b,\boldsymbol{\xi})}{\partial \alpha_{ij}} = 0 \Rightarrow \sum\nolimits_{k,l}\alpha_{kl}K_{ijkl} - \sum\nolimits_{k,l}\beta_{kl}h_{kl}K_{ijkl} - \sigma_{ij} = 0, \tag{3.27}$$

$$\frac{\partial L(\boldsymbol{\beta},\boldsymbol{\sigma},\boldsymbol{\nu},\boldsymbol{\alpha},b,\boldsymbol{\xi})}{\partial b} = 0 \Rightarrow \sum\nolimits_{i,j}\beta_{ij}h_{ij} = 0, \tag{3.28}$$

$$\frac{\partial L(\boldsymbol{\beta},\boldsymbol{\sigma},\boldsymbol{\nu},\boldsymbol{\alpha},b,\boldsymbol{\xi})}{\partial \xi_{ij}} = 0 \Rightarrow C - \beta_{ij} - \nu_{ij} = 0 \Rightarrow 0 \leq \beta_{ij} \leq C, \tag{3.29}$$

$$h_{ij}\left(\sum\nolimits_{k,l}\alpha_{kl}K_{ijkl} - b\right) - 1 + \xi_{ij} \geq 0, \xi_{ij} \geq 0,\ \alpha_{ij} \geq 0,\ \forall i, j, \tag{3.30}$$

$$\beta_{ij} \geq 0,\ \sigma_{ij} \geq 0,\ \nu_{ij} \geq 0, \quad \forall i, j, \tag{3.31}$$

$$\beta_{ij}\left[h_{ij}\left(\sum\nolimits_{k,l}\alpha_{kl}K_{ijkl} - b\right) - 1 + \xi_{ij}\right] = 0, \nu_{ij}\xi_{ij} = 0,\ \sigma_{ij}\alpha_{ij} = 0, \quad \forall i, j. \tag{3.32}$$

Here we introduce a coefficient vector $\boldsymbol{\eta}$, which satisfies $\sigma_{ij} = \sum_{k,l}\eta_{kl}\left\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\right\rangle$. Note that $\left\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\right\rangle$ is a positive definite kernel. So we can guarantee that every $\boldsymbol{\eta}$ corresponds to a unique $\boldsymbol{\sigma}$, and vice versa. Equation (3.27) implies the following relationship between $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\eta}$:

$$\alpha_{ij} = \beta_{ij}h_{ij} + \eta_{ij}, \quad \forall i, j. \tag{3.33}$$

Substituting (3.27)~(3.29) back into the Lagrangian, we get the Lagrange dual problem of NCML as follows:

$$\begin{aligned} \max_{\boldsymbol{\eta},\boldsymbol{\beta}} \; & -\frac{1}{2}\sum_{i,j,k,l}\left(\beta_{ij}h_{ij}+\eta_{ij}\right)\left(\beta_{kl}h_{kl}+\eta_{kl}\right)K_{ijkl}+\sum_{i,j}\beta_{ij} \\ \text{s.t.} \quad & \textstyle\sum_{k,l}\eta_{kl}K_{ijkl}\geq 0, 0\leq\beta_{ij}\leq C,\ \forall i,j \\ & \textstyle\sum_{i,j}\beta_{ij}h_{ij}=0. \end{aligned} \tag{3.34}$$

From Eq. (3.33), we can first solve the above dual problem, and then obtain the matrix $\mathbf{M}$ by

$$\mathbf{M}=\textstyle\sum_{i,j}(\beta_{ij}h_{ij}+\eta_{ij})\mathbf{X}_{ij}, \tag{3.35}$$

Analogous to PCML, we can use the KKT condition in (3.29) and (3.32) to compute b and ξ_{ij} in NCML. Equations (3.29) and (3.32) show that $\xi_{ij}=0$ if $\beta_{ij}<C$, and $h_{ij}\left(\sum_{kl}\alpha_{kl}\left\langle\mathbf{X}_{ij},\mathbf{X}_{kl}\right\rangle-b\right)-1+\xi_{ij}=0$ if $\beta_{ij}>0$. Thus we can simply take any training data point, for which $0<\beta_{ij}<C$, to compute b by

$$b=\textstyle\sum_{k,l}\alpha_{kl}K_{ijkl}-1/h_{ij}. \tag{3.36}$$

After obtain b, we can compute β_{ij} by

$$\xi_{ij}=\begin{cases} 0, \forall\,\beta_{ij}<C \\ \left[1-h_{ij}\left(\sum_{k,l}\alpha_{kl}K_{ijkl}-b\right)\right]_{+}, \forall\,\beta_{ij}=C \end{cases} \tag{3.37}$$

3.3.3 Optimization Algorithm

There are two groups of variables, $\boldsymbol{\eta}$ and $\boldsymbol{\beta}$, in problem (3.34). We adopt an alternating minimization approach to solve them. First, given $\boldsymbol{\eta}$, the variables β_{ij} can be

obtained by:

$$\begin{aligned} \max_{\boldsymbol{\beta}} \quad & -\frac{1}{2}\sum_{i,j,k,l}\beta_{ij}\beta_{kl}h_{ij}h_{kl}K_{ijkl} + \sum_{i,j}\delta_{ij}\beta_{ij} \\ \text{s.t.} \quad & 0 \le \beta_{ij} \le C,\ \forall i, j, \sum_{i,j}\beta_{ij}h_{ij} = 0, \end{aligned} \tag{3.38}$$

where $\boldsymbol{\delta}$ is the variable with $\delta_{ij} = \left(1 - h_{ij}\sum_{kl}\eta_{kl}\left\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\right\rangle\right)$. Clearly, the subproblem on $\boldsymbol{\beta}$ is similar to the dual of SVM, and it can be solved by LibSVM [17].

Given $\boldsymbol{\beta}$, the subproblem on $\boldsymbol{\eta}$ can be formulated as follows:

$$\begin{aligned} \min_{\boldsymbol{\eta}} \quad & \frac{1}{2}\sum\nolimits_{i,j}\sum\nolimits_{k,l}\eta_{ij}\eta_{kl}K_{ijkl} + \sum\nolimits_{i,j}\eta_{ij}\gamma_{ij} \\ \text{s.t.} \quad & \sum\nolimits_{k,l}\eta_{ij}K_{ijkl} \ge 0,\ \forall i, j, \end{aligned} \tag{3.39}$$

where $\gamma_{ij} = \sum_{kl}\beta_{kl}h_{kl}\left\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\right\rangle$. To simplify the subproblem on $\boldsymbol{\eta}$, we derive its Lagrange dual problem. The Lagrangian of the problem (3.39) is:

$$L(\boldsymbol{\mu}, \boldsymbol{\eta}) = \frac{1}{2}\sum\nolimits_{i,j}\sum\nolimits_{k,l}\eta_{ij}\eta_{kl}K_{ijkl} + \sum\nolimits_{i,j}\eta_{ij}\gamma_{ij} - \sum\nolimits_{i,j}\mu_{ij}\sum\nolimits_{k,l}\eta_{kl}K_{ijkl}, \tag{3.40}$$

where $\boldsymbol{\mu}$ is the Lagrange multiplier which satisfies $\mu_{ij} \ge 0, \forall i, j$. Converting the original problem to its dual problem needs the following KKT condition:

$$\frac{\partial L(\boldsymbol{\mu}, \boldsymbol{\eta})}{\partial \eta_{ij}} = 0 \Rightarrow \sum\nolimits_{k,l}\eta_{kl}K_{ijkl} + \gamma_{ij} - \sum\nolimits_{k,l}\mu_{kl}K_{ijkl} = 0. \tag{3.41}$$

Equation (3.41) implies the following relationship between $\boldsymbol{\mu}$, $\boldsymbol{\eta}$ and $\boldsymbol{\beta}$:

$$\eta_{ij} = \mu_{ij} - h_{ij}\beta_{ij}, \quad \forall i, j. \tag{3.42}$$

Substituting (3.41) and (3.42) back into the Lagrangian, we get the following Lagrange dual problem of the subproblem on $\boldsymbol{\eta}$:

$$\begin{aligned} \max_{\boldsymbol{\mu}} \quad & -\frac{1}{2}\sum_{i,j}\sum_{k,l}\mu_{ij}\mu_{kl}K_{ijkl} + \sum_{i,j}\gamma_{ij}\mu_{ij} - \frac{1}{2}\sum_{i,j}\sum_{k,l}\beta_{ij}\beta_{kl}h_{ij}h_{kl}K_{ijkl} \\ \text{s.t.} \quad & \mu_{ij} \ge 0, \forall i, j. \end{aligned} \tag{3.43}$$

Since $\boldsymbol{\beta}$ is fixed in this subproblem, $\sum_{i,j}\sum_{k,l}\beta_{ij}\beta_{kl}h_{ij}h_{kl}K_{ijkl}$ remains constant in (3.43). Thus we can omit this term and have the following simplified Lagrange dual problem:

$$\begin{aligned} \max_{\boldsymbol{\mu}} \quad & -\frac{1}{2}\sum_{i,j}\sum_{k,l}\mu_{ij}\mu_{kl}K_{ijkl} + \sum_{i,j}\gamma_{ij}\mu_{ij} \\ \text{s.t.} \quad & \mu_{ij} \geq 0, \forall i, j. \end{aligned} \tag{3.44}$$

Clearly, problem (3.44) is more simple and can be efficiently solved by the SVM solvers.

After obtaining $\boldsymbol{\mu}$ and $\boldsymbol{\beta}$, the solution of $\boldsymbol{\alpha}$ in problem (3.25) can be obtained by

$$\alpha_{ij} = \mu_{ij}, \quad \forall i, j. \tag{3.45}$$

We then have $\mathbf{M} = \sum_{ij}\alpha_{ij}\mathbf{X}_{ij}$. The NCML algorithm is summarized in **Algorithm 2**.

3.3.4 Optimality Condition

Our NCML training algorithm can reach global optimum. From (3.25) and (3.34), the duality gap in the nth iteration is

$$\begin{aligned} \text{DualGap}^{(n)}_{\text{NCML}} =& \frac{1}{2}\sum_{i,j,k,l}\alpha^{(n)}_{ij}\alpha^{(n)}_{kl}K_{ijkl} + \frac{1}{2}\sum_{i,j,k,l}(\beta^{(n)}_{ij}h_{ij} + \eta^{(n)}_{ij})(\beta^{(n)}_{kl}h_{kl} + \eta^{(n)}_{kl})K_{ijkl} \\ & - \sum_{i,j}\beta^{(n)}_{ij} + C\sum_{i,j}\xi^{(n)}_{ij}, \end{aligned} \tag{3.46}$$

where $\alpha^{(n)}_{ij}$ and $\xi^{(n)}_{ij}$ are the feasible solutions to the primal problem, $\beta^{(n)}_{ij}$ and $\eta^{(n)}_{ij}$ are the feasible solutions to the dual problem, and $\text{DualGap}^{(n)}_{\text{NCML}}$ is the duality gap in the nth iteration. As $\eta^{(n)}_{ij}$ and $\mu^{(n)}_{ij}$ are the optimal solutions to the primal subproblem on $\boldsymbol{\eta}$ in (3.39) and its dual problem in (3.44), respectively, the duality gap of subproblem

Algorithm 2 Algorithm of NCML

Input: Training set $\left\{\left(\mathbf{x}_i, \mathbf{x}_j\right), h_{ij}\right\}$.

Output: The matrix $\mathbf{M}$.

Initialize $\boldsymbol{\eta}^{(0)}$ with small random values, $t \leftarrow 0$.

repeat

1. Update $\boldsymbol{\delta}^{(t+1)}$ with $\delta_{ij}^{(t+1)} = \left(1 - h_{ij} \sum_{kl} \eta_{kl}^{(t)} K_{ijkl}\right)$.
2. Update $\boldsymbol{\beta}^{(t+1)}$ by solving the subproblem (3.38) using an SVM solver.
3. Update $\boldsymbol{\gamma}^{(t+1)}$ with $\gamma_{ij}^{(t+1)} = \sum_{kl} \beta_{kl}^{(t+1)} h_{kl} K_{ijkl}$.
4. Update $\boldsymbol{\mu}^{(t+1)}$ by solving the subproblem (3.44) using an SVM solver.
5. Update $\boldsymbol{\eta}^{(t+1)}$ with $\eta_{ij}^{(t+1)} \leftarrow \mu_{ij}^{(t+1)} - h_{ij} \beta_{ij}^{(t+1)}$.
6. $t \leftarrow t + 1$.

until convergence

$\mathbf{M} = \sum_{ij} \mu_{ij}^{(t)} \mathbf{X}_{ij}$.

return M

on $\boldsymbol{\eta}$ is zero, i.e.,

$$\begin{aligned} &\frac{1}{2} \sum_{i,j,k,l} \eta_{ij}^{(n)} \eta_{kl}^{(n)} K_{ijkl} + \sum_{i,j} \eta_{ij}^{(n)} \gamma_{ij}^{(n)} + \frac{1}{2} \sum_{i,j,k,l} \mu_{ij}^{(n)} \mu_{kl}^{(n)} K_{ijkl} - \sum_{i,j} \gamma_{ij}^{(n)} \mu_{ij}^{(n)} \\ &+ \frac{1}{2} \sum_{i,j,k,l} \beta_{ij}^{(n)} \beta_{kl}^{(n)} h_{ij} h_{kl} K_{ijkl} = 0. \end{aligned} \tag{3.47}$$

As shown in (3.45), $\alpha_{ij}^{(n)}$ and $\mu_{ij}^{(n)}$ should be equal. We substitute (3.47) into (3.46) as follows:

$$\text{DualGap}_{\text{NCML}}^{(n)} = C \sum\nolimits_{i,j} \xi_{ij}^{(n)} - \sum\nolimits_{i,j} \beta_{ij}^{(n)} + \sum\nolimits_{i,j} \mu_{ij}^{(n)} \gamma_{ij}^{(n)}. \tag{3.48}$$

Based on the KKT conditions of the NCML dual problem in (3.34), $\xi_{ij}^{(n)}$ can be

obtained by

$$\xi_{ij}^{(n)} = \begin{cases} 0 \quad \text{for all } \beta_{ij}^{(n)} < C \\ \left[1 - h_{ij}\left(\sum_{k,l} \alpha_{kl}^{(n)} K_{ijkl} - b^{(n)}\right)\right]_+ = \left[\delta_{ij}^{(n+1)} - h_{ij}\left(\gamma_{ij}^{(n)} - b^{(n)}\right)\right]_+ \quad \text{for all } \beta_{ij}^{(n)} = C. \end{cases} \tag{3.49}$$

where $[z] = \max(z, 0)$ and $b^{(n)}$ can be obtained by

$$b^{(n)} = \sum_{k,l} \alpha_{kl}^{(n)} K_{ijkl} - 1/h_{ij} = \gamma_{ij}^{(n)} - \delta_{ij}^{(n+1)}/h_{ij} \quad \text{for all } 0 < \beta_{ij}^{(n)} < C. \tag{3.50}$$

Please refer to Section 3.3.2 for the derivation of $\xi_{ij}^{(n)}$ and $b^{(n)}$.

Fig. 3.3 plots the curve of duality gap versus the number of iterations on *PenDigits* database by NCML. The duality gap is nearly zero within 10∼15 iterations, and NCML reaches the global optimum. In the implementation of **Algorithm 2**, we adopt the following termination condition:

$$\text{DualGap}_{\text{NCML}}^{(t)} < \varepsilon \cdot \text{DualGap}_{\text{NCML}}^{(1)}, \tag{3.51}$$

where ε is a small constant.

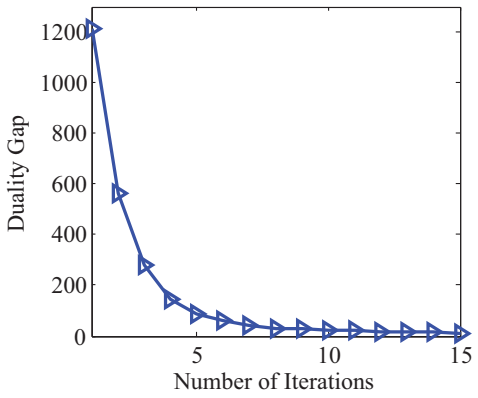


Figure 3.3 Duality gap vs. number of iterations on the *PenDigits* database for NCML.

3.3.5 Remarks

Computational complexity: We use the same strategy as that in PCML to construct the pairwise constraints. In each iteration, NCML calls for the SVM solver twice while PCML calls for it only once. When the SMO-type algorithm [103] is adopted for SVM training, the computational complexity of NCML is $O\left(k^2N^2d\right)$. One extra advantage of NCML lies in its lower computational cost with respect to d, which involves the computation of $K\left((\mathbf{x}_i,\mathbf{x}_j),(\mathbf{x}_k,\mathbf{x}_l)\right)$ and the construction of matrix $\mathbf{M}$. Since $K\left((\mathbf{x}_i,\mathbf{x}_j),(\mathbf{x}_k,\mathbf{x}_l)\right)=\left((\mathbf{x}_i-\mathbf{x}_j)^T(\mathbf{x}_k-\mathbf{x}_l)\right)^2$, the cost of kernel computation is $O(d)$. The cost of constructing the matrix $\mathbf{M}$ is less than $O\left(kNd^2\right)$, and this operation is required only once after obtaining $\boldsymbol{\beta}$ and $\boldsymbol{\mu}$.

Difference with Doublet-SVM in Chapter 2: Our PCML and NCML are related but distinctly different with doublet-SVM in Chapter 2. Like doublet-SVM, our PCML and NCML also cast metric learning as kernel classification problems. However, in doublet-SVM, $\mathbf{M}$ is first learned by ignoring the PSD constraint to exploit SVM solver and then projected onto the PSD cone. Thus, doublet-SVM is only a heuristic method and cannot obtain the global solution. In contrast, our PCML iterates between SVM training and PSD projection to learn $\mathbf{M}$, and our NCML iterates between two SVMs to learn $\mathbf{M}$. They provide a principled scheme to exploit SVM solver for metric learning. As analyzed in Sections 3.2.4 and 3.3.4, our algorithms can ensure the global optimality of $\mathbf{M}$. Moreover, by initializing $\mathbf{Y}$ with $\mathbf{0}$, doublet-SVM actually is a special case of PCML with one iteration.

3.4 Experimental Results

We evaluate our PCML and NCML methods for k-NN classification ($k = 1$) on handwritten digit classification, face verification, and person re-identification. PCML and NCML are implemented using the LibSVM[1] toolbox, and our codes are online available[2].

3.4.1 Evaluation on Handwritten Digit Classification Tasks

We use 4 handwritten digit databases to evaluate our methods. Table 3.1 provides a summary of these databases. On the *MNIST*, *PenDigits*, and *USPS* databases, the training set and test set are defined. On the *Semeion* databases, we use 10-fold CV to evaluate the metric learning models, and the classification error rate and training time are obtained by averaging over 10 runs of 10-fold cross-validation. As the dimensions of images in the *MNIST*, *Semeion* and *USPS* databases are relatively high, we use principal component analysis (PCA) to reduce the feature dimension to 100, and train the metrics in the PCA subspace.

Our PCML and NCML involve only one hyper-parameter, *i.e.*, the regularization parameter C. We simply adopt the cross-validation strategy to select C by investigating the influence of C on the classification error rate. Fig. 3.4 shows the curves of classification error rate versus C for PCML and NCML on the *USPS* database. We can observe that when $C < 1$, the classification error rates of NCML are low and stable. When C is higher than 1, the classification error rates of NCML jump dramatically. When $C = 0.1$ or 1, the classification error rates of PCML are

[1] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

[2] https://github.com/csfwang/ISVM

low. When C is in other values, the classification error rates of PCML are much higher. Thus, we set $C = 1$ in our experiments.

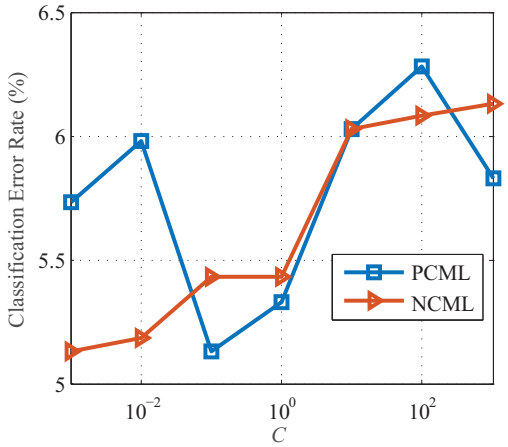


Figure 3.4 Classification error rates (%) versus C of PCML and NCML.

We compare PCML and NCML with the baseline Euclidean distance metric and 8 state-of-the-art metric learning models, including NCA [45], ITML [33], MCML [44], LDML [48], LMNN [140], PLML [134], DML-eig [149] and Doublet-SVM [131]. The source codes of NCA[3], ITML[4], MCML[5], LDML[6], LMNN[7], PLML[8], and DML-eig[9] are online available. We compare the classification error rates of the competing methods in Table 3.2. On *PenDigits* and *Semeion*, PCML achieves the lowest error rates. On *PenDigits*, NCML achieves the lowest error rates. According

[3] http://www.cs.berkeley.edu/~fowlkes/software/nca/

[4] http://www.cs.utexas.edu/~pjain/itml/

[5] http://homepage.tudelft.nl/19j49/Matlab_Toolbox_for_Dimensionality_Reduction.html

[6] http://lear.inrialpes.fr/people/guillaumin/code.php

[7] http://www.cse.wustl.edu/~kilian/code/code.html

[8] http://cui.unige.ch/~wangjun/

[9] http://empslocal.ex.ac.uk/people/staff/yy267/software.html

to [34], the average rank can provide a fair comparison of classification methods. Therefore, we provide the average ranks of competing methods in the last rows of Table 3.2. We do not report the error rate and training time of MCML on *MNIST* because MCML requires too large memory space (more than 30 GB) on this database and cannot run in our PC. From Table 3.2, PCML and NCML achieve the best average ranks on the handwritten digit databases, demonstrating their effectiveness for handwritten digit classification tasks.

Table 3.1 The handwritten digit databases used in the experiments.

Database	# of training samples	# of test samples	dimension	PCA dimension	# of classes
MNIST	60,000	10,000	784	100	10
PenDigits	7,494	3,498	16	N/A	10
Semeion	1,434	159	256	100	10
USPS	7,291	2,007	256	100	10

Finally, we compare the running time of PCML and NCML under different feature dimensions d. Fig. 3.6 shows the training time on *Semeion* with different PCA dimensions. When the dimension is lower than 110, the training time of NCML is longer than PCML, and it's better to use PCML in training. When the dimension is higher than 110, the training time of PCML increases and becomes longer than NCML, and it's better to use PCML in training. The results are consistent with the complexity analysis given in Sections 3.2.5 and 3.3.5.

Discussion

In this subsection, we give a brief discussion on the training efficiency and accuracy of PCML and NCML.

Table 3.2 Comparison of classification error rate (%) on the handwritten digit databases.

