Published in final edited form as:

IEEE Trans Pattern Anal Mach Intell. 2023 July ; 45(7): 9149–9168. doi:10.1109/TPAMI.2023.3237667.

Transforming Complex Problems into K-means Solutions

Hongfu Liu [Member, IEEE],
Department of Computer Science, Brandeis University, Waltham

Junxiang Chen [Member, IEEE],
Department of Biomedical Informatics, University of Pittsburgh

Jennifer Dy [Member, IEEE],
Department of Electrical & Computer Engineering, Northeastern University, Boston

Yun Fu [Fellow, IEEE]
Department of Electrical & Computer Engineering, Northeastern University, Boston

Abstract

K-means is a fundamental clustering algorithm widely used in both academic and industrial applications. Its popularity can be attributed to its simplicity and efficiency. Studies show the equivalence of K-means to principal component analysis, non-negative matrix factorization, and spectral clustering. However, these studies focus on standard K-means with squared Euclidean distance. In this review paper, we unify the available approaches in generalizing K-means to solve challenging and complex problems. We show that these generalizations can be seen from four aspects: data representation, distance measure, label assignment, and centroid updating. As concrete applications of transforming problems into modified K-means formulation, we review the following applications: iterative subspace projection and clustering, consensus clustering, constrained clustering, domain adaptation, and outlier detection.

Keywords

K-means; Consensus Clustering; Constrained Clustering; Domain Adaptation; Outlier Detection

1 Introduction

K-MEANS clustering is one of the most popular clustering algorithms [111]. It aims to identify K real or artificial points as the centroids to represent the data, where each sample in the space is assigned to its nearest centroid to achieve the clustering task. K-means clustering is recognized as one of the most favorable clustering tools with several merits, such as simplicity and efficiency. Based on this, some variants are proposed, including K-means++ [6], K-means−− [27], NEO-K-means [163], etc. Beyond the practical value, tremendous efforts have been made to explore the theoretical property of K-means in terms of convergence rate [18], initialization [6], and generalization [102].

hongfuliu@brandeis.edu.

Studies have shown that K-means is equivalent to principal component analysis (PCA) [40], non-negative matrix factorization (NMF) [41], and spectral clustering [37], providing a more straightforward alternative solution to these problems. However, previous studies predominately focus on standard K-means with squared Euclidean distance. In this review paper, we unify the available approaches in generalizing K-means in solving complex problems, especially non-standard cluster analysis problems. Specifically, we review the available literature on how generalized K-means can solve the following six complex problems:

1. Iterative Subspace Projection and Clustering. We review DisKmeans [179], an algorithm for simultaneous linear discriminant analysis subspace selection and clustering, which is equivalent to kernel K-means with a specific kernel Gram matrix.

2. Consensus Clustering. We review the K-means-based consensus clustering utility function and link it to flexible divergences [168], [169], where K-means can efficiently solve a rich family of utility functions of consensus clustering on a binary matrix.

3. Spectral Ensemble Clustering. We review spectral ensemble clustering [96], [102] that can be solved via weighted K-means clustering. These methods dramatically decrease the time and space complexities from $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$, respectively, to $\mathcal{O}(n)$ for both.

4. Partition Level Constrained Clustering. Inspired by the utility function that measures partition level similarity, a partition level constraint is employed for constrained clustering [94], [101], where they modify K-means by concatenating the feature matrix with side information and auxiliary zeros that do not contribute to centroid updating.

5. Structure-Preserved Unsupervised Domain Adaptation. We review some methods that achieve unsupervised domain adaptation using a K-means framework [99], [97]. After the source and target domain data are aligned in a shared space, a constrained K-means is employed to label the target data.

6. Joint Clustering and Outlier Detection. We review clustering with outlier removal, a joint clustering and outlier detection algorithm [95], where, via several basic partitions, the original feature space is transformed into partition space, and Holoentropy is employed to enhance the compactness of each cluster with outliers removed. This method introduces an auxiliary binary matrix to ensure the problem is solved by K-means−− [27].

In the literature, several surveys have been conducted on K-means from different aspects, including algorithm variants [79], [167], cluster number [80], [129], feature weighting [35], initialization [2], parallel computing [72], [49], theoretical analysis [17], and applications [3], [112]. In contrast to the above existing surveys, we focus on solving complex, especially non-standard cluster analysis problems, with K-means solutions. Specifically, we discuss how to generalize K-means regarding data input, distance, label assignment, and centroid updating. Subsequently, we present a general framework for converting a range of

problem domains into modified K-means formulations. None of these complex problems can be considered traditional clustering; however, with re-formulation, several complex problems can be elegantly solved by a simple (modified) K-means algorithm with theoretical guarantees. Beyond the aforementioned six problems, our framework provides a general direction to simplify other complex problems, such as consensus-guided feature selection [98], saliency-guided image co-segmentation [150], and knowledge-reused outlier detection [181].

Algorithm 1 Lloyd's K-means

1: Select K points as initial centroids;
2: **repeat**
3: Assign each point to its nearest centroid;
4: Recompute the centroid of each cluster;
5: **until** The centroids do not change.

The remaineder of this paper is organized as follows. In Section 2, we present preliminary knowledge of K-means clustering in terms of the objective function, algorithms, and properties. Section 3 describes how K-means can be generalized. In Section 4, we discuss K-means solutions for iterative subspace projection and clustering, consensus clustering, constrained clustering, domain adaptation, and outlier detection. In Section 5, we present experimental results that demonstrate these solutions are both effective and efficient. Finally, we conclude the paper in Section 6.

2 Preliminaries on K-means

K-means algorithm [111] is widely used to solve clustering problems. It separates samples into groups (clusters), such that samples in the same group are similar; while samples from different groups differ. In this section, we present some preliminaries on K-means clustering, including the objective function, optimization algorithms, and the advantages and disadvantages of K-means.

We use some conventional mathematical notations as follows. $\mathbf{R}$, $\mathbf{R}_+$, $\mathbf{R}_{++}$, $\mathbf{R}^d$ and $\mathbf{R}^{n\times d}$ are used to denote the sets of reals, non-negative reals, positive reals, d-dimensional real vectors, and $n\times d$ real matrices, respectively. For a d-dimensional real row vector $\mathbf{x}$, x_j denotes the j-th element of the vector $\mathbf{x}$, $\|\mathbf{x}\|_p$ denotes the L_p norm of $\mathbf{x}$, and $\mathbf{x}^\intercal$ denotes the transpose of $\mathbf{x}$. For a general matrix $\mathbf{X}$, $\mathbf{x}_i$ denotes the i-th row vector of $\mathbf{X}$ and x_{ij} denotes the element at the i-th row and j-th column of $\mathbf{X}$. The gradient of a single variable function f is denoted as ∇f, and the logarithm to the base 2 is denoted as log.

2.1 K-means Formulation

We first present the standard K-means objective function. Let $\mathbf{X}$ denote the $n\times d$ data matrix with n instances and d features, where $\mathbf{x}_l$ is a $1\times d$ row vector to present the l-th data point in $\mathbf{X}$. The objective function for K-means is given as folows:

$$\min_{\mathscr{C}_k, \mathbf{m}_k} \sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} \| \mathbf{x}_l - \mathbf{m}_k \|_2^2, \tag{1}$$

where $\| \cdot \|_2^2$ is the squared Euclidean distance, $\mathscr{C}_1, \cdots, \mathscr{C}_K$ are K disjoint clusters, with $\mathscr{C}_k \cap \mathscr{C}_{k'} = \emptyset$, $\forall k \neq k'$, $\cup_{k=1}^{K} \mathscr{C}_k$ covering all the samples, and $\mathbf{m}_k$ is a $1 \times d$ centroid row vector of $\mathscr{C}_k$. In standard K-means, the centroid vector is calculated by the arithmetic mean of the data points in one cluster, *i.e.,* $\mathbf{m}_k = \sum_{\mathbf{x}_l \in \mathscr{C}_k} \mathbf{x}_l / \mid \mathscr{C}_k \mid$, and each data point is assigned to the nearest centroid with the least squared Euclidean distance. When each data point only belongs to one cluster, this is called a crisp or hard partition [155]. The objective function in Eq. (1) minimizes the within-cluster sum of squared errors between each data point and its nearest centroid, which is equivalent to minimizing the within-cluster variance.

Algorithm 2 Hartigan's K-means

1: Initialize K centroids and label for each point;

2: **repeat**

3: For each point, find a new centroid via mostly decreasing Eq. (1) after label switching;

4: Recompute the old and new centroids by this point;

5: **until** The centroids do not change.

