Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Multi-Task Clustering with Model Relation Learning

Xiaotong Zhang[1,2], **Xianchao Zhang**[1,2], **Han Liu**[1,2], **Jiebo Luo**[3]
[1] School of Software, Dalian University of Technology
[2] Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province
[3] Department of Computer Science, University of Rochester
zxt.dut@hotmail.com, xczhang@dlut.edu.cn, liu.han.dut@gmail.com, jluo@cs.rochester.edu

Abstract

Multi-task clustering improves the clustering performance of each task by transferring knowledge among the related tasks. An important aspect of multi-task clustering is to assess the task relatedness. However, to our knowledge, only two previous works have assessed the task relatedness, but they both have limitations. In this paper, we propose a multi-task clustering with model relation learning (MTCMRL) method, which automatically learns the model parameter relatedness between each pair of tasks. The objective function of MTCMRL consists of two parts: (1) within-task clustering: clustering each task by introducing linear regression model into symmetric nonnegative matrix factorization; (2) cross-task relatedness learning: updating the parameter of the linear regression model in each task by learning the model parameter relatedness between the clusters in each pair of tasks. We present an effective alternating algorithm to solve the non-convex optimization problem. Experimental results show the superiority of the proposed method over traditional single-task clustering methods and existing multi-task clustering methods.

1 Introduction

Multi-task clustering, which improves the clustering performance of each task by transferring knowledge among the related tasks, receives increasing attention recently. Most existing multi-task clustering methods make an ideal assumption that the tasks are completely related, i.e., the tasks share the same latent categories. However, in many real applications, the tasks are usually partially related, i.e., only parts of the latent categories are shared among the tasks. Transferring knowledge among the partially related tasks without considering the task relatedness may cause negative transfer [Pan and Yang, 2010], which degrades the clustering performance. For example, we have two tasks to cluster the news from two news agencies by topics, one agency reports the news of Education, Sport and Science, the other agency reports the news of Education, Sport and Politics. Only the news on the topics of Education and Sport are useful for multi-task knowledge transfer, and transferring the knowledge of the news on the topics of Science and Politics may cause negative effect.

In the multi-task clustering literatures, only two methods can automatically assess the task relatedness. (1) DMTRC [Zhang, 2015] formulates the task relatedness as the covariance matrix by the Gaussian prior. But it assumes that the cluster numbers in all the tasks are the same and the label marginal distribution in each task is evenly distributed, which is too restrictive. (2) SAMTC [Zhang *et al.*, 2016] first finds possibly related clusters between each pair of tasks and obtains a pair of subtasks, then it further learns the relatedness between each pair of subtasks. But it only transfers the instance knowledge between each pair of subtasks, which may lose the useful instance knowledge not in the subtasks.

In light of the limitations of the existing multi-task clustering methods, in this paper, we propose a multi-task clustering with model relation learning (MTCMRL) method, which can automatically learn the model parameter relatedness between each pair of tasks. The overall objective function of MTCMRL consists of two parts. (1) Within-task clustering: this part is to find the clusters within each task and update the clusters by knowledge transfer from the other tasks. We introduce linear regression model into symmetric nonnegative matrix factorization [Kuang *et al.*, 2012] to accomplish this goal. The symmetric nonnegative matrix factorization captures the cluster structure embedded into the similarity matrix. The linear regression model provides a condition for transferring the knowledge of model parameters across the tasks. (2) Cross-task relatedness learning: this part is to learn the model parameter relatedness between the clusters in each pair of tasks, then the model parameter of each task can be updated by those of the other tasks. The basic intuition is that if the tasks are related, there are some clusters related to each other among these tasks. Thus the model parameters used for predicting the cluster indicator among these tasks should have some relatedness. We further present an effective alternating algorithm to solve the non-convex optimization problem. Experimental results show the superiority of the proposed method over traditional single-task clustering methods and existing multi-task clustering methods.

2 Related Work

Multi-Task Clustering: Most multi-task clustering methods make an ideal assumption that the tasks are completely re-

lated, i.e., they transfer knowledge among the tasks without considering the task relatedness. LSSMTC [Gu and Zhou, 2009] learns a subspace where the related tasks have the same centroids. LNKMTC and LSKMTC [Gu *et al.*, 2011] learn a kernel space where the task distributions are close to each other. MCDA [Zhang and Zhou, 2012] learns a subspace where the task distributions are close to each other. ITCC [Xie *et al.*, 2012] learns the feature relatedness among the related tasks by information theoretic co-clustering. MBC [Zhang and Zhang, 2010] and its improved methods (S-MBC and S-MKC) [Zhang *et al.*, 2015] alternatively update the clusters and learn the relationship between clusters of different tasks. SMT-NMF [Al-Stouhi and Reddy, 2014] introduces an inter-task bias to reweight the distance between any two samples in different tasks. DMTFC [Zhang, 2015] learns the feature relatedness through Gaussian prior. MTCTKI [Zhang *et al.*, 2017] transfers instance knowledge among the tasks under a shared subspace where the task distributions are close to each other. MTSC [Yang *et al.*, 2015] learns a lower dimensional feature space by incorporating an $l_{2,p}$-norm regularization term over the model parameters. However, in the real world, the tasks are usually partially related. Transferring knowledge among partially related tasks without considering the task relatedness may cause negative transfer [Pan and Yang, 2010], which degrades the clustering performance.

To our knowledge, in the multi-task clustering literatures, only DMTRC [Zhang, 2015] and SAMTC [Zhang *et al.*, 2016] can automatically assess the task relatedness, but they have some limitations. DMTRC formulates the task relatedness as the covariance matrix by the Gaussian prior. But it assumes that the tasks have the same cluster number and the label marginal distribution in each task is evenly distributed, which limits its applicability. SAMTC first finds the related clusters between each pair of tasks and obtains a pair of subtasks, then it further learns the relatedness between each pair of subtasks, finally it constructs the similarity matrix for each task by exploiting the instances from the other subtasks. But it only transfers the instance knowledge between each pair of subtasks, which may lose the useful instance knowledge not in the subtasks.

Besides the multi-task clustering (unsupervised multi-task learning) methods which can automatically assess the task relatedness, many supervised multi-task learning methods are proposed for learning the task relatedness [Zhang and Yang, 2017]. A recent method in [Murugesan *et al.*, 2017] learns both the task relationship matrix and feature relationship matrix by co-clustering the tasks and features. But it requires true labels to learn the task cluster matrix. In this paper, we concentrate on learning the task relatedness without labels.

