# Transfer Spectral Clustering

Wenhao Jiang and Fu-lai Chung

Department of Computing, Hong Kong Polytechnic University,
Hunghom, Kowloon, Hong Kong
{cswhjiang, cskchung}@comp.polyu.edu.hk

**Abstract.** Transferring knowledge from auxiliary datasets has been proved useful in machine learning tasks. Its adoption in clustering however is still limited. Despite of its superior performance, spectral clustering has not yet been incorporated with knowledge transfer or transfer learning. In this paper, we make such an attempt and propose a new algorithm called transfer spectral clustering (TSC). It involves not only the data manifold information of the clustering task but also the feature manifold information shared between related clustering tasks. Furthermore, it makes use of co-clustering to achieve and control the knowledge transfer among tasks. As demonstrated by the experimental results, TSC can greatly improve the clustering performance by effectively using auxiliary unlabeled data when compared with other state-of-the-art clustering algorithms.

**Keywords:** Transfer Learning, Spectral Clustering, Co-clustering.

## 1 Introduction

Clustering aims at finding groups of objects so that the objects in the same group are relatively similar while the objects in different group are relatively dissimilar. In the past decades, many clustering algorithms have been proposed, such as $k$-means clustering [1], spectral clustering [2, 3], Bregman divergence based clustering [4], etc. Focused on improving the clustering performance, prior knowledge in the form of must-link or cannot-link constraints [5] and auxiliary labeled data [6] have been introduced in clustering. Recently, auxiliary unlabeled data were also used to improve the performance of clustering by the so-called self-taught clustering (STC) [7]. To the best of our knowledge, STC is the first method that transfers knowledge from unlabeled data to facilitate more effective clustering. Its merit has been demonstrated in image clustering tasks.

Spectral clustering algorithms [2, 3] are well-known methods that use manifold information contained in the sample distribution to carry out the clustering task and very often outperform the traditional clustering algorithms such as the $k$-means algorithm. However, it seems that how to transfer knowledge from auxiliary unlabeled dataset to facilitate more effective spectral clustering has not yet been explored. In this paper, a transfer spectral clustering (TSC) algorithm is proposed and it works on the assumption that the related tasks share the same

low dimensional feature embedding. This assumption could be measured by the objective of bipartite graph co-clustering. The proposed algorithm involves not only the data manifold information of the individual task but also the feature manifold information shared between tasks. The experimental results show that TSC can greatly improve the clustering performance by effectively using auxiliary unlabeled data.

This work is presented as follows. In section 2, the related works are firstly highlighted. The formulation of our method is given in section 3 which also describes the corresponding optimization method. In section 4, the experimental results are reported. In section 5, we give the conclusions and discuss the future works.

## 2 Related Works

Our method is related to co-clustering, transfer learning and multitask clustering with the corresponding related works highlighted below.

### 2.1 Co-clustering

Co-clustering aims at performing clustering on both the samples and the attributes so that clustering of attributes can help improve the clustering quality of samples. Many co-clustering algorithms have been proposed, e.g. information theoretic co-clustering (ITCC) [8] and bipartite graph co-clustering [9]. ITCC is a co-clustering algorithm based on information theory, which attempts to find the co-clusters such that the mutual information between the clustered random variables is maximized subject to constraints on the number of row and column clusters. As a totally different method, co-clustering on bipartite graph [9] expresses samples and attributes by a bipartite graph and seeks minimum cuts on this graph such that samples and attributes are both well grouped. A bipartite graph based co-clustering will be exploited by our new method to be presented in the next section.

### 2.2 Transfer Learning

Transfer learning [10–12] attempts to improve the learning performance on a target dataset by utilizing auxiliary datasets. It has been proved to be beneficial in practice [13, 14]. Many works have been done on inductive transfer learning [15] and transductive transfer learning [16–18].

For clustering problems, Dai et al. [7] proposed self-taught clustering (STC) to cluster a small collection of target data with the help of a large amount of unlabeled auxiliary data. STC extends the information theoretic co-clustering algorithm (ITCC) [8] with the assumption that target dataset and auxiliary dataset share the same feature clustering. STC minimizes the same loss with ITCC for the two datasets simultaneously. In this paper, we propose a method based on a similar assumption. However, unlike STC which built on information theory, our method is built on graphs.

### 2.3 Multitask Clustering

Multitask learning [19–21] performs multiple learning tasks concurrently to improve the individual performance. But almost all existing works have been focused on supervised settings. Recently, multitask Bregman clustering (MBC) [22] was proposed to extend Bregman divergence based clustering [4] to multitask settings. MBC aims at minimizing a local loss function for each single task and carries out a task regularization involving all tasks. The task regularization of MBC is indeed the sum of divergence between two learned density models for any pair of different tasks.