K-means also indirectly evaluates the separation of clusters due to the following relationship:

$$\sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} \| \mathbf{x}_l - \mathbf{m}_k \|_2^2 + \sum_{k=1}^{K} \mid \mathscr{C}_k \mid \cdot \| \mathbf{m}_k - \mathbf{m} \|_2^2 = \sum_{l}^{n} \| \mathbf{x}_l - \mathbf{m} \|_2^2, \tag{2}$$

where $\mathbf{m} = \sum_l^n \mathbf{x}_l / n$ is the $1 \times d$ centroid row vector of the whole data matrix and $\mid \mathscr{C}_k \mid$ denotes the number of points in cluster $\mathscr{C}_k$. As the right-hand side of Eq. (2) is a constant, we have

$$\min_{\mathscr{C}_k, \mathbf{m}_k} \sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} \| \mathbf{x}_l - \mathbf{m}_k \|_2^2 \Leftrightarrow \max_{\mathscr{C}_k, \mathbf{m}_k} \sum_{k=1}^{K} \mid \mathscr{C}_k \mid \cdot \| \mathbf{m}_k - \mathbf{m} \|_2^2, \tag{3}$$

This indicates that minimizing the within-cluster sum of squared error is equivalent to maximizing the separation of clusters.

Some variants and extensions of K-means include fuzzy C-means [16], where each data point has a fuzzy degree of belonging to each cluster, K-medians [71] which uses the median in each dimension instead of the mean, K-medoids [77] which uses the medoid instead of the mean, X-means [125] which automatically determines the cluster number, and G-means [58] which repeatedly splits clusters to build a hierarchy.

2.2 Objective Function in Matrix Form

The objective function in Eq. (1) can be rewritten in a matrix-wise formulation as follows:

$$\min_{\mathbf{H},\mathbf{M}} \|\mathbf{X}-\mathbf{H}\mathbf{M}\|_{\mathrm{F}}^2, \ \ \mathrm{s.t.} \sum_{k}^{K} \mathbf{H}_{lk} = 1, \mathbf{H}_{lk} \in \{0,1\}, \tag{4}$$

where $\mathbf{H}$ is an $n \times K$ binary indicator matrix. $\mathbf{H}_{lk} = 1$ represents the l-th instance belongs to the k-th cluster, $1 \leq k \leq K$, $\mathbf{M} = (\mathbf{m}_1; \cdots; \mathbf{m}_K)$ is a $K \times d$ centroid matrix, and $\|\cdot\|_{\mathrm{F}}^2$ denotes the Frobenius norm.

Further, we can rewrite Eq. (4) into matrix form and introduce an $n \times K$ scaled indicator matrix $\mathbf{Q}$, which scales $\mathbf{H}$ by the square root of the cluster size, such that

$$\mathbf{Q} = \mathbf{H} \cdot \mathrm{diag}(|\mathscr{C}_1|, |\mathscr{C}_2|, \cdots, |\mathscr{C}_K|)^{(-1/2)}. \tag{5}$$

If $\mathbf{X}$ is sorted according to the clusters, then $\mathbf{Q} = \mathbf{H}(\mathbf{H}^{\mathsf{T}}\mathbf{H})^{-1/2} = (\mathbf{q}_1, \cdots, \mathbf{q}_K)$ and $\mathbf{q}_k$ is an $n \times 1$ column vector as follows:

$$\mathbf{q}_k = (0, \ldots, 0, \overbrace{1, \ldots, 1}^{|\mathscr{C}_k|}, 0, \ldots, 0)^{\mathsf{T}} / |\mathscr{C}_k|^{1/2}. \tag{6}$$

Based on Eqs. (5)&(6), we can rewrite the objective function of K-means as follows:

$$\begin{aligned} &\min_{\mathbf{H},\mathbf{M}} \|\mathbf{X}-\mathbf{H}\mathbf{M}\|_{\mathrm{F}}^2 \\ &\Leftrightarrow \min_{\mathbf{Q}} \left\{\mathrm{tr}(\mathbf{X}\mathbf{X}^{\mathsf{T}}) - \mathrm{tr}(\mathbf{Q}^{\mathsf{T}}\mathbf{X}\mathbf{X}^{\mathsf{T}}\mathbf{Q})\right\} \\ &\Leftrightarrow \max_{\mathbf{Q}} \mathrm{tr}(\mathbf{Q}^{\mathsf{T}}\mathbf{X}\mathbf{X}^{\mathsf{T}}\mathbf{Q}). \end{aligned} \tag{7}$$

Note that in Eq. (7), $\mathrm{tr}(\mathbf{X}\mathbf{X}^{\mathsf{T}})$ is a constant with respect to $\mathbf{Q}$ and can be ignored in the optimization.

2.3 Optimization Algorithms

K-means clustering is an NP-hard problem, even for two clusters [4], [33]. Greedy heuristic strategies have been proposed to pursue the local minimum. Among existing solvers, Lloyd's [105] and Hartigan's K-means [59] are two popular solvers with convergence guaranteed [143], as shown in Algorithms 1 and 2, respectively. The commonly used centroid initialization randomly chooses K observations from the dataset and uses these as the initial means [19], [57]. Lloyd's K-means has two iterative phases, assigning labels and updating the centroids. It is noteworthythat the centroids are fixed during label assignment. Note that Linde, Buzo, and Gray [92] proposed a methodology to improve Lloyd's technique. They extended Lloyd's results from one to a k-dimensional case. For this reason, the algorithm is known as the LBG (the authors initials) or Generalized Lloyd Algorithm [117]. In contrast, Hartigan's K-means updates the centroids after each point has changed

its label, where only the change of Eq. (1) is calculated. From the perspective of data processing, these two algorithms can be regarded as batch and incremental versions. Both methods have time complexities of $\mathcal{O}(tndK)$ [137], where t is the number K-means iterations. However, Lloyd's K-means is much faster as it can be implemented in parallel and is recognized as the most popular K-means solver. Both Lloyd's and Hartigan's K-means are guaranteed to find the local rather than the global optimum [60]. Some implementations using caching and the triangle inequality to create bounds and accelerate the K-means algorithm can be found in [43], [55], [56], [127], [174].

2.4 Advantages and Limitations of K-means

In this subsection, we summarize the advantages and disadvantages of K-means. K-means clustering is considered one of the fastest and simplest clustering algorithms. It can be distributed in a straightforward manner and scaled up for large-scale data clustering [69], [34]. With more than 50 years since its introduction, the efficiency and effectiveness of K-means have been verified in various practical scenarios [70]. Moreover, beyond the practical value, tremendous efforts have been made to explore the theoretical properties of K-means in terms of its convergence rate [5], [18], [148], initialization [6], [25], [78], and generalization [102]. Some equivalencies between standard K-means and PCA, NMF, spectral clustering, non-parametric Bayesian modeling, and high-order singular value decomposition (SVD) have been established in [37], [40], [41], [84].

Admittedly, several limitations of K-means exist [71]. For example, due to the prototypical assumption, K-means fails to capture non-spherical cluster structures; the sensitivity of K-means initialization heavily affects clustering performance. Some strategies have been developed to cope with these challenges, including divide-and-conquer, K-means++ [6], [8], Monte Carlo sampling [7], and global K-means [91]. It is also arguable that the pre-defined cluster number is another drawback of K-means. In fact, almost every clustering algorithm requires its own parameters, including K-means. However, further discussion regarding its selection is beyond the scope of this paper.

3 Generalizing K-means

This section focuses on how to generalize K-means for building connections with complex problems. We generalize K-means in terms of the objective function and algorithm. Specifically, the input data and K-means distance determine its objective function, while label assignment and centroid updating are two key components in the iterative algorithm. In the following points, we provide the details to generalize K-means in four aspects: *K-means data input, K-means distance, label assignment,* and *centroid updating.*

3.1 K-means Data Input

In the standard K-means formulation in Eq. (1), the input of K-means is the numerical record data $\mathbf{X} \in \mathbf{R}^{n \times d}$. Several K-means extensions have been proposed to learn clustering with different inputs, including categorical data [67], mixed data (both numerical and categorical) [68], and graph [38].

K-modes [26], [67], [185] extends K-means, enabling categorical data clustering. Let $\mathbf{X}$ be n samples, each of which contains d categorical features. Then the objective function of K-modes is given as follows:

$$\min_{\mathscr{C}_k, \mathbf{m}_k} \sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} \sum_{j=1}^{d} \delta(x_{lj}, m_{kj}), \tag{8}$$

where $\{\mathbf{m}_k\}_{k=1}^{K}$ represent the centroids. The symbol $\delta(x_{lj}, m_{kj})$ represents the Kronecker delta function that returns 1 if the features x_{lj} and m_{kj} are in the same category and 0, otherwise. Eq. (8) is equivalent to minimizing the total number of mismatched categories between each sample and the centroid associated with the cluster to which this sample belongs.

K-prototype [67] combines K-means and K-modes, such that it can be used to cluster mixed data. Let $\mathbf{x}_l$ be a d-dimensional mixed sample with p numerical features and $(d-p)$ categorical features. K-prototype objective function is given by the weighted sum of the objective of K-means and K-modes, such that

$$\min_{\mathscr{C}_k, \mathbf{m}_k} \sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} \left\{ \sum_{j=1}^{p} (x_{lj} - m_{kj})^2 + \lambda \sum_{j=p+1}^{d} \delta(x_{lj}, m_{kj}) \right\}, \tag{9}$$

where λ is a hyper-parameter that controls the trade-off between the numerical and categorical features.

