Transfer Prototype-based Fuzzy Clustering

Zhaohong Deng\textsuperscript{1,4}, Yizhang Jiang\textsuperscript{1}, Fu-Lai Chung\textsuperscript{2}, Kup-Sze Choi\textsuperscript{1}, Shitong Wang\textsuperscript{1,2}

\textsuperscript{1} School of Digital Media, Jiangnan University, Wuxi, Jiangsu, P.R. China
\textsuperscript{2} Department of Computing, Hong Kong Polytechnic University, Hong Kong
\textsuperscript{3} The Centre for Integrative Digital Health, Hong Kong Polytechnic University, Hong Kong
\textsuperscript{4} Department of Biomedical Engineering, University of California, Davis, CA, USA

Abstract: The traditional prototype based clustering methods, such as the well-known fuzzy $c$-mean (FCM) algorithm, usually need sufficient data to find a good clustering partition. If the available data is limited or scarce, most of the existing prototype based clustering algorithms will no longer be effective. While the data for the current clustering task may be scarce, there is usually some useful knowledge available in the related scenes/domains. In this study, the concept of transfer learning is applied to prototype based fuzzy clustering (PFC). Specifically, the idea of leveraging knowledge from the source domain is exploited to develop a set of transfer prototype based fuzzy clustering (TPFC) algorithms. Three prototype based fuzzy clustering algorithms, namely, FCM, fuzzy k-plane clustering (FKPC) and fuzzy subspace clustering (FSC), have been chosen to incorporate with knowledge leveraging mechanism to develop the corresponding transfer clustering algorithms. Novel objective functions are proposed to integrate the knowledge of source domain with the data of target domain for clustering in the target domain. The proposed algorithms have been validated on different synthetic and real-world datasets and the results demonstrate their effectiveness when compared with both the original prototype based fuzzy clustering algorithms and the related clustering algorithms like multi-task clustering and co-clustering.

Keywords: Prototype based clustering, Transfer learning, Knowledge leverage, Fuzzy $c$-means, Fuzzy k-plane clustering, Fuzzy subspace clustering
I. Introduction

Transfer learning [1] is receiving more and more attentions in the fields of machine learning and data mining. They work on the assumption that the data of the current scene is scarce for the current learning tasks, but some useful information of a reference scene is available. The current scene and the reference scene are commonly named as the target domain and the source domain respectively. Currently, transfer learning has been demonstrated as a promising approach to obtain an effective model for the target domain by effectively leveraging useful information from the source domain in the learning process. Its main characteristics can be summarized as follows. First, the data of target domain is not sufficient to train up a good model for the target domain. Second, the source domain is assumed to be related to the target domain, in a sense that they are different but similar to some extent, making the trained model on the source domain not directly adoptable by the target domain. Third, the modeling effect for the target domain is the main concern while that of the source domain is usually not a focus.

In the past decade or so, transfer learning has been studied extensively for different applications, such as text classification [2] and indoor WiFi location estimation [3]. The existing works can be generally divided into four categories: 1) transfer learning for classification [2-8]; 2) transfer learning for feature extraction [9, 10]; 3) transfer learning for regression [11-15] and 4) transfer learning for clustering [16, 17]. While the first three tasks have been studied quite extensively, works on transfer clustering are still limited despite of the wide range of real-world clustering applications. Clustering has long been a major research topic and many clustering algorithms have been proposed. With respect to the concern here, they can be classified as: 1) prototype based clustering algorithms [18-23]; 2) density based clustering algorithms [24-28]; 3) graph based clustering algorithms [29-32]; and 4) clustering algorithms based on other data modeling mechanisms [33-35]. Among these four types of algorithms, prototype based clustering such as *k*-means is perhaps the most popular one and in fact has been extensively studied. In this paper, we focus on transfer learning for this type of clustering algorithms and propose several representative transfer prototype based clustering algorithms in the context of fuzzy clustering.
Let us take a look at an example in Fig. 1 to illustrate a situation where transfer learning could be exploited. As shown and described in Fig. 1 (left part), the data in the target domain is difficult to obtain ideal clustering result with the traditional prototype-based clustering algorithms. However, if the information of the reference scene, i.e., the source domain, in Fig. 1 (right part) is considered, more promising clustering result can be expected. For example, if we know that the two domains are related with a prior, we can expect that the number of clusters in these two domains is the same and the induced knowledge in the source domain, such as the cluster centers, can be employed to guide the clustering process in the target domain. It can be seen from this example that how to realize effective knowledge/information transfer from the source domain to the target domain is the key to develop transfer clustering algorithms. It seems that the simplest way to obtain information from the source domain is to use the original data of the source domain directly in the target domain. However, borrowing data from the source domain may suffer from the following problems. First, important information such as the distribution difference between the source and the target domains will be lost if we simply merge the data from the two domains. Second, the data of the source domain may not be appropriate to be directly used in the target domain as there exist similarities as well as differences between the data distributions of the two domains and direct use of the source domain’s data may dominate the data distribution in the target domain. This is particularly true when the size of the dataset in the source domain is much larger. Third, due to the necessity of privacy protection in some applications, such as customers’ personal information, the raw data of source domain may not be directly available.
All these problems should be properly addressed in order to develop an effective transfer learning strategy for the prototype based clustering methods.

In this paper, transfer learning for prototype based clustering methods is exploited. In view of their representativeness, three PFC algorithms, i.e., fuzzy c-mean (FCM) with cluster center prototypes, fuzzy k-plane clustering (FKPC) with cluster direction prototypes and fuzzy subspace clustering (FSC) with cluster subspace prototypes, have been chosen to incorporate with knowledge leveraging mechanism to develop the corresponding TPFC algorithms. The knowledge induced in the source domain, such as the cluster centers and/or the subspace of the clustering results, has been appropriately used to boost the clustering performance in the target domain. A novel objective criterion is proposed to integrate the knowledge of the source domain with the data of the target domain for learning the cluster prototypes in the target domain. Thus, transferring knowledge of the source domain can effectively make up the deficiency in learning from the limited data in the target domain.

As mentioned, works in transfer clustering are scarce. In [16], the self-taught clustering (STC) has been proposed as a very first transfer clustering algorithm based on mutual information. In [17], a transfer learning version of spectral clustering is proposed. These two transfer clustering algorithms are not developed for prototype based clustering. There exist some obvious differences between these related works and the proposed TPFC algorithms. For the self-taught clustering, it is based on mutual information (MI) and therefore enough data in both of source and target domains are assumed available for estimating the probability densities properly and compute the MI accordingly [16]. Moreover, it assumes that the raw data of source domain are available and this may not always hold in some applications. For the proposed algorithms, raw data of the source domain is not required and data in the target domain can be assumed scarce. For the transfer spectral clustering algorithm [17], it is specifically designed for the spectral clustering while our work focuses on the prototype based fuzzy clustering methods.

Transfer clustering is much related to co-clustering [36], collaborative clustering [37, 38] and multi-task clustering [39, 40], where multiple clustering tasks are usually handled simultaneously and they may cooperate with each other in order to improve the clustering performance of all clustering tasks involved. By comparing co-clustering, collaborative and
multi-task clustering with the proposed transfer clustering algorithms, we can see that these three types of related clustering techniques assume that the multiple clustering tasks involved are equally important and each can benefit from the others in the clustering process. However, the proposed transfer clustering algorithms only focus on the clustering task in target domain and the source domain is only adopted to enhance the clustering performance in the target domain.

Another type of related works is semi-supervised clustering which can be taken as a type of knowledge based clustering that makes use of some useful knowledge in the clustering procedure [41-45]. For example, the most commonly used information is “must_link” or “should_not_link” of the data pairs [41]. For semi-supervised clustering, some prior information, such as labels of partial data, is assumed available and data usually comes from the same domain. Compared with semi-supervised clustering, the proposed transfer clustering algorithms are different in a sense that the data comes from more than one domain.

The rest of the paper is organized as follows. In section II, several representative PFC algorithms are briefly reviewed. In section III, the concepts of leveraging knowledge in clustering are introduced, and three TPFC algorithms are proposed. The experimental results are reported and discussed in section IV. Finally, conclusions are given in section V.

II. Prototype Based Clustering

Prototype based clustering is perhaps the most extensively studied clustering technique. It typically has the following form for objective function:

\[ J(U, \Theta) = f(U, \Theta), \]  

where \( U \) is the partition matrix and \( \Theta \) is the set of prototype parameters. Its aim is to optimize the objective function in Eq. (1) so as to obtain the optimal partition matrix \( U^* \) and the optimal prototype parameter set \( \Theta^* \). Let us firstly review the following three types of prototype based clustering, namely, cluster center prototype based clustering [18-21], cluster direction prototype based clustering [22, 46-50], and subspace cluster prototype based clustering [23, 51-53].

A. Cluster Center Prototype Based Clustering
Among the cluster center prototype based clustering methods, two well-known fuzzy algorithms are the fuzzy \(c\)-means (FCM) clustering algorithm [19] and the possibilistic \(c\)-means (PCM) clustering algorithm [20] with their objective functions defined respectively as:

**FCM:**

\[
\begin{align*}
\min_{U,V} & \quad J_{\text{FCM}} = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^m \|x_j - v_i\|^2 \\
\text{st.} & \quad u_{ij} \in [0,1], \quad \sum_{i=1}^{C} u_{ij} = 1, \quad 0 < \sum_{j=1}^{N} u_{ij} < N,
\end{align*}
\]

**PCM:**

\[
\begin{align*}
\min_{U,V} & \quad J_{\text{PCM}} = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^m \|x_j - v_i\|^2 + \sum_{i=1}^{C} \sum_{j=1}^{N} \eta_i (1 - u_{ij})^m \\
\text{st.} & \quad u_{ij} \in [0,1], \quad 0 < \sum_{j=1}^{N} u_{ij} < N,
\end{align*}
\]

where \(C\) is the number of clusters; \(x_j \in \mathbb{R}^d\) is the \(j\)th data sample; \(V = [v_1, \ldots, v_C]^T\) is the matrix of cluster centers with \(v_i \in \mathbb{R}^d\); \(U = [u_{ij}]_{C \times N}\) is the fuzzy/possibilistic partition matrix whose element denotes the membership of the \(j\)th data sample belonging to the \(i\)th class; \(m\) is the fuzzy index and \(\eta_i\) are positive parameters used by PCM. Another representative cluster center prototype based clustering method is the maximal entropy clustering (MEC) algorithm [21] whose objective function is defined as

**MEC:**

\[
\begin{align*}
\min_{U,V} & \quad J_{\text{MEC}} = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij} \ln(u_{ij}) + \gamma \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij} \ln(u_{ij}) \\
\text{st.} & \quad u_{ij} \in [0,1], \quad \sum_{i=1}^{C} u_{ij} = 1, \quad 0 < \sum_{j=1}^{N} u_{ij} < N,
\end{align*}
\]

where \(U = [u_{ij}]_{C \times N}\) is the probabilistic partition matrix whose element denotes the probability of the \(j\)th data sample belonging to the \(i\)th class, and \(\sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij} \ln(u_{ij})\) denotes the negative Shannon entropy.

**B. Cluster Direction Prototype Based Clustering**

Cluster direction prototype based clustering is also a representative branch of prototype based clustering. In this type of clustering methods, the directional vector is used to characterize the prototypes [23, 46-50]. The most representative cluster direction prototype based clustering methods are the \(k\)-plane clustering (KPC) algorithm [23] and the fuzzy KPC (FKPC) algorithm [47, 48]. The objective functions of KPC and FKPC can be written as:

**KPC/ FKPC:**

\[
\begin{align*}
\min_{U,V,b} & \quad J_{\text{KPC/FKPC}} = \sum_{i=1}^{K} \sum_{j=1}^{N} u_{ij}^m (x_j^T v_i + b_i) \\
\text{st.} & \quad u_{ij} \in [0,1] \quad \text{for FKPC}, \quad u_{ij} \in \{0,1\} \quad \text{for KPC}, \quad \sum_{i=1}^{K} u_{ij} = 1, \quad 0 < \sum_{j=1}^{N} u_{ij} < N, \quad v_i^T v_i = 1,
\end{align*}
\]

where \(K\) is the number of clusters; \(V = [v_1, \ldots, v_C]^T\) is the matrix of directional vectors of
the hyperplanes associated with clusters; \( \mathbf{b} = [b_1, \ldots, b_k] \) is the bias vector of the hyperplanes; \( \mathbf{U} = [u_{ij}]_{C \times N} \) is the fuzzy/crisp partition matrix; for the fuzzy index \( m \), \( m > 1 \) and \( m = 1 \) are set for FKPC and KPC, respectively. Another representative cluster direction prototype based clustering algorithm is the sphere k-means clustering (SKM) algorithm \([49, 50]\) with the following objective function:

\[
\text{SKM: } \min_{U,N} J_{\text{SKM}} = \sum_{i=1}^{K} \sum_{j=1}^{N} u_{ij}^T x_j \quad (3-2)
\]

\[
st. u_{ij} \in \{0,1\}, \sum_{i=1}^{K} u_{ij} = 1, \quad 0 < \sum_{j=1}^{N} u_{ij} < N, \quad v_j^T v_j = 1,
\]

where the variables are defined as in Eq.(3-1).