## 3 Transfer Spectral Clustering

### 3.1 Problem Formulation

Let us first describe the clustering tasks for our transfer spectral clustering (TSC) algorithm. Given two data matrix $X^{(1)} = [x_1^{(1)}, \cdots, x_{n_1}^{(1)}]$ and $X^{(2)} = [x_1^{(2)}, \cdots, x_{n_2}^{(2)}]$ containing $n_1$ and $n_2$ samples with $n_f$ features, where $x_i^{(1)} \in R^{n_f \times n_1}$ and $x_j^{(2)} \in R^{n_f \times n_2}$, TSC aims at performing clustering on both datasets simultaneously and achieving better performance than clustering them separately. Assume that the number of clusters on both datasets is the same, and is denoted as $k$. To achieve better performance, we try to find low dimensional embeddings for these two datasets so that they not only are smooth on the data manifold, like that in spectral clustering [2], but also maximize the task relationship measured by co-clustering objective. With the low dimensional embeddings obtained, the traditional clustering algorithms could be applied to get the partitions for both datasets.

Let $G^{(1)}$ and $G^{(2)}$ denote the $k$-nn graphs constructed for each task separately. The corresponding affinity matrices $W^{(1)}$ and $W^{(2)}$ are defined as

$$W_{ij}^{(1)} = \begin{cases} 1 \text{ if } x_i^{(1)} \in \mathcal{N}^{(1)}(x_j^{(1)}) \text{ or } x_j^{(1)} \in \mathcal{N}^{(1)}(x_i^{(1)}) \\ 0 \qquad\qquad\qquad\qquad \text{otherwise,} \end{cases} \tag{1}$$

$$W_{ij}^{(2)} = \begin{cases} 1 \text{ if } x_i^{(2)} \in \mathcal{N}^{(2)}(x_j^{(2)}) \text{ or } x_j^{(2)} \in \mathcal{N}^{(2)}(x_i^{(2)}) \\ 0 \qquad\qquad\qquad\qquad \text{otherwise,} \end{cases} \tag{2}$$

where $\mathcal{N}^{(1)}(x_j^{(1)})$ and $\mathcal{N}^{(2)}(x_j^{(2)})$ denote the set of neighbors of $x_j^{(1)}$ and $x_j^{(2)}$ respectively. The first part of our goal is to obtain embeddings that are smooth over the corresponding graph. Let $F^{(1)} \in R^{n_1 \times k}$ and $F^{(2)} \in R^{n_2 \times k}$ denote the embeddings, where each row is an embedding for the corresponding sample. The smoothness of $F^{(1)}$ on $G^{(1)}$ can be measured by

$$\frac{1}{2} \sum_{i,j=1}^{n_1} W_{ij}^{(1)} \| \frac{1}{\sqrt{D_{ii}^{(1)}}} F_i^{(1)} - \frac{1}{\sqrt{D_{jj}^{(1)}}} F_j^{(1)} \|^2, \tag{3}$$

where $F_i^{(1)}$ is the $i$th row of $F^{(1)}$ and $D^{(1)} = diag(W^{(1)}\mathbf{1})$. It can be easily shown that equation (3) can be simplified as

$$\mathrm{tr}({F^{(1)}}^T(I - W_N^{(1)})F^{(1)}),$$

where

$$W_N^{(1)} = {D^{(1)}}^{-\frac{1}{2}} W^{(1)} {D^{(1)}}^{-\frac{1}{2}}.$$

If the constraint ${F^{(1)}}^T F^{(1)} = I$ is added, the smoothness can then be measured by

$$\mathrm{tr}({F^{(1)}}^T W_N^{(1)} F^{(1)}). \tag{4}$$

Larger value means the embeddings being smoother on the graph. Similarly, for $F^{(2)}$, the smoothness on $G^{(2)}$ can be measured by

$$\mathrm{tr}({F^{(2)}}^T W_N^{(2)} F^{(2)}), \tag{5}$$

where

$$W_N^{(2)} = {D^{(2)}}^{-\frac{1}{2}} W^{(2)} {D^{(2)}}^{-\frac{1}{2}}$$

and

$$D^{(2)} = diag(W^{(2)}\mathbf{1}).$$

In addition to pursuing smoothness on the corresponding graphs, TSC also aims at linking the two embeddings together such that one embedding obtained in one data set facilitates the finding of the embedding in another dataset. This goal can be achieved by graph co-clustering as follows.

Recall that the purpose of graph based co-clustering [9] is to find minimum cuts on a bipartite graph whose nodes include samples and features. Given the data matrix $A \in R^{d\times n}$, where $d$ is the number of features and $n$ is the number of samples. The affinity matrix for bipartite graph is defined as

$$W = \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix}$$

and the corresponding graph Laplacian is defined as

$$L = \begin{bmatrix} D_1 & -A \\ -A^T & D_2 \end{bmatrix}$$

where $D_1 = diag(A\mathbf{1})$ and $D_2 = diag(A^T\mathbf{1})$. The objective function of co-clustering [9] could be expressed as

$$\begin{aligned} \min_z \ & z^T L z \\ \text{s.t. } & x^T D_1 x = 1, \\ & y^T D_2 y = 1, \end{aligned} \tag{6}$$

where

$$z = \begin{bmatrix} x \\ y \end{bmatrix}$$

with vector $x$ containing the embeddings for features, and vector $y$ containing the embeddings for samples. It can be easily shown that such problem is equivalent to

$$\begin{aligned} \max_{x,y} \; & x^T A y \\ \text{s.t. } & x^T D_1 x = 1, \\ & y^T D_2 y = 1. \end{aligned} \tag{7}$$