Database	**Euclidean**	**NCA**	**ITML**	**MCML**	**LDML**	**LMNN**
MNIST	2.87	3.75	2.89	N/A	6.05	**2.28**
PenDigits	2.26	2.23	2.29	2.26	6.20	2.23
Semeion	8.54	8.60	5.71	11.23	11.98	6.09
USPS	**5.08**	5.68	6.33	**5.08**	8.77	5.38
Average Rank	*4.75*	*7.00*	*6.50*	*7.00*	*10.75*	*3.75*
Database	**DML-eig**	**PLML**	**Doublet-SVM**	**PCML**	**NCML**	
MNIST	5.06	2.54	3.19	3.85	2.80	
PenDigits	3.75	2.46	**2.06**	**2.06**	**2.06**	
Semeion	5.72	7.66	5.21	**4.83**	5.53	
USPS	5.43	6.73	5.43	5.33	5.43	
Average Rank	*7.25*	*7.00*	*4.50*	***3.25***	***3.00***	

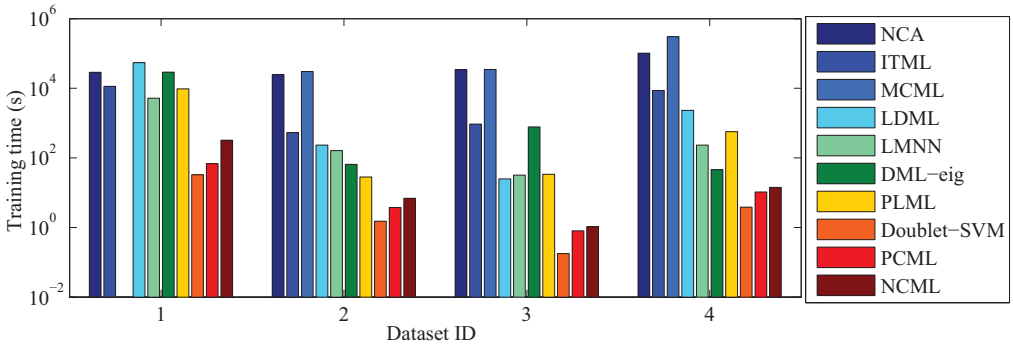


Figure 3.5 Training time (s) of NCA, ITML, MCML, LDML, LMNN, DML-eig, PLM-L, Doublet-SVM, PCML and NCML. From 1 to 4, the Database ID represents *MNIST*, *PenDigits*, *Semeion* and *USPS*.

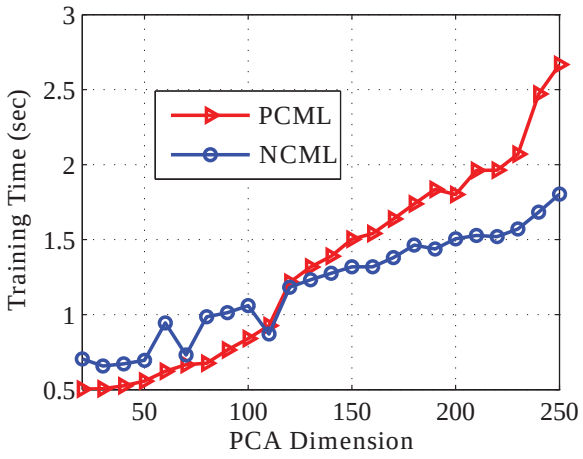


Figure 3.6 Training time (s) vs. PCA dimension on the *Semeion* database.

1. Training efficiency: Albeit lower in terms of computational complexity, NCM-L requires to run the SVM solver twice per iteration while PCML only once. Besides, the number of iterations may also be different for NCML and PCM-L. As shown in Fig. 3.6, when the feature dimension is lower, PCML is more efficient in training. As in most of our experiments, the dimensions of training samples are relatively low, making NCML less efficient than PCML.

2. Accuracy: From Theorem 1, the feasible domain of NCML is a subset of that of PCML. Thus, with sufficient training data, PCML has the opportunity to find distance metric in a larger searching domain. But in many practical problems, the training data generally are insufficient. Thus, the restriction of feasible domain by NCML may serve as some kind of regularization on the solution, and sometimes may even benefit classification performance.

3.4.2 Face Verification

We evaluate the proposed methods for face verification using the Labeled Faces in the Wild (LFW) [58] database. The face images in LFW were collected from the Internet and demonstrate large variations of pose, illumination, expression, etc. The database consists of 13,233 face images from 5,749 persons. Under the image restricted setting, the performance of a face verification method is evaluated by 10-fold CV. For each of the 10 runs, the database provides 300 positive pairs and 300 negative pairs for testing, and 5,400 image pairs for training. The verification rate and Receiver Operator Characteristic (ROC) curve of each method are obtained by averaging over the 10 runs.

In our experiments, we use the VGG-Face [102] feature to evaluate the face verification methods. Since the dimension of VGG-Face feature is high (i.e., 4096), PCA is used to reduce the feature dimension to 50. We transform each feature vector $\mathbf{x}$ by $\tilde{\mathbf{x}} = \mathbf{L}_{\mathcal{S}}^{-1}\mathbf{x}$, where $\mathbf{L}_{\mathcal{S}}\mathbf{L}_{\mathcal{S}}^{T} = \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$ [15]. Under the restricted setting, we only know whether two images are matched or not for the given pairs. In the training stage, we use the training pairs to train a Mahalanobis distance metric. In the test stage, we compare the distance of the test pair with the distance threshold to decide whether the two images are matched or not.

We report the ROC curves and verification accuracies of PCML, NCML, Doublet-SVM [131], ITML [33], DML-eig [149], KISSME [69], XQDA [81], DDML [56], TSML [157][10] and LM3L [57] in Fig. 3.7 and Table 3.3. It can be seen that our proposed PCML and NCML methods can achieve satisfactory verification accuracies which are higher or comparable to the competing methods. The training time

[10] As the ROC curve of TSML [157] hasn't been released, we haven't reported it in this chapter.

of PCML and NCML are much shorter than ITML [33] and DML-eig [149], but are longer than Doublet-SVM [131], KISSME [69] and XQDA [81]. We note that Doublet-SVM [131] is a two-stage method and KISSME is a one-pass optimization method. And they cannot guarantee to obtain the global optimum of the model. XQDA [81] is a subspace method, and its closed-form solution can be obtained by eigenvalue decomposition. In contrast, our proposed PCML and NCML methods solve a convex SDP problem and are able to reach the global optimum.

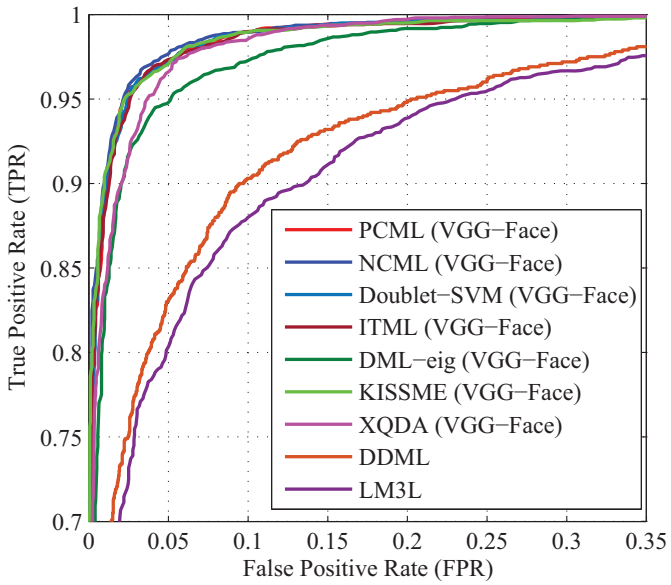


Figure 3.7 The ROC curves of different methods on the LFW database.

Table 3.3 Verification accuracies (%) and training time (s) of competing methods on the LFW-funneled database.

Methods	Verification Accuracy (%)	Training Time (s)
PCML (VGG-Face)	96.43	9.81
NCML (VGG-Face)	**96.63**	10.16
Doublet-SVM[131] (VGG-Face)	96.40	0.43
ITML[33] (VGG-Face)	96.40	194.92
DML-eig[149] (VGG-Face)	94.90	256.24
KISSME[69] (VGG-Face)	96.33	0.01
XQDA[81] (VGG-Face)	95.67	0.02
DDML[56]	90.68	-
TSML[157]	89.80	-
LM3L[57]	89.57	-

3.4.3 Person re-identification

In this subsection, we evaluate the performance of our methods for person re-identification, *i.e.*, recognizing a person by the pedestrian image at different locations and at different times [46]. We use the CUHK03 [76] and CUHK01 [75] databases to assess the performance of our methods.

CUHK03

CUHK03 database contains 14,096 pedestrian images which are taken from 1,467 persons by two cameras [76]. We randomly select 1,367 persons and use their images as the training set, and use the images from the rest 100 persons as the test set. For each person in the test set, we randomly select the images taken by one camera as the probe images, and use one of the images taken by another camera as

the gallery image. 20 partitions of training set and test sets are constructed, and the reported accuracies are averaged over all the partitions. We report the rank-1 accuracies and training time of PCML, NCML and the competing methods, *i.e.* ITML [33], DML-eig [149], LMNN [140], RANK [89], LDML [48], symmetry-driven accumulation of local features (SDALF) [39], eSDC [155], KISSME [69], XQDA [81], filter pairing neural network (FPNN) [76], Doublet-SVM [131] and Zhang *et al.* [151], on CUHK03 database with manually labeled and detected bounding boxes on single-shot setting in Table 3.4 and Table 3.5. For the methods with the rank-1 accuracy higher than 30%, we also report their CMC curves in Fig. 3.8. For FPNN [76], RANK [89], SDALF [39] and eSDC [155], we use the results in their original papers. As to the other methods, the results are obtained by using an effective feature representation named Local Maximal Occurrence (LOMO) [81]. One can see that the rank-1 accuracies of PCML and NCML are much higher than most of the competing methods, comparable to XQDA [81], but lower than Zhang *et al.* [151]. Note that Zhang *et al.* [151] learn nonlinear discriminative null space via kernelization, while what the other methods learned are Mahalanobis distance metric or linear subspace. And this might explain the superiority of Zhang *et al.* [151] over the other methods. Analogous to the results on LFW, the training time of PCML and NCML are much shorter than ITML [33], XQDA [81], Zhang *et al.* [151], LDML [48] and LMNN [140], comparable to DML-eig [149], and longer than Doublet-SVM [131] and KISSME [69].

Table 3.4 Rank-1 accuracies (%) on the CUHK03 database

Methods	CUHK03-Labeled	CUHK03-Detected
PCML (LOMO)	51.85	45.80
NCML (LOMO)	53.15	47.50
Doublet-SVM (LOMO) [131]	51.25	45.05
ITML (LOMO) [33]	46.40	43.25
DML-eig (LOMO) [149]	17.70	13.90
KISSME (LOMO) [69]	45.95	38.25
XQDA (LOMO) [81]	52.20	46.25
Zhang *et al.* (LOMO) [151]	58.90	53.70
LDML (LOMO) [48]	51.20	45.40
LMNN (LOMO) [140]	51.08	44.64
FPNN [76]	20.65	19.89
Euclidean (LOMO)	11.05	10.95
RANK [89]	10.42	8.52
SDALF [39]	5.60	4.87
eSDC [155]	8.76	7.68

Table 3.5 Training time (s) on the CUHK03 database with LOMO feature [81]

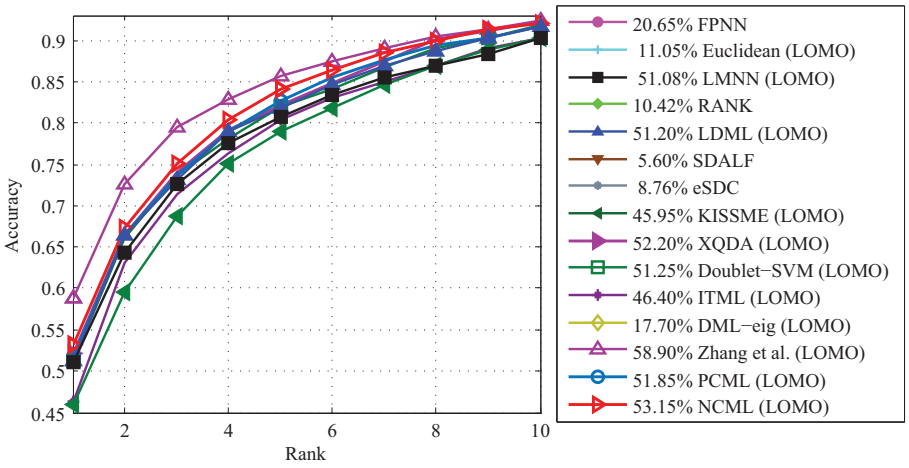
Methods	Training Time (s)
PCML	576.45
NCML	655.77
Doublet-SVM [131]	227.54
ITML [33]	1228.50
DML-eig [149]	523.33
KISSME [69]	0.85
XQDA [81]	902.35
Zhang *et al.* [151]	1954.60
LDML [48]	794.38
LMNN [140]	8383.60

3.5 Summary

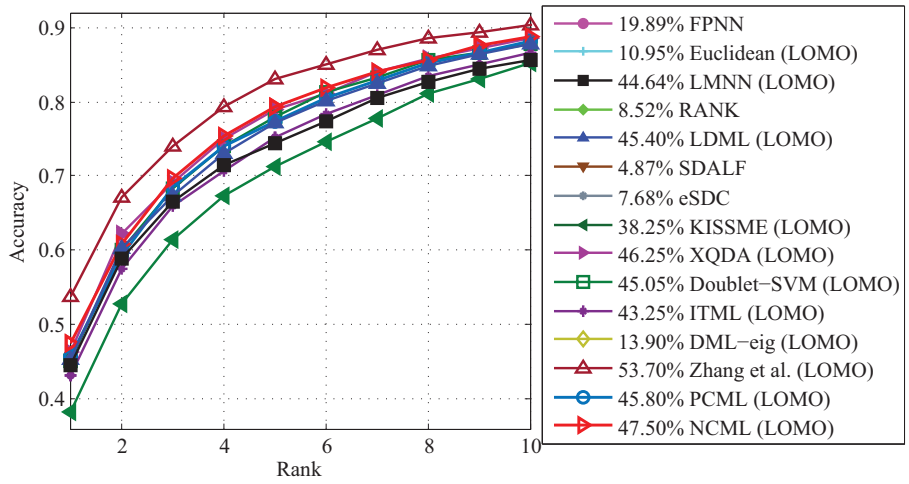
We proposed two distance metric learning models, namely PCML and NCML. The proposed models can guarantee the positive semidefinite property of the learned matrix $\mathbf{M}$, and can be solved efficiently by the existing SVM solvers. Experimental results on the handwritten digit recognition task showed that, compared with the state-of-the-art metric learning methods, including NCA [45], ITML [33], MCML [44], LDML [48], LMNN [140], PLML [134], DML-eig [149] and Doublet-SVM [131], the proposed PCML and NCML methods can not only achieve favorable classification accuracy, but also are efficient in training.

The experimental results on LFW, CUHK01 and CUHK03 databases indicate that the proposed methods also perform well in face verification and person re-identification. For face verification, PCML and NCML achieve higher or com-

parable accuracies to the competing methods on the LFW database. For person re-identification, our PCML and NCML can obtain better or comparable accuracy to most Mahalanobis distance metric learning or linear subspace methods, but are inferior to the kernelized subspace method by Zhang *et al.* [151].



(a)



(b)

Figure 3.8 The CMC curves of different methods on the CUHK03 database with (a) manually labeled bounding box and (b) automatically detected bounding box.

Chapter 4

Deep Similarity Learning via Combination of Single Image and Pairwise Image Representations for Person Re-identification

4.1 Introduction

Person re-identification aims at verifying whether the two images across camera views are from the same person [46]. It has been a hot topic in computer vision, and played an indispensable role in various surveillance applications [129, 130, 136]. However, person re-identification remains very challenging, and more studies are desired to alleviate the effect caused by the large variances in illumination, poses, viewpoints and background of pedestrian images.

Deep representation is crucial to the success of similarity learning methods, which play an important role in person re-identification. By exploiting the advantage of deep neural networks, the deep similarity learning can jointly learn the network parameters and the similarity measure. The deep similarity learning methods can be grouped into two categories. The first category is based on single image representation (SIR) of a given image [25, 35, 48, 54, 69, 74, 80, 88, 93]. Many similarity and metric learning methods have been proposed for the matching of handcrafted feature descriptors [25, 48, 54, 69, 74, 80–82, 93, 143, 154]. Recently, deep metric learning models have been proposed for the joint learning deep representation and distance metric [35, 147, 152]. For all these methods, a distance/similarity measure together with a threshold is deployed on the SIRs to predict whether two pedestrian images are matched or not. Given a gallery set of N images, their SIRs can be extracted in advance. In the matching stage, we only need to extract the SIR of the probe image and compute its distances to the SIRs of the gallery images. Thus, the SIR methods is usually very efficient in matching.

The second category of methods is based on the pairwise image representation (PIR), which is the representation of an image pair and is usually obtained by deep convolutional neural networks (CNNs) [1, 76, 86]. Using PIR, person re-identification can be accomplished by conducting an ordinary binary classification task [1, 28, 76, 86]. Different from the SIR-based methods, all the PIRs should be computed in the matching stage for the PIR-based method, *i.e.*, we need to extract the PIR between the probe image and each gallery image (*i.e.*, N times). Despite their computational inefficiency, PIR-based methods are effective in capturing the relationships between the two images. Under the PIR framework, several approach-

es have been suggested to address horizonal displacement by local patch matching [1, 76].

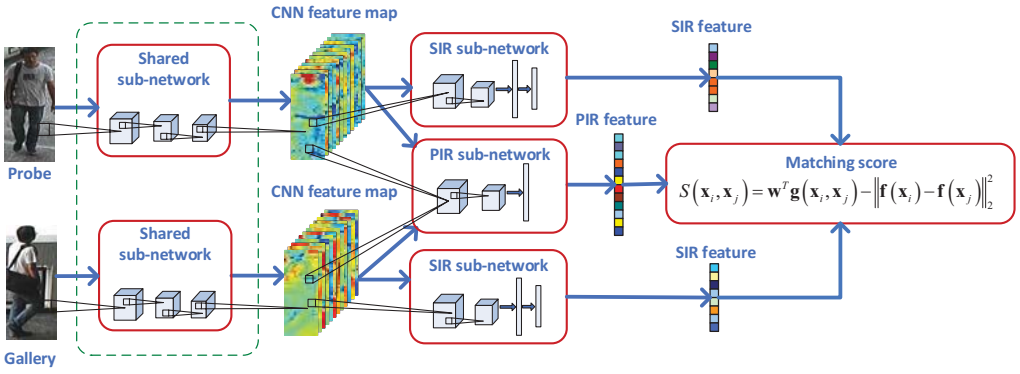


Figure 4.1 The sketch of the network for learning the single and pairwise image representations.

To sum up, both SIR and PIR have their respective advantages, which motivates us to develop a joint learning method to combine these two representations for better tradeoff between effectiveness and efficiency. To this end, we first analyze the connection between SIR and PIR. Denote by $\mathbf{x}_i$ and $\mathbf{x}_j$ two pedestrian images. We adopt $\left\|\mathbf{f}(\mathbf{x}_i)-\mathbf{f}\left(\mathbf{x}_j\right)\right\|_2^2$ to measure the dissimilarity between the SIRs $\mathbf{f}(\mathbf{x}_i)$ and $\mathbf{f}\left(\mathbf{x}_j\right)$. And $\mathbf{w}^T\mathbf{g}\left(\mathbf{x}_i,\mathbf{x}_j\right)$ is utilized as the classification score on PIR $\mathbf{g}(\mathbf{x}_i,\mathbf{x}_j)$. Here $\|\cdot\|_2$ denotes the L_2 norm, and $\mathbf{w}$ is the normal vector to the classification hyperplane. In this work, we analyze the connection between SIR and PIR. Let $\tilde{\mathbf{w}}=[\mathbf{I}]_{\text{vec}}$ and $\tilde{\mathbf{g}}\left(\mathbf{x}_i,\mathbf{x}_j\right)=[\left(\mathbf{f}(\mathbf{x}_i)-\mathbf{f}(\mathbf{x}_j)\right)\left(\mathbf{f}(\mathbf{x}_i)-\mathbf{f}(\mathbf{x}_j)\right)^T]_{\text{vec}}$, where $\mathbf{I}$ is the identity matrix, and $[\cdot]_{\text{vec}}$ denotes the vector form of a matrix. It is showed that the Euclidean distance based on SIR, *i.e.* $\left\|\mathbf{f}(\mathbf{x}_i)-\mathbf{f}\left(\mathbf{x}_j\right)\right\|_2^2$, can also be treated as a special case of classification on PIR, *i.e.* $\tilde{\mathbf{w}}^T\tilde{\mathbf{g}}\left(\mathbf{x}_i,\mathbf{x}_j\right)$. As illustrated in Sec. 4.2.1, Mahalanobis distance

and other distance or similarity measures [23, 80, 83] of SIR are also special cases of PIR.

The connection between SIR and PIR indicates that it is possible to incorporate SIR into the PIR model for better tradeoff between accuracy and efficiency. Thus, we suggest a generalized PIR model to combine SIR and PIR. As illustrated in Fig. 4.1, the final matching score by our model is the combination of the Euclidean distance on SIRs and the classification on PIR. We further develop a joint learning framework with deep CNN to exploit the advantages of these two representations. From Fig. 4.1, our proposed network consists of three sub-networks, *i.e.* one shared sub-network followed by two sub-networks for extracting SIR and PIR features, respectively. As the computational cost of extracting PIR feature is relatively high, we limit the depth of the PIR sub-network to include only one convolutional layer, one pooling layer and one fully-connected layer. In the test stage, we can store the CNN feature maps from the shared sub-network and SIR sub-network of the gallery images in advance. Thus, the shared feature maps and SIR of each probe image are required to be computed one time, and only the PIR sub-network is used to compute the PIR between the probe image and each gallery image. By using this network, we can fuse the SIR and PIR to improve the matching accuracy while maintaining the computational efficiency.

Furthermore, we extend our model by utilizing two different deep CNNs with the same architecture for joint SIR and PIR learning based on either pairwise comparison objective or triplet comparison objective, respectively. For the pairwise comparison based network, SIR and PIR are learned to make that the matching scores of the similar image pairs are higher than a given threshold and those of dis-

similar image pairs are lower than the threshold. For the triplet comparison based network, SIR and PIR are learned to make that the matching score of the similar image pairs are higher than that of the dissimilar ones. Finally, the matching scores of these two networks can be further combined to boost the person re-identification performance.

Experiments have been conducted on several public datasets for person re-identification, *i.e.* CUHK03 [76], CUHK01 [75] and VIPeR [47]. The results show that, joint SIR and PIR learning is effective in improving the person re-identification performance, and the matching accuracy can be further improved by combining the learned models based on pairwise and triplet comparison objectives. Compared with the state-of-the-art approaches, the proposed methods perform favorably in person re-identification.

