Chapter 2
Metric Learning with Biometric Applications

Abstract Learning a desired distance metric from given training samples plays a significant role in the field of machine learning. In this chapter, we first present two novel metric learning methods based on a support vector machine (SVM). We then present a kernel classification framework for metric learning that can be implemented efficiently by using the standard SVM solvers. Some novel kernel metric learning methods, such as the double-SVM and the triplet-SVM, are also introduced in this chapter.

2.1 Introduction

Distance metric learning aims to train a valid distance metric that enables samples of different classes to have larger distances and samples of the same class to have smaller distances (Bellet et al. 2013). Distance metric learning has been adopted successfully in many real-world applications, such as face verification (Guillaumin et al. 2009), object classification (Mensink et al. 2012), and visual tracking (Li et al. 2012).

In past years, numerous distance metric learning algorithms have been proposed (Balcan et al. 2008; Kedem et al. 2012; Guillaumin et al. 2009; Fu et al. 2008; Wang et al. 2011). On the basis of the availability of labels, available distance metric learning algorithms can be categorized into two groups: supervised distance metric learning and unsupervised distance metric learning (Yang and Jin 2006). Unsupervised distance metric learning can be viewed as a dimension reduction approach, including principal component analysis (PCA) and locally linear embedding (LLE), which aim to learn an underlying low-dimensional manifold without label information. Supervised distance metric learning methods utilize the information of class labels to improve the discrimination of the distance metric, and can be further divided into two groups: learn metrics with triplet or pairwise constraints. For each triplet, triplet constraint-based metric learning approaches restrict that the distance between a pair of samples from the same class should be smaller than that of those from different classes. The large margin nearest neighbor

D. Zhang et al., *Discriminative Learning in Biometrics*,
DOI 10.1007/978-981-10-2056-8_2

(LMNN) (Weinberger et al. 2009), BoostMetric (Shen et al. 2009) and FrobMetric (Shen et al. 2011) are typical triplet constraint-based metric learning methods. Compared to triplet constraint-based metric learning methods, pairwise constraint-based methods are more generally used in real applications of metric learning. For example, only pairwise constraint-based methods are available for face verification, particularly for the LFW face database (Huang et al. 2007). The neighborhood-component analysis method (NCA) (Goldberger et al. 2004), the information theoretic metric learning algorithm (ITML) (Davis et al. 2007), and the logistic discriminative-based metric learning method (LDML) (Guillaumin et al. 2009) are typical pairwise constraint-based metric learning methods. The NCA method learns a distance metric from the input data by finding a linear transformed space that has the maximum performance of the average leave-one-out classification. Globerson and Roweis (2005) proposed an effective metric learning method by maximizing intraclass distances while collapsing the interclass distance to zero. Huang et al. (2012) proposed a distance metric learning method which restricts distances between the same classes to be smaller than that of those from different classes. Both pairwise constraint-based and triplet constraint-based metric learning methods work in a fully supervised metric learning manner.

The remainder of this chapter is organized as follows. Section 2.2 presents two novel distance metric learning models based on SVM. In Sect. 2.3, we present a kernel classification framework for metric learning. The framework provides a new perspective on developing metric learning methods. Based on this framework, two novel kernel metric learning methods, doublet-SVM and triplet-SVM, are also presented.

2.2 Support Vector Machines for Metric Learning

2.2.1 *Positive Semidefinite Constrained Metric Learning (PCML)*

Suppose $\{(\mathbf{x}_i, y_i)|i = 1, 2, \ldots, N\}$ represents a training set, where $\mathbf{x}_i \in \mathbb{R}^d$ and y_i denote the ith training sample and its class label. The Mahalanobis distance of two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ is defined as

$$d^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) = \mathrm{tr}\left(\mathbf{M}^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}\right) = \left\langle \mathbf{M}, (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}} \right\rangle, \tag{2.1}$$

where $\mathbf{M}$ is a positive semidefinite (PSD) matrix, $\langle \mathbf{A}, \mathbf{B} \rangle = \mathrm{tr}(\mathbf{A}^{\mathrm{T}}\mathbf{B})$ is defined as the Frobenius inner product of two matrices $\mathbf{A}$ and $\mathbf{B}$, and $\mathrm{tr}(\cdot)$ denotes the matrix trace operator. For each pair samples $\mathbf{x}_i$ and $\mathbf{x}_j$, we define $\mathbf{X}_{ij} = (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}$. Thus, the Mahalanobis distance can be rewritten as $d^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{M}, \mathbf{X}_{ij} \rangle$.

2.2.1.1 PCML and Its Dual Problem

Suppose $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{x}_j)$: $\mathbf{x}_i$ and $\mathbf{x}_j$ have the same class labels$\}$ represents the set of similar pairs, and $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{x}_j) : \mathbf{x}_i$ and $\mathbf{x}_j$ have different class labels$\}$ denotes the set of dissimilar pairs. The objective function of the PCML model is

$$\begin{aligned} \min_{\mathbf{M},b,\xi} \quad & \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum_{ij}\xi_{ij} \\ \text{s.t.} \quad & h_{ij}\left(\langle \mathbf{M}, \mathbf{X}_{ij}\rangle + b\right) \geq 1 - \xi_{ij}, \quad \xi_{ij} \geq 0, \quad \forall\ i,j, \mathbf{M} \succeq 0, \end{aligned} \tag{2.2}$$

where ξ_{ij} is the slack variables, b is the bias, and $\|\cdot\|_F$ denotes the Frobenius norm. h_{ij} is an indicator variable, and is defined as:

$$h_{ij} = \begin{cases} 1, & \text{if } (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{D} \\ -1, & \text{if } (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{S}. \end{cases} \tag{2.3}$$

The PCML model is convex and can be solved by the standard semidefinite programming (SDP) solvers. However, the general-purpose interior-point SDP solver is very complex, which is not suitable for metric learning with large samples. In order to improve the efficiency of the Mahalanobis distance metric learning, we present an iterating SVM training algorithm based on the PSD projection in this chapter.

First, a Lagrange multiplier λ and a PSD matrix $\mathbf{Y}$ are introduced to the PCML model. The Lagrange dual problem of the PCML model in Eq. (2.3) can be formulated as

$$\begin{aligned} \max_{\lambda,\mathbf{Y}} \quad & -\frac{1}{2}\left\|\sum_{ij}\lambda_{ij}h_{ij}\mathbf{X}_{ij} + \mathbf{Y}\right\|_F^2 + \sum_{ij}\lambda_{ij} \\ \text{s.t.} \quad & \sum_{ij}\lambda_{ij}h_{ij} = 0, \quad 0 \leq \lambda_{ij} \leq C, \quad \mathbf{Y} \succeq 0. \end{aligned} \tag{2.4}$$

The detailed derivation of the dual problem can be found in Appendix 2.1. Based on the Karush-Kuhn-Tucker (KKT) conditions, matrix $\mathbf{M}$ can be calculated by

$$\mathbf{M} = \sum_{ij}\lambda_{ij}h_{ij}\mathbf{X}_{ij} + \mathbf{Y}. \tag{2.5}$$

Based on the strong duality, if the Lagrange dual problem in Eq. (2.4) can be solved, matrix $\mathbf{M}$ can also be obtained by Eq. (2.5). However, due to the PSD constraint $\mathbf{Y} \succeq 0$, the Lagrange dual problem in Eq. (2.4) is still difficult to solve.

2.2.1.2 Alternative Optimization Algorithm

In order to improve the efficiency of general SDP solvers, we present an optimization scheme by updating λ and $\mathbf{Y}$ alternatively. When $\mathbf{Y}$ is obtained, we introduce a new variable η with $\eta_{ij} = 1 - h_{ij}\langle \mathbf{X}_{ij}, \mathbf{Y} \rangle$ to obtain λ. Therefore, λ can be updated by solving the following formula:

$$\begin{aligned} \max_{\lambda} \quad & -\frac{1}{2}\sum_{ij}\sum_{kl}\lambda_{ij}\lambda_{kl}h_{ij}h_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + \sum_{ij}\eta_{ij}\lambda_{ij} \\ \text{s.t.} \quad & \sum_{ij}\lambda_{ij}h_{ij} = 0, \quad 0 \le \lambda_{ij} \le C, \ \ \forall \ \ i,j. \end{aligned} \tag{2.6}$$

The above subproblem is a QP problem, and can be solved efficiently by using existing SVM solvers, such as LibSVM (Chang and Lin 2011). When λ is updated by solving Eq. (2.6), $\mathbf{Y}$ also can be updated by solving the following objective function:

$$\min_{\mathbf{Y}} \quad \|\mathbf{Y} - \mathbf{Y}_0\|_F^2 \quad \text{s.t.} \quad \mathbf{Y} \succeq 0, \tag{2.7}$$

where $\mathbf{Y}_0 = -\sum_{ij}\lambda_{ij}h_{ij}\mathbf{X}_{ij}$. Through the eigen-decomposition of $\mathbf{Y}_0$, i.e., $\mathbf{Y}_0 = \mathbf{U}\Lambda\mathbf{U}^{\mathrm{T}}$, Y can be rewritten as $\mathbf{Y} = \mathbf{U}\Lambda_+\mathbf{U}^{\mathrm{T}}$, where $\Lambda_+ = \max(\Lambda, 0)$, Λ denotes the diagonal matrix of eigenvalues. Finally, the alternative optimization scheme of PCML algorithm is summarized in Algorithm 1.

Algorithm 2.1 Algorithm of PCML

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

Output: $\mathbf{M}$.

1. **Initialize** $\mathbf{Y}^{(0)}$, $t \leftarrow 0$.
2. **Repeat**
3. Update $\eta^{(t+1)}$ with $\eta_{ij}^{(t+1)} = 1 - h_{ij}\langle \mathbf{X}_{ij}, \mathbf{Y}^{(t)} \rangle$.
4. Update $\lambda^{(t+1)}$ by solving the subproblem (2.6) using an SVM solver.
5. Update $\mathbf{Y}_0^{(t+1)} = -\sum_{ij}\lambda_{ij}^{(t+1)}h_{ij}\mathbf{X}_{ij}$.
6. Update $\mathbf{Y}^{(t+1)} = \mathbf{U}^{(t+1)}\Lambda_+^{(t+1)}\mathbf{U}^{(t+1)\mathrm{T}}$, where $\mathbf{Y}_0^{(t+1)} = \mathbf{U}^{(t+1)}\Lambda^{(t+1)}\mathbf{U}^{(t+1)\mathrm{T}}$ and $\Lambda_+^{(t+1)} = \max\left(\Lambda^{(t+1)}, 0\right)$.
7. $t \leftarrow t + 1$.
8. **Until** convergence
9. $\mathbf{M} = \sum_{ij}\lambda_{ij}^{(t)}h_{ij}\mathbf{X}_{ij} + \mathbf{Y}^{(t)}$.
10. **Return M**

2.2.1.3 Optimality Condition

The general alternating minimization approach would converge to the correct solution (Gunawardana and Byrne 2005; Csisz and Tusnády 1984). By updating λ and $\mathbf{Y}$ alternatively, the presented optimization PCML algorithm can find the global optimum of the problems in Eqs. (2.2) and (2.4) quickly. Figure 2.1 shows an example convergence curve of the PCML algorithm in the *PenDigits dataset*, and we can see that it converges in less than 20 iterations.

We further use the duality gap in each iteration to verify the optimality condition of the presented optimization PCML algorithm. The duality gap is defined as the difference between the primal and dual objective values, and is calculated as

$$\mathrm{DualGap}_{\mathrm{PCML}}^{(n)} = \frac{1}{2}\left\|\mathbf{M}^{(n)}\right\|_F^2 + C\sum_{ij}\xi_{ij}^{(n)} - \sum_{ij}\lambda_{ij}^{(n)} + \frac{1}{2}\left\|\sum_{ij}\lambda_{ij}^{(n)}h_{ij}\mathbf{X}_{ij} + \mathbf{Y}^{(n)}\right\|_F^2, \tag{2.8}$$

where $\mathrm{DualGap}_{\mathrm{PCML}}^{(n)}$ denotes the duality gap in the nth iteration, and $\mathbf{M}^{(n)}$, $\xi^{(n)}$, $\lambda^{(n)}$, and $\mathbf{Y}^{(n)}$ are feasible primal and dual variables. According to Eq. (2.5), we can derive that

$$\mathbf{M}^{(n)} = \sum_{i,j}\lambda_{ij}^{(n)}h_{ij}\mathbf{X}_{ij} + \mathbf{Y}^{(n)} = \mathbf{Y}^{(n)} - \mathbf{Y}_0^{(n)}. \tag{2.9}$$

As discussed in Sect. 2.2.1.2, $\mathbf{Y}_0^{(n)} = \mathbf{U}^{(n)}\Lambda^{(n)}\mathbf{U}^{(n)\mathrm{T}}$, $\mathbf{Y}^{(n)} = \mathbf{U}^{(n)}\Lambda_+^{(n)}\mathbf{U}^{(n)\mathrm{T}}$; therefore, $\mathbf{M}^{(n)} = \mathbf{U}^{(n)}\Lambda_-^{(n)}\mathbf{U}^{(n)\mathrm{T}}$, where $\Lambda_-^{(n)} = \Lambda_+^{(n)} - \Lambda^{(n)}$. Thus, $\left\|\mathbf{M}^{(n)}\right\|_F^2$ can be calculated by

$$\begin{aligned}\left\|\mathbf{M}^{(n)}\right\|_F^2 &= \mathrm{tr}\left(\mathbf{M}^{(n)T}\mathbf{M}^{(n)}\right) = \mathrm{tr}\left(\mathbf{U}^{(n)}\Lambda_-^{(n)}\mathbf{U}^{(n)T}\mathbf{U}^{(n)}\Lambda_-^{(n)}\mathbf{U}^{(n)\mathrm{T}}\right)\\ &= \mathrm{tr}\left(\mathbf{U}^{(n)}\Lambda_-^{(n)2}\mathbf{U}^{(n)\mathrm{T}}\right) = \mathrm{tr}\left(\Lambda_-^{(n)2}\right).\end{aligned} \tag{2.10}$$