K-means can also handle a kernel matrix $\mathbf{X}\mathbf{X}^{\mathsf{T}} \in \mathbf{R}^{n \times n}$, which defines similarity over pairs of data points, as the input, leading to Kernel K-means [135]. In some applications, we may apply a non-linear transform function $\psi(\cdot)$ to each sample to generate high-dimensional features. Let $\mathbf{\Psi} = \{\psi(\mathbf{x}_1), \psi(\mathbf{x}_2), \cdots, \psi(\mathbf{x}_n)\} \in \mathbf{R}^{n \times d'}$ denote the data matrix after this non-linear transformation, and $\boldsymbol{\kappa} = \mathbf{\Psi}\mathbf{\Psi}^{\mathsf{T}} \in \mathbf{R}^{n \times n}$ denotes the kernel matrix. Then the objective function in Eq. (7) can be rewritten as follows:

$$\max_{\mathbf{Q}} \mathrm{tr}(\mathbf{Q}^{\mathsf{T}}\boldsymbol{\kappa}\mathbf{Q}) = \mathrm{tr}(\mathbf{Q}^{\mathsf{T}}\mathbf{\Psi}\mathbf{\Psi}^{\mathsf{T}}\mathbf{Q}). \tag{10}$$

This objective function is expressed as a function of the inner product $\mathbf{\Psi}\mathbf{\Psi}^{\mathsf{T}}$, which can be computed with a properly defined kernel function, such as the radial basis function (RBF) kernel [156]. Therefore, in the computation, we can directly use the kernel function to compute the inner products. It is not necessary to directly compute the coordinates of the data after the non-linear transformation function $\psi(\cdot)$, which may be difficult to compute and possibly in an infinite-dimensional space. Note that $\mathbf{\Psi}\mathbf{\Psi}^{\mathsf{T}}$ can be regarded as a graph input for K-means. In some applications, both graph (or kernel) and record data are available. Several methods extend kernel K-means [63], [161], allowing them to simultaneously utilize the graph (or kernel) and record data.

Another common strategy for cluster graphs is to utilize graph embedding techniques [52], [22], which turn graph data into record data before applying K-means. The most

well-known methods in this category include spectral clustering [140], [118], [160]. Note that although K-means can take both record data and a graph as the input, the corresponding time complexities differ. Lloyd's K-means is designed particularly to handle record data with time complexity $\mathcal{O}(n)$. In contrast, spectral clustering involves an eigenvalue problem, which can be solved with singular value decomposition (SVD) with a time complexity of $\mathcal{O}(n^3)$ [65]. The higher computational complexity of solving the graph clustering problem of K-means motivates researchers to develop scalable methods [29], [162], [62] to improve the efficiency.

Researchers also use K-means and its variants to cluster time-series data [1], [130]. One straightforward strategy is to treat raw time-series data as record data and directly conduct clustering analysis [128], [50], [90], [61]. Alternatively, feature extraction methods, such as SVD [81], wavelet transform [159], and independent component analysis (ICA) [53], are applied to turn time-series data into record data before clustering.

In this paper, we focus on generalized K-means to solve complex problems, especially non-standard cluster analysis problems. Specifically, we present several works that build a connection on the objective function between K-means and other problems. Intuitively, these problems are difficult to solve using K-means directly. Therefore, data transformation is necessary. For example, a kernel matrix is designed to link iterative subspace projection and clustering into a kernel clustering problem; one-hot encoding in consensus clustering is employed to transform the basic partitions into a binary matrix; a co-association graph is decomposed into the record matrix and its transpose; data augmentation concatenates the original feature and partial labels in constrained clustering and domain adaptation; an auxiliary binary matrix is designed to fit the objective function in the clustering and outlier removal. We expand on the details with specific applications regarding these transformations in the following section.

3.2 K-means Distance

The standard K-means applies squared Euclidean distance to calculate the distance between data points and centroids. Beyond squared Euclidean distance, there are rich distance functions suitable for K-means clustering. The K-means objective can now be expressed with the following general formulation to accommodate general K-means distance[1] functions $f(\cdot,\cdot)$:

$$\min_{\mathscr{C}_k,\mathbf{m}_k} \sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} f(\mathbf{x}_l, \mathbf{m}_k)\,. \tag{11}$$

Bregman divergence [10] is a family of distances that fits K-means with arithmetic centroids to guarantee the algorithmic convergence. Let $\phi: \mathbf{R}^d \to \mathbf{R}$ be a differentiable strictly-convex function, then the Bregman loss function $f: \mathbf{R}^d \times \mathbf{R}^d \to \mathbf{R}$ defined by

[1]We use the term K-means distance to represent divergence between the data point and centroids, which is similar to a metric, however, may not satisfy symmetry and triangle inequality.

$$f(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) - \phi(\mathbf{y}) - (\mathbf{x} - \mathbf{y})^{\top} \nabla \phi(\mathbf{y}) . \tag{12}$$

can be used as K-means distance. For example, let $\phi(\mathbf{x}) = \|\mathbf{x}\|^2$, we have $f(\mathbf{x}, \mathbf{y}) = \|\mathbf{x}\|^2 - \|\mathbf{y}\|^2 - (\mathbf{x} - \mathbf{y}) \cdot 2\mathbf{y} = \|\mathbf{x} - \mathbf{y}\|^2$, which is the squared Euclidean distance in standard K-means. Bregman divergences include a large number of useful loss functions such as squared loss, KL-divergence [39], logistic loss, Mahalanobis distance [114], Itakura-Saito distance [21], and I-divergence [113]. Later, Point-to-Centroid (P2C) distance [172] generalizes Bregman divergence with the relaxation on the non-unique minimizer, which has the same mathematical expression as the Bregman divergence in Eq. (12) and also guarantees the convergence of K-means algorithms. In particularly, P2C distance include the widely-used cosine similarity into the K-means distance. Table 1 provides some examples of Bregman divergence and P2C distance. It is worth noting that cosine similarity is a widely used metric in high-dimensional clustering. However, this cannot be generalized into Bregman divergence.

Based on P2C distance, we can rewrite the generalized K-means objective function in Eq. (1) as follows:

$$\begin{aligned} &\sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} f(\mathbf{x}_l, \mathbf{m}_k) \\ &= \sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} \phi(\mathbf{x}_l) - \sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} \phi(\mathbf{m}_k) \\ &- \sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} (\mathbf{x}_l - \mathbf{m}_k)^{\top} \nabla \phi(\mathbf{m}_k) \\ &= \sum_{l=1}^{n} \phi(\mathbf{x}_l) - \sum_{k=1}^{K} |\mathscr{C}_k| \, \phi(\mathbf{m}_k), \end{aligned} \tag{13}$$

where $\sum_{l=1}^{n} \phi(\mathbf{x}_l)$ is a constant and $\sum_{\mathbf{x}_l \in \mathscr{C}_k} (\mathbf{x}_l - \mathbf{m}_k)$ is zero due to the definition of the arithmetic centroid. Therefore, we have

$$\min_{\mathscr{C}_k, \mathbf{m}_k} \sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} f(\mathbf{x}_l, \mathbf{m}_k) \Leftrightarrow \max_{\mathscr{C}_k, \mathbf{m}_k} \sum_{k=1}^{K} |\mathscr{C}_k| \cdot \phi(\mathbf{m}_k) . \tag{14}$$

In Lloyd's K-means, the partition and centroids are iteratively updated. The data points in the same cluster are used to calculate the centroids; while the centroids segment the space into K disjoint parts to determine the partition. The left side of Eq. (14) represents K-means in the partition level, while the right side interprets K-means in the centroid level. In essence, K-means aims to seek K centroids for segmentation according to a certain distance.

If we closely consider the generalized K-means in Eq. (14), the data input discussed in Section 3.1 and distance function discussed in Section 3.2 are two identifying components of the K-means objective function. In particular, extending the distance function allows for solving complex problems with K-means solutions. For example, distance functions

are linked to utility functions in consensus clustering and Holoentropy in outlier detection, which is further discussed in Section 4.

Here, we emphasize that the centroid updating or calculation should match the K-means distance to guarantee algorithmic convergence. Thus far, we have focused on the arithmetic centroids, which fit the above P2C distance. In other words, if K-means or related vector quantization [54], [82] uses an arbitrary metric to calculate the distance between each data point and centroids, the centroid calculation might not be the arithmetic mean of the data points in that cluster anymore. K-medians [71] uses the L_1 norm as the K-means distance, and the centroid is the component-wise median of the points in that cluster [147]; in K-modes [67], the centroid is the mode for each categorical feature in each component; in multi-view K-means [23], the authors used $L_{2,1}$, norm as the K-means distance and the corresponding centroid is updated by setting the derivative of the whole objective function with respect to the centroid to be zero. Accordingly, Eqs. (13)& (14) do not hold for non-arithmetic centroids.

