Learning Nonparametric Kernel Matrices from Pairwise Constraints

Steven C. H. Hoi[†] CHHOI@CSE.CUHK.EDU.HK
Rong Jin[‡] RONGJIN@CSE.MSU.EDU
Michael R. Lyu[†] LYU@CSE.CUHK.EDU.HK
[†]Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong
[‡]Department of Computer Science and Engineering, Michigan State University, USA

Abstract

Many kernel learning methods have to assume parametric forms for the target kernel functions, which significantly limits the capability of kernels in fitting diverse patterns. Some kernel learning methods assume the target kernel matrix to be a linear combination of parametric kernel matrices. This assumption again importantly limits the flexibility of the target kernel matrices. The key challenge with nonparametric kernel learning arises from the difficulty in linking the nonparametric kernels to the input patterns. In this paper, we resolve this problem by introducing the graph Laplacian of the observed data as a regularizer when optimizing the kernel matrix with respect to the pairwise constraints. We formulate the problem into Semi-Definite Programs (SDP), and propose an efficient algorithm to solve the SDP problem. The extensive evaluation on clustering with pairwise constraints shows that the proposed nonparametric kernel learning method is more effective than other state-of-the-art kernel learning techniques.

1. Introduction

Kernel-based learning methods (Vapnik, 1998; Schölkopf & Smola, 2002) have been studied extensively in the machine learning community due to their outstanding performance in many real-world applications. One of the key issues with all the kernel-based learning methods is how to identify the appropriate kernel function or kernel matrix that is consistent with the characteristics of data. Recently, a number of algorithms have been proposed to automatically learn the kernel function/matrix from the labeled examples (Cristianini et al., 2001; Kondor & Lafferty, 2002; Lanckriet et al., 2004; Chapelle et al., 2003; Zhu et al., 2005; Hoi et al., 2006; Kulis et al., 2006).

Most recent studies of kernel learning focus on semi-supervised learning (Chapelle et al., 2003; Zhang & Ando, 2005; Kulis et al., 2006), in which the kernels are learned from a mixture of labeled and unlabeled examples. To facilitate the learning procedure, many kernel learning algorithms assumed certain parametric forms for the target kernel functions. Example algorithms in this category include cluster kernels (Chapelle et al., 2003), diffusion kernels (Kondor & Lafferty, 2002), and the Gaussian random fields approach (Zhu et al., 2003). In addition, several studies (Lanckriet et al., 2004; Zhu et al., 2005; Cristianini et al., 2001; Hoi et al., 2006; Kulis et al., 2006) are devoted to nonparametric kernel learning that does not assume any parametric form of the kernel functions. Instead of learning the kernel functions from the labeled examples, these studies focus on learning appropriate kernel matrices for both labeled and unlabeled examples. Despite the claim of nonparametric kernel learning, however, most of them assume the target kernel matrix to be a linear combination of base matrices that are constructed beforehand. For instance, in (Zhu et al., 2005; Hoi et al., 2006), these base matrices are constructed by the principal eigenvectors of the observed data. These approaches significantly limit the choice of the kernel matrices, and often the performance of kernel learning depends on the choice of the base kernels.

The key challenge of nonparametric kernel learning arises from the difficulty in linking the input patterns of examples to the nonparametric kernel matrix. To learn the kernel matrix in a nonparametric setup, it is often convenient to cast the kernel learning problem into an optimization problem that includes the entire kernel matrix as a single variable. As a result, the dependency of kernel matrices on the input patterns of

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007.

examples is difficult to be explored directly within this framework. In order to resolve this problem, we propose a novel method for nonparametric kernel learning. Specifically, we first measure the pairwise similarity between any two examples, and construct a graph Laplacian matrix based on the pairwise similarities. This graph Laplacian matrix is then used to regularize the nonparametric kernel learning. By minimizing the regularizer that is based on the graph Laplacian, we enforce the learned kernel matrix to be consistent with the structure of both labeled and unlabeled examples, and therefore bridge the gap between the input patterns of examples and the nonparametric kernel matrix. Since the related optimization problem is formulated into a Semi-Definite Programming (SDP) problem, another challenge faced by this work is how to solve the SDP problem with a large number of unlabeled data. To address the computation problem, we propose to solve the dual problem first, and then derive the solution for the primal problem through the Karush-Kuhn-Tucker (KKT) conditions (Boyd & Vandenberghe, 2003). Furthermore, an efficient algorithm that is similar to the strategy used in Sequential Minimal Optimization (SMO) (Platt, 1999) is proposed to solve the dual problem.

It is important to note that our work is closely related to the previous work on learning low rank kernel matrix from pairwise constraints (Kulis et al., 2006). Similar to our work, the work in (Kulis et al., 2006) looks for the nonparametric kernel matrix that is consistent with the given pairwise constraints. The key difference between these two approaches is that the approach in (Kulis et al., 2006) requires an initial low-rank kernel matrix to begin with. In contrast, our proposed approach allows for the pairwise similarity matrix to be indefinite. This relaxation is important because many similarity measurements are hand crafted by domain experts and are likely to violate the Mercer's conditions (Schölkopf & Smola, 2002). We also will show empirically that the proposed method is more effective than the approach in (Kulis et al., 2006).

