Structured Clustering with Automatic Kernel Adaptation

Weike Pan and James T. Kwok

Abstract— **Clustering is an invaluable data analysis tool in a variety of applications. However, existing algorithms often assume that the clusters do not have any structural relationship. Hence, they may not work well in situations where such structural relationships are present (e.g., it may be given that the document clusters are residing in a hierarchy). Recently, the development of the kernel-based structured clustering algorithm CLUHSIC [9] tries to alleviate this problem. But since the input kernel matrix is defined purely based on the feature vectors of the input data, it does not take the output clustering structure into account. Consequently, a direct alignment of the input and output kernel matrices may not assure good performance. In this paper, we reduce this mismatch by learning a better input kernel matrix using techniques from semi-supervised kernel learning. We combine manifold information and output structure information with pairwise clustering constraints that are automatically generated during the clustering process. Experiments on a number of data sets show that the proposed method outperforms existing structured clustering algorithms.**

I. Introduction

CLUSTERING is an invaluable data analysis tool in a large variety of real-world applications. The most representative one is the k-means clustering algorithm, which groups similar patterns into the same class by minimizing the intra-class variance and maximizing the inter-class variance simultaneously. However, existing clustering algorithms often assume that the output is structure-less, which may not be the case in many real-world applications. For example, internet document clusters usually reside in a taxonomy, and images can often be organized in a semantically hierarchical or sequential structure. The goal of structured clustering, which is analogous to structured prediction in supervised learning [11], is to utilize structure information as additional side information to improve clustering performance.

CLUHSIC (which stands for "Clustering using HSIC") [9] is a recent clustering algorithm that can easily utilize structure information on the cluster outputs in the clustering process. It maximizes the dependence between cluster labels and data observations according to the Hilbert Schmidt Independence Criterion (HSIC) [4]. However, since the input kernel matrix is defined purely based on the feature vectors of the input data, it does not take the output clustering structure into account. Consequently, a direct alignment of the input and output kernel matrices may not assure good performance.

To address this problem and further improve the clustering performance of CLUHSIC, we will borrow ideas from semi-supervised clustering. There are three main approaches. The first one uses the so-called must-link and cannot-link pairwise constraints [12]. These pairwise constraints define whether the two patterns involved should be grouped into the same class/cluster or not. Some works on metric learning are also based on this observation [13]. The second approach uses information on the data manifold [10], and requires the locally nearby patterns to have similar cluster labels [1]. To obtain even much better performance, the last approach uses both pairwise constraints and local structure [5], [8], and performs constraint-driven metric learning and manifold-driven constraint propagation. However, all these approaches are designed for outputs with no structure relationships, and thus cannot be applied to our structured clustering scenario.

In this paper, we combine pairwise constraints and manifold information with the output structure information to learn a better input kernel matrix. This can then be input to some structured clustering algorithm such as CLUHSIC. However, the pairwise constraints are partial label information, and may not be feasible or available in some clustering applications. To alleviate this problem, we propose a method that, instead of asking for these constraints from the user, can automatically generate artificially labeled pairwise constraints based on the current clustering solution.

The rest of the paper is organized as follows. Section II gives brief reviews on some related works. Section III describes the proposed structured clustering algorithm. Experimental results are presented in Section IV, and the last section gives some concluding remarks.

II. Related Works

A. CLUHSIC (Clustering using HSIC)

Given a set of samples $\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$, the linear dependence between $\mathbf{x}_i$'s and $\mathbf{y}_i$'s can be easily estimated by simple statistics such as linear correlation. However, nonlinear dependencies are more difficult to measure. A recently proposed dependence (or, more precisely, independence) measure is the Hilbert Schmidt Independence Criterion (HSIC) [4], which is based on the Hilbert-Schmidt norm of a cross-covariance operator from the reproducing kernel Hilbert space (RKHS) of the input to the RKHS of the output [3]. An empirical estimate of HSIC is

$$(n-1)^{-2}\mathrm{tr}(\mathbf{K}^x\mathbf{K}^y), \tag{1}$$

where $\mathrm{tr}(\cdot)$ denotes the matrix trace, $\mathbf{K}^x, \mathbf{K}^y \in \mathbb{R}^{n\times n}$ are kernel matrices defined on $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and $\{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$, respectively. As in [9], we will always assume that $\mathbf{K}^x$ is centered. Recent studies show that HSIC has several advantages over other independence measures. First, its empirical estimate in (1) is easy to compute. Moreover, it has good

Weike Pan and James T. Kwok are with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong (email: {weikep, jamesk}@cse.ust.hk).