3.3 Non-Exhaustive Overlapping Label Assignment

In K-means, we calculate the distance between each data point and K centroids and assign the data point to its nearest centroid. The indicator matrix $\mathbf{H}$ in the standard K-means in Eq. (4) is binary, where only one non-zero element exists in each row. Recently, Chawla and Gionis [27] and Whang *et al.* [163], [165] extended the traditional label assignment strategy for non-exhaustive or overlapping clustering, where the constraint $\sum_k \mathbf{H}_{ik} = 1$ in Eq. (4) is relaxed to $\sum_k \mathbf{H}_{ik} \in \{0, 1, \ldots, K\}$, and $\mathbf{H}$ remains a binary matrix.

K-means−− [27] simultaneously detects o outliers and partitions the rest $(n - o)$ points into K clusters. During the assignment phase, the distances between each data point and its nearest centroid are calculated. Subsequently, these nearest distances are sorted, where o data points with the largest distances are regarded as the outliers and not assigned to any clusters, such that

$$\begin{cases} \sum_k \mathbf{H}_{ik} = 1, & \text{if } \mathbf{x}_i \text{ is an inlier} \\ \sum_k \mathbf{H}_{ik} = 0, & \text{if } \mathbf{x}_i \text{ is an outlier} \end{cases}. \tag{15}$$

Similarly, NEO-K-means [163], [165] simultaneously considers non-exhaustive and overlapping clustering, where each data point may be an outlier that belongs to none of the clusters, or may belong to one or multiple clusters. In NEO-K-means, the first step is similar to K-mean−−, where $(n - o)$ data points are assigned to their closest clusters. Then, among the remaining $n \times k - (n - o)$ distances, $(o + r)$ assignments are made by taking the smallest distances. As a result, $(n + r)$ assignments are made, where r is a parameter that controls the number of extra assignments. The indicator matrix $\mathbf{H}$ remains binary, where some rows are all zeros or have several non-zero elements, such that

$$\begin{cases} \sum_k \mathbf{H}_{lk} > 1, \text{ if } \mathbf{x}_l \text{ belongs to multiple clusters} \\ \sum_k \mathbf{H}_{lk} = 1, \text{ if } \mathbf{x}_l \text{ belongs to only one cluster} \\ \sum_k \mathbf{H}_{lk} = 0, \text{ if } \mathbf{x}_l \text{ is an outliner} \end{cases}. \tag{16}$$

For both the non-exhaustive and overlapping label assignment cases, it is still possible to employ the arithmetic average to update the centroid, and the convergence of K-means−− and NEO-K-means are guaranteed.

3.4 Incomplete Centroid Updating

In the standard K-means, we calculate the centroid as the arithmetic value of the whole cluster. However, the data matrix may contain missing elements due to device failure, transmission loss, or artificial zeros (See Sections 4.4 and 4.5), which heavily affect the clustering process. In such cases, the updating centroid rule can be changed without missing values included as follows:

$$\mathbf{m}_k = \frac{\sum_{\mathbf{x}_l \in \mathscr{C}_k \bigcap \mathscr{P}} \mathbf{x}_l}{|\mathscr{C}_k \bigcap \mathscr{P}|}, \tag{17}$$

where $\mathscr{P}$ is the set of samples with non-missing values. The missing values should not contribute to the centroid, leading to a smaller denominator than the cluster size. It is noteworthy that the set $\mathscr{P}$ in K-means−− [27] and NEO-K-means [163], [165] is dynamically updated, rather than a fixed one. The convergence of the aforementioned incomplete centroid updating is guaranteed as well [94], [101]. Note that the way of centroid updating should match the K-means distance to ensure algorithmic convergence. Please refer to Section 3.2.

4 Complex Problems: a K-means View

In Section 3, we present the strategies to generalize K-means based on four aspects: data transformation, distance function, label assignment strategy, and centroid updating. In this section, we discuss how these strategies can be applied to solve the following six complex application problems: *iterative subspace selection and clustering, K-means-based consensus clustering, spectral ensemble clustering, partition level constrained clustering, structure-preserved unsupervised domain adaptation* and *clustering with outlier removal*. Table 2 provides a summary of the modifications, denoted as √ necessary to generalize K-means for solving these problems.

4.1 Iterative Subspace Projection and Clustering

In this subsection, we review discriminative clustering [36] that jointly conducts linear discriminant analysis and clustering, which can be solved by two iteratively optimizing the projection matrix and clustering partition. Later, Ye *et al.* demonstrated that iterative

subspace selection and clustering are equivalent to kernel K-means with a specific kernel Gram matrix [179].

Problem Definition.—Beyond clustering algorithms, input data features have significant impact on clustering performance. Many feature engineering practices conducted before clustering are applied to project the original feature onto a low-dimensional subspace. For example, unsupervised dimensionality reduction techniques include principal component analysis (PCA) [74], [45], and various manifold learning algorithms [14], [131]. Subspace learning and deep learning techniques [15], [158], [175], [28], [123] can be used to seek a better representation. However, the aforementioned two separated steps may not necessarily improve the separability of the data for clustering.

One natural solution to tackle this limitation is to *iteratively conduct subspace projection and clustering* in a joint framework [180], [88]. Discriminative clustering [36], a pioneering work along this direction, performs clustering and linear discriminant analysis (LDA) [9] dimensionality reduction simultaneously, where clustering provides the pseudo labels for LDA and LDA seeks the low-dimensional subspace for clustering.

Recall the equivalent relationship between within-cluster variance and inter-cluster separation in Eq. (2). For simplicity, we assume the data is centered, that is, $\sum_{i=1}^{n} \mathbf{x}_i = \mathbf{0}$. Then, we have between-cluster scatter and total scatter matrices as follows:

$$\mathbf{S}_b = \mathbf{X}^\top \mathbf{Q}\mathbf{Q}^\top \mathbf{X}, \text{ and } \mathbf{S}_t = \mathbf{X}^\top \mathbf{X}, \tag{18}$$

where $\mathbf{Q}$ is defined in Eq. (7). $\mathrm{tr}(\mathbf{S}_b)$ captures the inter-cluster distance and $\mathrm{tr}(\mathbf{S}_t)$ captures the total of intra-cluster and inter-cluster distance.

If we consider the scaled cluster indicator $\mathbf{Q}$ as the pseudo label, the supervised dimension reduction can be used to seek a better feature space. Linear Discriminant Analysis (LDA) aims to learn a linear projection matrix $\mathbf{U} \in \mathbf{R}^{d \times d'}$ that maps $\mathbf{X}$ in the d-dimensional space to $\widehat{\mathbf{X}}$ in the d'-dimensional space ($d' \leq d$), i.e., $\widehat{\mathbf{X}} = \mathbf{X}\mathbf{U}$. The optimal solution of $\mathbf{U}$ can be obtained by maximizing the following objective function [9]:

$$\max_{\mathbf{U}} \mathrm{tr}((\mathbf{U}^\top \mathbf{S}_t \mathbf{U})^{-1} \mathbf{U}^\top \mathbf{S}_b \mathbf{U}). \tag{19}$$

To avoid a non-invertible matrix, a regularization technique by adding the identity matrix with a positive regularization parameter λ is widely used to adjust $\mathbf{S}_t$, i.e.m $\widetilde{\mathbf{S}}_t = \mathbf{S}_t + \lambda \mathbf{I}_d$.

In discriminant clustering [42], [36], [178], the transformation matrix $\mathbf{U}$ and the scaled cluster indicator matrix $\mathbf{Q}$ are computed by maximizing the following objective function:

$$\begin{aligned} &\max_{\mathbf{U},\mathbf{Q}} \mathrm{tr}((\mathbf{U}^\top \widetilde{\mathbf{S}}_t \mathbf{U})^{-1} \mathbf{U}^\top \mathbf{S}_b \mathbf{U}) \\ &= \mathrm{tr}((\mathbf{U}^\top (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}_d)\mathbf{U})^{-1} \mathbf{U}^\top \mathbf{X}^\top \mathbf{Q}\mathbf{Q}^\top \mathbf{X}\mathbf{U}). \end{aligned} \tag{20}$$

The above problem can be solved iteratively by alternating between updating $\mathbf{U}$ for a given $\mathbf{Q}$ and updating $\mathbf{Q}$ for a given $\mathbf{U}$ [42], [36], [178]. Later, Ye *et al.* demonstrated that the iterative subspace selection and clustering are equivalent to kernel K-means with a specific kernel Gram matrix [179]. In the following points, we demonstrate a kernel K-means solution to the iterative subspace projection and clustering problem.