Motivated by graph co-clustering, the second part of our objective function can be defined as

$$\begin{aligned} & \Omega(F^{(1)}, F^{(2)}, F^{(3)}) \\ & = \mathrm{tr}({F^{(3)}}^T X^{(1)} F^{(1)}) + \mathrm{tr}({F^{(3)}}^T X^{(2)} F^{(2)}), \end{aligned} \tag{8}$$

where the rows of matrix $F^{(3)}$ are the embeddings for features. Larger value of the above equation means better co-clustering on both datasets which share the same feature clustering. In our TSC algorithm with this measurement, we only consider normalized data matrices which are defined as

$$X_N^{(1)} = {D_1^{(1)}}^{-\frac{1}{2}} X^{(1)} {D_2^{(1)}}^{-\frac{1}{2}}, \tag{9}$$

$$X_N^{(2)} = {D_1^{(2)}}^{-\frac{1}{2}} X^{(2)} {D_2^{(2)}}^{-\frac{1}{2}}, \tag{10}$$

where

$$\begin{aligned} D_1^{(1)} &= diag(X^{(1)}\mathbf{1}), \\ D_2^{(1)} &= diag({X^{(1)}}^T\mathbf{1}), \\ D_1^{(2)} &= diag(X^{(2)}\mathbf{1}), \\ D_2^{(2)} &= diag({X^{(2)}}^T\mathbf{1}). \end{aligned}$$

Therefore, by combining these two goals, i.e., to obtain smooth embeddings on the data manifold and to maximize the co-cluster objective, our objective function can be expressed as

$$\begin{aligned} \max_{F^{(1)},F^{(2)},F^{(3)}} \; & \mathrm{tr}({F^{(1)}}^T W_N^{(1)} F^{(1)}) + \mathrm{tr}({F^{(2)}}^T W_N^{(2)} F^{(2)}) + \\ & \lambda(\mathrm{tr}({F^{(3)}}^T X_N^{(1)} F^{(1)}) + \mathrm{tr}({F^{(3)}}^T X_N^{(2)} F^{(2)})) \\ \text{s.t. } & {F^{(1)}}^T F^{(1)} = I, \\ & {F^{(2)}}^T F^{(2)} = I, \\ & {F^{(3)}}^T F^{(3)} = I, \end{aligned} \tag{11}$$

where $\lambda > 0$ is the trade off between the smoothness on graphs and the co-clustering objective. If $\lambda$ is set as 0, our method becomes the classical spectral clustering.

### 3.2 Another View of the Formulation

In this section, we provide another view of our formulation. Let us define a graph $\tilde{G}$ whose schematic diagram is shown in Figure 1. And the corresponding affinity matrix is defined as

$$\tilde{W} = \begin{bmatrix} W_N^{(1)} & 0 & \frac{\lambda}{2} {X_N^{(1)}}^T \\ 0 & W_N^{(2)} & \frac{\lambda}{2} {X_N^{(2)}}^T \\ \frac{\lambda}{2} X_N^{(1)} & \frac{\lambda}{2} X_N^{(2)} & 0 \end{bmatrix}.$$

We can see that the left part and right part of $\tilde{G}$ are actually $G^{(1)}$ and $G^{(2)}$ respectively, and they are linked by features. If we define

$$\tilde{F} = \begin{bmatrix} F^{(1)} \\ F^{(2)} \\ F^{(3)} \end{bmatrix},$$

it can be shown that $\mathrm{tr}(\tilde{F}^T \tilde{W} \tilde{F})$ is actually the objective function of problem (11). Hence, our formulation is to find the representation for features and samples based on this graph. Our definition is different from the graphs defined in EigenTransfer [11]. $\tilde{G}$ does not include label information and considers direct relations between samples within tasks. Moreover, one might control the quantity of knowledge transferred by the parameter $\lambda$ introduced. As a result, it is more suitable for clustering tasks.

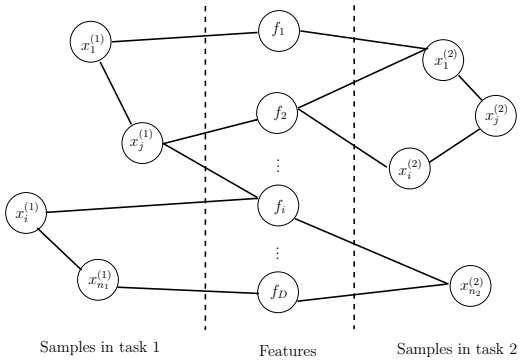



**Fig. 1.** Another View of TSC

### 3.3 Algorithm

Now, we present the algorithm to find solutions for our formulation. Problem (11) is not a convex optimization problem and hence, only a local solution can be obtained. Let us first initialize $F^{(1)}$ and $F^{(2)}$ with the top $k$ eigenvectors of $W_N^{(1)}$ and $W_N^{(2)}$ respectively, and $F^{(3)}$ with the top $k$ left singular vectors of the matrix normalized from

$$X = \left[ X_N^{(1)} \; X_N^{(2)} \right]$$

in accordance with (9) and (10). Then, they are updated such that the value of the objective function increases until convergence is achieved.

