Learning a Kernel for Multi-Task Clustering

Quanquan Gu, Zhenhui Li and Jiawei Han
Department of Computer Science
University of Illinois at Urbana-Champaign
{qgu3,zli28,hanj}@illinois.edu

Abstract

Multi-task learning has received increasing attention in the past decade. Many supervised multi-task learning methods have been proposed, while unsupervised multi-task learning is still a rarely studied problem. In this paper, we propose to learn a kernel for multi-task clustering. Our goal is to learn a Reproducing Kernel Hilbert Space, in which the geometric structure of the data in each task is preserved, while the data distributions of any two tasks are as close as possible. This is formulated as a unified kernel learning framework, under which we study two types of kernel learning: nonparametric kernel learning and spectral kernel design. Both types of kernel learning can be solved by linear programming. Experiments on several cross-domain text data sets demonstrate that kernel k-means on the learned kernel can achieve better clustering results than traditional single-task clustering methods. It also outperforms the newly proposed multi-task clustering method.

Introduction

Multi-task learning (Caruana 1997) has received increasing interest in the past decade. It is based on the philosophy that learning multiple tasks together may achieve better generalization ability than learning these tasks individually, provided that the relation between these tasks are well utilized. For example, given the web pages from two universities, i.e., A and B, we aim to classify (cluster) the web pages of each university into four categories, e.g., student, faculty, staff and course. We refer to classifying (clustering) the web pages of each university as a task. Intuitively, these two tasks are related to each other, since the contents of the web pages are similar and the label spaces are the same. If we can use such kind of relationship between the two tasks, then classifying (clustering) them simultaneously may lead to better performance. One intuitive way is to combine the web pages from the two universities together, followed with traditional single-task learning approach. However, it is likely to result in poor performance, because University A and University B exhibit different characteristics and hence the distributions of the web pages from them are different. Thus the combined data violates the i.i.d. assumption of single-task learning. The fundamental problem in multi-task learning is how to characterize the task relation. Representative methods characterizing task relation include learning a shared subspace (Ando and Zhang 2005) (Argyriou, Evgeniou, and Pontil 2006), using the common prior of model parameters (Evgeniou and Pontil 2004) (Zhang, Ghahramani, and Yang 2005) (Zhang and yan Yeung 2010) and kernel methods (Micchelli and Pontil 2004) (Pan, Kwok, and Yang 2008) (Duan et al. 2009), etc.

The methods mentioned above are all supervised multi-task learning. Recently, multi-task clustering (Gu and Zhou 2009) was proposed, which clusters the data from related tasks simultaneously. In particular, it assumes that there is a common subspace shared by multiple related tasks, in which the data of all the tasks share an identical set of cluster centers. However, there is no principled way to choose the dimensionality of the shared subspace. More importantly, it does not explicitly consider the distributions of the tasks, which are essential to measure the relation between any two tasks (Pan, Kwok, and Yang 2008) (Duan et al. 2009).

In this paper, based on the above observation, we aim to explicitly exploit the distribution information of related tasks for multi-task clustering. We present a multi-task clustering framework based on kernel learning. The basic idea of our framework is to find a Reproducing Kernel Hilbert Space (RKHS), in which the mapped data distributions of the related tasks are close to each other. Hence single-task clustering algorithm such as kernel k-means can be applied to this RKHS. Through the learned RKHS, the knowledge of one task can be transferred to another, while there is no need to choose the dimensionality of this RKHS. Similar ideas have also been adopted in several cross-domain classification approaches (Pan, Kwok, and Yang 2008) (Duan et al. 2009). On the other hand, we hope that the geometric structure of the data in each task is preserved after feature mapping, since it is fundamental for better clustering performance (Shi and Malik 2000). This can be achieved via graph regularization (Chung 1997) in the RKHS. Both of the above two objectives are unified in a kernel learning framework. In particular, we study two kinds of kernel learning: nonparametric kernel learning (Lanckriet et al. 2004) (Hoi, Jin, and Lyu 2007) and spectral kernel design (Zhu et al. 2004), both of which can be solved by linear programming (Boyd and Vandenberghe 2004). Experiments on text data sets demonstrate that the proposed methods outperform the recently proposed multi-task clustering method as well as many state of the art single-task clustering approaches.

The Proposed Method

Problem Formulation

Suppose we are given m clustering tasks, each with a set of data points, i.e. $\mathcal{X}^k = \{\mathbf{x}_1^k, \mathbf{x}_2^k, \dots, \mathbf{x}_{n_k}^k\} \in \mathbb{R}^d, 1 \leq k \leq m$, where n_k is the number of data points in the k-th task and $n = \sum_{k=1}^m n_k$ is the total number of data points from all the tasks. The goal of multi-task clustering is to partition the data set $\mathcal{X}^k$ of each task into c clusters $\{\mathcal{C}_j^k\}_{j=1}^c$. Note that we assume the dimensionality of the feature vector of all the tasks is the same, i.e. d. It is appropriate since we could augment the feature vectors of all the tasks to make the dimensionality same. In fact, the bag-of-words document representation used in our experiments implicitly does the augmentation. Moreover, we assume that the number of clusters in each task is the same, i.e. $c_1 = c_2 = \dots = c_m = c$, which is also assumed in existing multi-task learning literature. $\mathbf{I}$ denotes an identity matrix with appropriate size.