This work was supported by the Research Grants Council of the Hong Kong Special Administrative Region under grant 615209.

978-1-4244-9636-5/11/$26.00 ©2011 IEEE

uniform convergence guarantees and very little bias even in high dimensions.

CLUHSIC [9] is a clustering algorithm which is based on maximizing the dependence between the input and the output as measured by the HSIC. On a set of c clusters, we first define an output kernel matrix $\mathbf{A} \in \mathbb{R}^{c\times c}$. Let $\mathbf{\Pi} \in \{0,1\}^{n\times c}$ be an assignment matrix such that its ith row specifies the assignment of the ith pattern to one of the c clusters, i.e., $\mathbf{\Pi}_{ij} \in \{0,1\}$ and $\mathbf{\Pi 1} = \mathbf{1}$, where $\mathbf{1}$ is the vector of ones. The kernel matrix defined on the $\mathbf{y}_i$'s can then be written as $\mathbf{K}^y = \mathbf{\Pi A \Pi}'$. CLUHSIC aims at finding the cluster assignment such that the resultant $\mathbf{K}^y$ is maximally dependent on the kernel matrix $\mathbf{K}^x$ defined on the $\mathbf{x}_i$'s. Using the definition of HSIC in (1), this is then formulated as the following optimization problem:

$$\begin{aligned} \max_{\mathbf{\Pi}} \quad & \mathrm{tr}(\mathbf{K}^x \mathbf{\Pi A \Pi}') \\ \text{s.t.} \quad & \mathbf{\Pi 1} = \mathbf{1}, \\ & \mathbf{\Pi}_{ij} \in \{0,1\}, \;\; i = 1,\ldots,n, j = 1,\ldots,c. \end{aligned}$$

B. Maximum Margin Clustering for Structured Outputs

A related discriminative clustering algorithm for structured outputs is recently proposed in [15]. It is based on the idea of maximum margin clustering [14] that trains a support vector machine (SVM) by maximizing the margin and minimizing the loss over all possible cluster labellings. However, maximum margin clustering is computationally much harder than maximum margin classification. Existing methods typically rely on reformulating and relaxing the non-convex optimization problem as a large semidefinite programs (SDP) that are computationally expensive. To combat this problem, Xu *et al.* [15] proposed a heuristic procedure that avoids SDP entirely. However, as will be shown in Section IV, its empirical performance is not quite satisfactory.

C. Pairwise Constraint Propagation (PCP)

Pairwise constraint propagation (PCP) [8] is a kernel learning algorithm that considers both the pairwise must-link / cannot-link constraints and the manifold's local structure. A must-link between two patterns i and j specifies that i and j are very similar, while a cannot-link between i and j specifies that i and j are very dissimilar. Assuming that the target kernel has values in the range of 0 and 1. This can thus be represented by the constraints that

$$\mathbf{K}_{ij} = \begin{cases} 1 & \text{if } i,j \text{ are must-linked,} \\ 0 & \text{if } i,j \text{ are cannot-linked.} \end{cases}$$

Let $\mathbf{W}$ be the affinity matrix of a graph representing an underlying data manifold. PCP learns the target kernel matrix $\mathbf{K}$ that maximally conforms to the manifold structure via its (normalized) graph Laplacian $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ (where $\mathbf{D}$, with $\mathbf{D}_{ii} = \sum_{j=1}^{n} \mathbf{W}_{ij}$, is the diagonal degree matrix), and is also consistent with the provided must-link / cannot-link constraints. Let $\mathcal{M}$ be the set of must-link constraints and $\mathcal{C}$ the set of cannot-link constraints. This then leads to the following semidefinite programming problem:

$$\begin{aligned} \max_{\mathbf{K}} \quad & \mathrm{tr}(\mathbf{LK}) \\ \text{s.t.} \quad & \mathbf{K}_{ii} = 1, \; i = 1,\ldots,n, \\ & \mathbf{K}_{ij} = 1, (i,j) \in \mathcal{M}, \\ & \mathbf{K}_{ij} = 0, (i,j) \in \mathcal{C}, \\ & \mathbf{K} \succeq 0, \end{aligned} \tag{2}$$

where $\mathbf{K} \succeq 0$ denotes that $\mathbf{K}$ is positive semi-definite.