Multi-View Learning: Different from multi-task learning which deals with the data from multiple tasks, multi-view learning [Zhao *et al.*, 2017] deals with the data which have the features from different views. Multi-view learning improves the learning performance by maximizing the agreement on multiple distinct views. There are mainly three representative ways for multi-view learning. Co-training [Blum and Mitchell, 1998; Appice and Malerba, 2016] uses the most confident predictions in each view to iteratively train the learning model for all the views. Co-regularization [Kumar *et al.*, 2011; Minh *et al.*, 2016] takes the disagreement between the classification or clustering results of any two views as a regularization term in the objective function. Canonical correlation analysis [Hotelling, 1936; Podosinnikova *et al.*, 2016] aims to find the linear projections which have maximum correlation among different views. Recently, multi-view learning has been successfully applied to the gene network reconstruction [Ceci *et al.*, 2015].

3 The Proposed Method

3.1 Problem Formulation

We are given $\mathcal{T}$ clustering tasks, each with a set of data points, i.e., $X^t = \{x_1^t, x_2^t, \ldots, x_{n^t}^t\} \in \mathbb{R}^{d \times n^t} (t = 1, \ldots, \mathcal{T})$, where n^t is the number of data points in the t-th task, d is the dimensionality of the feature vectors. The similarity matrix of the t-th task is $M^t \in \mathbb{R}^{n^t \times n^t}$. Each data set X^t is to be partitioned into h^t clusters, i.e., $C^t = \{C_1^t, C_2^t, \ldots, C_{h^t}^t\}$. The cluster indicator of the t-th task is $Y^t \in \mathbb{R}^{n^t \times h^t}$.

3.2 Within-Task Clustering

This component is to find the clusters within each task and update the clusters by knowledge transfer from the other tasks. We introduce linear regression model (LRM) into symmetric nonnegative matrix factorization (SNMF) to accomplish this goal. SNMF partitions the data points of each task into clusters, LRM provides a condition for transferring the knowledge of model parameters across the tasks.

Denote the parameter of the linear regression model in the t-th task as $W^t \in \mathbb{R}^{d \times h^t}$, the objective function of within-task clustering for the t-th task is

$$\begin{aligned} &\min_{Y^t, W^t} J_{in}^t = \frac{1}{2}||M^t - Y^t(Y^t)^T||_F^2 \\ &+ \lambda||Y^t - (X^t)^T W^t||_F^2 + \mu||W^t||_F^2, \\ &s.t. Y^t \geq 0. \end{aligned} \tag{1}$$

where M^t is the similarity matrix of the t-th task, which can be computed by the similarity metric that accommodates to the data set. For example, we would use cosine similarity for document data sets, and adopt the Itakura-Saito divergence for music data sets. λ is a trade-off parameter to control the importance of LRM. μ is a regularization parameter for LRM.

In Eq.(1), the first term clusters the data points in the t-th task by SNMF. SNMF is a clustering technique based on the similarity matrix, which has the following advantages. (1) The cluster indicator Y^t computed by SNMF is nonnegative, thus we can directly get the clustering results without post-clustering on Y^t. (2) SNMF retains the near-orthogonality of columns of the cluster indicator Y^t, i.e., $(Y^t)^T Y^t = I$, which is important for data clustering. The second term uses LRM to refine the cluster indicator Y^t by learning a model parameter W^t on the data X^t. LRM is a predictive approach which fits the output variable by using the linear combination of the features of the input sample. By updating the model parameter W^t through cross-task relatedness learning, the cluster indicator Y^t will be improved. The third term is a regularization term for the model parameter W^t of LRM.

3.3 Cross-Task Relatedness Learning

This component is to learn the cross-task relatedness. Specifically, we learn the task relatedness by computing the model parameter relatedness between any two clusters in each pair of tasks. The basic intuition is that if the tasks are related, there are some clusters related to each other among these tasks. Therefore, we explore the relation of the clusters in different tasks by learning the relatedness of the model parameters corresponding to these clusters, then the model parameter of each task can be updated by those of the other tasks.

Denote the cluster model parameter relatedness between the t-th task and the s-th task as $G^{(t,s)} \in \mathbb{R}^{h^t \times h^s}$, the objective of learning $G^{(t,s)}$ is

$$\begin{aligned} &\min_{G^{(t,s)}} J^{ts}_{cross} = \sum_{i=1}^{h^t}\sum_{j=1}^{h^s} ||W^t_i - W^s_j||_2^2 G^{(t,s)}_{ij} + \beta||G^{(t,s)}||_F^2, \\ &s.t. \sum_{i=1}^{h^t}\sum_{j=1}^{h^s} G^{(t,s)}_{ij} = 1, 0 \le G^{(t,s)}_{ij} \le 1(t \ne s), \end{aligned} \tag{2}$$

where $W^t_i \in \mathbb{R}^{d\times 1}$ is the i-th column of W^t, which is a model parameter corresponding to the i-th cluster of the t-th task. $G^{(t,s)}_{ij}$ is the similarity between the i-th cluster of the t-th task and the j-th cluster of the s-th task. The first term means that the smaller the squared Euclidean distance between W^t_i and W^s_j is, the higher the similarity between them is. The second term is to avoid the trivial solution that only the parameters W^t_i and W^s_j with the minimum squared Euclidean distance have the similarity 1, otherwise they have the similarity 0. β is a regularization parameter.

In Eq.(2), a larger $G^{(t,s)}_{ij}$ means that the model parameters between the i-th cluster in the t-th task and the j-th cluster in the s-th task have a higher relatedness. If $G^{(t,s)}_{ij}$ equals to 0, the model parameters between the i-th cluster in the t-th task and the j-th cluster in the s-th task have no relatedness, thus no model parameter knowledge will be transferred between these two clusters.

3.4 The Overall Objective Function

We integrate within-task clustering and cross-task relatedness learning into the overall objective function as follows.

$$\begin{aligned} &\min_{Y^t, W^t, G^{(t,s)}} J_{all} = \sum_{t=1}^{\mathcal{T}} (\frac{1}{2}||M^t - Y^t(Y^t)^T||_F^2 \\ &+ \lambda||Y^t - (X^t)^T W^t||_F^2 + \mu||W^t||_F^2) \\ &+ \alpha \sum_{t=1}^{\mathcal{T}}\sum_{s\ne t}^{\mathcal{T}} (\sum_{i=1}^{h^t}\sum_{j=1}^{h^s} ||W^t_i - W^s_j||_2^2 G^{(t,s)}_{ij} + \beta||G^{(t,s)}||_F^2), \\ &s.t. Y^t \ge 0, \sum_{i=1}^{h^t}\sum_{j=1}^{h^s} G^{(t,s)}_{ij} = 1, 0 \le G^{(t,s)}_{ij} \le 1(t \ne s), \end{aligned} \tag{3}$$

where α is a trade-off parameter to control the importance of cross-task relatedness learning. According to Theorem 2 in [Ding and He, 2005], the orthogonality constraint $(Y^t)^T Y^t = I$ is retained in SNMF, thus we do not need to incorporate this constraint into optimization.

3.5 Optimization

Optimizing Eq.(3) is with respect to variables Y^t, W^t and $G^{(t,s)}$. We alternatively optimize each variable by fixing the other variables.