The rest of this paper is organized as follows. Section 2 outlines our methodology of learning nonparametric kernel matrices from pairwise constraints. Section 3 presents an efficient algorithm to solve our optimization. Section 4 presents the empirical evaluation of the proposed algorithm. Section 5 concludes this work.

2. Problem Formulation

Let the entire data collection be denoted by $\mathcal{U} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$ where each data point $\mathbf{x}_i \in \mathbb{R}^d$ is a vector of length d. Let $f(\mathbf{x}, \mathbf{x}')$ be the similarity function that measures the similarity between any two data points $\mathbf{x}$ and $\mathbf{x}'$. Let $S \in \mathbb{R}^{N \times N}$ be the similarity matrix where each element $S_{i,j} = f(\mathbf{x}_i, \mathbf{x}_j)$ represents the similarity between any two data points $\mathbf{x}_i$ and $\mathbf{x}_j$. Note that the similarity function $f(\cdot, \cdot)$ does not have to be a kernel function that satisfies the Mercer's condition. For the convenience of presentation, we assume $S_{i,i} = 0$ for all the examples. Based on the similarity matrix S, we then construct the graph Laplacian L as follows:

$$L = (1+\delta)I - D^{-1/2} S D^{-1/2}$$

where $D = \mathrm{diag}(d_1, d_2, \ldots, d_N)$ is a diagonal matrix with the diagonal element defined as $d_i = \sum_{j=1}^{N} f(\mathbf{x}_i, \mathbf{x}_j)$. A small $\delta > 0$ is used as the smoothing parameter to prevent the matrix L from being singular. Let $\mathcal{S}$ denote the collection of the data pairs that share the same classes, and $\mathcal{D}$ denote the collection of the data pairs that share different classes. We construct a matrix $T \in \mathbb{R}^{N \times N}$ to represent all the pairwise constraints, i.e.,

$$T_{i,j} = \begin{cases} +1 & (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{S} \\ -1 & (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{D} \\ 0 & \text{otherwise} \end{cases}$$

Given the matrix T, our goal is to identify a kernel matrix $Z \in \mathbb{R}^{N \times N}$ that is consistent with all the pairwise constraints in T. Following the principle of maximum margin theory (Vapnik, 1998), we consider the following optimization problem for learning the kernel matrix Z:

$$\underset{Z = V^\top V}{\arg\min} \ \|V\|_2^2 + c \sum_{(i,j) \in (\mathcal{S} \cup \mathcal{D})} \max\left(0, 1 - T_{i,j} Z_{i,j}\right) \quad (1)$$

In the above formula, we introduce a matrix $V = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N)$ that can be viewed as a new data representation for all the examples. In particular, $\mathbf{v}_i$, a column vector of matrix V, can be viewed as a representation vector for example $\mathbf{x}_i$. Thus, by minimizing the term $\|V\|_2^2 = \sum_{i=1}^{n} \|\mathbf{v}_i\|_2^2$, we essentially look for a sparse data representation. Note that unlike the non-negative matrix factorization (Lee & Seung, 2000), we do not specify the rank for V, which on one hand broadens the choice of the matrix V, and on the other hand avoids the non-convex optimization problem. Constant c is introduced in (1) to balance the sparseness of the solution (i.e., the first term) against the classification error (i.e., the second term).

However, the problem with the formula in (1) is that it is independent from the input patterns of examples. In order to introduce the dependency of input patterns into the kernel matrix Z, we modify the regularizer

$\|V\|_2^2$ by measuring the inconsistency between the new data representation V and the similarity matrix S, i.e.,

$$l(V,S) = \sum_{i,j=1}^{n} \frac{S_{i,j}}{\sqrt{d_i d_j}} \|\mathbf{v}_i - \mathbf{v}_j\|_2^2 = \mathrm{tr}(VLV^\top) \quad (2)$$

The above regularizer is similar to the manifold regularization approach (Belkin et al., 2004), where the graph Laplacian is used to regularize the classification results. Using the regularizer in (2), we modify the problem in (1) as follows:

$$\arg\min_{Z=V^\top V} \mathrm{tr}(VLV^\top) + c \sum_{(i,j)\in(\mathcal{S}\cup\mathcal{D})} \max(0, 1 - T_{i,j}Z_{i,j})$$

We can simplify the above formulism by using the fact $\mathrm{tr}(VLV^\top) = \mathrm{tr}(LV^\top V) = \mathrm{tr}(LZ)$. This results in the following formulism

$$\begin{aligned} \arg\min_{Z,\epsilon} \quad & \sum_{i,j=1}^{N} L_{i,j}Z_{i,j} + c \sum_{(i,j)\in(\mathcal{S}\cup\mathcal{D})} \epsilon_{i,j} \quad (3) \\ \text{s. t.} \quad & \forall (i,j) \in (\mathcal{S}\cup\mathcal{D}),\ T_{i,j}Z_{i,j} \geq 1 - \epsilon_{i,j}, \epsilon_{i,j} \geq 0 \\ & Z \succeq 0 \end{aligned}$$

It is not difficult to see that the problem in (3) is a semi-definitive programming problem, and therefore can be solved by using the standard software package, such as SeDuMi/YALMIP (Löfberg, 2004). However, it could be computationally expensive to solve the problem in (3) directly, if the number of examples is large. In the following, we present an efficient algorithm, whose computational complexity only depends on the number of pairwise constraints and is independent of the number of examples.