III. The Proposed Method

In this section, we first discuss the deficiencies of using CLUHSIC and PCP in Sections III-A and III-B, respectively. To address these deficiencies, an extension of the PCP that can utilize structure information is then proposed in Section III-C. However, this requires the presence of labeled patterns during the clustering process. To cater for situations where this may not be feasible, we propose a further extension in Section III-D that can automatically generate these labeled patterns.

A. Deficiency with CLUHSIC

Recall that CLUHSIC tries to maximally align the input kernel matrix with the output kernel matrix via the HSIC criterion. However, the input kernel matrix is defined purely based on the feature vectors of the input data, and does not take the output clustering structure into account. Consequently, a direct application of kernel target alignment [2] or dependence maximization between the input and output kernel matrices [9] may not assure good performance.

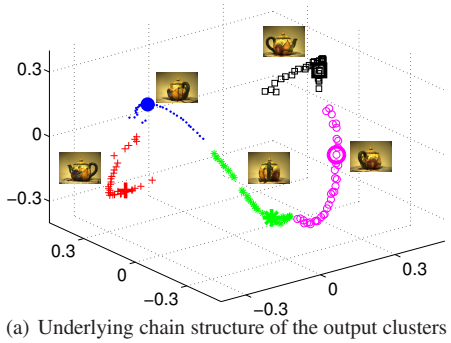
As an example, Figure 1(a) shows a set of images [9] for a teapot rotated from 1° – 160°. The images are grouped into five clusters arranged in the form of a chain (which reflects the angle of rotation). As suggested in [9], an appropriate 5×5 kernel matrix defined on these five clusters, which reflects such a chain structure, is

$$\begin{bmatrix} 2 & 1 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 1 & 2 \end{bmatrix}. \tag{3}$$

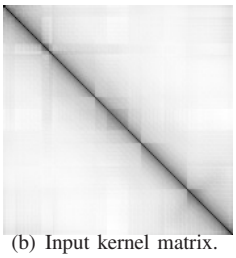
In words, each cluster is similar to itself and, to a less extent, its two neighboring clusters; but not similar to the other clusters. Figure 1(b) shows the corresponding input kernel matrix[1], which is defined using a Gaussian kernel based on the pixel values of the image. As can be seen, it does not show a clear structure as in (3). More precisely, the matrix is essentially block-diagonal, meaning that patterns in the same cluster are similar to each other. However, pattern pairs that are in neighboring clusters do not show a higher similarity than those in non-neighboring clusters.

Hence, in general, there can be a significant mismatch between the input kernel matrix and output kernel matrix.

[1]For clearer visualization, the entries in the kernel matrix have been arranged in order of the rotation angle of the corresponding image.



(a) Underlying chain structure of the output clusters (different clusters are in different colors).



(b) Input kernel matrix.

Fig. 1. Illustrations on the teapot data showing a mismatch in the input and output kernel matrices.

Consequently, the resultant cluster assignment matrix obtained CLUHSIC may not be accurate. A better way to define the input kernel matrix is thus needed.

B. Deficiency with PCP

Since PCP is a kernel learning algorithm, it thus offers the hope of being able to produce a better input kernel matrix for use in CLUHSIC. However, recall that in PCP, when two patterns i and j are connected by a must-link, the corresponding kernel entry $\mathbf{K}_{ij}$ is always set to one; whereas when i and j are connected by a cannot-link, the corresponding $\mathbf{K}_{ij}$ is always zero. Hence, the value of $\mathbf{K}_{ij}$ does not depend on the similarity of the underlying clusters that i and j belong to. In other words, PCP does not utilize the structure information among output clusters.