Because of the orthonormality constraints, $F_1$, $F_2$ and $F_3$ actually are on the Stiefel manifold [23]. The problem (11) could be solved by repeatedly updating $F_1$, $F_2$ and $F_3$ on the manifold alternatively. During iterations, if $F_1$, $F_2$ and $F_3$ are not on the manifold, they are updated with their projections on the manifold.

For a rank $p$ matrix $Z \in R^{n \times p}$, the projection of $Z$ on Stiefel manifold is defined as

$$\pi(Z) = \arg \min_{Q^T Q = I} \|Z - Q\|_F^2. \tag{12}$$

If the singular value decomposition (SVD) of $Z$ is $Z = U \Sigma V^T$, the projection could be computed as $\pi(Z) = U I_{n,p} V^T$ [23]. Hence, we can update $F_1$, $F_2$ and $F_3$ by moving them in the direction of increasing the value of the objective function. For convenience of descriptions, we denote the objective function as

$$\begin{aligned} & g(F^{(1)}, F^{(2)}, F^{(3)}) \\ =& \mathrm{tr}({F^{(1)}}^T W_N^{(1)} F^{(1)}) + \mathrm{tr}({F^{(2)}}^T W_N^{(2)} F^{(2)}) + \\ & \lambda(\mathrm{tr}({F^{(3)}}^T X_N^{(1)} F^{(1)}) + \mathrm{tr}({F^{(3)}}^T X_N^{(2)} F^{(2)})). \end{aligned} \tag{13}$$

Hence, the partial derivatives of $g$ with respect to $F^{(1)}$, $F^{(2)}$ and $F^{(3)}$ can be computed as

$$\frac{\partial g}{\partial F^{(1)}} = 2 W_N^{(1)} F^{(1)} + \lambda {X_N^{(1)}}^T F^{(3)}, \tag{14}$$

$$\frac{\partial g}{\partial F^{(2)}} = 2 W_N^{(2)} F^{(2)} + \lambda {X_N^{(2)}}^T F^{(3)}, \tag{15}$$

$$\frac{\partial g}{\partial F^{(3)}} = \lambda X_N^{(1)} F^{(1)} + \lambda X_N^{(2)} F^{(2)}. \tag{16}$$

The steps of our TSC algorithm are summarized in Table 1.

### 3.4 Time Complexity

The computational cost of the proposed clustering algorithm can be analyzed as follows. As exact SVD of a $m \times n$ matrix has time complexity $O(\min\{mn^2, m^2 n\})$,

**Algorithm 1** Transfer Spectral Clustering

**Input:** $W_N^{(1)}$, $W_N^{(2)}$, $X_N^{(1)}$, $X_N^{(2)}$, $k$, $\lambda$ and step length $t$

Initialize $F^{(1)}$, $F^{(2)}$ and $F^{(3)}$.
**repeat**
Set $F^{(1)} = \pi(F^{(1)} + t\frac{\partial g}{\partial F^{(1)}}/\|\frac{\partial g}{\partial F^{(1)}}\|)$.
Set $F^{(2)} = \pi(F^{(2)} + t\frac{\partial g}{\partial F^{(2)}}/\|\frac{\partial g}{\partial F^{(2)}}\|)$.
Set $F^{(3)} = \pi(F^{(3)} + t\frac{\partial g}{\partial F^{(3)}}/\|\frac{\partial g}{\partial F^{(3)}}\|)$.
**until** Converge

Form matrix $Z^{(1)}$ and $Z^{(2)}$ by normalizing each rows of $F^{(1)}$ and $F^{(2)}$ to have unit length.
Treat each row of $Z^{(1)}$ and $Z^{(2)}$ as samples and run $k$-means to get partitions $P^{(1)}$ and $P^{(2)}$.
**Output:** $P^{(1)}$ and $P^{(2)}$.

the initialization step, which actually involves eigenvalue decomposition and SVD, has time complexity $O(n_1^2k+n_2^2k+\min\{D^2(n_1+n_2), D(n_1+n_2)^2\})$. During iterations, TSC computes the new embeddings and projections. The computations of all three new embeddings involve matrix products and additions and the time complexity is $O(n_1^2k + n_2^2k + D^2k)$. To find projections, computing SVD of matrices $F_1$, $F_2$ and $F_3$ are needed. As such, the time complexity for computing new embeddings and projections is $O(n_1^2k + n_2^2k + D^2k)$. Thus, the time complexity of the iteration part is $O(iter(n_1^2k + n_2^2k + D^2k))$, where $iter$ is the number of iterations. Since $k$ is usually small compared with the number of features and samples, our method is computationally efficient.

## 4 Experimental Results

In this section, an evaluation of the TSC algorithm on text clustering tasks and a comparison with spectral clustering (SC) [2], self-taught clustering (STC) [7] and multitask Bregman clustering (MBC) [22] are reported.

### 4.1 Datasets

We test our algorithms on clustering tasks generated from the 20 Newsgroups (20NG) dataset[1] and Reuters-21578 data set[2]. The 20NG dataset used in our experiments is the bydate version. This version is sorted by date into training(60%) and test(40%) sets. After preprocessing, the 20NG dataset contains 18774 documents and 61188 terms and our datasets have been extracted from it as follows. Each dataset contains two separate parts, with each part corresponding to a clustering task. The categories selected for these datasets are listed in Table 1.