From Eqs. (2.8)–(2.10), we can obtain the duality gap

$$\mathrm{DualGap}_{\mathrm{PCML}}^{(n)} = C\sum_{i,j}\xi_{ij}^{(n)} - \sum_{i,j}\lambda_{ij}^{(n)} + \mathrm{tr}\left(\Lambda_-^{(n)2}\right). \tag{2.11}$$

Based on the KKT conditions of the PCML dual problem in Eq. (2.4), $\xi_{ij}^{(n)}$ can be obtained by

$$\xi_{ij}^{(n)} = \begin{cases} 0 & \text{for all } \lambda_{ij}^{(n)} < C \\ \left[1 - h_{ij}\left(\left\langle \mathbf{M}^{(n)}, \mathbf{X}_{ij}\right\rangle + b^{(n)}\right)\right]_+ & \text{for all } \lambda_{ij}^{(n)} = C, \end{cases} \tag{2.12}$$

where

$$b^{(n)} = \frac{1}{h_{ij}} - \left\langle \mathbf{M}^{(n)}, \mathbf{X}_{ij} \right\rangle \quad \text{for all } 0 < \lambda_{ij}^{(n)} < C. \tag{2.13}$$

The detailed derivation of $\xi_{ij}^{(n)}$ and $b^{(n)}$ can be found in Appendix 2.1. The duality gap is always nonnegative, and approaches zero when the primal problem is convex. Thus, it can be used as the termination condition of the presented iterative algorithm. From Fig. 2.1, we can see that the duality gap would converge to zero, which indicates that the presented algorithm would reach the global optimum. In the implementation of Algorithm 2.1, the following termination condition is adopted:

$$\text{DualGap}_{\text{PCML}}^{(t)} - \text{DualGap}_{\text{PCML}}^{(t-1)} < \varepsilon \cdot \text{DualGap}_{\text{PCML}}^{(1)}, \tag{2.14}$$

where ε is a small constant and is set at 0.01 in the experiment.

2.2.1.4 Remarks

Warm start: In the presented optimization PCML algorithm, we use a simple warm-start strategy to iteratively calculate λ. The solution to the previous iteration is used as the initialization for the next iteration. By using the iteration scheme, the optimal λ can be obtained rapidly, and the efficiency of the PCML algorithm can be greatly improved.

Construction of pairwise constraints: Suppose there are N^2 pairwise constraints in the training set. This involves a high computation cost as a result of using the N^2 pairwise constraints directly for metric learning. In practice, we can reduce the computational cost by using a subset of pairwise constraints rather than the entire dataset. For each sample, we select its k-nearest neighbors to construct similar pairs and its k farthest neighbors to construct dissimilar pairs. Thus, $2kN$ pairwise

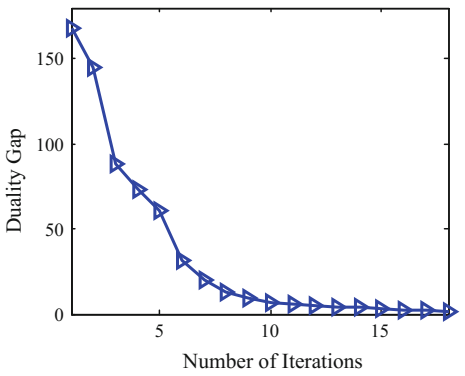

Fig. 2.1 Duality gap versus number of iterations in the PenDigits dataset for PCML (Zuo et al. 2015)

constraints are achieved in total. In practice, k is generally chosen as $1 \sim 3$. Thus, compared with N^2 pairwise constraints, the $2kN$ pairwise constraints can improve the efficiency effectively.

Computational Complexity: In the training of SVM, we use the LibSVM library, where the computational complexity of SMO-type algorithms (Platt 1999) is $O(N^2d)$. For PSD projection, the complexity of the conventional SVD algorithm is $O(d^3)$.

2.2.2 Nonnegative-Coefficient Constrained Metric Learning (NCML)

Given a set of rank-1 PSD matrices $\mathbf{M}_t = \mathbf{m}_t\mathbf{m}_t^{\mathrm{T}}$ ($t = 1, \ldots, T$), a linear combination of $\mathbf{M}_t$ is defined as $\mathbf{M} = \sum_t \alpha_t\mathbf{M}_t$, where αt denotes the scalar combination coefficient. One can easily prove the following Theorem 2.1.

Theorem 2.1 *Assume the scalar coefficient* $\alpha_t \geq 0, \forall\, t$, *matrix* $\mathbf{M} = \sum_t \alpha_t\mathbf{M}_t$ *is a PSD matrix, where* $\mathbf{M}_t = \mathbf{m}_t\mathbf{m}_t^{\mathrm{T}}$ *is a rank-1 PSD matrix.*

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

$$\mathbf{u}^{\mathrm{T}}\mathbf{M}\mathbf{u} = \mathbf{u}^{\mathrm{T}}\left(\sum_t \alpha_t\mathbf{m}_t\mathbf{m}_t^{\mathrm{T}}\right)\mathbf{u} = \sum_i \alpha_i\mathbf{u}^{\mathrm{T}}\mathbf{m}_t\mathbf{m}_t^{\mathrm{T}}\mathbf{u} = \sum_i \alpha_i\left(\mathbf{u}^{\mathrm{T}}\mathbf{m}_t\right)^2.$$

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

2.2.2.1 NCML and Its Dual Problem

Inspired by Theorem 2.1, we present a nonnegative-coefficient constrained metric learning (NCML) method by reparameterizing the distance metric $\mathbf{M}$. Given the training data $\mathcal{S}$ and $\mathcal{D}$, a rank-1 PSD matrix $\mathbf{X}_{ij}$ can be constructed for each pair of $(\mathbf{x}_i, \mathbf{x}_j)$. By assuming that the learned matrix should be the linear combination of $\mathbf{X}_{ij}$ with the nonnegative-coefficient constraint, the NCML model can be formulated as

$$\begin{aligned} &\min_{\mathbf{M},b,\alpha,\xi} \quad \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum_{ij}\xi_{ij} \\ &\text{s.t.} \quad h_{ij}\left(\langle\mathbf{M}, \mathbf{X}_{ij}\rangle + b\right) \geq 1 - \xi_{ij}, \xi_{ij} \geq 0, \quad \alpha_{ij} \geq 0, \quad \forall\, i,j, \quad \mathbf{M} = \sum_{ij}\alpha_{ij}\mathbf{X}_{ij}. \end{aligned} \tag{2.15}$$

By substituting $\sum_{ij} \alpha_{ij}\mathbf{X}_{ij}$ for $\mathbf{M}$, the NCML model is transformed as follows:

$$\begin{aligned} \min_{\alpha,b,\xi} \quad & \frac{1}{2}\sum_{ij}\sum_{kl}\alpha_{ij}\alpha_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + C\sum_{ij}\xi_{ij} \\ \text{s.t.} \quad & h_{ij}\left(\sum_{kl}\alpha_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + b\right) \geq 1-\xi_{ij}, \quad \xi_{ij}\geq 0, \quad \alpha_{ij}\geq 0, \quad \forall\, i,j. \end{aligned} \tag{2.16}$$

By introducing two variables η and β, the Lagrange dual of the primal problem in Eq. (2.16) can be formulated as

$$\begin{aligned} \max_{\eta,\beta} \quad & -\frac{1}{2}\sum_{ij}\sum_{kl}(\beta_{ij}h_{ij}+\eta_{ij})(\beta_{kl}h_{kl}+\eta_{kl})\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + \sum_{ij}\beta_{ij} \\ \text{s.t.} \quad & \sum_{kl}\eta_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle \geq 0, \quad 0\leq \beta_{ij}\leq C, \quad \forall i,j, \quad \sum_{ij}\beta_{ij}h_{ij}=0. \end{aligned} \tag{2.17}$$

The detailed derivation of the dual problem can be found in Appendix 2.2. Based on KKT conditions of the dual problem, coefficient α_{ij} can be obtained by

$$\alpha_{ij} = \beta_{ij}h_{ij} + \eta_{ij}. \tag{2.18}$$

Matrix $\mathbf{M}$ can then be obtained by

$$\mathbf{M} = \sum_{ij}(\beta_{ij}h_{ij}+\eta_{ij})\mathbf{X}_{ij}. \tag{2.19}$$

2.2.2.2 Optimization Algorithm

If the two Lagrange multipliers η and β are obtained from Eq. (2.19), the distance metric can also be obtained. In this subsection, an alternative optimization approach to calculate the two groups of variables is presented. First, assuming η is known, variable β can be obtained by solving the following formula:

$$\begin{aligned} \max_{\beta} \quad & -\frac{1}{2}\sum_{ij}\sum_{kl}\beta_{ij}\beta_{kl}h_{ij}h_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + \sum_{ij}\delta_{ij}\beta_{ij} \\ \text{s.t.} \quad & 0\leq \beta_{ij}\leq C, \quad \forall i,j, \quad \sum_{ij}\beta_{ij}h_{ij}=0, \end{aligned} \tag{2.20}$$

where $\delta_{ij} = 1 - h_{ij}\sum_{kl}\eta_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle$. β also can be solved efficiently by the standard SVM solvers, such as LibSVM (Chang and Lin 2011).

After β has been solved, η can be obtained by solving the following formula:

$$\begin{aligned} \min_{\eta} \quad & \frac{1}{2}\sum_{ij}\sum_{kl}\eta_{ij}\eta_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + \sum_{ij}\eta_{ij}\gamma_{ij} \\ \text{s.t.} \quad & \sum_{kl}\eta_{ij}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle \geq 0, \quad \forall i,j, \end{aligned} \tag{2.21}$$

where $\gamma_{ij} = \sum_{kl}\beta_{kl}h_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle$. Based on the KKT condition of the Lagrange dual problem, the subproblem of η can be simplified as:

$$\eta_{ij} = \mu_{ij} - h_{ij}\beta_{ij}, \quad \forall i,j, \tag{2.22}$$

where μ is the Lagrange dual multiplier. The Lagrange dual problem of Eq. (2.21) is transformed as follows:

$$\begin{aligned} \max_{\mu} \quad & -\frac{1}{2}\sum_{ij}\sum_{kl}\mu_{ij}\mu_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + \sum_{ij}\gamma_{ij}\mu_{ij} \\ \text{s.t.} \quad & \mu_{ij} \geq 0, \quad \forall i,j. \end{aligned} \tag{2.23}$$

The detailed derivation of above dual problem can be found in Appendix 2.3. The Lagrange dual problem of Eq. (2.23) can also be solved efficiently by standard SVM solvers, such as LibSVM.

After updating μ and β alternatively, the optimal solutions of μ and β can be obtained; the optimal solution for α in Eq. (2.16) can then be obtained as follows:

$$\alpha_{ij} = \mu_{ij}, \quad \forall i,j, \tag{2.24}$$

and the distance metric matrix $\mathbf{M} = \sum_{ij}\alpha_{ij}\mathbf{X}_{ij}$ is then obtained. The presented NCML algorithm is summarized in Algorithm 2.2.

Algorithm 2.2 Algorithm of NCML
Input: Training set $\{(\mathbf{x}_i, \mathbf{x}_j), h_{ij}\}$.
Output: The matrix $\mathbf{M}$.

1. **Initialize** $\eta^{(0)}$ with small random values, $t \leftarrow 0$.
2. **Repeat**
3. Update $\delta^{(t+1)}$ with $\delta_{ij}^{(t+1)} = 1 - h_{ij}\sum_{kl}\eta_{kl}^{(t)}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle$.
4. Update $\beta^{(t+1)}$ by solving the subproblem (2.20) using an SVM solver.
5. Update $\gamma^{(t+1)}$ with $\gamma_{ij}^{(t+1)} = \sum_{kl}\beta_{kl}^{(t+1)}h_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle$.
6. Update $\mu^{(t+1)}$ by solving the subproblem (2.23) using an SVM solver.
7. Update $\eta^{(t+1)}$ with $\eta_{ij}^{(t+1)} \leftarrow \mu_{ij}^{(t+1)} - h_{ij}\beta_{ij}^{(t+1)}$.

8. $t \leftarrow t + 1$.
9. **Until** convergence
10. $\mathbf{M} = \sum_{ij} \mu_{ij}^{(t)} \mathbf{X}_{ij}$
11. **Return M**

Similarly to the PCML method presented, the Lagrange dual multipliers β and μ of the NCML method can also be obtained rapidly by using the warm-start strategy. Figure 2.2 shows the duality gap versus the number of iterations in the PenDigits dataset for NCML. Figure 2.2, one can see that the presented NCML algorithm would converge after 10–15 iterations.