The rest of this chapter is organized as follows. Section 4.2 describes the proposed model. Section 4.3 presents the deep CNN architecture and the training algorithms. Section 4.4 reports the experimental results, and Section 4.5 summarizes this chapter.

4.2 Joint SIR and PIR Learning

In this section, we first discuss the connection between SIR and PIR. Motivated by their connection, we present a generalized model by incorporating SIR into PIR. Then, two formulations (*i.e.* pairwise comparison formulation and triplet comparison formulation) are suggested for joint learning of SIR and PIR.

4.2.1 Connection between SIR and PIR

With the SIR features, there are five commonly used distance/similarity measures for person re-identification, *i.e.* Euclidean distance, Mahalanobis distance, joint Bayesian [23], locally adaptive decision function (LADF) [80] and generalized similarity measure (GSM) [83]. As explained in Sec. 4.1, Euclidean distance on SIRs can be regarded as a special case of PIR-based classification score, *i.e.* $\tilde{\mathbf{w}}^T\tilde{\mathbf{g}}\left(\mathbf{x}_i,\mathbf{x}_j\right)$. In the following, we will show that the other measures are also special cases of PIR-based classification score.

The Mahalanobis distance based on the SIR $\mathbf{z}_i = \mathbf{f}\left(\mathbf{x}_i\right)$ can be formulated as $s\left(\mathbf{x}_i,\mathbf{x}_j\right) = \left(\mathbf{z}_i-\mathbf{z}_j\right)^T\mathbf{M}\left(\mathbf{z}_i-\mathbf{z}_j\right)$, where $\mathbf{M}$ is positive semi-definite. This formulation is equivalent to $\tilde{\mathbf{w}}^T\tilde{\mathbf{g}}\left(\mathbf{x}_i,\mathbf{x}_j\right)$ when $\tilde{\mathbf{w}} = [\mathbf{M}]_{\text{vec}}$ and $\tilde{\mathbf{g}}\left(\mathbf{x}_i,\mathbf{x}_j\right) = \left[\left(\mathbf{z}_i-\mathbf{z}_j\right)\left(\mathbf{z}_i-\mathbf{z}_j\right)^T\right]_{\text{vec}}$.

The joint Bayesian formulation [23] is defined as follows

$$s\left(\mathbf{x}_i,\mathbf{x}_j\right) = \mathbf{z}_i^T\mathbf{A}\mathbf{z}_i + \mathbf{z}_j^T\mathbf{A}\mathbf{z}_j - 2\mathbf{z}_i^T\mathbf{G}\mathbf{z}_j, \tag{4.1}$$

which is the generalization of Mahalanobis distance. By setting $\tilde{\mathbf{w}} = \begin{pmatrix}[\mathbf{A}]_{\text{vec}}^T & [\mathbf{G}]_{\text{vec}}^T\end{pmatrix}^T$ and $\tilde{\mathbf{g}}\left(\mathbf{x}_i,\mathbf{x}_j\right) = \begin{pmatrix}\left[\mathbf{z}_i\mathbf{z}_i^T+\mathbf{z}_j\mathbf{z}_j^T\right]_{\text{vec}}^T & \left[-2\mathbf{z}_j\mathbf{z}_i^T\right]_{\text{vec}}^T\end{pmatrix}^T$ in $\tilde{\mathbf{w}}^T\tilde{\mathbf{g}}\left(\mathbf{x}_i,\mathbf{x}_j\right)$, joint Bayesian can be regarded as a classifier $\tilde{\mathbf{w}}$ on the PIR $\tilde{\mathbf{g}}(\mathbf{x}_i,\mathbf{x}_j)$.

The LADF [80] is defined as follows

$$s\left(\mathbf{x}_i,\mathbf{x}_j\right) = \frac{1}{2}\mathbf{z}_i^T\mathbf{A}\mathbf{z}_i + \frac{1}{2}\mathbf{z}_j^T\mathbf{A}\mathbf{z}_j + \mathbf{z}_i^T\mathbf{B}\mathbf{z}_j + \mathbf{c}^T\left(\mathbf{z}_i+\mathbf{z}_j\right) + b, \tag{4.2}$$

which is the generalization of Mahalanobis distance and joint Bayesian. It can also be viewed as a special case of $\tilde{\mathbf{w}}^T\tilde{\mathbf{g}}\left(\mathbf{x}_i,\mathbf{x}_j\right)$ when $\tilde{\mathbf{w}} = \begin{pmatrix}[\mathbf{A}]_{\text{vec}}^T & [\mathbf{B}]_{\text{vec}}^T & \mathbf{c}^T & b\end{pmatrix}^T$ and $\tilde{\mathbf{g}}(\mathbf{x}_i,\mathbf{x}_j) = \begin{pmatrix}\frac{1}{2}\left[\mathbf{z}_i\mathbf{z}_i^T+\mathbf{z}_j\mathbf{z}_j^T\right]_{\text{vec}}^T & \left[\mathbf{z}_j\mathbf{z}_i^T\right]_{\text{vec}}^T & \left(\mathbf{z}_i+\mathbf{z}_j\right)^T & 1\end{pmatrix}^T$.

The GSM [83] is defined as follows

$$s\left(\mathbf{x}_i, \mathbf{x}_j\right) = \mathbf{z}_i^T \mathbf{A} \mathbf{z}_i + \mathbf{z}_j^T \mathbf{B} \mathbf{z}_j + 2\mathbf{z}_i^T \mathbf{C} \mathbf{z}_j + 2\mathbf{d}^T \mathbf{z}_i + 2\mathbf{e}^T \mathbf{z}_j + f, \tag{4.3}$$

which can be explained as a generalization of Mahalanobis distance, joint Bayesian [23] and LADF [80] [83]. By setting $\tilde{\mathbf{w}} = ([\mathbf{A}]_{\text{vec}}^T \quad [\mathbf{B}]_{\text{vec}}^T \quad [\mathbf{C}]_{\text{vec}}^T \quad \mathbf{d}^T \quad \mathbf{e}^T \quad f)^T$ and $\tilde{\mathbf{g}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = ([\mathbf{z}_i\mathbf{z}_i^T]_{\text{vec}}^T \quad [\mathbf{z}_j\mathbf{z}_j^T]_{\text{vec}}^T \quad [2\mathbf{z}_i\mathbf{z}_j^T]_{\text{vec}}^T \quad 2\mathbf{z}_i^T \quad 2\mathbf{z}_j^T \quad 1)^T$, GSM also can be viewed as a special case of PIR-based matching score $\tilde{\mathbf{w}}^T \tilde{\mathbf{g}}\left(\mathbf{x}_i, \mathbf{x}_j\right)$.

In summary, most SIR-based methods can also be explained from the PIR perspective, but SIR does have its own advantage in terms of efficiency. For the SIR-based method, the SIR features of the gallery set can be precomputed in advance. For each probe image, we only need to extract its SIR and compute its distance/similarity measure with the precomputed SIRs from the gallery images, making SIR computationally efficient for person re-identification. On the other hand, PIR $\mathbf{g}\left(\mathbf{x}_i, \mathbf{x}_j\right)$ as a general representation of image pair also has its distinct advantage in modeling the complex relationships between the gallery and probe images.

4.2.2 Matching Score based on SIR and PIR

Motivated by the connection between SIR and PIR, we incorporate the SIR, resulting in the following general PIR-based matching score,

$$S\left(\mathbf{x}_i, \mathbf{x}_j\right) = \widehat{\mathbf{w}}^T \widehat{\mathbf{g}}\left(\mathbf{x}_i, \mathbf{x}_j\right). \tag{4.4}$$

Our generalized PIR includes two components, *i.e.* $\widehat{\mathbf{g}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left[\mathbf{g}^T\left(\mathbf{x}_i, \mathbf{x}_j\right) \quad -\tilde{\mathbf{g}}^T\left(\mathbf{x}_i, \mathbf{x}_j\right)\right]^T$. The weight $\widehat{\mathbf{w}}$ is formulated as $\widehat{\mathbf{w}} = \left[\mathbf{w}^T \quad \tilde{\mathbf{w}}^T\right]^T$. Here we following the Euclidean

distance on SIRs to set $\tilde{\mathbf{g}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = [(\mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_j))(\mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_j))^T]_{\text{vec}}$ and $\tilde{\mathbf{w}} = [\mathbf{I}]_{\text{vec}}$. Therefore, the matching score in Eqn. (4.4) can be derived as

$$S\left(\mathbf{x}_i, \mathbf{x}_j\right) = \mathbf{w}^T \mathbf{g}\left(\mathbf{x}_i, \mathbf{x}_j\right) - \left\|\mathbf{f}\left(\mathbf{x}_i\right) - \mathbf{f}\left(\mathbf{x}_j\right)\right\|_2^2, \tag{4.5}$$

which can be viewed as the matching score by combining both SIR and PIR. Thus, optimizing $S\left(\mathbf{x}_i, \mathbf{x}_j\right)$ from training data can offer an effective means for both joint learning and adaptive combination of SIR and PIR for person re-identification.

4.2.3 Pairwise Comparison Formulation

Based on the matching score in Eqn. (4.5), the pairwise comparison model formulation is proposed. Denote by $\left\{\left(\left(\mathbf{x}_i, \mathbf{x}_j\right), h_{ij}\right)\right\}$ the training set of sample pairs, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are the ith and jth training samples, respectively. h_{ij} is the label assigned to the pair $\left(\mathbf{x}_i, \mathbf{x}_j\right)$. If $\mathbf{x}_i$ and $\mathbf{x}_j$ are from the same class, then $h_{ij} = 1$, otherwise $h_{ij} = -1$. Let $\mathbf{f}\left(\mathbf{x}_i\right)$ and $\mathbf{f}\left(\mathbf{x}_j\right)$ be the SIRs of $\mathbf{x}_i$ and $\mathbf{x}_j$, $\mathbf{g}\left(\mathbf{x}_i, \mathbf{x}_j\right)$ be the PIR of $\left(\mathbf{x}_i, \mathbf{x}_j\right)$, and b be a distance threshold. In the pairwise comparison formulation, the matching score of the positive pair is expected to be higher than the threshold b, while the matching score of the negative pairs is expected to be lower than b. The matching score for any pair $\left(\mathbf{x}_i, \mathbf{x}_j\right)$ is enforced to satisfy the following constraints:

$$\begin{aligned} S\left(\mathbf{x}_i, \mathbf{x}_j\right) &\geq b + 1 - \xi_{ij}^P \quad \text{if } h_{ij} = 1 \\ S\left(\mathbf{x}_i, \mathbf{x}_j\right) &\leq b - 1 + \xi_{ij}^P \quad \text{if } h_{ij} = -1, \end{aligned} \tag{4.6}$$

where ξ_{ij}^P is a nonnegative slack variable which should be as small as possible. Then the loss function of pairwise comparison model is defined as

$$\mathcal{L}_P = \sum_{i,j} \left[1 + h_{ij}\left(b - S\left(\mathbf{x}_i, \mathbf{x}_j\right)\right)\right]_+, \tag{4.7}$$

where $[z]_+ = \max(z, 0)$.

4.2.4 Triplet Comparison Formulation

The triplet comparison formulation is trained on a series of triplets $\left(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k\right)$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are from the same class, while $\mathbf{x}_i$ and $\mathbf{x}_k$ are from different classes. To make the matching score of $\mathbf{x}_i$ and $\mathbf{x}_j$ higher than the one of $\mathbf{x}_i$ and $\mathbf{x}_k$, for any triplet $\left(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k\right)$, the matching score in Eqn. (4.5) should satisfy the following constraint:

$$S\left(\mathbf{x}_i, \mathbf{x}_j\right) - S\left(\mathbf{x}_i, \mathbf{x}_k\right) \geq 1 - \xi_{ijk}^T, \tag{4.8}$$

where ξ_{ijk}^T is a nonnegative slack variable which should be as small as possible. Then the loss function of our triplet comparison formulation is defined as

$$\mathcal{L}_T = \sum_{i,j,k} \left[1 - S\left(\mathbf{x}_i, \mathbf{x}_j\right) + S\left(\mathbf{x}_i, \mathbf{x}_k\right)\right]_+. \tag{4.9}$$

4.2.5 Prediction

After model training, $S\left(\mathbf{x}_i, \mathbf{x}_j\right)$ can be used to produce the matching score between $\mathbf{x}_i$ and $\mathbf{x}_j$. We combine the matching scores learned by pairwise and triplet comparison formulations, which are denoted by $S_P\left(\mathbf{x}_i, \mathbf{x}_j\right)$ and $S_T\left(\mathbf{x}_i, \mathbf{x}_j\right)$, respectively. The combined matching score is $S_{P\&T}\left(\mathbf{x}_i, \mathbf{x}_j\right) = S_P\left(\mathbf{x}_i, \mathbf{x}_j\right) + \mu S_T\left(\mathbf{x}_i, \mathbf{x}_j\right)$, where μ is a tradeoff parameter and we set it as $\mu = 0.5$ in the experiments. Finally we use the 1-Nearest Neighbor strategy to predict the identity of the probe image.

4.3 Deep Convolutional Neural Networks

In this section, we describe the deep CNN architecture and training algorithms for joint learning of SIR and PIR. We first introduce the network architecture of our

pairwise comparison model and triplet comparison model, and then propose their training algorithms, including model initialization and learning.

4.3.1 Network Architecture

Instead of designing the hand-crafted image features, we develop a framework to jointly learn the SIRs and PIRs with a deep CNN. For the pairwise comparison formulation, we learn the SIRs ($\mathbf{f}(\mathbf{x}_i)$ and $\mathbf{f}\left(\mathbf{x}_j\right)$) and PIR ($\mathbf{g}\left(\mathbf{x}_i, \mathbf{x}_j\right)$) for the image pair $\left(\mathbf{x}_i, \mathbf{x}_j\right)$. For the triplet comparison formulation, we learn the SIRs ($\mathbf{f}(\mathbf{x}_i)$, $\mathbf{f}\left(\mathbf{x}_j\right)$ and $\mathbf{f}(\mathbf{x}_k)$) and the PIRs ($\mathbf{g}\left(\mathbf{x}_i, \mathbf{x}_j\right)$ and $\mathbf{g}(\mathbf{x}_i, \mathbf{x}_k)$) for the image triplet $\left(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k\right)$. The deep architectures of the pairwise and triplet comparison models are illustrated in Fig. 4.2 and Fig. 4.3, respectively. Each of these two networks consists of a SIR learning sub-network (green part), a PIR learning sub-network (red part), and a sub-network shared by SIR and PIR learning (blue part). For each of the probe and gallery images, its CNN feature maps (yellow part) from the shared sub-network and the SIR feature are computed once. Only the PIR learning sub-network is used to extract the PIR features for each image pair of probe image and gallery image.

Shared sub-network

The sub-network in the blue part of Figs. 4.2 and 4.3 is shared by SIR learning and PIR learning. It consists of two convolutional layers with rectified linear unit (ReLU) activation. Each of them is followed by a pooling layer. The kernel sizes of the first and second convolutional layers are 5×5 and 3×3, respectively. The stride of the convolutional layers is 1 pixel. The kernel sizes of the first and second pooling layers are set to 3×3 and 2×2, respectively.

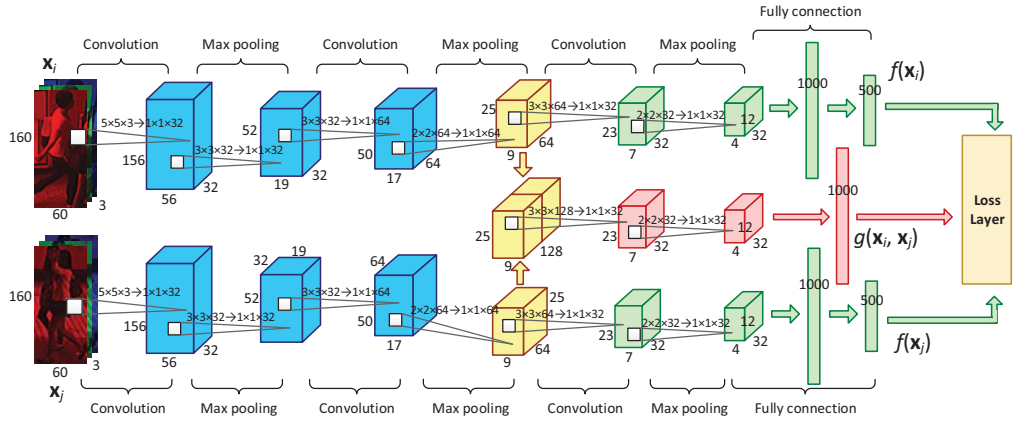


Figure 4.2 The proposed deep architecture of the pairwise comparison model (best viewed in color)

SIR sub-network

We use the sub-network in the green part of Figs. 4.2 and 4.3 to learn the SIR $\mathbf{f}(\mathbf{x}_i)$ for the input image $\mathbf{x}_i$. This sub-network contains one convolutional layers with ReLU activation, a pooling layer and two fully-connected layers. The kernel sizes of the convolutional layer and the pooling layer are 3×3 and 2×2. The output dimensions of these two fully-connected layers are 1000 and 500, respectively. For the pairwise and triplet comparison model, there are two and three sub-networks, which share the same parameter, to learn the SIR, respectively.

PIR sub-network

We use the sub-network in the red part of Figs. 4.2 and 4.3 to learn the PIR $\mathbf{g}\left(\mathbf{x}_i, \mathbf{x}_j\right)$ for the input image pair $\left(\mathbf{x}_i, \mathbf{x}_j\right)$. This sub-network contains one convolutional layer with ReLU activation followed by one pooling layer and one fully-connected layer.

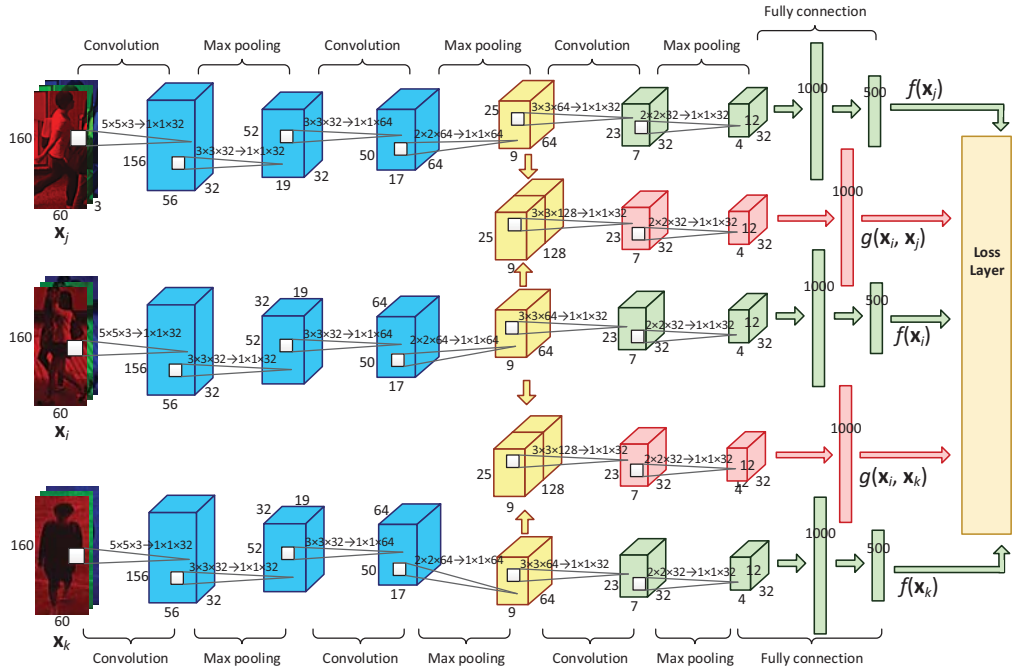


Figure 4.3 The proposed deep architecture of the triplet comparison model (best viewed in color)

The kernel sizes of the convolutional layer and the pooling layer are 3×3 and 2×2. The output dimension of the fully-connected layer is 1000. Denote by $\phi_p(\mathbf{x}_i)$ the pth channel of the CNN feature map of $\mathbf{x}_i$ from the shared sub-network. When we extract the PIR of $(\mathbf{x}_i, \mathbf{x}_j)$, the PIR sub-network is feeded by the CNN feature maps of $\mathbf{x}_i$ and $\mathbf{x}_j$ from the shared sub-network. The first convolutional layer of PIR sub-network is used to compute the pairwise image feature map as follows

$$\varphi_r(\mathbf{x}_i, \mathbf{x}_j) = \max\left(0, b_r + \sum_q \mathbf{k}_{q,r} * \phi_q(\mathbf{x}_i) + \mathbf{l}_{q,r} * \phi_q(\mathbf{x}_j)\right), \tag{4.10}$$

where $\varphi_r(\mathbf{x}_i, \mathbf{x}_j)$ is the rth channel of pairwise image feature map, $\mathbf{k}_{q,r}$ and $\mathbf{l}_{q,r}$ are different convolutional kernels of the qth channel of the shared sub-network feature

map and the rth channel of pairwise image feature map. The similar operation has also been used in [86].

4.3.2 Initialization

Before the training process, we need to preprocess the training data and generate the doublets or triplets. The details are described as follows.

Data preprocessing. First, we resize all the input images to 180×80 pixels. To make the model robust to the image translation variance, we randomly crop the input images before the training process. We randomly select the cropped image center from $[80, 100] \times [30, 50]$ and crop the original image to 160×60 pixels. We also augment the training set by generating the horizontal mirror image of each sample.

Doublet/triplet generation based on mini-batch strategy. Since the training set may be too large to be loaded into the memory, we divide the training set into multiple mini-batches. Following the strategy in [35], for each iteration, we randomly select 80 classes from the training set, and construct 60 doublets or triplets for each class. Using this strategy, we can generate 4,800 doublets or triplets in each round of training, and we randomly select 500 doublets or triplets for training.

4.3.3 Network Training

In this subsection, we present the algorithms for learning network parameters of pairwise/triplet comparison models. The network parameters are denoted by $\omega = [\omega_{\text{SIR}}, \omega_{\text{PIR}}, \omega_{\text{S}}]$, where ω_{SIR}, ω_{PIR} and ω_{S} are the parameters of SIR, PIR and shared

sub-networks, respectively. To use BP to learn the network parameters ω, we compute the gradients of the loss functions $\mathcal{L}_P$ and $\mathcal{L}_T$ with respect to ω. In the following, we introduce the learning methods for the SIR sub-network, the PIR sub-network and the shared sub-network, respectively.

SIR sub-network

For the pairwise comparison model, the gradient of the loss function $\mathcal{L}_P$ with respect to network parameter ω_{SIR} is

$$\mathbf{P}_{\text{S}}(\omega_{\text{SIR}}) = \frac{\partial \mathcal{L}_P}{\partial \omega_{\text{SIR}}} = \sum_{i,j} \mathbf{r}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\text{SIR}}). \tag{4.11}$$

In Eqn. (4.11), if $h_{ij}S(\mathbf{x}_i, \mathbf{x}_j) \geq h_{ij}b + 1$, then $\mathbf{r}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\text{SIR}}) = \mathbf{0}$, otherwise

$$\mathbf{r}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\text{SIR}}) = 2h_{ij}\mathbf{R}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\text{SIR}})(\mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_j)), \tag{4.12}$$

where $\mathbf{R}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\text{SIR}}) = \frac{\partial \mathbf{f}^T(\mathbf{x}_i)}{\partial \omega_{\text{SIR}}} - \frac{\partial \mathbf{f}^T(\mathbf{x}_j)}{\partial \omega_{\text{SIR}}}$, which can be computed by BP, and $\mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_j)$ can be obtained by forward propagation.