We consider the problem in a Reproducing Kernel Hilbert Space $\mathcal{H}$ (Schölkopf and Smola 2002) induced by some nonlinear mapping $\phi : \mathbb{R}^d \rightarrow \mathcal{H}$. For a proper chosen ϕ, the inner product $\langle , \rangle$ in $\mathcal{H}$ is defined as

$$\langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle_{\mathcal{H}} = K(\mathbf{x}, \mathbf{y}), \tag{1}$$

where $K(,) : \mathbb{X} \times \mathbb{X} \longrightarrow \mathbb{R}$ is a positive semi-definite kernel function.

Kernel Mean Matching

In multi-task learning, one fundamental question is how to evaluate the difference in distribution between two tasks given finite samples. (Gretton et al. 2006) introduced the Maximum Mean Discrepancy (MMD) for comparing distributions based on the Reproducing Kernel Hilbert Space (RKHS) distance.

Definition 0.1. *(Gretton et al. 2006) Let p and q be distributions defined on a domain $\mathbb{X}$, and let $\mathcal{F}$ be a class of functions $f : \mathbb{X} \to \mathbb{R}$. Given observations $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_{n_1}\}$ and $\mathcal{Y} = \{\mathbf{y}_1, \dots, \mathbf{y}_{n_2}\}$, then the maximum mean discrepancy (MMD) and its empirical estimates are defined as*

$$\begin{aligned} MMD[\mathcal{F}, p, q] &= \sup_{f \in \mathcal{F}} (\mathbb{E}_{\mathbf{x} \sim p[f(\mathbf{x})]} - \mathbb{E}_{\mathbf{y} \sim q[f(\mathbf{y})]}) \\ MMD[\mathcal{F}, \mathcal{X}, \mathcal{Y}] &= \sup_{f \in \mathcal{F}} (\frac{1}{n_1} \sum_{i=1}^{n_1} f(\mathbf{x}_i) - \frac{1}{n_2} \sum_{i=1}^{n_2} f(\mathbf{y}_i)). \end{aligned}$$

Note that MMD is able to capture the high order statistics of the data when the samples are mapped into a high dimensional or ever infinite dimensional space. Based on this observation, (Gretton et al. 2006) proposed to select $\mathcal{F}$ to be the unit ball in a universal RKHS $\mathcal{H}$. As a result, the distance between the distributions of two tasks is simply the distance between the two mean elements in the RKHS. Let $\mathcal{X}^k = \{\mathbf{x}_1^k, \dots, \mathbf{x}_{n_k}^k\}$ and $\mathcal{X}^l = \{\mathbf{x}_1^l, \dots, \mathbf{x}_{n_l}^l\}$ be two sets of data points from the k-th task and the l-th task, then the distance between the k-th task and the l-th task is

$$\text{dist}(\mathcal{X}^k, \mathcal{X}^l) = ||\frac{1}{n_k} \sum_{i=1}^{n_k} \phi(\mathbf{x}_i^k) - \frac{1}{n_l} \sum_{i=1}^{n_l} \phi(\mathbf{x}_i^l)||_{\mathcal{H}}^2, \tag{2}$$

which can be simplified as

$$\text{dist}(\mathcal{X}^k, \mathcal{X}^l) = \text{tr}(\mathbf{\Phi}^{kl} \mathbf{S}^{kl} (\mathbf{\Phi}^{kl})^T) = \text{tr}(\mathbf{S}^{kl} \mathbf{K}^{kl}), \tag{3}$$

where $\mathbf{\Phi}^{kl} = [\phi(\mathbf{x}_1^k), \dots, \phi(\mathbf{x}_{n_k}^k), \phi(\mathbf{x}_1^l), \dots, \phi(\mathbf{x}_{n_l}^l)]$, $\mathbf{K}^{kl} = (\mathbf{\Phi}^{kl})^T \mathbf{\Phi}^{kl}$ and $\mathbf{S}^{kl} \in \mathbb{R}^{(n_k+n_l)\times(n_k+n_l)}$ is defined as

$$S_{ij}^{kl} = \begin{cases} \frac{1}{n_k^2}, & \text{if } \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}_k \\ \frac{1}{n_l^2}, & \text{if } \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}_l \\ -\frac{1}{n_k n_l}, & \text{otherwise} \end{cases}. \tag{4}$$

