Flexible Nonparametric Kernel Learning with Different Loss Functions

En-Liang Hu[1,2] and James T. Kwok[1]

[1] Department of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong
[2] Department of Mathematics
Yunnan Normal University, Yunnan, China
{jamesk,ynelhu}@cse.ust.hk

Abstract. Side information is highly useful in the learning of a nonparametric kernel matrix. However, this often leads to an expensive semidefinite program (SDP). In recent years, a number of dedicated solvers have been proposed. Though much better than off-the-shelf SDP solvers, they still cannot scale to large data sets. In this paper, we propose a novel solver based on the alternating direction method of multipliers (ADMM). The key idea is to use a low-rank decomposition of the kernel matrix $\mathbf{Z} = \mathbf{X}^\top \mathbf{Y}$, with the constraint that $\mathbf{X} = \mathbf{Y}$. The resultant optimization problem, though non-convex, has favorable convergence properties and can be efficiently solved without requiring eigen-decomposition in each iteration. Experimental results on a number of real-world data sets demonstrate that the proposed method is as accurate as directly solving the SDP, but can be one to two orders of magnitude faster.

1 Introduction

Kernel methods have been highly successful in classification, regression, clustering, ranking, and dimensionality reduction. Because of the central role of the kernel, it is important to identify an appropriate kernel function or matrix for the task at hand. Over the past decade, there have been a large body of literature on this kernel learning problem [8,1]. While a parametric form of the kernel or a combination of multiple kernels are often assumed, nonparametric kernel learning, which takes no such assumptions, is more flexible and has received significant interest in recent years [10,7,6,14,11].

To facilitate kernel learning, obviously one has to utilize information from the data. The most straightforward approach is to use class labels. However, obtaining label information may sometimes be expensive and time-consuming. In this paper, we focus on a weaker form of supervisory information, namely, the so-called *must-link* and *cannot-link* pairwise constraints [12]. These pairwise constraints, or *side information*, define whether the two patterns involved should belong to the same class or not. Another useful source of information, which is commonly used in semi-supervised learning, is the data manifold [2]. This encourages patterns that are locally nearby on the manifold to have similar predicted labels.

M. Lee et al. (Eds.): ICONIP 2013, Part II, LNCS 8227, pp. 116–123, 2013.

To learn a kernel matrix $\mathbf{Z}$, we consider the following SDP problem

$$\min_{\mathbf{Z}\succeq 0} \mathscr{R}(\mathbf{Z}) + \lambda \ell(\mathbf{Z}, \mathbf{T}), \tag{1}$$

where $\mathbf{Z} = [Z_{ij}] \in \mathbb{R}^{n\times n}$ is the kernel matrix to be learned (which has to be symmetric and positive semidefinite (psd), denoted $\mathbf{Z} \succeq \mathbf{0}$), $\mathbf{L}$ is the graph Laplacian matrix of the data manifold, $\mathbf{T} = [T_{ij}]$ with a similarity/dissimilarity (resp. *must-link*/*cannot-link*) indicator matrix such that $T_{ij} = \begin{cases} 1 & (i,j) \in \mathcal{S}, \\ -1 & (i,j) \in \mathcal{D}, \end{cases}$, and λ is a regularization parameter. As in [6,9,11], we use a low-rank approximation on the kernel matrix $\mathbf{Z}$. In other words, $\mathbf{Z}$ is approximated as $\mathbf{Z} \simeq \mathbf{X}^\top \mathbf{X}$, where $\mathbf{X} \in \mathbb{R}^{r\times n}$ and rank $r \ll n$. Problem (1) can then be rewritten as

$$\min_{\mathbf{X},\mathbf{Z}} \mathscr{R}(\mathbf{Z}) + \lambda \ell(\mathbf{Z}, \mathbf{T}) \ : \ \mathbf{Z} = \mathbf{X}^\top \mathbf{X}. \tag{2}$$

In this paper, we present a novel solver for (2) based on the alternating direction method of multipliers (ADMM) [3]. Our key observation is that (2) can often be decoupled as $\sum_{ij} \left(\tilde{\mathscr{R}}(Z_{ij}) + \tilde{\ell}(Z_{ij}, T_{ij}) \right)$. Thus, solving (2) reduces to the solving of each individual entry Z_{ij}, which is easier and more efficient.