2.2.2.3 Optimality Condition

In this subsection, we also use the duality gap to analyze the optimality condition of NCML. From the primal and dual objectives in Eqs. (2.16) and (2.17), the duality gap of the NCML method in the nth iteration is

$$\begin{aligned} \text{DualGap}_{\text{NCML}}^{(n)} = & \frac{1}{2} \sum_{ij} \sum_{kl} \alpha_{ij}^{(n)} \alpha_{kl}^{(n)} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle + C \sum_{ij} \xi_{ij}^{(n)} \\ & + \frac{1}{2} \sum_{ij} \sum_{kl} \left(\beta_{ij}^{(n)} h_{ij} + \eta_{ij}^{(n)} \right) \left(\beta_{kl}^{(n)} h_{kl} + \eta_{kl}^{(n)} \right) \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle - \sum_{ij} \beta_{ij}^{(n)}, \end{aligned} \tag{2.25}$$

where $\alpha_{ij}^{(n)}$ and $\xi_{ij}^{(n)}$ denote the feasible solutions to the primal problem, and $\beta_{ij}^{(n)}$ and $\eta_{ij}^{(n)}$ represent the feasible solutions to the dual problem, as $\eta_{ij}^{(n)}$ and $\mu_{ij}^{(n)}$ are the optimal solutions to the primal subproblem of η in Eq. (2.21) and its dual problem in Eq. (2.23), respectively; thus, the duality gap of subproblem of η is zero

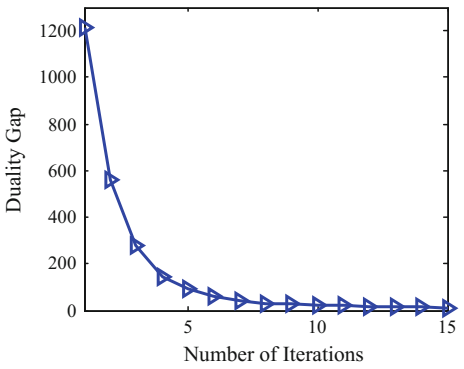

Fig. 2.2 Duality gap versus number of iterations in the PenDigits dataset for NCML (Zuo et al. 2015)

$$\begin{aligned}&\frac{1}{2}\sum_{ij}\sum_{kl}\eta_{ij}^{(n)}\eta_{kl}^{(n)}\left\langle \mathbf{X}_{ij},\mathbf{X}_{kl}\right\rangle+\sum_{ij}\eta_{ij}^{(n)}\gamma_{ij}^{(n)}\\&\quad+\frac{1}{2}\sum_{ij}\sum_{kl}\mu_{ij}^{(n)}\mu_{kl}^{(n)}\left\langle \mathbf{X}_{ij},\mathbf{X}_{kl}\right\rangle-\sum_{ij}\gamma_{ij}^{(n)}\mu_{ij}^{(n)}=0.\end{aligned}\tag{2.26}$$

As shown in Eq. (2.24), $\alpha_{ij}^{(n)}$ and $\mu_{ij}^{(n)}$ should be equal. Substituting Eq. (2.26) into Eq. (2.25) produces the following:

$$\mathrm{DualGap}_{\mathrm{NCML}}^{(n)}=C\sum_{i,j}\xi_{ij}^{(n)}-\sum_{i,j}\beta_{ij}^{(n)}+\sum_{i,j}\mu_{ij}^{(n)}\gamma_{ij}^{(n)}.\tag{2.27}$$

Based on the KKT conditions of the dual problem in Eq. (2.15), we can obtain $\xi_{ij}^{(n)}$ as follows:

$$\begin{aligned}\xi_{ij}^{(n)}&=\begin{cases}0 & \text{for all } \beta_{ij}^{(n)}<C\\ \left[1-h_{ij}\left(\sum_{k,l}\alpha_{kl}^{(n)}\left\langle \mathbf{X}_{ij},\mathbf{X}_{kl}\right\rangle+b^{(n)}\right)\right]_{+} & \text{for all } \beta_{ij}^{(n)}=C,\end{cases}\\&=\begin{cases}0 & \text{for all } \beta_{ij}^{(n)}<C\\ \left[\delta_{ij}^{(n+1)}-h_{ij}\left(\gamma_{ij}^{(n)}+b^{(n)}\right)\right]_{+} & \text{for all } \beta_{ij}^{(n)}=C\end{cases}\end{aligned}\tag{2.28}$$

where [z] = max (z, 0) and

$$b^{(n)}=\frac{1}{h_{ij}}-\sum_{k,l}\alpha_{kl}^{(n)}\left\langle \mathbf{X}_{ij},\mathbf{X}_{kl}\right\rangle=\frac{\delta_{ij}^{(n+1)}}{h_{ij}}-\gamma_{ij}^{(n)}\quad\text{for all } 0<\beta_{ij}^{(n)}<C.\tag{2.29}$$

The detailed derivation of $\xi_{ij}^{(n)}$and $b^{(n)}$ can be found in Appendix 2.2. Figure 2.2 shows the duality gap versus the number of iterations of the presented NCML method in the PenDigits dataset. From Fig. 2.2, we can see that the duality gap would converge to zero in 15 iterations, and the presented NCML method reaches the global optimum. In the implementation of the NCML algorithm, the following termination condition is chosen:

$$\mathrm{DualGap}_{\mathrm{NCML}}^{(t)}-\mathrm{DualGap}_{\mathrm{NCML}}^{(t-1)}<\varepsilon\cdot\mathrm{DualGap}_{\mathrm{NCML}}^{(1)},\tag{2.30}$$

where ε is a small constant. In the experiment, we simply set $\varepsilon = 0.01$.

2.2.2.4 Remarks

Computational Complexity: In Sect. 2.2.1.4, we discussed the pairwise constraint construction scheme for the PCML method. For the NCML method, the same scheme is used to construct the pairwise constraints. In each iteration, NCML uses the SVM solver twice, while the PCML only uses it once. When the SMO-type algorithm (Bellet et al. 2012) is used for SVM training, the computational complexity of the presented NCML algorithm is $O(N^2d)$. This indicates that the computational cost of the NCML algorithm is only with respect to d, which involves two parts: the computation of $\langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle$ and the construction of matrix $\mathbf{M}$. Since $\langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle = \left((\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}} (\mathbf{x}_k - \mathbf{x}_l) \right)^2$, the computation cost of $\langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle$ is $O(d)$. After the convergence of β and μ, matrix $\mathbf{M}$ can be directly obtained; thus, the construction cost of matrix $\mathbf{M}$ is less than $O(kNd^2)$.

Nonlinear extensions: Note that $\langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle = \mathrm{tr}\left(\mathbf{X}_{ij}^{\mathrm{T}} \mathbf{X}_{kl} \right)$ can be treated as an inner product of two pairs of samples: $(\mathbf{x}_i, \mathbf{x}_j)$ and $(\mathbf{x}_k, \mathbf{x}_l)$. If some kernels $K\big((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)\big)$ are performed on $(\mathbf{x}_i, \mathbf{x}_j)$ and $(\mathbf{x}_k, \mathbf{x}_l)$, we can extend the presented method to new linear or even nonlinear metric learning algorithms by using $K\big((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l)\big)$ to replace $\langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle$. Furthermore, the Mahalanobis distance between any two samples $\mathbf{x}_m$ and $\mathbf{x}_n$ can be rewritten as $(\mathbf{x}_m - \mathbf{x}_n)^{\mathrm{T}} \mathbf{M} (\mathbf{x}_m - \mathbf{x}_n) = \sum_{i,j} \alpha_{ij} K\big((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_m, \mathbf{x}_n)\big)$. Another nonlinear extension strategy is to perform a kernel $k(\mathbf{x}_i, \mathbf{x}_j)$ on $\mathbf{x}_i$ and $\mathbf{x}_j$; following this, we can substitute $\big(k(\mathbf{x}_i, \mathbf{x}_k) - k(\mathbf{x}_i, \mathbf{x}_l) - k(\mathbf{x}_j, \mathbf{x}_k) + k(\mathbf{x}_j, \mathbf{x}_l)\big)^2$ for $\langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle$ and rewrite the Mahalanobis distance between $\mathbf{x}_m$ and $\mathbf{x}_n$ as $(\mathbf{x}_m - \mathbf{x}_n)^{\mathrm{T}} \mathbf{M} (\mathbf{x}_m - \mathbf{x}_n) = \sum_{i,j} \alpha_{ij} \big(k(\mathbf{x}_i, \mathbf{x}_m) - k(\mathbf{x}_i, \mathbf{x}_n) - k(\mathbf{x}_j, \mathbf{x}_m) + k(\mathbf{x}_j, \mathbf{x}_n)\big)^2$. This illustrates that the NCML method has the ability to learn nonlinear metrics for histogram and structural data by using proper kernel functions and incorporating appropriate regularizations in α. Metric learning for structural data beyond vector data has received much attention in recent years (Kedem et al. 2012; Bellet et al. 2012), and NCML can provide a new perspective on this topic.

Other SVM Solvers: Although the implementation of the presented algorithm is based on LibSVM, there are some well-studied SVM training algorithms, such as core vector machines (Tsang et al. 2005, 2007), LaRank (Bordes et al. 2007), BMRM (Teo et al. 2007), and Pegasos (Shalev-Shwartz et al. 2011), which can be utilized for large-scale metric learning. Moreover, we can refer to the progress in kernel methods (Evgeniou and Pontil 2004; Belkin et al. 2006; Andrews et al. 2002), and extend the presented scheme to semi-supervised, multiple instance, and multitask metric learning approaches.

2.2.3 Experimental Results

In this subsection, we use four handwritten digit datasets to evaluate the PCML and NCML models for k-NN classification ($k = 1$). The PCML and NCML algorithms are compared to the Euclidean distance metric and the state-of-the-art metric learning models, including NCA (Goldberger et al. 2004), ITML (Davis et al. 2007), MCML (Globerson and Roweis 2005), LDML (Guillaumin et al. 2009), LMNN (Weinberger et al. 2009), PLML (Wang et al. 2012), and DML-eig (Ying and Li 2012). PCML and NCML are implemented using the LibSVM[1] toolbox. The source codes for NCA[2], ITML[3], MCML[4], LDML[5], LMNN[6], PLML[7], and DML-eig[8] are available online, and we tune their parameters to get the best results.

Table 2.1 lists the basic information of the four handwritten digit datasets. We use the defined training sets to train the metrics, and classify the defined test sets to get the classification error rates. For the Semeion dataset, we use 10-fold cross-validation to evaluate the metric learning methods, and the classification error rate and training time are obtained by averaging over 10 runs of 10-fold cross-validation.

For MNIST, Semeion and USPS datasets, we first reduce the feature dimension to 100 by using the principal component analysis (PCA) method, and then learn the distance metrics in the PCA subspace. Table 2.2 shows the classification error rates of the ten competing methods on the four handwritten digit datasets. The last row in Table 2.2 lists the average ranks of the competing methods. In the experiment, we do not list the error rate and training time of MCML in the MNIST dataset, because MCML requires too large a memory space (more than 30 GB) for this dataset and cannot run on our PC. From Table 2.2, one can see that both PCML and NCML achieve the best performances.

Figure 2.3 shows the training time of the above distance metric learning algorithms. We can see that the PCML and NCML methods are much faster than the other models.

Finally, we compare the training time of PCML and NCML on different feature dimensions. The computation complexities of PCML and NCML are $O(N^2d + d^3)$ and $O(N^2d)$, respectively. Figure 2.4 shows the training time versus the PCA dimension in the PenDigits dataset. From Fig. 2.4, one can see that when the PCA dimension is lower than 110, the training time of NCML is longer than that of

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

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

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

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

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

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

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

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

Table 2.1 The handwritten digit datasets used in the experiments (Zuo et al. 2015)

Dataset	# of training samples	# of test samples	Dimension	PCA dimension	# of classes
MNIST	60,000	10,000	784	100	10
PenDigits	7494	3498	16	N/A	10
Semeion	1434	159	256	100	10
USPS	7291	2007	256	100	10

PCML. When the PCA dimension is higher than 110, the training time of PCML increases and becomes longer than that of NCML.

2.3 A Kernel Classification Framework for Metric Learning

Most metric learning models depend on convex or non-convex optimization techniques. However, these algorithms are inefficient for learning the distance metrics for large-scale problems. In this section, we present a kernel classification framework that can unify many state-of-the-art metric learning methods, and that can improve the efficiency greatly. The connections between the kernel framework and LMNN, ITML, and LDML will also be discussed in this section.