C. PCP using Structure Information

Suppose that we have a small amount of label information in the clustering process. In other words, we are given a set of patterns $\{\mathbf{x}_i\}$ and the corresponding cluster labels $\{c(i)\}$. Using the output kernel matrix $\mathbf{A}$, we define $\mathbf{K}^*_{ij}$ as

$$\mathbf{K}^*_{ij} = \mathbf{A}_{c(i)c(j)}. \tag{4}$$

In the special case where there is no structure among the clusters, $\mathbf{A}$ becomes the identity matrix $\mathbf{I}$, and (4) reduces to the binary setting

$$\mathbf{K}^*_{ij} = \begin{cases} 1 & i,j \text{ belong to the same cluster,} \\ 0 & \text{otherwise,} \end{cases}$$

as in PCP.

To see how (4) works, consider an example of facial expression data shown in Figure 2. It consists of three types of facial expression images from three subjects. The patterns are hierarchically grouped into nine clusters, first by subject and then by expression. According to [9], an appropriate output kernel matrix defined on such a hierarchy is

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \tag{5}$$

where $\otimes$ is the Kronecker product. In words, for two clusters a and b,

$$\mathbf{A}_{ab} = \begin{cases} 2 & a = b, \\ 1 & a \text{ and } b \text{ are for the same subject,} \\ 0 & \text{otherwise.} \end{cases}$$

Hence, using (4), for two images i and j,

$$\mathbf{K}^*_{ij} = \begin{cases} 2 & i,j \text{ are from the same subject and expression,} \\ 1 & i,j \text{ are from same subject but diff expressions,} \\ 0 & \text{otherwise.} \end{cases}$$

This is more meaningful and flexible than the simple 0/1 setting in PCP.

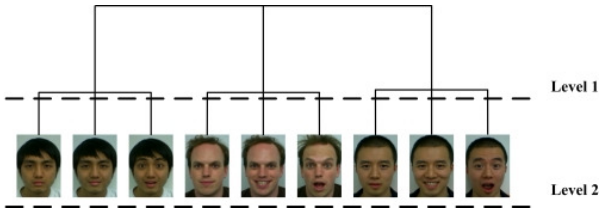


Fig. 2. Hierarchical structure of the facial expression data.

With this definition of $\mathbf{K}^*_{ij}$ (scaled to [0,1]), we can extend the kernel learning formulation in PCP as:

$$\begin{aligned} \max_{\mathbf{K}} \quad & \text{tr}(\mathbf{LK}) \\ \text{s.t.} \quad & \mathbf{K}_{ii} = 1,\ i = 1, \ldots, n, \\ & \mathbf{K}_{ij} = \mathbf{K}^*_{ij},\ \ i,j \in \mathcal{P}, \\ & \mathbf{K} \succeq 0, \end{aligned} \tag{6}$$

where $\mathcal{P}$ is the set of labeled patterns. Note that this is still a standard SDP, and thus can be readily solved by off-the-shelf SDP solvers. The learned kernel matrix $\mathbf{K}$ can then be used as the input kernel matrix in the CLUHSIC algorithm for structured clustering. The procedure, called PCPSI (PCP using Structure Information), is shown in Algorithm 1.

Algorithm 1 PCPSI (PCP using Structure Information).

Input: Data set $\mathcal{S}$, output kernel matrix $\mathbf{A}$ defined on clusters, a set of labeled patterns $\mathcal{P}$.

Output: Assignment matrix $\mathbf{\Pi}$.

1: Construct $\mathbf{K}^*_{ij}$ for each $i,j \in \mathcal{P}$ using (4).
2: Construct the affinity matrix $\mathbf{W}$.
3: Compute the graph Laplacian $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$.
4: Obtain kernel matrix $\mathbf{K}$ from the SDP in (6).
5: Obtain $\mathbf{\Pi}$ from CLUHSIC, by using the obtained $\mathbf{K}$ as input kernel matrix and $\mathbf{A}$ as output kernel matrix.

D. Automatic Generation of Constraints

A major deficiency of PCPSI is that it requires the presence of some labeled patterns, which may not be feasible in some clustering applications. To alleviate this problem, we will generate artificially labeled patterns based on the most confidently labeled patterns in the current clustering solution.