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/
[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

The second column of Table 1 is for task 1 while the third column is for task 2. We can see that the subcategories of tasks 1 and 2 of the first 2 data sets are different, but belong to a more general category. We select 500 samples for each class. For datasets 20NG3-10, the subcategories of the two tasks are the same and we use the whole test set in task 2 and select the same number of samples from the training set in task 1. Hence, samples of these two tasks follow different distribution. For the 20NG11-16 datasets, the samples in the two tasks share the same top categories, but they are selected from different subcategories. Therefore, the distributions of samples in these two tasks are different. For each class, we also select 500 samples. Reuters1-3 were generated in a similar way as 20NG11-16, but with all the samples used in our experiments. All datasets are represented in *tf-idf* format and normalized such that each sample has unit length.

**Table 1.** Datasets Generated from 20 Newsgroups and Reuters-21578 Datasets

| Datasets | Task 1 | Task 2 |
|---|---|---|
| 20NG1 | sci.crypt<br>sci.electronics | sci.med<br>sci.space |
| 20NG2 | talk.politics.guns<br>talk.politics.mideast | talk.politics.misc<br>talk.religion.misc |
| 20NG3 | comp.graphics<br>comp.os.ms-windows.misc | comp.graphics<br>comp.os.ms-windows.misc |
| 20NG4 | rec.autos<br>rec.motorcycles | rec.autos<br>rec.motorcycles |
| 20NG5 | sci.crypt<br>sci.electronics | sci.crypt<br>sci.electronics |
| 20NG6 | talk.politics.guns<br>talk.religion.misc | talk.politics.guns<br>talk.religion.misc |
| 20NG7 | alt.atheism<br>comp.graphics | alt.atheism<br>comp.graphics |
| 20NG8 | misc.forsale<br>rec.sport.hockey | misc.forsale<br>rec.sport.hockey |
| 20NG9 | rec.sport.hockey<br>sci.electronics | rec.sport.hockey<br>sci.electronics |
| 20NG10 | sci.space<br>soc.religion.christian | sci.space<br>soc.religion.christian |
| 20NG11 | comp.*, rec.* | comp.*, rec.* |
| 20NG12 | comp.*, sci.* | comp.*, sci.* |
| 20NG13 | comp.*, talk.* | comp.*, talk.* |
| 20NG14 | rec.*, sci.* | rec.*, sci.* |
| 20NG15 | rec.*, talk.* | rec.*, talk.* |
| 20NG16 | sci.*, talk.* | sci.*, talk.* |
| Reuters1 | orgs.*, places.* | orgs.*, places.* |
| Reuters2 | people.*, places.* | people.*, places.* |
| Reuters3 | orgs.*, people.* | orgs.*, people.* |

### 4.2 Clustering Performance Measures

In order to measure the clustering performance, the normalized mutual information (NMI), cluster purity (purity) and rand index (RI) [24] were adopted. For $\Omega = \{\omega_1, \cdots, \omega_K\}$ denoting a set of clusters and $\mathbb{C} = \{c_1, \cdots, c_K\}$ referring to a set of classes, where $\omega_i$ is interpreted as the samples in cluster $i$, and $c_j$ as the set of samples in class $j$, these three measures are defined as

$$NMI = \frac{\sum_{jl} n_{jl} \log\left(\frac{N n_{jl}}{n_j n_l}\right)}{\sqrt{\left(\sum_j n_j \log \frac{n_j}{N}\right)\left(\left(\sum_l n_l \log \frac{n_l}{N}\right)\right.}}, \tag{17}$$

$$purity = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j|, \tag{18}$$

$$RI = \frac{TP + TN}{TP + FP + FN + TN}, \tag{19}$$

where $n_j$, $n_l$ are the numbers of samples in class $j$ and in cluster $l$, respectively, $n_{jl}$ is the number of samples occurring in both class $j$ and cluster $l$, and $TN+TP$ is the number of correct decisions for a pair of objects, $FP+FN$ is the number of wrong decisions. RI computes the percentage of correct pair-wise relationships. For all these three measures, larger value means better cluster performance. For all the methods in our evaluation, we compute the average performance of the two tasks.

### 4.3 Empirical Analysis

In all the experiments, we construct $k$-nn graphs with $k = 27$. We apply spectral clustering on data from individual task ($\text{SC}_{ind}$) and on data combined from all tasks ($\text{SC}_{com}$). For $\text{SC}_{com}$, we find the embeddings for all tasks and perform $k$-means within tasks. For self-taught clustering (STC), we set $\lambda$ as 1, because the two tasks are equally important in all our experiments. The number of feature clusters is set as 32 as in [7]. The maximum number of iterations for STC is set as 20. MBC and STC are initialized with spectral clustering which usually achieves better performance than $k$-means. The Bregman divergence used in MBC is the Euclidean distance and the only parameter for MBC is set as 0.5 as in [22]. For our method, we set $\lambda = 3$, and the step length as 1. In each experiment, we run the $k$-means algorithm 100 times with random starting points and the most frequent cluster assignment is used. Spectral clustering is not designed for more than one clustering tasks. Hence, we perform spectral clustering on each task separately and report the average performance.