Notations: In the sequel, matrices and vectors are denoted in bold, with upper-case letters for matrices and lower-case for vectors. The transpose of a vector/matrix is denoted by the superscript $^\top$. Moreover, $\mathbf{I}$ is the identity matrix.

2 Alternating Direction Method of Multipliers (ADMM)

ADMM is a simple but powerful algorithm that has been successfully used in machine learning and data mining. The standard ADMM is for solving convex problems. Here, we consider the more general bi-convex problem [3]:

$$\min_{\mathbf{x},\mathbf{y}} F(\mathbf{x}, \mathbf{y}) \ : \ G(\mathbf{x}, \mathbf{y}) = \mathbf{0}, \tag{3}$$

where $F(\cdot,\cdot)$ is bi-convex and $G(\cdot,\cdot)$ is bi-affine[1]. As in the method of multipliers, the more general ADMM considers the augmented Lagrangian of (3): $\mathscr{L}(\mathbf{x}, \mathbf{y}, \boldsymbol{\Lambda}) = F(\mathbf{x}, \mathbf{y}) + \boldsymbol{\Lambda}^\top G(\mathbf{x}, \mathbf{y}) + \frac{\rho}{2} \|G(\mathbf{x}, \mathbf{y})\|^2$, where $\boldsymbol{\Lambda}$ is the vector of Lagrangian multipliers, and $\rho > 0$ is a penalty parameter. At the kth iteration, the values of $\mathbf{x}, \mathbf{y}$ and $\boldsymbol{\Lambda}$ (denoted $\mathbf{x}^k, \mathbf{y}^k$ and $\boldsymbol{\Lambda}^k$) are updated as

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \mathscr{L}(\mathbf{x}, \mathbf{y}^k, \boldsymbol{\Lambda}^k), \ \mathbf{y}^{k+1} = \arg\min_{\mathbf{y}} \mathscr{L}(\mathbf{x}^{k+1}, \mathbf{y}, \boldsymbol{\Lambda}^k),$$

$$\boldsymbol{\Lambda}^{k+1} = \boldsymbol{\Lambda}^k + \rho G(\mathbf{x}^{k+1}, \mathbf{y}^{k+1}).$$

Note that while the method of multipliers minimizes $\mathscr{L}(\mathbf{x}, \mathbf{y}, \boldsymbol{\Lambda}^k)$ w.r.t. $\mathbf{x}$ and $\mathbf{y}$ jointly, ADMM allows easier decomposition of the optimization problem by minimizing them in an alternating manner.

[1] In other words, for any fixed $\mathbf{x}, \mathbf{y}$, $F(\cdot, \mathbf{y})$ and $F(\mathbf{x}, \cdot)$ are convex; while $G(\cdot, \mathbf{y})$ and $G(\mathbf{x}, \cdot)$ are affine.

3 Kernel Learning by ADMM

In this section, we introduce an extra variable $\mathbf{Y}$ to allow easier decoupling of the optimization problem. Specifically, (2) can be equivalently formulated as

$$\min_{\mathbf{X},\mathbf{Y},\mathbf{Z}} \mathscr{R}(\mathbf{Z}) + \lambda\ell(\mathbf{Z},\mathbf{T}) \ : \ \mathbf{Z} = \mathbf{X}^\top\mathbf{Y},\ \mathbf{X} = \mathbf{Y}. \tag{4}$$