For all the tasks, we summate the pairwise distances between any two tasks as follows

$$\begin{aligned} \sum_{k,l} \text{dist}(\mathcal{X}^k, \mathcal{X}^l) &= \sum_{k,l} \text{tr}(\mathbf{\Phi}^{kl} \mathbf{S}^{kl} (\mathbf{\Phi}^{kl})^T) \\ &= \text{tr}(\mathbf{\Phi} \mathbf{S} \mathbf{\Phi}^T) = \text{tr}(\mathbf{S}\mathbf{K}), \end{aligned} \tag{5}$$

where $\mathbf{\Phi} = [\phi(\mathbf{x}_1^1), \dots, \phi(\mathbf{x}_{n_1}^1), \dots, \phi(\mathbf{x}_1^m), \dots, \phi(\mathbf{x}_{n_m}^m)]$ and $\mathbf{S} \in \mathbb{R}^{n \times n}$ is defined as

$$S_{ij} = \begin{cases} \frac{m-1}{n_k^2}, & \text{if } \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}_k \\ -\frac{1}{n_k n_l}, & \text{if } \mathbf{x}_i \in \mathcal{X}_k \text{ and } \mathbf{x}_j \in \mathcal{X}_l \end{cases}. \tag{6}$$

In a word, the smaller the objective value of Eq.(5) is, the closer the distributions from any two tasks will be.

Graph Regularization

For good clustering performance, we aim to find a feature mapping $\phi(\mathbf{x})$, which is smooth with respect to the intrinsic manifold structure of the data. Suppose the data is a compact submanifold $\mathcal{M} \in \mathbb{R}^d$, we can use the Laplacian-Beltrami operator on the manifold $||\phi||_{\mathcal{M}}^2$ to measure the smoothness of the feature mapping.

$$||\phi||_M^2 = \int_{\mathbf{x} \in \mathcal{M}} ||\nabla_{\mathcal{M}} \phi(\mathbf{x})||^2 d\mathbf{x}, \tag{7}$$

where $\nabla_{\mathcal{M}} \phi$ is the gradient of ϕ along the manifold $\mathcal{M}$.

In reality, the data manifold $\mathcal{M}$ is unknown. So $||\phi||_M^2$ in Eq.(7) can not be computed. Recent study on spectral graph theory (Chung 1997) has illustrated that $||\phi||_M^2$ can be discretely approximated through the graph Laplacian. In detail, we construct an undirected weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ on the data points of all the task, whose vertex set $\mathcal{V}$ corresponds to the data points $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$. Then the Laplacian-Beltrami operator can be approximated as follows,

$$\begin{aligned} \Omega(\phi) &= \frac{1}{2} \sum_{i,j=1}^n W_{ij} ||\frac{\phi(\mathbf{x}_i)}{\sqrt{D_{ii}}} - \frac{\phi(\mathbf{x}_j)}{\sqrt{D_{jj}}}||_2^2 \\ &= \text{tr}(\mathbf{\Phi} \mathbf{L} \mathbf{\Phi}^T) = \text{tr}(\mathbf{L} \mathbf{\Phi}^T \mathbf{\Phi}) = \text{tr}(\mathbf{L}\mathbf{K}), \end{aligned} \tag{8}$$

where $|| \cdot ||_2$ is 2-norm, $\mathbf{\Phi} = [\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_n)]$, $\mathbf{W} = [W_{ij}]$ is the adjacency matrix, $\mathbf{D}$ is the diagonal degree matrix with $D_{ii} = \sum_j W_{ij}$, and $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}} \mathbf{W} \mathbf{D}^{-\frac{1}{2}}$ is the normalized graph Laplacian (Chung 1997) of graph $\mathcal{G}$. The smaller is $\Omega(\phi)$, the smoother ϕ will be.

We aim to find a nonlinear mapping ϕ, such that if the data points $\mathbf{x}_i$ and $\mathbf{x}_j$ are close to each other in the input space, then their images in the feature space $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ are close as well. So we define the adjacency matrix $\mathbf{W}$ as follows (Cai, He, and Han 2005),

$$W_{ij} = \begin{cases} \text{cosine}(\mathbf{x}_i, \mathbf{x}_j), & \text{if } \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}_k \\ & \text{and } \mathbf{x}_j \in \mathcal{N}(\mathbf{x}_i) \text{ or } \mathbf{x}_i \in \mathcal{N}(\mathbf{x}_j) \\ 0, & \text{otherwise.} \end{cases} . \tag{9}$$

where $\text{cosine}(\mathbf{x}_i, \mathbf{x}_j)$ is the cosine distance between $\mathbf{x}_i$ and $\mathbf{x}_j$, $\mathcal{N}(\mathbf{x}_i)$ denotes the k-nearest neighbor of $\mathbf{x}_i$.