In Table 2 and Table 3, the clustering performance of the five methods on these datasets is reported. We can see that our method achieves the best performance in most datasets. It can be explained by the fact that the feature embeddings computed by our method have taken into considerations of the embeddings of samples in the two clustering tasks and such a sharing could improve the clustering performance in quite a significant manner.

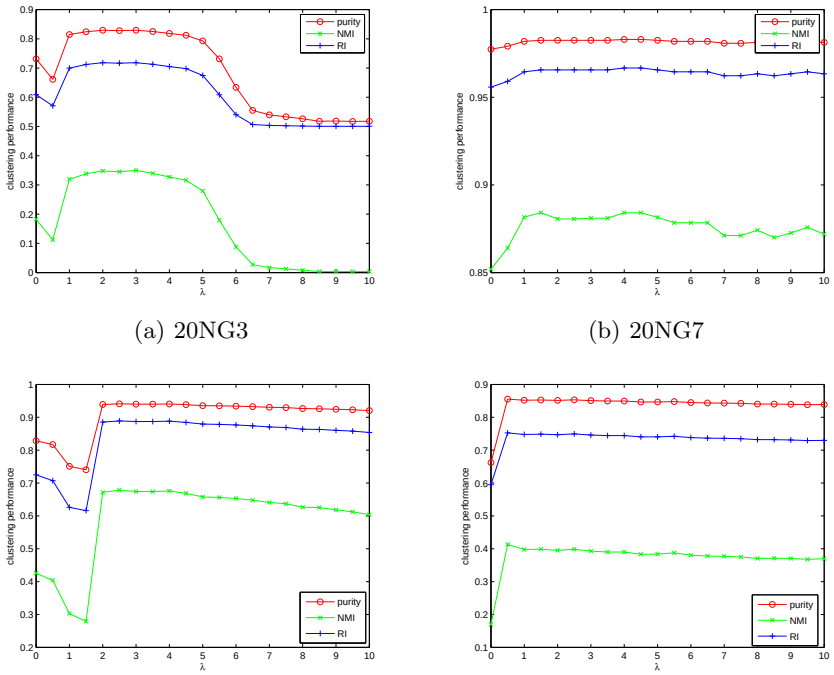
**Table 2.** Clustering performances (row 1: purity, row 2: NMI, row 3: RI) of different algorithms on the first ten datasets in Table 1

| Datasets | $SC_{ind}$ | $SC_{com}$ | STC | MBC | TSC |
|---|---|---|---|---|---|
| 20NG1 | **0.9605** | 0.951 | 0.9295 | 0.941 | 0.955 |
| | **0.7643** | 0.718 | 0.6339 | 0.7201 | 0.7405 |
| | **0.9243** | 0.9068 | 0.8692 | 0.8894 | 0.9141 |
| 20NG2 | 0.9425 | 0.8495 | 0.9035 | **0.9445** | 0.94 |
| | 0.6894 | 0.5175 | 0.5743 | **0.7048** | 0.6883 |
| | 0.8923 | 0.7701 | 0.8311 | **0.8962** | 0.8894 |
| 20NG3 | 0.7314 | 0.8237 | 0.7308 | 0.7603 | **0.8295** |
| | 0.1822 | 0.3324 | 0.1685 | 0.2261 | **0.3497** |
| | 0.6092 | 0.7108 | 0.6067 | 0.6355 | **0.7184** |
| 20NG4 | 0.8643 | 0.8883 | 0.7841 | 0.8883 | **0.904** |
| | 0.4615 | 0.5294 | 0.2788 | 0.5464 | **0.5679** |
| | 0.7671 | 0.8017 | 0.6623 | 0.8025 | **0.8265** |
| 20NG5 | 0.9353 | **0.948** | 0.8934 | 0.9194 | 0.9416 |
| | 0.6725 | **0.7166** | 0.5196 | 0.6307 | 0.6961 |
| | 0.8805 | **0.9024** | 0.8107 | 0.8529 | 0.8919 |
| 20NG6 | 0.9114 | 0.935 | 0.9 | **0.939** | 0.935 |
| | 0.5776 | 0.668 | 0.5255 | **0.679** | 0.6687 |
| | 0.8422 | 0.879 | 0.8208 | **0.8873** | 0.8797 |
| 20NG7 | 0.9745 | **0.9788** | 0.9668 | 0.9724 | 0.9781 |
| | 0.835 | **0.8613** | 0.791 | 0.818 | 0.8567 |
| | 0.9506 | **0.9589** | 0.9359 | 0.9464 | 0.9573 |
| 20NG8 | 0.9571 | 0.9654 | 0.9597 | 0.9558 | **0.9731** |
| | 0.7746 | 0.7951 | 0.766 | 0.7451 | **0.8359** |
| | 0.9185 | 0.9334 | 0.9227 | 0.9159 | **0.9478** |
| 20NG9 | 0.9842 | **0.9855** | 0.9754 | 0.971 | **0.9855** |
| | 0.8849 | 0.8927 | 0.8441 | 0.835 | **0.8964** |
| | 0.969 | **0.9714** | 0.952 | 0.9436 | **0.9714** |
| 20NG10 | 0.9728 | 0.9791 | 0.9715 | 0.9665 | **0.9835** |
| | 0.825 | 0.8549 | 0.8155 | 0.8108 | **0.8815** |
| | 0.9471 | 0.9591 | 0.9447 | 0.9352 | **0.9677** |