For the triplet comparison model, the gradient of the loss function $\mathcal{L}_T$ with respect to network parameter ω_{SIR} is

$$\mathbf{T}_{\text{S}}(\omega_{\text{SIR}}) = \frac{\partial \mathcal{L}_T}{\partial \omega_{\text{SIR}}} = \sum_{i,j,k} \mathbf{r}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\text{SIR}}). \tag{4.13}$$

In Eqn. (4.13), if $S(\mathbf{x}_i, \mathbf{x}_j) - S(\mathbf{x}_i, \mathbf{x}_k) \geq 1$, then $\mathbf{r}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) = \mathbf{0}$, otherwise

$$\mathbf{r}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\text{SIR}}) = 2\mathbf{R}(\mathbf{x}_k, \mathbf{x}_j, \omega_{\text{SIR}})\mathbf{f}(\mathbf{x}_i) + 2\mathbf{R}(\mathbf{x}_j, \mathbf{x}_i, \omega_{\text{SIR}})\mathbf{f}(\mathbf{x}_j) + 2\mathbf{R}(\mathbf{x}_i, \mathbf{x}_k, \omega_{\text{SIR}})\mathbf{f}(\mathbf{x}_k). \tag{4.14}$$

We can use BP to compute $\mathbf{R}(\mathbf{x}_k, \mathbf{x}_j, \omega_{\text{SIR}})$, $\mathbf{R}(\mathbf{x}_j, \mathbf{x}_i, \omega_{\text{SIR}})$ and $\mathbf{R}(\mathbf{x}_i, \mathbf{x}_k, \omega_{\text{SIR}})$, and use forward propagation to get $\mathbf{f}(\mathbf{x}_i)$, $\mathbf{f}(\mathbf{x}_j)$ and $\mathbf{f}(\mathbf{x}_k)$. So the gradient in Eqn. (4.13) can be obtained.

PIR sub-network

For the pairwise comparison model, the gradient of the loss function $\mathcal{L}_P$ with respect to network parameter ω_{PIR} is

$$\mathbf{P}_{\text{C}}(\omega_{\text{PIR}}) = \frac{\partial \mathcal{L}_P}{\partial \omega_{\text{PIR}}} = \sum_{i,j} \mathbf{q}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\text{PIR}}). \tag{4.15}$$

In Eqn. (4.15), if $h_{ij}S(\mathbf{x}_i, \mathbf{x}_j) \geq h_{ij}b + 1$, then $\mathbf{q}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\text{PIR}}) = \mathbf{0}$, otherwise

$$\mathbf{q}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\text{PIR}}) = -h_{ij}\frac{\partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_j)}{\partial \omega_{\text{PIR}}}\mathbf{w}. \tag{4.16}$$

As the term $\frac{\partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_j)}{\partial \omega_{\text{PIR}}}$ can be computed using BP, the gradient in Eqn. (4.15) can be obtained.

For the triplet comparison model, the gradient of the loss function $\mathcal{L}_T$ with respect to network parameter ω_{PIR} is

$$\mathbf{T}_{\text{C}}(\omega_{\text{PIR}}) = \frac{\partial \mathcal{L}_T}{\partial \omega_{\text{PIR}}} = \sum_{i,j,k} \mathbf{q}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\text{PIR}}). \tag{4.17}$$

In Eqn. (4.17), if $S(\mathbf{x}_i, \mathbf{x}_j) - S(\mathbf{x}_i, \mathbf{x}_k) \geq 1$, then $\mathbf{q}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\text{PIR}}) = \mathbf{0}$, otherwise

$$\mathbf{q}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\text{PIR}}) = \left(\frac{\partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_k)}{\partial \omega_{\text{PIR}}} - \frac{\partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_j)}{\partial \omega_{\text{PIR}}}\right)\mathbf{w}. \tag{4.18}$$

In Eqn. (4.18), the term $\frac{\partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_k)}{\partial \omega_{\text{PIR}}} - \frac{\partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_j)}{\partial \omega_{\text{PIR}}}$ can be computed using BP. Thus the gradient in Eqn. (4.15) can be obtained.

Shared sub-network

As the parameters of shared sub-network are utilized to learn both the SIR and PIR, the gradients of $\mathcal{L}_P$ and $\mathcal{L}_T$ with respect to ω_{S} are

$$\frac{\partial \mathcal{L}_P}{\partial \omega_{\text{S}}} = \sum_{i,j} \Big(\mathbf{r}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\text{S}}) + \mathbf{q}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\text{S}})\Big), \tag{4.19}$$

$$\frac{\partial \mathcal{L}_T}{\partial \omega_{\mathrm{S}}} = \sum_{i,j,k} \Big(\mathbf{r}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{S}}) + \mathbf{q}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{S}}) \Big). \tag{4.20}$$

The calculation of $\mathbf{r}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{S}})$, $\mathbf{q}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{S}})$, $\mathbf{r}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{S}})$ and $\mathbf{q}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{S}})$ are shown in Eqns. (4.12), (4.16), (4.14) and (4.18), respectively. Substituting them into Eqns. (4.19) and (4.20), the gradients of $\mathcal{L}_P$ and $\mathcal{L}_T$ to ω_{S} can be calculated. Finally, the training algorithm of pairwise and triplet comparison models are summarized in Algorithms 1 and 2, respectively.

4.4 Experiments

In this section, we evaluate the proposed method using three person re-identification datasets, *i.e.* CUHK03 [76][1], CUHK01 [75][1] and VIPeR [47][2]. The proposed method is implemented based on the Caffe framework [62]. We set the momentum as $\gamma = 0.5$ and set the weight decay as $\mu = 0.0005$. We train the network for 200,000 iterations. It takes about 63 hours in training with a NVIDIA Tesla K40 GPU. The learning rates of pairwise and triplet comparison models are 1×10^{-3} and 3×10^{-4} before the 100,000th iteration, respectively. After that their learning rates reduce to 1×10^{-4} and 3×10^{-5}. All of the reported results are based on the single-shot setting.

4.4.1 CUHK03 Dataset

The CUHK03 dataset contains 14,096 pedestrian images, which were taken from 1,467 persons by two surveillance cameras [76]. Each person has 4.8 images on

[1] http://www.ee.cuhk.edu.hk/~rzhao/

[2] http://vision.soe.ucsc.edu/projects

Algorithm 3 Algorithm of Pairwise Comparison Model

Input: Doublet training set $\left\{\left(\left(\mathbf{x}_i, \mathbf{x}_j\right), h_{ij}\right)\right\}$, iteration number T.

Output: Network parameter ω.

Initialize ω, $t \leftarrow 0$.
repeat
  $\mathbf{P}_{\mathrm{S}}(\omega_{\mathrm{SIR}}) \leftarrow \mathbf{0}$, $\mathbf{P}_{\mathrm{C}}(\omega_{\mathrm{PIR}}) \leftarrow \mathbf{0}$, $\frac{\partial \mathcal{L}_P}{\partial \omega_{\mathrm{S}}} \leftarrow \mathbf{0}$.
  for each $(\mathbf{x}_i, \mathbf{x}_j)$ **do**
    Calculate $\mathbf{f}(\mathbf{x}_i)$ and $\mathbf{f}(\mathbf{x}_j)$ by forward propagation.
    Calculate $\mathbf{R}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{SIR}})$, $\frac{\partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_j)}{\partial \omega_{\mathrm{PIR}}}$, $\mathbf{R}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{S}})$ and $\frac{\partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_j)}{\partial \omega_{\mathrm{S}}}$ by BP.
    Calculate $\mathbf{r}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{SIR}})$, $\mathbf{q}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{PIR}})$, $\mathbf{r}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{S}})$ and $\mathbf{q}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{S}})$ by Eqns. (4.12) and (4.16).
    $\mathbf{P}_{\mathrm{S}}(\omega_{\mathrm{SIR}}) \leftarrow \mathbf{P}_{\mathrm{S}}(\omega_{\mathrm{SIR}}) + \mathbf{r}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{SIR}})$
    $\mathbf{P}_{\mathrm{C}}(\omega_{\mathrm{PIR}}) \leftarrow \mathbf{P}_{\mathrm{C}}(\omega_{\mathrm{PIR}}) + \mathbf{q}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{PIR}})$
    $\frac{\partial \mathcal{L}_P}{\partial \omega_{\mathrm{S}}} \leftarrow \frac{\partial \mathcal{L}_P}{\partial \omega_{\mathrm{S}}} + \mathbf{r}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{S}}) + \mathbf{q}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \omega_{\mathrm{S}})$
  end for
  $\omega_{\mathrm{SIR}} \leftarrow \omega_{\mathrm{SIR}} - \theta \mathbf{P}_{\mathrm{S}}(\omega_{\mathrm{SIR}})$
  $\omega_{\mathrm{PIR}} \leftarrow \omega_{\mathrm{PIR}} - \theta \mathbf{P}_{\mathrm{C}}(\omega_{\mathrm{PIR}})$
  $\omega_{\mathrm{S}} \leftarrow \omega_{\mathrm{S}} - \theta \frac{\partial \mathcal{L}_P}{\partial \omega_{\mathrm{S}}}$
  $t \leftarrow t + 1$
until $t > T$
return $\omega = [\omega_{\mathrm{SIR}}, \omega_{\mathrm{PIR}}, \omega_{\mathrm{S}}]$.

average. All of the images are collected from five video clips. The dataset provides both the manually labeled bounding box and the automatically detected bounding box with a pedestrian detector [40]. Following the testing protocol in [76], the

Algorithm 4 Algorithm of Triplet Comparison Model

Input: Triplet training set $\left\{\left(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k\right)\right\}$, iteration number T.

Output: Network parameter ω.

Initialize ω, $t \leftarrow 0$.
repeat
  $\mathbf{T}_{\mathrm{S}}(\omega_{\mathrm{SIR}}) \leftarrow \mathbf{0}$, $\mathbf{T}_{\mathrm{C}}(\omega_{\mathrm{PIR}}) \leftarrow \mathbf{0}$ and $\frac{\partial \mathcal{L}^T}{\partial \omega_{\mathrm{S}}} \leftarrow \mathbf{0}$.
  for each $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)$ **do**
    Calculate $\mathbf{f}(\mathbf{x}_i)$, $\mathbf{f}(\mathbf{x}_j)$, and $\mathbf{f}(\mathbf{x}_k)$ by forward propagation.
    Calculate $\mathbf{R}(\mathbf{x}_k, \mathbf{x}_j, \omega_{\mathrm{SIR}})$, $\mathbf{R}(\mathbf{x}_j, \mathbf{x}_i, \omega_{\mathrm{SIR}})$, $\mathbf{R}(\mathbf{x}_i, \mathbf{x}_k, \omega_{\mathrm{SIR}})$, $\mathbf{R}(\mathbf{x}_k, \mathbf{x}_j, \omega_{\mathrm{S}})$, $\mathbf{R}(\mathbf{x}_j, \mathbf{x}_i, \omega_{\mathrm{S}})$, $\mathbf{R}(\mathbf{x}_i, \mathbf{x}_k, \omega_{\mathrm{S}})$, $\frac{\partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_k) - \partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_j)}{\partial \omega_{\mathrm{PIR}}}$ and $\frac{\partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_k) - \partial \mathbf{g}^T(\mathbf{x}_i, \mathbf{x}_j)}{\partial \omega_{\mathrm{S}}}$ by BP.
    Calculate $\mathbf{r}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{SIR}})$, $\mathbf{q}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{PIR}})$, $\mathbf{r}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{S}})$ and $\mathbf{q}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{S}})$ by Eqns. (4.14) and (4.18).
    $\mathbf{T}_{\mathrm{S}}(\omega_{\mathrm{SIR}}) \leftarrow \mathbf{T}_{\mathrm{S}}(\omega_{\mathrm{SIR}}) + \mathbf{r}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{SIR}})$.
    $\mathbf{T}_{\mathrm{C}}(\omega_{\mathrm{PIR}}) \leftarrow \mathbf{T}_{\mathrm{C}}(\omega_{\mathrm{PIR}}) + \mathbf{q}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{PIR}})$.
    $\frac{\partial \mathcal{L}^T}{\partial \omega_{\mathrm{S}}} \leftarrow \frac{\partial \mathcal{L}^T}{\partial \omega_{\mathrm{S}}} + \mathbf{r}_{\mathbf{T}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{S}}) + \mathbf{q}_{\mathbf{P}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \omega_{\mathrm{S}})$
  end for
  $\omega_{\mathrm{SIR}} \leftarrow \omega_{\mathrm{SIR}} - \theta \mathbf{T}_{\mathrm{S}}(\omega_{\mathrm{SIR}})$
  $\omega_{\mathrm{PIR}} \leftarrow \omega_{\mathrm{PIR}} - \theta \mathbf{T}_{\mathrm{C}}(\omega_{\mathrm{PIR}})$
  $\omega_{\mathrm{S}} \leftarrow \omega_{\mathrm{S}} - \theta \frac{\partial \mathcal{L}^T}{\partial \omega_{\mathrm{S}}}$
  $t \leftarrow t + 1$
until $t > T$
return $\omega = [\omega_{\mathrm{SIR}}, \omega_{\mathrm{PIR}}, \omega_{\mathrm{S}}]$.

identities in this dataset are randomly divided into non-overlapping training and test set. The training set consists of 1,367 persons and the test set consists of 100

persons. By this strategy, 20 partitions of training and test set are constructed. The reported cumulative matching characteristic (CMC) curve and accuracy are averaged by these 20 groups. For each person in the test set, we randomly select one camera view to construct the probe set, and use one image from another camera view to form the gallery set.

Comparison of different model settings: To evaluate the effect of joint SIR and PIR learning [130], we design two models for comparison, the matching scores of which only consist of the SIR-based and PIR-based matching scores, respectively. We report the accuracies of SIR-based, PIR-based and fused similarities of the proposed pairwise and triplet comparison models in Table 4.2. From the results, we can see that the SIR and PIR-based matching have comparable results. However, their combination achieves a higher accuracy than either of them. The accuracy of triplet comparison model is higher than pairwise comparison model, and their combination also outperforms both of them. We also report the training time of the proposed model in Table 4.3. Compared with SIR learning, the proposed joint SIR and PIR learning model can achieve substantial improvement of matching accuracy with moderate increase of training time.

Table 4.1 The rank-1 accuracies (%) of the proposed pairwise and triplet comparison models on CUHK03 dataset with detected bounding box

Model	SIR	PIR	Fused
Pairwise	38.25	36.10	44.94
Triplet	45.16	45.32	53.57
Combination	46.29	47.44	55.08

Comparison of alternative network architectures: We compare our proposed

Table 4.2 The rank-1 accuracies (%) of the proposed pairwise and triplet comparison models on CUHK03 dataset with labeled bounding box

Model	SIR	PIR	Fused
Pairwise	43.38	43.29	50.20
Triplet	50.38	49.00	58.00
Combination	51.85	51.78	59.73

Table 4.3 The training times of the proposed pairwise and triplet comparison models

Model	SIR	SIR&PIR
Pairwise	18h20m	24h10m
Triplet	24h36m	38h17m

network architecture with two alternative networks, which are denoted by *A* and *B*. The architectures of these two networks are shown in Table 4.4. To simplify the notation, we denote the convolution layer, pooling layer and fully connected layer by "CONV<receptive field size>– <number of channels>", "POOL<receptive field size>" and "FC–<number of dimensions>", respectively. Compared with our network, *A* has a shallower shared sub-network and deeper SIR and PIR sub-networks, while *B* has a deeper shared sub-network and shallower SIR and PIR sub-networks. We use the CUHK03 dataset with manually labeled bounding box to evaluate these networks. Fig. 4.4 shows the rank-1 accuracies, training and testing time of them. We can see that our proposed network achieves better accuracy than the other two networks. The training time and testing time of our proposed network are shorter than those of network *A*, and slightly longer than those of network *B*.

Comparison of joint learning and separate learning procedures: To prove

Table 4.4 The architectures of our proposed network together with two similar networks.

Abbreviation	*A*	Ours	*B*
Shared sub-network	CONV5-32	CONV5-32	CONV5-32
	POOL3	POOL3	POOL3
		CONV3-64	CONV3-64
		POOL2	POOL2
			CONV3-32
			POOL2
SIR sub-network	CONV3-64	CONV3-32	FC-1000
	POOL2	POOL2	FC-500
	CONV3-32	FC-1000	
	POOL2	FC-500	
	FC-1000		
	FC-500		
PIR sub-network	CONV3-64	CONV3-32	FC-1000
	POOL2	POOL2	
	CONV3-32	FC-1000	
	POOL2		
	FC-1000		

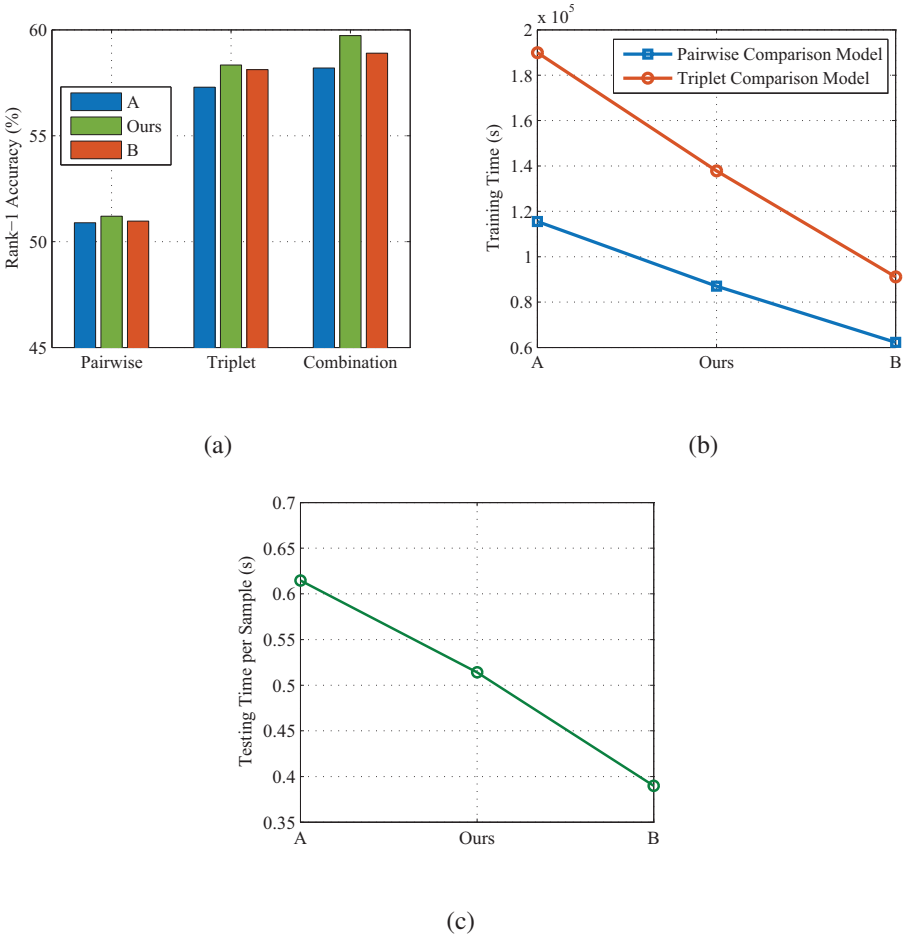


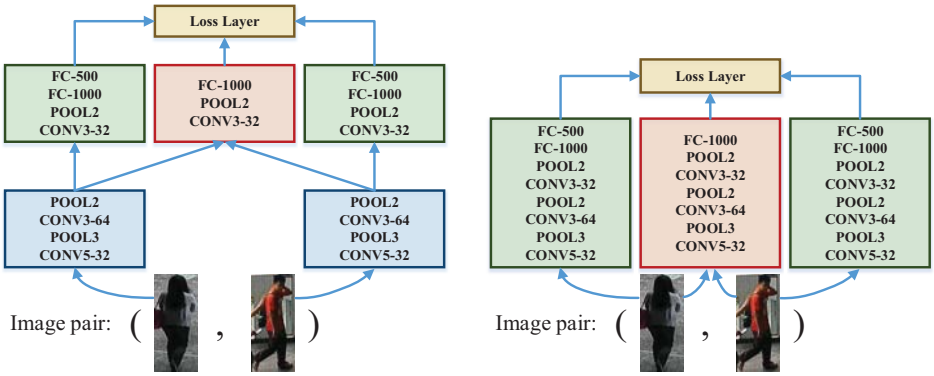
Figure 4.4 The rank-1 accuracies, training and testing time of three similar networks. (a) is the rank-1 accuracies of pairwise comparison, triplet comparison and combined models. (b) is the training time of pairwise and triplet comparison models. (c) is the testing time of these networks.

the effectiveness of the joint learning procedure, we compare our joint learning network with the separate learning network, which learns the SIR and PIR with two independent networks. The network sketches of joint learning and separate learning networks are shown in Fig. 4.5 (a) and (b). We compared their CMC curves and rank-1 accuracies under the CUHK03 dataset with manually labeled bounding box. The results are shown in Fig. 4.5 (c). From the results, we can see that our proposed joint learning network can achieve higher accuracy than the separate learning network.

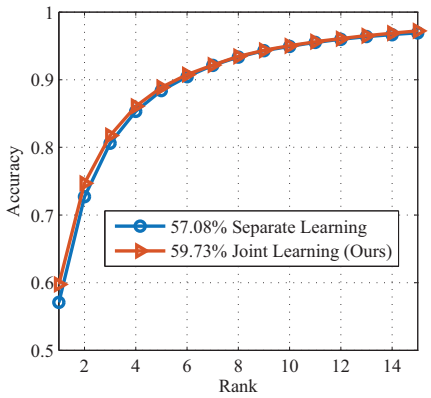
Comparison with other state-of-the-art methods: We also compare the performances of the proposed method and some other state-of-the-art methods, including improved deep learning architecture (IDLA) [1], Cross-view Quadratic Discriminant Analysis (XQDA) [81], MLAPG [82], general similarity measure (GSM) [83] and Discriminative Null Space (DNS) [151]. Fig. 4.6 and Fig. 4.7 illustrate the CMC curves and the rank-1 accuracies of these methods on CUHK03 dataset with detected and labeled bounding boxes, respectively. As the CMC curve of GSM on the CUHK03 dataset with detected bounding box are not released, we do not report it in Fig. 4.6. We can see that the rank-1 accuracies of the proposed method can reach 55.08% (for detected bounding box) and 59.73% (for labeled bounding box), which are 1.38% and 0.83% higher than the method with the second best performance, respectively.