Optimizing $G^{(t,s)}$: Given Y^t and W^t, optimizing Eq.(3) with respect to $G^{(t,s)}$ is equivalent to optimize

$$\begin{aligned} &\min_{G^{(t,s)}} J^{ts}_{cross} = \sum_{i=1}^{h^t}\sum_{j=1}^{h^s} ||W^t_i - W^s_j||_2^2 G^{(t,s)}_{ij} + \beta||G^{(t,s)}||_F^2, \\ &s.t. \sum_{i=1}^{h^t}\sum_{j=1}^{h^s} G^{(t,s)}_{ij} = 1, 0 \le G^{(t,s)}_{ij} \le 1(t \ne s), \end{aligned} \tag{4}$$

Eq.(4) can be rewritten as

$$\begin{aligned} &\min_{G^{(t,s)}} J^{ts}_{cross} = \min_g \frac{1}{2} g^T P g + f^T g, \\ &s.t. \sum_{i=1}^{h^t\times h^s} g_i = 1, 0 \le g_i \le 1, \end{aligned} \tag{5}$$

where $g = (G^{(t,s)}_{1,:}, G^{(t,s)}_{2,:}, \ldots, G^{(t,s)}_{h^t,:})^T$, where $G^{(t,s)}_{i,:}$ is the i-th row of $G^{(t,s)}$. $f = (A^{(t,s)}_{1,:}, A^{(t,s)}_{2,:}, \ldots, A^{(t,s)}_{h^t,:})^T$, where $A^{(t,s)}_{ij} = ||W^t_i - W^s_j||_2^2$, $A^{(t,s)}_{i,:}$ is the i-th row of $A^{(t,s)}$. g and f are both $h^t \times h^s$ dimensional vectors. $P = 2\beta I \in \mathbb{R}^{(h^t\times h^s)\times(h^t\times h^s)}$. Eq.(5) is a convex quadratic programming problem, which can be solved by the quadratic programming optimizer such as the QUAD function in MATLAB.

Optimizing W^t: Given Y^t and $G^{(t,s)}$, optimizing Eq.(3) with respect to W^t is equivalent to optimize

$$\begin{aligned} &\min_{W^t} \lambda||Y^t - (X^t)^T W^t||_F^2 + \mu||W^t||_F^2 \\ &+ \alpha \sum_{s\ne t}^{\mathcal{T}} (\sum_{i=1}^{h^t}\sum_{j=1}^{h^s} ||W^t_i - W^s_j||_2^2 G^{(t,s)}_{ij}). \end{aligned} \tag{6}$$

By omitting the constant, Eq.(6) can be rewritten as

$$\begin{aligned} &\min_{W^t} \lambda||Y^t - (X^t)^T W^t||_F^2 + \mu tr(W^t(W^t)^T) \\ &+ \alpha \sum_{s\ne t}^{\mathcal{T}} tr(W^t H^{(t,s)}(W^t)^T - 2W^t G^{(t,s)}(W^s)^T), \end{aligned} \tag{7}$$

where $H^{(t,s)}$ is a diagonal matrix with $H^{(t,s)}_{ii} = \sum_{j=1}^{h^s} G^{(t,s)}_{ij}$.

Eq.(7) can be transformed to a convex formula with constraint.

$$\begin{aligned} &\min_{W^t} J_{W^t} = \lambda tr(\xi^t(\xi^t)^T) + \mu tr(W^t(W^t)^T) \\ &+ \alpha \sum_{s\ne t}^{\mathcal{T}} tr(W^t H^{(t,s)}(W^t)^T - 2W^t G^{(t,s)}(W^s)^T), \\ &s.t. Y^t - (X^t)^T W^t = \xi^t. \end{aligned} \tag{8}$$

The Lagrangian function of Eq.(8) is

$$L(W^t,\xi^t) = \lambda tr(\xi^t(\xi^t)^T) + \mu tr(W^t(W^t)^T) + \alpha\sum_{s\neq t}^{\mathcal{T}} tr(W^tH^{(t,s)}(W^t)^T - 2W^tG^{(t,s)}(W^s)^T) + tr(\Gamma^t(Y^t - (X^t)^TW^t - \xi^t)^T), \quad (9)$$

where $\Gamma^t \in \mathbb{R}^{n^t\times h^t}$ is a Lagrangian multiplier. Eq.(9) is convex with respect to W^t and ξ^t. According to Karush-Kuhn-Tucker (KKT) condition $\frac{\partial L(W^t,\xi^t)}{\partial W^t} = 0$ and $\frac{\partial L(W^t,\xi^t)}{\partial \xi^t} = 0$ [Boyd and Vandenberghe, 2004], we have

$$W^t = (2\alpha\sum_{s\neq t}^{\mathcal{T}} W^s(G^{(t,s)})^T + X^t\Gamma^t)(2\mu I + 2\alpha\sum_{s\neq t}^{\mathcal{T}} H^{(t,s)})^{-1}. \quad (10)$$

$$\xi^t = \frac{1}{2\lambda}\Gamma^t. \quad (11)$$

By introducing Eq.(10) and Eq.(11) into Eq.(9), we get the dual problem [Boyd and Vandenberghe, 2004] of Eq.(8).

$$\max_{\Gamma^t} J_{\Gamma^t} = -\frac{1}{4\lambda}tr(\Gamma^t(\Gamma^t)^T) + tr(\Gamma^t(Y^t)^T) - \frac{1}{2}tr(X^t\Gamma^t(2\mu I + 2\alpha\sum_{s\neq t}^{\mathcal{T}} H^{(t,s)})^{-1}(\Gamma^t)^T(X^t)^T) - tr(\Gamma^t(2\mu I + 2\alpha\sum_{s\neq t}^{\mathcal{T}} H^{(t,s)})^{-1}(2\alpha\sum_{s\neq t}^{\mathcal{T}} G^{(t,s)}(W^s)^T)X^t). \quad (12)$$

Setting $\frac{\partial J_{\Gamma^t}}{\partial \Gamma^t} = 0$, we have

$$\Gamma^t\frac{1}{2\lambda}(2\mu I + 2\alpha\sum_{s\neq t}^{\mathcal{T}} H^{(t,s)}) + (X^t)^TX^t\Gamma^t = Y^t(2\mu I + 2\alpha\sum_{s\neq t}^{\mathcal{T}} H^{(t,s)}) - 2\alpha\sum_{s\neq t}^{\mathcal{T}}(X^t)^TW^s(G^{(t,s)})^T. \quad (13)$$

Eq.(13) is the Sylvester equation, which can be solved by the off-the-shelf LYAP function in MATLAB.