3. An Efficient Dual Algorithm

3.1. The Dual Problem and KKT Conditions

In this section, we present our solution of a fast dual solution. In particular, we improve the computational efficiency of SDP in the following two aspects:

(i) Instead of solving the primal problem directly, we solve the dual problem of the SDP problem in (3) first. Then, the solution to the primal problemin (3) can be recovered by utilizing the KKT conditions (Boyd & Vandenberghe, 2003).

(ii) To improve the efficiency in solving the dual problem, we present an algorithm that is similar to the approach of Sequential Minimal Optimization (SMO) (Platt, 1999). The key idea is to decompose a multi-variate SDP problem into a number of univariate SDP problems that possess closed form solutions.

To obtain the dual problem of (3), we first construct the Lagrangian function as follows:

$$\begin{aligned} \mathcal{L} \quad = \quad & \sum_{i,j=1}^{N} L_{i,j}Z_{i,j} + c \sum_{(i,j)\in(\mathcal{S}\cup\mathcal{D})} \epsilon_{i,j} \\ & - \sum_{(i,j)\in(\mathcal{S}\cup\mathcal{D})} Q_{i,j}\left(T_{i,j}Z_{i,j} - 1 + \epsilon_{i,j}\right) \\ & - \sum_{(i,j)\in(\mathcal{S}\cup\mathcal{D})} \xi_{i,j}\epsilon_{i,j} - \mathrm{tr}(MZ) \end{aligned}$$

where $Q_{i,j} \geq 0$ and $M \succeq 0$ are the Lagrangian multipliers. By setting the first order derivative of $\mathcal{L}$, we have

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \epsilon_{i,j}} \quad = \quad & c - Q_{i,j} - \xi_{i,j} = 0 \rightarrow Q_{i,j} \leq c \\ \frac{\partial \mathcal{L}}{\partial Z_{i,j}} \quad = \quad & L_{i,j} - Q_{i,j}T_{i,j} - M_{i,j} = 0 \rightarrow L \succeq Q \otimes T \end{aligned}$$

where the operator $\otimes$ stands for the element-wise multiplication between two matrices. Hence, the dual form for the problem in (3) is written as follows:

$$\begin{aligned} \arg\max_{Q} \quad & \sum_{(i,j)\in\mathcal{S}} Q_{i,j} + \sum_{(i,j)\in\mathcal{D}} Q_{i,j} \\ \text{s. t.} \quad & 0 \leq Q_{i,j} \leq c,\ \forall (i,j) \in (\mathcal{S}\cup\mathcal{D}) \\ & L \succeq Q \otimes T \quad (4) \end{aligned}$$

Note that the number of variables in the above problem is equal to the number of pairwise constraints. Therefore, the above optimization problem can be solved efficiently when the number of pairwise constraints is small.

After solving the dual problem in (4), the next question is to recover the kernel matrix Z for the primal problem in (3). To efficiently compute Z, we explore the following KKT conditions

$$M = L - Q \otimes T, \qquad MZ = 0$$

The first condition in the above gives the solution for the Lagrangian multiplier M. From the second condition, i.e., $MZ = 0$, we know that each column of the matrix Z is the eigenvector of the matrix M for the zero eigenvalue. Let $\{\mathbf{u}_i, i = 1, 2, \ldots, K\}$ be the eigenvectors of the matrix M whose eigenvalues are zero. Then, the kernel matrix Z can be expressed in the following form:

$$Z \quad = \quad (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K)A = UA \quad (5)$$

where $U = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K)$ and A is a matrix of size $K \times n$. Furthermore, since both Z and M are symmetric matrices, we will also have $ZM = 0$. Following

the above logic, we have $Z = \tilde{A}U$. Putting these two factors together, we obtain the following expression:

$$Z = UBU^\top$$

where B is a positive semi-definitive matrix of size $K \times K$. Thus, instead of solving Z directly, by using the dual formulism, we can solve B first, and then compute Z based on the estimated B. It is important to note that $K \leq |\mathcal{S}| + |\mathcal{D}| + 1$. This is because the number of non-zero elements in matrix $Q \otimes T$ is no more than $|\mathcal{S}| + |\mathcal{D}|$. Therefore, the rank of matrix $Q \otimes T$ is no more than $|\mathcal{S}| + |\mathcal{D}|$. In the meantime, the rank of the graph Laplacian L is $n - 1$ (the minimum eigenvalue is 0). Since $M = L - Q \otimes T$, the rank of M is no less than $n - 1 - |\mathcal{S}| + |\mathcal{D}|$, which implies the number of eigenvectors for zero eigenvalue is no more than $|\mathcal{S}| + |\mathcal{D}| + 1$. Thus, given a number of pairwise constraints, i.e. $|S| + |D| \ll n$, the size of the matrix B, i.e., K^2, is significantly smaller than that of the matrix Z, n^2. Hence, solving the matrix B could be computationally much more efficient than solving matrix Z.