Consider $\mathscr{R}(\mathbf{Z}) = tr(\mathbf{ZL})$. Notice that both $tr(\mathbf{ZL})$ and $\ell(\mathbf{Z},\mathbf{T})$ are bi-convex, and that the constraints are bi-affine. The augmented Lagrangian of (4) is

$$\begin{aligned}\mathscr{L}(\mathbf{X},\mathbf{Y},\mathbf{Z};\boldsymbol{\Lambda},\boldsymbol{\Pi}) = tr(\mathbf{ZL}) + \lambda\ell(\mathbf{Z},\mathbf{T}) + \boldsymbol{\Lambda}\bullet(\mathbf{Z}-\mathbf{X}^\top\mathbf{Y}) \\ +\frac{\alpha}{2}\left\|\mathbf{Z}-\mathbf{X}^\top\mathbf{Y}\right\|^2 + \boldsymbol{\Pi}\bullet(\mathbf{X}-\mathbf{Y}) + \frac{\beta}{2}\left\|\mathbf{X}-\mathbf{Y}\right\|^2,\end{aligned}$$

where $\boldsymbol{\Lambda},\boldsymbol{\Pi}$ are the Lagrange multipliers. ADMM then updates the variables as

$$\mathbf{X}^{k+1} = \arg\min_{\mathbf{X}} \mathscr{L}(\mathbf{X},\mathbf{Y}^k,\mathbf{Z}^k;\boldsymbol{\Lambda}^k,\boldsymbol{\Pi}^k), \mathbf{Y}^{k+1} = \arg\min_{\mathbf{Y}} \mathscr{L}(\mathbf{X}^{k+1},\mathbf{Y},\mathbf{Z}^k;\boldsymbol{\Lambda}^k,\boldsymbol{\Pi}^k),$$

$$\mathbf{Z}^{k+1} = \arg\min_{\mathbf{Z}} \mathscr{L}(\mathbf{X}^{k+1},\mathbf{Y}^{k+1},\mathbf{Z};\boldsymbol{\Lambda}^k,\boldsymbol{\Pi}^k),$$

$$\boldsymbol{\Lambda}^{k+1} = \boldsymbol{\Lambda}^k + \alpha(\mathbf{Z}^{k+1}-\mathbf{S}^{k+1}),\ \boldsymbol{\Pi}^{k+1} = \boldsymbol{\Pi}^k + \beta(\mathbf{X}^{k+1}-\mathbf{Y}^{k+1}), \tag{5}$$

where $\mathbf{S}^k \equiv {\mathbf{X}^k}^\top\mathbf{Y}^k$. By straightforward differentiation, the optimization subproblems of $\mathbf{X}^{k+1},\mathbf{Y}^{k+1}$ and $\mathbf{Z}^{k+1}$ can be solved as

$$\mathbf{X}^{k+1} = (\mathbf{A}^k)^{-1}\mathbf{c}^k, \tag{6}$$

where $\mathbf{A}^k = \beta\mathbf{I} + \alpha\mathbf{Y}^k{\mathbf{Y}^k}^\top$, and $\mathbf{c}^k = \mathbf{Y}^k(\boldsymbol{\Lambda}^k + \alpha\mathbf{Z}^k + \beta\mathbf{I})^\top - \boldsymbol{\Pi}^k$;

$$\mathbf{Y}^{k+1} = (\mathbf{B}^k)^{-1}\mathbf{d}^k, \tag{7}$$

where $\mathbf{B}^k = \beta\mathbf{I} + \alpha\mathbf{X}^{k+1}{\mathbf{X}^{k+1}}^\top$, and $\mathbf{d}^k = \mathbf{X}^{k+1}(\boldsymbol{\Lambda}^k + \alpha\mathbf{Z}^k + \beta\mathbf{I}) + \boldsymbol{\Pi}^k$;

$$Z_{ij}^{k+1} = \begin{cases} \min\limits_z f_{ij}(z) & (i,j)\in\Omega \quad (8a) \\ \left(\mathbf{S}^{k+1} - \frac{1}{\alpha}(\boldsymbol{\Lambda}^k + \mathbf{L})\right)_{ij} & (i,j)\notin\Omega, \quad (8b)\end{cases}$$

where $\Omega = \mathcal{S}\cup\mathcal{D}$, and

$$f_{ij}(z) = \lambda\tilde{\ell}(z,T_{ij}) + \frac{\alpha}{2}\left(z - S_{ij}^{k+1} + \frac{\Lambda_{ij}^k + L_{ij}}{\alpha}\right)^2. \tag{9}$$

Hence, the only remaining issue is how to solve (8a).