**Table 3.** Clustering performances (row 1: purity, row 2: NMI, row 3: RI) of different algorithms on the last nine datasets in Table 1

| Datasets | $SC_{ind}$ | $SC_{com}$ | STC | MBC | TSC |
|---|---|---|---|---|---|
| 20NG11 | 0.9785 | **0.979** | 0.947 | 0.9725 | 0.9745 |
| | 0.8531 | **0.8553** | 0.7131 | 0.8217 | 0.8323 |
| | 0.9581 | **0.9589** | 0.9007 | 0.9466 | 0.9505 |
| 20NG12 | 0.867 | 0.851 | 0.8385 | 0.853 | **0.8875** |
| | 0.522 | 0.4916 | 0.4176 | 0.485 | **0.5304** |
| | 0.7825 | 0.7619 | 0.74 | 0.7605 | **0.806** |
| 20NG13 | 0.9785 | 0.975 | 0.9735 | 0.977 | **0.9845** |
| | 0.8507 | 0.8332 | 0.8243 | 0.8475 | **0.8856** |
| | 0.9579 | 0.9513 | 0.9484 | 0.9551 | **0.9695** |
| 20NG14 | 0.89 | 0.8755 | 0.868 | 0.8815 | **0.922** |
| | 0.5918 | 0.5641 | 0.4835 | 0.5953 | **0.6452** |
| | 0.8183 | 0.7973 | 0.7767 | 0.8082 | **0.8608** |
| 20NG15 | 0.8695 | 0.849 | 0.861 | 0.874 | **0.9605** |
| | 0.6049 | 0.5758 | 0.553 | 0.5625 | **0.7838** |
| | 0.7975 | 0.7764 | 0.784 | 0.7852 | **0.9247** |
| 20NG16 | 0.8285 | 0.8035 | 0.815 | 0.831 | **0.94** |
| | 0.4257 | 0.3876 | 0.4019 | 0.4202 | **0.6742** |
| | 0.7249 | 0.722 | 0.7085 | 0.7203 | **0.8872** |
| Reuters1 | 0.6621 | 0.6779 | 0.6675 | 0.8313 | **0.8506** |
| | 0.1697 | 0.1861 | 0.1612 | 0.349 | **0.3929** |
| | 0.5949 | 0.6098 | 0.594 | 0.7201 | **0.746** |
| Reuters2 | 0.72 | 0.7408 | 0.7387 | 0.7388 | **0.7647** |
| | 0.1775 | 0.182 | 0.1806 | 0.2102 | **0.2196** |
| | 0.6037 | 0.616 | 0.6162 | 0.6177 | **0.6408** |
| Reuters3 | 0.6356 | **0.6903** | 0.6476 | 0.6253 | 0.6527 |
| | 0.0706 | **0.0921** | 0.0734 | 0.0394 | 0.0784 |
| | 0.5429 | **0.5727** | 0.55 | 0.5303 | 0.5531 |

In the proposed TSC algorithm, the parameter $\lambda$ controls the quantity of information to be transferred and a sensitivity analysis of this parameter has been carried out. Figure 2 presents the clustering performance with respect to different $\lambda$ on datasets 20NG3, 20NG7, 20NG16, and Reuters1. We can see that the high performance of our algorithm is stable in a range of $\lambda$. The performance first goes up when $\lambda$ is small, and drops when $\lambda$ is too big. Thus, a proper amount of knowledge should be transfered and we set $\lambda = 3$ in all our experiments.




(a) 20NG3 (b) 20NG7

(c) 20NG16 (d) Reuters1

**Fig. 2.** Clustering performance with different $\lambda$

The convergence property of TSC was also studied. As the objective function of TSC is a continuous function of $F^{(1)}$, $F^{(2)}$ and $F^{(3)}$, its value will increase monotonically as long as the step size $t$ is small enough. Hence, TSC will converge in a fixed number of iterations. Moreover, the required number of iterations to converge depends on the threshold required for checking convenience. Usually, the smaller the threshold, the more iterations are needed. In our experiments, we use 0.00001, and the number of iterations to converge is less than a hundred. Figure 3 shows the learning curve of TSC in dataset Reuters1. We can see that TSC converges very well in this dataset.

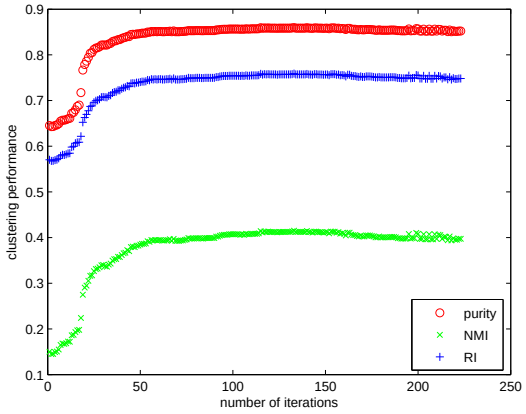



**Fig. 3.** The learning curve of the TSC algorithm in dataset Reuters1

## 5 Conclusions and Future Work

In this paper, an approach to transfer knowledge from other datasets to facilitate more effective spectral clustering is presented. Our assumption is that embeddings of features could link different clustering tasks (from different datasets) and hence improve the clustering performance with respect to each other. Based on this assumption, a co-clustering objective is proposed to design a new spectral clustering algorithm called transfer spectral clustering (TSC). The experimental results show that with the help of co-clustering, TSC has outperformed several state-of-the-art clustering algorithms.