Optimizing Y^t: Given W^t and $G^{(t,s)}$, optimizing Eq.(3) with respect to Y^t is equivalent to optimize

$$\min_{Y^t} J_{Y^t} = \frac{1}{2}||M^t - Y^t(Y^t)^T||_F^2 + \lambda||Y^t - (X^t)^TW^t||_F^2, \quad s.t. Y^t \geq 0. \quad (14)$$

The Lagrangian function of Eq.(14) is

$$L(Y^t) = \frac{1}{2}||M^t - Y^t(Y^t)^T||_F^2 + \lambda||Y^t - (X^t)^TW^t||_F^2 - tr(\triangle(Y^t)^T), \quad (15)$$

Algorithm 1 MTCMRL

Input: $\mathcal{T}$ tasks $\{X^t\}_{t=1}^{\mathcal{T}}$, cluster number of each task $\{h^t\}_{t=1}^{\mathcal{T}}$. The parameters λ, μ, α and β. Initializing $M^t(t = 1,\ldots,\mathcal{T})$ by the similarity metric which accommodates to the data set. Initializing $Y^t(t = 1,\ldots,\mathcal{T})$ by the k-means method, then setting $Y^t = Y^t + 0.2$. Initializing $W^t(t = 1,\ldots,\mathcal{T})$ as an $n^t \times h^t$ matrix of ones.
Output: Partitions $\{C^t\}_{t=1}^{\mathcal{T}}$.
repeat
  for $t = 1$ to $\mathcal{T}$ **do**
    for $s = 1$ to $\mathcal{T}$ **do**
      if $s \neq t$ **then**
        Compute the parameter relatedness $G^{(t,s)}$ by Eq.(5).
      end if
    end for
    Compute the model parameter W^t by Eq.(10).
    Compute the cluster indicator Y^t by Eq.(19).
  end for
until Eq.(3) is convergent.

where $\triangle \in \mathbb{R}^{n^t\times h^t}$ is a Lagrangian multiplier. Setting $\frac{\partial L(Y^t)}{\partial Y^t} = 0$, we have

$$\triangle = -2M^tY^t + 2Y^t(Y^t)^TY^t + 2\lambda Y^t - 2\lambda(X^t)^TW^t. \quad (16)$$

According to KKT condition $\triangle_{ij}Y_{ij}^t = 0$, we get

$$(-M^tY^t + Y^t(Y^t)^TY^t + \lambda Y^t - \lambda(X^t)^TW^t)_{ij}Y_{ij}^t = 0. \quad (17)$$

By introducing $Q = (X^t)^TW^t$ and $Q = Q^+ - Q^-$ [Ding *et al.*, 2010], where $Q^+ = (|Q|+Q)/2$ and $Q^- = (|Q|-Q)/2$. Eq.(17) can be rewritten as

$$(-M^tY^t - \lambda Q^+ + Y^t(Y^t)^TY^t + \lambda Y^t + \lambda Q^-)_{ij}Y_{ij}^t = 0. \quad (18)$$

Eq.(18) leads to the following multiplicative update formula

$$Y_{ij}^t \leftarrow Y_{ij}^t\sqrt{\frac{[M^tY^t + \lambda Q^+]_{ij}}{[Y^t(Y^t)^TY^t + \lambda Y^t + \lambda Q^-]_{ij}}}. \quad (19)$$

When initializing the cluster indicator Y^t, we follow the traditional nonnegative matrix factorization methods, i.e., initializing Y^t by the k-means method, then setting $Y^t = Y^t + 0.2$ [Ding *et al.*, 2006].

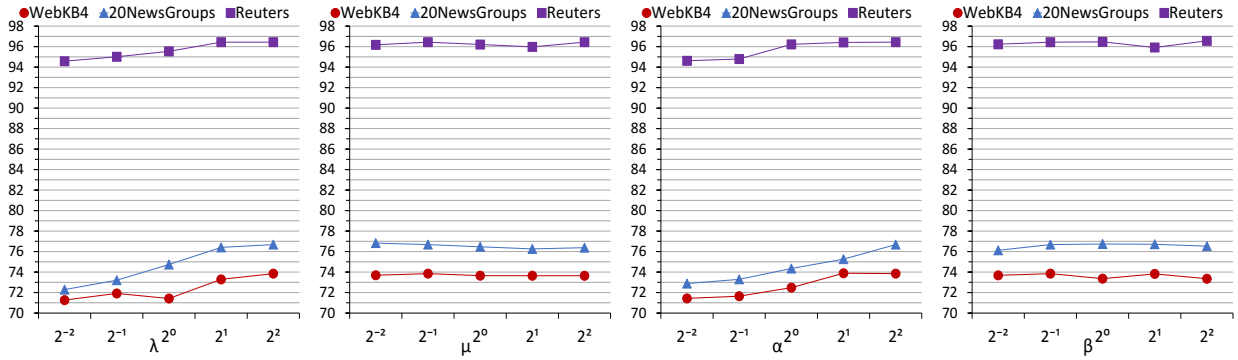
The overall optimizing process of MTCMRL is listed in Algorithm 1. The clustering performance will be affected by different initializations of W^t. Generally we can initialize it as a fixed matrix which is an $n^t \times h^t$ matrix of ones.

3.6 Time Complexity Analysis

Denote n as the sample number in each task, d as the feature number, h^t and h^s as the cluster numbers of the t-th task and the s-th task, $\mathcal{T}$ as the task number, $\tilde{I}$ as the iterations of k-means, $\hat{I}$ as the iterations of MTCMRL. The time complexity of the initialization process for running k-means and computing the similarity matrix M^t is $O(\mathcal{T}\tilde{I}h^tdn + \mathcal{T}n^2d)$. The time complexity of computing $G^{(t,s)}$ is $O(\hat{I}\mathcal{T}^2(h^th^s)^3)$ through quadratic programming, but since the cluster numbers of the tasks are usually much smaller compared with the sample number n and feature number d, optimizing $G^{(t,s)}$ usually does not cost too much time. If the cluster number becomes very larger, we can use the Frank-Wolfe method [Wen *et al.*, 2015] instead of using the quadratic programming. Then the time complexity of computing $G^{(t,s)}$ can be

Data set	Task id	Categories(#Sample)				#Feature
WebKB4	Task 1	C1: Cornell.course(44)	C2: Cornell.faculty(34)	C3: Cornell.project(20)	C4: Cornell.student(127)	2500
	Task 2	C1: Texas.course(38)	C2: Texas.faculty(46)	C3: Texas.project(20)	C4: Texas.student(146)	2500
	Task 3	C1: Washington.course(77)	C2: Washington.faculty(31)	C3: Washington.project(21)	C4: Washington.student(126)	2500
	Task 4	C1: Wisconsin.course(85)	C2: Wisconsin.faculty(42)	C3: Wisconsin.project(25)	C4: Wisconsin.student(154)	2500
20NewsGroups	Task 1	C1: Comp.graphics(387)	C2: Rec.auto(395)	C3: Sci.crypt(395)		3000
	Task 2	C1: Comp.os.ms-win.misc(391)	C2: Rec.motocycle(397)	C3: Sci.electronics(393)	C4: Talk.politic.mideast(376)	3000
	Task 3	C1: Comp.sys.ibm.pc.hw(392)	C2: Rec.sport.baseball(396)	C3: Sci.med(392)		3000
	Task 4	C1: Comp.sys.mac.hw (383)	C2: Rec.sport.hockey(399)	C3: Sci.space(392)	C4: Talk.religion.misc(250)	3000
Reuters	Task 1	C1: Economic_index.gnp(63)	C2: Metal.gold(90)	C3: Food.cocoa(53)		6439
	Task 2	C1: Economic_index.cpi(60)	C2: Energy.nat_gas(33)	C3: Metal.iron_steel(37)		6439
	Task 3	C1: Economic_index.ipi(36)	C2: Metal.copper(44)	C3: Food.coffee(110)		6439

Table 1: Data sets.