Recall that $\mathbf{\Pi}$ is the cluster assignment matrix, and $\mathbf{\Pi}_{ia}$ denotes the "chance" of assigning pattern i to cluster a. For each cluster a, we pick the pattern i whose $\mathbf{\Pi}_{ia}$ value is the largest[2] among $\{\mathbf{\Pi}_{1a}, \ldots, \mathbf{\Pi}_{na}\}$. This is thus the pattern whose cluster membership is the most certain from cluster a's perspective. At the end, a total of c most confident patterns are obtained. We use these patterns to create $\mathbf{K}^*_{ij}$ values for input to PCPSI. For any two of these patterns (e.g., i and j), the kernel evaluation $\mathbf{K}^*_{ij}$ is computed as

$$\mathbf{K}^*_{ij} = \sum_{a=1}^{c}\sum_{b=1}^{c} \mathbf{\Pi}_{ia}\mathbf{\Pi}_{jb}\mathbf{A}_{ab} = \mathbf{\Pi}_{i:}\mathbf{A}\mathbf{\Pi}_{j:}', \tag{7}$$

where $\mathbf{\Pi}_{i:}$ is the ith row of $\mathbf{\Pi}$.

The whole procedure, called SCSC (Self-Constrained Structured Clustering) is shown in Algorithm 2. Initially, we start with no labeled data, and obtain the assignment matrix $\mathbf{\Pi}$ from CLUHSIC (which is purely unsupervised and does not require labeled data). Since this assignment matrix may not be very accurate, we use a permutation matrix $\boldsymbol{\pi}$ to permute the output clusters so as to achieve a better alignment with the input kernel matrix. Specifically, let the original input kernel matrix be $\mathbf{K}^0$. We obtain $\boldsymbol{\pi}$ by solving the following problem:

$$\max_{\boldsymbol{\pi}} \ \mathrm{tr}\left[\boldsymbol{\pi}\left(\mathbf{\Pi}'\mathbf{K}^0\mathbf{\Pi}\right)\boldsymbol{\pi}'\mathbf{A}\right], \tag{8}$$

where $\boldsymbol{\pi}$, as a permutation matrix, satisfies $\boldsymbol{\pi} \in \{0,1\}^{c\times c}, \boldsymbol{\pi}\mathbf{1} = \mathbf{1}$, and $\boldsymbol{\pi}'\mathbf{1} = \mathbf{1}$. This can be solved by iteratively using a standard linear assignment problem (LAP) solver [6]. Then, we update $\mathbf{\Pi}$ as $\mathbf{\Pi} \leftarrow \mathbf{\Pi}\boldsymbol{\pi}'$. New $\mathbf{K}^*_{ij}$ entries are generated as described above, and the process reiterates by using these new constraints as input to PCPSI.

Algorithm 2 SCSC (Self-Constrained Structured Clustering).

Input: Data set $\mathcal{S}$, output kernel matrix $\mathbf{A}$.
Output: Assignment matrix $\mathbf{\Pi}$.

1: **repeat**
2: Learn $\mathbf{\Pi}$ using PCPSI (use CLUHSIC instead in the first iteration).
3: Learn $\boldsymbol{\pi}$ by solving (8).
4: Update $\mathbf{\Pi} \leftarrow \mathbf{\Pi}\boldsymbol{\pi}'$.
5: Generate new $\mathbf{K}^*_{ij}$'s from (7), and add them to $\mathcal{P}$.
6: **until** convergence.

The algorithm converges when either the newly generated $\mathbf{K}^*_{ij}$ entries are the same as the ones in the previous iteration, or the maximum number of iterations is reached. In the experiments, the algorithm often converges quickly (in fewer than 10 iterations).

[2]In general, more than one patterns may be drawn in each iteration. For simplicity, we just use one here.

IV. Experiments

In this section, we perform experiments on a number of commonly used benchmark data sets with structured outputs. These include the teapot (Section IV-B), facial expression (Section IV-C) and newsgroups (Section IV-D).

A. Experimental Setup

We compare the proposed SCSC algorithm with the following clustering algorithms: (1) CLUHSIC [9]; (2) the discriminative unsupervised training algorithm (denoted XWSS) in [15]; and (3) k-means clustering. In SCSC, one pattern for each cluster is used to construct the new constraints in each iteration, and the maximum number of iterations is set to to 10. For XWSS, its SDP formulation is computationally expensive and can only be used on very small data sets. Hence, in all the experiments on XWSS, the heuristic iterative procedure proposed in [15] is adopted. As for k-means, the traditional version does not utilize structure information. Hence, we implement a variant that recursively subdivides the data according to its hierarchical structure. Take the face data (Figure 2) as an example. The root has 3 children, so we first apply k-means to cluster the whole data set into 3 clusters. As each child also has 3 grand-children, so we further apply k-means to divide each cluster into 3 subclusters.