Substituting the expression $Z = UBU^\top$ into the primal problem, we have

$$\begin{aligned} \arg\min_{B \succeq 0} \quad & \sum_{i,j=1}^{N} L_{i,j} Z_{i,j} + c \sum_{(i,j) \in (\mathcal{S} \cup \mathcal{D})} \epsilon_{i,j} \\ \text{s. t.} \quad & \forall (i,j) \in (\mathcal{S} \cup \mathcal{D}),\ T_{i,j} Z_{i,j} \geq 1 - \epsilon_{i,j}, \epsilon_{i,j} \geq 0 \\ & Z = UBU^\top \end{aligned}$$

or equivalently,

$$\begin{aligned} \arg\min \quad & \operatorname{tr}(BU^\top LU) + c \sum_{(i,j) \in (\mathcal{S} \cup \mathcal{D})} \epsilon_{i,j} \qquad (6) \\ \text{s. t.} \quad & \forall (i,j) \in (\mathcal{S} \cup \mathcal{D}),\ T_{i,j} \mathbf{u}_i^\top B \mathbf{u}_j \geq 1 - \epsilon_{i,j} \\ & \forall (i,j) \in (\mathcal{S} \cup \mathcal{D}), \epsilon_{i,j} \geq 0 \\ & B \succeq 0 \end{aligned}$$

where $\mathbf{u}_i$ corresponds to the ith row of the matrix U.

3.2. An Efficient Algorithm for Solving the Dual Problem

Although the number of variables in (4) is small when the number of constraints is not large, solving the SDP problem in (4) could still be expensive when the number of examples is large. This is because the LMI constraint $L \succeq Q \otimes T$ involves an $n \times n$ matrix. Here, we present an approach that is similar to the SMO algorithm for SVM (Platt, 1999). In particular, we solve the SDP problem iteratively. In each iteration, we select one variable $Q_{i,j}$, and try to solve the SDP problem with a single variable. As we will demonstrate below, the SDP problem in (4) with a single variable could be solved analytically.

Let $\tilde{Q}$ denote the solution Q that is obtained so far. Let $Q_{k,l}$ denote the variable that is selected for the current iteration. Then, the corresponding optimization problem for variable $Q_{k,l}$ is written as follows:

$$\begin{aligned} \arg\max_{Q_{k,l}} \quad & Q_{k,l} \qquad (7) \\ \text{s. t.} \quad & 0 \leq Q_{i,j} \leq c,\ A^{k,l} - T_{k,l} Q_{k,l} I^{k,l} \succeq 0 \end{aligned}$$

where matrix $A^{k,l}$ is defined as

$$A^{k,l} = L - (\tilde{Q} - \tilde{Q}_{k,l} I^{k,l}) \otimes T. \qquad (8)$$

Note $I^{k,l}$ is a $n \times n$ matrix and is defined as

$$[I^{k,l}]_{i,j} = \begin{cases} 1 & (k = i \text{ and } l = j) \\ 1 & (k = j \text{ and } l = i) \\ 0 & \text{otherwise} \end{cases}$$

Similar to the SMO algorithm for SVM, it turns out that the problem in (7) has a closed form solution. To see this, we rearrange the rows and the columns of the matrix $A^{k,l} - T_{k,l} Q_{k,l} I^{k,l}$ by moving the kth and the lth row/column to the first and the second row/column. As a result, we can write the matrix $A^{k,l} - T_{k,l} Q_{k,l} I^{k,l}$ as follows:

$$\begin{pmatrix} A_1 & W \\ W^\top & A_2 \end{pmatrix}$$

where $W \in \mathbb{R}^{(n-2) \times 2}$, $A_2 \in \mathbb{R}^{(n-2) \times (n-2)}$, and A_1 is

$$A_1 = \begin{pmatrix} L_{k,k} & L_{k,l} - Q_{k,l} T_{k,l} \\ L_{k,l} - Q_{k,l} T_{k,l} & L_{l,l} \end{pmatrix}$$

Using the Schur complement (Boyd & Vandenberghe, 2003), the matrix inequality $A^{k,l} - T_{i,j} Q_{k,l} I_{k,l} \succeq 0$ is equivalent to $A_1 \succeq W^\top A_2^{-1} W$. Let us denote $W^\top A_2^{-1} W$ by

$$W^\top A_2^{-1} W \equiv G = \begin{pmatrix} G_{1,1} & G_{1,2} \\ G_{2,1} & G_{2,2} \end{pmatrix} \qquad (9)$$

Then, $A_1 \succeq V^\top A_2^{-1} V$ is equivalent to

$$\begin{pmatrix} L_{k,k} - G_{1,1} & L_{k,l} - Q_{k,l} T_{k,l} - G_{1,2} \\ L_{k,l} - Q_{k,l} T_{k,l} - G_{1,2} & L_{l,l} - G_{2,2} \end{pmatrix} \succeq 0$$

The necessary and sufficient conditions for the 2×2 matrix to be positive semi-definite are

1. $L_{k,k} - G_{1,1} \geq 0$,
2. $L_{l,l} - G_{2,2} \geq 0$, and
3. the determinant of the above matrix is non-negative, i.e., $(G_{1,2} + T_{k,l} Q_{k,l} - L_{k,l})^2 \leq (L_{k,k} - G_{1,1})(L_{l,l} - G_{2,2})$.