Data Transformation.—The problem in Eq. (20) follows from the representer theorem [136] that the optimal linear projection $\mathbf{U}$ can be expressed as $\mathbf{U} = \mathbf{X}^\top\mathbf{V}$, for some matrix $\mathbf{V} \in \mathbf{R}^{n \times d'}$. Let $\mathbf{G} = \mathbf{X}\mathbf{X}^\top$ denote the Gram matrix, the problem in Eq. (20) can be rewritten as follows:

$$\max_{\mathbf{V},\mathbf{Q}} \mathrm{tr}((\mathbf{H}^\top(\mathbf{G}\mathbf{G} + \lambda\mathbf{G})\mathbf{H})^{-1}\mathbf{V}^\top\mathbf{G}\mathbf{Q}\mathbf{Q}^\top\mathbf{G}\mathbf{H}) . \tag{21}$$

By the following theorem, the matrix $\mathbf{H}$ can be factored from the above equation, and the problem in Eq. (20) can be solved using a kernel K-means algorithm.[2]

Theorem 4.1 ([179]). Let $\mathbf{G} = \mathbf{X}\mathbf{X}^\top$ be the Gram matrix and $\lambda > 0$ be the regularization parameter. Let $\mathbf{U}^*$ and $\mathbf{Q}^*$ be the optimal solution to the problem in Eq. (20). Then $\mathbf{Q}^*$ can be obtained by the following maximization problem:

$$\max_{\mathbf{Q}} \mathrm{tr}(\mathbf{Q}^\top(\mathbf{I}_n - (\mathbf{I}_n + \frac{1}{\lambda}\mathbf{G})^{-1})\mathbf{Q}) . \tag{22}$$

By this means, the scaled indicator matrix $\mathbf{Q}$ solving the maximization problem in Eq. (22) can be computed by solving a kernel K-means problem with the kernel Gram matrix given as follows:

$$\boldsymbol{\kappa} = \mathbf{I}_n - (\mathbf{I}_n + \frac{1}{\lambda}\mathbf{X}\mathbf{X}^\top)^{-1} . \tag{23}$$

Distance Function.—The distance function is the squared Euclidean distance, according to the objective functions in the iterative subspace projection and clustering [42], [36], [178], [179].

Label Assignment.—Due to the kernel matrix, the centroids cannot be explicitly presented as the vector formulation. However, it is still possible to calculate the distance between each instance after a non-linear transform function and the centroids with the kernel inputs, and then assign the clustering labels according to its nearest centroid. Let $\psi(\cdot)$ be the non-linear transform function that corresponds to the kernel matrix $\boldsymbol{\kappa}$. Then, the distance can be computed as follows:

[2]·All the proofs can be found in the original papers.

$$\begin{aligned}\|\psi(\mathbf{x}_l)-\mathbf{m}_k\|_2^2 &= \|\psi(\mathbf{x}_l)-\sum_{\mathbf{x}_j\in\mathscr{C}_k}\frac{1}{|\mathscr{C}_k|}\psi(\mathbf{x}_j)\|_2^2\\ &= \psi(\mathbf{x}_l)\psi(\mathbf{x}_l)-\frac{2}{|\mathscr{C}_k|}\sum_{\mathbf{x}_j\in\mathscr{C}_k}\psi(\mathbf{x}_l)\psi(\mathbf{x}_j)\\ &+\frac{1}{|\mathscr{C}_k|^2}\sum_{\mathbf{x}_j\in\mathscr{C}_k}\sum_{\mathbf{x}_{j'}\in\mathscr{C}_k}\psi(\mathbf{x}_j)\psi(\mathbf{x}_{j'})\\ &= \boldsymbol{\kappa}_{ll}-\frac{2}{|\mathscr{C}_k|}\sum_{\mathbf{x}_j\in\mathscr{C}_k}\boldsymbol{\kappa}_{lj}+\frac{1}{|\mathscr{C}_k|^2}\sum_{\mathbf{x}_j\in\mathscr{C}_k}\sum_{\mathbf{x}_{j'}\in\mathscr{C}_k}\boldsymbol{\kappa}_{jj'}.\end{aligned} \tag{24}$$

Centroid Updating.—Due to no explicit vector formulation for centroid, there is no explicit centroid updating.

4.2 K-means-based Consensus Clustering

In this subsection, we review K-means-based consensus clustering (KCC) algorithms that transform consensus clustering with a utility function into a K-means clustering problem. The key to such a transformation is to build the connection between the K-means centroids on basic partitions and the utility function. As a pioneering work, Topchy *et al.* [151] proposed a K-means-based method to tackle consensus clustering with a category utility function. Later, Wu *et al.* [168], [169] generalized Topchy's work, identifying the sufficient and necessary condition for a KCC utility function. Subsequently, Wu *et al.* [170] extended the KCC framework to address fuzzy consensus clustering.

Problem Definition.—*Consensus clustering,* also known as *ensemble clustering,* aims to fuse multiple existing basic partitions into an integrated one [145], [116], [152], which is a fusion problem, rather than a clustering problem. The existing consensus clustering methods can be categorized into two categories, i.e., the methods with and without utility functions, which can also be categorized by measuring similarity between partitions or samples, respectively. The methods that employ the utility function measure similarity between basic partitions and the consensus one [151], [87], [152], [110], [176]. Conversely, the methods that do not employ the utility function use some heuristics or meta-heuristics to transform basic partitions into sample-wise similarities, followed by a graph partition algorithm [46], [109], [145], [44], [30]. More consensus clustering methods can be found in this recent survey [100].

In this subsection, we focus on utility function-based consensus clustering. To understand this problem, we begin by introducing some basic mathematical notations for consensus clustering. Given r basic partitions of $\mathbf{X}$ in $\boldsymbol{\Pi}=\{\boldsymbol{\pi}_1,\boldsymbol{\pi}_2,\cdots,\boldsymbol{\pi}_r\}$, where the basic partitions can be obtained by different clustering algorithms, the same algorithms with different parameters, or the same algorithms on sampled data, the consensus clustering goal aims to fuse these basic partitions into an integrated one. Note that as a fusing problem, consensus clustering inputs are a set of basic partitions $\boldsymbol{\Pi}$, rather than the data matrix $\mathbf{X}$. Consensus clustering with a utility function has the following objective function:

$$\max_{\boldsymbol{\pi}} \Gamma(\boldsymbol{\pi}, \boldsymbol{\Pi}) = \sum_{i=1}^{r} w_i U(\boldsymbol{\pi}, \boldsymbol{\pi}_i), \tag{25}$$

where $\Gamma(\boldsymbol{\pi}, \boldsymbol{\Pi}): \mathbf{Z}_{++}^{n} \times \mathbf{Z}_{++}^{n \times r} \mapsto \mathbf{R}$ is a *consensus function* and $U: \mathbf{Z}_{++}^{n} \times \mathbf{Z}_{++}^{n} \mapsto \mathbf{R}$ is a *utility function* to measure similarity between two partitions, *i.e.*, one basic partition and the consensus one, and $w_i \in [0, 1]$ is the weight for $\boldsymbol{\pi}_i$, with $\sum_{i=1}^{r} w_i = 1$.

The challenges for solving the problem in Eq. (25) can be divided into two aspects, how to design an effective utility function and how to solve it efficiently. To better understand utility functions, we present the contingency matrix in Table 3. Given two partitions: $\boldsymbol{\pi}$ and $\boldsymbol{\pi}_i$, containing K and K_i clusters, respectively. In the table, $n_{kj}^{(i)}$ denotes the number of data objects belonging to both clusters $\mathscr{C}_j^{(i)}$ in $\boldsymbol{\pi}_i$ and cluster $\mathscr{C}_k$ in $\boldsymbol{\pi}$, $n_{k+} = \sum_{j=1}^{K_i} n_{kj}^{(i)}$, and $n_{+j}^{(i)} = \sum_{k=1}^{K} n_{kj}^{(i)}$, $1 \le j \le K_i$, $1 \le k \le K$. By dividing the numbers in the table by the total number of data points, we have $p_{kj}^{(i)} = n_{kj}^{(i)} / n$, $p_{k+} = n_{k+} / n$, and $p_{+j}^{(i)} = n_{+j}^{(i)} / n$, based on which utility functions can be defined. For example, categorical utility function [115] is one of the most widely used utility functions, and can be computed as follows:

$$U_c(\boldsymbol{\pi}, \boldsymbol{\pi}_i) = \sum_{k=1}^{K} p_{k+} \sum_{j=1}^{K_i} \left(\frac{p_{kj}^{(i)}}{p_{k+}}\right)^2 - \sum_{j=1}^{K_i} \left(p_{+j}^{(i)}\right)^2. \tag{26}$$

From the definition in Eq. (26), the categorical utility function measures the difference between how to predict the consensus partition $\boldsymbol{\pi}$ with and without $\boldsymbol{\pi}_i$. It is noteworthy that the second term is a constant given $\boldsymbol{\pi}_i$.