2.3.1 Doublets and Triplets

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

Suppose $D = \{(\mathbf{x}_i, y_i)|i = 1, 2, \ldots, n\}$ denotes a training dataset, $\mathbf{x}_i \in \mathbb{R}^d$ is the vector of ith training sample, and y_i is the class label of $\mathbf{x}_i$. For any two samples, we define a doublet ($\mathbf{x}_i$, $\mathbf{x}_j$), and assign a class 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 its m_1 most similar samples from D, denoted by $\{\mathbf{x}^s_{i,1}, \ldots, \mathbf{x}^s_{i,m_1}\}$, and its m_2 nearest dissimilar neighbors, denoted by $\{\mathbf{x}^d_{i,1}, \ldots, \mathbf{x}^d_{i,m_2}\}$; we then construct $(m_1 + m_2)$ doublets $\{(\mathbf{x}_i, \mathbf{x}^s_{i,1}), \ldots, (\mathbf{x}_i, \mathbf{x}^s_{i,m_1}), (\mathbf{x}_i, \mathbf{x}^d_{i,1}), \ldots, (\mathbf{x}_i, \mathbf{x}^d_{i,m_2})\}$. We define a doublet set $\{\mathbf{z}_1, \cdots, \mathbf{z}_{N_d}\}$ as denoting all such doublets constructed from the training dataset, where $\mathbf{z}_l = (\mathbf{x}_{l,1}, \mathbf{x}_{l,2})$, and l = 1,2,…,N_d. h_l represents the class label of each

Table 2.2 Comparison of the classification error rate on the handwritten digit datasets (%) (Zuo et al. 2015)

c	Euclidean	NCA	ITML	MCML	LDML	LMNN	DML-eig	PLML	PCML	NCML
MNIST	2.87	5.46	2.89	N/A	6.05	2.28	5.06	2.54	3.85	2.80
PenDigits	2.26	2.23	2.29	2.26	6.20	2.52	3.75	2.46	2.06	2.06
Semeion	8.54	8.60	5.71	11.23	11.98	6.09	5.72	7.66	4.83	5.53
USPS	5.08	5.68	6.33	5.08	8.77	5.38	11.36	6.73	5.33	5.43
Average rank	4.00	6.25	5.25	4.67	9.50	4.50	7.50	5.75	2.75	2.75

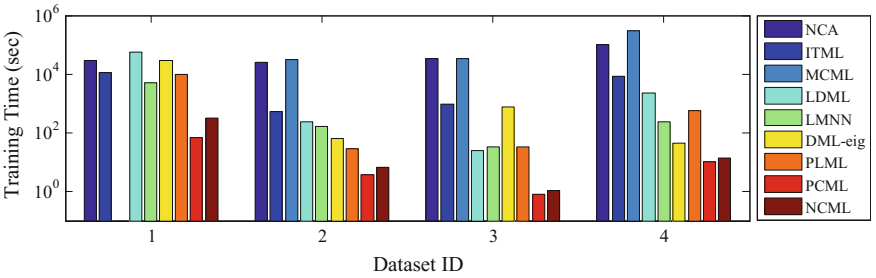

Fig. 2.3 Training time (s) of NCA, ITML, MCML, LDML, LMNN, DML-eig, PLML, PCML and NCML. From 1–4, the Dataset ID represents MNIST, PenDigits, Semeion, and USPS (Zuo et al. 2015)

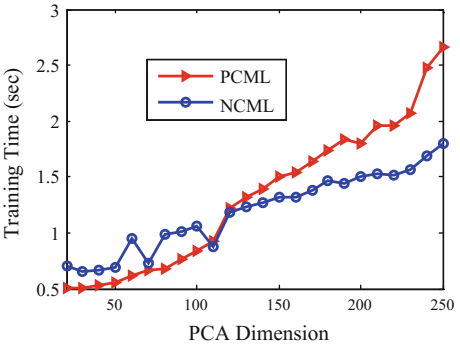

Fig. 2.4 Training time (s) versus PCA dimension in the PenDigits dataset (Zuo et al. 2015)

doublet z_l. Note that doublet-based constraints are used in ITML (Davis et al. 2007) and LDML (Guillaumin et al. 2009), but the details of the construction of the doublets are not introduced.

For any three samples $\mathbf{x}_i$, $\mathbf{x}_j$, and $\mathbf{x}_k$ from D, we define a triplet ($\mathbf{x}_i$, $\mathbf{x}_j$, $\mathbf{x}_k$), 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 m_1 nearest neighbors $\{\mathbf{x}^s_{i,1}, \ldots, \mathbf{x}^s_{i,m_1}\}$ that have the same class label as $\mathbf{x}_i$, and m_2 nearest neighbors $\{\mathbf{x}^d_{i,1}, \ldots, \mathbf{x}^d_{i,m_2}\}$ that have different class labels from $\mathbf{x}_i$. Thus, for each sample $\mathbf{x}_i$, we can construct $m_1 m_2$ triplets $\{(\mathbf{x}_i, \mathbf{x}^s_{i,j}, \mathbf{x}^d_{i,k}) | j = 1, \ldots, m_1;\ k = 1, \ldots, m_2\}$. Combining all the triplets, we form a triplet set $\{\mathbf{t}_1, \ldots, \mathbf{t}_{N_t}\}$, where $\mathbf{t}_l = (\mathbf{x}_{l,1}, \mathbf{x}_{l,2}, \mathbf{x}_{l,3})$, l = 1,2, …,N_t. Note that, for the convenience of expression, we have removed the superscript "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 (Weinberger et al. 2009), based on the k-nearest neighbors of each sample.

2.3.2 A Family of Degree-2 Polynomial Kernels

In this subsection, we introduce a family of degree-2 polynomial kernel functions that can operate on pairwise constraints of the doublets or triplets. With the degree-2 polynomial kernels, distance metric learning can be readily formulated as a kernel classification problem.

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

$$K_p(\mathbf{x}_i, \mathbf{x}_j) = \mathrm{tr}(\mathbf{x}_i \mathbf{x}_i^{\mathrm{T}} \mathbf{x}_j \mathbf{x}_j^{\mathrm{T}}), \tag{2.31}$$

where $\mathrm{tr}(\cdot)$ is the trace operator of a matrix. $K_p(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^{\mathrm{T}} \mathbf{x}_j)^2$ is a degree-2 polynomial kernel, and $K_p(\mathbf{x}_i, \mathbf{x}_j)$ satisfies Mercer's condition (Shawe-Taylor and Cristianini 2004).

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

$$\begin{aligned} K_p(\mathbf{z}_i, \mathbf{z}_j) &= \mathrm{tr}\Big((\mathbf{x}_{i,1} - \mathbf{x}_{i,2})(\mathbf{x}_{i,1} - \mathbf{x}_{i,2})^{\mathrm{T}} (\mathbf{x}_{j,1} - \mathbf{x}_{j,2})(\mathbf{x}_{j,1} - \mathbf{x}_{j,2})^{\mathrm{T}} \Big) \\ &= \Big[(\mathbf{x}_{i,1} - \mathbf{x}_{i,2})^{\mathrm{T}} (\mathbf{x}_{j,1} - \mathbf{x}_{j,2}) \Big]^2. \end{aligned} \tag{2.32}$$

The kernel function in Eq. (2.32) defines an inner product of two doublets. Based on the degree-2 polynomial kernel function, we can learn a decision function to obtain the class label of the doublet, and can also identity whether the two samples in the doublet are of the same class or not.

Similarly to doublets, for 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 function as

$$K_p(\mathbf{t}_i, \mathbf{t}_j) = \mathrm{tr}\Big(\mathbf{T}_i \mathbf{T}_j \Big), \tag{2.33}$$

where

$$\begin{aligned} \mathbf{T}_i &= \big(\mathbf{x}_{i,1} - \mathbf{x}_{i,3}\big)\big(\mathbf{x}_{i,1} - \mathbf{x}_{i,3}\big)^{\mathrm{T}} - \big(\mathbf{x}_{i,1} - \mathbf{x}_{i,2}\big)\big(\mathbf{x}_{i,1} - \mathbf{x}_{i,2}\big)^{\mathrm{T}}, \\ \mathbf{T}_j &= \big(\mathbf{x}_{j,1} - \mathbf{x}_{j,3}\big)\big(\mathbf{x}_{j,1} - \mathbf{x}_{j,3}\big)^{\mathrm{T}} - \big(\mathbf{x}_{j,1} - \mathbf{x}_{j,2}\big)\big(\mathbf{x}_{j,1} - \mathbf{x}_{j,2}\big)^{\mathrm{T}}. \end{aligned}$$

The triplet kernel function in Eq. (2.33) defines an inner product of two triplets. With this triplet kernel function, a decision function to identify the class label of triplets can be learned, and the class label of the query sample can then be identified. In Sect. 2.3.3, we will discuss the connection between metric learning and kernel decision function learning.

2.3.3 Metric Learning Via Kernel Methods

With the degree-2 polynomial kernels presented in Sect. 2.3.2, the doublet-based and triplet-based distance metric learning can easily be extended to the kernel space. More specifically, any kernel classification method can be utilized to learn a kernel classifier. For doublet-based and triplet-based distance metric learning, the corresponding class label can be obtained by the following two forms:

$$g_d(\mathbf{z}) = \operatorname{sgn}\left(\sum_l h_l \alpha_l K_p(\mathbf{z}_l, \mathbf{z}) + b\right), \tag{2.34}$$

$$g_t(\mathbf{t}) = \operatorname{sgn}\left(\sum_l \alpha_l K_p(\mathbf{t}_l, \mathbf{t})\right), \tag{2.35}$$

where $\mathbf{z}_l$ and $\mathbf{t}_l$, l = 1,2, …, N, denote the doublet and triplet constructed from the training dataset. $\mathbf{z} = (\mathbf{x}_{(1)}, \mathbf{x}_{(2)})$ is the test doublet, $\boldsymbol{t}$ represents the test triplet, α_l is the weight, and b is the bias. For the doublet-based method, kernel decision function $g_d(\mathbf{z})$ can be used to determine whether two samples $\mathbf{x}_{(1)}$ and $\mathbf{x}_{(2)}$ are from the same class.

For the doublet, we have

$$\begin{aligned}&\sum_l h_l \alpha_l \operatorname{tr}\left((\mathbf{x}_{l,1} - \mathbf{x}_{l,2})(\mathbf{x}_{l,1} - \mathbf{x}_{l,2})^{\mathrm{T}}(\mathbf{x}_{(i)} - \mathbf{x}_{(j)})(\mathbf{x}_{(i)} - \mathbf{x}_{(j)})^{\mathrm{T}}\right) + b \\ &= (\mathbf{x}_{(i)} - \mathbf{x}_{(j)})^{\mathrm{T}}\mathbf{M}(\mathbf{x}_{(i)} - \mathbf{x}_{(j)}) + b,\end{aligned} \tag{2.36}$$

where

$$\mathbf{M} = \sum_l h_l \alpha_l (\mathbf{x}_{l,1} - \mathbf{x}_{l,2})(\mathbf{x}_{l,1} - \mathbf{x}_{l,2})^{\mathrm{T}}, \tag{2.37}$$

where matrix $\mathbf{M}$ is the Mahalanobis distance metric.

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

Theorem 2.2 *For the triplet-based decision function defined in* Eq. (2.35), *matrix* $\mathbf{M}$ *of the Mahalanobis distance metric is*

$$\mathbf{M} = \sum_l \alpha_l \mathbf{T}_l = \sum_l \alpha_l \left[(\mathbf{x}_{l,1} - \mathbf{x}_{l,3})(\mathbf{x}_{l,1} - \mathbf{x}_{l,3})^{\mathrm{T}} - (\mathbf{x}_{l,1} - \mathbf{x}_{l,2})(\mathbf{x}_{l,1} - \mathbf{x}_{l,2})^{\mathrm{T}}\right], \tag{2.38}$$

and $\sum_l \alpha_l K_p(\mathbf{t}_l, \mathbf{t})$ *then represents the relative difference of the Mahalanobis distance between* $\mathbf{x}_{(i)}$ *and* $\mathbf{x}_{(k)}$ *and the Mahalanobis distance between* x(i) *and* x(j).

Proof Let $\mathbf{T}_l = (\mathbf{x}_{l,1} - \mathbf{x}_{l,3})(\mathbf{x}_{l,1} - \mathbf{x}_{l,3})^{\mathrm{T}} - (\mathbf{x}_{l,1} - \mathbf{x}_{l,2})(\mathbf{x}_{l,1} - \mathbf{x}_{l,2})^{\mathrm{T}}$. Based on the definition of $K_p(\mathbf{t}_l, \mathbf{t})$, we have

$$\begin{aligned}
&\sum_l \alpha_l K_p(\mathbf{t}_l, \mathbf{t}) = \sum_l \alpha_l \mathrm{tr}(\mathbf{T}_l \mathbf{T}) \\
&= \sum_l \alpha_l tr\left(\mathbf{T}_l \left((\mathbf{x}_{(i)} - \mathbf{x}_{(k)})(\mathbf{x}_{(i)} - \mathbf{x}_{(k)})^T - (\mathbf{x}_{(i)} - \mathbf{x}_{(j)})(\mathbf{x}_{(i)} - \mathbf{x}_{(j)})^T\right)^T\right) \\
&= \sum_l \alpha_l tr\left(\mathbf{T}_l \left((\mathbf{x}_{(i)} - \mathbf{x}_{(k)})(\mathbf{x}_{(i)} - \mathbf{x}_{(k)})^T\right)^T\right) - \sum_l \alpha_l tr\left(\mathbf{T}_l \left((\mathbf{x}_{(i)} - \mathbf{x}_{(j)})(\mathbf{x}_{(i)} - \mathbf{x}_{(j)})^T\right)^T\right) \\
&= (\mathbf{x}_{(i)} - \mathbf{x}_{(k)})^T \sum_l \alpha_l \mathbf{T}_l (\mathbf{x}_{(i)} - \mathbf{x}_{(k)}) - (\mathbf{x}_{(i)} - \mathbf{x}_{(j)})^T \sum_l \alpha_l \mathbf{T}_l (\mathbf{x}_{(i)} - \mathbf{x}_{(j)}) \\
&= (\mathbf{x}_{(i)} - \mathbf{x}_{(k)})^T \mathbf{M} (\mathbf{x}_{(i)} - \mathbf{x}_{(k)}) - (\mathbf{x}_{(i)} - \mathbf{x}_{(j)})^T \mathbf{M} (\mathbf{x}_{(i)} - \mathbf{x}_{(j)})
\end{aligned} \tag{2.39}$$

By setting $\mathbf{M} = \sum_l \alpha_l \mathbf{T}_l$ as matrix $\mathbf{M}$ in the Mahalanobis distance metric, we can see that $\sum_l \alpha_l K_p(\mathbf{t}_l, \mathbf{t})$ is the difference of the Mahalanobis distance between $\mathbf{x}_{(i)}$ and $\mathbf{x}_{(k)}$ and the Mahalanobis distance between $\mathbf{x}_{(i)}$ and $\mathbf{x}_{(j)}$.