C. Subspace Cluster Prototype Based Clustering

Subspace cluster prototype based clustering has attracted more and more attentions of researchers in recent years \([22, 51-53]\). It is also commonly termed as soft subspace clustering. In this type of clustering methods, the prototype of each cluster is characterized by a cluster center and a weighting vector representing the soft subspace for this cluster. One representative algorithm is the fuzzy subspace clustering (FSC) algorithm \([51]\), whose objective function is defined as

\[
\text{FSC: } \min_{U,V,W} J_{\text{FSC}} = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij} \sum_{k=1}^{d} w_{ik} (x_{jk} - v_{ik})^2 + \sigma \sum_{i=1}^{C} \sum_{k=1}^{d} w_{ik}^T \quad (4-1)
\]

\[
st. u_{ij} \in \{0,1\}, \sum_{i=1}^{C} u_{ij} = 1, \quad 0 < \sum_{j=1}^{N} u_{ij} < N,
\]

\[
w_{ik} \in [0,1], \sum_{k=1}^{d} w_{ik} = 1, \quad 0 < \sum_{i=1}^{C} w_{ik} < C,
\]

where \( \mathbf{W} = [w_{ik}]_{C \times d} \) is the matrix of weighting vectors and \( \tau \) is the fuzzy index of the fuzzy weighting; \( \mathbf{U} = [u_{ij}]_{C \times N} \) is the crisp partition matrix, with other parameters defined as in Eq.(2-1). Another representative subspace cluster prototype based clustering algorithm is the entropy weighting k-means (EWKM) algorithm \([52]\), whose objective function is defined as

\[
\text{EWKM: } \min_{U,V,W} J_{\text{EWKM}} = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij} \sum_{k=1}^{d} w_{ik} (x_{jk} - v_{ik})^2 + \gamma \sum_{i=1}^{C} \sum_{k=1}^{d} w_{ik} \ln(w_{ik}) \quad (4-2)
\]

\[
st. u_{ij} \in \{0,1\}, \sum_{i=1}^{C} u_{ij} = 1, \quad 0 < \sum_{j=1}^{N} u_{ij} < N,
\]

\[
w_{ik} \in [0,1], \sum_{k=1}^{d} w_{ik} = 1, \quad 0 < \sum_{i=1}^{C} w_{ik} < C,
\]

where \( \sum_{i=1}^{C} \sum_{k=1}^{d} w_{ik} \ln(w_{ik}) \) is the negative Shannon entropy and other parameters are defined as in (4-1).

The prototype based clustering algorithms have been studied in depth in the past decades.
However, in the context of transfer learning, they are scarcely studied. In the next section, transfer learning is applied to prototype based fuzzy clustering and several TPFC algorithms are developed.

III. Transfer Prototype Based Fuzzy Clustering

In this section, a knowledge leverage based transfer learning mechanism is firstly introduced, based upon which three TPFC algorithms are proposed.

A. Knowledge Leverage Based TPFC

1) Framework

As mentioned in the introductory section, when the data is scarce or very limited, the existing prototype based clustering algorithms will no longer be effective and their performance will become inferior. A promising strategy to cope with this challenge is to learn by referring to the information in the source domain which might have useful information for the target domain. From the source domain, there are usually two kinds of information available, namely, the original data and the induced knowledge, as shown in Fig. 2. When the data in the source domain is used directly, it may not always be appropriate. In fact, it is highly difficult to control and balance the similarity and difference of data distributions between the two domains as there possibly exists a drifting of data distribution between domains. On the other hand, the influence of the induced knowledge from source domain can be conveniently controlled as the induced knowledge is much clearer and more concise than the data in source domain. Thus, we consider that it is more appropriate to exploit the use of knowledge rather than data from the source domain to help the clustering task in target domain. In this study, three TPFC algorithms with such a strategy will be presented. Fig.3 shows an illustration of the framework of knowledge leverage based TPFC.

![Fig.2 Useful information from source domain for the clustering task in target domain](image)
2) Objective Function

The following general objective function is proposed for TPFC based on the aforementioned knowledge leverage based transfer learning strategy:

$$\min_{U, \Theta} J_{TPFC} = f_i(U, \Theta) + f_s(U, \Theta, \hat{\Theta}),$$

which consists of two different functional terms. The first functional term $f_i(U, \Theta)$ is inherited from the classical PFC algorithms and is used to learn from the data in target domain. The second functional term $f_s(U, \Theta, \hat{\Theta})$ is introduced to carry out knowledge leveraging from the source domain. This term can be in various forms, of which the design depends on what prototype based clustering algorithm is formulated.

With this general objective function, three TPFC algorithms, namely, transfer fuzzy $c$-mean (TFCM), transfer fuzzy $k$-plane clustering (TFKPC) and transfer fuzzy subspace clustering (TFSC), are proposed in this paper. In fact, transfer clustering algorithm for other prototype based clustering algorithms such as possibilistic $c$-mean and entropy weighting $k$-mean subspace clustering can be similarly formulated. Before proceeding to present the new transfer clustering algorithms, let us give the assumptions of our formulations. First, the knowledge about the clusters in source domain has been induced in advance by some learning procedures and is available for the clustering task of target domain. Second, the number of
clusters in target and source domains is the same. Our TPFC algorithms to be proposed are based on these two assumptions. The second one is quite restrictive but there exist difficulties in the practical design for the corresponding algorithms if it is not in place. We consider that developing TPFC algorithms without such an assumption is a valuable and significant research topic in the future.

B. TPFC Algorithms

1) Transfer FCM

Among the cluster center prototype based clustering methods, fuzzy c-means [19] is of high popularity and is adopted here to develop our first TPFC algorithm, i.e. transfer FCM (TFCM). To do so, the following objective function is defined:

\[
\min_{u, \mathcal{V}} J_{TFCM} = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij} \| \mathbf{x}_j - \mathbf{v}_i \|^2 + \lambda_1 \sum_{i=1}^{C} \sum_{j=1}^{N} u^m_{ij} \| \mathbf{x}_j - \mathbf{v}_i \|^2 \\
+ \lambda_2 \sum_{i=1}^{C} (\sum_{j=1}^{N} u^m_{ij}) \left\| \mathbf{v}_i - \mathbf{v}^* \right\|^2
\]

where \( u_{ij} \in [0, 1] \), \( \sum_{i=1}^{C} u_{ij} = 1 \), \( 0 < \sum_{j=1}^{N} u_{ij} < N \), and \( \mathbf{v}_i \) denotes the \( i \)th cluster center obtained in the source domain. The concepts behind such a knowledge leverage based transfer learning mechanism can be given as follows.

(i) The first term in Eq. (6) is directly inherited from the classical FCM, which is mainly used to learn from the data available in target domain.

(ii) The second and third terms are used to learn with the knowledge from source domain, where the knowledge is the cluster centers obtained in source domain by a certain clustering task.

(iii) By having the second term, the clustering procedure is apt to obtaining the partition matrix by using the cluster centers obtained in source domain, which implies the use of a supervised learning procedure, such as 1-nearest neighbor (1-NN) classification, in the target domain.

(iv) By having the third term, the desired cluster centers will be apt to approximate the ones obtained in source domain, and thus guiding the clustering procedure to learn from the knowledge of source domain.

(v) The parameters \( \lambda_1 \) and \( \lambda_2 \), are used to balance the influence of data in target domain and knowledge in source domain. A parameter analysis will be given in our experimental
By using similar optimization strategy in FCM, one can obtain the following learning rules based on Eq. (6).

$$

v_j = \frac{\sum_{j=1}^{N} u_{ij}^m x_j + \lambda_2 \sum_{j=1}^{N} u_{ij}^m \tilde{v}_j}{\sum_{j=1}^{N} u_{ij}^m + \lambda_2 \sum_{j=1}^{N} u_{ij}^m},
$$

(7)

$$

u_{ij} = \left\{ \begin{array}{l}
1 \\
\frac{1}{\left[ \sum_{j=1}^{N} |x_j - v_j|^2 + \lambda_1 |x_j - \tilde{v}_j|^2 + \lambda_2 |\tilde{v}_j - v_j|^2 \right]^{\frac{1}{2(m-1)}}} \\
\end{array} \right\}^{1/(m-1)}
$$

(8)

Based on Eqs.(7) and (8), the TFCM algorithm is described below.

**TFCM Algorithm**

1) Initialize the number of iterations $t = 0$ and the partition matrix $U(0)$ randomly; Set the maximum number of iterations $t_{\text{max}}$ and threshold $\varepsilon$; Set the balance parameters $\lambda_1, \lambda_2$.
2) Update the matrix of cluster centers $V(t)$ using (7);
3) Set $t = t + 1$;
4) Update the partition matrix $U(t)$ using (8);
5) If $\|U(t) - U(t-1)\| < \varepsilon$ or $t = t_{\text{max}}$, then terminate; else go to 2).

For other popular center-prototype based clustering algorithms, such as PCM [20], one can develop the corresponding transfer learning versions in a similar manner.

**2) Transfer FKPC**

Among the cluster direction prototype based clustering methods, FKPC [47] is adopted here to develop the corresponding transfer clustering algorithm, i.e., TFKPC. First, the following objective function is considered:

$$

\min_{v, b} J_{\text{FKPC}} = \sum_{i=1}^{K} \sum_{j=1}^{N} u_{ij}^m (x_j^T v_i + b_i)^2 + \lambda \cdot \sum_{i=1}^{K} \sum_{j=1}^{N} u_{ij}^m (x_j^T \tilde{v}_i + \tilde{b}_i)^2
$$

(9)

st. $u_{ij} \in [0, 1] \ , \sum_{i=1}^{K} u_{ij} = 1, \ 0 < \sum_{j=1}^{N} u_{ij} < N, \ v_i^T v_i = 1$

where the first term is directly inherited from the original FKPC algorithm and the second term is designed to learn from the knowledge of source domain. Here the second term implies that the proposed algorithm will learn the partition matrix from cluster prototypes, i.e., plane parameters, $\tilde{v}_i, \tilde{b}_i$, obtained in source domain. Thus, by minimizing Eq.(9), the clustering procedure can learn from the data of target domain and the knowledge of source domain.
simultaneously. The parameter $\lambda$ is used to balance the influence of data in target domain and knowledge in source domain.

Here, we want to give an additional explanation about the difference between Eq.(6) and Eq.(9). When comparing Eq.(6) and Eq.(9), we can see that Eq.(6) has introduced two terms, i.e., the assignment of data to the source domain clusters and the matching of the clusters obtained in source and target domains for TFCM, however, only the former term, i.e., the assignment of data to the source domain clusters is used in Eq.(9) for TFKPC. This can be justified as follows. When we design the second part of Eq. (5), i.e., the term for knowledge leveraging, certain factors should be considered. First, the introduced terms should be reasonable, which can really give the useful information for knowledge leveraging. Second, the introduced terms should not bring great difficulty for the solution of the developed objective functions. When the matching term of the clusters obtained in the source and target domains was used to construct the objective function for TFKPC, we found that it is very difficult to solve the corresponding objective function. Thus, only the assignment of data to the source domain clusters is used to develop the knowledge leveraging term of TFKPC.