In the literature, Topchy *et al.* [151] proposed a K-means-based method to tackle the consensus clustering with the category utility function, which attracted significant interest due to its simplicity and efficiency. Along this direction, Wu *et al.* [168], [169] provided a theoretic framework of K-means-based consensus clustering for the utility function-based consensus clustering. Initially, no connection exists between consensus clustering and K-means clustering, which are different research problems, in essence. Data transformation and distance function reformulation are necessary to rewrite consensus clustering into an objective function with the K-means formulation.

Data Transformation.—The consensus clustering input is a set of basic partitions. Wu *et al.* introduced the binary matrix for K-means clustering [168], [169]. Let $\mathbf{B} = \{\mathbf{b}_l \mid 1 \le l \le n\}$ be an $n \times \sum_{i=1}^{r} K_i$ binary data set derived from the set of r basic partitions $\boldsymbol{\Pi}$ as follows:

$$\begin{aligned} &\mathbf{b}_l = (\mathbf{b}_{l,1}, \cdots, \mathbf{b}_{l,i}, \cdots, \mathbf{b}_{l,r}), \quad \text{with} \\ &\mathbf{b}_{l,i} = (\mathbf{b}_{l,i,1}, \cdots, \mathbf{b}_{l,i,j}, \cdots, \mathbf{b}_{l,i,Ki}), \quad \text{and} \\ &b_{l,i,j} = \begin{cases} 1, & \text{if } L_{\pi_i}(\mathbf{x}_l) = j \\ 0, & \text{otherwise} \end{cases}. \end{aligned} \tag{27}$$

From the aforementioned Eq. (27), the binary matrix is the concatenation of each basic partition with one-hot encoding.

Distance Function.—Recall that in the K-means objective function in Eq. (14), the centroid and ϕ in the distance function are two components. Wu *et al.* linked the distance function with the utility function and provided the KCC utility function [168], [169], which is the utility function for the K-means solution.

When running K-means clustering on the binary matrix $\mathbf{B}$, the following lemma shows the centroid formulation.

Lemma 4.2 ([169]). For K-means clustering on the binary data set $\mathbf{B}$, the k-th centroid $\mathbf{m}_k$ satisfies

$$\begin{aligned}&\mathbf{m}_k = (\mathbf{m}_{k,1}, \cdots, \mathbf{m}_{k,i}, \cdots, \mathbf{m}_{k,r}), \text{ with}\\ &\mathbf{m}_{k,i} = \left(\frac{p_{k1}^{(i)}}{p_{k+}}, \cdots, \frac{p_{kj}^{(i)}}{p_{k+}}, \cdots, \frac{p_{k\kappa_i}^{(i)}}{p_{k+}},\right), \forall\, k, i\,.\end{aligned} \tag{28}$$

While Lemma 4.2 is extremely simple, it unveils critical information about the construction of KCC. Upon close consideration of the first term in the categorical utility function in Eq. (26), it is interesting to observe that the categorical utility function employs the elements in the centroid vector in Eq. (28). By this means, consensus clustering in Eq. (25) with the categorical utility function can be solved by K-means clustering on $\mathbf{B}$ with the squared Euclidean distance [151]. Beyond the categorical utility function, Wu *et al.* [168], [169] also provided other types of utility functions that benefit from the K-means solution. Therefore, they formally introduced a definition of the KCC utility function, which acts as a utility function for the consensus function, and relies on the K-means heuristic to find the consensus partition.

Definition 4.3 (KCC Utility Function [169]). A utility function U is a KCC utility function, if $\forall \mathbf{\Pi} = \{\pi_1, \cdots, \pi_r\}$ and $K \geq 2$, there exists a distance function f such that

$$\max_{\pi \in F} \sum_{i=1}^{r} w_i U(\pi, \pi_i) \Leftrightarrow \max_{\pi \in F} \sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathcal{C}_k} f(\mathbf{b}_l, \mathbf{m}_k), \tag{29}$$

where F is the space of all possible clustering solutions with n data points.

Based on the aforementioned definition, the following theorem uncovers the sufficient and necessary condition for utility functions to become a KCC utility function.

Theorem 4.4 ([169]). U is a KCC utility function, if and only if $\forall \Pi = \{\pi_1, \cdots, \pi_r\}$ and $K \geq 2$, there exists a set of continuously differentiable convex functions $\{\mu_1, \cdots, \mu_r\}$ such that:

$$U(\boldsymbol{\pi}, \boldsymbol{\pi}_i) = \sum_{k=1}^{K} p_{k+} \mu_i \left(\frac{p_{k1}^{(i)}}{p_{k+}}, \cdots, \frac{p_{kj}^{(i)}}{p_{k+}}, \cdots, \frac{p_{k\kappa_i}^{(i)}}{p_{k+}},\right), \forall\, i\,. \tag{30}$$

The convex function ϕ for the corresponding K-means distance in Eq. (12) is given by:

$$\phi(\mathbf{m}_k) = \sum_{i=1}^{r} w_i v_i(\mathbf{m}_{k,i}), \forall\, k, \text{ with}$$
$$v_i(\mathbf{x}) = a\mu_i(\mathbf{x}) + c_i, \forall\, i, a \in \mathbf{R}_{++}, c_i \in \mathbf{R}. \tag{31}$$

Theorem 4.4 provides the necessary and sufficient condition for a KCC utility function, which is also serves as the criterion to verify whether a given utility function is a KCC utility function. That is, a KCC utility function must be a weighted average of a set of convex functions defined on $(p_{k1}^{(i)} / p_{k+}, \cdots, p_{kj}^{(i)} / p_{k+}, \cdots, p_{kK_i}^{(i)} / p_{k+})$, $1 \le i \le r$, respectively, which is actually the centroid of K-means on the binary matrix **B**. Table 4 provides sample KCC utility functions.

With data transformation and modification of the distance function, Wu *et al.* [168], [169] mapped consensus clustering with the KCC utility function into a K-means objective function. Therefore, Lloyd's algorithm can be used to find a solution efficiently, which is described as follows:

Label Assignment.—Each data point is assigned to its nearest centroid based on some distance function according to Eq. (12) and (31).

Centroid Updating.—The centroid is updated by the arithmetic mean of each cluster (standard approach).

4.3 Spectral Ensemble Clustering

In this subsection, we present spectral ensemble clustering (SEC) [96], [102], a method in second category of consensus clustering using the co-association matrix to measure similarity between data points. By transforming the co-association matrix into a binary matrix and its transpose, SEC is solved with a weighted K-means clustering, and dramatically reduces the time and space complexities of standard spectral clustering on the co-association matrix from $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$, respectively, to both $\mathcal{O}(n)$.

Problem Definition.—Beyond the utility-based consensus clustering methods in Section 4.2, co-association matrix-based methods provide an alternative strategy for learning a consensus clustering solution, where a co-association matrix is constructed to measure the number of a pair of instances occurring simultaneously in the same cluster among different basic partitions. Based on that, consensus clustering, a fusion problem, can be cast into the conventional graph partition problem, where agglomerative hierarchical clustering and spectral clustering can be followed to solve the problem [46], [109]. However, these methods also suffer from some non-ignored drawbacks, *i.e.,* the high time and space complexities of $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$ prevent them from handling large-scale data.

To reduce the huge time and space complexities, Liu *et al.* proposed spectral ensemble clustering (SEC) [96], [102], which initially aims to apply spectral clustering on the co-association matrix for the final consensus partition, and finally solves it by a weighted

K-means clustering. Here, we continue to use the variables in Section 4.2. Given r basic partitions $\mathbf{\Pi} = \{\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \cdots, \boldsymbol{\pi}_r\}$, a *co-association matrix* $\mathbf{S} = \{s_{lq} \mid 1 \leq l, q \leq n\} \in \mathbf{R}^{n \times n}$ is defined as follows [46]:

$$s_{lq} = \sum_{i=1}^{r} \delta(\boldsymbol{\pi}_i(\mathbf{x}_l), \boldsymbol{\pi}_i(\mathbf{x}_q)), \tag{32}$$

where $\delta(\boldsymbol{\pi}_i(\mathbf{x}_l), \boldsymbol{\pi}_i(\mathbf{x}_q))$ represents the Kronecker delta function that returns 1 if the features $\mathbf{x}_l$ and $\mathbf{x}_q$ are with the same category in the basic partition $\boldsymbol{\pi}_i$ and 0, otherwise.

The objective function of normalized-cut spectral clustering on $\mathbf{S}$ can be expressed as the following trace maximization problem [37]:

$$\max_{\mathbf{Z}} \frac{1}{K} \mathrm{tr}(\mathbf{Z}^\top \mathbf{D}^{-1/2} \mathbf{S} \mathbf{D}^{-1/2} \mathbf{Z}), \ \mathrm{s.t.} \ \mathbf{Z}^\top \mathbf{Z} = \mathbf{I}, \tag{33}$$

where $\mathbf{D}$ is a diagonal matrix of $\mathbf{S}$ with $\mathbf{D} = \mathrm{diag}(d_1, \cdots, d_l, \cdots, d_n)$ with $d_l = \sum_{q=1}^{n} s_{lq}$, $\mathbf{Z} = \mathbf{D}^{1/2} \mathbf{H} (\mathbf{H}^\top \mathbf{D} \mathbf{H})^{-1/2}$, and $\mathbf{H}$ is the partition indicator matrix.