Clearly, Eqs. (2.34)–(2.39) provide us with a new perspective for understanding the distance metric matrix $\mathbf{M}$ under a kernel classification framework. Meanwhile, using this kernel framework to learn a distance metric may be much easier and more efficient than the previous metric learning methods. In the following, we introduce two kernel classification methods for metric learning: regularized kernel SVM and kernel logistic regression. It should be pointed out that, based on modifying the construction of the doublet or triplet set, this kernel framework can be extended to other, new metric learning algorithms by using different kernel classifier models, or by adopting different optimization algorithms.

2.3.3.1 Kernel SVM-Like Model

For the doublet- or triplet-based distance metric learning, a SVM-like model is defined as

$$\begin{aligned}
&\min_{\mathbf{M},b,\xi} \; r(\mathbf{M}) + \rho(\xi) \\
&\text{s.t.} \quad f_l^{(d)}\left((\mathbf{x}_{l,1} - \mathbf{x}_{l,2})^{\mathrm{T}} \mathbf{M} (\mathbf{x}_{l,1} - \mathbf{x}_{l,2}), b, \xi_l\right) \geq 0 \,(\text{doublet set}) \\
&\quad f_l^{(t)}\left((\mathbf{x}_{l,1} - \mathbf{x}_{l,3})^{\mathrm{T}} \mathbf{M} (\mathbf{x}_{l,1} - \mathbf{x}_{l,3}) - (\mathbf{x}_{l,1} - \mathbf{x}_{l,2})^{\mathrm{T}} \mathbf{M} (\mathbf{x}_{l,1} - \mathbf{x}_{l,2}), \xi_l\right) \geq 0 \,(\text{triplet set}) \\
&\qquad \xi_l \geq 0,
\end{aligned} \tag{2.40}$$

where $r(\mathbf{M})$ is the regularization term, $\rho(\xi)$ denotes the margin loss term, constraint $f_l^{(d)}$ is a linear function of $(\mathbf{x}_{l,1} - \mathbf{x}_{l,2})^{\mathrm{T}} \mathbf{M} (\mathbf{x}_{l,1} - \mathbf{x}_{l,2})$, b, and ξ_l, and constraint $f_l^{(t)}$ is a

linear function of $(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})^{\mathrm{T}}\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})-(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^{\mathrm{T}}\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})$ and ξ_l. In the implementation, we can simply choose a convex regularizer $r(\mathbf{M})$ and a convex margin loss $\rho(\xi)$ to guarantee that Eq. (2.40) is convex. By plugging Eq. (2.37) or (2.38) into Eq. (2.40), matrix $\mathbf{M}$ can be calculated simply by the SVM and kernel methods.

If we adopt the l_2-norm to regularize $\mathbf{M}$ and the hinge loss penalty on ξ_l, the model in Eq. (2.40) would become the standard SVM. SVM and its variants have been well studied (Schölkopf et al. 2001; Müller et al. 2001; Vapnik 2013), and various SVM-based algorithms have been proposed for large-scale problems (Tsang et al. 2005; Collobert et al. 2002). Thus, the SVM-like modeling in Eq. (2.40) enables us to learn good metrics from large-scale training data efficiently.

2.3.3.2 Kernel Logistic Regression

For the kernel logistic regression model (KLR) (Keerthi et al. 2005), we let class label $h_l = 1$ if the samples of doublet z_l belong to the same class and let $h_l = 0$ if two samples belong to different classes. For a query doublet z_l, the probability of $h_l = 1$ is defined as

$$P(p_l = 1|\mathbf{z}_l) = \frac{1}{1+\exp\left((\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^{\mathrm{T}}\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})+b\right)}. \tag{2.41}$$

Matrix $\mathbf{M}$ and bias b can be obtained by solving the following log-likelihood function:

$$(\mathbf{M},b) = \arg\max_{\mathbf{M},b}\left\{\mathrm{l}(\mathbf{M},b) = \sum_l h_l \ln P(p_l = 1|\mathbf{z}_l) + (1-h_l)\ln P(p_l = 0|\mathbf{z}_l)\right\}. \tag{2.42}$$

KLR is a powerful probabilistic approach to classification. By modeling metric learning as a KLR problem, we can easily use the existing KLR algorithms to learn the desired distance metric. Moreover, the various improved KLR, such as sparse KLR (Koh et al. 2007), can also be used to develop new metric learning methods.

2.3.4 Connections with LMNN, ITML, and LDML

The kernel classification framework can be viewed as a unified model of many state-of-the-art metric learning methods. In this subsection, we show the connections of the presented framework and some typical distance metric learning methods, such as LMNN, ITML and LDML.

2.3.4.1 LMNN

LMNN (Weinberger et al. 2009) 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 using a set of triplets $\{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)\}$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ have the same class labels, and $\mathbf{x}_i$ and $\mathbf{x}_k$ have different class labels. The objective function of LMNN is defined as follows:

$$\begin{aligned} \min_{\mathbf{M},\xi_{ijk}} & \sum_{i,j} (\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}} \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j) + C \sum_{i,j,k} \xi_{ijk} \\ \text{s.t.} & (\mathbf{x}_i - \mathbf{x}_k)^{\mathrm{T}} \mathbf{M} (\mathbf{x}_i - \mathbf{x}_k) - (\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}} \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j) \geq 1 - \xi_{ijk} \\ & \xi_{ijl} \geq 0, \quad \mathbf{M} \succcurlyeq 0. \end{aligned} \tag{2.43}$$

For LMNN, matrix $\mathbf{M}$ is required to be positive and semidefinite; thus, we introduce the following indicator function:

$$\iota_{\succcurlyeq}(\mathbf{M}) = \begin{cases} 0, & \text{if} \quad \mathbf{M} \succcurlyeq 0 \\ +\infty, & \text{otherwise} \end{cases}, \tag{2.44}$$

and choose the following regularizer and margin loss:

$$r_{\mathrm{LMNN}}(\mathbf{M}) = \sum_{i,j} (\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}} \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j) + \iota_{\succcurlyeq}(\mathbf{M}), \tag{2.45}$$

$$\rho_{\mathrm{LMNN}}(\xi) = C \sum_{ijk} \xi_{ijk}. \tag{2.46}$$

We can then define the following SVM-like model for the same triplet set:

$$\begin{aligned} \min_{\mathbf{M},\xi} \; & r_{\mathrm{LMNN}}(\mathbf{M}) + \rho_{\mathrm{LMNN}}(\xi) \\ \text{s.t.} \quad & (\mathbf{x}_i - \mathbf{x}_k)^{\mathrm{T}} \mathbf{M} (\mathbf{x}_i - \mathbf{x}_k) - (\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}} \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j) \geq 1 - \xi_{ijk}, \quad \xi_{ijk} \geq 0. \end{aligned} \tag{2.47}$$

From Eqs. (2.41) and (2.47), one can observe that the SVM-like model is equivalent to the LMNN model.

2.3.4.2 ITML

ITML (Davis et al. 2007) operates on a set of doublets $\{(\mathbf{x}_i, \mathbf{x}_j)\}$, and its objective model is as follows:

$$\begin{aligned}\min_{\mathbf{M},\xi}\ & D_{ld}(\mathbf{M},\mathbf{M}_0)+\gamma\cdot D_{ld}(\mathrm{diag}(\xi),\mathrm{diag}(\xi_0))\\ \mathrm{s.t.}\ & (\mathbf{x}_i-\mathbf{x}_j)^{\mathrm{T}}\mathbf{M}(\mathbf{x}_i-\mathbf{x}_j)\le\xi_{u(i,j)}\quad (i,j)\in\mathcal{S}\\ & (\mathbf{x}_i-\mathbf{x}_j)^{T}\mathbf{M}(\mathbf{x}_i-\mathbf{x}_j)\ge\xi_{l(i,j)}\quad (i,j)\in\mathcal{D}\\ & \mathbf{M}\succcurlyeq 0,\end{aligned}\tag{2.48}$$

where $\mathbf{M}_0$ is the given prior of the metric matrix, ξ_0 is the given prior to ξ, $\mathcal{S}$ denotes the doublet set in which two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ have same class label, $\mathcal{D}$ represents the doublet set in which two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ have different class labels, and $D_{ld}(\cdot,\cdot)$ is the LogDet divergence of two matrices defined as

$$D_{ld}(\mathbf{M},\mathbf{M}_0)=\mathrm{tr}\left(\mathbf{M}\mathbf{M}_0^{-1}\right)-\log\det\left(\mathbf{M}\mathbf{M}_0^{-1}\right)-n.\tag{2.49}$$

By introducing the following regularizer and margin loss,

$$r_{\mathrm{ITML}}(\mathbf{M})=D_{ld}(\mathbf{M},\mathbf{M}_0)+\iota_{\succcurlyeq}(\mathbf{M}),\tag{2.50}$$

$$\rho_{\mathrm{ITML}}(\xi)=\gamma\cdot D_{ld}(\mathrm{diag}(\xi),\mathrm{diag}(\xi_0)),\tag{2.51}$$

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

$$\begin{aligned}\min_{\mathbf{M},b,\xi}\ & r_{\mathrm{ITML}}(\mathbf{M})+\rho_{\mathrm{ITML}}(\xi)\\ \mathrm{s.t.}\ & (\mathbf{x}_i-\mathbf{x}_j)^{\mathrm{T}}\mathbf{M}(\mathbf{x}_i-\mathbf{x}_j)\le\xi_{u(i,j)}\quad (i,j)\in\mathcal{S}\\ & (\mathbf{x}_i-\mathbf{x}_j)^{\mathrm{T}}\mathbf{M}(\mathbf{x}_i-\mathbf{x}_j)\ge\xi_{l(i,j)}\quad (i,j)\in\mathcal{D}\\ & \xi_{ij}\ge 0,\end{aligned}\tag{2.52}$$

where $\mathbf{z}_{ij}=\left(\mathbf{x}_i,\mathbf{x}_j\right)$. From Eqs. (2.52) and (2.48), it is obvious that the SVM-like model is equivalent to the ITML model.

2.3.4.3 LDML

LDML (Guillaumin et al. 2009) is a logistic discriminant-based metric learning approach, which learns the metric from a set of doublets. Suppose $z_l = (\mathbf{x}_l^{(i)}, \mathbf{x}_l^{(j)})$ and hl are a doublet and its class label, respectively. $y_{l(i)}$ and $y_{l(j)}$ are class labels of two samples $\mathbf{x}_l^{(i)}$, $\mathbf{x}_l^{(j)}$, respectively. For LDML, the probability of $y_l^{(i)} = y_l^{(j)}$ is defined as follows:

$$p_l=P(y_{l(i)}=y_{l(j)}|\mathbf{x}_{l(i)},\mathbf{x}_{l(j)},\mathbf{M},b)=\sigma(b-d_{\mathbf{M}}(\mathbf{x}_{l(i)},\mathbf{x}_{l(j)})),\tag{2.53}$$

where $\sigma(z)$ is the sigmoid function, b is the bias, and $d_{\mathbf{M}}(\mathbf{x}_{l(i)},\mathbf{x}_{l(j)})=(\mathbf{x}_{l(i)}-\mathbf{x}_{l(j)})^{\mathrm{T}}\mathbf{M}(\mathbf{x}_{l(i)}-\mathbf{x}_{l(j)})$. With probability p_l defined in

Eq. (2.53), LDML learns metric matrix $\mathbf{M}$ and bias b by solving the following log-likelihood:

$$\max_{\mathbf{M},b}\left\{l(\mathbf{M},b)=\sum_l h_l \ln p_l+(1-h_l)\ln(1-p_l)\right\}. \tag{2.54}$$

Unlike LMNN, metric matrix $\mathbf{M}$ is not required to be positive definite in LDML.

With the same doublet set, suppose α is the solution obtained by the kernel logistic model in Eq. (2.42), and metric matrix $\mathbf{M}$ is the solution to LDML in Eq. (2.54). It is obvious that

$$\mathbf{M}=-\sum_l \alpha_l(\mathbf{x}_{l(i)}-\mathbf{x}_{l_1(j)})(\mathbf{x}_{l(i)}-\mathbf{x}_{l(j)})^{\mathrm{T}}. \tag{2.55}$$

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

2.3.5 Metric Learning Via SVM

In Sect. 2.3.4, we proved that the kernel classification framework is a generalized model for existing metric learning models. In this section, we will present two metric learning models, namely, the doublet-SVM and the triplet-SVM, developed according to the kernel framework.

2.3.5.1 Doublet-SVM

In the doublet-SVM, we adopt the l_2-norm regularizer $r_{\mathrm{SVM}}(\mathbf{M})=\frac{1}{2}\|\mathbf{M}\|_F^2$, and the margin loss term $\rho_{\mathrm{SVM}}(\xi)=C\sum_l \xi_l$. The model for the doublet-SVM is defined as follows:

$$\begin{aligned}&\min_{\mathbf{M},b,\xi}\ \frac{1}{2}\|\mathbf{M}\|_F^2+C\sum_l \xi_l\\ &\text{s.t.}\quad h_l\Big((\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^{\mathrm{T}}\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})+b\Big)\geq 1-\xi_l,\quad \xi_l\geq 0,\ \ \forall l,\end{aligned} \tag{2.56}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. The Lagrange dual problem of the above doublet-SVM model is:

$$\begin{aligned} &\max_{\alpha} -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j h_i h_j K_p(\mathbf{z}_i,\mathbf{z}_j)+\sum_i \alpha_i \\ &\text{s.t.} \quad 0\le\alpha_l\le C, \quad \sum_l \alpha_l h_l = 0, \quad \forall l. \end{aligned} \tag{2.57}$$

The above Lagrange dual problem can also be easily solved by standard SVM solvers, such as LIBSVM (Chang and Lin 2011). Please refer to Appendix 2.4 for the detailed deduction of the dual problem of the doublet-SVM.