Nonparametric Kernel Learning

Up to now, we have introduced kernel mean matching and graph regularization in RKHS. Our main idea is to seek a kernel $\mathbf{K}$ such that the mapped data in the feature space are not only smooth with respect to the geometric structure of the input data from each task, but also the data distributions of different tasks are as close as possible. Mathematically, it can be formulated as the following optimization problem

$$\begin{aligned} &\min_{\mathbf{K}} \text{tr}(\mathbf{LK}) + C\text{tr}(\mathbf{SK}), \\ \text{s.t.} \quad & \mathbf{0} \preceq \mathbf{K} \preceq \mathbf{I}, \text{tr}(\mathbf{K}) = b, \end{aligned} \tag{10}$$

where C is a positive regularization parameter, $\text{tr}(\mathbf{K}) = b$ is introduced to avoid trivial solution. Note that we do not require the kernel to be universal. Universal kernel guarantees that $MMD(\mathcal{F}, p, q) = 0$ if and only if $p = q$. However, we aim at finding a kernel by which p and q are close, but not necessarily the same.

The problem in Eq.(10) is a kind of non-parametric kernel learning (Lanckriet et al. 2004) (Hoi, Jin, and Lyu 2007), which is a semi-definite programming (Boyd and Vandenberghe 2004). It can be solved by off-the-shelf SDP package such as YALMIP (Lofberg 2004). However, when the number of data points n is large, it is computationally expensive. Fortunately, we can solve it by linear programming.

Theorem 0.2. *Suppose $\mathbf{A}$ is a symmetric matrix such that $\mathbf{A} = \mathbf{P}\Sigma\mathbf{P}^T$, where $\mathbf{P}$ contains columns of orthonormal eigenvectors of $\mathbf{A}$ and $\Sigma = diag(\sigma_1, \ldots, \sigma_n)$ is a diagonal matrix of the corresponding eigenvalues, and b is any positive constant, the optimal solution $\mathbf{K}^*$ to the following semi-definite programming problem*

$$\begin{aligned} &\min_{\mathbf{K}} tr(\mathbf{AK}) \\ s.t. \quad & \mathbf{0} \preceq \mathbf{K} \preceq \mathbf{I}, tr(\mathbf{K}) = b, \end{aligned} \tag{11}$$

can be expressed as $\mathbf{K}^* = \mathbf{P\Gamma P}^T$ where $\mathbf{\Gamma} = diag(\gamma_1, \ldots, \gamma_n)$ is the solution of the following linear programming,

$$\begin{aligned} &\min_{\gamma_i} \sum_{i=1}^{n} \gamma_i \sigma_i \\ s.t. \quad & 0 \leq \gamma_i \leq 1, 1 \leq i \leq n, \sum_{i=1}^{n} \gamma_i = b. \end{aligned} \tag{12}$$

Proof. The key step is to prove that $\mathbf{A}$ and $\mathbf{K}$ can be jointly diagonalized (Golub and Loan 1996). For the space limit, we omit it here. □

In particular, to solve the problem in Eq.(10), let $\mathbf{A} = \mathbf{L} + C\mathbf{S}$, we can obtain the optimal $\mathbf{K}^*$ via linear programming. It is much more efficient than solving it by off-the-shelf SDP package (Lofberg 2004). In summary, we present the algorithm for optimizing the problem in Eq.(10) in Algorithm 1.

Algorithm 1 Learning Nonparametric Kernel for Multi-Task Clustering (LNKMTC)

```
Input: $m$ tasks, $\{\mathbf{X}^k\}_{k=1}^m$, regularization parameter $C$, real positive constant $b$;
1. Compute $\mathbf{S}$ as in Eq.(6);
2. Construct a k-NN graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ on the data points of all the tasks, and compute the normalized graph Laplacian $\mathbf{L}$ as in Eq.(9);
3. Let $\mathbf{A} = \mathbf{L} + C\mathbf{S}$, compute the optimal nonparametric kernel $\mathbf{K}^*$ by Theorem 0.2;
4. Perform kernel kmeans on $\mathbf{K}^*$.
```

Spectral Kernel Learning

Instead of learning a fully non-parametric kernel matrix, (Zhu et al. 2004) suggested to learn a spectral kernel. Suppose $\mathbf{L} = \mathbf{V\Lambda V}^T = \sum_{t=1}^{n} \lambda_t \mathbf{v}_t \mathbf{v}_t^T$, where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_n)$ with $0 \leq \lambda_1 \leq \ldots, \lambda_n$ are the eigenvalues and $||\mathbf{v}_t|| = 1$ are the eigenvectors. It is worth noting that $\mathbf{v}_t$ can be seen as a function defined on the graph $\mathcal{G}$ (Chung 1997). According to Eq.(8), we have $\Omega(\mathbf{v}_t) = \mathbf{v}_t^T \mathbf{L} \mathbf{v}_t = \lambda_t$, which tells us that the smoothness of the function (eigenvector) $\mathbf{v}_t$ is measured by its eigenvalue λ_t. Following (Zhu et al. 2004), we aim to learn a kernel matrix in the form

$$\mathbf{K} = \sum_{t=1}^{r} \mu_t \mathbf{v}_t \mathbf{v}_t^T, \mu_t \geq \mu_{t+1}, t = 1, 2, \ldots, r-1, \tag{13}$$