However, performing the standard spectral clustering on the co-association matrix suffers from a significant time complexity. To address this challenge, SEC builds the connection between spectral clustering on the co-association matrix and weighted K-means, as described in the following points.

Data Transformation.—The input co-association matrix can be regarded as a graph that measures the pairwise similarity between instances. Liu *et al.* [96], [102] decomposed the co-association matrix into the record data to accelerate computation. According to the co-association matrix definition, $\mathbf{S} = \mathbf{B}\mathbf{B}^\top$, where $\mathbf{B}$ is the $n \times \sum_{i=1}^{r} K_i$ binary matrix defined in Eq. (27). A weighted K-means is employed on the matrix with $\mathbf{b}_l / w_l$, rather than $\mathbf{B}$ itself due to the objective function in Eq. (33), where $w_l = d_l = \sum_{i=1}^{r} \sum_{q=1}^{n} \delta(\boldsymbol{\pi}_i(\mathbf{x}_l), \boldsymbol{\pi}_i(\mathbf{x}_q))$. The weight of each data point is the summation of the cluster size to which the data point belongs in each basic partition.

Distance Function.—Due to the transformation between the trace formulation and Frobenius norm, the squared Euclidean calculates the distance between each instance and centroids. With the data transformation and distance function, the objective function can be written in Eq. (33) in the K-means version using the following theorem.

Theorem 4.5 ([102]). Given a set of basic partitions $\mathbf{\Pi}$, the spectral clustering on $\mathbf{S}$ is equivalent to a weighted K-means clustering of a variant of $\mathbf{B}$; that is,

$$\begin{aligned} &\max_{\mathbf{Z}} \frac{1}{K} \mathrm{tr}(\mathbf{Z}^\top \mathbf{D}^{-1/2} \mathbf{S} \mathbf{D}^{-1/2} \mathbf{Z}) \\ \Leftrightarrow &\max_{\mathscr{C}_k, \mathbf{m}_k} \sum_{k=1}^{K} \sum_{\mathbf{x}_l \in \mathscr{C}_k} w_l \| \frac{\mathbf{b}_l}{w_l} - \mathbf{m}_k \|^2, \end{aligned} \tag{34}$$

where $\mathbf{m}_k = \sum_{\mathbf{x}_l \in \mathscr{C}_k} \mathbf{b}_l / \sum_{\mathbf{x}_l \in \mathscr{C}_k} w_l$, and $w_l = d_l = \sum_{i=1}^{r} \sum_{q=1}^{n} \delta(\boldsymbol{\pi}_i(\mathbf{x}_l), \boldsymbol{\pi}_i(\mathbf{x}_q))$.

By the above transformation, the time complexity of SEC is $\mathcal{O}(tnrK)$. Thus, the transformation dramatically reduces the time and space complexities from $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$ of the standard spectral clustering on the co-association matrix, respectively, to $\mathcal{O}(n)$. Note that there is only one non-zero element in $\mathbf{b}_{l,i}$. Accordingly, while the weighted K-means is conducted on a highly sparse matrix, the real dimensionality in computation is merely r, the number of basic partitions. Note that w_l in Eq. (34) is the weight for l-th instance, while w_i in Eq. (25) is the weight for i-th basic partition.

Dhillon *et al.* [37] uncovered the connection between the general spectral clustering and weighted kernel K-means. Here, Liu *et al.* [96], [102] considered the spectral ensemble clustering, a special case of spectral clustering, and discovered the kernel mapping function, which is the binary data dividing its corresponding weight, i.e., $\psi(\mathbf{x}_l) = \mathbf{b}_l / w_l$ according to the property of the kernel matrix $\boldsymbol{\kappa} = \mathbf{S} = \mathbf{B}\mathbf{B}^\intercal$. By this means, SEC is transformed into a weighted K-means clustering, where the data transformation is crucial for gaining high efficiency for SEC and ensures its practical feasibility. Finally, the standard Lloyd's algorithm can be used for an efficient solution using the following label assignment and centroid updating.

Label Assignment.—Each data point is assigned to its nearest centroid according to the squared Euclidean distance.

Centroid Updating.—As weighted K-means is employed, the centroid is updated by the weighted arithmetic average of each cluster by Eq. (34).

4.4 Partition Level Constrained Clustering

In this subsection, we present partition level constrained clustering [94], [101] by exploring the intrinsic structure from the data with the guidance from the side information. Based on the strategies described in Section 3, the authors introduced a concatenated matrix to the original data matrix and partition-level side information and solved it via a modified K-means distance function and centroid updating rule.

Problem definition.—*Constrained clustering* applies the side information to guide the clustering process [164], [166], [124], [12]. Pairwise constraints are one type of the side information, where must-link and cannot-link constraints indicate whether two instances should lie in the same cluster or not, respectively [11], [144], [89]. However, it is typically challenging to make pairwise decisions in real-world applications because prior knowledge or references are generally insufficient. In contrast to pairwise constraints, Liu *et al.* [94], [101] proposed another type of the side information, named partition level side information or partial labels, which is defined as follows.

Definition 4.6 (p-***Partition Level Side Information [101]).*** A portion $p \in (0, 1)$ of n data instances is annotated as the cluster labels from 1 to K, where K is the user-predefined cluster number. Such the label annotation is called p–partition level side information.

Unlike pairwise constraints, partition level side information treats the side information as a whole, which is of high consistency and avoids self-contradictory derived from the pairwise constraints. The clustering problem with partition level side information is different from the conventional classification problem. The former takes all labeled and unlabeled data for training and discovers the whole structure, while the latter only uses labeled data for training and seeks a decision boundary. It is noteworthy that the cluster number in the partition level side information may be different from the one in the later clustering process. In such a scenario, the classification problem cannot assist in discovering novel classes.

Partition level constrained clustering aims to find a partition that captures the intrinsic structure from the data and is of high consistency with the partition level side information. Recall that the utility function in the above two subsections plays a role in measuring the similarity of two partitions. It inspires us to employ the categorical utility function in Eq. (26) as a regularizer for partition level constrained clustering.

Let $\mathbf{X}$ be the $n \times d$ data matrix and $\mathbf{P}$ be an $np \times K$ side information matrix containing np instances in K clusters, where $\mathbf{P}$ is in the format of one-of-K coding. The objective function of the partition level constrained clustering is as follows:

$$\min_{\mathbf{H},\mathbf{M}} \|\mathbf{X} - \mathbf{H}\mathbf{M}\|_{\mathrm{F}}^2 - \lambda U_c(\mathbf{H} \otimes \mathbf{P}, \mathbf{P}), \tag{35}$$

where $\mathbf{H}$ is the $n \times K$ indicator matrix, $\mathbf{M}$ is the $K \times d$ centroid matrix, $\otimes$ is an operator to trim $\mathbf{H}$ according to the common instance in both $\mathbf{H}$ and $\mathbf{P}$, and λ is a positive parameter to present the side information confidence degree. In the original papers [94], [101], the authors set $\lambda = 100$ as the default setting. Later, one following work [144] found that the method performs more stably for $\lambda = 100 \cdot \mathrm{tr}(\mathbf{X}\mathbf{X}^{\mathsf{T}})$, i.e., for λ being proportional to the trace of the sample covariance matrix of data set $\mathbf{X}$.

The objective function in Eq. (35) consists of two parts. The first term is the standard K-means, and the second is the categorical utility function measuring the disagreement between the side information $\mathbf{P}$ and the counterpart in $\mathbf{H}$. The first term in Eq. (35) is the K-means formulation. It is necessary to transform the second term in Eq. (35), such that the problem can be solved with a unified K-means framework.

Data Transformation.—To begin with, we first present the following lemma that transforms the second term in Eq. (35).

Lemma 4.7 ([101]). Given one fixed partition $\mathbf{P}$, we have

$$\max_{\mathbf{H}} U_c(\mathbf{H}, \mathbf{P}) \Leftrightarrow \min_{\mathbf{H},\mathbf{G}} \|\mathbf{P} - \mathbf{H}\mathbf{G}\|_{\mathrm{F}}^2, \tag{36}$$

where $\mathbf{G}$ is a $K \times K$ matrix, where the k-th row of $\mathbf{G}$ is $(p_{k1} / p_{k+}, \cdots, p_{kK} / p_{k+})$.