Based on similar optimization strategy in FKPC, we can obtain the following learning rules (Eqs.[10]-[12]) for TFKPC:

$$X(D_i - S_i)X^T v_i = 2\lambda v_i$$  \hspace{1cm} (10)

with

$$X = [x_1, \cdots, x_N], \quad D_i = \begin{bmatrix} u_{i1}^m & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & u_{iN}^m \end{bmatrix}, \quad S_i = \frac{u_i u_i^T}{u^T_i e}, \quad u_i = (u_{i1}^m, u_{i2}^m, \cdots u_{iN}^m)^T,$$

the eigenvector associated with the minimal positive eigenvalue obtained by Eq.(10) is just the desired hyperplane direction vector $v_i$;

$$b_i = \frac{\sum_{j=1}^{N} u_{ij}^m v_i^T x_j}{\sum_{j=1}^{N} u_{ij}^m}$$  \hspace{1cm} (11)

$$u_{ij} = \frac{1}{\left[(x_j^i v_i + b_j)^T + \lambda_i (x_j^i \tilde{v}_i + \tilde{b}_j)^T\right]^{1/(m-1)}}$$  \hspace{1cm} (12)

Thus, the TFKPC algorithm can be described as follows.
**TFKPC Algorithm**

1) Initialize the number of iterations \( t = 0 \) and the partition matrix \( U(0) \) randomly; Set the maximum number of iterations \( t_{\text{max}} \) and threshold \( \varepsilon \); Set the balance parameters \( \lambda \).

2) Update the matrix of directional vectors of planes \( V(t) \) using (10);

3) Update the bias of the planes \( b_j(t) \) using (11);

4) Let \( t = t + 1 \);

5) Update the partition matrix \( U(t) \) using (12);

6) If \( \| U(t) - U(t-1) \| < \varepsilon \) or \( t = t_{\text{max}} \), then terminate; else go to 2).

**3) Transfer FSC**

Among the existing subspace cluster prototype based clustering methods, the FSC [51] is adopted to study the corresponding transfer learning version, i.e., TFSC. Thus, the following objective function is proposed:

\[
\min_{U,N,W} J_{\text{TFSC}} = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij} \sum_{d=1}^{D} w_{i,j}^d \left( x_{i,j} - v_{i,j} \right)^2 + \sigma \cdot \sum_{i=1}^{C} \sum_{k=1}^{D} w_{i,k}^d \\
+ \lambda_1 \cdot \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij} \sum_{k=1}^{D} \tilde{w}_{i,j}^d \left( x_{i,j} - \tilde{v}_{i,j} \right)^2 + \lambda_2 \cdot \sum_{i=1}^{C} \left( \sum_{j=1}^{N} u_{ij}^w \right) \sum_{k=1}^{D} \tilde{w}_{i,k}^d \left( \tilde{v}_{i,k} - v_{i,k} \right)^2
\]

\[
\text{st. } u_{ij} \in [0,1], \quad \sum_{i=1}^{C} u_{ij} = 1, \quad 0 < \sum_{j=1}^{N} u_{ij} < N, \quad w_{i,j} \in [0,1], \quad \sum_{k=1}^{D} w_{i,j}^d = 1. \quad (13)
\]

In Eq.(13), the first two terms are directly inherited from the classical FSC algorithm except that we have extended the hard partition \( u_{ij} \in [0,1] \) in FSC to the fuzzy partition \( u_{ij} \in [0,1] \) for easy optimization and knowledge leveraging. The third and fourth terms are used to learn from the knowledge of source domain. The third term implies that we can learn the partition matrix by referring to subspace cluster prototypes obtained in the source domain, characterized by the cluster centers and weighting vectors, i.e., \( \tilde{v}_{i,k} \) and \( \tilde{w}_{i,k}^d \). The fourth term intends to make the desired cluster centers apt to approximate the ones obtained in the source domain under the subspace constraints of source domain. Thus, by minimizing Eq. (13), the clustering procedure can learn from the data of target domain and the knowledge of source domain simultaneously. Again, by following the optimization strategy in FSC, we can obtain the following learning rules:

\[
v_{i,k} = \frac{\sum_{j=1}^{N} u_{ij}^w v_{i,j}^d x_{i,j} + \lambda_2 \cdot \sum_{j=1}^{N} u_{ij}^w \tilde{v}_{i,j}^d x_{i,j} + \lambda_3 \cdot \sum_{j=1}^{N} u_{ij}^w \tilde{v}_{i,k}^d \tilde{v}_{i,k} \sum_{j=1}^{N} u_{ij}^w \tilde{w}_{i,k}^d}{\sum_{j=1}^{N} u_{ij}^w \tilde{w}_{i,k}^d + \lambda_2 \cdot \sum_{j=1}^{N} u_{ij}^w \tilde{w}_{i,k}^d + \lambda_3 \cdot \sum_{j=1}^{N} u_{ij}^w \tilde{w}_{i,k}^d}, \quad (14)
\]
Based on Eqs. (14)-(17), the TFSC algorithm can be described below.

**TFSC Algorithm**

1) Initialize the number of iterations \( t = 0 \) and the partition matrix \( U(0) \) randomly; Set the maximum number of iterations \( t_{\text{max}} \) and threshold \( \varepsilon \);

   Set the balance parameters \( \lambda_1, \lambda_2 \).

2) Update the matrix of cluster centers \( V(t) \) using (14);

3) Update the matrix of weighting vectors \( W(t) \) using (15);

4) Set \( t = t + 1 \);

5) Update the partition matrix \( U(t) \) using (16);

6) If \( \|U(t) - U(t-1)\| < \varepsilon \) or \( t = t_{\text{max}} \), then terminate; else go to 2).

---

**C. Computational Complexity and Convergence Analysis**

The computational complexities of the proposed algorithms can be described as follows. For the TFCM and TFKPC algorithms, their computational complexities are \( O(TNC + TC) \), where \( T \) is the total number of iterations, \( N \) is the size of dataset, and \( C \) is the number of clusters. In other words, they are of the same order as those of the classical FCM and FKPC algorithms. For the TFSC algorithm, its computational complexity is \( O(TNC + TC + TCd) \), where \( d \) is the number of features, which is of the same order as that of the classical FSC.

For the convergence of the three proposed TPFC algorithms, Zangwill’s convergence theorem can be adopted as in the convergence studies of many existing clustering algorithms, e.g. [51]. Let’s recall Zangwill’s convergence theorem 1 [54, 55].

**Theorem 1** (Zangwill’s convergence theorem): Let \( A: V \rightarrow P(V) \) be a point-to-set map. Given a point \( z^{(0)} \in V \), let \( A \) determine an algorithm that generates the sequence \( \{z^{(k)}\} \). Let \( S \subset V \) be the solution set. Assume that

(i) All points \( z^{(k)} \) are in a compact subset of \( V \).

(ii) There is a continuous function \( J: V \rightarrow \mathbb{R} \), satisfying (a) if \( z \notin S \), then for any \( y \in A(z) \), \( J(y) < J(z) \); (b) if \( z \in \Omega \), then either the algorithm terminates or for any \( y \in A(z) \), \( J(y) \leq J(z) \).
(ii) The map \(A\) is closed at \(z\) if \(z \notin S\).

Then, either the algorithm stops at a solution or the limit of any converged subsequence is a solution.

Based on the above theorem, our aim is to prove that the three conditions in theorem 1 are satisfied by the proposed three algorithms. Here, we first prove the convergence of the TFCM algorithm. With similar strategy in [51], we can prove the corresponding theorems 2a-2c with the auxiliary definitions 1-4. In order to save space of this paper, we omit the details of proofs about theorems 2a-2c.

**Definition 1.** Let \(D = \{x_1, \cdots, x_N\}\) be a finite data set in the Euclidean space \(\mathbb{R}^d\), the set of all fuzzy \(C\)-partition of \(D\) is denoted by \(M_C\), i.e., \(M_C = \{U \in B_{CN} \mid U\) satisfies the constraint \(A\}\), where \(B_{CN}\) denotes the vector space of all real \(C \times N\) matrices and constraint \(A\) is defined as

\[
\begin{align*}
&u_{ij} \in [0, 1], \ 1 \leq i \leq C, \ 1 \leq j \leq N, \\
&\sum_{i=1}^{C} u_{ij} = 1, \ 1 \leq j \leq N, \\
&0 < \sum_{j=1}^{N} u_{ij} < N, \ 1 \leq i \leq C.
\end{align*}
\]

**Definition 2.** Let \(G_1 : M_C \rightarrow R^{Cd}\) be a function defined as \(G_1(U) = Z = (z_1, z_2, \cdots, z_C)^T\), where the vectors \(z_i = (z_{i1}, z_{i2}, \cdots, z_{id})^T \in R^d, \ 1 \leq i \leq C\), are computed via Eq. (10) using \(U\).

**Definition 3.** Let \(G_2 : R^{Cd} \rightarrow M_C\) be a point-to-set map defined as \(G_2(Z) = \{U \mid U\) satisfies constraint \(A\}\).

**Definition 4.** \(T_{\text{TFCM}}(V, U) = \{\tilde{V}, \tilde{U} \mid \tilde{V} = G_1(U), \tilde{U} \in G_2(\tilde{V})\}\), i.e., TFCM iteration can be expressed using a point-to-set map \(T_{\text{TFCM}} : R^{Cd} \times M_C \rightarrow P(R^{Cd} \times M_C)\) defined by the composition \(T_{\text{TFCM}} = A_2 \circ A_1\), where \(A_1(V, U) = (G_1(U), U)\) and \(A_2(V, U) = \{(V, U) \mid U \in G_2(V)\}\).

Thus, \((\tilde{V}, \tilde{U}) \in T_{\text{TFCM}}(V, U) = A_2 \circ A_1(V, U) = A_2(G_1(U), U) = A_2(\tilde{V}, \tilde{U}) = \{(V, U) \mid U \in G_2(\tilde{V})\}\).

**Theorem 2a.** Let \(D = \{x_1, \cdots, x_N\}\) contain at least \(C\) (\(< N\) ) distinct points and let \((V^{(0)}, U^{(0)})\) be the starting point of iteration of \(T_{\text{TFCM}}\) with \(U^{(0)} \in M_C\) and \(V^{(0)} = G_1(U^{(0)})\).

Then, the iteration sequence \(\{V^{(r)}, (U^{(r)})\}_{r=1}^{\infty}\), \(r=1, 2, \cdots\), is contained in a compact subset of \(R^{Cd} \times M_C\).

**Theorem 2b.** Let \(m > 1, \ \lambda_1 \geq 0\) and \(\lambda_2 \geq 0\) are fixed and \(D = \{x_1, \cdots, x_N\}\) contain at least
\( C \ ( < N ) \) distinct points. Let the solution set \( S \) of the optimization problem
\[
\min_{(V,U) \in \mathbb{R}^{C_d} \times M_c} J_{\text{TFCM}} (V, U)
\]
be defined as
\[
S = \{ (\hat{V}, \hat{U}) \in \mathbb{R}^{C_d} \times M_c \mid J_{\text{TFCM}} (\hat{V}, \hat{U}) \leq J_{\text{TFCM}} (V, U) \ \forall (V, U) \in M_c \ \text{and} \ J_{\text{TFCM}} (\hat{V}, \hat{U}) < J_{\text{TFCM}} (V, U) \ \forall (V, U) \neq (\hat{V}, \hat{U}) \}
\]
Let \((\hat{V}, \hat{U}) \in \mathbb{R}^{C_d} \times M_c\). Then, \( J_{\text{TFCM}} (\hat{V}, \hat{U}) \leq J_{\text{TFCM}} (V, U) \) for every \((V, U) \in T_{\text{TFCM}} (\hat{V}, \hat{U})\) and the inequality is strict if \((\hat{V}, \hat{U}) \notin S\).

**Theorem 2c.** Let \( m > 1, \ \lambda_i \geq 0 \) and \( \lambda_z \geq 0 \) are fixed and \( D = \{ x_1, \ldots, x_N \} \) contain at least \( C \ ( < N ) \) distinct points. Then the point-to-set map \( T_{\text{TFCM}} : \mathbb{R}^{C_d} \times M_c \to \mathbb{P} (\mathbb{R}^{C_d} \times M_c) \) is closed at every point in \( \mathbb{R}^{C_d} \times M_c \).

Since the function defined in Eq.(6) is continuous, the following theorem, which establishes the convergence for the TFCM algorithm, follows immediately from Theorems 2a-2c and Zangwill’s convergence Theorem1.

**Theorem 3.** (Convergence of TFCM). Let \( D = \{ x_1, \ldots, x_N \} \) contain at least \( C \ ( < N ) \) distinct points and let \( J_{\text{TFCM}} (V, U) \) define as in Eq.(6). Let \((V(0), U(0))\) be the starting point of iterations with \( T_{\text{TFCM}} \) with \( U(0) \in M_c \) and \( V(0) = G_i (U(0)) \). Then, the iteration sequence \( \{ V(r), U(r) \}, \ r = 1, 2, \ldots \), either terminates as a point \((V^*, U^*)\) in the solution set \( S \) or there is a subsequence converging to a point in \( S \).

Similarly, the convergence of TFKPC and TFSC algorithms can be proved in a similar manner and is omitted here.