(a) Acc(%)

(b) NMI(%)

Figure 1: The average Acc and NMI of MTCMRL under each parameter λ, μ, α and β.

reduced to $O(\hat{I}\mathcal{T}^2h^th^s)$. The time complexity of computing W^t is $O(\hat{I}\mathcal{T}(dh^th^s + dnh^t))$. The time complexity of computing Y^t is $O(\hat{I}\mathcal{T}(n^2h^t + ndh^t))$. Since $h^t, h^s, \mathcal{T}, \tilde{I}, \hat{I}$ are usually much smaller than n and d, the overall time complexity of MTCMRL by omitting them is $O(n^2d)$. This time complexity is the same as that of the traditional clustering method kernel k-means [Dhillon *et al.*, 2004].

4 Experiments

4.1 Methods and Evaluation Metrics

We compare MTCMRL with the single-task clustering methods k-means, symmetric nonnegative matrix factorization (SNMF) [Kuang *et al.*, 2012], constrained Laplacian rank graph-based clustering (CLR) [Nie *et al.*, 2016] and symmetric nonnegative matrix factorization with linear regression model (SNMF+LR), which is a single-task version of MTCMRL with $\alpha = 0$, $\beta = 0$. We also compare MTCMRL with the multi-task clustering methods: the shared subspace learning multi-task clustering (LSSMTC) method [Gu and Zhou, 2009], the smart multi-task Bregman and Kernel clustering (S-MBC and S-MKC) methods [Zhang *et al.*, 2015], the convex discriminative multi-task feature clustering (DMTFC) method [Zhang, 2015], the convex discriminative multi-task relationship clustering (DMTRC) method [Zhang, 2015], the multi-task spectral clustering (MTSC) method [Yang *et al.*, 2015], and the self-adapted multi-task clustering (SAMTC) method [Zhang *et al.*, 2016].

We use two performance measures in [Xu *et al.*, 2003]: clustering accuracy (Acc) and normalized mutual information

Method	Task 1		Task 2		Task 3		Task 4	
	Acc(%)	NMI(%)	Acc(%)	NMI(%)	Acc(%)	NMI(%)	Acc(%)	NMI(%)
k-means	64.00±0.00	18.49±0.01	58.00±5.64	12.11±1.11	51.57±2.67	6.74±3.65	58.27±0.80	20.04±3.24
SNMF	72.89±0.00	37.11±0.00	65.04±3.61	30.55±2.09	71.18±3.34	39.25±2.47	75.42±3.14	46.65±2.25
CLR	58.67	19.35	44.80	15.78	56.47	16.71	66.67	30.56
SNMF+LR	73.33±0.00	37.66±0.00	67.68±0.77	30.18±0.47	71.57±3.64	39.92±3.05	75.39±2.98	46.52±2.13
LSSMTC	63.38±5.18	28.16±4.17	64.36±4.46	23.30±6.97	61.02±5.41	27.74±3.61	65.88±10.16	37.78±6.53
S-MBC	57.91±8.43	24.95±2.80	63.84±7.37	26.97±2.23	57.80±5.38	26.72±3.36	70.85±6.56	39.38±5.45
S-MKC	46.98±2.15	19.81±4.10	45.08±4.51	22.82±4.29	49.25±5.85	23.72±7.21	52.94±3.65	31.38±3.12
DMTFC	**75.11**	38.31	**68.80**	**36.80**	62.35	30.00	71.24	49.38
DMTRC	40.89	13.56	45.60	16.96	60.01	**46.55**	55.88	36.91
MTSC	53.82±0.99	25.84±2.15	58.40±0.00	26.10±0.00	58.20±6.28	28.40±0.22	66.99±0.00	45.22±0.00
SAMTC	66.27±6.04	35.12±6.51	62.96±5.36	32.64±4.97	58.04±6.39	28.65±5.46	72.43±6.80	42.51±5.19
MTCMRL	73.78±0.00	**39.29±0.00**	68.48±0.17	31.99±1.51	**73.45±0.93**	41.53±2.03	**79.67±2.89**	**49.57±2.10**

Table 2: Clustering results on WebKB4.

Method	Task 1		Task 2		Task 3		Task 4	
	Acc(%)	NMI(%)	Acc(%)	NMI(%)	Acc(%)	NMI(%)	Acc(%)	NMI(%)
k-means	33.90±0.00	2.90±0.00	27.26±0.58	7.00±1.58	33.93±0.11	2.89±0.09	28.15±0.14	2.23±0.40
SNMF	80.99±0.00	48.78±0.00	61.84±2.77	41.04±1.11	59.58±0.00	30.02±0.00	83.64±7.55	61.54±10.14
CLR	36.53	7.50	27.42	7.59	34.49	4.81	28.72	5.83
SNMF+LR	81.05±0.13	48.70±0.52	62.94±0.33	41.46±0.65	60.15±2.89	29.94±1.20	83.68±7.61	61.83±10.40
S-MBC	44.06±4.46	14.61±11.98	46.64±5.75	22.51±5.52	43.93±0.89	19.45±3.52	70.88±9.52	40.61±8.56
S-MKC	74.26±1.97	38.53±3.32	63.45±1.98	41.70±3.43	62.83±3.16	30.63±3.44	64.47±5.30	37.19±5.05
MTSC	76.72±0.00	45.44±0.00	62.48±0.08	42.29±0.21	61.53±0.24	38.62±4.44	60.06±0.03	43.87±0.01
SAMTC	68.21±10.29	40.10±7.61	58.16±6.55	41.55±3.22	67.14±9.64	45.54±9.33	68.18±9.04	50.16±8.12
MTCMRL	**82.07±0.00**	**49.80±0.00**	**65.52±0.10**	**43.42±0.15**	**77.97±0.00**	**45.94±0.00**	**85.96±0.00**	**63.19±0.00**

Table 3: Clustering results on 20NewsGroups.