2.3.5.2 Triplet-SVM

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

$$\begin{aligned} &\min_{\mathbf{M},\xi} \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum_l \xi_l \\ &\text{s.t.} \quad (\mathbf{x}_{l,1}-\mathbf{x}_{l,3})^{\mathrm{T}}\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,3}) - (\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^{\mathrm{T}}\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2}) \ge 1-\xi_l \\ &\qquad \xi_l \ge 0, \quad \forall l. \end{aligned} \tag{2.58}$$

In fact, the triplet-SVM can be regarded as a one-class SVM model, and the formulation of the triplet-SVM is similar to the one-class SVM in (Schölkopf et al. 2001). The dual problem of the triplet-SVM is

$$\begin{aligned} &\max_{\alpha} -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j K_p(\mathbf{t}_i,\mathbf{t}_j)+\sum_i \alpha_i \\ &\text{s.t.} \quad 0\le\alpha_l\le C, \quad \forall l, \end{aligned} \tag{2.59}$$

which can also be solved efficiently by the standard SVM solvers (Chang and Lin 2011). Please refer to Appendix 2.5 for the detailed deduction of the dual problem of the triplet-SVM.

2.3.6 *Experimental Results*

In this subsection, the doublet-SVM, the triplet-SVM and some state-of-the-art metric learning algorithms for k-NN classification are compared on 10 UCI datasets. Five representative and state-of-the-art metric learning models are selected, namely LMNN (Weinberger et al. 2009), ITML (Davis et al. 2007), LDML

Table 2.3 The UCI datasets used in the experiment

Dataset	# of training samples	# of test samples	Feature dimension	# of classes
Parkinsons	176	19	22	2
Sonar	188	20	60	2
Statlog segmentation	2079	231	19	7
Breast tissue	96	10	9	6
ILPD	525	58	10	2
Statlog satellite	4435	2000	36	6
Blood transfusion	674	74	4	2
SPECTF Heart	80	187	44	2
Cardiotocography	1914	212	21	10
Letter	16,000	4000	16	26

(Guillaumin et al. 2009), neighborhood component analysis (NCA) (Goldberger et al. 2004) and maximally collapsing metric learning (MCML) (Globerson and Roweis 2005). We implemented the doublet-SVM and the triplet-SVM based on the popular SVM toolbox LINSVM[9]. The source codes for LMNN[10], ITML[11], LDML[12], NCA[13], and MCML[14] are also available online, and we tuned their parameters to get the best results.

Table 2.3 shows the basic information for the 10 UCI datasets (Frank and Asuncion 2010). For the Statlog Satellite, SPECTF Heart, and Letter datasets, we use the defined training and test sets to perform the experiment. For the other seven datasets, 10-fold cross-validation were chosen to evaluate the competing metric learning methods, and the reported error rate and training time were obtained by averaging over 10 runs.

Both the doublet-SVM and the triplet-SVM have three hyperparameters, namely, m_1, m_2, and C. Using the Statlog Segmentation dataset as an example, we analyzed the connections between the classification error rate and the hyperparameters. We first analyzed the influence of m_2 on the classification performance of the two SVM methods. Figure 2.5 shows the classification error rate versus m_2 for the doublet-SVM and the triplet-SVM when $m_1 = 1$ and $C = 1$. From Fig. 2.5, we can see that both SVM methods obtained the lowest classification error rates when

[9] http://www.csie.ntu.edu.tw/∼cjlin/libsvm/

[10] http://www.cse.wustl.edu/∼kilian/code/code.html

[11] http://www.cs.utexas.edu/∼pjain/itml/

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

[13] http://www.cs.berkeley.edu/∼fowlkes/software/nca/

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

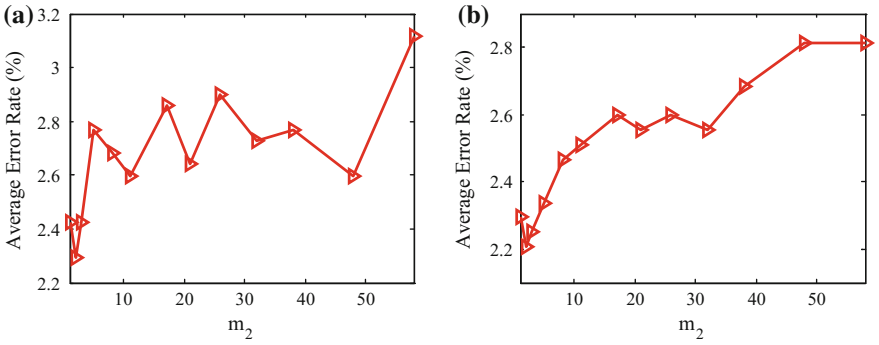

Fig. 2.5 Classification error rate (%) versus m_2 for **a** doublet-SVM and **b** triplet-SVM with $m_1 = 1$ and $C = 1$. © 2015 IEEE. Reprinted with permission, from Wang et al. (2013)

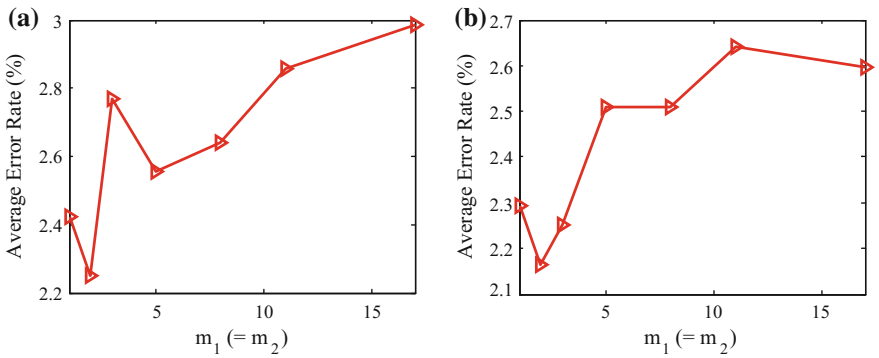

Fig. 2.6 Classification error rate (%) versus m_1 (=m_2) for **a** doublet-SVM and **b** triplet-SVM with $C = 1$. © 2015 IEEE. Reprinted with permission, from Wang et al. (2013)

$m_2 = 2$. Moreover, the error rates for both SVM methods tended to be a little higher when $m_2 > 3$. Thus, we set m_2 to $1 \sim 3$ in our experiments.

By setting $m_1 = m_2$, we investigated the influence of m_1 on the classification error rate of two SVM methods. Figure 2.6 shows the classification error rate versus m_1(=m_2) for the two SVM methods, respectively. From Fig. 2.6, it is obvious that both SVM methods achieved the lowest classification error when $m_1 = m_2 = 2$. Thus, m_1 is set to $1 \sim 3$ in our experiments.

By setting $m_1 = m_2 = 2$, we further studied the relation between C and the classification error rate of the two SVM methods. Figure 2.7 shows the classification error rate versus C for the two SVM methods above. From Fig. 2.7, we can see that the error rate is insensitive to C in a wide range, but it jumps when C is no less than 10^4 for the doublet-SVM and no less than 101 for the triplet-SVM. Thus, we set $C < 10^4$ for the doublet-SVM and $C < 10^1$ for the triplet-SVM in our experiments.

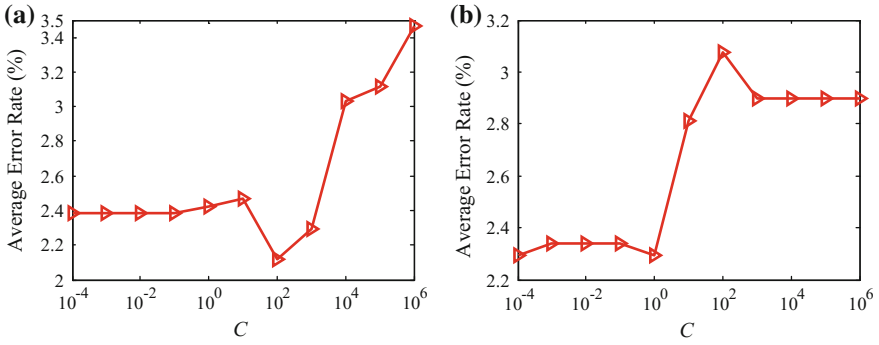

Fig. 2.7 Classification error rate (%) versus C for **a** doublet-SVM and **b** triplet-SVM with $m_1 = m_2 = 2$. © 2015 IEEE. Reprinted with permission, from Wang et al. (2013)

Table 2.4 shows the classification error rates of seven metric learning models on 10 UCI datasets. From Table 2.4, it obvious that the doublet-SVM method achieves the lowest error rates on the Letter, ILPD, and SPECTF Heart datasets. On the Statlog Segmentation dataset, the triplet-SVM achieves the best performance.

We further adopted the average ranks of these models to compare the recognition performances of different distance metric learning models. For each dataset, the rank is calculated based on classification error rates of the compared metric learning methods; thus, rank 1 denotes the best method and rank 2 is the second best method, and so on. The average rank is defined as the mean rank of one method over the 10 datasets, which can provide a fair comparison of the algorithms (Demšar 2006). The rank of different metric learning methods is shown in the last row in Table 2.4.

From Table 2.4, we can see that the doublet-SVM achieves the best average rank and the triplet-SVM achieves the fifth best average rank. This proves that, by incorporating the degree-2 polynomial kernel into the standard (one-class) kernel SVM classifier, the kernel classification based metric learning framework can lead to highly competitive classification accuracy with state-of-the-art metric learning methods. It is interesting to see that, although the doublet-SVM achieves better performance than the triplet-SVM for most datasets, the triplet-SVM performs better than the doublet-SVM for large datasets, such as Statlog Segmentation, Statlog Satellite and Cardiotocography, and achieves a very close error rate to that of the doublet-SVM for the large dataset Letter. The experiments indicate that the doublet-SVM is more suitable for small-scale datasets, while the triplet-SVM is more suitable for large-scale datasets in which each class has many training samples.

All the experiments were executed using the same hardware and software conditions. We should point out that, in the training stage, the doublet-SVM, ITML, LDML, MCML, and NCA were worked on the doublet set, while the triplet-SVM and LMNN were implemented on the triplet set. Thus, there are five doublet-based

Table 2.4 The classification error rates (%) and average ranks of the competing methods on the UCI datasets

Method	Doublet-SVM	Triplet-SVM	NCA	LMNN	ITML	MCML	LDML
Parkin sons	5.68	7.89	4.21	5.26	6.32	12.94	7.15
Sonar	13.07	14.29	14.43	11.57	14.86	24.29	22.86
Statlog segmenttion	2.42	2.29	2.68	2.64	2.51	2.77	2.86
Breast tissue	38.37	33.37	30.75	34.37	36.75	30.75	48.00
ILPD	32.09	35.16	34.79	34.12	34.12	34.79	35.84
Statlog satellite	10.80	10.75	10.95	10.05	11.75	15.65	15.90
Blood transfusion	29.47	34.37	28.38	28.78	27.86	31.89	31.40
SPECTF Heart	27.27	33.69	38.50	34.76	35.29	29.95	33.16
Cardiotocography	20.71	19.34	21.84	19.21	18.96	20.76	22.26
Letter	2.47	2.77	2.47	3.45	2.77	4.20	11.05
Average rank	2.7	3.8	3.5	2.9	3.5	5.0	6.1

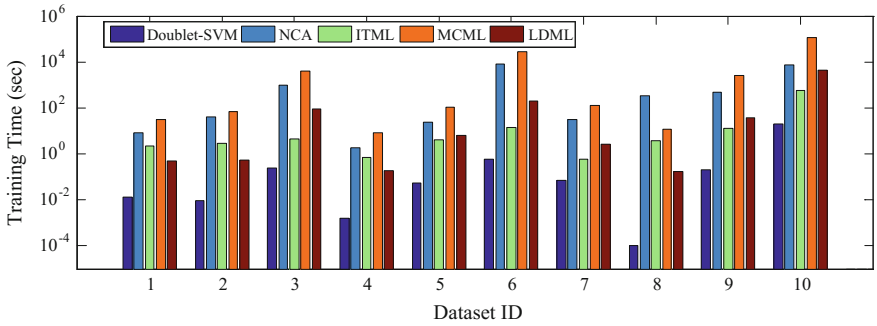

Fig. 2.8 Training time (s) for the doublet-SVM, NCA, ITML, MCML and LDML. From 1–10, the Dataset ID represents Parkinsons, Sonar, Statlog Segmentation, Breast Tissue, ILPD, Statlog satellite, Blood Transfusion, SPECTF Heart, Cardiotocography, and Letter. © 2015 IEEE. Reprinted with permission, from Wang et al. (2013)

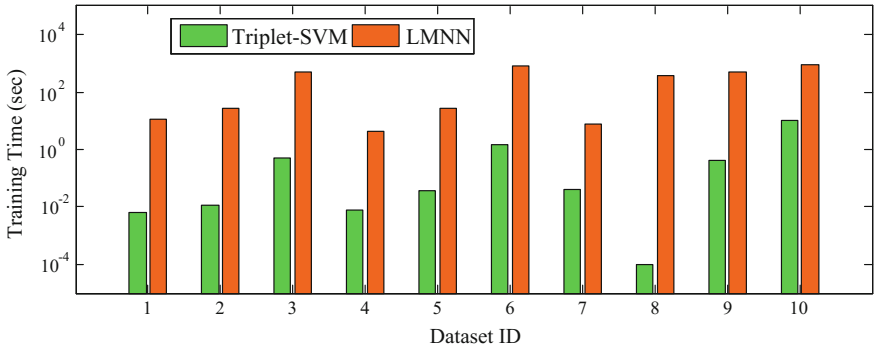

Fig. 2.9 Training time (s) for the triplet-SVM and LMNN. From 1–10, the Dataset ID represents Parkinsons, Sonar, Statlog Segmentation, Breast Tissue, ILPD, Statlog satellite, Blood Transfusion, SPECTF Heart, Cardiotocography, and Letter. © 2015 IEEE. Reprinted with permission, from Wang et al. (2013)

metric learning methods, and two triplet-based methods are compared in the experiments. Figure 2.8 shows the training time of five doublet-based distance metric learning methods. From Fig. 2.8, it is obvious that the doublet-SVM method is much faster than are the other four doublet-based methods. On average, it is 2000 times faster than the second fastest algorithm, ITML. Figure 2.9 shows the training time of two triplet-based distance metric learning methods, the triplet-SVM and LMNN. From Fig. 2.9, we can see that the triplet-SVM is about 100 times faster than LMNN on the ten datasets.