The optimization of CLUHSIC involves integer programming. However, since the focus of this paper is not on how to solve this integer program, we will simply follow the iterative greedy local optimization procedure in [9] and obtain an approximate solution. The procedure often converges in fewer than 20 iterations. More sophisticated approaches, such as nonnegative matrix factorization [7] and low-rank SDP [16], can also be used. The k-means clustering result is used to initialize the other methods. To reduce statistical variability, all the experimental results reported are averaged over 10 repetitions.

The following performance measures are used:

1) Clustering accuracy (i.e., the percentage of patterns that are correctly clustered);
2) Tree loss (for tree structures), which is defined as the height of the first common ancestor of the true and predicted cluster labels in the hierarchy [11]. It measures how well the hierarchy of the outputs are observed and has been commonly used in structured output prediction. Note that the tree loss is the same as accuracy when measured at the first level of the hierarchy. Hence, tree loss values will not be reported for the first level;
3) Chain loss (for chain structures), which is the distance from the ground truth position, and has been commonly used for sequential outputs.

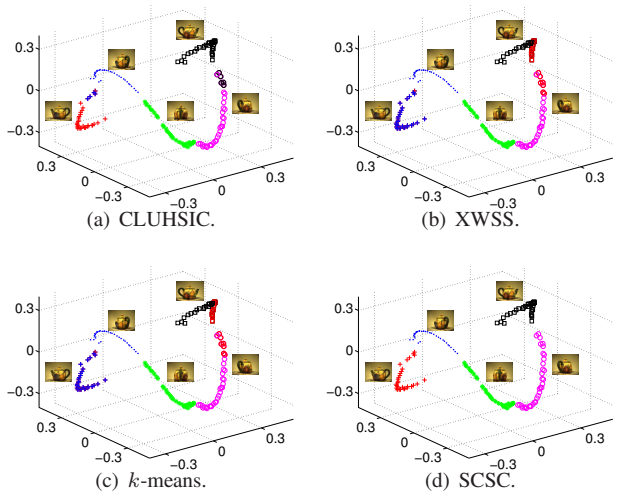
B. Results on the Teapot Data

The teapot data[3] has been used in [9]. Here, we use a subset of 175 images (size 76×101) rotated from 1° –

[3]http://www.it.usyd.edu.au/~lesong/cluhsic_datasets.html

160° to form a chain. They are grouped into 5 clusters, each having 35 images (Figure 1(a)). The input kernel matrix is constructed from the Gaussian kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\sigma\|\mathbf{x}_i - \mathbf{x}_j\|^2)$, with $\sigma = 1/d$ where d is the input dimensionality of the data; while (3) is used as the output kernel matrix.

1) Structured Clustering: We first illustrate CLUHSIC and SCSC's unique structure-preserving property. For visualization, the data points are projected into the space spanned by its top 3 eigenvectors. Results for a typical run are shown in Figure 3, and the obtained clusters are shown in different colors. As can be seen, SCSC can well separate the patterns, which is followed by CLUHSIC, XWSS and k-means. However, only CLUHSIC and SCSC preserve the structure of the clusters. Specifically, in the ground truth solution (Figure 1(a)), the five clusters are arranged in the order of red, blue, green, magenta and black. This ordering is only preserved in the results of CLUHSIC and SCSC.



(a) CLUHSIC. (b) XWSS. (c) k-means. (d) SCSC.

Fig. 3. Clustering results on the teapot data in a typical run.

Figure 4 shows the kernel matrix obtained by SCSC. Clearly, this resembles the output kernel matrix in (3), and shows the underlying chain structure of the clusters.