4.4.2 CUHK01 Dataset

The CUHK01 dataset consists of 3,884 pedestrian images taken by two surveillance cameras from 971 persons. Each person has 4 images. This dataset has been ran-



(a) Joint Learning Network (Ours) (b) Separate Learning Network



(c) CMC Curves

Figure 4.5 The comparison of the structures and CMC curves of joint learning network and separate learning network. (a) and (b) are the sketch architectures of joint learning and separate learning networks. (c) is the CMC curves of these two networks.

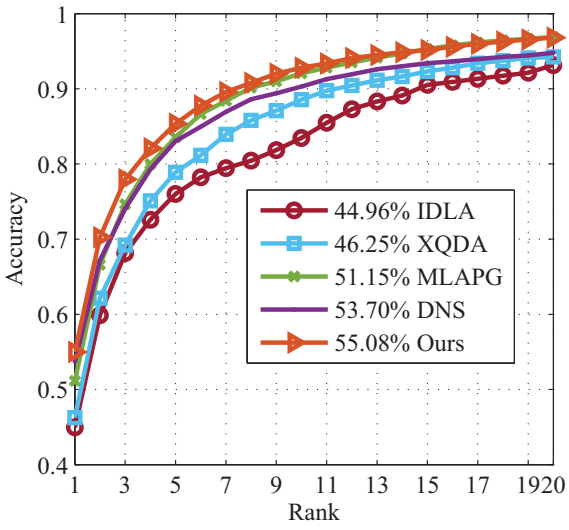


Figure 4.6 The rank-1 accuracies and CMC curves of different methods on the CUHK03 dataset with detected bounding box (best viewed in color)

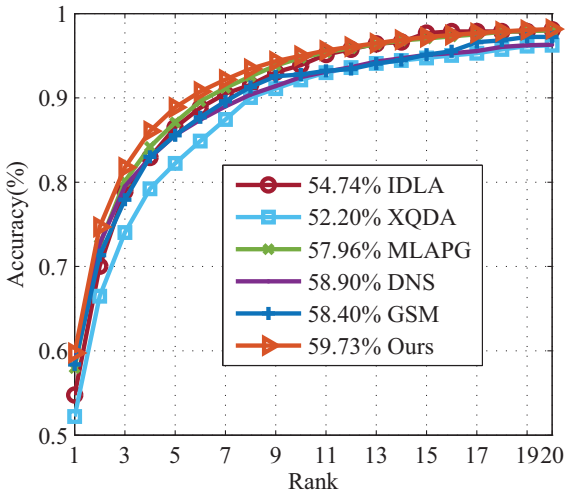


Figure 4.7 The rank-1 accuracies and CMC curves of different methods on the CUHK03 dataset with labeled bounding box (best viewed in color)

domly divided into 10 partitions of training and test sets, and the reported CMC curves and rank-1 accuracies are averaged on these 10 groups.

Following the protocol in [1], we use 871 persons for training and 100 persons for testing. On the basis of the single-shot setting, we report the CMC curves and rank-1 accuracies of the proposed model and the other state-of-the-art person re-identification methods, including FPNN [76], KISSME [69], IDLA [1] and GSM [83] in Fig. 4.8. The rank-1 accuracy of the proposed method is higher than the other competing methods.

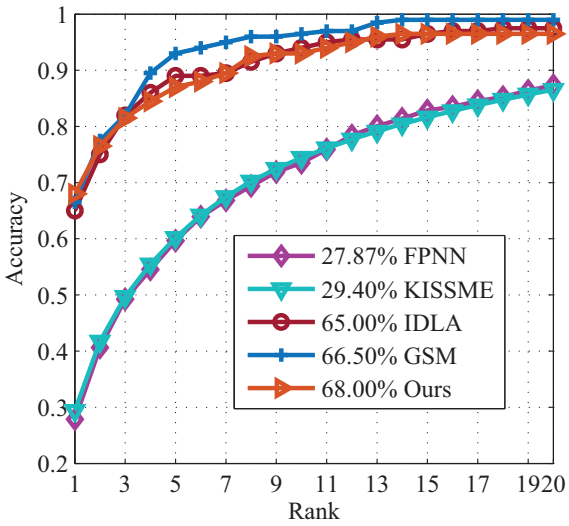


Figure 4.8 The rank-1 accuracies and CMC curves of different methods on the CUHK01 dataset (best viewed in color)

4.4.3 VIPeR Dataset

The VIPeR dataset consists of 1,264 images from 632 persons [47]. These images are taken by two camera views. We randomly select 316 persons for training, and use the rest 316 persons for testing. For each person in the test set, we randomly select one camera view as the probe set, and use the other camera view as the gallery set. Following the testing protocol in [1], we pretrain the CNN using CUHK03 and CUHK01 datasets, and fine-tune the network on the training set of VIPeR. We report the CMC curves and rank-1 accuracies of LADF [80], mid-level filters (mFilter) [156], visWord [153], saliency matching (SalMatch) [154], IDLA [1], XQDA [81], MLAPG [82], DNS [151] and the proposed model. The proposed method performs better than most of the other competing methods. Although the rank-1 accuracies of our method is lower than that of XQDA [81], MLAPG [82] and DNS [151], the CMC curves of our method are comparable to them. Our performance is lower than mFilter [156]+LADF [80] which is the combination of two methods, and higher than either one of mFilter [156] and LADF [80].

4.5 Summary

In this work, we propose a joint SIR and PIR learning approach for deep similarity learning. SIR is efficient in learning and matching, and PIR is effective in modeling the relationship between two images. Based on their connection, we suggest a generalized PIR model to utilize the advantages of both SIR and PIR. Based on the proposed matching score, we present a pairwise comparison model and a triplet comparison model for joint SIR and PIR learning. For each of these two models,

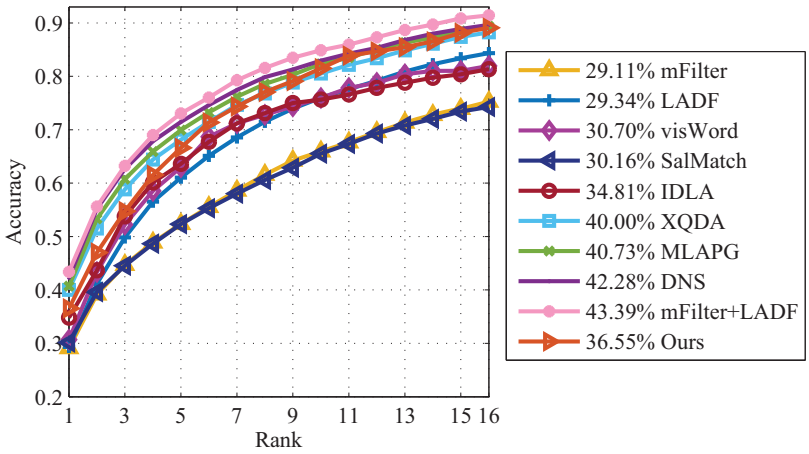


Figure 4.9 The rank-1 accuracies and CMC curves of different methods on the VIPeR dataset (best viewed in color)

we formulate a deep CNN to jointly learn the SIR and PIR. Experimental results demonstrate the effectiveness of joint SIR and PIR learning and show the promising results of our proposed models in person re-identification.

Chapter 5

Learning of Single Relative Order Relationship by Deep Siamese Networks

5.1 Introduction

Many computer vision tasks aim to learn the pairwise relationship of images. Given an image pair $(\mathbf{x}, \mathbf{y})$, they learn a function to predict whether it belongs to a particular pairwise relationship. For example, the similarity learning methods, which have played an important role in many computer vision tasks [15, 48, 81, 82], aim to learn the similarity relationship between two images, in which the images from the same class are more similar, and the images from different classes are less similar [8]. So the similarity can be regarded as a special case of pairwise relationship by defining the prediction function as $s(\mathbf{x}, \mathbf{y})$, which denotes the similarity of $\mathbf{x}$ and $\mathbf{y}$.

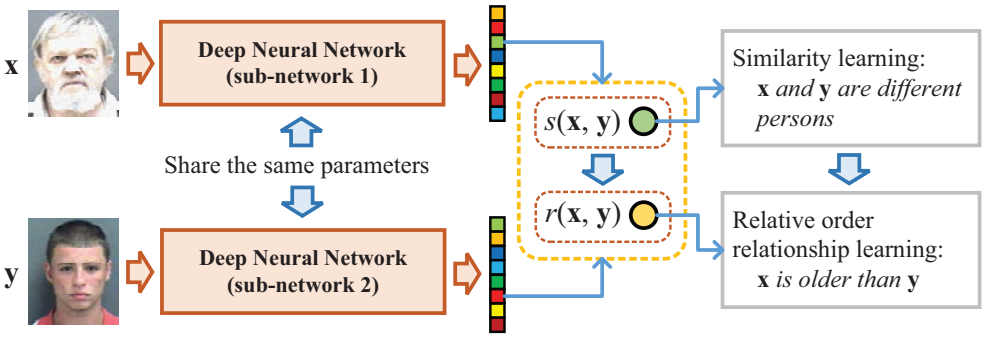


Figure 5.1 Extension of deep siamese network for relative order relationship learning. In previous work, siamese network has been applied to similarity learning by matching the deep features from two samples with the similarity function $s(\mathbf{x}, \mathbf{y})$. By utilizing the deep siamese network architecture, we replace the symmetric $s(\mathbf{x}, \mathbf{y})$ with the antisymmetric relative order prediction function $r(\mathbf{x}, \mathbf{y})$ for relative order relationship learning.

In recent years, deep siamese network has been successfully applied in similarity learning [83, 112, 150]. As illustrated in Fig. 5.1, there are two sub-networks in the deep siamese network which have the same architecture and share the same network parameters. With two images $\mathbf{x}$ and $\mathbf{y}$ feeded into the deep siamese network, the two sub-networks extract the deep features of $\mathbf{x}$ and $\mathbf{y}$, respectively. Then the deep features are matched by the similarity function $s(\mathbf{x}, \mathbf{y})$ in the top layer.

It is clear that similarity is the symmetric relationship since $s(\mathbf{x}, \mathbf{y}) = s(\mathbf{y}, \mathbf{x})$. However, in some applications, the relationship of two images is not symmetric. For example, the relative attribute ranking task aims to predict the relative order of attributes, *e.g.* natural, perspective, *etc*, of two images. In this task, the prediction function is defined as $r(\mathbf{x}, \mathbf{y})$, which denotes the relative attribute value of $\mathbf{x}$ compared with $\mathbf{y}$. Different from the similarity, this relationship is antisymmetric,

since $r(\mathbf{x}, \mathbf{y}) = -r(\mathbf{y}, \mathbf{x})$. Here we call this relationship $r(\mathbf{x}, \mathbf{y})$ as the *relative order relationship*.

In this work, we extend the deep siamese network from learning the similarity relationship to relative order relationship. As illustrated in Fig. 5.1, our proposed model also use the two sub-networks to extract the deep features of images $\mathbf{x}$ and $\mathbf{y}$. Different from the traditional deep siamese network for similarity learning, the deep features are combined using the relative order prediction function $r(\mathbf{x}, \mathbf{y})$. Because of the marked performance in the bilinear pooling [84] and second order pooling [16], we formulate the proposed model as a bilinear model, in which we design the second-order representation of images and suggest the relative order prediction function. The loss function of our extended deep siamese network is composed of three terms, *i.e.* relative order loss, mean square error (MSE) loss and softmax loss. The relative order loss makes the predicted relative order to be consistent with the ground-truth. The MSE loss minimizes the error between the predicted value and the ground-truth label. To make the feature more discriminative and stable, we formulate the image feature as the normalized probabilities of each prediction value, which can be learned by the softmax loss.

Furthermore, we demonstrate that our proposed model can also be applied into the regression task, *e.g.* age estimation, in which the pairwise relationship is defined as the relative age order of $\mathbf{x}$ and $\mathbf{y}$. This relationship is also the relative order relationship, and its prediction function $r(\mathbf{x}, \mathbf{y})$ is defined as the age difference between $\mathbf{x}$ and $\mathbf{y}$. We find that although this task is not aimed at predicting the pairwise relationship, the relative order information also benefits in model training. We also show that the relative order loss is complementary to the MSE loss and softmax

loss, and it can help to improve the prediction accuracy of the age estimation task.

To sum up, the main contribution of this work is three-fold. First, we propose the second-order representation based relative order prediction function, which can achieve better performance than the traditional linear prediction function. Second, we propose a model by extending the deep siamese network to learn the relative order of images. Third, we applied our proposed model to relative attribute ranking and age estimation, and show the effectiveness of our proposed model in terms of the prediction accuracy.

The rest of this chapter is organized as follows. Section 5.2 demonstrates the proposed model. Section 5.3 describes the network architecture and training approach. Section 5.4 reports the experimental results, and Section 5.5 concludes this chapter.

5.2 Learning the Relative Order Relationship

In this section, we first introduce the relative order prediction function, then propose the model for relative order relationship learning. The model is composed of three loss terms, *i.e.* relative order loss term, MSE loss term and softmax loss term.

5.2.1 Relative Order Prediction Function

Motivated by the good performance of bilinear pooling [84] and second-order pooling [16], we formulate the proposed model as a bilinear model $\mathcal{B} = \left(f, \widetilde{f}, h\right)$, where $f(\mathbf{x})$ is the learned feature of image $\mathbf{x}$ by CNN and $\widetilde{f}(\mathbf{x}) = \left(f^T(\mathbf{x}), 1\right)^T$. h is the prediction function. The bilinear feature combination of f and $\widetilde{f}$ is their outer product

which is formulated as

$$\begin{aligned} g(\mathbf{x}) &= \left(f(\mathbf{x})\widetilde{f}^T(\mathbf{x})\right)_{\text{vec}} \\ &= \left(\left(f(\mathbf{x})f^T(\mathbf{x})\right)_{\text{vec}}^T \quad f^T(\mathbf{x})\right)^T \end{aligned} \tag{5.1}$$

where $(\cdot)_{\text{vec}}$ denotes the vector form of a matrix. Based on Eq. (5.1), we formulate the prediction value of $\mathbf{x}$ as

$$h(\mathbf{x}) = \mathbf{u}^T g(\mathbf{x}) = f^T(\mathbf{x})\mathbf{M}f(\mathbf{x}) + \mathbf{w}^T f(\mathbf{x}) \tag{5.2}$$

where $\mathbf{u} = \left((\mathbf{M})_{\text{vec}}^T, \mathbf{w}^T\right)^T$ is the weight.

To predict the relative order of two samples, we propose a prediction function as the indicator of relative order. Given two images $\mathbf{x}_i$ and $\mathbf{x}_j$, their relative order prediction function is formulated as

$$r(\mathbf{x}_i, \mathbf{x}_j) = h(\mathbf{x}_i) - h(\mathbf{x}_j) \tag{5.3}$$

Compared with the bilinear model in [84], our proposed model has the following differences. First, our proposed model does not need the pooling function, since the model is based on the global representation. Second, f and $\widetilde{f}$ are from different CNNs in [84] in most cases, while in our model, f and $\widetilde{f}$ are from the same CNN. Third, the model in [84] uses a classification function C for the image classification task, while our proposed model replaces it with the prediction function h.

5.2.2 Relative Order Loss Term

Denote by $\{(\mathbf{x}_i, y_i)|i = 1, 2, \cdots, N\}$ the training set, where $\mathbf{x}_i$ is the ith sample, y_i is the ground-truth label of $\mathbf{x}_i$, and N is the number of training samples. Based on the training set, the ordered pair set $\mathcal{P} = \left\{(\mathbf{x}_i, \mathbf{x}_j)|y_i < y_j\right\}$ and unordered pair

set $\mathcal{Q} = \left\{(\mathbf{x}_i, \mathbf{x}_j)|y_i = y_j\right\}$ can be constructed. In the relative attribute ranking and age estimation tasks, the labels y_i are integers. So each image pair $(\mathbf{x}_i, \mathbf{x}_j)$ in set $\mathcal{P}$ satisfies $y_i \leq y_j - 1$. For the image pairs in $\mathcal{P}$, we hope that the predicted order of $(\mathbf{x}_i, \mathbf{x}_j)$ satisfies $h(\mathbf{x}_i) \leq h(\mathbf{x}_j) - 1$. For the image pairs in $\mathcal{Q}$, we hope that the prediction values of $\mathbf{x}_i$ and $\mathbf{x}_j$ are as close as possible. Motivated by [101, 146], we build the following constraints

$$\begin{aligned} &r(\mathbf{x}_i, \mathbf{x}_j) \leq -1 + \xi_{ij}, \xi_{ij} \geq 0 \quad \forall(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{P} \\ &r^2(\mathbf{x}_i, \mathbf{x}_j) \leq \zeta_{ij}, \zeta_{ij} \geq 0 \quad \forall(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{Q} \end{aligned} \tag{5.4}$$

where ξ_{ij} and ζ_{ij} are slack variables. We rewrite the constraints into the following loss term

$$L_1 = \frac{1}{|\mathcal{P}|} \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{P}} \Big[r(\mathbf{x}_i, \mathbf{x}_j) + 1\Big]_+ + \frac{1}{|\mathcal{Q}|} \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{Q}} r^2(\mathbf{x}_i, \mathbf{x}_j) \tag{5.5}$$

where $[a]_+ = \max(a, 0)$.

5.2.3 Mean Square Error Loss Term

To make the predicted value $h(\mathbf{x}_i)$ close to the ground-truth y_i, we introduce the MSE loss term to our model, which is formulated as

$$L_2 = \frac{1}{N} \sum_i (h(\mathbf{x}_i) - y_i)^2 \tag{5.6}$$

5.2.4 Softmax Loss Term

As the MSE loss term may be unstable to the outliers [107], we introduce the softmax loss to learn the feature. We partition the possible prediction values into T non-overlapping ranges, and denote $p_k(\mathbf{x})$ as the normalized probability of the predicted

value in the kth range. We formulate the feature as $f(\mathbf{x}) = (p_0(\mathbf{x}), \cdots, p_T(\mathbf{x}))^T$, and it can be learned using the softmax loss as follows,

$$L_3 = -\sum_{i,k} s_{ik} \log\left(p_k(\mathbf{x}_i)\right) \tag{5.7}$$

where $s_{ik} = 1$ if y_i belongs to the kth range, otherwise $s_{ik} = 0$.

Overall, the loss function of our proposed model is the integration of (5.5), (5.6) and (5.7), which is formulated as

$$L = L_1 + \lambda L_2 + \beta L_3 \tag{5.8}$$

where λ and β are coefficients. In the experiments, we set $\lambda = 0.1$ and $\beta = 1$.

5.2.5 Discussion

The proposed model can be applied into the ranking and regression tasks. In our proposed model, relative order loss term, MSE loss term and softmax loss term are complementary to each other. The relative order loss term is supervised by the relative order information of image pairs. The MSE and softmax loss terms use the ground-truth label of target value as the supervisory signal. The relative order of image pairs is a weakly-supervised signal, while the ground-truth target value is the strong supervision. We show that in the ranking and regression tasks, these two signals are complementary to each other, and both of them have the positive effect on the performances of these tasks.

For the ranking task, the traditional methods mainly use the relative order of image pairs as the supervision [101, 121, 146], and their loss functions are similar to the relative order loss term in our proposed method. In some applications, *e.g.*

relative attribute ranking, the attribute value labels of training images are available in some datasets [70, 72, 99]. So our proposed method can make the relative attribute ranking model to be stronger supervised by introducing the MSE loss and softmax loss.

For the regression task, most of the existing methods are supervised by the ground-truth label of each images by adopting the MSE loss or softmax loss [98, 107]. However, this kind of supervision is only labeled for each single image, and the relationship of different images are not considered. By introducing the relative order loss term, our proposed method can not only learn from the ground-truth label of training images, but also learn from the relative order of training image pairs.

Weakly-supervised learning: In the relative attribute ranking task, sometimes the attribute labels of each training images are not available, and we can only get the relative order annotations of training image pairs. In this case, we can remove the MSE loss and softmax loss from our proposed model by setting the parameters $\lambda = 0$ and $\beta = 0$. Thus this model becomes a weakly-supervised model which can be applied to the weakly annotated data.

5.3 Deep Convolutional Neural Network

In this section, we describe the deep CNN framework of our proposed model. First, we introduce the network architecture of the model. Then we give the approach to train this network.

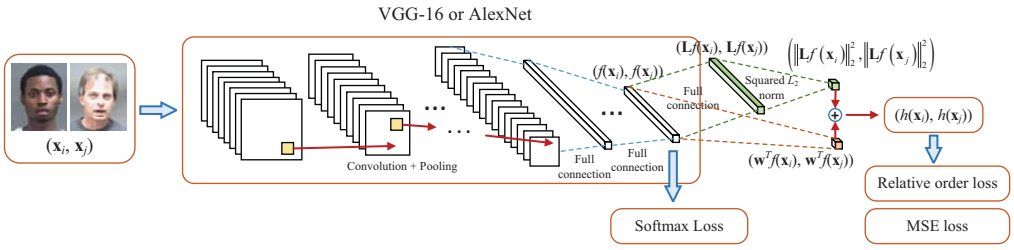


Figure 5.2 The proposed relative order relationship learning framework. It takes the image pair $(\mathbf{x}_i, \mathbf{x}_j)$ as input. The VGG-16 or AlexNet is used to extract their deep features as $\left(f(\mathbf{x}_i), f(\mathbf{x}_j)\right)$, which are feeded into the softmax loss layer. We use an extra fully-connected layer with $\left(f(\mathbf{x}_i), f(\mathbf{x}_j)\right)$ as its input to generate $\left(\mathbf{L}f(\mathbf{x}_i), \mathbf{L}f(\mathbf{x}_j)\right)$ and $\left(\mathbf{w}^T f(\mathbf{x}_i), \mathbf{w}^T f(\mathbf{x}_j)\right)$. Afterwards, we can use Eq. (5.10) to obtain $h(\mathbf{x}_i)$ and $h(\mathbf{x}_j)$, which are input into the relative order loss and MSE loss layers in training.

5.3.1 Network Architecture

Similar to the traditional deep siamese model, our extended deep siamese model also has two sub-networks which share the same architecture and the same parameters. For each of the two sub-networks, we use the AlexNet [71] and VGG-16 [120] as the network architecture in relative attribute ranking and age estimation, respectively. The framework of our proposed model is illustrated in Fig. 5.2. As the architectures and parameters of these two sub-networks are the same, we only illustrate one sub-network in Fig. 5.2 for simplicity.