Method	Task 1		Task 2		Task 3	
	Acc(%)	NMI(%)	Acc(%)	NMI(%)	Acc(%)	NMI(%)
k-means	73.16±0.46	44.70±0.78	64.62±3.42	41.59±5.71	45.89±1.33	21.48±1.27
SNMF	**97.57±0.00**	**89.49±0.00**	94.77±8.17	85.74±10.83	93.21±3.16	77.86±5.00
CLR	79.61	64.31	69.23	51.97	75.79	54.62
SNMF+LR	**97.57±0.00**	**89.49±0.00**	94.08±8.80	84.23±10.49	93.32±2.83	77.98±4.64
LSSMTC	91.55±6.87	79.32±9.58	90.31±6.13	75.79±9.48	72.53±6.66	53.93±4.11
S-MBC	89.47±9.99	76.48±13.64	92.54±5.89	79.71±7.48	75.58±7.64	55.66±6.16
S-MKC	95.49±1.73	83.85±4.50	91.23±2.06	75.58±5.58	76.21±4.88	53.69±6.05
DMTFC	94.66	83.69	92.31	73.25	72.63	38.99
DMTRC	72.82	30.50	81.54	64.08	61.05	53.17
MTSC	**97.57±0.00**	**89.49±0.00**	96.15±0.00	85.00±0.00	90.16±8.55	74.04±11.05
SAMTC	96.94±3.38	89.45±7.35	96.69±0.73	86.35±2.54	90.21±9.27	76.05±13.12
MTCMRL	**97.57±0.00**	**89.49±0.00**	**97.00±1.93**	**88.67±5.21**	**94.74±0.00**	**80.58±0.00**

Table 4: Clustering results on Reuters.

(NMI) to evaluate the clustering performance.

4.2 Data Sets

WebKB4[1]: This data set contains web pages collected from computer science department websites at 4 universities: Cornell, Texas, Washington and Wisconsin. They are divided into 7 categories, we choose 4 most populous categories such as course, faculty, project and student for clustering.

20NewsGroups[2]: This data set consists of the news documents under 20 categories, we choose 4 most populous root categories such as comp, rec, sci and talk for clustering.

Reuters[3]: This data set is composed of the news documents under 135 categories from the Reuters newswire, we choose 4 most populous root categories such as economic index, energy, food and metal for clustering.

There are three typical cases for multi-task data setting. The first case is that the tasks are completely related (i.e., the tasks share the same categories), we use the WebKB4 data set to represent this case. The second case is that the tasks are partially related (i.e., the tasks share parts of categories) and the cluster numbers in all the tasks are not the same, we use the 20NewsGroups data set to represent this case. The third case is that the tasks are partially related and the cluster numbers in all the tasks are the same, we use the Reuters data set to represent this case. The details of constitutions of these data sets are shown in Table 1.

4.3 Parameter Investigation

We investigate the impact of the parameters λ, μ, α and β on the clustering performance for MTCMRL. For each data set, we repeat the MTCMRL method 10 times by setting one parameter to search the grid and fixing the other parameters, then compute the average Acc and NMI of all the tasks under this parameter. More specifically, we successively set one parameter to search the grid $\{2^{-2}, 2^{-1}, 2^0, 2^1, 2^2\}$ by fixing $\lambda, \alpha = 2^2$ and $\mu, \beta = 2^{-1}$. From the parameter investigation in Figure 1, it can be seen that MTCMRL is more sensitive to the linear regression model trade-off parameter λ and the cross-task relatedness learning trade-off parameter α than their corresponding regularization parameters μ and β. The span of clustering performance with respect to λ and α is $2\% \sim 5\%$, whereas the span of clustering performance

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

[2] http://qwone.com/∼jason/20Newsgroups/

[3] http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html

with respect to μ and β is less than 2%. λ and α control the main body of the linear regression model and the cross-task relatedness learning, thus playing major roles in optimizing the variables.

4.4 Parameter Setting

We exploit the grid searching method which is commonly used in multi-task clustering [Gu and Zhou, 2009] to identify the optimal parameters. Based on the parameter investigation, for MTCMRL, we set λ and α to search the grid $\{2^{-2}, 2^{-1}, 2^0, 2^1, 2^2\}$, $\mu = 2^{-1}$, $\beta = 2^{-1}$, and we choose cosine similarity to compute the similarity matrix. For CLR, the neighborhood size is searched from $\{10, 20, \ldots, 100\}$. For LSSMTC, λ is set by searching the grid $\{0.1, 0.2, \ldots, 0.9\}$, the dimensionality of the shared subspace is set by searching the grid $\{2, 4, 6, 8, 10\}$. For S-MBC and S-MKC, λ is set by searching the grid $\{0.1, 0.2, \ldots, 1\}$. For DMTFC and DMTRC, λ_1 and λ_2 are both set by searching the grid $\{2^{-10}, 2^{-8}, \ldots, 2^{-2}\}$. For MTSC, α and β are both searched from $\{2^{-2}, 2^{-1}, 2^0, 2^1, 2^2\}$. For SAMTC, the number of nearest neighbors within each task and across the other tasks are both searched from $\{30, 60, 90, 120\}$, and we choose cosine similarity to compute the nearest neighbors.

4.5 Clustering Results

As CLR, DMTFC and DMTRC are convex optimization methods, we perform them once under each parameter setting, and show the clustering results under the best parameter setting. For the other methods, we repeat each method 10 times under each parameter setting, and show the mean clustering results and the standard deviations under the best parameter setting. As LSSMTC, DMTFC and DMTRC can only apply to the case that the cluster numbers of all the tasks are the same, we perform them on WebKB4 and Reuters. We report the clustering results in Table 2, Table 3 and Table 4, and it can be seen that:

1. MTCMRL performs better than the single-task clustering methods, since MTCMRL exploits the information from the related tasks, whereas the single-task clustering methods only utilize the information within each task. SNMF and SNMF+LR have the same clustering performance with MTCMRL on the Task 1 of Reuters, but they perform worse than MTCMRL on the Task 2 and Task 3 of Reuters. Because SNMF and SNMF+LR perform the same on the Task 1 of Reuters, showing that the linear regression model can not affect the clustering performance. Therefore, updating the model parameter by cross-task relatedness learning does not help improve the clustering performance on the Task 1 of Reuters.

2. LSSMTC performs worse than MTCMRL, because it assumes that there exists a shared subspace in which all the tasks share the same centroids, which is too idealized and restrictive. Moreover, LSSMTC may suffer from negative transfer when the tasks do not share the same categories like the tested data set Reuters.

3. S-MBC and S-MKC perform worse than MTCMRL, because they require the distributions of the tasks to be the same or similar, but the task distributions in the tested data sets do not show very high similarity.

4. DMTFC performs worse than MTCMRL in most cases, because DMTFC requires that the tasks share the same categories, which is seldom satisfied. As the tested data set WebKB4 meets this requirement and DMTFC is a convex optimization method, DMTFC performs the best on the Task 2 of WebKB4, whereas MTCMRL performs the second best.

5. Although DMTRC can automatically learn the task relatedness, its clustering performance is not so good as we expected. Because it assumes that the label marginal distribution in each task is evenly distributed, which is too strict.