where $\mathbf{v}_t$ is the eigenvector of $\mathbf{L}$ corresponding to the t-th smallest eigenvalue λ_t. Note that we keep the smoothest $r \ll n$ functions (eigenvectors) $\mathbf{v}_t$ only. Thus Eq.(8) is transformed to

$$\text{tr}(\mathbf{LK}) = \text{tr}(\mathbf{V\Lambda V}^T \mathbf{VUV}^T) = \text{tr}(\mathbf{\Lambda U}) = \sum_{t=1}^{r} \lambda_t \mu_t, \tag{14}$$

where $\mathbf{U} = \text{diag}(\mu_1, \ldots, \mu_r, 0, \ldots, 0)$. Kernel matrix constructed from the graph Laplacian via adapting the eigenvalues of the graph Laplacian in Eq.(13) is typically called graph kernel or spectral kernel (Zhu et al. 2004). The optimization in Eq.(10) can be transformed to

$$\begin{aligned} &\min_{\mu_t} \sum_{t=1}^{r} \lambda_t \mu_t + C \sum_{t=1}^{r} \text{tr}(\mathbf{S} \mathbf{v}_t \mathbf{v}_t^T) \mu_t \\ s.t. \quad & \text{tr}(\sum_{t=1}^{r} \mu_t \mathbf{v}_t \mathbf{v}_t^T) = b \\ & \mu_{t+1} \leq \mu_t, t = 1, \ldots, r-1 \\ & 0 \leq \mu_t \leq 1, t = 1, \ldots, r, \end{aligned} \tag{15}$$

where tr$(\sum_{t=1}^{r}\mu_t \mathbf{v}_t\mathbf{v}_t^T) = b$ is resulted from tr$(\mathbf{K}) = b$. Eq.(15) is a linear programming (Boyd and Vandenberghe 2004).

We summarize the algorithm for optimizing the problem in Eq.(15) in Algorithm 2.

Algorithm 2 Learning Spectral Kernel for Multi-Task Clustering (LSKMTC)

```
Input: $m$ tasks, $\{\mathbf{X}^k\}_{k=1}^m$, regularization parameter $C$, the number of retained eigenvectors $r$, and real positive constant $b$;
1. Compute $\mathbf{S}$ as in Eq.(6);
2. Construct a k-NN graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ on the data points of all the tasks, and compute the normalized graph Laplacian $\mathbf{L}$ as in Eq.(9);
3. Solve the sparse eigen-decomposition $\mathbf{L}\mathbf{v}_t = \lambda_t\mathbf{v}_t$, and retain the $r \ll n$ smallest eigenvalues $\{\lambda_t\}$ and corresponding eigenvectors $\{\mathbf{v}_t\}$ such that $\lambda_1 \leq \ldots \leq \lambda_r$;
4. Solve the linear programming in Eq.(15), and compute the optimal spectral kernel $\mathbf{K}^* = \sum_{t=1}^{r}\mu_t\mathbf{v}_t\mathbf{v}_t^T$ as in Eq.(13);
5. Perform kernel kmeans on $\mathbf{K}^*$.
```

Experiments

In our experiments, we compare the proposed multi-task clustering methods, i.e., LNKMTC and LSKMTC, with typical single-task clustering methods, e.g., K-means, Kernel K-means (KKM) and Normalized Cut (NCut) (Shi and Malik 2000). We also present the experimental results of clustering the data of all the tasks together using K-means, KKM and NCut. Furthermore, we compare the proposed methods with the multi-task clustering method proposed in (Gu and Zhou 2009), namely *Learning the Shared Subspace for Multi-Task Clustering* (LSSMTC). In addition, we also evaluate the proposed methods with $C = 0$, which means we only preserve the geometric structures of different tasks, but do not require the distributions of different tasks to be close. We refer to these two specific methods as LNKMTC0 and LSKMTC0. All of our experiments have been performed on a Intel Core2 Duo 2.8GHz Windows XP machine with 3GB memory.

Evaluation Metrics

To evaluate the clustering results, we adopt two performance measures (Xu, Liu, and Gong 2003), which are widely used in the literature.

Clustering Accuracy is defined as follows:

$$Acc = \frac{\sum_{i=1}^{n}\delta(map(r_i), l_i)}{n}, \tag{16}$$

where r_i denotes the cluster label of $\mathbf{x}_i$, and l_i denotes the true class label, n is the total number of documents, $\delta(x, y)$ is the delta function that equals one if $x = y$ and equals zero otherwise, and $map(r_i)$ is the permutation mapping function that maps each cluster label r_i to the equivalent label from the data set.