3.1 Different Loss Functions

In this section, we show that problem (8a) can be easily solved for a variety of loss functions.

ℓ_2-Loss $\tilde{\ell}(z, T_{ij}) = \frac{1}{2}(z - T_{ij})^2$. Problem (8a) can be easily solved by setting the derivative of the objective in (9) to zero, leading to

$$z^* = \frac{1}{\alpha + \lambda}(\alpha S_{ij}^{k+1} + \lambda T_{ij} - \Lambda_{ij}^k - L_{ij}). \tag{10}$$

Hinge Loss $\tilde{\ell}(z, T_{ij}) = \max\{1 - zT_{ij}, 0\}$. Problem (8a) can be rewritten as $\min_{z,\epsilon}\ \lambda\epsilon + \frac{\alpha}{2}\left(z - S_{ij}^{k+1} + \frac{\Lambda_{ij}^k + L_{ij}}{\alpha}\right)^2 :\ T_{ij}z \geq 1 - \epsilon,\ \epsilon \geq 0$. Let θ be the Lagrange multiplier for the constraint $T_{ij}z \geq 1 - \epsilon$. Using the standard method of Lagrange multipliers, it can be easily shown that

$$z^* = \frac{\theta T_{ij} - \Lambda_{ij}^k - L_{ij}}{\alpha} + S_{ij}^{k+1}, \tag{11}$$

where $\theta = \min\left\{\max\left\{\frac{\alpha - T_{ij}(\alpha S_{ij}^{k+1} - \Lambda_{ij}^k - L_{ij})}{T_{ij}^2}, 0\right\}, \lambda\right\}$.

Squared Hinge Loss $\tilde{\ell}(z, T_{ij}) = \frac{1}{2}\max\{1 - zT_{ij}, 0\}^2$. Problem (8a) can be rewritten as $\min_{z,\epsilon}\ \frac{\lambda}{2}\epsilon^2 + \frac{\alpha}{2}\left(z - S_{ij}^{k+1} + \frac{\Lambda_{ij}^k + L_{ij}}{\alpha}\right)^2 :\ T_{ij}z \geq 1 - \epsilon$. Similar to the hinge loss, the optimal solution can be obtained as

$$z^* = \frac{\theta T_{ij} - \Lambda_{ij}^k - L_{ij}}{\alpha} + S_{ij}^{k+1}, \tag{12}$$

where $\theta = \max\left\{\frac{\alpha - T_{ij}(\alpha S_{ij}^{k+1} - \Lambda_{ij}^k - L_{ij})}{\frac{\alpha}{\lambda} + T_{ij}^2}, 0\right\}$.

ℓ_1-Loss $\tilde{\ell}(z, T_{ij}) = |z - T_{ij}|$. With the ℓ_1-loss, $f_{ij}(z)$ in (9) can be written as

$$\begin{aligned} f_{ij}(z) &= \frac{\lambda}{\alpha}|z - T_{ij}| + \frac{1}{2}\left(z - S_{ij}^{k+1} + \frac{\Lambda_{ij}^k + L_{ij}}{\alpha}\right)^2 \\ &= \frac{\lambda}{\alpha}|\hat{z}| + \frac{1}{2}\left(\hat{z} + T_{ij} - S_{ij}^{k+1} + \frac{\Lambda_{ij}^k + L_{ij}}{\alpha}\right)^2, \end{aligned}$$