Since the first two conditions are independent of the variable $Q_{k,l}$, we only include the third condition as the part of the optimization problem. Therefore, the optimization problem in (7) is eventually equivalent to the following optimization problem:

$$\begin{aligned} \max_{Q_{k,l}} \quad & Q_{k,l} \\ \text{s.t.} \quad & 0 \leq Q_{k,l} \leq c \\ & |G_{1,2} + T_{k,l}Q_{k,l} - L_{k,l}| \leq \mu_{k,l} \end{aligned} \tag{10}$$

where $\mu_{k,l} = \sqrt{(L_{k,k} - G_{1,1})(L_{l,l} - G_{2,2})}$. The optimal solution for $Q_{k,l}$ is

$$Q_{k,l} = \min\left(c, \mu_{k,l} - T_{k,l}G_{1,2} + T_{k,l}L_{k,l}\right)$$

Evidently, the main computational cost lies in computing the matrix G, which requires computing the matrix inverse A_2^{-1}. To avoid the cost of computing the matrix inverse for large matrices, we can convert the computation into an optimization problem. Let $W = (\mathbf{w}_1, \mathbf{w}_2)$. Then, according to the definition of G in (9), we have

$$G_{a,b} = \mathbf{w}_a^\top A_2^{-1}\mathbf{w}_b$$

where indices a and b could be either 1 or 2. The quantity $\mathbf{w}_a^\top A_2^{-1}\mathbf{w}_a$ is indeed the maximum value for the following optimization problem:

$$\max_{\mathbf{x}} \quad -\mathbf{x}^\top A_2\mathbf{x} + 2\mathbf{w}_a^\top\mathbf{x}$$

The above optimization can be solved efficiently using the conjugate gradient methods (Shewchuk, 1994). To compute $\mathbf{w}_a^\top A_2^{-1}\mathbf{w}_b$, we can first compute $(\mathbf{w}_a + \mathbf{w}_b)^\top A_2^{-1}(\mathbf{w}_a + \mathbf{w}_b)$, $\mathbf{w}_a^\top A_2^{-1}\mathbf{w}_a$, and $\mathbf{w}_b^\top A_2^{-1}\mathbf{w}_b$ using the conjugate gradient method. Then, $\mathbf{w}_a^\top A_2^{-1}\mathbf{w}_b$ can be computed as follows:

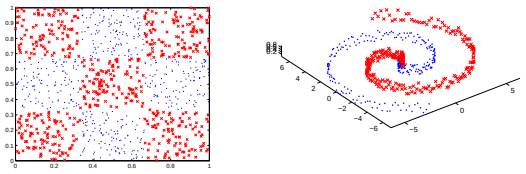
$$\mathbf{w}_a^\top A_2^{-1}\mathbf{w}_b = \frac{1}{2}\Big((\mathbf{w}_a + \mathbf{w}_b)^\top A_2^{-1}(\mathbf{w}_a + \mathbf{w}_b) - \mathbf{w}_a^\top A_2^{-1}\mathbf{w}_a - \mathbf{w}_b^\top A_2^{-1}\mathbf{w}_b\Big)$$

4. Experimental Results

To evaluate the proposed algorithm for nonparametric kernel learning, we engage data clustering as the application. More specifically, a kernel matrix is learned from the pairwise constraints using the proposed algorithm. The learned kernel matrix will then be employed by the K-means algorithms to cluster examples. The clustering accuracy of the K-means algorithm will be used to evaluate the quality of the estimated kernel matrix. In our empirical study, nine datasets are applied, including two artificial datasets and seven datasets from the UCI machine learning repository. Table 1 summarizes the information about the nine datasets, and Fig. 1 shows the data distribution of the two artificial datasets.

Table 1. The nine datasets used in our experiments. The first two are the artificial datasets and the others datasets are from the UCI machine learning repository.

Dataset	#Classes	#Instances	#Features
Chessboard	2	100	2
Double-Spiral	2	100	3
Glass	6	214	9
Heart	2	270	13
Iris	3	150	4
Protein	6	116	20
Sonar	2	208	60
Soybean	4	47	35
Wine	3	178	12



(a) Chessboard (b) Double-Spiral

Figure 1. Two artificial datasets used in our experiment.