Normalized Mutual Information is used for determining the quality of clusters. Given a clustering result, the NMI is estimated by

$$NMI = \frac{\sum_{i=1}^{c}\sum_{j=1}^{c} n_{i,j}\log\frac{n_{i,j}}{n_i\hat{n}_j}}{\sqrt{(\sum_{i=1}^{c} n_i \log\frac{n_i}{n})(\sum_{j=1}^{c}\hat{n}_j\log\frac{\hat{n}_j}{n})}}, \tag{17}$$

where n_i denotes the number of data contained in the cluster $\mathcal{C}_i (1 \leq i \leq c)$, $\hat{n}_j$ is the number of data belonging to the $\mathcal{L}_j (1 \leq j \leq c)$, and $n_{i,j}$ denotes the number of data that are in the intersection between the cluster $\mathcal{C}_i$ and the class $\mathcal{L}_j$. The larger the NMI is, the better the clustering result will be.

Data Sets

In order to evaluate the proposed methods, we use two text data sets, which are used in (Gu and Zhou 2009).

WebKB[1] data set contains web pages gathered from university computer science departments (Cornell, Texas, Washington, Wisconsin). There are about 8280 documents and they are divided into 7 categories, and we choose student, faculty, course and project these four most populous entity-representation categories for clustering, named WebKB4. We consider clustering the web pages of each university as one task. Therefore, we have 4 tasks.

20Newsgroup[2] is a collection of approximately 20000 newsgroup documents, partitioned across 20 different newsgroups nearly evenly. We generate 2 cross-domain data sets, i.e., Rec.vs.Talk and Comp.vs.Sci, for evaluating multi-task clustering methods. In detail, two top categories are chosen, one as positive and the other as negative. Then the data are split based on sub-categories. The task is defined as top category classification. The splitting ensures the data in different tasks are related but different, since they are drawn from the same top category but different sub-categories. The detailed constitutions of the two data sets are summarized in Table 1.

Table 1: Constitution of the two data sets generated from 20Newsgroup

Data set	Task id	Class 1	Class 2
Rec.vs.Talk	Task 1	rec.autos	talk.politics.guns
	Task 2	rec.sport.baseball	talk.politics.mideast
Comp.vs.Sci	Task 1	comp.os.ms-windows.misc	sci.crypt
	Task 2	comp.sys.mac.hardware	sci.space

Table 2 summarizes the characteristics of the three data sets used in our experiments.

Methods & Parameter Settings

We set the number of clusters equal to the true number of classes for all the clustering algorithms. The results of Kmeans, NCut and LSSMTC are taken from (Gu and Zhou 2009).

KKM: We use Gaussian kernel for kernel k-means, the width of the Gaussian kernel σ is set by searching the grid $\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3\}$.

[1] http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/

[2] http://people.csail.mit.edu/jrennie/20Newsgroups/

Table 2: Description of the data sets

Data set	Task id	#Sample	#Feature	#Class
WebKB4	Task 1	227	2000	4
	Task 2	250	2000	4
	Task 3	248	2000	4
	Task 4	304	2000	4
Rec.vs.Talk	Task 1	1844	2000	2
	Task 2	1545	2000	2
Comp.vs.Sci	Task 1	1875	2000	2
	Task 2	1827	2000	2

LNKMTC: The neighborhood size of the k-NN graph is set to 10, the regularization parameter C is set by searching the grid $\{0.1, 1, 10, 100, 500, 1000\}$, b is set to 30.

LSKMTC: The construction of the k-NN graph and the setting of the regularization parameter C are the same as that in LNKMTC, b is set to 1, the number of retained eigenvectors is set to 30.

Under each parameter setting, we repeat clustering 10 times, and the mean result as well as the standard deviation are computed. We report the mean and standard deviation corresponding to the best parameter setting for each method to compare with each other.

Clustering Results

The average results are shown in Table 3, Table 4 and Table 5. "All" refers to clustering the data of all the tasks together. From the tables, we observe that:

Table 4: Clustering Results on Rec.vs.Talk

	Task 1		Task 2	
Method	Acc (%)	NMI (%)	Acc (%)	NMI (%)
Kmeans	64.67±3.82	18.84±3.07	64.54±9.67	15.68±13.79
KKM	68.55±0.15	22.05±0.35	70.77±10.65	25.01±13.82
NCut	67.79±0.00	22.16±0.00	68.87±0.00	21.22±0.00
All Kmeans	65.51±3.82	17.42±1.60	58.98±2.71	5.58±4.43
All KKM	69.01±0.15	14.63±0.23	63.11±0.38	10.03±0.25
All NCut	68.66±0.00	26.04±0.00	61.88±0.00	7.83±0.00
LSSMTC	84.33±8.04	43.06±5.82	78.95±8.27	34.73±8.35
LNKMTC0	66.19±8.92	11.06±10.03	72.83±10.80	22.65±15.56
LSKMTC0	64.34±10.57	10.17±11.54	72.49±12.75	22.87±18.74
LNKMTC	82.87±2.07	41.78±0.66	**85.66±7.70**	**47.46±6.99**
LSKMTC	**87.97±1.61**	**49.53±1.30**	84.05±5.87	42.64±5.23

1. Multi-task clustering methods (e.g. LSSMTC, LNKMTC and LSKMTC) indeed outperform single-task clustering methods such as K-means, Kernel K-means and NCut. The improvement owes to the utilization of the relationship among the tasks.