where $\hat{z} = z - T_{ij}$. Hence, problem (8a) becomes a standard problem with ℓ_2-loss and ℓ_1-regularizer, and the optimal $\hat{z}^*$ can be obtained as

$$\hat{z}^* = \mathrm{Th}_{\frac{\lambda}{\alpha}}\left(S_{ij}^{k+1} - T_{ij} - \frac{L_{ij} + \Lambda_{ij}^k}{\alpha}\right),$$

where $\mathrm{Th}_\nu(x) = \begin{cases} x - \nu & x > \nu \\ 0 & -\nu \leq |x| \leq \nu \\ x + \nu & x < -\nu, \end{cases}$ is the soft-thresholding operator. Consequently,

$$z^* = \hat{z}^* + T_{ij} = \mathrm{Th}_{\frac{\lambda}{\alpha}}\left(S_{ij}^{k+1} - T_{ij} - \frac{L_{ij} + \Lambda_{ij}^k}{\alpha}\right) + T_{ij}. \tag{13}$$

3.2 Algorithm

To monitor convergence, we require the primal and dual residuals at iteration $k+1$

$$\Delta_{\text{primal}_1} = \left\|\mathbf{Z}^{k+1} - \mathbf{S}^{k+1}\right\|, \Delta_{\text{primal}_2} = \left\|\mathbf{X}^{k+1} - \mathbf{Y}^{k+1}\right\|, \Delta_{\text{dual}} = \left\|\mathbf{X}^{k+1} - \mathbf{X}^{k}\right\| \tag{14}$$

to be small [3]. Moreover, to improve convergence, it is common to vary the penalty parameters in each ADMM iteration. Specifically, following [3], we update them as

$$\alpha \leftarrow \begin{cases} 2\alpha & \Delta_{\text{primal}_1} > 10\Delta_{\text{dual}} \\ \max(\alpha/2, 1.5) & \Delta_{\text{dual}} > 10\Delta_{\text{primal}_1} \end{cases}, \tag{15}$$

and

$$\beta \leftarrow \begin{cases} 2\beta & \Delta_{\text{primal}_2} > 10\Delta_{\text{dual}} \\ \max(\beta/2, 1.5) & \Delta_{\text{dual}} > 10\Delta_{\text{primal}_2} \end{cases}. \tag{16}$$

The whole procedure is shown in Algorithm 1.

Algorithm 1. Kernel learning by ADMM.

1: **Input:** $\mathbf{X}^0, \mathbf{Y}^0, \mathbf{Z}^0$, parameters ε and $IterMax$.
2: **Output:** $\mathbf{Z} = \mathbf{X}^{k\top}\mathbf{X}^k$ (or $\mathbf{Y}^{k\top}\mathbf{Y}^k$).
3: $k \leftarrow 0$;
4: **repeat**
5: update $\mathbf{X}^{k+1}, \mathbf{Y}^{k+1}$ by (6) and (7) respectively;
6: update $\{Z_{ij}^{k+1} \mid (i,j) \notin \Omega\}$ by (8a);
7: update $\{Z_{ij}^{k+1} \mid (i,j) \in \Omega\}$ by (10), (11), (12) or (13), depending on the loss;
8: update $\boldsymbol{\Lambda}^{k+1}, \boldsymbol{\Pi}^{k+1}$ by (5);
9: update α and β using (15) and (16);
10: compute the primal and dual residuals in (14);
11: $k \leftarrow k+1$;
12: **until** $\max(\Delta_{\text{primal}_1}, \Delta_{\text{primal}_2}, \Delta_{\text{dual}}) < \varepsilon$ or $k > IterMax$.

3.3 Convergence

With the low-rank decomposition, problem (4) is nonconvex w.r.t. $\mathbf{X}$ and $\mathbf{Y}$, and so we can only consider local convergence [3]. As in [13], we show below a necessary condition for local convergence.