2.4 Summary

In this chapter, we first presented two distance metric learning models based on the support vector machine. The two models can be solved efficiently by the standard SVM solvers. In Sect. 2.3, we presented a general kernel classification framework for metric learning. By coupling a degree-2 polynomial kernel and a positive semidefinite constraint with some kernel methods, the framework can unify many representative and state-of-the-art metric learning methods, such as LMNN, ITML and LDML. On the basis of the kernel classification framework, two novel metric learning methods, namely the doublet-SVM and the triplet-SVM, were presented in detail. Experimental results show that the presented methods obtained better performance than state-of-the-art metric learning methods.

Appendix 1: The Dual of PCML

The original problem of PCML is formulated as

$$\begin{aligned} \min_{\mathbf{M},b,\xi} \quad & \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum_{i,j}\xi_{ij} \\ \text{s.t.} \quad & h_{ij}\big(\langle \mathbf{M}, \mathbf{X}_{ij}\rangle + b\big) \geq 1 - \xi_{ij}, \xi_{ij} \geq 0,\ \forall i,j, \mathbf{M} \succcurlyeq 0. \end{aligned} \tag{2.60}$$

Its Lagrangian is

$$\begin{aligned} L(\lambda,\kappa,\mathbf{Y},\mathbf{M},b,\xi) = \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum_{i,j}\xi_{ij} - \sum_{i,j}\lambda_{ij}\big[h_{ij}\big(\langle \mathbf{M}, \mathbf{X}_{ij}\rangle + b\big) - 1 + \xi_{ij}\big] \\ - \sum_{i,j}\kappa_{ij}\xi_{ij} - \langle \mathbf{Y}, \mathbf{M}\rangle, \end{aligned} \tag{2.61}$$

where λ, κ, and $\mathbf{Y}$ are the Lagrange multipliers that satisfy $\boldsymbol{\lambda}_{ij} \geq 0$, $\boldsymbol{\kappa}_{ij} \geq 0$, $\forall i, j$, and $\mathbf{Y} \succcurlyeq 0$. Based on the KKT conditions, the original problem can be converted into the dual problem. KKT conditions are defined as follows:

$$\frac{\partial L(\lambda,\kappa,\mathbf{Y},\mathbf{M},b,\xi)}{\partial \mathbf{M}} = 0 \Rightarrow \mathbf{M} - \sum_{i,j}\lambda_{ij}h_{ij}\mathbf{X}_{ij} - \mathbf{Y} = 0, \tag{2.62}$$

$$\frac{\partial L(\lambda,\kappa,\mathbf{Y},\mathbf{M},b,\xi)}{\partial b} = 0 \Rightarrow \sum_{i,j}\lambda_{ij}h_{ij} = 0, \tag{2.63}$$

$$\frac{\partial L(\lambda,\kappa,\mathbf{Y},\mathbf{M},b,\xi)}{\partial \xi_{ij}} = C-\lambda_{ij}-\kappa_{ij}=0 \Rightarrow 0\le\lambda_{ij}\le C,\ \forall i,j, \tag{2.64}$$

$$h_{ij}\left(\left\langle \mathbf{M},\mathbf{X}_{ij}\right\rangle + b\right) - 1 + \xi_{ij} \ge 0, \quad \xi_{ij}\ge 0, \tag{2.65}$$

$$\lambda_{ij}\ge 0, \quad \kappa_{ij}\ge 0, \quad \mathbf{Y}\succcurlyeq 0, \tag{2.66}$$

$$\lambda_{ij}\left[h_{ij}\left(\left\langle \mathbf{M},\mathbf{X}_{ij}\right\rangle + b\right) - 1 + \xi_{ij}\right] = 0,\ \kappa_{ij}\xi_{ij}=0. \tag{2.67}$$

Equation (2.62) implies the relationship between λ, $\mathbf{Y}$ and $\mathbf{M}$ as follows:

$$\mathbf{M} = \sum_{i,j}\lambda_{ij}h_{ij}\mathbf{X}_{ij} + \mathbf{Y}. \tag{2.68}$$

Substituting Eqs. (2.62)–(2.69) back into the Lagrangian, we get the following Lagrange dual problem of PCML:

$$\begin{aligned} \max_{\lambda,\mathbf{Y}} \quad & -\frac{1}{2}\left\|\sum_{i,j}\lambda_{ij}h_{ij}\mathbf{X}_{ij}+\mathbf{Y}\right\|_F^2 + \sum_{i,j}\lambda_{ij} \\ \text{s.t.} \quad & \sum_{i,j}\lambda_{ij}h_{ij}=0, 0\le\lambda_{ij}\le C, \mathbf{Y}\succcurlyeq 0. \end{aligned} \tag{2.69}$$

From Eqs. (2.68) and (2.69), we can see that matrix $\mathbf{M}$ is explicitly determined by the training procedure, and b is not. Nevertheless, b can be easily obtained by using the KKT complementarity condition in Eqs. (2.64) and (2.67), which shows that $\xi_{ij} = 0$ if $\lambda_{ij} < C$, and $h_{ij}\left(\left\langle \mathbf{M},\mathbf{X}_{ij}\right\rangle + b\right) - 1 + \xi_{ij} = 0$ if $\lambda_{ij} > 0$. Thus, we can simply take any training point for which $0 < \lambda_{ij} < C$ to calculate b by

$$b = \frac{1}{h_{ij}} - \left\langle \mathbf{M},\mathbf{X}_{ij}\right\rangle, \quad \text{for all } 0<\lambda_{ij}<Cs. \tag{2.70}$$

Note that it is reasonable to take the average of all such training points. After we obtained b, we can calculate ξ_{ij} by

$$\xi_{ij} = \begin{cases} 0 & \text{for all } \lambda_{ij}<C \\ \left[1-h_{ij}\left(\left\langle \mathbf{M},\mathbf{X}_{ij}\right\rangle + b\right)\right]_+ & \text{for all } \lambda_{ij}=C, \end{cases} \tag{2.71}$$

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

Appendix 2: The Dual of NCML

The primal problem of NCML is as follows:

$$\begin{aligned} \min_{\alpha,b,\xi} \quad & \frac{1}{2}\sum_{i,j}\sum_{k,l}\alpha_{ij}\alpha_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + C\sum_{i,j}\xi_{ij} \\ \text{s.t.} \quad & h_{ij}\left(\sum_{k,l}\alpha_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + b\right) \geq 1-\xi_{ij}, \quad \xi_{ij}\geq 0, \quad \alpha_{ij}\geq 0, \quad \forall i,j. \end{aligned} \tag{2.72}$$

Its Lagrangian can be defined as

$$\begin{aligned} L(\beta,\sigma,v,\alpha,b,\xi) = & \frac{1}{2}\sum_{i,j}\sum_{k,l}\alpha_{ij}\alpha_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + C\sum_{i,j}\xi_{ij} - \sum_{i,j}\sigma_{ij}\alpha_{ij} \\ & - \sum_{i,j}\beta_{ij}\left[h_{ij}\left(\sum\nolimits_{kl}\alpha_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + b\right) - 1 + \xi_{ij}\right] - \sum_{i,j}v_{ij}\xi_{ij}, \end{aligned} \tag{2.73}$$

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

$$\frac{\partial L(\beta,\sigma,v,\alpha,b,\xi)}{\partial \alpha_{ij}} = 0 \Rightarrow \sum_{k,l}\alpha_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle - \sum_{k,l}\beta_{kl}h_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle - \sigma_{ij} = 0, \tag{2.74}$$

$$\frac{\partial L(\beta,\sigma,v,\alpha,b,\xi)}{\partial b} = 0 \Rightarrow \sum_{i,j}\beta_{ij}h_{ij} = 0, \tag{2.75}$$

$$\frac{\partial L(\beta,\sigma,v,\alpha,b,\xi)}{\partial \xi_{ij}} = 0 \Rightarrow C-\beta_{ij}-v_{ij} = 0 \Rightarrow 0 \leq \beta_{ij} \leq C, \tag{2.76}$$

$$h_{ij}\left(\sum_{k,l}\alpha_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + b\right) - 1 + \xi_{ij} \geq 0, \quad \xi_{ij}\geq 0, \quad \alpha_{ij}\geq 0, \quad \forall i,j, \tag{2.77}$$

$$\beta_{ij}\geq 0, \ \sigma_{ij}\geq 0, \ v_{ij}\geq 0, \quad \forall i,j, \tag{2.78}$$

$$\beta_{ij}\left[h_{ij}\left(\sum_{kl}\alpha_{kl}\langle \mathbf{X}_{ij}, \mathbf{X}_{kl}\rangle + b\right) - 1 + \xi_{ij}\right] = 0, \quad v_{ij}\xi_{ij} = 0, \quad \sigma_{ij}\alpha_{ij} = 0, \quad \forall i,j. \tag{2.79}$$

Here, we introduce a coefficient vector η, which satisfies $\sigma_{ij} = \sum_{k,l} \eta_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle$, where $\langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle$ denotes a positive definite kernel. Thus, we can guarantee that every η has a unique corresponding $\boldsymbol{\sigma}$, and vice versa. According to Eq. (2.74), the relationship between α, β, and η is

$$\alpha_{ij} = \beta_{ij} h_{ij} + \eta_{ij}, \quad \forall i, j. \tag{2.80}$$

Substituting Eqs. (2.74)–(2.76) back into the Lagrangian, the Lagrange dual problem of NCML can be rewritten as follows:

$$\begin{aligned} \max_{\eta,\beta} \quad & -\frac{1}{2} \sum_{i,j} \sum_{k,l} \left(\beta_{ij} h_{ij} + \eta_{ij}\right)\left(\beta_{kl} h_{kl} + \eta_{kl}\right) \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle + \sum_{i,j} \beta_{ij} \\ \text{s.t.} \quad & \sum_{k,l} \eta_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle \geq 0, \quad 0 \leq \beta_{ij} \leq C, \quad \forall i, j, \quad \sum_{i,j} \beta_{ij} h_{ij} = 0. \end{aligned} \tag{2.81}$$

Analogous to PCML, we can use the KKT complementarity condition in Eq. (2.75) to compute b and ξ_{ij} in NCML. Eqs. (2.76) and (2.79) show that $\xi_{ij} = 0$ if $\beta_{\mathrm{ij}} < C$, and $h_{ij}\left(\sum_{kl} \alpha_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle + b\right) - 1 + \xi_{ij} = 0$ if $\beta_{\mathrm{ij}} > 0$. With any training point for which $0 < \beta_{\mathrm{ij}} < C$, b can be obtained by

$$b = \frac{1}{h_{ij}} - \sum_{kl} \alpha_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle. \tag{2.82}$$

Therefore, β_{ij} can also be obtained by

$$\xi_{ij} = \begin{cases} 0 & \text{for all } \beta_{ij} < C \\ \left[1 - h_{ij}\left(\sum_{k,l} \alpha_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle + b\right)\right]_+ & \text{for all } \beta_{ij} = C, \end{cases} \tag{2.83}$$

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

Appendix 3: The Dual of the Subproblem of η in NCML

The subproblem of η is defined as follows:

$$\begin{aligned} \min_{\eta} \quad & \frac{1}{2} \sum_{i,j} \sum_{k,l} \eta_{ij} \eta_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle + \sum_{i,j} \eta_{ij} \gamma_{ij} \\ \text{s.t.} \quad & \sum_{k,l} \eta_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle \geq 0, \quad \forall i, j, \end{aligned} \tag{2.84}$$

where $\gamma_{ij} = \sum_{k,l} \beta_{kl} h_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle$. Its Lagrangian is

$$L(\mu, \eta) = \frac{1}{2} \sum_{i,j} \sum_{k,l} \eta_{ij} \eta_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle + \sum_{i,j} \eta_{ij} \gamma_{ij} - \sum_{i,j} \mu_{ij} \sum_{k,l} \eta_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle, \quad (2.85)$$

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

$$\frac{\partial L(\mu, \eta)}{\partial \eta_{ij}} = 0 \Rightarrow \sum_{k,l} \eta_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle + \gamma_{ij} - \sum_{k,l} \mu_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle = 0. \quad (2.86)$$

According to Eq. (2.86), the relationship between μ, η and β is

$$\eta_{ij} = \mu_{ij} - h_{ij} \beta_{ij}, \quad \forall i, j. \quad (2.87)$$

Substituting Eqs. (2.86) and (2.87) back into the Lagrangian, we get the following Lagrange dual problem of the subproblem of η

$$\begin{aligned} \max_{\mu} \quad & -\frac{1}{2} \sum_{i,j} \sum_{k,l} \mu_{ij} \mu_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle + \sum_{i,j} \gamma_{ij} \mu_{ij} \\ & -\frac{1}{2} \sum_{i,j} \sum_{k,l} \beta_{ij} \beta_{kl} h_{ij} h_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle \\ \text{s.t.} \quad & \mu_{ij} \geq 0, \forall i, j. \end{aligned} \quad (2.88)$$