To learn the probabilities of each ranges of the predicted value, we modify the neuron number of softmax loss layer to be T. We have defined the prediction function in Eq. (5.2). To make it straightforward to be implemented, we assume that $\mathbf{M}$ is a positive semidefinite (PSD) matrix. So we can write $\mathbf{M}$ as $\mathbf{M} = \mathbf{L}^T\mathbf{L}$, where $\mathbf{L}$

is a matrix with the same size of $\mathbf{M}$. In this case, Eq. (5.2) can be rewritten as

$$h(\mathbf{x}) = f^T(\mathbf{x})\mathbf{L}^T\mathbf{L}f(\mathbf{x}) + \mathbf{w}^T f(\mathbf{x}) = ||\mathbf{L}f(\mathbf{x})||_2^2 + \mathbf{w}^T f(\mathbf{x}) \tag{5.9}$$

In Fig. 5.2, we add an additional fully-connected layer with input as $f(\mathbf{x})$. Its output has $T + 1$ dimensions, with the first T dimensions as $\mathbf{L}f(\mathbf{x})$, and the last dimension as $\mathbf{w}^T f(\mathbf{x})$. By this layer Eq. (5.9) can be easily implemented as

$$h(\mathbf{x}) = ||f_{\mathbf{L}}(\mathbf{x})||_2^2 + f_{\mathbf{w}}(\mathbf{x}) \tag{5.10}$$

where $f_{\mathbf{L}}(\mathbf{x}) = \mathbf{L}f(\mathbf{x})$ and $f_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^T f(\mathbf{x})$.

5.3.2 Network Training

In this subsection, we give the training approach of the proposed model. To use back propagation (BP) to train this network, we compute the gradient of the loss functions (5.5), (5.6) and (5.7) to each network parameters.

Prediction Function: As the prediction function can be realized by Eq. (5.10), the gradients of $h(\mathbf{x})$ to $f_{\mathbf{L}}(\mathbf{x})$ and $f_{\mathbf{w}}(\mathbf{x})$ are straightforward, which are given by

$$\frac{\partial h(\mathbf{x})}{\partial f_{\mathbf{L}}(\mathbf{x})} = 2f_{\mathbf{L}}(\mathbf{x}), \frac{\partial h(\mathbf{x})}{\partial f_{\mathbf{w}}(\mathbf{x})} = 1 \tag{5.11}$$

Relative Order Loss: As the gradient of $h(\mathbf{x})$ can be computed by Eq. (5.11), we only need to compute the gradient of L_1 to $h(\mathbf{x}_i)$. From Eq. (5.5), its sample-based gradient is computed as follows

$$\begin{aligned}\frac{\partial L_1}{\partial h(\mathbf{x}_i)} = &\frac{1}{|\mathcal{P}|}\sum_{\{\mathbf{x}_j|(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{P}\}} 1\left\{r(\mathbf{x}_i,\mathbf{x}_j)+1>0\right\} \\ &-\frac{1}{|\mathcal{P}|}\sum_{\{\mathbf{x}_j|(\mathbf{x}_j,\mathbf{x}_i)\in\mathcal{P}\}} 1\left\{r(\mathbf{x}_j,\mathbf{x}_i)+1>0\right\} \\ &+\frac{4}{|\mathcal{Q}|}\sum_{\{\mathbf{x}_j|(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{Q}\}} r(\mathbf{x}_i,\mathbf{x}_j)\end{aligned} \tag{5.12}$$

where $1\{\cdot\}$ is 1 if the inner condition is true, and 0 otherwise. From the sample-based gradient in Eq. (5.12), we can see that when the batch size is fixed, the number of image pairs have little effect on the computational cost. Therefore, given a mini-batch of n images, we can construct all the possible $n(n-1)/2$ image pairs $(\mathbf{x}_i, \mathbf{x}_j)$ which satisfies $y_i \leq y_j$ in each iteration. Then these pairs are put into $\mathcal{P}$ or $\mathcal{Q}$ depending on whether $y_i < y_j$ or $y_i = y_j$.

MSE Loss and Softmax Loss: From (5.6) and (5.7), the gradients of L_2 and L_3 are

$$\begin{aligned} \frac{\partial L_2}{\partial h(\mathbf{x}_i)} &= 2\left(h(\mathbf{x}_i) - y_i\right) \\ \frac{\partial L_3}{\partial p_k(\mathbf{x}_i)} &= -\frac{s_{ik}}{p_k(\mathbf{x}_i)} \end{aligned} \tag{5.13}$$

By (5.11), (5.12) and (5.13), we can realize the end-to-end training of our network, in which the deep feature and relative order prediction function can be jointly learned in the training process.

5.4 Experiments

In this section, we evaluate the proposed method based on the relative attribute ranking and age prediction tasks. We use the Caffe [62] framework to implement the proposed method. The experiments are conducted using an NVIDIA TITAN X GPU.

5.4.1 Relative attribute ranking

For the relative attribute ranking task, we use the Outdoor Scene Recognition (OSR) [99], Public Figure Face (PubFig) [72] and Shoes [70] datasets for evaluation.

In the OSR dataset, there are 2,688 images from 8 categories. Each of them is related to 6 attributes, *i.e.* natural, open, perspective, size-large, diagonal-plane and depth-close. Following the protocol in [146], we use 240 images for training, and 2,448 images for testing. The PubFig dataset consists of 772 face images from 8 persons. They are assigned with 11 attributes, *i.e.* male, white, young, smiling, chubby, visible-forehead, bushy-eyebrows, narrow-eyes, pointy-nose, big-lips and round-face. We use 241 images as the training set, and use 531 images as the test set. The Shoes dataset includes 14,658 shoe images from 10 categories, which is related to 10 attributes, *i.e.* pointy-at-the-front, open, bright-in-color, covered-with-ornaments, shiny, high-at-the-heel, long-on-the-leg, formal, sporty, and feminine. For the Shoes dataset, we use 240 images for training, and use the remaining images for testing.

As the training sets of OSR, PubFig and Shoes datasets are relatively small, we utilize the AlexNet [71] network architecture in our model. The images are resized to size 227×227 before input into the network. The network is pre-trained on ImageNet [71], then fine-tuned on the evaluation dataset. The momentum is set as 0.9. The weight decay is set as 0.00005. The network is trained for 4,000 iterations. The number of predicted value ranges T is set to be 6, 8 and 10 in the OSR, PubFig and Shoes datasets, respectively. We use the mini-batch strategy in training, and set the batch size as 64.

The ranking accuracies of OSR, PubFig and Shoes datasets on each attributes are reported in Table 5.1, Table 5.2 and Table 5.3, respectively. From the results, we can see that our proposed method can achieve higher ranking accuracies than the other state-of-the-art relative attribute ranking methods in OSR, PubFig and Shoes

datasets.

Table 5.1 Ranking accuracies (%) of different methods on the OSR dataset. The highest and second highest accuracies are highlighted in red and blue colors, respectively.

Attributes	RA [101]	MTL [27]	DRA [146]	Singh and Lee [121]	Ours	Ours (Weakly-supervised)
Natural	94.82	96.47	99.47	98.89	99.70	99.57
Open	91.01	92.88	97.81	97.20	98.26	97.90
Perspective	86.56	88.39	97.19	96.31	97.49	97.17
Size-large	86.37	88.50	96.88	95.98	96.87	97.04
Diagonal-plane	88.00	90.87	98.46	97.64	98.24	98.47
Depth-close	88.35	89.05	97.24	96.10	97.85	97.23
Average	89.19	91.03	97.84	97.02	98.07	97.90

We also evaluate our proposed model with only the relative order annotations of image pairs. We train our proposed model without the MSE loss and softmax loss. The ranking accuracies are reported as "*Ours (Weakly-supervised)*" in Table 5.1, Table 5.2 and Table 5.3. We can see that the accuracies of our proposed method with only the relative order annotations is slightly lower than the model with MSE loss and softmax loss. But its accuracies are still higher than the other competing methods. This result shows that the MSE loss and softmax loss make the model able to learn from the labeled attributes of each image. It also suggests that the second-order image representation of our proposed model is superior than the traditional first-order representation in relative attribute ranking.

Table 5.2 Ranking accuracies (%) of different methods on the PubFig dataset. The highest and second highest accuracies are highlighted in red and blue colors, respectively.

Attributes	RA [101]	MTL [27]	DRA [146]	Ours	Ours (Weakly-supervised)
Male	82.57	84.52	90.82	91.84	91.44
White	79.14	80.11	87.12	87.98	88.53
Young	82.52	83.91	91.49	92.80	92.00
Smiling	81.37	82.19	92.68	93.68	93.12
Chubby	77.80	79.16	89.30	90.14	89.72
Visible-forehead	88.75	89.86	94.39	94.56	95.21
Bushy-eyebrows	80.63	82.06	90.19	91.21	91.72
Narrow-eyes	81.68	81.48	90.60	91.67	91.46
Pointy-nose	79.01	79.86	91.03	91.50	89.83
Big-lips	80.38	81.20	90.35	91.05	91.20
Round-face	82.37	83.43	91.99	93.10	92.14
Average	81.47	82.52	90.91	91.78	91.49

Table 5.3 Ranking accuracies (%) of different methods on the Shoes dataset. The highest and second highest accuracies are highlighted in red and blue colors, respectively.

Attributes	RA [101]	MTL [27]	DRA [146]	Ours	Ours (Weakly-supervised)
Pointy-at-the-front	79.32	84.66	88.34	89.12	88.79
Open	76.41	77.37	87.02	87.75	86.48
Bright-in-color	53.09	64.06	74.97	74.95	74.77
Covered-with-ornaments	57.96	71.20	79.86	80.81	79.97
Shiny	66.61	80.53	86.92	87.36	87.05
High-at-the-heel	78.38	80.92	87.50	88.17	87.64
Long-on-the-leg	68.35	73.61	84.30	84.82	85.00
Formal	73.93	74.16	81.76	82.20	81.81
Sporty	69.84	80.46	87.72	88.22	87.67
Feminine	77.84	84.06	87.98	88.66	88.05
Average	70.17	77.10	84.64	85.21	84.72

5.4.2 Age estimation

We use the Craniofacial Longitudinal Morphological Face Dataset (MORPH) [106] and Cross-Age Celebrity Dataset (CACD) [22] to evaluate the proposed model. We use the mean absolute error (MAE) as the performance measure, which is defined as the average of the absolute error between the predicted age and the ground-truth age.

As the evaluation datasets are relatively large, we adopt the VGG-16 [120] network architecture in our model because of its excellent performance in ImageNet challenge. Before the training process, the input images are resized to size 256×256, and then randomly cropped to size 224×224. The momentum is set as 0.9, and the weight decay is set as 0.0005. We assume that the possible age ranges from 0 to 100 years, so we set the number of age ranges as $T = 101$, with each range covers one year. The network is trained for 100,000 iterations. The batch size is set to be 20. The network parameters are initialized with the model pre-trained on ImageNet [120]. Then this network with only the softmax loss is further trained on the IMDB-WIKI dataset [107], which is a large-scale face dataset with age annotation. Finally we fine-tune the whole network on the dataset which we evaluate.

MORPH Dataset

The MORPH dataset [106] consists of 55,134 images from 13,618 individuals. The ages of these images range from 16 to 77. In our experiments, we use 80% of these images for training, and use the remaining images for testing.

First, we compare the second-order representation with the traditional first-order representation, in which the prediction function is formulated as $h'(\mathbf{x}) = \mathbf{v}^T f(\mathbf{x})$,

where $\mathbf{v}$ is the coefficient vector. We train two models based on the second-order and first-order representations, respectively. Their MAEs are reported in Table 5.4. The MAEs under different iteration numbers are shown in Fig. 5.3(a). From the results, we can see that the model based on second-order representation can achieve better performance than the model based on first-order representation.

Table 5.4 Performance comparison between the models based on the second-order and first-order representations on the MORPH dataset

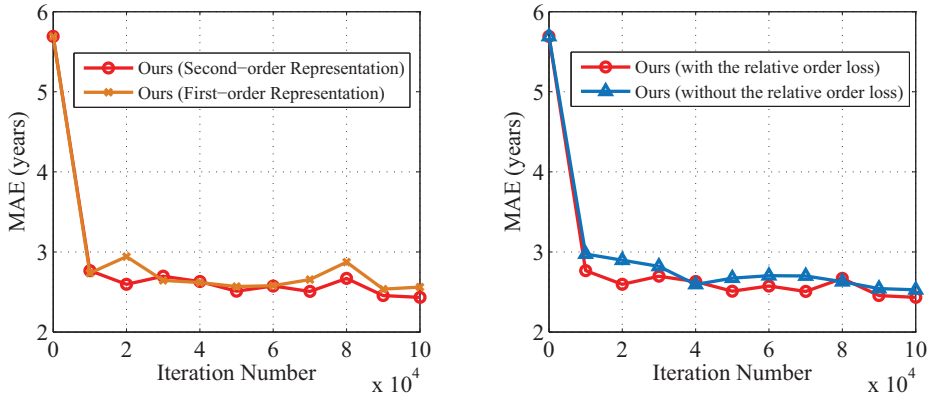
Representation type	MAE (years)
First-order representation	2.56
Second-order representation	2.43

Second, we evaluate the effect of the relative order loss term. We compare the models with and without the relative loss term, respectively. The MAEs of these two models are reported in Table 5.5. The MAEs trained with different iteration numbers are shown in Fig. 5.3(b). We can see that the model can achieve better performance with the relative order loss term. This result demonstrates the effectiveness of the relative order loss in age estimation.

Table 5.5 Performance comparison between the models with and without the relative order loss term on the MORPH dataset

Model type	MAE (years)
Our model (without the relative order loss term)	2.53
Our model (with the relative order loss term)	2.43

We also compare the MAEs of our proposed method with the other state-of-the-art age estimation methods, *i.e.* BIFs+LSVR [52], BIFs+CCA [53], BIFs+OR-



(a) (b)

Figure 5.3 The MAEs versus iteration number on the MORPH dataset

SVM [18], BIFs+OHRank [19], MR-CNN [98] and OR-CNN [98]. The results are reported in Table 5.6. We can see the MAE of our proposed method is significantly lower than the other methods. To make fair comparison with the other methods, we also report the performance with the same evaluation protocol as [19, 26, 50, 107, 108, 137], which use a subset of the MORPH dataset with 5,475 images. In this subset, 4,380 images are used for training and 1,095 images are used for testing. As this subset is relatively small, we train the network for 5,000 iterations. In this setting, we compare our proposed method with SVR [50], CA-SVR [26], OHRank [19], DLA [137], the work by Rothe *et al.* [108] and DEX [107] in Table 5.6. It can be seen that our proposed method also outperforms the other comparison methods in this evaluation protocol.

Table 5.6 Performance comparison of different models on the MORPH dataset (* means that this method uses the MORPH subset of 5,475 images)

Method	MAE (years)
BIFs+LSVR [52]	4.31 [98]
BIFs+CCA [53]	4.73 [98]
BIFs+OR-SVM [18]	4.21 [98]
BIFs+OHRank [19]	3.82 [98]
MR-CNN [98]	3.42
OR-CNN [98]	3.27
Ours	2.43
SVR* [50]	5.77
CA-SVR* [26]	5.88
OHRank* [19]	5.69
DLA* [137]	4.77
Rothe *et al.* * [108]	3.45
DEX* [107]	2.68
Ours*	2.50

CACD Dataset

The CACD dataset includes 163,446 images from 2000 celebrities. Following the evaluation protocol in [22], we use the images of 1800 persons for training, use the images from 80 images for validation, and use the remaining images of 120 persons for testing. The validation set is manually annotated, while the training set is not. We compare the MAEs of our proposed method and the DEX [107] method. We report the MAEs when training on the training images and validation images in Table 5.7, respectively. We can see that our proposed method achieves better performance than DEX, whether our proposed model is trained on the training set or validation set. We can also see that when training on the validation set, the performance gain of our proposed method is more evident. This suggests that our proposed method can also achieves satisfactory performance when training on a small dataset.

Table 5.7 Comparison of MAEs of DEX [107] and our proposed model on the CACD dataset

Model	Training on the training set	Training on the validation set
DEX [107]	4.785	6.521
Ours	4.767	5.731

5.5 Summary

In this chapter, we proposed a model to extend the deep siamese network to learn the relative order relationship of images. We formulated the relative order prediction

function based on the second-order image representation. To make the predicted relative order to be consistent with the ground-truth, we introduced the relative order loss to the model. We also used the MSE loss to minimize the error between predicted value and the ground-truth label. To make the learned feature discriminative and stable, we introduced the softmax loss to learn the image feature. The experimental results on relative attribute ranking and age estimation tasks demonstrate that our proposed model perform better than the comparison methods.

Chapter 6

Joint Learning of Multiple Relative Order Relationships by Deep Siamese Networks for Camera Pose Estimation

6.1 Introduction

In some computer vision tasks, we need to learn and predict the relative order relationships of multiple components. For example, in the camera pose estimation task, we need to predict the pose $\mathbf{p} = [\mathbf{s}, \mathbf{q}]$ of a camera from an input image, where $\mathbf{s}$ and $\mathbf{q}$ denote the position and orientation of the camera, respectively. Given two cameras $\mathbf{x}$ and $\mathbf{y}$, each component of their relative pose $\mathbf{r}(\mathbf{x}, \mathbf{y})$ satisfies the relative order relationship since $\mathbf{r}(\mathbf{x}, \mathbf{y}) = -\mathbf{r}(\mathbf{x}, \mathbf{y})$.

As the CNN based deep learning approaches have been successfully applied into many computer vision tasks, *e.g.* image classification [71], face verification [112], person re-identification [156], etc, some methods also learn and predict the camera pose [66, 67] based on CNN. In these works, the camera pose estimation is considered as a regression task. They usually formulate a deep CNN with the mean square error loss to make the predicted pose close to the ground-truth pose. However, these approaches regard each training image as an independent sample, and learn from each image separately. They haven't considered the relative order relationships between the camera poses from different images.

In this chapter, we consider the camera pose estimation task as an Multi-Task Learning (MTL) problem, in which the learning of each pose component is regarded as a learning task. We aim to learn these tasks jointly rather than to learn these tasks separately to discover the potential connection of pose components. Therefore, we propose a camera pose estimation method based on deep siamese networks. Given two images $\mathbf{x}$ and $\mathbf{y}$, this network predicts the relative pose $\mathbf{r}(\mathbf{x}, \mathbf{y})$ of the image pair $(\mathbf{x}, \mathbf{y})$. Similar to Chapter 5, we also use the second-order representation of images to learn the relative order relationship, and adopt the relative order loss and mean square error (MSE) loss to make the predicted poses and their relative order to be consistent with the ground-truth. Different from Chapter 5, we learn multiple relative order relationships of camera pose jointly using the deep siamese network. Our deep siamese network architecture is illustrated in Fig. 6.1. It consists of two branches which share the same parameters. Each branch consists of two kinds of sub-networks. One of them is mainly composed of the convolutional and pooling layers, and it aims to learn the spatial feature of the input image. So we call it as

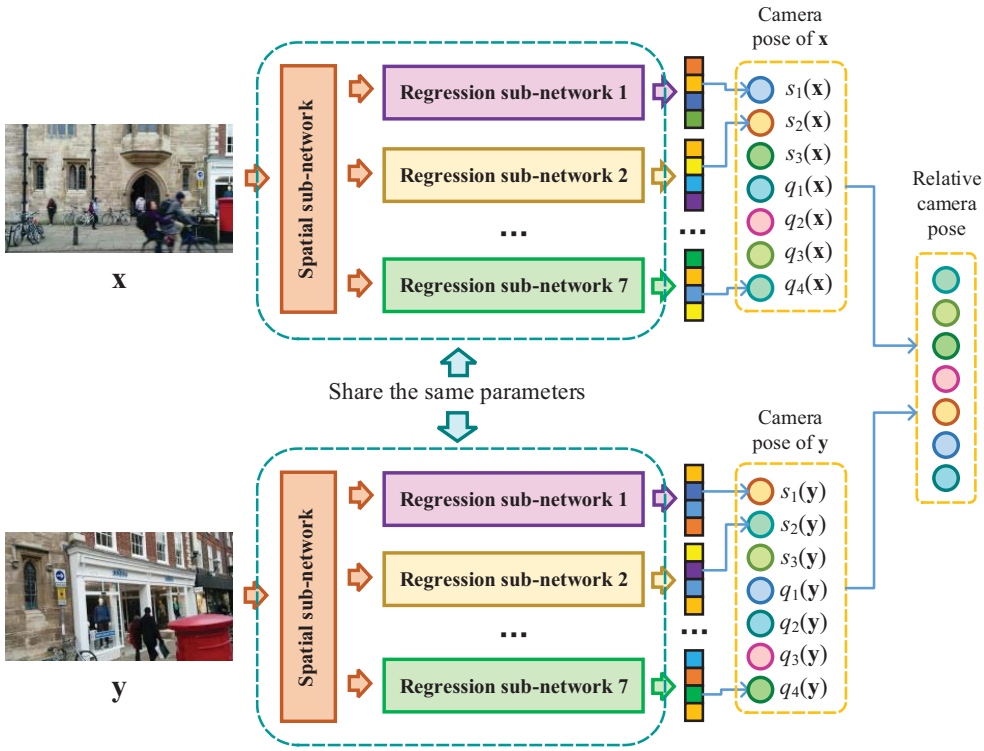


Figure 6.1 The proposed framework for relative camera pose estimation.

spatial sub-network. As the spatial feature is related to all pose components, the spatial sub-network parameters are shared across all the learning tasks, and it can capture the generality between different pose components. Another sub-network mainly consists of the fully connected layers, and it learns the regressors to regress the pose from the image spatial feature. Thus it is called as *regression sub-network*. As the regressors of the pose components are different, the regression sub-network of different pose components are separated. So it can capture the specificity of each pose component.

Our proposed model is different from the single relative order relationship learn-

ing model in Chapter 5. First, our proposed model can learn the relative camera pose, which belongs to the multiple relative order relationship. Second, we design the proposed network as the combination of spatial sub-network and regression sub-networks to reach the trade-off between the generality and specificity of each relative order relationship.

To sum up, the contribution of this work is three fold. First, it applies the second-order representation for camera pose estimation, and we demonstrated that the second-order representation is not only benefit to single relative order relationship learning, but also can be successfully applied into the multiple relative order relationship learning tasks such as camera pose estimation. Second, we propose the relative order loss term to extend the deep siamese network to multiple relative order relationship learning. Third, we design a novel network architecture with the spatial sub-network and regression sub-network to learn multiple camera pose components.

The rest of this chapter is organized as follows. Section 6.2 formulates the proposed model. Section 6.3 describes the network architecture of our proposed model. Section 6.4 reports the experimental results. Section 6.5 summarizes this chapter.

6.2 Learning Multiple Relative Order Relationship

In this section, we first propose the multiple relative order prediction functions. Then we formulate the model for multiple relative order relationships learning based on the prediction functions. Similar to the single relative order relation-

ship learning model in Chapter 5, the proposed multiple relative order relationships learning model is also composed of the relative order loss term and MSE loss term.