2. Our methods (LNKMTC and LSKMTC) are better than LSSMTC. As we mentioned before, LSSMTC aims to find a subspace in which all the tasks share an identical set of cluster centers. However, the dimensionality of the subspace is difficult to determine, which has to be tuned by grid search. Moreover, it does not consider the distributions of different tasks explicitly. In contrast, our methods are based on kernel mean matching, which considers the distributions of different tasks and owns solid theoretical foundation. In addition, there is no need to choose the dimensionality of the RHKS. This is the main reason why our methods are superior to LSSMTC.

Table 5: Clustering Results % on Comp.vs.Sci

	Task 1		Task 2	
Method	Acc (%)	NMI (%)	Acc (%)	NMI (%)
Kmeans	61.30±2.02	17.27±2.28	67.16±0.00	20.87±0.00
KKM	81.33±15.36	44.98±22.12	67.26±0.50	17.48±3.62
NCut	66.83±0.00	23.27±0.00	66.78±0.00	16.94±0.00
All Kmeans	66.56±7.27	23.30±9.24	54.07±2.11	5.32±1.91
All KKM	74.57±6.99	22.92±6.79	52.19±2.62	1.40±2.69
All NCut	63.52±0.00	19.41±0.00	55.06±0.00	6.27±0.00
LSSMTC	88.01±0.76	53.76±1.55	80.16±6.14	33.47±14.07
LNKMTC0	62.63±10.35	9.06±11.11	59.06±8.55	4.46±7.45
LSKMTC0	66.83±13.17	14.68±17.46	64.56±7.04	7.69±6.34
LNKMTC	**91.57±1.58**	**56.26±5.60**	82.33±6.31	38.00±7.18
LSKMTC	90.16±5.27	55.29±6.42	**84.24±6.71**	**40.73±6.82**

3. Generally speaking, LSKMTC is comparable to or even better than LNKMTC, this is consistent with existing observation that spectral kernel learning is usually better than nonparametric kernel learning (Zhu et al. 2004).

4. As can be seen, simply clustering the data of all the tasks together (e.g. All Kmeans, All KKM and All NCut) does not necessarily improve the clustering result, because the data distributions of different tasks are not the same, and combining the data together directly violates the i.i.d. assumption in single-task clustering. In our methods, after kernel mean matching, the distributions of all the tasks are similar. As a result, single task clustering algorithm (KKM in our experiments) can be performed on the mapped data.

5. In addition, LNKMTC0 and LSKMTC0 perform poorly. The reason is that they are essentially combing the data together and doing unsupervised non-parametric kernel learning or spectral kernel learning for kernel k-means. As a result, they also violate the i.i.d. assumption in single-task clustering. This indicates that the kernel mean matching term in the proposed methods is very crucial for multi-task clustering.

Computational Time

In this subsection, we investigate the computational time of the 3 multi-task clustering methods. We plot the computational time of LSSMTC ($l = 2, \lambda = 0.5$), LNKMTC ($C = 1$) and LSKMTC ($C = 1$) on the WebKB4, Rec.vs.Talk and Comp.vs.Sci data sets in Figure 1. We can see that LSKMTC is much more efficient than the other two methods. Considering both the effectiveness and efficiency of our methods, LSKMTC is superior to LNKMTC.

Conclusion

In this paper, we propose to learn a Reproducing Kernel Hilbert Space, in which the geometric structure of the data from each task is preserved, while the data distributions of any two tasks are as close as possible. We study two kinds of kernel learning: nonparametric kernel learning and spectral kernel design, both of which can be solved by linear programming. Experiments on text data sets demonstrate that

Table 3: Clustering Results on WebKB4

	Task 1		Task 2		Task 3		Task 4	
Method	Acc (%)	NMI (%)	Acc (%)	NMI (%)	Acc (%)	NMI (%)	Acc (%)	NMI (%)
Kmeans	57.84±9.96	27.60±7.53	56.70±6.97	25.52±5.51	56.71±9.03	28.14±6.03	67.70±7.73	35.52±9.49
KKM	53.57±8.35	19.70±6.68	53.04±5.56	31.55±5.92	60.08±4.86	33.94±3.70	71.45±6.04	43.47±2.10
NCut	49.07±1.88	28.16±3.42	67.20±0.00	36.32±0.00	52.82±0.00	34.66±0.00	60.79±0.15	25.55±0.68
All Kmeans	54.76±8.37	18.46±7.36	59.44±4.47	31.78±5.14	59.80±9.14	21.47±12.55	58.98±11.58	31.08±11.35
All KKM	56.83±4.36	10.96±2.62	56.96±3.62	4.74±1.82	50.00±0.95	3.45±1.82	63.16±9.29	27.54±15.44
All NCut	56.83±0.00	25.05±0.00	59.20±0.00	27.21±0.00	49.60±0.00	23.40±0.00	51.32±0.00	26.20±0.00
LSSMTC	62.47±3.36	33.69±1.44	63.04±3.64	34.16±1.01	66.77±4.08	35.52±1.47	73.29±3.33	42.40±0.96
LNKMTC0	48.11±8.83	23.62±7.69	52.00±5.78	25.50±8.47	63.43±5.39	32.68±4.96	47.76±6.91	23.48±8.71
LSKMTC0	46.12±9.24	24.64±6.23	55.16±5.57	30.11±5.09	59.15±4.33	28.24±6.32	50.99±8.69	21.57±5.82
LNKMTC	66.39±1.96	36.34±1.65	69.20±1.53	42.09±1.80	69.60±2.50	41.95±2.06	71.32±2.23	45.53±1.21
LSKMTC	**69.21±1.13**	**40.85±2.16**	**72.80±1.26**	**45.41±2.00**	**70.40±1.27**	**42.46±2.09**	**76.38±1.94**	**50.97±1.16**