**IV. Experimental Results**

The proposed three TPFC algorithms, i.e., TFCM, TFKPC and TFSC, have been extensively evaluated on synthetic and real-world datasets. In this section, the indices used for performance evaluation and the experimental setup are first described. Then, the performance of the proposed algorithms on synthetic and real-world text datasets is reported and discussed. A further comparison with other related algorithms is conducted on several synthetic and real-world datasets. All the experiments were implemented with Matlab codes on a computer with 1.66GHz CPU and 2GB RAM.
A. Performance Indices and Experimental Setup

Two metrics, i.e., the rand index (RI) and the normalized mutual information (NMI) [56], are used to evaluate the performance of the clustering algorithms. RI is commonly defined as

\[ RI = \frac{f_{00} + f_{11}}{N(N - 1)/2} \]

where \( f_{00} \) is the number of pairs of data points having different class labels and belonging to different clusters; \( f_{11} \) is the number of pairs of data points having the same class labels and belonging to the same clusters; \( N \) is the size of the whole dataset. NMI is defined and computed according to the formula

\[ NMI = \frac{\sum_{i=1}^{C} \sum_{j=1}^{C} N_{i,j} \log \frac{N \cdot N_{i,j}}{N_i \cdot N_j}}{\sqrt{\sum_{i=1}^{C} N_i \log \frac{N_i}{N} \cdot \sum_{j=1}^{C} N_j \log \frac{N_j}{N}}} \]

where \( N_{i,j} \) is the number of agreements between cluster \( i \) and class \( j \); \( N_i \) is the number of data points in cluster \( i \); \( N_j \) is the number of data points in class \( j \) and \( N \) is the size of the whole dataset. Both RI and NMI take a value within the interval \([0, 1]\). The higher the values, the better the clustering performance is.

When the RI and NMI are computed, the prior labels of the data must be available. In addition to them, an index that does not require the labels of data is also used for performance evaluation. Since most of the adopted methods for comparison are fuzzy measure based methods, the fuzzy validity measure is naturally very appropriate. Here, the classical fuzzy validity measure, i.e., Xie-Beni (XB) index, and its two modified versions were adopted to further evaluate and compare the performance of different fuzzy measure based algorithms.

The classical XB index and its extended version \( XB_m \) can be respectively defined for FCM as:

\[ XB(U, V; X) = \frac{1}{N} \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^2 \| x_j - v_i \|^2 = \left( \frac{\sigma}{N} \right)^2 \frac{1}{\text{sep}(V)} \]

\[ XB_m(U, V; X) = \frac{1}{N} \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^m \| x_j - v_i \|^2 = \left( \frac{\sigma}{N} \right)^m \frac{1}{\text{sep}(V)} \]

Usually, the smaller the index values, the better the clustering performance is.
As the XB index was originally designed for FCM only, two modified XB indices are presented to evaluate the direction prototype and subspace cluster prototype based algorithms here. For FKPC and TFKPC, we propose to use the following modified XB index:

\[
XB_{FKPC}(U, V; X) = \frac{1}{N} \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^{m} (x_{ij} + b_{j})^{2} \min_{i \neq j} \left\{ \|v_{i} - v_{j}\| + |b_{i} - b_{j}| \right\} = \left( \frac{\sigma}{N} \right) \text{sep}(V).
\]

For FSC and TFSC, another modified XB index which includes the subspace information is proposed:

\[
XB_{FSC}(U, V, W; X) = \frac{1}{N} \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^{m} \sum_{k=1}^{d} w_{ik}^{p} (x_{ik} - v_{ik})^{2} \min_{i \neq j} \left\{ \sum_{k=1}^{d} w_{ik} (v_{ik} - v_{jk})^{2} \right\} = \left( \frac{\sigma}{N} \right) \text{sep}(V, W).
\]

The proposed transfer clustering algorithms are compared with the corresponding non-transfer learning counterparts in sub-sections IV-B & IV-C. In section IV-D, a comparison with several related learning algorithms is presented. In section IV-E some discussions are presented. In our experiments, the clustering process of all algorithms is repeated 20 times for each parameter setting on a dataset. The maximum number of iterations, i.e., \(t_{\text{max}}\), is set to 100 for all iteration based clustering algorithms.

**B. Synthetic Datasets**

1) **Evaluation of TFCM Algorithm**

In this subsection, a synthetic dataset, denoted as D1, with predetermined cluster structures is used to evaluate the performance of the proposed TFCM algorithm. Data was generated for the source domain and the target domain, respectively. Each domain contains three clusters with different Gaussian distributions as shown in Fig 4 and the parameters used to generate the data are listed in Table I. From Table I, we can see that the three clusters in the two domains have similar but different means and covariance. Fig. 4 shows that the data in the target domain is scarce while the data in the source domain should be enough to guide the clustering task in the target domain.
Table I Parameters used to generate the synthetic dataset (D1) for evaluating TFCM algorithm

| Domain | Source domain | Target domain |
|--------|---------------|---------------|
| Cluster | Cluster-1 | Cluster-2 | Cluster-3 | Cluster-1 | Cluster-2 | Cluster-3 |
| **u** | $[-1, 8]$ | $[15, 8]$ | $[9, 27]$ | $[-1, 12]$ | $[15, 13]$ | $[9, 25]$ |
| **Σ** | $[10, 0]$ | $[25, 0]$ | $[30, 0]$ | $[9.5, 0]$ | $[23, 0]$ | $[28, 0]$ |
| Size of dataset | 600 | 80 |

* u and Σ denote the mean and covariance of each class, respectively.

Fig. 4 Synthetic dataset (D1) generated for evaluating TFCM algorithm: (a) data generated in the source domain; (b) data generated in the target domain.

The performance of the TFCM algorithm is compared with that of the classical FCM algorithm in order to evaluate its transfer learning abilities. The clustering results, in terms of RI, NMI and XB values for 20 runs, are tabulated in Tables II-V.

Table II Performance (optimized parameters) comparison of FCM and TFCM on synthetic dataset D1

| Index | FCM | TFCM |
|-------|-----|------|
| Source domain | Target domain | Target domain |
| NMI | 0.5845 | 0.8502 | 0.6908 | 0.1372 | 0.7187 | 0.7468 | 0.7352 | 0.0141 | 0.8084 | 0.8084 | 2.34e-16 |
| RI | 0.7071 | 0.9622 | 0.8091 | 0.1317 | 0.8631 | 0.8788 | 0.8625 | 0.0092 | 0.9117 | 0.9117 | 0.9117 | 0 |
| XB | 0.1078 | 0.5999 | 0.1570 | 0.1556 | 0.1549 | 0.1568 | 0.1560 | 2.12e-03 | 0.1291 | 0.1291 | 0.1291 | 3.70e-17 |

In Table II, the optimized results with respect to different parameters involved obtained by the two algorithms are reported. It shows that the optimized clustering results of TFCM are obviously superior to those of the classical FCM algorithm. For example, in terms of the NMI index, TFCM obtains the mean and standard deviation as 0.8084 and 2.34e-16 respectively for the data in target domain, while FCM only obtains the corresponding values as 0.7352 and 0.0141.

In addition, Table II shows that FCM has obtained better mean values of different
performance indices in the target domain rather than in the source domain. It can be explained by the fact that the clusters in the source domain of Fig.4 are unbalanced, i.e., two smaller clusters and one bigger cluster, which make the FCM more sensitive to initialization. For the data in target domain, the three clusters are more balanced. Thus, we can observe that the mean clustering result ($RI$: 0.8091) in the source domain for 20 runs is inferior to that ($RI$: 0.8625) obtained in the target domain due to initialization sensitivity. However, we can also see that the best clustering result ($RI$: 0.9622) obtained in a certain run for the source domain is obviously better than that ($RI$: 0.8788) obtained in the target domain. This indicates that when FCM has good initialization in both domains, the source domain can contribute useful information to the target domain in obtaining better clustering results.

Tables III-V record the means and standard deviations of the $RI$, $NMI$ and $XB$ values obtained by the proposed TFCM algorithm using different parameter settings. Based on these results, the following observations can be obtained.

(1) When the two parameters in Eq.(6) are set as $\lambda_1 = 0$ and $\lambda_2 = 0$, the TFCM algorithm obtains the worst clustering effect as indicated by the mean $NMI$ and $RI$ values in Part A of Tables III and IV, while clustering performances have been improved when $\lambda_1$ and $\lambda_2$ are set to different positive values. Here, when $\lambda_1 = 0$ and $\lambda_2 = 0$, TFCM degenerates to the classical FCM. Thus this confirms the advantage of knowledge leveraging in our transfer clustering algorithm.

(2) From Part B of Tables III, IV and V, we can see that when the knowledge of source domain is used, the standard deviations of the three indices ($RI$, $NMI$ and $XB$) have been reduced. This implies that the introduction of the knowledge of source domain has enhanced the stability/robustness of the algorithm.

(3) Tables III-V show that the transfer of the knowledge of source domain to the clustering in target domain has to be properly controlled. Although the knowledge of source domain can improve the clustering performance, the improvement will be degraded if very large $\lambda_1$ and $\lambda_2$ values are used. From our experimental results, setting $\lambda_1 \in [0.5, 1]$ and $\lambda_2 \in [0.5, 1]$ can obtain more promising clustering performance on the adopted synthetic dataset D1.
(4) Tables III-V show that the trends of the clustering results with the three indices are similar under different parameter settings, i.e., when the value of one index is better, usually but not absolutely, the other two are also better.

(5) Tables III-V also show that $XB$ index is more sensitive to the parameter setting than $RI$ and $NMI$ indices. $XB$ index only obtains the best mean value in a small range of parameter values, while $RI$ and $NMI$ indices have obtained the best mean values in a much larger range of parameter values. However, $XB$ index also has its own advantage, i.e., the label information is not required in the calculation of the index values.

| $\lambda$ | 0   | 0.005 | 0.1  | 0.5  | 0.7  | 1    | 1.5  | 10   | 50   | 100  |
|-----------|-----|-------|------|------|------|------|------|------|------|------|
| $\lambda_c$ | 0.7352 | 0.7405 | 0.7468 | 0.7468 | 0.7746 | 0.8084 | 0.8084 | 0.7584 | 0.7584 | 0.7584 |
| 0.005    | 0.7426 | 0.7325 | 0.7468 | 0.7468 | 0.7746 | 0.8084 | 0.8084 | 0.7584 | 0.7584 | 0.7584 |
| 0.1     | 0.7468 | 0.7468 | 0.7468 | 0.7746 | 0.7746 | 0.8084 | 0.8084 | 0.7584 | 0.7584 | 0.7584 |
| 0.5   | 0.7468 | 0.7468 | 0.7746 | 0.8084 | 0.8084 | 0.8084 | 0.8084 | 0.7584 | 0.7584 | 0.7584 |
| 0.7 | 0.7746 | 0.7746 | 0.7746 | 0.8084 | 0.8084 | 0.8084 | 0.8084 | 0.7584 | 0.7584 | 0.7584 |
| 1 | 0.8084 | 0.8084 | 0.8084 | 0.8084 | 0.8084 | 0.8084 | 0.8084 | 0.7584 | 0.7584 | 0.7584 |
| 1.5 | 0.8084 | 0.8084 | 0.8084 | 0.8084 | 0.8084 | 0.8084 | 0.8084 | 0.7584 | 0.7584 | 0.7584 |
| 10 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 |
| 50 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 |
| 100 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 | 0.7584 |

| $\lambda$ | 0   | 0.005 | 0.1  | 0.5  | 0.7  | 1    | 1.5  | 10   | 50   | 100  |
|-----------|-----|-------|------|------|------|------|------|------|------|------|
| $\lambda_c$ | 0.0141 | 0.0102 | 0   | 0   | 1.17e-16 | 2.34e-16 | 2.34e-16 | 0   | 0   | 0   |
| 0.005    | 0.0089 | 0.0166 | 0   | 0   | 1.17e-16 | 2.34e-16 | 2.34e-16 | 0   | 0   | 0   |
| 0.1     | 0   | 0   | 0   | 1.17e-16 | 1.17e-16 | 2.34e-16 | 2.34e-16 | 0   | 0   | 0   |
| 0.5   | 0   | 0   | 0   | 1.17e-16 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 0   | 0   | 0   |
| 0.7 | 1.17e-16 | 1.17e-16 | 1.17e-16 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 0   | 0   | 0   |
| 1 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 0   | 0   | 0   |
| 1.5 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 2.34e-16 | 0   | 0   | 0   |
| 10 | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
| 50 | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
| 100 | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

Part A: Means

Part B: Standard deviations
In this subsection, another synthetic dataset, denoted as D2, with predetermined cluster structures is used to evaluate the performance of the proposed TFKPC algorithm. Data was generated for the source domain and the target domain, respectively, where each domain
contains three clusters located at different hyperplanes with different cluster direction prototypes as shown in Fig 5. The parameters used to generate the data are listed in Table VI. From Table VI, we can see that the three clusters in the two domains have similar but different hyperplane parameters. In particular, Gaussian noise has been added in the synthetic dataset with the given noise parameter, i.e., the standard deviation $\sigma$ as shown in Table VI. Fig. 5 shows that the data in the target domain is scarce while the data in the source domain is sufficient and helpful to improve the clustering effect in the target domain.

| Domain | Source domain | Target domain |
|--------|---------------|---------------|
| Cluster | Cluster-1  | Cluster-2  | Cluster-3  | Cluster-1  | Cluster-2  | Cluster-3  |
| $v$     | [0.9487]   | [-0.9487]  | [-0.9487]  | [0.9615]   | [-0.9285]  | [-0.9285]  |
|         | [0.3162]   | [0.3162]   | [0.3162]   | [0.2747]   | [0.3714]   | [0.3714]   |
| $b$     | -6.9570    | 3.4785     | -5.3759    | -4.9450    | 3.3425     | -7.0564    |
| $\sigma^*$ | 0.3162    | 0.3162     | 0.3162     | 0.2747     | 0.3714     | 0.3714     |
| Size    | 600         | 60          |
| $x_i$   | [-0.05, 10] | [0.05, 10]  | [-10, 5]   | [-2, 8]    | [0.05, 12] | [-10, 5]   |

*denotes the standard deviation of Gaussian noise with zero means.