6. MTSC performs worse than MTCMRL, except for the Task 1 of Reuters on which MTSC and MTCMRL show the same clustering performance. Because it manually controls the degree of sharing the lower dimensional feature space among the tasks, and it does not explicitly consider how to set the intertask correlation from the tasks themselves.

7. SAMTC performs worse than MTCMRL, because it only transfers the instance knowledge between each pair of subtasks, which may lose the useful instance knowledge not in the subtasks.

8. The reasons stated above are also why LSSMTC, S-MBC, S-MKC, DMTFC, DMTRC, MTSC and SAMTC sometimes even perform worse than the single-task clustering methods.

9. MTCMRL performs much better than the compared multi-task clustering methods in most cases, because MTCMRL can automatically learn the model parameter relatedness between each pair of tasks, which can exploit the positive relatedness among the tasks and avoid negative transfer. An exception is that for the Task 2 of WebKB4, MTCMRL performs the second best, whereas DMTFC performs the best, because DMTFC is a convex optimization method, and the WebKB4 data set happens to meet the requirement of DMTFC that the tasks share the same categories. Among the existing methods, DMTRC and SAMTC can also automatically learn the task relatedness. But DMTRC is based on a strict assumption that the label marginal distribution in each task distributes evenly and the tasks have the same cluster number, which is seldom satisfied. SAMTC only transfers the instance knowledge between each pair of subtasks, which may miss some potentially useful information that are not in the subtasks. These limitations will degrade their clustering performance.

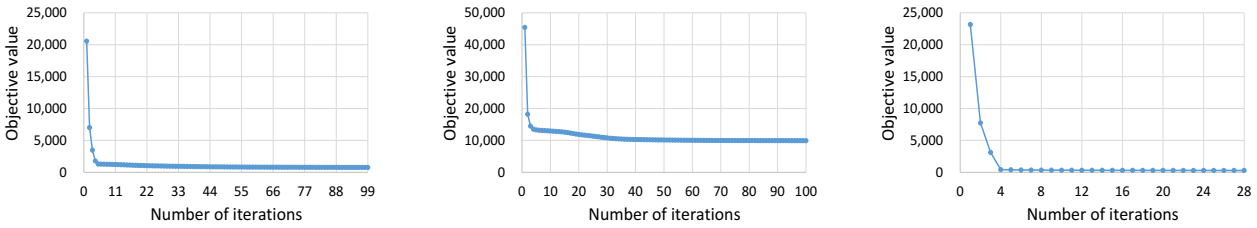
Summarizing the above discussions, it can be concluded that MTCMRL performs the best among the compared methods.

4.6 Cluster Model Parameter Relatedness

For each data set with $\mathcal{T}$ tasks, there are $\mathcal{T}(\mathcal{T}-1)$ cluster model parameter relatedness matrices $G^{(t,s)}(t, s = 1, \ldots, \mathcal{T}, s \neq t)$. We show the cluster model parameter relatedness matrix $G^{(1,2)}$ between Task 1 and Task 2 in the WebKB4, 20NewsGroups and Reuters data sets, respectively.

For WebKB4, the learnt cluster model parameter relatedness matrix $G^{(1,2)}$ is

$$G^{(1,2)} = \begin{bmatrix} 0.0816 & 0.0675 & 0.0823 & 0.0000 \\ 0.0860 & 0.1231 & 0.0907 & 0.0305 \\ 0.1002 & 0.0903 & 0.1404 & 0.0000 \\ 0.0000 & 0.0201 & 0.0203 & 0.0670 \end{bmatrix}.$$



(a) WebKB4 (b) 20NewsGroups (c) Reuters

Figure 2: The convergence curves of the WebKB4, 20NewsGroups and Reuters data sets.

For 20NewsGroups, the learnt cluster model parameter relatedness matrix $G^{(1,2)}$ is

$$G^{(1,2)} = \begin{bmatrix} 0.1730 & 0.0000 & 0.1236 & 0.0699 \\ 0.0000 & 0.1507 & 0.1321 & 0.0000 \\ 0.1365 & 0.0000 & 0.1923 & 0.0219 \end{bmatrix}.$$

For Reuters, the learnt cluster model parameter relatedness matrix $G^{(1,2)}$ is

$$G^{(1,2)} = \begin{bmatrix} 0.1957 & 0.1005 & 0.0899 \\ 0.0788 & 0.1259 & 0.2013 \\ 0.0000 & 0.1075 & 0.1004 \end{bmatrix}.$$

A larger $G_{ij}^{(1,2)}$ means that cluster i in Task 1 and cluster j in Task 2 have a higher relatedness. Note that the cluster labels have been mapped to their true labels in the tested data sets through the permutation mapping function in the clustering accuracy performance measure [Gu *et al.*, 2011]. Generally the clusters with the same label between Task 1 and Task 2 have the highest relatedness, e.g., for the 20NewsGroups data set, cluster 1 with the label "Comp.graphics" in Task 1 has the highest relatedness with cluster 1 with the label "Comp.os.ms-win.misc" in Task 2 (0.1730 is the biggest value in the first row of $G^{(1,2)}$). The same phenomenon can be observed for the clusters with the same label between the other pairs of tasks, we omitted these matrices due to space limitation.

4.7 Convergence Curves

The convergence criterion of the MTCMRL algorithm is to set a convergence threshold, which is 1 in the experiments. Once the objective value of Eq.(3) in the previous step minus that in the current step is smaller than 1, Algorithm 1 will converge. In the tested data sets, Algorithm 1 will converge after 99, 100 and 28 iterations for WebKB4, 20NewsGroups and Reuters, respectively. The convergence curves are shown in Figure 2.

4.8 Running Time of Cluster Model Parameter Relatedness Learning

In this section, we investigate the running time of learning the cluster model parameter relatedness $G^{(t,s)}$ under different cluster number settings. More specifically, we select Task 1 and Task 2 in 20NewsGroups as the tested data set, and set the cluster numbers h^1 and h^2 ($h^1 = h^2$) in Task 1 and Task 2 by searching the grid $\{5, 15, 25, 35, 45, 55, 65, 75, 85, 95\}$. Then

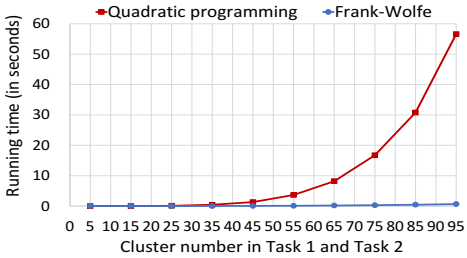


Figure 3: The running time of learning $G^{(1,2)}$ by using the quadratic programming method and the Frank-Wolfe method.

we use the quadratic programming method and the Frank-Wolfe method to learn $G^{(1,2)}$, respectively. The running time of learning $G^{(1,2)}$ under different cluster number settings is shown in Figure 3.