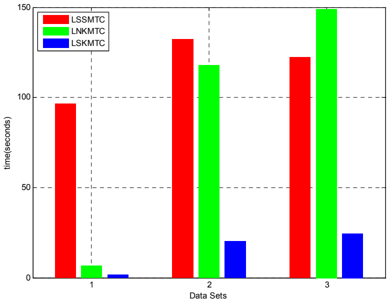

Figure 1: The computational time of LSSMTC, LNKMTC and LSKMTC on the WebKB4, Rec.vs.Talk and Comp.vs.Sci data sets.

the proposed multi-task clustering methods outperform existing multi-task clustering method as well as state of the art single-task clustering methods greatly.

Acknowledgments

The work was supported in part by NSF IIS-09-05215, U.S. Air Force Office of Scientific Research MURI award FA9550-08-1-0265, and the U.S. Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053 (NS-CTA). We thank the anonymous reviewers for their helpful comments.

References

Ando, R. K., and Zhang, T. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research* 6:1817–1853.

Argyriou, A.; Evgeniou, T.; and Pontil, M. 2006. Multi-task feature learning. In *NIPS*, 41–48.

Boyd, S., and Vandenberghe, L. 2004. *Convex optimization*. Cambridge: Cambridge University Press.

Cai, D.; He, X.; and Han, J. 2005. Document clustering using locality preserving indexing. *IEEE Trans. Knowl. Data Eng.* 17(12):1624–1637.

Caruana, R. 1997. Multitask learning. *Machine Learning* 28(1):41–75.

Chung, F. R. K. 1997. *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92) (Cbms Regional Conference Series in Mathematics)*. American Mathematical Society.

Duan, L.; Tsang, I. W.-H.; Xu, D.; and Maybank, S. J. 2009. Domain transfer svm for video concept detection. In *CVPR*, 1375–1381.

Evgeniou, T., and Pontil, M. 2004. Regularized multi–task learning. In *KDD*, 109–117.

Golub, G. H., and Loan, C. F. V. 1996. *Matrix computations (3rd ed.)*. Baltimore, MD, USA: Johns Hopkins University Press.

Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. J. 2006. A kernel method for the two-sample-problem. In *NIPS*, 513–520.

Gu, Q., and Zhou, J. 2009. Learning the shared subspace for multi-task clustering and transductive transfer classification. In *ICDM*, 159–168.

Hoi, S. C. H.; Jin, R.; and Lyu, M. R. 2007. Learning nonparametric kernel matrices from pairwise constraints. In *ICML*, 361–368.

Lanckriet, G. R. G.; Cristianini, N.; Bartlett, P. L.; Ghaoui, L. E.; and Jordan, M. I. 2004. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research* 5:27–72.

Lofberg, J. 2004. YALMIP: a toolbox for modeling and optimization in Matlab. In *IEEE Intern. Symp. Computer Aided Control Systems Design*, 284–289.

Micchelli, C. A., and Pontil, M. 2004. Kernels for multi–task learning. In *NIPS*.

Pan, S. J.; Kwok, J. T.; and Yang, Q. 2008. Transfer learning via dimensionality reduction. In *AAAI*, 677–682.

Schölkopf, B., and Smola, A. 2002. *Learning with Kernels*. Cambridge, MA, USA: MIT Press.

Shi, J., and Malik, J. 2000. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 22(8):888–905.

Xu, W.; Liu, X.; and Gong, Y. 2003. Document clustering based on non-negative matrix factorization. In *SIGIR*, 267–273.

Zhang, Y., and yan Yeung, D. 2010. A convex formulation for learning task relationships in multi-task learning. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence (UAI)*.

Zhang, J.; Ghahramani, Z.; and Yang, Y. 2005. Learning multiple related tasks using latent independent component analysis. In *NIPS*.

Zhu, X.; Kandola, J. S.; Ghahramani, Z.; and Lafferty, J. D. 2004. Nonparametric transforms of graph kernels for semi-supervised learning. In *NIPS*.