The performance of TFKPC is compared with that of the classical FKPC algorithm and the results like those in previous subsection are recorded in Tables VII-X. From the results, one can obtain the following observations.

1. Table VII shows that the parameter optimized clustering results of TFKPC are obviously superior to those of the classical FKPC algorithm. For example, TFKPC obtained the mean and standard deviation as 0.8898 and 0 on the data in the target domain with $RI$ index, but FKPC only obtained the corresponding values as 0.7506 and 0.1472.

2. In particular, when $\lambda=0$, TFKPC is degenerated into FKPC. Tables VIII-X show that

Fig.5 Synthetic dataset (D2) generated for evaluating TFKPC algorithm. (a) data generated in the source domain; (b) data generated in the target domain

The performance of TFKPC is compared with that of the classical FKPC algorithm and the results like those in previous subsection are recorded in Tables VII-X. From the results, one can obtain the following observations.

1. Table VII shows that the parameter optimized clustering results of TFKPC are obviously superior to those of the classical FKPC algorithm. For example, TFKPC obtained the mean and standard deviation as 0.8898 and 0 on the data in the target domain with $RI$ index, but FKPC only obtained the corresponding values as 0.7506 and 0.1472.

2. In particular, when $\lambda=0$, TFKPC is degenerated into FKPC. Tables VIII-X show that
TFKPC with \( \lambda = 0 \) usually obtain the worse clustering performance while the clustering performance of TFKPC improves when \( \lambda \) is set to non-zero positive values. This once again confirms the knowledge leveraging ability of the proposed transfer clustering algorithm.

(3) From Tables VIII-X, we can see that the FKPC algorithm is more sensitive to initialization, which can be observed from the higher standard deviation values of the performance indices obtained in 20 runs. The proposed TFKPC has introduced the knowledge leveraging term in the objective function, which can effectively weaken the sensitivity to initialization to some extent, and thus the clustering results become more stable. For example, in Table VIII, when \( \lambda \) is smaller, meaning that the influence of knowledge leveraging is smaller, the clustering results of 20 runs are more fluctuated. On the other hand, when \( \lambda \) becomes larger, the trend of standard deviation of the performance index shows that the clustering results are becoming steadier. The observation does not always hold as the initialization of TFKPC algorithm is random for 20 runs. For example, in Table VIII, the standard deviations of RI are 0.0524 and 0.0941 for \( \lambda = 0.5 \) and 0.7, respectively. But in general, when the knowledge of source domain is used effectively, the standard deviations of the three performance indices can be reduced. This demonstrates that the introduction of knowledge from the source domain really enhances the stability of the proposed TFKPC algorithm.

(4) Tables VIII-X also show that although the knowledge of source domain may effectively improve the clustering performance, it should not be overused. The very large \( \lambda \) values cannot result in getting the optimal clustering results. As in TFCM, an appropriate \( \lambda \) value is needed for TFKPC to balance the influence of knowledge of the source domain and data of the target domain. From our experimental results, \( \lambda \in [0.7, 1] \) is appropriate for TFKPC to obtain much better clustering effect on the synthetic dataset D2.

| Table VII Performance (optimized parameters) comparison of FKPC and TFKPC on synthetic dataset D2 |
|---------------------------------|---------------------------------|---------------------------------|
| Index                          | FKPC Source domain | FKPC Target domain | TFKPC Target domain |
|                                | min | max | mean | std | min | max | mean | std | min | max | mean | std |
| NMI                            | 0.2246 | 0.7794 | 0.7239 | 0.3056 | 0.0727 | 0.6566 | 0.4367 | 0.2949 | 0.7320 | 0.7320 | 0.7320 | 0 |
| RI                             | 0.6436 | 0.9285 | 0.9000 | 0.1569 | 0.5698 | 0.8614 | 0.7506 | 0.1472 | 0.8898 | 0.8898 | 0.8898 | 0 |
| XB_{sec}                       | 3.1712 | 19.986 | 8.488 | 7.952 | 5.696 | 81.34 | 55.308 | 34.824 | 5.014 | 5.014 | 5.014 | 1.9e-07 |
Table VIII Performance index $\text{NMI}$ obtained by TFKPC on synthetic dataset D2 under different parameters

| $\lambda$ | 0       | 0.005   | 0.1     | 0.5     | 0.7     | 1       | 1.5     | 1.8     | 5       | 10      | 50      | 100     |
|-----------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| Min       | 0.0727  | 0.0748  | 0.4845  | 0.6040  | 0.5798  | 0.7320  | 0.6549  | 0.6073  | 0.6289  | 0.6289  | 0.6289  | 0.6289  |
| Max       | 0.6566  | 0.6240  | 0.6703  | 0.7052  | 0.7545  | 0.7320  | 0.6549  | 0.6073  | 0.6289  | 0.6289  | 0.6289  | 0.6289  |
| Mean      | 0.4367  | 0.3324  | 0.5525  | 0.6781  | 0.7280  | 0.7320  | 0.6549  | 0.6073  | 0.6289  | 0.6289  | 0.6289  | 0.6289  |
| Std       | 0.2849  | 0.2747  | 0.0937  | 0.0524  | 0.0941  | 0.135e-16| 0   | 0   | 0   | 0   | 0   | 0   |

Table IX Performance index $\text{RI}$ obtained by TFKPC on synthetic dataset D2 under different parameters

| $\lambda$ | 0       | 0.005   | 0.1     | 0.5     | 0.7     | 1       | 1.5     | 1.8     | 5       | 10      | 50      | 100     |
|-----------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| Min       | 0.5698  | 0.5712  | 0.7487  | 0.8354  | 0.8196  | 0.8898  | 0.8540  | 0.8407  | 0.8540  | 0.8540  | 0.8540  | 0.8540  |
| Max       | 0.6146  | 0.8470  | 0.8586  | 0.8887  | 0.9038  | 0.8898  | 0.8540  | 0.8407  | 0.8540  | 0.8540  | 0.8540  | 0.8540  |
| Mean      | 0.7506  | 0.6988  | 0.8057  | 0.8737  | 0.8898  | 0.8898  | 0.8540  | 0.8407  | 0.8540  | 0.8540  | 0.8540  | 0.8540  |
| Std       | 0.1472  | 0.1380  | 0.0549  | 0.0275  | 0.0451  | 0   | 0   | 0   | 0   | 0   | 0   | 0   |

Table X Performance index $X_{BFKPC}$ obtained by TFKPC on synthetic dataset D2 under different parameters

| $\lambda$ | 0       | 0.005   | 0.1     | 0.5     | 0.7     | 1       | 1.5     | 1.8     | 5       | 10      | 50      | 100     |
|-----------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| Min       | 5.69    | 3.86    | 7.02    | 5.26    | 4.77    | 5.014   | 5.507   | 5.56    | 5.46    | 5.34    | 5.26    |
| Max       | 81.34   | 26.41   | 52.90   | 16.93   | 19.04   | 5.014   | 5.507   | 5.56    | 5.46    | 5.34    | 5.26    |
| Mean      | 55.30   | 11.39   | 20.24   | 7.23    | 7.84    | 5.014   | 5.507   | 5.56    | 5.46    | 5.34    | 5.26    |
| Std       | 34.82   | 6.92    | 17.65   | 4.22    | 5.90    | 1.98e-07| 4.68e-08| 5.11e-08| 4.22e-08| 1.14e-07| 6.54e-08| 2.32e-07|

Fig. 6 Synthetic dataset D3 generated for evaluating the proposed TFSC algorithm: (a) data generated in the source domain; (b) data generated in the target domain

Table XI Parameters used to generate synthetic dataset D3 for evaluating the TFSC algorithm

| Important attribute set | Source domain | Target domain |
|-------------------------|---------------|---------------|
| Cluster-1               | [1:31]        | [1:32]        |
| Cluster-2               | [10:40]       | [25:40]       |
| Cluster-3               | [20:55]       | [35:55]       |
| size                    | 600           | 60            |

3) Evaluation of TFSC Algorithm

In this subsection, another synthetic dataset, denoted as D3, with predetermined cluster structures is used to evaluate the performance of the proposed TFSC algorithm. Data was generated for the source domain and the target domain, respectively, where each domain contains three clusters located at different subspace as shown in Fig 6 and the corresponding
parameters used to generate the data are listed in Table XI. Fig. 6(a) and Fig. 6(b) contain three sub-figures, each of which corresponds to a cluster where the high dimensional data in a cluster are taken as the sequences and plotted in the corresponding sub-figure with the sequence number of features and the value of features as the x-coordinate and y-coordinate, respectively. From these sub-figures, we can see the corresponding subspace of each cluster, in which the features are more important than other features to the associated cluster. For example, from three sub-figures in Fig. 6(a), we can see that most features of three clusters have the feature values uniformly distributed in the interval [0, 100], but for each cluster there is a subset of features whose values are distinctive from other clusters. As shown in Fig. 6(a), three clusters in the source domain have the distinctive feature subsets with the sequence number of features in the intervals of [1, 31], [10, 40] and [20, 55], respectively.

The performance of the TFSC algorithm is compared with that of the classical FSC algorithm with the results reported in Table XII. Here, the parameter analysis results like those in Tables III-V have been omitted to save space. Basically, the observations for TFSC are very similar to those for TFCM and TFKPC. Let’s summarize them as follows.

(1) Table XII shows that the parameter optimized clustering result of TFSC are obviously superior to that of the classical FSC algorithm. In particular, TFSC has obtained the mean values of both NMI and RI as 1 for the data in the target domain, which implies that all the data have been correctly clustered, while the clustering effect of the traditional FSC is inferior to that of TFSC obviously.

(2) Although omitted here, the parameter analysis results show that the introduction of knowledge of source domain has enhanced the stability/robustness of the proposed TFSC algorithm. Again, appropriate parameter values should be set in order to optimize the advantages of the proposed TFSC algorithm. For our experiments with synthetic dataset D3, parameter $\lambda_2$ should not be set too large, typically, $\lambda_2 \leq 0.1$. On the other hand, parameter $\lambda_1$ can be set within a wider range, e.g., $\lambda_1 \in [0.5, 5]$.