The proposed TSC algorithm assumes that the two datasets have the same number of clusters, which limits its applications in practice. In the future, we plan to extend our method to allow clustering datasets into different groups. We will also explore other more efficient optimization method for TSC. In addition, we will try to find a way to tune the parameter $\lambda$ so that the quantity of knowledge transferred can be made automatic.

## Acknowledgment

We want to thank the anonymous reviewers for their helpful comments and suggestions. This work was supported by the Hong Kong Polytechnic University under Research Studentship.

## References

1. Jain, A.K., Dubes, R.C.: Algorithms for clustering data. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1988)

2. Ng, A.Y., Jordan, M.I., Weiss, Y.: On Spectral Clustering: Analysis and an algorithm. (2001) 849–856
3. Shi, J., Malik, J.: Normalized Cuts and Image Segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8) (August 2000) 888–905
4. Banerjee, A., Merugu, S., Dhillon, I.S., Ghosh, J.: Clustering with Bregman Divergences. J. Mach. Learn. Res. **6** (December 2005) 1705–1749
5. Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S.: Constrained K-means Clustering with Background Knowledge. In: Proceedings of the Eighteenth International Conference on Machine Learning. ICML '01, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (2001) 577–584
6. Finley, T., Joachims, T.: Supervised clustering with support vector machines. In: Proceedings of the 22nd International Conference on Machine Learning. ICML '05, New York, NY, USA, ACM (2005) 217–224
7. Dai, W., Yang, Q., Xue, G.R., Yu, Y.: Self-taught clustering. In: Proceedings of the 25th International Conference on Machine Learning. ICML '08, New York, NY, USA, ACM (2008) 200–207
8. Dhillon, I.S., Mallela, S., Modha, D.S.: Information-theoretic co-clustering. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '03, New York, NY, USA, ACM (2003) 89–98
9. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. KDD '01, New York, NY, USA, ACM (2001) 269–274
10. Pan, S.J., Yang, Q.: A Survey on Transfer Learning. IEEE Transactions on Knowledge and Data Engineering **22** (2010) 1345–1359
11. Dai, W., Jin, O., Xue, G.R., Yang, Q., Yu, Y.: EigenTransfer: a unified framework for transfer learning. In: Proceedings of the 26th Annual International Conference on Machine Learning. ICML '09, New York, NY, USA, ACM (2009) 193–200
12. Mahmud, M.M., Ray, S.: Transfer Learning using Kolmogorov Complexity: Basic Theory and Empirical Evaluations. In Platt, J., Koller, D., Singer, Y., Roweis, S., eds.: Advances in Neural Information Processing Systems 20. MIT Press, Cambridge, MA (2008) 985–992
13. Pan, S.J., Zheng, V.W., Yang, Q., Hu, D.H.: Transfer learning for WiFi-based indoor localization. In: in Proceedings of the Workshop on Transfer Learning for Complex Task of the 23rd AAAI Conference on Artificial Intelligence. (2008)
14. Blitzer, J., Dredze, M., Pereira, F.: Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, Prague, Czech Republic, Association for Computational Linguistics (June 2007) 440–447
15. Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.Y.: Self-taught learning: transfer learning from unlabeled data. In: Proceedings of the 24th International Conference on Machine Learning. ICML '07, New York, NY, USA, ACM (2007) 759–766
16. Daumé, I.H., Marcu, D.: Domain adaptation for statistical classifiers. J. Artif. Int. Res. **26** (May 2006) 101–126
17. Zadrozny, B.: Learning and evaluating classifiers under sample selection bias. In: Proceedings of the Twenty-first International Conference on Machine Learning. ICML '04, New York, NY, USA, ACM (2004) 903–910
18. Shimodaira, H.: Improving predictive inference under covariate shift by weighting the log-likelihood function. Journal of Statistical Planning and Inference **90**(2) (2000) 227–244

19. Caruana, R.: Multitask Learning. Machine Learning **28** (1997) 41–75
20. Argyriou, A., Micchelli, C.A., Pontil, M., Ying, Y.: A Spectral Regularization Framework for Multi-Task Structure Learning. In Platt, J., Koller, D., Singer, Y., Roweis, S., eds.: Advances in Neural Information Processing Systems 20. MIT Press, Cambridge, MA (2008) 25–32
21. Ando, R.K., Zhang, T.: A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data. J. Mach. Learn. Res. **6** (December 2005) 1817–1853
22. Zhang, J., Zhang, C.: Multitask Bregman Clustering. In Fox, M., Poole, D., eds.: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI Press (2010) 655–660
23. Manton, J.: Optimization algorithms exploiting unitary constraints. IEEE Transactions on Signal Processing **50**(3) (March 2002) 635–650
24. Manning, C.D., Raghavan, P., Schtze, H.: Introduction to Information Retrieval. Cambridge University Press, New York, NY, USA (2008)