Lemma 1. *Let* $\mathbf{W} \equiv (\mathbf{X}, \mathbf{Y}, \mathbf{Z})$, $\boldsymbol{\Gamma} \equiv (\boldsymbol{\Lambda}, \boldsymbol{\Pi})$, *and* $\{\mathbf{W}^k, \boldsymbol{\Gamma}^k)\}$ *be a sequence generated by Algorithm 1. Then,* $\mathscr{L}(\mathbf{W}^k, \cdot) - \mathscr{L}(\mathbf{W}^{k+1}, \cdot) \geq \mu \left\|\mathbf{W}^k - \mathbf{W}^{k+1}\right\|^2$, *and* $\mathscr{L}(\boldsymbol{\Gamma}^k, \cdot) - \mathscr{L}(\boldsymbol{\Gamma}^{k+1}, \cdot) \geq -\frac{1}{\mu}\left\|\mathbf{W}^k - \mathbf{W}^{k+1}\right\|^2$, *where* $\mu = \min\{\alpha, \beta\}$, *and* $\mathscr{L}(\mathbf{X}, \cdot)$ *denotes that all the variables in* $\mathscr{L}$ *except* $\mathbf{X}$ *are fixed.*

Proposition 1. *Let* $\{\mathbf{X}^k, \mathbf{Y}^k, \mathbf{Z}^k, \boldsymbol{\Lambda}^k, \boldsymbol{\Pi}^k\}$ *be a sequence generated by Algorithm 1. If* $\{\boldsymbol{\Lambda}^k, \boldsymbol{\Pi}^k\}$ *is bounded and* $\sum_{k=0}^{\infty}\left(\left\|\boldsymbol{\Lambda}^{k+1} - \boldsymbol{\Lambda}^k\right\|^2 + \left\|\boldsymbol{\Pi}^{k+1} - \boldsymbol{\Pi}^k\right\|^2\right) < \infty$, *then* $\mathbf{X}^k - \mathbf{X}^{k+1} \to \mathbf{0}, Y^k - \mathbf{Y}^{k+1} \to \mathbf{0}, \mathbf{Z}^k - \mathbf{Z}^{k+1} \to \mathbf{0}$, *and any accumulation point of* $\{\mathbf{X}^k, \mathbf{Y}^k, \mathbf{Z}^k\}$ *satisfies the Karush-Kuhn-Tucker condition of (4).*

4 Experiments

As in [5,10], we study the performance of the proposed approach in the context of data clustering. Specifically, a kernel matrix is learned from the pairwise constraints (i.e., *must-link* and *cannot-link*), which is then used for clustering by the kernel k-means algorithm. Experiments are performed on a number of

Table 1. Data sets used in the experiment

data set	#classes	#patterns	#features	#constraints
glass	6	214	9	256
heart	2	270	13	324
iris	3	150	4	180
protein	6	116	20	140
sonar	2	208	60	250
wine	3	178	12	214
odd-even	2	4000	256	4800

Table 2. Comparison on clustering accuracy (%) on the data sets with the squared loss (SL), hinge loss (HL), squared hinge loss (SHL), and ℓ_1-loss (ℓ_1). The best and comparable results (according to the pairwise t-test with 95% confidence) are highlighted.