6.2.1 Multiple Relative Order Prediction Function

As there are multiple components which belongs to the relative order relationship, we propose the relative order prediction function for each component. Analogous to the single relative order relationship learning model, we construct the bilinear model for each component. Denote by $\mathcal{B}_c = \left(f_c, \widetilde{f_c}, h_c\right)$ the bilinear model of the cth component, where $\widetilde{f_c}$ is the learned feature of the image $\mathbf{x}$ by deep CNN, and $\widetilde{f_c}(\mathbf{x}) = \left(f_c^T(\mathbf{x}), 1\right)^T$. h_c is the prediction function of the cth component. Thus the bilinear feature combination of f_c and $\widetilde{f_c}$ is $g_c(\mathbf{x}) = \left(\left(f_c(\mathbf{x}) f_c^T(\mathbf{x})\right)_{\text{vec}}^T \quad f_c^T(\mathbf{x})\right)^T$. So we formulate the prediction values of $\mathbf{x}$ as $\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), ..., h_N(\mathbf{x})]$, where N is the number of components. For the cth component, its prediction value is

$$h_c(\mathbf{x}) = \mathbf{u}_c^T g_c(\mathbf{x}) = f_c^T(\mathbf{x}) \mathbf{M}_c f_c(\mathbf{x}) + \mathbf{w}_c^T f_c(\mathbf{x}) \tag{6.1}$$

where $[\cdot]_{\text{vec}}$ is the vector form of a matrix, and $\mathbf{u}_c = \left((\mathbf{M}_c)_{\text{vec}}^T, \mathbf{w}_c^T\right)^T$ is the weight vector.

Based on the prediction function, the multiple relative order prediction function of images $\mathbf{x}_i$ and $\mathbf{x}_j$ is formulated as

$$\mathbf{r}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{h}(\mathbf{x}_i) - \mathbf{h}(\mathbf{x}_j) \tag{6.2}$$

6.2.2 Multiple Relative Order Loss Term

Denote by $\{(\mathbf{x}_i, \mathbf{y}_i) | i = 1, 2, \cdots, N\}$ the training set, where $\mathbf{x}_i$ is the ith training image and $\mathbf{y}_i$ is the image label. Different from the single relative order relationship

learning problem, here $\mathbf{y}_i$ consists of multiple image labels. Denote by $y_i[c]$ the label of the cth component of $\mathbf{x}_i$. We construct the ordered image pair as $\mathcal{P}_c = \left\{(\mathbf{x}_i, \mathbf{x}_j)|y_i[c] < y_j[c]\right\}$, and the unordered image pair as $\mathcal{Q}_c = \left\{(\mathbf{x}_i, \mathbf{x}_j)|y_i[c] = y_j[c]\right\}$. Following [101, 146], the following constraints are built based on the sets of image pairs.

$$\begin{aligned} r_c(\mathbf{x}_i, \mathbf{x}_j) \leq \xi_{ij}^c, \xi_{ij}^c \geq 0 \quad \forall(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{P}_c \\ r_c^2(\mathbf{x}_i, \mathbf{x}_j) \leq \zeta_{ij}^c, \zeta_{ij}^c \geq 0 \quad \forall(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{Q}_c \end{aligned} \tag{6.3}$$

where ξ_{ij}^c and ζ_{ij}^c are slack variables of image pair $(\mathbf{x}_i, \mathbf{x}_j)$, component c. We reformulate the constraints as the following loss function

$$L_1 = \sum_c \frac{1}{|\mathcal{P}_c|} \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{P}_c} \left[r_c(\mathbf{x}_i, \mathbf{x}_j)\right]_+ + \sum_c \frac{1}{|\mathcal{Q}_c|} \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{Q}_c} r_c^2(\mathbf{x}_i, \mathbf{x}_j) \tag{6.4}$$

6.2.3 MSE Loss Term

The MSE loss term makes the predicted pose $\mathbf{p}(\mathbf{x}_i) = [\mathbf{s}(\mathbf{x}_i), \mathbf{q}(\mathbf{x}_i)]$ close to the ground-truth $\mathbf{p}_{gt}(\mathbf{x}_i) = \left[\mathbf{s}_{gt}(\mathbf{x}_i), \mathbf{q}_{gt}(\mathbf{x}_i)\right]$. As we use the quaternion to represent the camera orientation, it should be normalized to unit length. Following [66, 67], we formulate the MSE loss term as follows,

$$L_2 = \frac{1}{N} \sum_i \left\|\mathbf{s}(\mathbf{x}_i) - \mathbf{s}_{gt}(\mathbf{x}_i)\right\|_2^2 + \beta \left\| \frac{\mathbf{q}(\mathbf{x}_i)}{\|\mathbf{q}(\mathbf{x}_i)\|_2} - \mathbf{q}_{gt}(\mathbf{x}_i) \right\|_2^2 \tag{6.5}$$

where β is the trade-off parameter.

6.3 Deep Convolutional Neural Network

6.3.1 Network Architecture

As different prediction components have the potential connections and variances, we hope the deep CNN can learn both of them. One possible solution is to train a single relative order relationship model for each component. However, this approach may lost the potential connections of different components. If we use an united deep network to learn the relative order relationships, it would be insufficient to model the variance of different components. Therefore, to find the trade-off between the potential connections and variances of different prediction components is crucial for the network architecture.

We design the network architecture of our proposed model based on the GoogLeNet [123]. As we describes in Section 6.1, our proposed network consists of the spatial sub-network and the regression sub-networks. In our proposed network architecture, the spatial sub-network is mainly composed of the inception, convolution, pooling, and local response normalization layers, which learns the spatial feature of the input image. As the spatial sub-network is shared across different outputs, it is able to model the potential connections of different prediction components. The regression sub-network consists of a fully connected layer and a second-order representation module to regress the pose component of the input image. As the regression sub-networks are not shared, it can learn the variances of different pose components. We incorporate the regression sub-networks in three regression branches of GoogLeNet [123] with the feature maps from different layers as inputs. Analogous to the GoogLeNet for classification [123], two of these three branches are

used for propagating the gradient into the lower layers to deal with the vanishing gradient problem. In the testing stage, we discard the two branches in the lower layers, and only use the output of the last branch as the prediction result. Our network architecture is illustrated in Fig. 6.2.

In our proposed network architecture, we use the second-order representation module to compute the predicted pose component based on the second-order representation by Eq. (6.1). Similar to the single relative order relationship learning method in Chapter 5, we assume that $\mathbf{M}_c$ is a positive semidefinite (PSD) matrix, and rewrite it as $\mathbf{M}_c = \mathbf{L}_c^T \mathbf{L}_c$. Thus, we rewrite the prediction function of Eq. (6.1) as follows

$$\begin{aligned} h_c(\mathbf{x}) &= f_c^T(\mathbf{x}) \mathbf{L}_c^T \mathbf{L}_c f_c(\mathbf{x}) + \mathbf{w}_c^T f_c(\mathbf{x}) \\ &= \|\mathbf{L}_c f_c(\mathbf{x})\|_2^2 + \mathbf{w}_c^T f_c(\mathbf{x}) \\ &= \|f_{\mathbf{L}_c}(\mathbf{x})\|_2^2 + f_{\mathbf{w}_c}(\mathbf{x}) \end{aligned} \tag{6.6}$$

where $f_{\mathbf{L}_c}(\mathbf{x}) = \mathbf{L}_c f_c(\mathbf{x})$ and $f_{\mathbf{w}_c}(\mathbf{x}) = \mathbf{w}_c^T f_c(\mathbf{x})$. In the second-order representation module shown in Fig. 6.3, we use two fully connected layers to compute $f_{\mathbf{L}_c}(\mathbf{x})$ and $f_{\mathbf{w}_c}(\mathbf{x})$, respectively. Thus the prediction value in Eq. (6.6) can be implemented.

6.4 Experiments

In this section, we evaluate the proposed method based on the camera pose estimation task. We use the Cambridge Landmarks [67] with 5 datasets in the experiments. The basic information of these datasets are presented in Table 6.1 [67]. The proposed method is implemented using the Caffe [62] framework. We use the PoseNet [67] model trained by these datasets as the pre-trained model, and fine-tune them

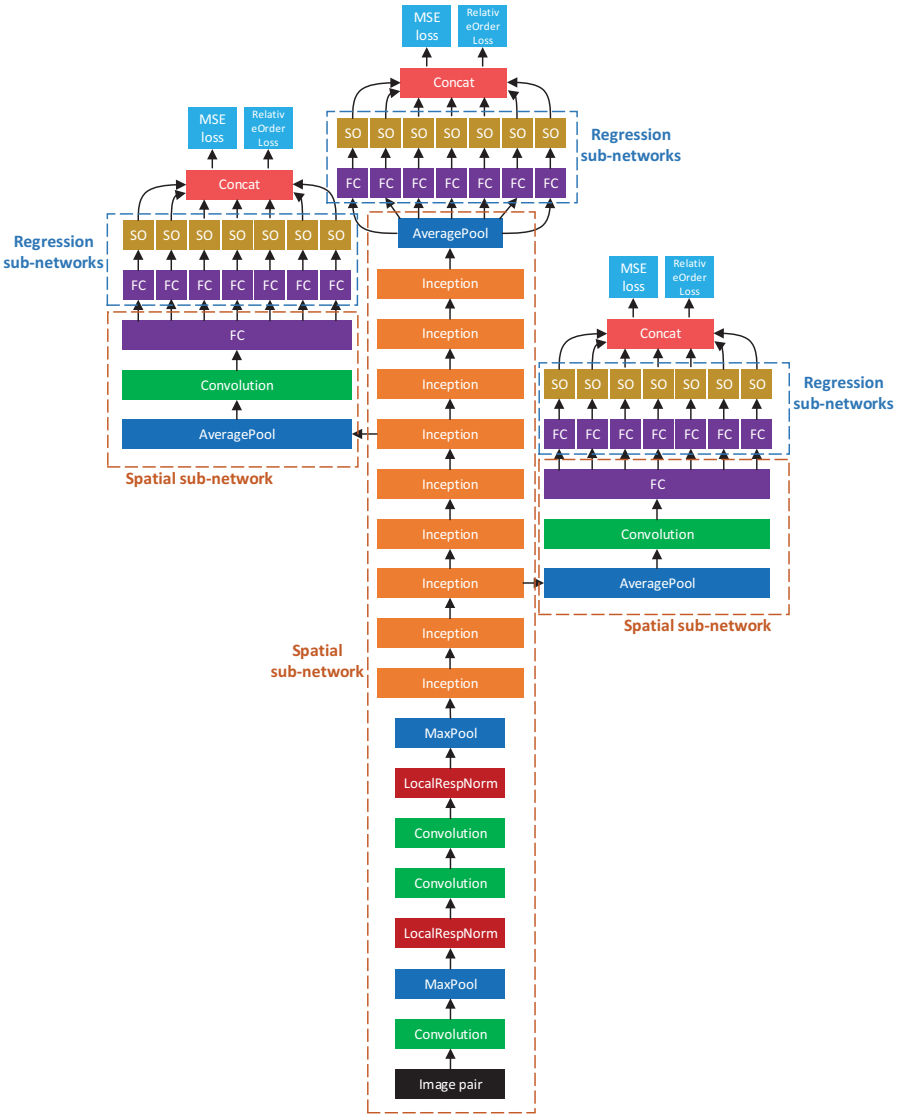


Figure 6.2 The proposed multiple relative order relationship learning framework (FC: Fully connected layer; SO: Second-order representation module).

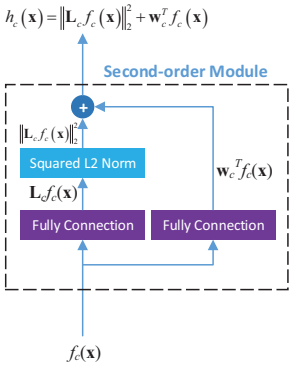


Figure 6.3 The structure of the second-order representation module

Table 6.1 The basic information of the datasets used in the experiments.

Dataset	Training Frames	Testing Frames	Spatial Extent
King's College	1220	343	$140m \times 40m$
Old Hospital	895	182	$50m \times 40m$
Shop Facade	231	103	$35m \times 25m$
St Mary's Church	1487	530	$80m \times 60m$

using our proposed method for 1600 iterations. The learning rate is set to be 10^{-5}, and the weight decay is set as 0.5. The experiments are conducted using an NVIDIA TITAN X GPU.

6.4.1 Comparison of Alternative Network Architectures

First, we compare the performances of our proposed network with and without the relative order loss term in Table 6.2. We can see that when we introduce the relative order loss term in our proposed network, its performances are better in most of the datasets. Therefore, the relative order loss term is effective in camera pose estimation. It demonstrates that the multiple relative order relationship learning is of benefit to camera pose estimation.

Table 6.2 The median prediction errors of our proposed network with and without the relative order loss term

Dataset	Ours (With the relative order loss term)	Ours (Without the relative order loss term)
King's College	**1.86m**, **2.65°**	1.88m, 2.67°
St Mary's Church	2.31m, **4.15°**	**2.30m**, 4.20°
Old Hospital	**2.33m**, **2.69°**	2.34m, 2.71°
Shop Facade	**1.52m**, **4.02°**	1.65m, 4.07°

Second, we evaluate the effect of the second-order representation in our proposed model. We compare the performances of our proposed model with the second-order and the traditional first-order representations, respectively. Table 6.3 reports the median prediction errors of our proposed network with different types of representations. We can see that our proposed network with the second-order repre-

sentation performs better than that with the first-order representation in 3 out of 4 datasets. Therefore, the second-order representation is also beneficial to the multiple relative order relationship learning task.

Table 6.3 The median prediction errors of our proposed network with the second-order and the first-order representations

Dataset	Ours (With the second-order representation)	Ours (With the first-order representation)
King's College	**1.86m**, **2.65°**	1.87m, 2.66°
St Mary's Church	**2.31m**, **4.15°**	2.36m, 4.16°
Old Hospital	2.33m, 2.69°	**2.32m**, **2.65°**
Shop Facade	**1.52m**, **4.02°**	1.65m, 4.07°

In our proposed network, we use 7 independent regression sub-networks in each classification branch, each of the regression sub-networks learns one component of camera pose. To evaluate the effectiveness of this network architecture, we compare it with the network with one unite regression sub-network in each prediction branch. The performances of these two network architectures are reported in Table 6.4. We can see that the network with independent regression sub-networks can achieve lower prediction errors than the network with the united regression sub-network in most of the datasets. Therefore, the designation of independent regression sub-networks is effective in camera pose estimation.

The main reason for the advantage of the independent regression sub-networks is that it can capture the differences of the pose components since it learns different regressors for the pose components. We conduct a toy experiment on the St Mary's Church dataset to validate this reason. We denote the parameters of the sev-

Table 6.4 The median prediction errors of our proposed network with the independent and united regression sub-networks

Dataset	Ours (With the independent regression sub-networks)	Ours (With the united regression sub-networks)
King's College	**1.86m, 2.65°**	1.88m, 2.67°
St Mary's Church	**2.31m, 4.15°**	2.37m, 4.18°
Old Hospital	**2.33m**, 2.69°	2.39m, **2.61°**
Shop Facade	**1.52m, 4.02°**	1.55m, 4.13°

en regression sub-networks by $\mathbf{W_x}$, $\mathbf{W_y}$, $\mathbf{W_z}$, $\mathbf{W_w}$, $\mathbf{W_p}$, $\mathbf{W_q}$ and $\mathbf{W_r}$, and compare the neural impact scores of several different pairs of regression sub-network parameters in Fig. 6.4. It shows that there are significant differences between the parameters of different regression sub-networks. Therefore, we can see that the regressors learned by the regression sub-networks are clearly different.

6.4.2 Comparison between the State-of-the-Art Methods

We compare the performance of our proposed model with several state-of-the-art camera pose estimation approaches, *i.e.* PoseNet [67] and Nearest Neighbor Classifier [67]. Table 6.5 shows the median position and orientation errors of PoseNet [67], Nearest Neighbor [67] and our proposed models. From Table 6.5, we can see that our proposed method has lower prediction error than the PoseNet [67] and Nearest Neighbor Classifier [67] in most datasets. In the Old Hospital and Shop Facade datasets, the prediction translation errors of our proposed method are slightly higher than PoseNet. One possible reason is that the training sets of these two datasets are relatively small. Since our proposed network has more parameters than

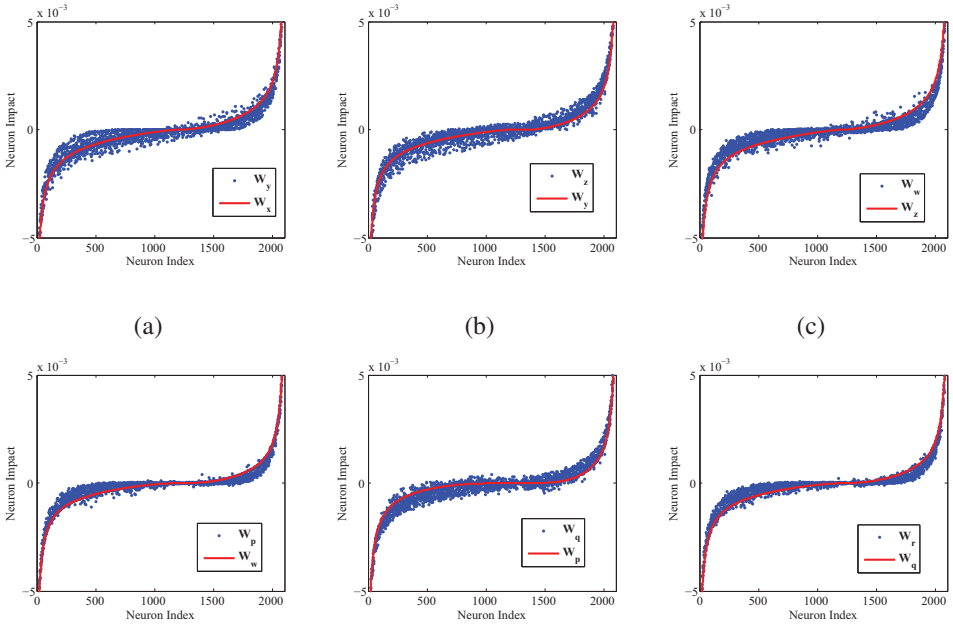


Figure 6.4 The neuron impact scores between different regression sub-networks, (a) $(\mathbf{W_y}, \mathbf{W_x})$, (b) $(\mathbf{W_z}, \mathbf{W_y})$, (c) $(\mathbf{W_w}, \mathbf{W_z})$, (d) $(\mathbf{W_p}, \mathbf{W_w})$, (e) $(\mathbf{W_q}, \mathbf{W_p})$, (f) $(\mathbf{W_r}, \mathbf{W_q})$. For each pair of regression sub-networks (A, B), the neurons are sorted with respect to the neuron impact scores of sub-network B.

Table 6.5 The median prediction errors of our proposed network and the other camera pose estimation methods

Dataset	Nearest Neighbor [67]	PoseNet [67]	Ours
King's College	3.34m, 2.96°	1.92m, 2.70°	**1.86m**, **2.65°**
St Mary's Church	4.48m, 5.65°	2.65m, 4.24°	**2.31m**, **4.15°**
Old Hospital	5.38m, 4.51°	**2.31m**, **2.69°**	2.33m, **2.69°**
Shop Facade	2.10m, 5.20°	**1.46m**, 4.04°	1.52m, **4.02°**

PoseNet, training with small dataset using our proposed network will be easier to lead to overfitting.

6.5 Summary

In this chapter, we proposed a deep siamese network for camera pose estimation, which is considered as a multiple relative order relationship learning problem. This network adopts the relative order loss to learn the relative order relationship of camera pose, and uses the MSE loss to make the predicted camera pose close to the ground-truth. The propose network utilizes the shared spatial sub-network and the separated regression sub-network to learn the generality and specificity of different camera pose components, respectively. The experimental results demonstrated the effectiveness of our proposed model in camera pose estimation.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this thesis, we studied the learning methods of image similarity and relative order relationships as they are two most common image pairwise relationships. For the similarity learning problem, we investigated the Mahalanobis distance metric learning and deep similarity learning, respectively. For the relative order relationship learning problem, we studied both of single and multiple relative order relationship learning.

For the distance metric learning problem, we first proposed a general kernel classification framework which can unify many representative and state-of-the-art metric learning approaches. The proposed framework also provides a good platform for developing new metric learning algorithms. Two metric learning methods, *i.e.*, doublet-SVM and triplet-SVM, were developed and they can be efficiently implemented by the standard SVM solvers. Our experimental results on the handwritten

digit classification and person re-identification tasks showed that doublet-SVM and triplet-SVM are much faster than state-of-the-art methods in terms of training time, while they achieve very competitive results in terms of classification error rate.

As doublet-SVM and triplet-SVM are heuristic methods and cannot obtain the global solution, we then proposed two distance metric learning models, namely PCML and NCML. The proposed models can guarantee the positive semidefinite property of the learned matrix $\mathbf{M}$, and can be solved efficiently by the existing SVM solvers. Experimental results on handwritten digit classification tasks showed that, compared with the state-of-the-art metric learning methods, the proposed PCML and NCML methods can not only achieve favorable classification accuracy, but also are efficient in training. The experimental results on LFW, CUHK01 and CUHK03 databases indicate that the proposed methods also perform well in face verification and person re-identification.

For the deep similarity learning problem, we proposed a joint SIR and PIR learning approach for person re-identification. SIR is efficient in learning and matching, and PIR is effective in modeling the relationship between two images. Based on their connection, we suggest a generalized PIR model to utilize the advantages of both SIR and PIR. Based on the proposed matching score, we present a pairwise comparison model and a triplet comparison model for joint SIR and PIR learning. For each of these two models, we formulate a deep CNN to jointly learn the SIR and PIR. Experimental results demonstrate the effectiveness of joint SIR and PIR learning and show the promising results of our proposed models in person re-identification. In the future, we will investigate the explicit modeling on patch correspondence for SIR and PIR learning and model-level fusion of pairwise and

triplet comparisons.

For the single relative order relationship learning problem, we proposed a model to extend the deep siamese network to learn the relative order relationship of images. We formulated the relative order prediction function based on the second-order image representation. To make the predicted relative order to be consistent with the ground-truth, we introduced the relative order loss to the model. We also used the MSE loss to minimize the error between predicted value and the ground-truth label. To make the learned feature discriminative and stable, we introduced the softmax loss to learn the image feature. The experimental results on relative attribute ranking and age estimation tasks demonstrate that our proposed model perform better than the comparison methods.

For the multiple relative order relationship learning problem, we proposed a deep siamese network for camera pose estimation. This network adopts the relative order loss to learn the relative order relationship of camera pose, and uses the MSE loss to make the predicted camera pose close to the ground-truth. These loss functions are based on the second-order deep representation of the images. The propose network is composed of the shared spatial sub-network and the separated regression sub-network, which learn the generality and specificity of different camera pose components, respectively. The experimental results demonstrated the effectiveness of our proposed model in camera pose estimation.

7.2 Future Work

This thesis shows many potential research directions of learning image pairwise relationships. In the future work, we plan to further improve the proposed methods and develop more works for image pairwise relationship learning.

In this thesis, we have shown that the PIR is effective in capturing the complex relationships of image pairs, and combined the SIR and PIR in deep similarity learning. In the future, we consider to also apply the combination of SIR and PIR to the relative order relationship learning to show whether this methodology is also effective in other kinds of pairwise relationships.

We have worked on similarity and relative order relationships learning in this thesis as they are two most common image pairwise relationships. In the future, we will investigate more kinds of image pairwise relationships and develop their learning and prediction models.