(3) As in TFCM, the parameter analysis results of TFSC show that the trends of the clustering results with the three indices are similar under different parameter settings, i.e., when the value of one index is better, the other two are typically better.
| Index | Source domain | Target domain | Target domain | Target domain | Target domain |
|-------|--------------|--------------|--------------|--------------|--------------|
|       | FSC          | TFSC         | FSC          | TFSC         | FSC          |
|       | min          | max          | mean         | std          | min          | max          | mean         | std          | min          | max          | mean         | std          |
| NMI   | 0.9943       | 0.9943       | 0.9943       | 0            | 0.5434       | 0.8163       | 0.7084       | 0.0990       | 1            | 1            | 1            | 1.17e-016    |
| RI    | 0.9999       | 0.9999       | 0.9999       | 0            | 0.7386       | 0.8961       | 0.8220       | 0.0553       | 1            | 1            | 1            | 0            |
| XB_FSC| 0.1298       | 0.1298       | 0.1298       | 0            | 2.3113       | 2.5413       | 2.3418       | 0.0702       | 0.4221       | 0.4221       | 0.4221       | 2.89e-05     |

In addition to the aforementioned experimental results, the subspace identification ability of TFSC is also compared with that of FSC and the results are reported in Table XIII with the identification rates computed. Obviously, the identification rate of FSC (35/69) has been improved to that of TFSC (52/69) by leveraging the knowledge of source domain. Specifically, Table XIII shows that TFSC has identified more truly important features than FSC in the corresponding embedded subspace for each cluster. For example, TFSC has detected 24 important features while FSC only identifies 9 important features for the 1\textsuperscript{st} cluster.

### Table XIII: Comparison of identification abilities of the traditional FSC and the proposed TFSC

| No. of cluster | True | Identified by FSC | Identified by TFSC |
|----------------|------|------------------|--------------------|
|                | True | Identified       | Identified         |
| 1              | 32   | features: 1:32    | features: 32       |
|                |      | with top weights: | with top weights:  |
|                |      | {37,51,36,26*20,38,179,43,25, | {12,16,27,13,98,63,156,8,5,144, |
|                |      | 191,42,141,104,183,12,41,23,11, | 162,6,183,3,106,160,1,23,30,24, |
|                |      | 39,144,16,130,146,28,33,31,101, | 7,22,10,4,11,21,15,31,20,17,14,2, |
|                |      | 143,46,17,100,126,} | } |
| 2              | 16   | features: 25,40   | features: 21       |
|                |      | with top weights: | with top weights:  |
|                |      | {37,51,36,26,20,38,179,43,25, | {37,33,40,39,29,36,35,28,148, |
|                |      | 191,42,141,104,183,12,41} | 155,132,175,63,115,142,74} |
| 3              | 21   | features: 35,55   | features: 21       |
|                |      | with top weights: | with top weights:  |
|                |      | {35,37,38,40,36,52,49,43,51,50, | {52,49,38,43,39,51,47,55,48,37,45, |
|                |      | 46,48,42,41,45,47,53,49,55,44} | 50,46,42,44,53,41,36,35,65} |

Identification Rate: (9+5+21)/69=35/69 (24+8+20)/69=52/69

* The bold number denotes the detected truly important features associated with the embedded subspaces.

### C. 20 Newsgroups Text Clustering

In this subsection, the proposed three TPFC algorithms are further experimented to evaluate their text clustering performance. The 20 newsgroups (20NG) text datasets [57] were adopted. In order to simulate the application scenarios studied in this paper, four categories of text data were adopted and each of them contains different sub-clusters in the source domain and target domain with the details shown in Table XIV. From Table XIV, we can see that the constructed datasets have the following distinctive characteristics.

1. The data in the source and target domains are related. The data in both domains belong to
four different clusters, i.e., *comp*, *rec*, *sci* and *talk*.

(2) Although the data in both domains are related, there exist obvious differences between them. The same cluster in different domains contains very different sub-clusters. For example, the source data of the *rec* cluster belong to three sub-clusters, i.e., *rec.autos*, *rec.motorcycles* and *rec.sport.baseball*, while the target data only belong to the sub-cluster *rec.sport.hockey*. Thus, the data distributions in both domains are related but different to some extent.

(3) The size of dataset in the target domain is much smaller than that in the source domain.

There is a practical requirement to improve the clustering effect on the data in the target domain by effectively using the knowledge of source domain.

For the adopted data, dimensionality reduction has been applied by using the BOW toolkit [58] to effectively preprocess the high dimensional data into the final data containing 800 effective features used for clustering. For the proposed algorithms, the parameter settings are similar to that used in subsection IV-B. The proposed three TPFC algorithms, i.e., TFCM, TFKPC and TFSC, and their non-transfer counterparts have been evaluated on the constructed text datasets. In our experiments, we found that the cluster direction prototype based algorithms, i.e., FKPC and TFKPC, failed to provide meaningful clustering results on this dataset due to the high dimensionality involved and thus their clustering results are omitted here.

In Table XV, the clustering results with the optimized parameters are reported, including the means and standard deviations of $NMI$, $RI$ and $XB/XB_{FSC}$ values after 20 runs of every algorithm with the optimal parameter setting. Meanwhile, the clustering results of the proposed TFSC algorithm under different parameter settings are reported in Tables XVI-XVIII for the performance index $NMI$, $RI$ and $XB_{FSC}$, respectively. To save space, the clustering results of the proposed TFCM algorithm under different parameter settings are omitted here since the trends of its clustering results are similar to that of TFSC. From the results, we can obtain the following observations.

(1) As shown in Table XV, in this high dimensional dataset, the two subspace clustering algorithms FSC and TFSC have shown obvious advantage to the two non-subspace clustering algorithms FCM and TFCM. Again, the experimental results follow those in
the previous experiments that the proposed TPFC algorithms TFCM and TFSC are superior to their non-transfer learning counterparts.

(3) Tables XVI-XVIII show that the introduction of knowledge of source domain has really improved the clustering effect of the proposed transfer clustering algorithms. Meanwhile, Tables XVI-XVIII show that appropriate parameter values should be set in order to optimize the advantages of the proposed TFSC algorithm.

(4) By comparing Tables XVI-XVIII for different performance indices, we can see that in most situations when $XB_{FSC}$ is better, the corresponding $NMI$ and $RI$ indices are also better. Since $XB_{FSC}$ can be directly calculated without requiring the prior label information of the data, it is a feasible strategy to take the $XB_{FSC}$ as a criterion to determine the approximate parameter values.

Table XIV 20 newsgroups text data used for evaluating the proposed TPFC algorithms

| Domain | Source domain | Target domain |
|--------|---------------|---------------|
| Cluster | Sub-cluster and the size | Sub-cluster and the size |
| comp | comp. graphics | comp. windows.x |
| | comp.os.ms | comp.sys. ibm.pc. |
| | -windows. | hardware |
| | misc | comp. sys.mac. |
| | hardware | |
| | | |
| | | |
| rec | rec. autos | rec. sport. hockey |
| | rec. motorcycles | - |
| | rec. sport. baseball | |
| | | |
| | | |
| sci | sci.crypt | sci.electronics |
| | | sci.med |
| | | - |
| | | sci.space |
| | | |
| talk | talk. politics. | talk. religion. |
| | uns | misc |
| | | |
| | | |
| | | |

Table XV Performance (optimized parameters) comparison between the classical clustering algorithms and the transfer prototype based clustering algorithms on 20NG text dataset

| Indices | FCM | T-FCM | FKPC | T-FKPC | FSC | TFSC |
|---------|-----|-------|------|-------|-----|------|
| NMI-mean | 0.0223 | 0.0554 | - | - | 0.4889 | 0.6182 |
| NMI-std | 0.0011 | 0.0011 | 0.0122 | 0.0010 |
| RI-mean | 0.3397 | 0.3909 | - | - | 0.7342 | 0.8650 |
| RI-std | 0.0029 | 0.0249 | 0.0273 |
| $XB_{FKPC}/XB_{FSC}$-mean | 31.4645 | 27.6234 | 2.54e-03 | 4.59e-04 |
| $XB_{FKPC}/XB_{FSC}$-std | 0.9494 | 0.6294 | 4.52e-04 | 6.00e-05 |
and for the LSSMTC algorithm, the regularization parameter 

 clustering algorithms, the parameter setting is the same as that described in subsection IV-B, on both synthetic and real-world datasets used in subsections IV-B & C. For the proposed CombKM and co-clustering (DRCC) [39] and a multi-task clustering algorithm LSSMTC [40]

| \( \lambda \) | 0.1380 | 0.1369 | 0.1361 | 0.1298 | 0.1322 | 0.1069 | 0.1055 | 0.1058 | 0.1040 | 0.1061 | 0.1064 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.4889 | 0.5084 | 0.5102 | 0.5403 | 0.6079 | 0.5265 | 0.5004 | 0.5310 | 0.4726 | 0.3945 | 0.5393 |
| 0.001 | 0.4952 | 0.5300 | 0.5551 | 0.5728 | 0.6057 | 0.5189 | 0.4705 | 0.5063 | 0.4638 | 0.4844 | 0.4267 |
| 0.01 | 0.5039 | 0.5511 | 0.5651 | 0.5914 | 0.6133 | 0.5393 | 0.4184 | 0.4634 | 0.4735 | 0.4641 | 0.4471 |
| 0.05 | 0.5015 | 0.5678 | 0.5512 | 0.6001 | 0.6182 | 0.5257 | 0.4144 | 0.4042 | 0.3900 | 0.3682 | 0.4033 |
| 0.1 | 0.4511 | 0.3805 | 0.3794 | 0.3792 | 0.3833 | 0.3155 | 0.3086 | 0.3137 | 0.2919 | 0.3081 | 0.3169 |
| 0.2 | 0.3614 | 0.3234 | 0.3203 | 0.2985 | 0.3381 | 0.2481 | 0.2452 | 0.2542 | 0.2508 | 0.2441 | 0.2567 |
| 0.3 | 0.2593 | 0.2599 | 0.2608 | 0.2551 | 0.2476 | 0.2070 | 0.2067 | 0.2106 | 0.2118 | 0.2066 | 0.2114 |
| 0.4 | 0.2202 | 0.2119 | 0.2163 | 0.2189 | 0.1969 | 0.1777 | 0.1873 | 0.1844 | 0.1803 | 0.1882 | 0.1702 |
| 0.5 | 0.1887 | 0.1898 | 0.1952 | 0.1823 | 0.1847 | 0.1602 | 0.1621 | 0.1684 | 0.1614 | 0.1699 | 0.1624 |
| 0.7 | 0.1696 | 0.1778 | 0.1673 | 0.1663 | 0.1683 | 0.1267 | 0.1304 | 0.1321 | 0.1319 | 0.1326 | 0.1338 |
| 1 | 0.1380 | 0.1369 | 0.1361 | 0.1298 | 0.1322 | 0.1069 | 0.1055 | 0.1058 | 0.1040 | 0.1061 | 0.1064 |

Table XVII Mean values of performance index \(RI\) obtained by TFSC on the 20NG text dataset under different parameter settings

| \( \lambda \) | 0.7342 | 0.7530 | 0.7253 | 0.7680 | 0.8016 | 0.7095 | 0.7392 | 0.7111 | 0.6935 | 0.6874 | 0.7655 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.001 | 0.7571 | 0.7971 | 0.7549 | 0.8319 | 0.8604 | 0.7735 | 0.6712 | 0.7380 | 0.7329 | 0.7013 | 0.7417 |
| 0.01 | 0.7563 | 0.8018 | 0.8046 | 0.8601 | 0.8641 | 0.7323 | 0.6832 | 0.7246 | 0.6748 | 0.6615 | 0.6782 |
| 0.05 | 0.7547 | 0.8152 | 0.8118 | 0.8637 | 0.8850 | 0.6991 | 0.7087 | 0.7145 | 0.6632 | 0.6787 | 0.6865 |
| 0.1 | 0.7027 | 0.6922 | 0.6916 | 0.6873 | 0.7036 | 0.6687 | 0.6638 | 0.6560 | 0.6702 | 0.6556 | 0.6609 |
| 0.2 | 0.6965 | 0.6788 | 0.6770 | 0.6696 | 0.6728 | 0.6702 | 0.6737 | 0.6594 | 0.6533 | 0.6498 | 0.6577 |
| 0.3 | 0.6696 | 0.6721 | 0.6816 | 0.6710 | 0.6665 | 0.6690 | 0.6619 | 0.6624 | 0.6588 | 0.6559 | 0.6550 |
| 0.4 | 0.6697 | 0.6698 | 0.6702 | 0.6647 | 0.6636 | 0.6619 | 0.6611 | 0.6615 | 0.6554 | 0.6568 | 0.6526 |
| 0.5 | 0.6636 | 0.6662 | 0.6709 | 0.6603 | 0.6593 | 0.6575 | 0.6577 | 0.6559 | 0.6560 | 0.6546 | 0.6536 |
| 0.7 | 0.6536 | 0.6534 | 0.6632 | 0.6512 | 0.6509 | 0.6498 | 0.6488 | 0.6489 | 0.6486 | 0.6491 | 0.6491 |
| 1 | 0.6450 | 0.6453 | 0.6529 | 0.6426 | 0.6418 | 0.6415 | 0.6409 | 0.6412 | 0.6404 | 0.6413 | 0.6411 |

D. Comparison with Related Clustering Algorithms

In this subsection, the proposed TPFC algorithms are finally compared with several related works, namely, two transfer clustering algorithms, i.e., self-taught clustering (STC) [16] and transfer spectral clustering (TSC) [17], two collaborative clustering algorithms CombKM and co-clustering (DRCC) [39] and a multi-task clustering algorithm LSSMTC [40] on both synthetic and real-world datasets used in subsections IV-B & C. For the proposed clustering algorithms, the parameter setting is the same as that described in subsection IV-B, and for the LSSMTC algorithm, the regularization parameter \( \lambda \in [0,1] \) has been tried with
the values in the set of \{0.01, 0.1, 0.3, 0.5, 0.7, 0.9, 1\} and for the co-clustering algorithm, the related regularization parameters, \( \lambda \) and \( \mu \), have been set as \( \lambda = \mu \) and tried in the set of \{0.01, 0.1, 0.3, 0.5, 0.7, 0.9, 1\}. The parameter setting about TSC and STC is referred to [17]. Especially, for TSC algorithm it is required that the dimensional number of data is not smaller than the number of clusters. Thus it is not applicable to two low dimensional datasets, i.e., D1 and D2.