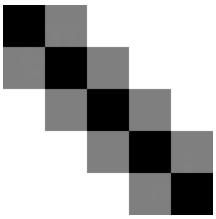


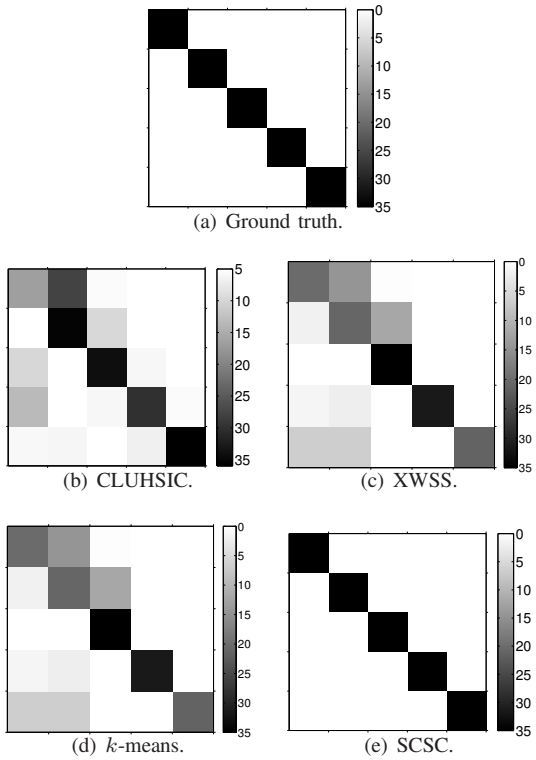
Fig. 4. Kernel matrix obtained by SCSC on the face data.

2) Clustering Performance: The clustering performance is shown in Table I. As can be seen, SCSC outperforms the other methods on both performance metrics and the improvements are statistically significant. To aid visualization, we also show the confusion matrices (averaged over the 10 repetitions) in Figure 5. k-means performs poorly as it does not consider the cluster structure as a whole but only in a layer-by-layer manner. CLUHSIC considers the output structure as a whole, while SCSC can further benefit from the self-constructed constraints.

TABLE I

CLUSTERING PERFORMANCE ON THE TEAPOT DATA.

	accuracy (%)	chain loss
CLUHSIC	92.9 ± 0.55	0.071 ± 0.055
XWSS	91.7 ± 1.39	0.093 ± 0.161
k-means	91.7 ± 1.39	0.093 ± 0.161
SCSC	**100 ± 0.00**	**0.000 ± 0.000**



(a) Ground truth. (b) CLUHSIC. (c) XWSS. (d) k-means. (e) SCSC.

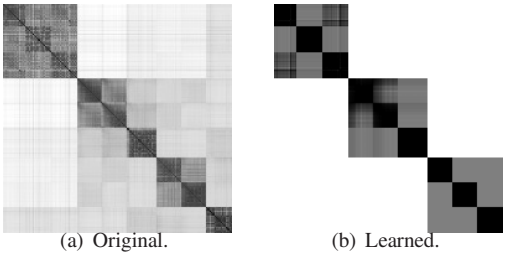
Fig. 5. Confusion matrices on teapot (averaged over 10 repetitions).

C. Results on the Facial Expression Data

The facial expression data[4] has been used in [9]. It consists of 185 images (size 217×308) of three types of facial expressions from three subjects. The facial expressions of the same person are first grouped together in the hierarchy (Figure 2). As in Section IV-B, the Gaussian kernel is used on the input; while (5) is used as the output kernel matrix.

Figure 6 compares the original input kernel matrix with the one learned by SCSC. As can be seen, the learned matrix clearly exhibits the hierarchical structure consistent with the output kernel matrix in (3). In terms of the clustering performance, SCSC again outperforms the others on all metrics (Table II), and the improvements are all statistically significant.

[4] http://www.it.usyd.edu.au/~lesong/cluhsic_datasets.html



(a) Original. (b) Learned.

Fig. 6. Input kernel matrices on the facial expression data.