We compare our proposed kernel learning method with the following state-of-art clustering approaches:

- **K-means**: This is the regular K-means algorithm using the default Euclidean metric. This approach does not utilize any pairwise constraints, and is engaged as the reference approach.
- **Constrained K-means**: This approach modifies the standard K-means clustering algorithm by enforcing each pair of data points in the equivalence constraints to be assigned to the same clusters (Wagstaff et al., 2001).
- **Constrained K-means + RCA**: This approach improves the constrained K-means algorithm by using the metric that is learned by Relevant Component Analysis (RCA) (Bar-Hillel et al., 2005).
- **Constrained K-means + Xing**: This approach improves the constrained K-means algorithm by the data distance metric that is learned from the pairwise constraints using the convex programming approach proposed in (Xing et al., 2002).
- **Constrained Kernel K-means + RBF**: This approach improves the kernel K-means algorithm

by enforcing the equivalence constraints during the clustering. In this method, the kernel matrix is a standard RBF kernel, whose parameter is chosen via cross-validation.

- **Constrained Kernel K-means + MPK**: This constrained kernel K-means algorithm uses a kernel matrix that is formed by a linear combination of multiple parametric kernels, in which the weights are learned by a kernel target alignment method (Lanckriet et al., 2004). In our experiment, we adopt a combination of three standard kernels, i.e., Linear, Polynomial and RBF.

- **Constrained Kernel K-means + LRK**: This constrained kernel K-means algorithm uses a kernel matrix learned by a low-rank kernel (LRK) learning technique in (Kulis et al., 2006). We slightly modify the original algorithm to make it suitable for learning with pairwise constraints. [1]

- **Constrained Kernel K-means + NPK**: This approach improves the constrained kernel K-means algorithm using the kernel matrix that is learned by our proposed nonparametric kernel (NPK) learning algorithm. For the graph Laplacian, we use a standard method, i.e., by calculating the distance matrix by Euclidean distance, then constructing the adjacency matrix with 5 nearest neighbors, and finally normalizing the graph to achieve the final Laplacian matrix.

We adopt the metric of clustering accuracy (Xing et al., 2002) for evaluating the clustering results. In particular, this metric measures the percentage of data pairs that are correctly clustered together. It is defined as follows

$$Accuracy = \sum_{i>j} \frac{\mathbf{1}\{\mathbf{1}\{c_i = c_j\} = \mathbf{1}\{\hat{c}_i = \hat{c}_j\}\}}{0.5n(n-1)} , \quad (11)$$

where $\mathbf{1}\{\cdot\}$ is the indicator function that outputs 1 when the input argument is true and 0 otherwise. c_i and $\hat{c}_i$ denote the true cluster memberhip and the predicted cluster membership of the ith data point, respectively. n is the number of examples in the dataset.

To examine the performance of the clustering algorithms in a full spectrum, we consider two different scenarios that are similar to the settings used in the previous study (Xing et al., 2002). In the first scenario, each algorithm is given with "**little**" side-information. Specifically, we randomly select a number of data pairs for pairwise constraints, such that the number of resulting connected components is roughly equal to 90% of the dataset size. In the second scenario, each algorithm is given with "**much**" side-information, where we randomly select a number of data pairs for pairwise constraints, such that the number of resulting connected components is roughly equal to 70% of the dataset size. Note that the number of resulting connected components is equal to the dataset size when no any positive constraints are provided. To avoid the unbalance distribution between equivalence constraints and inequivalence constraints, the labeled pairs are chosen such that one half of them are equivalence constraints and the other are inequivalence constraints.

Each clustering experiment is run 20 times with multiple restarts, and parameter c in the NPK algorithm is fixed to 1 for all experiments. All clustering algorithms are given with the same random set of initial cluster centers in each experimental trial. Fig. 2 shows the average clustering accuracy of the eight different clustering algorithms for the nine datasets. Each diagram in Fig. 2 corresponds to a different dataset, and is consisted of two parts: the eight bars on the left side correspond to the clustering results with "little" side-information, and the eight bars on the right correspond to the clustering results with "much" side-information.

We can make several observations. First, we observe that for most cases, the clustering accuracy of the K-means algorithm is improved by incorporating the metric learned from pairwise constraints. Second, compared to the other metric learning algorithms, the proposed NPK algorithm is significantly more effective than the others in improving the clustering accuracies in most cases. The most impressive cases are dataset "Double-Spiral" and "Sonar", in which the clustering accuracy of the K-means algorithm is improved from about 50% to around 90% and 70%, respectively. In contrast, the other four metric learning algorithms can only improve the clustering accuracy with no more than 5% for these two datasets. Last, compared to the multi-parametric kernel learning and the low-rank kernel learning approaches, our NPK method performs significantly better in most cases.

Finally, we empirically evaluate the computational efficiency of the NPK algorithm by two different approaches, i.e., (1) NPK-SDP: the SDP algorithm that is solved in primal, and (2) NPK-SMO: the SMO-like algorithm that is solved in dual. To examine the scalability, we evaluate their time performance of differ-

[1] The original algorithm requires a low-rank kernel matrix as the input matrix, for which the authors suggest a random matrix. However, our empirical study found that LRK worked poorly for clustering settings when using random input matrices. Consequently, we adopt an RBF kernel as the input matrix that empirically achieves better performances.