loss	method	glass	heart	iris	protein	sonar	wine	odd-even
SL	ADMM	**80.9±1.1**	**93.3±1.9**	**99.0±1.1**	**87.7±1.9**	**95.5±2.5**	**83.2±1.8**	**99.37±0.06**
	BCD	**80.8±1.0**	**93.4±2.5**	**99.2±0.9**	86.0±1.9	**95.7±2.5**	82.9±1.7	**99.57±0.03**
	S-NPKL	80.3±1.8	91.7±2.5	**99.2±1.0**	86.7±2.5	**95.2±2.6**	82.9±1.8	**99.35±0.09**
HL	ADMM	**80.1±1.2**	**92.4±2.5**	**99.2±1.0**	**87.2±1.7**	**95.9±2.4**	**83.4±1.6**	**99.35±0.05**
	BCD	79.6±1.9	85.9±3.0	98.9±1.0	85.9±2.0	94.2±3.9	**83.2±1.8**	**99.55±0.05**
	S-NPKL	**80.6±1.1**	88.1±4.1	**99.1±0.8**	85.0±2.5	**95.5±2.4**	**83.3±1.6**	**99.42±0.08**
	SDPLR	79.6±1.5	90.0±2.8	98.9±0.7	83.1±2.8	94.9±1.2	82.1±2.3	98.35±0.20
SHL	ADMM	80.7±1.6	**93.4±2.0**	**99.0±1.1**	**86.8±1.8**	**95.5±2.5**	**83.1±2.1**	**99.35±0.10**
	BCD	**81.0±1.6**	**93.4±2.5**	**99.2±0.9**	**86.4±2.3**	**95.7±2.5**	**82.9±1.7**	**99.57±0.03**
	S-NPKL	80.5±2.0	**93.2±2.2**	**99.2±1.0**	82.0±2.2	**95.5±2.5**	**83.0±2.0**	**99.35±0.09**
ℓ_1	ADMM	79.1±1.8	88.0±4.5	98.2±1.6	84.4±2.5	94.7±2.4	80.5±2.4	96.62±0.12

Table 3. Comparison on CPU time (second) on the data setswith the squared loss (SL), hinge loss (HL), squared hinge loss (SHL), and ℓ_1-loss (ℓ_1). The best and comparable results (according to the pairwise t-test with 95% confidence) are highlighted.

loss	method	glass	heart	iris	protein	sonar	wine	odd-even
SL	ADMM	**5.5±1.2**	**10.4±2.7**	2.3±1.0	**1.6±1.6**	**6.8±1.8**	**3.6±2.3**	**359±12.5**
	BCD	9.9±4.2	28.9±10.1	**0.9±0.1**	2.2±1.5	8.3±4.9	4.0±2.1	653±18.4
	S-NPKL	46.2±19.3	311.0±149.2	23.4±7.6	60.8±40.2	176.0±58.7	37.0±17.5	3,680±219.8
HL	ADMM	**6.9±1.8**	**12.8±7.8**	**2.9±1.5**	**1.8±1.5**	**7.4±3.0**	**4.9±2.2**	**361±10.4**
	BCD	163.6±49.7	319.7±44.8	34.8±26.8	41.8±16.9	217.8±45.3	81.8±35.5	5,694±184.2
	S-NPKL	59.8±13.5	224.0±106.2	41.9±6.5	44.0±8.06	180.7±82.7	49.1±23.1	1,955±352.2
	SDPLR	17.4±6.6	40.1±11.2	8.8±2.0	4.7±0.9	21.6±9.8	9.4±5.7	20,065±374.6
SHL	ADMM	**5.1±1.4**	**11.5±5.4**	**1.9±1.2**	**1.2±0.6**	**5.1±1.7**	**3.9±2.4**	**357±11.1**
	BCD	78.3±42.1	231.9±72.8	5.5±2.9	27.2±25.1	56.8±43.8	44.8±21.3	6,030±237.7
	S-NPKL	57.76±17.6	157.0±86.0	15.3±3.1	43.6±8.9	131.2±56.3	55.4±29.5	1,736±284.4
ℓ_1	ADMM	7.7±1.8	13.6±5.8	3.6±1.8	1.7±0.3	8.4±1.7	5.5±2.2	413±8.2

benchmark data sets[2] (Table 1) that have been commonly used for nonparametric kernel learning [5,6,14].

The proposed *Algorithm 1* (denoted "ADMM") is compared with the three solvers: block coordinate descent method (denoted "BCD") [6], simple nonparametric kernel learning (denoted "S-NPKL") [14] and low-rank SDP (denoted "SDPLR") [4]. Similar to ADMM, all are based on a rank-r approximation of $\mathbf{Z}$. Moreover, we follow [14,6] and set the rank of the kernel matrix to the largest r satisfying $r(r+1)/2 \leq m$, where m is the total number of constraints in Ω.