Since β is fixed in this subproblem, $\sum_{i,j} \sum_{k,l} \beta_{ij} \beta_{kl} h_{ij} h_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle$ remains constant in Eq. (2.88). Thus, we can omit this term and derive the simplified Lagrange dual problem as follows:

$$\begin{aligned} \max_{\mu} \quad & -\frac{1}{2} \sum_{i,j} \sum_{k,l} \mu_{ij} \mu_{kl} \langle \mathbf{X}_{ij}, \mathbf{X}_{kl} \rangle + \sum_{i,j} \gamma_{ij} \mu_{ij} \\ \text{s.t.} \quad & \mu_{ij} \geq 0, \quad \forall i, j. \end{aligned} \quad (2.89)$$

Appendix 4: The Dual of the Doublet-SVM

According to the original problem of the doublet-SVM defined in Eq. (2.56), its Lagrange function can be defined as follows

$$L(\mathbf{M}, b, \xi, \alpha, \beta) = \frac{1}{2}\|\mathbf{M}\|_F^2 + C\sum_l \xi_l - \sum_l \alpha_l \left[h_l \left((\mathbf{x}_{l,1} - \mathbf{x}_{l,2})^{\mathrm{T}} \mathbf{M} (\mathbf{x}_{l,1} - \mathbf{x}_{l,2}) + b \right) - 1 + \xi_l \right] - \sum_l \beta_l \xi_l, \tag{2.90}$$

where α and β are the Lagrange multipliers that satisfy $\alpha_l \geq 0$ and $\beta_l \geq 0$, $\forall\, l$. To convert the original problem to its dual needs the following KKT conditions:

$$\frac{\partial L(\mathbf{M}, b, \xi, \alpha, \beta)}{\partial \mathbf{M}} = 0 \Rightarrow \mathbf{M} - \sum_l \alpha_l h_l \left(\mathbf{x}_{l,1} - \mathbf{x}_{l,2} \right) \left(\mathbf{x}_{l,1} - \mathbf{x}_{l,2} \right)^{\mathrm{T}} = 0, \tag{2.91}$$

$$\frac{\partial L(\mathbf{M}, b, \xi, \alpha, \beta)}{\partial b} = 0 \Rightarrow \sum_l \alpha_l h_l = 0, \tag{2.92}$$

$$\frac{\partial L(\mathbf{M}, b, \xi, \alpha, \beta)}{\partial \xi_l} = 0 \Rightarrow C - \alpha_l - \beta_l = 0 \Rightarrow 0 < \alpha_l < C, \quad \forall l. \tag{2.93}$$

According to Eq. (2.91), the relationship between $\mathbf{M}$ and α is

$$\mathbf{M} = \sum_l \alpha_l h_l \left(\mathbf{x}_{l,1} - \mathbf{x}_{l,2} \right) \left(\mathbf{x}_{l,1} - \mathbf{x}_{l,2} \right)^{\mathrm{T}}. \tag{2.94}$$

Substituting Eqs. (2.91)–(2.93) back into the Lagrangian function, we have

$$L(\alpha) = -\frac{1}{2}\sum_{i,j} \alpha_i \alpha_j h_i h_j K_p \left(\mathbf{z}_i, \mathbf{z}_j \right) + \sum_i \alpha_i. \tag{2.95}$$

Thus, the dual problem of the doublet-SVM can be formulated as follows:

$$\begin{aligned} \max_{\alpha} \; & -\frac{1}{2}\sum_{i,j} \alpha_i \alpha_j h_i h_j K_p \left(\mathbf{z}_i, \mathbf{z}_j \right) + \sum_i \alpha_i \\ \text{s.t.} \quad & 0 \leq \alpha_l \leq C, \quad \sum_l \alpha_l h_l = 0, \quad \forall l. \end{aligned} \tag{2.96}$$

Appendix 5: The Dual of the Triplet-SVM

According to the original problem of the triplet-SVM in Eq. (2.58), its Lagrange function can be defined as follows:

$$L(\mathbf{M},\xi,\alpha,\beta)=\frac{1}{2}\|\mathbf{M}\|_F^2+C\sum_l\xi_l-\sum_l\alpha_l[(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})^{\mathrm{T}}\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,3}) - (\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^{\mathrm{T}}\mathbf{M}(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})]+\sum_l\alpha_l-\sum_l\alpha_l\xi_l-\sum_l\beta_l\xi_l, \tag{2.97}$$

where α and β are the Lagrange multipliers. To convert the original problem to its dual, we let the derivative of the Lagrangian function requires the following KKT conditions:

$$\begin{aligned}&\frac{\partial L(\mathbf{M},b,\xi,\alpha,\beta)}{\partial\mathbf{M}}=0\Rightarrow\\&\mathbf{M}-\sum_l\alpha_l\Big[(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})^{\mathrm{T}}-(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^{\mathrm{T}}\Big]=0,\end{aligned} \tag{2.98}$$

$$\frac{\partial L(\mathbf{M},b,\xi,\alpha,\beta)}{\partial\xi_l}=0\Rightarrow C-\alpha_l-\beta_l=0,\ \forall l. \tag{2.99}$$

According to Eq. (2.98), the relationship between $\mathbf{M}$ and α is:

$$\mathbf{M}=\sum_l\alpha_l\Big[(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})(\mathbf{x}_{l,1}-\mathbf{x}_{l,3})^{\mathrm{T}}-(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})(\mathbf{x}_{l,1}-\mathbf{x}_{l,2})^{\mathrm{T}}\Big]. \tag{2.100}$$

Substituting Eqs. (2.98) and (2.99) back into the Lagrangian, we get

$$L(\alpha)=-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_jK_p(\mathbf{t}_i,\mathbf{t}_j)+\sum_i\alpha_i. \tag{2.101}$$

Thus, the dual problem of the triplet-SVM can be rewritten as follows:

$$\begin{aligned}&\max_{\alpha}\ -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_jK_p(\mathbf{t}_i,\mathbf{t}_j)+\sum_i\alpha_i\\&\text{s.t.}\quad 0\le\alpha_l\le C,\quad\forall l.\end{aligned} \tag{2.102}$$

References

S. Andrews, I. Tsochantaridis, T. Hofmann, Support vector machines for multiple-instance learning, in *Proceedings of Advances in Neural Information Processing Systems* (2002), pp. 561–568

M.-F. Balcan, A. Blum, N. Srebro, A theory of learning with similarity functions. Mach. Learn. **72** (1–2), 89–112 (2008)

M. Belkin, P. Niyogi, V. Sindhwani, Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. J. Mach. Learn. Res. **7**, 2399–2434 (2006)
A. Bellet, A. Habrard, M. Sebban, Good edit similarity learning by loss minimization. Mach. Learn. **89**(1–2), 5–35 (2012)
A. Bellet, A. Habrard, M. Sebban, A survey on metric learning for feature vectors and structured data. arXiv preprint arXiv:1306.6709 (2002)
A. Bordes, L. Bottou, P. Gallinari, J. Weston, Solving multiclass support vector machines with LaRank, in *Proceedings of the 24th International Conference on Machine Learning* (ACM, 2007), pp. 89–96
C.-C. Chang, C.-J. Lin, LIBSVM: A library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**(3), 27 (2011)
R. Collobert, S. Bengio, Y. Bengio, A parallel mixture of SVMs for very large scale problems. Neural Comput. **14**(5), 1105–1114 (2002)
I. Csisz, G. TUSN DY, Information geometry and alternating minimization procedures. Stat. Decis. **1**, 205–237 (1984)
J.V. Davis, B. Kulis, P. Jain, S. Sra, I.S. Dhillon, Information-theoretic metric learning, in *Proceedings of the 24th International Conference on Machine Learning* (ACM, 2007), pp. 209–216
J. Demšar, Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006)
T. Evgeniou, M. Pontil, Regularized multi-task learning, in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2004), pp. 109–117
A. Frank, A. Asuncion, UCI machine learning repository (2010). Available: http://archive.ics.uci.edu/ml
Y. Fu, S. Yan, T.S. Huang, Correlation metric for generalized feature extraction. IEEE Trans. Pattern Anal. Mach. Intell. **30**(12), 2229–2235 (2008)
A. Globerson, S.T. Roweis, Metric learning by collapsing classes, in *Proceedings of Advances in Neural Information Processing Systems* (2005), pp. 451–458
J. Goldberger, G.E. Hinton, S.T. Roweis, R. Salakhutdinov, Neighbourhood components analysis, in *Proceedings of Advances in Neural Information Processing Systems* (2004), pp. 513–520
M. Guillaumin, J. Verbeek, C. Schmid, Is that you? Metric learning approaches for face identification, in *Proceedings of IEEE International Conference on Computer Vision* (IEEE, 2009), pp. 498–505
A. Gunawardana, W. Byrne, Convergence theorems for generalized alternating minimization procedures. J. Mach. Learn. Res. **6**, 2049–2073 (2005)
C. Huang, S. Zhu, K. Yu, Large scale strongly supervised ensemble metric learning, with applications to face verification and retrieval. arXiv preprin arXiv:1212.6094 (2012)
G.B. Huang, M. Ramesh, T. Berg, E. Learned-Miller, Labeled faces in the wild: a database for studying face recognition in unconstrained environments. Technical Report 07-49 (University of Massachusetts, Amherst, 2007)
D. Kedem, S. Tyree, F. Sha, G.R. Lanckriet, K.Q. Weinberger, Non-linear metric learning, in *Proceedings of Advances in Neural Information Processing Systems* (2012), pp. 2573–2581
S.S. Keerthi, K. Duan, S.K. Shevade, A.N. Poo, A fast dual algorithm for kernel logistic regression. Mach. Learn. **61**(1–3), 151–165 (2005)
K. Koh, S.-J. Kim, S.P. Boyd, An interior-point method for large-scale l1-regularized logistic regression. J. Mach. Learn. Res. **8**(8), 1519–1555 (2007)
X. Li, C. Shen, Q. Shi, A. Dick, A. Van den Hengel, Non-sparse linear representations for visual tracking with online reservoir metric learning, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2012), pp. 1760–1767
K.-R. M Ller, S. MIKA, G. R TSCH, K. TSUDA, B. SCH LKOPF, An introduction to kernel-based learning algorithms. IEEE Trans. Neural Netw. **12**(2), 181–201 (2001)

T. Mensink, J. Verbeek, F. Perronnin, G. Csurka, Metric learning for large scale image classification: generalizing to new classes at near-zero cost, in *Proceedings of Computer Vision–ECCV* (Springer, Berlin, 2012), pp. 488–501
J. Platt, Fast training of support vector machines using sequential minimal optimization. Adv. Kernel Methods Support Vector Learn. **3**, 185–208 (1999)
B. Schlkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, R.C. Williamson, Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)
S. Shalev-Shwartz, Y. Singer, N. Srebro, A. Cotter, Pegasos: primal estimated sub-gradient solver for SVM. Math. Program. **127**(1), 3–30 (2011)
J. Shawe-Taylor, N. Cristianini, *Kernel Methods for Pattern Analysis* (Cambridge University Press, Cambridge, 2010)
C. Shen, J. Kim, L. Wang, A scalable dual approach to semidefinite metric learning, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2011), pp. 2601–2608
C. Shen, J. Kim, L. Wang, A. Hengel, Positive semidefinite metric learning with boosting, in *Proceedings of Advances in Neural Information Processing Systems* (2009), pp. 1651–1659
C.H. Teo, A. Smola, S. Vishwanathan, Q.V. Le, A scalable modular convex solver for regularized risk minimization, in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2007), pp. 727–736
I.W. Tsang, A. Kocsor, J.T. Kwok, Simpler core vector machines with enclosing balls, in *Proceedings of the 24th International Conference on Machine Learning* (ACM, 2007), pp. 911–918
I.W. Tsang, J.T. Kwok, P.-M. Cheung, Core vector machines: fast SVM training on very large data sets. J. Mach. Learn. Res. 363–392 (2005)
V. Vapnik, *The Nature of Statistical Learning Theory* (Springer Science & Business Media, Berlin, 2013)
F. Wang, W. Zuo, L. Zhang, D. Meng, D. Zhang, A kernel classification framework for metric learning (2013)
J. Wang, H.T. Do, A. Woznica, A. Kalousis, Metric learning with multiple kernels, in *Proceedings of Advances in Neural Information Processing Systems* (2011), pp. 1170–1178
J. Wang, A. Kalousis, A. Woznica, Parametric local metric learning for nearest neighbor classification, in *Proceedings of Advances in Neural Information Processing Systems* (2012), pp. 1601–1609
K.Q. Weinberger, J. Blitzer, L.K. Saul, Distance metric learning for large margin nearest neighbor classification. J. Mach. Learn. Res. **10**, 207–244 (2009)
L. Yang, R. Jin, *Distance Metric Learning: A comprehensive Survey,* vol. 2 (Michigan State University, 2006)
Y. Ying, P. Li, Distance metric learning with eigenvalue optimization. J. Mach. Learn. Res. **13**(1), 1–26 (2012)
W. Zuo, F. Wang, D. Zhang, L. Lin, Y. Huang, D. Meng, L. Zhang, Iterated support vector machines for distance metric learning. arXiv preprint arXiv:1502.00363 (2015)

http://www.springer.com/978-981-10-2055-1

Discriminative Learning in Biometrics

Zhang, D.; Xu, Y.; Zuo, W.

2016, XIII, 266 p. 110 illus., 73 illus. in color., Hardcover
ISBN: 978-981-10-2055-1