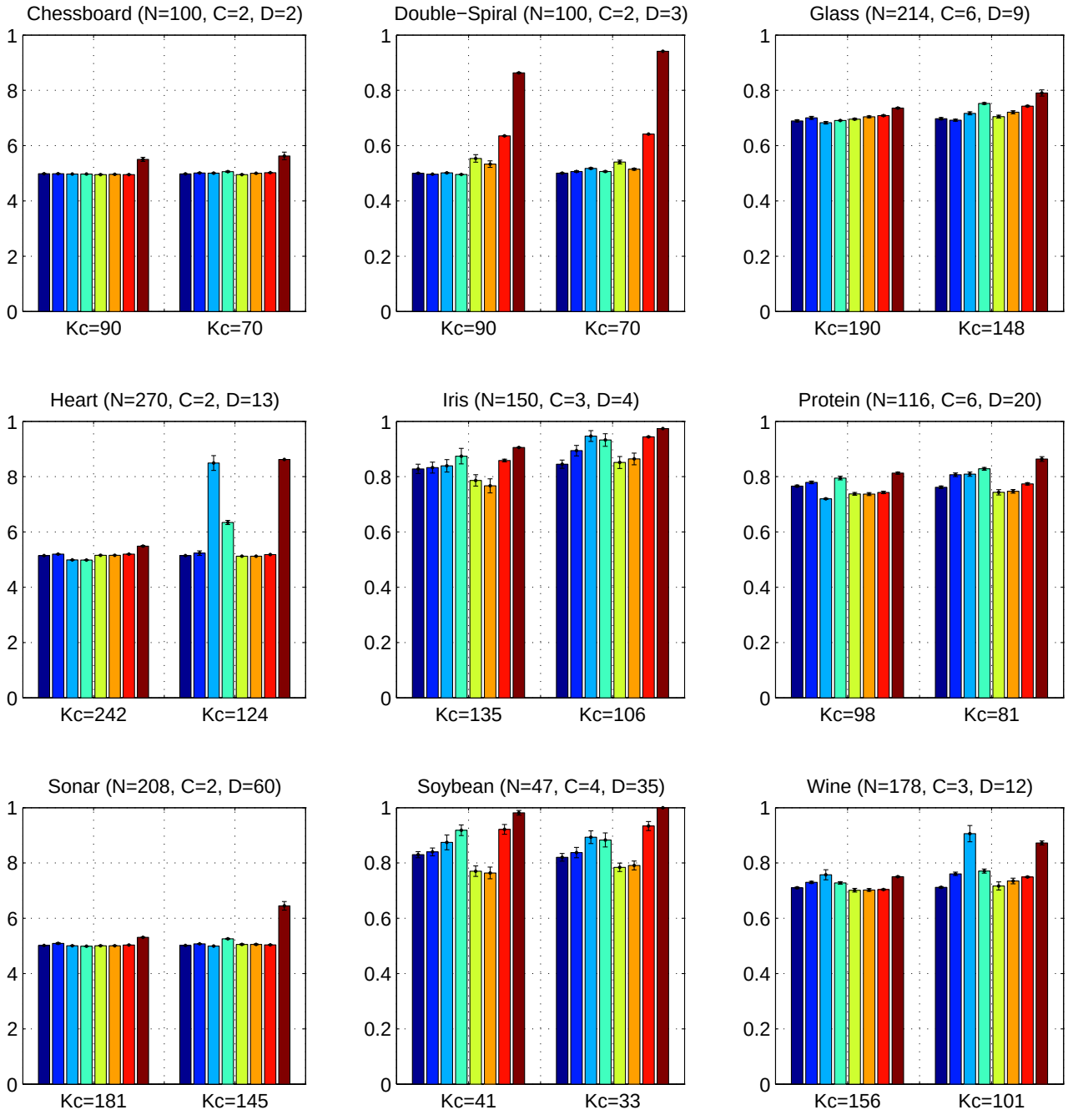


Figure 2. The clustering accuracy of the K-means algorithm using different data metrics that are learned from pairwise constraints. In each individual diagram, the eight bars on the left side correspond to the experiment with a small number of pairwise constraints ("little" side-information), and the eight bars on the right correspond to the experiment a large number of pairwise constraints ("much" side-information). From left to right, the eight bars are respectively K-means, Constrained K-means, Constrained K-means + RCA metric, Constrained K-means + Xing's metric, Constrained Kernel K-means + RBF kernel, Constrained Kernel K-means + Multi-Parametric Kernel (MPK), Constrained Kernel K-means + Low-Rank Kernel (LRK), and Constrained Kernel K-means + Non-Parametric Kernel (NPK). The value of Kc denotes the number of connected components formed with respect to a set of pairwise constraints. Standard error bars are also shown in the figure.