TABLE II

CLUSTERING PERFORMANCE ON THE FACIAL EXPRESSION DATA.

	accuracy (%)		tree loss
	level 1	level 2	level 2
CLUHSIC	89.0 ± 15.5	80.8 ± 15.0	0.302 ± 0.289
XWSS	70.1 ± 16.0	55.0 ± 16.3	0.749 ± 0.311
k-means	70.1 ± 4.40	54.6 ± 4.20	0.753 ± 0.084
SCSC	**100 ± 0.00**	**100 ± 0.00**	**0.000 ± 0.000**

D. Results on the Newsgroups Data

We use a subset of four discussion groups (comp.os.ms-windows.misc, comp.sys.mac.hardware, talk.politics.guns and talk.politics.mideast) from the newsgroups data[5], and construct the taxonomy shown in Figure 7. For each newsgroup, we randomly select 50 documents, and thus a total of 200 documents are used in the experiment. Each document is represented as a ℓ_2-normalized TFIDF vector. The linear kernel is used to form the input kernel matrix. Similar to the facial expression data, the output kernel matrix for this hierarchical structure is $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$.

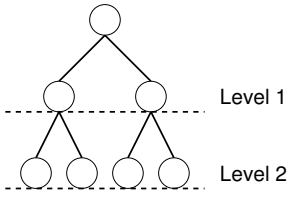


Fig. 7. Hierarchical structure of the newsgroups data. From left to right, the level 2 nodes are the newsgroups comp.os.ms-windows.misc, comp.sys.mac.hardware, talk.politics.guns and talk.politics.mideast.

Results are shown in Table III. As can be seen, the performance of SCSC (in both accuracy and tree loss) is again significantly better than the other methods.

TABLE III

CLUSTERING PERFORMANCE ON THE NEWSGROUPS DATA.

	accuracy (%)	tree loss
CLUHSIC	43.2 ± 10.1	0.857 ± 0.155
XWSS	42.3 ± 4.28	0.795 ± 0.140
k-means	42.0 ± 4.27	0.798 ± 0.139
SCSC	**60.4 ± 15.4**	**0.573 ± 0.241**

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

V. CONCLUSION

In this paper, we proposed a novel method that performs clustering of structured outputs with the manifold's local structure, the output structure information and pairwise constraints that are automatically constructed during the process. By reducing the mismatch between the input kernel matrix and output kernel matrix, the proposed algorithm achieves significantly better performance than existing structured clustering algorithms.

REFERENCES

[1] M. Belkin and P. Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, 56(1-3):209–239, 2004.

[2] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola. On kernel-target alignment. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[3] K. Fukumizu, F.R. Bach, and M.I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

[4] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In *Proceedings of the International Conference on Algorithmic Learning Theory*, pages 63–77, Singapore, October 2005.

[5] S. Hoi, R. Jin, and M. Lyu. Learning non-parametric kernel matrices from pairwise constraints. In *Proceedings of the International Conference on Machine Learning*, pages 361–368, Corvalis, Oregon, USA, June 2007.

[6] R. Jonker and A. Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38:325–340, 1987.

[7] D.D. Lee and H. Seung. Algorithms for non-negative matrix factorization. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, Cambridge, MA, 2001. MIT Press.

[8] Z. Li, J. Liu, and X. Tang. Pairwise constraint propagation by semidefinite programming for semi-supervised classification. In *Proceedings of the International Conference on Machine Learning*, pages 576–583, Helsinki, Finland, 2008.

[9] L. Song, A. Smola, A. Gretton, and K.M. Borgwardt. A dependence maximization view of clustering. In *Proceedings of the International Conference on Machine Learning*, pages 815–822, Corvallis, Oregon, USA, June 2007.

[10] R. Souvenir and R. Pless. Manifold clustering. In *Proceedings of the Tenth IEEE International Conference on Computer Vision*, volume 1, pages 648–653, Beijing, China, October 2005.

[11] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, December 2005.

[12] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 577–584, Williamstown, MA, USA, 2001.

[13] E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

[14] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In *Advances in Neural Information Processing Systems 17*, Cambridge, MA, 2005. MIT Press.

[15] L. Xu, D. Wilkinson, F. Southey, and D. Schuurmans. Discriminative unsupervised learning of structured predictors. In *Proceedings of the International Conference on Machine Learning*, pages 1057–1064, Pittsburgh, PA, USA, June 2006.

[16] W. Yang, J.T. Kwok, and B. Lu. Spectral and semidefinite relaxations of the CLUHSIC algorithm. In *Proceedings of the SIAM International Conference on Data Mining*, pages 106–117, Columbus, Ohio, USA, April 2010.