From Figure 3, it can be seen that the running time of learning $G^{(1,2)}$ by quadratic programming rises sharply when the cluster number is larger than 35. Whereas the running time of learning $G^{(1,2)}$ by the Frank-Wolfe method rises slowly with the increase of cluster number. Based on this observation, when the cluster number is smaller than 35, we can use quadratic programming to learn the cluster model parameter relatedness $G^{(t,s)}$, otherwise we will use the Frank-Wolfe method to learn $G^{(t,s)}$.

5 Conclusion

In this paper, we have proposed a multi-task clustering with model relation learning (MTCMRL) method, which automatically learns the model parameter relatedness between each pair of tasks. MTCMRL first introduces the linear regression model into symmetric nonnegative matrix factorization, then learns the model parameter relatedness between the clusters in each pair of tasks by constructing a similarity matrix. We further present an effective alternating algorithm to solve the non-convex optimization problem. Experimental results on several real data sets show the superiority of the proposed method over traditional single-task clustering methods and existing multi-task clustering methods.

Acknowledgments

This work was supported by National Science Foundation of China (No. 61632019).

References

[Al-Stouhi and Reddy, 2014] Samir Al-Stouhi and Chandan K. Reddy. Multi-task clustering using constrained symmetric non-negative matrix factorization. In *SDM*, pages 785–793, 2014.

[Appice and Malerba, 2016] Annalisa Appice and Donato Malerba. A co-training strategy for multiple view clustering in process mining. *IEEE Trans. Services Computing*, 9(6):832–845, 2016.

[Blum and Mitchell, 1998] Avrim Blum and Tom M. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, pages 92–100, 1998.

[Boyd and Vandenberghe, 2004] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, UK, 2004.

[Ceci *et al.*, 2015] Michelangelo Ceci, Gianvito Pio, Vladimir Kuzmanovski, and Saso Dzeroski. Semi-supervised multi-view learning for gene network reconstruction. *PLos One*, 10(12):e0144031, 2015.

[Dhillon *et al.*, 2004] Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means, spectral clustering and normalized cuts. In *KDD*, pages 551–556, 2004.

[Ding and He, 2005] Chris H. Q. Ding and Xiaofeng He. On the equivalence of nonnegative matrix factorization and spectral clustering. In *SDM*, pages 606–610, 2005.

[Ding *et al.*, 2006] Chris H. Q. Ding, Tao Li, Wei Peng, and Haesun Park. Orthogonal nonnegative matrix t-factorizations for clustering. In *KDD*, pages 126–135, 2006.

[Ding *et al.*, 2010] Chris H. Q. Ding, Tao Li, and Michael I. Jordan. Convex and semi-nonnegative matrix factorizations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(1):45–55, 2010.

[Gu and Zhou, 2009] Quanquan Gu and Jie Zhou. Learning the shared subspace for multi-task clustering and transductive transfer classification. In *ICDM*, pages 159–168, 2009.

[Gu *et al.*, 2011] Quanquan Gu, Zhenhui Li, and Jiawei Han. Learning a kernel for multi-task clustering. In *AAAI*, pages 368–373, 2011.

[Hotelling, 1936] Harold Hotelling. Relations between two sets of variables. *Biometrika*, 28:312–377, 1936.

[Kuang *et al.*, 2012] Da Kuang, Haesun Park, and Chris H. Q. Ding. Symmetric nonnegative matrix factorization for graph clustering. In *SDM*, pages 106–117, 2012.

[Kumar *et al.*, 2011] Abhishek Kumar, Piyush Rai, and Hal Daumé III. Co-regularized multi-view spectral clustering. In *NIPS*, pages 1413–1421, 2011.

[Minh *et al.*, 2016] Ha Quang Minh, Loris Bazzani, and Vittorio Murino. A unifying framework in vector-valued reproducing kernel hilbert spaces for manifold regularization and co-regularized multi-view learning. *Journal of Machine Learning Research*, 17:25:1–25:72, 2016.

[Murugesan *et al.*, 2017] Keerthiram Murugesan, Jaime G. Carbonell, and Yiming Yang. Co-clustering for multitask learning. *CoRR*, abs/1703.00994, 2017.

[Nie *et al.*, 2016] Feiping Nie, Xiaoqian Wang, Michael I. Jordan, and Heng Huang. The constrained laplacian rank algorithm for graph-based clustering. In *AAAI*, pages 1969–1976, 2016.

[Pan and Yang, 2010] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, 2010.

[Podosinnikova *et al.*, 2016] Anastasia Podosinnikova, Francis R. Bach, and Simon Lacoste-Julien. Beyond CCA: moment matching for multi-view models. In *ICML*, pages 458–467, 2016.

[Wen *et al.*, 2015] Junfeng Wen, Russell Greiner, and Dale Schuurmans. Correcting covariate shift with the frank-wolfe algorithm. In *IJCAI*, pages 1010–1016, 2015.

[Xie *et al.*, 2012] Saining Xie, Hongtao Lu, and Yangcheng He. Multi-task co-clustering via nonnegative matrix factorization. In *ICPR*, pages 2954–2958, 2012.

[Xu *et al.*, 2003] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *SIGIR*, pages 267–273, 2003.

[Yang *et al.*, 2015] Yang Yang, Zhigang Ma, Yi Yang, Feiping Nie, and Heng Tao Shen. Multitask spectral clustering by exploring intertask correlation. *IEEE Trans. Cybernetics*, 45(5):1069–1080, 2015.

[Zhang and Yang, 2017] Yu Zhang and Qiang Yang. A survey on multi-task learning. *CoRR*, abs/1707.08114, 2017.

[Zhang and Zhang, 2010] Jianwen Zhang and Changshui Zhang. Multitask Bregman clustering. In *AAAI*, pages 655–660, 2010.

[Zhang and Zhou, 2012] Zhihao Zhang and Jie Zhou. Multi-task clustering via domain adaptation. *Pattern Recognition*, 45(1):465–473, 2012.

[Zhang *et al.*, 2015] Xianchao Zhang, Xiaotong Zhang, and Han Liu. Smart multitask Bregman clustering and multitask Kernel clustering. *ACM Trans. Knowl. Disc. Data*, 10(1):8, 2015.

[Zhang *et al.*, 2016] Xianchao Zhang, Xiaotong Zhang, and Han Liu. Self-adapted multi-task clustering. In *IJCAI*, pages 2357–2363, 2016.

[Zhang *et al.*, 2017] Xiaotong Zhang, Xianchao Zhang, Han Liu, and Xinyue Liu. Multi-task clustering through instances transfer. *Neurocomputing*, 251:145–155, 2017.

[Zhang, 2015] Xiao-Lei Zhang. Convex discriminative multitask clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(1):28–40, 2015.

[Zhao *et al.*, 2017] Jing Zhao, Xijiong Xie, Xin Xu, and Shiliang Sun. Multi-view learning overview: Recent progress and new challenges. *Information Fusion*, 38:43–54, 2017.