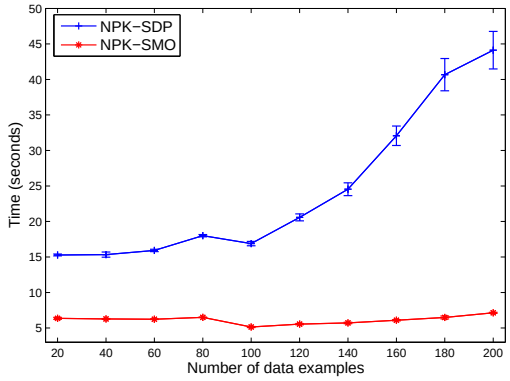


Figure 3. Time cost of different numbers of data examples. The number of pairwise constraints is fixed to 100.

ent dataset sizes. Specially, we generate a number of "Double-Spiral" datasets with sizes from 20 to 200. Fig. 3 shows the result that is averaged over 10 trials, in which the number of pairwise constraints is fixed to 100 for each trial. Note that we cut the data size to 200 because the NPK-SDP algorithm is computationally prohibited for a dataset with more than 200 examples. We clearly see that the computational time of the NPK-SDP method increases dramatically when the dataset size is greater than 100. On the contrary, the computational time of the NPK-SMO algorithm remains almost unchanged when the number of examples is increased. This result shows that the proposed NPK-SMO algorithm is computationally efficient.

5. Conclusion

In this paper we proposed a novel algorithm that learns nonparametric kernel matrices from given pairwise constraints. We formulated the nonparametric kernel learning problem into a Semi-Definite Programming problem. We then presented an efficient algorithm that first solves the dual problem of the related SDP problem and then infers the solution for the primal problem using the KKT condition. Furthermore, we presented an SMO-like algorithm to efficiently solve the dual problem. We studied the proposed method for nonparametric kernel learning with application to clustering with pairwise constraints. The encouraging empirical results show that our proposed method is more effective than the traditional approaches for clustering with side-information.

Acknowledgments

The work described in this paper was fully supported by a grant from the Shun Hing Institute of Advanced Engineering (SHIAE), CUHK.

References

Bar-Hillel, A., Hertz, T., Shental, N., & Weinshall, D. (2005). Learning a mahalanobis metric from equivalence constraints. *JMLR*, *6*, 937–965.

Belkin, M., Matveeva, I., & Niyogi, P. (2004). Regularization and semi-supervised learning on large graphs. *Annual Conference on Learning Theory (COLT2004)*.

Boyd, S., & Vandenberghe, L. (2003). *Convex optimization*. Cambridge University Press.

Chapelle, O., Weston, J., & Schölkopf, B. (2003). Cluster kernels for semi-supervised learning. *NIPS2003*.

Cristianini, N., Shawe-Taylor, J., Elisseeff, A., & Kandola, J. (2001). On kernel-target alignment. *NIPS2001*.

Hoi, S. C. H., Lyu, M. R., & Chang, E. Y. (2006). Learning the unified kernel machines for classification. *KDD2006* (pp. 187–196). Philadelphia, PA, US.

Kondor, R., & Lafferty, J. (2002). Diffusion kernels on graphs and other discrete structures. *ICML2002*.

Kulis, B., Sustik, M., & Dhillon, I. S. (2006). Learning low-rank kernel matrices. *ICML2006* (pp. 505–512).

Lanckriet, G., Cristianini, N., Bartlett, P., Ghaoui, L. E., & Jordan, M. (2004). Learning the kernel matrix with semi-definite programming. *JMLR*, *5*, 27–72.

Lee, D. D., & Seung, H. S. (2000). Algorithms for non-negative matrix factorization. *NIPS'13* (pp. 556–562).

Löfberg, J. (2004). YALMIP: A toolbox for modeling and optimization in MATLAB. *The CACSD Conference*.

Platt, J. C. (1999). Fast training of support vector machines using sequential minimal optimization. *Advances in kernel methods: support vector learning*, 185–208.

Schölkopf, B., & Smola, A. (2002). *Learning with kernels*. MIT Press.

Shewchuk, J. (1994). *An introduction to the conjugate gradient method without the agonizing pain*Technical Report CMU-CS-94-125). Carnegie Mellon University.

Vapnik, V. N. (1998). *Statistical learning theory*. John Wiley & Sons.

Wagstaff, K., Cardie, C., Rogers, S., & Schroedl, S. (2001). Constrained k-means clustering with background knowledge. *ICML2000* (pp. 577–584).

Xing, E. P., Ng, A. Y., Jordan, M. I., & Russell, S. (2002). Distance metric learning with application to clustering with side-information. *NIPS2002*.

Zhang, T., & Ando, R. K. (2005). Analysis of spectral kernel design based semi-supervised learning. *NIPS2005*.

Zhu, X., Ghahramani, Z., & Lafferty, J. (2003). Semi-supervised learning using gaussian fields and harmonic functions. *ICML2003*.

Zhu, X., Kandola, J., Ghahramani, Z., & Lafferty, J. (2005). Nonparametric transforms of graph kernels for semi-supervised learning. *NIPS2005*.