For all the algorithms, the results obtained with the optimal parameter setting are reported in term of the means of 20 runs. The experimental results are reported in Table XIX based upon which the following observations can be made.

1. The proposed TFCM and TFSC algorithms have shown highly competitive performance with respect to the five related algorithms which have used the knowledge or data in the other tasks. In particular, the TFSC algorithm has shown better performance than the other algorithms in most of the adopted datasets.

2. Although the proposed TFKPC algorithm is inferior to the other non-cluster direction prototype based algorithms in most of the datasets, it shows an obvious advantage on the D2 dataset, which is a direction oriented dataset. Thus, TFKPC is appropriate for the datasets of this type.

3. Among the three proposed transfer fuzzy clustering algorithms, TFSC is obviously superior to the other two, particularly in the high dimensional datasets, such as D3 and 20NG datasets.

### Table XIX Performance comparison of the best results obtained by the proposed TPFC algorithms and the related algorithms on different synthetic and real-world datasets by using the NMI and RI indices

| Dataset | Indices | LSSMTC | CombKM | DRCC | STC | TSC | TFCM | TFKPC | TFSC |
|---------|---------|--------|--------|------|-----|-----|------|-------|------|
| D1      | NMI-mean| 0.7469 | 0.7584 | 0.7308| 0.7501| -   | 0.8084| 0.1983| 0.8336|
|         | NMI-std | 0      | 1.1e-016| 0    | 0.0574| -   | 0    | 2.92e-017| 0    |
|         | RI-mean | 0.8788 | 0.8959 | 0.8668| 0.8924| -   | 0.9117| 0.6203| 0.9319|
|         | RI-std  | 0      | 7.58e-004| 0    | 0.0580| -   | 0    | 1.7e-016 | 0    |
| D2      | NMI-mean| 0.2579 | 0.0346 | 0.0328| 0.3208| -   | 0.3957| 0.7320| 0.5329|
|         | NMI-std | 0      | 0.0189 | 0.0061| 0     | -   | 0    | 0     | 0.0016|
|         | RI-mean | 0.6028 | 0.5549 | 0.5550| 0.6487| -   | 0.6818| 0.8898| 0.7246|
|         | RI-std  | 1.17e-016| 0.0034| 0.0016| 0    | -   | 0    | 0     | 0.0011|
| D3      | NMI-mean| 0.5266 | 0.0561 | 0.4530| 0.9382| 0.9875| 0.5536| 0.0366| 1     |
|         | NMI-std | 1.17e-016| 7.31e-018| 5.85e-017| 0.0207| 0.0023| 7.31e-018| 0 | 1.7e-016|
|         | RI-mean | 0.7463 | 0.4542 | 0.7056| 0.9853| 0.9927| 0.7791| 0.5096| 1     |
|         | RI-std  | 2.34e-016| 5.85e-017| 1.17e-016| 0.0112| 1.73e-03| 1.17e-016| 0 | 0     |
| NG-20   | NMI-mean| 0.0598 | 0.0123 | 0.0289| 0.5681| 0.5384| 0.0554| - | 0.6182|
|         | NMI-std | 0      | 0      | 3.65e-018| 0 | 1.36e-16| 0 | -  | 0.0010|
|         | RI-mean | 0.2823 | 0.2519 | 0.2534| 0.8331| 0.8297| 0.3909| -  | 0.8650|
|         | RI-std  | 5.85e-017| 2.36e-017| 5.85e-017| 0 | 0 | 0 | - | 0.0273|
In addition, the performance index $XB$ and modified $XB$ obtained by different fuzzy measure based algorithms are compared in Table XX. From the results, we can see that the proposed transfer fuzzy clustering algorithms have obtained improved clustering performance compared with the corresponding no-transfer counterparts in term of the $XB$ and modified $XB$ indices.

**Table XX** Performance comparison of the best results of different fuzzy measure based algorithms by using the $XB$ and the modified $XB$ indices

| Dataset | $XB_{FCM}$ | $XB_{TFCM}$ | $XB_{FKPC}$ | $XB_{TFKPC}$ | $XB_{FSC}$ | $XB_{TFSC}$ |
|---------|------------|-------------|-------------|--------------|------------|-------------|
| D1      | 0.1560     | 0.1291      | 30.2447     | 12.5926      | 3.1623     | 2.2190      |
|         | 2.12e-03   | 3.70e-17    | 22.2008     | 4.32e-11     | 0.6852     | 0.3285      |
| D2      | 0.1920     | 0.1801      | 55.308      | 5.014        | 2.9259     | 2.6701      |
|         | 3.39e-07   | 0           | 34.824      | 1.9e-07      | 0.6890     | 0.8061      |
| D3      | 2.04e+06   | 1.93e+05    | 0.0086      | 0.0049       | 2.3418     | 0.4221      |
|         | 6.47e+06   | 5.10e+05    | 0.0039      | 0.0018       | 0.0702     | 2.89e-05    |
| NG-20   | 31.4645    | 27.6234     | -           | -            | 2.54e-03   | 4.59e-04    |
|         | 0.9494     | 0.6294      | -           | -            | 4.52e-04   | 6.00e-05    |

Finally, the running time of different algorithms on several datasets has been reported in Table XXI. From these results, we can see that the speed of the proposed transfer clustering algorithms is similar to that of the corresponding non-transfer counterparts, which also validates the theoretical analyses in section III-C.

**Table XXI** Comparison of different algorithms on running time (seconds)

| Dataset | FCM  | TFCM | FKPC | TFKPC | FSC  | TFSC |
|---------|------|------|------|-------|------|------|
| D1      | 0.0179 | 0.0173 | 0.2117 | 0.2176 | 0.0135 | 0.0125 |
|         | 0.0039 | 0.0014 | 0.0666 | 0.0683 | 0.0062 | 0.0087 |
| D2      | 0.0403 | 0.0448 | 0.2543 | 0.2588 | 0.0158 | 0.0121 |
|         | 0.0064 | 0.0111 | 0.1063 | 0.1028 | 0.0031 | 0.0059 |
| D3      | 0.0465 | 0.0480 | 4.0536 | 4.1325 | 0.0156 | 0.0192 |
|         | 0.0128 | 0.0076 | 0.9942 | 1.0517 | 0.0033 | 0.0072 |
| NG-20   | 27.5435 | 27.3509 | -     | -     | 16.340 | 18.478 |
|         | 0.49433 | 0.3684 | -     | -     | 3.8050 | 5.0250 |

**E. Discussion**

In this subsection, let us further discuss three issues. The first one is about the relations/limitations between the two dataset sizes (source/target) and the performance, which can be analyzed from three aspects. First, in this study, an assumption is given that the data in target domain is scarce such that transfer learning is required to improve the clustering performance in the target domain. Therefore, when the size of the data in the target domain is
small, the proposed transfer clustering algorithms may obtain improved clustering results when compared with the non-transfer counterparts. Second, when the size of the dataset in target domain is much larger, the information is usually sufficient for obtaining effective clustering by the traditional non-transfer clustering algorithms. In this case, even if transfer clustering algorithms are adopted, its advantage will not be obvious. Third, in the proposed transfer clustering algorithms, the original raw data in the source domain is not used and only some knowledge, such as cluster centers, are adopted in the clustering procedure in the target domain. Thus, the reliability of the knowledge obtained in the source domain has an important influence on the clustering performance of the proposed transfer clustering algorithms. If the size of data in the source domain is small, the induced knowledge may not be reliable. When the size of data in the source domain is much larger, more reliable knowledge can be expected from the source domain. With such an analysis, we can infer that the larger the size of dataset in the source domain is, the more reliable the knowledge obtained in the source domain and the more effective the proposed transfer clustering algorithms will be in the target domain.

The second issue is about the influence of the balance parameters in the proposed objective functions, such as parameter $\lambda$ for TFKPC. As shown by the experimental results, the balance parameter values are set at about 0.5 or larger to get good results for most synthetic datasets, but for the real-world datasets they are set at about 0.1 or less for good performance. With respect to this observation, we give the following explanations. First, the balance parameters control the influence of knowledge in the source domain to the target domain. It is expected that if the knowledge induced in the source is more useful to the target domain, the balance parameter should be set with a much larger value. For the synthetic datasets, they have been designed specifically in accordance with the scene where transfer learning is really required. Thus, the knowledge obtained in the source is more useful to the target domain and the balance parameters can be set with much larger value, such as 0.5, to get a good performance. However, for the applications to real-world datasets, the balance parameter has been set with much smaller values. It may be due to the fact that real-world datasets are very complicated and the amount of knowledge to be transferred is not known. It is better to take a more conservative approach to set the balance parameter’s value. The
experimental results demonstrate that even if only a smaller value is used, knowledge leveraging is still important and effective and the performance of the proposed algorithms has been obviously improved when compared with the non-transfer counterparts.

The third issue is how to determine the appropriate values for the related parameters. In the proposed algorithms, it is necessary to set the balance parameters, i.e., $\lambda_1$ and $\lambda_2$ or $\lambda$, appropriately for good performance. Unlike supervised learning, no supervised information can be used by parameter selection strategies, such as the commonly used cross validation strategy. For most unsupervised learning algorithms, it is still an open problem to determine the optimal values for the parameters involved. In our experiments, we can see that with a pretty wide range of parameter values, the proposed transfer clustering algorithms can obtain improved performance compared with the non-transfer counterparts. In fact, the optimal parameters are usually dependent on applications. Thus, for a specific application, users can determine an appropriate range for the involved parameters of the adopted algorithm by the prior knowledge or the available validation dataset. In particular, a promising strategy is to determine the appropriate values for the involved parameters in the proposed fuzzy measure based clustering algorithms by using some evaluation indices, such as the $XB$ and modified $XB$ indices. The conclusion here is obtained based on the facts that (i) the $XB$ index and modified $XB$ indices, unlike the $RI$ and $NMI$ indices, can be computed without the supervised label information; (ii) in most situations when the $XB$ index or the modified $XB$ index is better in a certain parameter setting, the obtained $RI$ and $NMI$ indices are usually much better accordingly. In addition, in order to weaken the influence of the parameters, some other investigations can also be done as suggested in [59, 60]. For example, the ensemble learning technique seems to be a promising one. By integrating the clustering results obtained under different parameter settings, ensemble clustering will provide comparable result with the case under the optimal parameter setting [59, 60].

V. Conclusions

In this study, the concept of knowledge leveraging is used to develop a set of transfer prototype based fuzzy clustering methods for the application scenarios where the data is limited or insufficiently available for effective clustering. Based on the introduced
knowledge-leverage transfer learning mechanism, several transfer prototype based clustering algorithms are exploited to address the deficiency caused by the scarceness of data in the target domain. The proposed prototype based clustering algorithms learn from not only the data of the target domain but also the knowledge of the source domain. The experimental results have demonstrated the attractiveness and effectiveness of the proposed transfer prototype based fuzzy clustering algorithms when compared with the existing classical prototype based clustering algorithms without the transfer learning abilities, and other related algorithms such as the multi-task clustering algorithm.