Results on the clustering accuracy and CPU time are shown in Tables 2 and 3, respectively. As can be seen, ADMM is more efficient than the other methods, while yielding comparable clustering accuracy.

5 Conclusion

In this paper, we proposed an efficient solver for nonparametric low-rank kernel learning. Using ADMM, it decouples the optimization problem into computationally inexpensive subproblems that involve only individual entries of the kernel matrix. Moreover, with an explicit low-rank factorization, it no longer needs to enforce the psd constraint that would lead to expensive eigen-decomposition in each iteration. Experimental results on a number of real-world data sets demonstrate that the proposed method is as accurate as directly solving the SDP, but is much faster than existing solvers.

Acknowledgment. This research was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region (Grant No. 614012), the National Natural Science Foundation of China (No. 61165012) , the Natural Science Foundations Fund (Grant No. 2011FZ074) and the Department of Education Fund (No. 2011Y305) of Yunnan Province.

References

1. Bach, F.R., Lanckriet, G.R.G., Jordan, M.I.: Multiple kernel learning, conic duality, and the SMO algorithm. In: Proceedings of the Twenty-First International Conference on Machine, Banff, Alberta, Canada, pp. 6–13 (July 2004)
2. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. Journal of Machine Learning Research 7, 2399–2434 (2006)
3. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning, 1–122 (2011)
4. Burer, S., Choi, C.: Computational enhancements in low-rank semidefinite programming. Optimization Methods and Software 21(3), 493–512 (2006)

[2] Downloaded from the UCI repository (`http://archive.ics.uci.edu/ml/`).

5. Hoi, S.C.H., Jin, R., Lyu, M.R.: Learning nonparametric kernel matrices from pairwise constraints. In: Proceedings of the Twenty-Fourth International Conference on Machine Learning, Corvallis, Oregon, USA, pp. 361–368 (June 2007)
6. Hu, E.-L., Wang, B., Chen, S.: BCDNPKL: Scalable non-parametric kernel learning using block coordinate descent. In: Proceedings of the Twenty-Eighth International Conference on Machine Learning, Bellevue, WA, USA, pp. 209–216 (June 2011)
7. Kulis, B., Sustik, M., Dhillon, I.: Low-rank kernel learning with Bregman matrix divergences. Journal of Machine Learning Research 10, 341–376 (2009)
8. Lanckriet, G.R.G., Cristianini, N., Bartlett, P., El Ghaoui, L., Jordan, M.I.: Learning the kernel matrix with semidefinite programming. Journal of Machine Learning Research 5, 27–72 (2004)
9. Li, Z., Liu, J.: Constrained clustering by spectral regularization. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, Miami, Florida, USA, pp. 421–428 (June 2009)
10. Li, Z., Liu, J., Tang, X.: Pairwise constraint propagation by semidefinite programming for semi-supervised classification. In: Proceedings of the Twenty-Fifth International Conference on Machine Learning, Helsinki, Finland, pp. 576–583 (July 2008)
11. Shang, F., Jiao, L.C., Wang, F.: Semi-supervised learning with mixed knowledge information. In: Proceedings of the Eighteenth Conference on Knowledge Discovery and Data Mining, Beijing, China, pp. 732–740 (August 2012)
12. Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Constrained K-means clustering with background knowledge. In: Proceedings of the Eighteenth International Conference on Machine Learning, Williamstown, MA, USA, pp. 577–584 (June 2001)
13. Xu, Y., Yin, W., Wen, Z., Zhang, Y.: An alternating direction algorithm for matrix completion with nonnegative factors. Technical Report TR11-03, Department of Computational and Applied Mathematics, Rice University (2011)
14. Zhuang, J., Tsang, I.W., Hoi, S.C.H.: A family of simple non-parametric kernel learning algorithms. Journal of Machine Learning Research 12, 1313–1347 (2011)