Furthermore, how to effectively extract the potential connection of image pair is also an important issue. Therefore, we will investigate the explicit modeling on patch correspondence for SIR and PIR learning and model-level fusion of image pairwise relationship learning.

Bibliography

[1] E. Ahmed, M. Jones, and T.K. Marks. An improved deep learning architecture for person re-identification. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 3908–3916, 2015.

[2] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *Proceedings of Neural Information Processing Systems*, pages 561–568, 2002.

[3] G. Antipov, M. Baccouche, S. A. Berrani, and J. L. Dugelay. Apparent age estimation from face images combining general and children-specialized deep learning models. In *CVPR Workshop*, 2016.

[4] M. S. Baghshah and S. B. Shouraki. Semi-supervised metric learning using pairwise constraints. In *Proceedings of 21st Int. Joint Conf. Artif. Intell.*, pages 1217–1222, 2009.

[5] M. F. Balcan, A. Blum, and N. Srebro. A theory of learning with similarity functions. *Machine Learning*, 72(1-2):89–112, Aug. 2008.

[6] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of*

Machine Learning Research, 7:2399–2434, 2006.

[7] A. Bellet, A. Habrard, and M. Sebban. Good edit similarity learning by loss minimization. *Machine Learning*, 89(1–2):5–35, Oct. 2012.

[8] A. Bellet, A. Habrard, and M. Sebban. A survey on metric learning for feature vectors and structured data. *arXiv:1306.6709*, 2013.

[9] J. Bi, D. Wu, L. Lu, M. Liu, Y. Tao, and M. Wolf. Adaboost on low-rank psd matrices for metric learning. In *Proceedings of 2011 IEEE International Conference on Computer Vision Pattern Recognition (CVPR 2011)*, pages 2617–2624, 2011.

[10] W. Bian and D. Tao. Constrained empirical risk minimization framework for distance metric learning. *IEEE Transactions on Neural Networks and Learning Systems*, 23(8):1194–1205, 2012.

[11] A. Bordes, L. Bottou, P. Gallinari, and J. Weston. Solving multiclass support vector machines with larank. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 89–96, 2007.

[12] L. Bottou, O. Chapelle, D. DeCoste, and J. Weston. *Large-scale kernel machines*. Cambridge, MA: MIT Press, 2007.

[13] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[14] C. Brunner, A. Fischer, K. Luig, and T. Thies. Pairwise support vector machines and their applications to large scale problems. *Journal of Machine Learning Research*, 13:2279–2292, 2012.

[15] Q. Cao, Y. Ying, and P. Li. Similarity metric learning for face recognition. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*,

pages 2408–2415, 2013.

[16] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2012.

[17] C. C. Chang and C. J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intell. Syst. Technol.*, 2:1–27, 2011.

[18] K. Y. Chang, C. S. Chen, and Y. P. Hung. A ranking approach for human ages estimation based on face images. In *Proceedings of International Conference on Pattern Recognition (ICPR)*, 2010.

[19] K. Y. Chang, C. S. Chen, and Y. P. Hung. Ordinal hyperplanes ranker with cost sensitivities for age estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[20] G. Chechik, V. Sharma, U. Shalit, and S. Bengio. An online algorithm for large scale image similarity learning. In *Proceedings of Neural Information Processing Systems*, pages 306–314, 2009.

[21] G. Checkik, V. Sharma, U. Shalit, and S. Bengio. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11:1109–1135, 2010.

[22] B. C. Chen, C. S. Chen, and W. H. Hsu. Face recognition and retrieval using cross-age reference coding with cross-age celebrity dataset. *IEEE Transactions on Multimedia*, 17(6):804–815, 2015.

[23] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun. Bayesian face revisited: A joint formulation. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 566–579, 2012.

[24] D. Chen, Z. Yuan, B. Chen, and N. Zheng. Similarity learning with spatial constraints for person re-identification. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 1268–1277, 2016.

[25] J. Chen, Z. Zhang, and Y. Wang. Relevance metric learning for person re-identification by exploiting listwise similarities. *IEEE Transactions on Image Processing*, 24(12):4741–4755, 2015.

[26] K. Chen, S. Gong, T. Xiang, and C. C. Loy. Cumulative attribute space for age and crowd density estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[27] L. Chen, Q. Zhang, and B. Li. Predicting multiple attributes via relative multi-task learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[28] S.Z. Chen, C.C. Guo, and J.H. Lai. Deep ranking for person re-identification via joint representation learning. *IEEE Transactions on Image Processing*, 25(5):2353–2367, 2016.

[29] D. Cheng, Y. Gong, S. Zhou, J. Wang, and N. Zheng. Person re-identification by multi-channel parts-based CNN with improved triplet loss function. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 1335–1344, 2016.

[30] S. Choudhary and P. J. Narayanan. Visibility probability structure from sfm datasets and applications. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 130–143, 2012.

[31] R. Collobert, S. Bengio, and Y. Bengio. A parallel mixture of svms for very

large scale problems. *Neural Computation*, 14(5):1105–1114, May 2002.

[32] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

[33] J. Davis, B. Kulis, P. Jain, S. Sra, and I. Dhillon. Information-theoretic metric learning. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 209–216, 2007.

[34] J. Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[35] S. Ding, L. Lin, G. Wang, and H. Chao. Deep feature learning with relative distance comparison for person re-identification. *Pattern Recognition*, 48(10):2993–3003, 2015.

[36] H. Do, A. Kalousis, J. Wang, and A. Woznica. A metric learning perspective of SVM: on the relation of SVM and LMNN. *arXiv:1201.4714*, 2012.

[37] J. Dong, A. Krzyzak, and C. Y. Suen. Fast svm training algorithm with decomposition on very large data sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):603–618, Apr. 2005.

[38] T. Evgeniou, F. France, and M. Pontil. Regularized multi-task learning. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 109–117, 2004.

[39] M. Farenzena, L. Bazzani, A. Perina, V. Murino, and M. Cristani. Person re-identification by symmetry-driven accumulation of local features. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2360–2367, 2010.

[40] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[41] Y. Fu and T. S. Huang. Human age estimation with regression on discriminative aging manifold. *IEEE Transactions on Multimedia*, 10(4):578–584, 2008.

[42] Y. Fu, S. Yan, and T. S. Huang. Correlation metric for generalized feature extraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(12):2229–2235, Dec. 2008.

[43] Z. Fu, G. Lu, K. M. Ting, and D. Zhang. Learning sparse kernel classifiers for multi-instance classification. *IEEE Transactions on Neural Networks and Learning Systems*, 24(9):1377–1389, 2013.

[44] A. Globerson and S. Roweis. Metric learning by collapsing classes. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 451–458, 2005.

[45] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 513–520, 2004.

[46] S. Gong, M. Cristani, S. Yan, and C. C. Loy. *Person Re-identification*. Springer, 2014.

[47] D. Gray and H. Tao. Viewpoint invariant pedestrian recognition with an ensemble of localized features. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 262–275, 2008.

[48] M. Guillaumin, J. Verbeek, and C. Schmid. Is that you? metric learning

approaches for face identification. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 498–505, 2009.

[49] M. Guillaumin, J. Verbeek, and C. Schmid. Multiple instance metric learning from automatically labeled bags of faces. In *Proceedings of 11th European Conf. Comp. Vis.*, pages 634–647, 2010.

[50] G. Guo, Y. Fu, C. R. Dyer, and T. S. Huang. Image-based human age estimation by manifold learning and locally adjusted robust regression. *IEEE Transactions on Image Processing*, 17(7):1178–1188, 2008.

[51] G. Guo and G. Mu. Simultaneous dimensionality reduction and human age estimation via kernel partial least squares regression. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[52] G. Guo, G. Mu, Y. Fu, and T. Huang. Human age estimation using bio-inspired features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[53] G. Guo, G. Mu, Y. Fu, and T. Huang. Joint estimation of age, gender and ethnicity: CCA vs. PLS. In *Proceedings of International Conference on Automatic Face and Gesture Recognition (FGR)*, 2013.

[54] M. Hirzer, P. M. Roth, M. Köstinger, and H. Bischof. Relaxed pairwise learned metric for person re-identification. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 780–793, 2012.

[55] S. Hoi, W. Liu, and S. Chang. Semi-supervised distance metric learning for collaborative image retrieval. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–7, 2008.

[56] J. Hu, J. Lu, and Y. P. Tan. Discriminative deep metric learning for face

verification in the wild. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1875–1882, 2014.

[57] J. Hu, J. Lu, J. Yuan, and Y. P. Tan. Large margin multi-metric learning for face and kinship verification in the wild. In *Proceedings of Asian Conference on Computer Vision (ACCV)*, pages 252–267, 2014.

[58] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Univ. of Massachusetts, 2007.

[59] K. Huang, Y. Ying, and C. Campbell. Gsml: A unified framework for sparse metric learning. In *Proceedings of 9th IEEE International Conference on Data Mining*, pages 189–198, 2009.

[60] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *Proceedings of Neural Information Processing Systems (NIPS)*, 2015.

[61] P. Jain, B. Kulis, J. Davis, and I. S. Dhillon. Metric and kernel learning using a linear transformation. *Journal of Machine Learning Research*, 13(1):519–547, Jan. 2012.

[62] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM International Conference on Multimedia*, pages 675–678, 2014.

[63] R. Jin, S. Wang, and Y. Zhou. Regularized distance metric learning: Theory and algorithm. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 862–870, 2009.

[64] D. Kedem, S. Tyree, K. Weinberger, F. Sha, and G. Lanckriet. Non-linear metric learning. In *Proceedings of Neural Information Processing Systems*, pages 2582–2590, 2012.

[65] S. S. Keerthi, K. B. Duan, S. K. Shevade, and A. N. Poo. A fast dual algorithm for kernel logistic regression. *Machine Learning*, 61(1-3):151–165, Nov. 2005.

[66] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[67] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[68] K. Koh, S. Kim, and S. Boyd. An interior-point method for large-scale l1-regularized logistic regression. *Journal of Machine Learning Research*, 8(8):1519–1555, Dec. 2007.

[69] M. Köstinger, M. Hirzer, P. Wohlhart, P.M. Roth, and H. Bischof. Large scale metric learning from equivalence constraints. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 2288–2295, 2012.

[70] A. Kovashka, D. Parikh, and K. Grauman. Whittlesearch: Image search with relative attribute feedback. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[71] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Proceedings of Neural Information*

Processing Systems (NIPS), 2012.

[72] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *Proceedings of IEEE International Conference on Comput. Vis. (ICCV)*, pages 365–372, 2009.

[73] A. Lanitis, C. Draganova, and C. Christodoulou. Comparing different classifiers for automatic age estimation. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 34(1):621–628, 2004.

[74] W. Li and X. Wang. Locally aligned feature transforms across views. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 3594–3601, 2013.

[75] W. Li, R. Zhao, and X. Wang. Human re-identification with transferred metric learning. In *Proceedings of Asian Conference on Computer Vision (ACCV)*, pages 31–44, 2012.

[76] W. Li, R. Zhao, T. Xiao, and X. Wang. DeepReID: Deep filter pairing neural network for person re-identification. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 152–159, 2014.

[77] Y. Li, N. Snavely, and D. Huttenlocher. Location recognition using prioritized feature matching. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 791–804, 2010.

[78] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide pose estimation using 3d point clouds. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 15–29, 2012.

[79] Z. Li, S. Chang, F. Liang, T. S. Huang, L. Cao, and J. R. Smith. Learning

locally-adaptive decision functions for person verification. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 3610–3617, 2013.

[80] Z. Li, S. Chang, F. Liang, T.S. Huang, L. Cao, and J.R. Smith. Learning locally-adaptive decision functions for person verification. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 3610–3617, 2013.

[81] S. Liao, Y. Hu, X. Zhu, and S.Z. Li. Person re-identification by local maximal occurrence representation and metric learning. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 2197–2206, 2015.

[82] S. Liao and S.Z. Li. Efficient PSD constrained asymmetric metric learning for person re-identification. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 3685–3693, 2015.

[83] L. Lin, G. Wang, W. Zuo, X. Feng, and L. Zhang. Cross-domain visual matching via generalized similarity measure and feature learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1089–1102, 2017.

[84] T. Y. Lin, A. RoyChowdhury, and S. Maji. Bilinear CNN models for fine-grained visual recognition. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2015.

[85] M. Liu and B. C. Vemuri. A robust and efficient doubly regularized metric learning approach. In *Proceedings of European Conference onComputer Vision (ECCV)*, pages 646–659, 2012.

[86] S. Liu, X. Liang, L. Liu, X. Shen, J. Yang, C. Xu, L. Lin, X. Cao, and S. Yan. Matching-CNN meets KNN: Quasi-parametric human parsing. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 1419–1427, 2015.

[87] J. Lu, X. Zhou, Y. Tan, Y. Shang, and J. Zhou. Neighborhood repulsed metric learning for kinship verification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(2):331–345, Feb. 2014.

[88] N. Martinel, C. Micheloni, and G. L. Foresti. Saliency weighted features for person re-identification. In *ECCV Workshop on Visual Surveillance and Re-identification*, pages 191–208, 2014.

[89] B. McFee and G. Lanckriet. Metric learning to rank. In *Proceedings of 27th International Conference on Machine Learning (ICML 2010)*, pages 775–782, 2010.

[90] N. McLaughlin, J. Martinez del Rincon, and P. Miller. Recurrent convolutional network for video-based person re-identification. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 1325–1334, 2016.

[91] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu. Relative camera pose estimation using convolutional neural networks. *arXiv:1702.01381*, 2017.

[92] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Metric learning for large scale image classification: generalizing to new classes at near-zero cost. In *Proceedings of European Conference onComputer Vision (ECCV)*, pages 488–501, 2012.

[93] A. Mignon and F. Jurie. PCCA: A new approach for distance learning from

sparse pairwise constraints. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 2666–2672, 2012.

[94] K. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.

[95] H. V. Nguyen and L. Bai. Cosine similarity metric learning for face verification. In *Proceedings of 10th Asian Conf. Comp. Vis.*, pages 709–720, 2010.

[96] N. Nguyen and Y. Guo. Metric learning: A support vector approach. In *Proceedings of ECML/PKDD*, pages 125–136, 2008.

[97] G. Niu, B. Dai, M. Yamada, and M. Sugiyama. Information-theoretic semi-supervised metric learning via entropy regularization. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 89–96, 2012.

[98] Z. Niu, M. Zhou, L. Wang, X. Gao, and G. Hua. Ordinal regression with multiple output CNN for age estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[99] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, May 2001.

[100] S. Parameswaran and K. Weinberger. Large margin multi-task metric learning. In *Proceedings of Neural Information Processing Systems*, pages 1867–1875, 2010.

[101] D. Parikh and K. Grauman. Relative attributes. In *Proceedings of IEEE*

International Conference on Computer Vision (ICCV), 2011.

[102] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *Proceedings of British Machine Vision Conference (BMVC)*, pages 41.1–41.12, 2015.

[103] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods: Support Vector Learning*, pages 185–208. Cambridge, MA: MIT Press, 1999.

[104] G.-J. Qi, J. Tang, Z.-J. Zha, T.-S. Chua, and H.-J. Zhang. An efficient sparse metric learning in high-dimensional space via l_1-penalized log-determinant regularization. In *Proceedings of International Conference on Machine Learning*, pages 841–848, 2009.

[105] Q. Qian, R. Jin, J. Yi, L. Zhang, and S. Zhu. Efficient distance metric learning by adaptative sampling and mini-batch stochastic gradient descent (sdd). *arXiv: 1304.1192v1*, 2013.

[106] K. Ricanek and T. Tesafaye. MORPH: a longitudinal image database of normal adult age-progression. In *Proceedings of International Conference on Automatic Face and Gesture Recognition (FGR)*, 2006.

[107] R. Rothe, R. Timofte, and L. V. Gool. Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision*, August 2016.

[108] R. Rothe, R. Timofte, and L. V. Gool. Some like it hot - visual guidance for preference prediction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[109] R. N. Sandeep, Y. Verma, and C. V. Jawahar. Relative parts: Distinctive parts

for learning relative attributes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[110] T. Sattler, B. Leibe, and L. Kobbelt. Improving image-based localization by active correspondence search. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 752–765, 2012.

[111] B. Scholkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, Jul. 2001.

[112] F. Schroff, D. Kalenichenko, and J. Philbin. FaceNet: A unified embedding for face recognition and clustering. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 815–823, 2015.

[113] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. Pegasos: primal estimated sub-gradient solver for SVM. *Mathematical Programming*, 127(1):3–30, 2011.

[114] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge Univ. Press, 2004.

[115] C. Shen, J. Kim, F. Liu, L. Wang, and A. Hengel. Efficient dual approach to distance metric learning. *IEEE Transactions on Neural Networks and Learning Systems*, 25(2):394–406, 2014.

[116] C. Shen, J. Kim, and L. Wang. Scalable large-margin mahalanobis distance metric learning. *IEEE Transactions on Neural Networks*, 21(9):1524–1530, 2010.

[117] C. Shen, J. Kim, and L. Wang. A scalable dual approach to semidefinite met-

ric learning. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 2601–2608, 2011.

[118] C. Shen, J. Kim, L. Wang, and A. Hengel. Positive semidefinite metric learning with boosting. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 1651–1659, 2009.

[119] C. Shen, J. Kim, L. Wang, and A. Hengel. Positive semidefinite metric learning using boosting-like algorithms. *Journal of Machine Learning Research*, 13:1007–1036, 2012.

[120] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

[121] K. K. Singh and Y. J. Lee. End-to-end localization and ranking for relative attributes. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2016.

[122] L. Svarm, O. Enqvist, M. Oskarsson, and F. Kahl. Accurate localization and pose estimation for large 3d models. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 532–539, 2014.

[123] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[124] C. H. Teo, Q. Le, A. Smola, and S. V. N. Vishwanathan. A scalable modular convex solver for regularized risk minimization. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Min-*

ing, pages 727–736, 2007.

[125] L. Torresani and K. Lee. Large margin component analysis. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 1385–1392, 2006.

[126] D. Tran and A. Sorokin. Human activity recognition with metric learning. In *Proceedings of European Conference onComputer Vision (ECCV)*, pages 548–561, 2008.

[127] I. W. Tsang, J. T. Kwok, and P. M. Cheung. Core vector machines: Fast SVM training on very large data sets. *Journal of Machine Learning Research*, 3:363–392, 2005.

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

[129] R. Vezzani, D. Baltieri, and R. Cucchiara. People reidentification in surveillance and forensics: A survey. *ACM Computing Surveys (CSUR)*, 46(2):29, 2013.

[130] F. Wang, W. Zuo, L. Lin, D. Zhang, and L. Zhang. Joint learning of single-image and cross-image representations for person re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1288–1296, June 2016.

[131] F. Wang, W. Zuo, L. Zhang, D. Meng, and D. Zhang. A kernel classification framework for metric learning. *IEEE Transactions on Neural Networks and Learning Systems*, 26(9):1950–1962, 2015.

[132] J. Wang, H. Do, A. Woznica, and A. Kalousis. Metric learning with multiple kernels. In *Proceedings of Neural Information Processing Systems*, pages 1170–1178, 2011.

[133] J. Wang, A. Woznica, and A. Kalousis. Learning neighborhoods for metric learning. *arXiv: 1206.6883v1*, 2012.

[134] J. Wang, A. Woznica, and A. Kalousis. Parametric local metric learning for nearest neighbor classification. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 1610–1618, 2012.

[135] Q. Wang, P. C. Yuen, and G. Feng. Semi-supervised metric learning via topology preserving multiple semi-supervised assumptions. *Pattern Recognition*, 46(9):2576–2587, Mar. 2013.

[136] X. Wang. Intelligent multi-camera video surveillance: A review. *Pattern Recognition Letters*, 34:3–19, 2013.

[137] X. Wang, R. Guo, and C. Kambhamettu. Deeply-learned feature for age estimation. In *WACV*, 2015.

[138] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 1473–1480, 2005.

[139] K. Q. Weinberger and L. K. Saul. Fast solvers and efficient implementations for distance metric learning. In *Proceedings of 25th International Conference on Machine Learning (ICML 2008)*, pages 1160–1167, 2008.

[140] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

[141] F. Xiao and Y. J. Lee. Discovering the spatial extent of relative attributes. In *IEEE Conference on Computer Vision (ICCV)*, 2015.

[142] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learn-

ing with application to clustering with side-information. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 505–512, 2002.

[143] F. Xiong, M. Gou, O. Camps, and M. Sznaier. Person re-identification using kernel-based metric learning methods. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 1–16, 2014.

[144] Y. Xu and W. Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on Imaging Sciences*, 6(3):1758–C1789, 2013.

[145] P. Yang, K. Huang, and C. L. Liu. A multi-task framework for metric learning with common subspace. *Neural Computing Applicat.*, 22(7–8):1337–1347, June 2012.

[146] X. Yang, T. Zhang, C. Xu, S. Yan, M. S. Hossain, and A. Ghoneim. Deep relative attributes. *IEEE Transactions on Multimedia*, 18(9):1832–1842, 2016.

[147] D. Yi, Z. Lei, S. Liao, and S.Z. Li. Deep metric learning for person re-identification. In *Proceedings of International Conference on Pattern Recognition (ICPR)*, pages 34–39, 2014.

[148] Y. Ying, K. Huang, and C. Campbell. Sparse metric learning via smooth optimization. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 2214–2222, 2009.

[149] Y. Ying and P. Li. Distance metric learning with eigenvalue optimization. *Journal of Machine Learning Research*, 13:1–26, 2012.

[150] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *IEEE Conference on Computer Vision and*

Pattern Recognition (CVPR), 2015.

[151] L. Zhang, T. Xiang, and S. Gong. Learning a discriminative null space for person re-identification. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 1239–1248, 2016.

[152] R. Zhang, L. Lin, R. Zhang, W. Zuo, and L. Zhang. Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification. *IEEE Transactions on Image Processing*, 24(12):4766–4779, Dec. 2015.

[153] Z. Zhang, Y. Chen, and V. Saligrama. A novel visual word co-occurrence model for person re-identification. In *ECCV Workshop on Visual Surveillance and Re-identification*, pages 122–133, 2014.

[154] R. Zhao, W. Ouyang, and X. Wang. Person re-identification by salience matching. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 2528–2535, 2013.

[155] R. Zhao, W. Ouyang, and X. Wang. Unsupervised salience learning for person re-identification. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 3586–3593, 2013.

[156] R. Zhao, W. Ouyang, and X. Wang. Learning mid-level filters for person re-identification. In *Proceedings of IEEE International Conference on Computer Vision Pattern Recognition (CVPR)*, pages 144–151, 2014.

[157] L. Zheng, K. Idrissi, C. Garcia, S. Duffner, and A. Baskurt. Triangular similarity metric learning for face verification. In *Proceedings of International Conference on Automatic Face and Gesture Recognition (FGR)*, 2015.

[158] W. Zuo, F. Wang, D. Zhang, L. Lin, Y. Huang, D. Meng, and L. Zhang.

Distance metric learning via iterated support vector machines. *IEEE Transactions on Image Processing*, 26(10):4937–4950, Oct 2017.