In future, more works can be addressed about the proposed transfer clustering technique. As mentioned in subsection III-A-2, our work assumes that the number of clusters in the target domain is the same as that in the source domain. This assumption facilitates simplicity in the development of transfer prototype based clustering algorithms. However, it also prohibits the proposed algorithms from some applications, i.e., the cases where the number of clusters may be different between the source domain and the target domain. For such a challenge, the following problems should be well considered to develop the extended algorithms. First, an effective method should be investigated to check whether the cluster numbers between the target domain and the source domain are the same. Second, when we know that the numbers of clusters in the target and source domains are different, how to effectively transfer useful information from the source to the target domain has to be addressed. This is a challenging problem for the development of transfer prototype based clustering algorithms. Our future work on transfer clustering will also include works for non-prototype based clustering methods.

**Acknowledgements**

This work was supported in part by the Hong Kong Polytechnic University under Grant G-UA68, National Natural Science Foundation of China under Grants 61170122, 2012NSFC, Natural Science Foundation of Jiangsu Province under Grant 2012JSNSFC and JiangSu 333 Expert Engineering Grant (BRA2011142).
References

[1] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Trans. Knowledge Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[2] J.W. Tao, F.L. Chung, and S.T. Wang, “On minimum distribution discrepancy support vector machine for domain adaptation,” *Pattern Recognition*, vol. 45, no. 11, pp. 3962-3984, 2012.

[3] Z. Sun, Y.Q. Chen, J. Qi, and J.F. Liu, “Adaptive localization through transfer learning in indoor Wi-Fi environment,” *Proc. 7th International Conference on Machine Learning and Applications*, pp. 331–336, 2008.

[4] S. Bickel, M. Brückner, and T. Scheffer, “Discriminative learning for differing training and test distributions,” *Proc. 24th Int. Conf. Machine Learning*, pp. 81-88, 2007.

[5] N. D. Lawrence and J. C. Platt, “Learning to learn with the informative vector machine,” *Proc. 21st Int. Conf. Machine Learning*, 2004.

[6] J. Gao, W. Fan, J. Jiang, and J. Han, “Knowledge transfer via multiple model local structure mapping,” *Proc. 14th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pp. 283-291, 2008.

[7] L. Mihalkova and R. J. Mooney, “Transfer learning by mapping with minimal target data,” *Proc. Assoc. for the Advancement of Artificial Intelligence (AAAI ’08) Workshop Transfer Learning for Complex Tasks*, 2008.

[8] J. Davis and P. Domingos, “Deep transfer via second-order Markov logic,” *Proc. Assoc. for the Advancement of Artificial Intelligence (AAAI ’08) Workshop Transfer Learning for Complex Tasks*, 2008.

[9] S.J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, “Domain adaptation via transfer component analysis,” *IEEE Trans. Neural Networks*, vol. 22, no.2, pp.199-210, 2011.

[10] Z. Wang, Y. Song, and C. Zhang, “Transferred dimensionality reduction,” *Proc. European Conf. Machine Learning and Knowledge Discovery in Databases (ECML/PKDD ’08)*, pp. 550-565, 2008.

[11] P. Yang, Q. Tan, and Y. Ding, “Bayesian task-level transfer learning for non-linear regression,” *Proc. Int. Conf. on Computer Science and Software Engineering*, pp. 62-65, 2008.

[12] L. Borzemski and G. Starczewski, “Application of transfer regression to TCP throughput prediction,” *Proc. First Asian Conference on Intelligent Information and Database Systems*, pp.
J. Liu, Y. Chen, and Y. Zhang, “Transfer regression model for indoor 3D location estimation,” *Lecture Note on Computer Science 5916*, pp. 603–613, 2010.

Z.H. Deng, Y.Z. Jiang, K.S. Choi, F.L. Chung, and S.T. Wang, “Knowledge-Leverage based TSK fuzzy system modeling,” *IEEE Trans. Neural Networks and Learning Systems*, vol. 24, no. 8, pp. 1200-1212, 2013.

Z.H. Deng, Y.Z. Jiang, F.L. Chung, H. Ishibuchi, and S.T. Wang, “Knowledge-Leverage based fuzzy system and its modeling,” *IEEE Trans. Fuzzy systems*, vol. 21, no. 4, pp. 597-609, 2013.

W. Dai, Q. Yang, G. Xue, and Y. Yu, “Self-taught clustering,” *Proc. 25th Int. Conf. Machine Learning*, pp. 200-207, 2008.

W.H. Jiang and F.L. Chung, “Transfer spectral clustering,” *In Machine Learning and Knowledge Discovery in Databases*, pp. 789-803, 2012.

J. Dunn, “A fuzzy relative of the ISODATA process and its use in detecting compact well separated clusters,” *Journal of Cybernetics*, vol. 3, no. 3, pp.32–57, 1973.

J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York Plenum, 1981.

R. Krishnapuram and J. M. Keller, “A possibilistic approach to clustering.” *IEEE Trans. on Fuzzy Systems*, vol.1, no. 2, pp. 98-110, 1993.

R.P. Li and M.A. Mukaidono, “A maximum entropy approach to fuzzy clustering,” *Proc on IEEE Int Conf Fuzzy Syst*. Yokohama, Japan, pp. 2227-2232, 1995.

J.Z. Huang, M.K. Ng, H. Rong, and Z. Li, “Automated variable weighting in k-means type clustering,” *IEEE Trans. Pattern Anal. Mach. Intell*. 27 (5) (2005) 657–668.

P. S. Bradley, and Q. L. Mangasarian, “K-plane clustering,” *Journal of Global Optimization*, vol. 16, no. 1, pp. 23-32, 2000.

M. Ester, H.P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” *Proc. 2nd International Conference on KDD*, pp. 226-231, 1996.

J. Sander, M. Ester, H.P. Kriegel, and X. Xu, “Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications,” *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 169-194, 1998.
[26] A. Hinneburg and D.A. Keim, “An efficient approach to clustering in large multimedia databases with noise,” Proc. 2nd International Conference on KDD, pp. 58–65, 1998.

[27] M. Ankerst, M.M. Breunig, H.P. Kriegel, and J. Sander, “OPTICS: Ordering points to identify the clustering structure,” Proc. International Conference on Management of Data, pp. 49–60, 1999.

[28] T. Pei, A. Jasra, D.J. Hand, A.X. Zhu, and C. Zhou, “DECODE: a new method for discovering clusters of different densities in spatial data,” Data Mining and Knowledge Discovery, vol. 18, no. 3, pp. 337–369, 2009.

[29] J. Shi and J. Malik, “Normalized cuts and image segmentation,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pp. 888–905, 2000.

[30] E. Hartuv and R. Shamir, “A clustering algorithm based on graph connectivity,” Information Processing Letter, vol. 76, no. 4, pp. 175–181, 2000.

[31] A.Y. Ng, M.I. Jordan, and Y. Weiss, “On spectral clustering: Analysis and an algorithm,” Advances in Neural Information Processing Systems, vol. 2, pp. 849-856, 2002.

[32] P.J. Qian, F.L. Chung, S.T. Wang, and Z.H. Deng, “Fast graph-based relaxed clustering for large data sets using minimal enclosing ball,” IEEE Trans. Systems, Man, and Cybernetics, Part B-Cybernetics, vol. 42, no. 3, pp. 672-687, 2012.

[33] A. Ben-Hur, D. Horn, H. Siegelmann, and V. Vapnik, “Support vector clustering,” Journal of Machine Learning Research, vol. 2, pp. 125–137, 2001.

[34] L. Tseng and S. Yang, “A genetic approach to the automatic clustering problem,” Pattern Recognition, vol. 34, no. 2, pp. 415–424, 2001.

[35] T. Kohonen, “The self-organizing map,” Proc. IEEE, vol. 78, no. 9, pp.1464–1480, 1990.

[36] Q.Q. Gu, and J. Zhou, “Co-clustering on manifolds,” Proc. 15th International Conference on KDD, pp. 359-368, 2009.

[37] W. Pedrycz, “Collaborative fuzzy clustering,” Pattern Recognition Letters, vol. 23, no. 14, pp. 1675-1686, 2002.

[38] W. Pedrycz, and P. Rai, “Collaborative clustering with the use of Fuzzy C-Means and its quantification,” Fuzzy Sets and Systems, vol. 159, no. 18, pp. 2399-2427, 2008.

[39] Q.Q. Gu and J. Zhou, “Learning the shared subspace for multi-task clustering and transductive transfer classification,” Proc. 9th IEEE International Conference on Data Mining, pp. 159-168,
2009.

[40] Z.H. Zhang, and J. Zhou, “Multi-task clustering via domain adaptation,” Pattern Recognition, vol. 45, no. 1, pp. 465-473, 2012.

[41] M. Bilenko, S. Basu, and R.J. Mooney, “Integrating constraints and metric learning in semi-supervised clustering,” Proc. 21th International Conference on Machine learning, pp.81-88, 2004.

[42] V. Loia, W. Pedrycz, and S. Senatore, “Semantic web content analysis: A study in proximity-based collaborative clustering,” IEEE Trans. Fuzzy Systems, vol. 15, no. 6, pp. 1294–1312, 2007.

[43] W. Pedrycz, “Fuzzy clustering with a knowledge-based guidance,” Pattern Recognition Letters, vol. 25, no. 4, pp. 469–480, 2004.

[44] W. Pedrycz, V. Loia, and S. Senatore, “P-FCM: A proximity based fuzzy clustering,” Fuzzy Sets and Systems, vol. 148, no. 1, pp. 21–41, 2004.

[45] F.L. Chung, S.T. Wang, Z.H. Deng, S.Chen, and D.W. Hu, “Clustering analysis of gene expression data based on semi-supervised visual clustering algorithm,” Soft Comput. vol. 10, no. 11, pp. 981-993, 2006.

[46] M. Tabatabaei-Pour, K. Salahshoor, and B. Moshiri. “A modified k-plane clustering algorithm for identification of hybrid systems,” Proc. 6th World Congress on Intelligent Control and Automation, pp. 1333-1337, 2006.

[47] Y. Wang, S.C Chen, D.Q. Zhang, and X.B. Yang, “Fuzzy k-Plane clustering algorithm,” Pattern Recognition and Artificial Intelligence, vol. 20, no. 5, pp. 704-710, 2007.

[48] J. Liu and T.D. Pham, “A spatially constrained fuzzy hyper-prototype clustering algorithm,” Pattern Recognition, vol. 45, no. 4, pp. 1759-1771, 2012.

[49] A Banerjee, I.S. Dhillon, J. Ghosh, et al, “Generative model based clustering of directional data,” Conference on Knowledge Discovery in Data, Washington, DC, 2003.

[50] I.S. Dhillon and D.S.Modha1, “Concept decompositions for large sparse text data using clustering,” Machine Learning, vol. 42 no.1, pp.143-175, 2001.

[51] G.J. Gan and J.H. Wu, “A convergence theorem for the fuzzy subspace clustering (FSC) algorithm,” Pattern Recognition, vol. 41, no. 6, pp. 1939–1947, 2008.

[52] L.P. Jing, M.K. Ng, and Z.X. Huang, “An entropy weighting k-means algorithm for subspace
clustering of high-dimensional sparse data,” *IEEE Trans. Knowledge and Data Engineering*, vol. 19, no. 8, pp. 1026–1041, 2007.

[53] Z.H. Deng, K.S. Choi, F.L. Chung, and S.T. Wang, “Enhanced soft subspace clustering integrating within-cluster and between-cluster information,” *Pattern Recognition*, vol. 43, no. 3, pp. 767-781, 2010.

[54] R. Hathaway, J. Bezdek, and W. Tucker, “An improved convergence theorem for the fuzzy c-means clustering algorithms,” in: J. Bezdek (Ed.), *Analysis of Fuzzy Information*, vol. III, CRC Press, Boca Raton, pp. 123–131, 1987.

[55] W. Zangwill, *Non-linear Programming: A Unified Approach*, Prentice-Hall, Englewood Cliffs, NJ, 1969.

[56] J. Liu, J. Mohammed, J. Carter, et al., “Distance-based clustering of CGH data,” *Bioinformatics*, vol. 22, no. 16, pp. 1971–1978, 2006.

[57] W.Y. Dai, G.R. Xue, Q. Yang, and Y. Yu, “Co-clustering based classification for out-of-domain documents,” *Proc. 13th International Conference on KDD*, pp. 210-219, 2007.

[58] A. K. McCallum, “Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering,” http://www-2.cs.cmu.edu/~mccallum/bow/.

[59] C. Domeniconi and M. Al-Razgan, “Weighted cluster ensembles: Methods and analysis,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 2, no. 4, pp.1-40, 2009.

[60] G. Francesco, C. Domeniconi, and A. Tagarelli, “Projective clustering ensembles,” *Data Mining and Knowledge Discovery*, vol. 26, no. 3, pp. 452-511, 2013.