The above equivalent relationship between $\|\mathbf{P} - \mathbf{H}\mathbf{G}\|_{\mathrm{F}}^2$ and $U_c(\mathbf{H}, \mathbf{P})$ holds for any $\mathbf{H}$, because $\|\mathbf{P} - \mathbf{H}\mathbf{G}\|_{\mathrm{F}}^2 + np \cdot U_c(\mathbf{H}, \mathbf{P})$ is a constant with given $\mathbf{P}$. Lemma 4.7 introduces one extra variable

G to capture the mapping relationship between **P** and **H**. After aligning **P** to **H** with **G**, the objective function in Eq. (35) can be rewritten as follows:

$$\min_{\mathbf{H}_1, \mathbf{H}_2, \mathbf{M}, \mathbf{G}} \|\mathbf{X}_1 - \mathbf{H}_1\mathbf{M}\|_{\mathrm{F}}^2 + \|\mathbf{X}_2 - \mathbf{H}_2\mathbf{M}\|_{\mathrm{F}}^2 + \lambda\|\mathbf{P} - \mathbf{H}_1\mathbf{G}\|_{\mathrm{F}}^2, \tag{37}$$

where the data matrix **X** and the indicator matrix **H** are separated into $\mathbf{X}_1$, $\mathbf{H}_1$ with np instances and $\mathbf{X}_2$, and $\mathbf{H}_2$ with the rest $n(1-p)$ instances, respectively, according to the side information **P**. Based on the new objective function in Eq. (37), Liu *et al.* [94], [101] provided a K-means-like optimization with a modified distance function and a new centroid updating rule.

The partition-level constrained clustering input involves two parts, the original data matrix, and the side information. However, a record data formulation is required to convert to a generalized K-means. It is natural to concatenate the original data matrix and side information. However, some data points do not consist of side information, which makes the concatenated matrix incomplete. To address this, Liu *et al.* [94], [101] used zeros to fill up the matrix. Therefore, the $n \times (d + K)$ concatenated matrix $\mathbf{D} = \{\mathbf{d}_l \mid 1 \le l \le n\}$ is described as follows:

$$\mathbf{D} = \begin{bmatrix} \mathbf{X}_1 & \mathbf{P} \\ \mathbf{X}_2 & \mathbf{0} \end{bmatrix}. \tag{38}$$

Further **D** can be decomposed into two parts $\mathbf{D} = [\mathbf{D}_1\, \mathbf{D}_2]$, where $\mathbf{D}_1 = \mathbf{X}$ and $\mathbf{D}_2 = [\mathbf{P}\, \mathbf{0}]^{\mathsf{T}}$. Zeros in this matrix are the auxiliary fill-ups, rather than observed data. Therefore, the distance function and centroid updating are modified accordingly to handle these zeros.

Distance Function.—Since the concatenated matrix has two parts, the distance function also consists of two components, where the auxiliary zeros are not involved in the calculation.

$$f(\mathbf{d}_l, \mathbf{m}_k) = \|\mathbf{d}_l^{(1)} - \mathbf{m}_k^{(1)}\|_2^2 + \lambda \mathbf{1}(\mathbf{d}_l \in \mathbf{P})\|\mathbf{d}_l^{(2)} - \mathbf{m}_k^{(2)}\|_2^2. \tag{39}$$

In the above equation, $\mathbf{1}(\mathbf{d}_l \le \mathbf{P}) = 1$ indicates the side information contains $\mathbf{x}_l$, and 0 otherwise. $\mathbf{m}_k^{(1)}$ and $\mathbf{m}_k^{(2)}$ denote the centroid in the original and side information space.

Label Assignment.—Each data point is assigned to its nearest centroid according to the distance function defined in Eq. (39).

Centroid Updating.—As the partition level side information guides the clustering process in a utility way, those auxiliary zero values should not contribute to similarity of two partitions, which will affect the centroid updating. Let $\mathbf{m}_k = (\mathbf{m}_k^{(1)}, \mathbf{m}_k^{(2)})$ be the k-th centroid of K-means, where $\mathbf{m}_k^{(1)} = (m_{k,1}, \cdots, m_{k,d})$ and $\mathbf{m}_k^{(2)} = (m_{k,d+1}, \cdots, m_{k,d+K})$. Liu *et al.* [94], [101] modified the computation of centroids as follows:

$$\mathbf{m}_k^{(1)} = \frac{\sum_{\mathbf{d}_l \in \mathscr{C}_k} \mathbf{d}_l^{(1)}}{|\mathscr{C}_k|}, \ \mathbf{m}_k^{(2)} = \frac{\sum_{\mathbf{d}_l \in \mathscr{C}_k \bigcap \mathbf{P}} \mathbf{d}_l^{(2)}}{|\mathscr{C}_k \bigcap \mathbf{P}|}. \tag{40}$$

Here, in Eq. (40), our centroids have two parts $\mathbf{m}_k^{(1)}$ and $\mathbf{m}_k^{(2)}$. For $\mathbf{m}_k^{(1)}$, the denominator is still $|\mathscr{C}_k|$; albeit for $\mathbf{m}_k^{(2)}$, it is $|\mathscr{C}_k \cap \mathbf{P}|$.

Considering these four aforementioned points, Liu *et al.* [94], [101] provided the following Theorem 4.8 for the K-means solution.

Theorem 4.8 ([101]). Given the data matrix $\mathbf{X}$, side information $\mathbf{P}$ and auxiliary matrix $\mathbf{D} = \{\mathbf{d}_l\} 1 \leq l \leq n$, we have

$$\begin{aligned} &\min_{\mathbf{H},\mathbf{M},\mathbf{G}} \|\mathbf{X} - \mathbf{H}\mathbf{M}\|_{\mathrm{F}}^2 + \lambda \|\mathbf{P} - (\mathbf{H} \otimes \mathbf{P})\mathbf{G}\|_{\mathrm{F}}^2 \\ \Leftrightarrow &\min_{\mathscr{C}_k, \mathbf{m}_k} \sum_{k=1}^{K} \sum_{\mathbf{d}_l \in \mathscr{C}_k} f(\mathbf{d}_l, \mathbf{m}_k), \end{aligned} \tag{41}$$

where $\mathbf{m}_k$ is the k-th centroid calculated by Eq. (40) and the distance function f is calculated in Eq. (39).

Theorem 4.8 maps the problem in Eq. (35) into a K-means-like optimization with modified a distance function and centroid updating rules, providing an elegant formulation that can be solved with high efficiency. By this means, the problem in Eq. (35) is solved by iteratively assigning the points to the centroid by Eq. (39) and updating the centroids by Eq. (40). Upon close consideration of the concatenated matrix $\mathbf{D}$, the side information can be regarded as new features, which provides an approach to cluster mixed data.

As constrained clustering, as addressed here, is solved by a modified K-means clustering with incomplete centroid updating and the partial distance function, the objective function is still guaranteed to converge to a local minimum by Theorem 4.9.

Theorem 4.9. The objective function value of the problem in Eq. (35) would continuously decrease and converge to a local minimum via K-means clustering with the centroid updating rule in Eq. (40) and the distance function in Eq. (39).

4.5 Structure-Preserved Unsupervised Domain Adaptation

In this subsection, we present structure-preserved unsupervised domain adaptation for single- and multi-source scenarios [99], [97]. Specifically, the authors employed the aforementioned partition level constrained clustering to address the domain adaptation problem via a K-means solution.

Unsupervised domain adaptation aims to recognize the target data with the assistance of auxiliary labeled source data. Due to the divergence between the source and data domains, domain alignment and knowledge transfer are two crucial processes in domain adaptation. Most unsupervised domain adaptation methods focus on the first problem, which can be approximately separated into four branches, discrepancy-, adversarial-learning, self-training,

and optimal-transport-based. Discrepancy-based domain adaptation estimates and minimizes the marginal and conditional distribution difference between the source and target domain [154], [106], [108], [146]. Adversarial-learning-based domain adaptation learns the domain-invariant representation to align the source and target domain [48], [153], [133], [183], [182], [86], [31]. Recently, self-training with networks has become a promising alternative towards domain adaptation [186], [104], [103], which involves an iterative process training a network on the target domain, and generated pseudo labels are used to re-train the network. Optimal transport has been applied in domain adaptation to mitigate the domain gap by minimizing the cost of complex distributions and aligning the representations across domains [139], [32], [119]. In contrast to the above categories, Liu *et al.* [99], [97] addressed the second problem that with well-aligned representation, how knowledge should be effectively transferred from source to target domain. Assuming that the projection matrix is given and the source data have labels, they formulated the domain adaptation problem as a partition level constrained clustering, which can be regarded as an extension of Section 4.4 in a different application scenario. In the following points, we present their methods for handling unsupervised domain adaptation in terms of single- and multi-source scenarios in Sections 4.5.1 and 4.5.2, respectively.

4.5.1 Single-source unsupervised domain adaptation

Problem Definition.: To capture the structure of different domains for effective transfer, Liu *et al.* [99], [97] formulated this as a constrained clustering problem. After domain alignment, they put the source and target data together for clustering with a partition level constraint on the source data. Let $\mathbf{Z}_S \in \mathbf{R}^{n_s \times d}$ and $\mathbf{Z}_T \in \mathbf{R}^{n_t \times d}$ denote the representations in the shared space of source and target data and $\mathbf{Y}_S \in \mathbf{R}^{n_s \times K}$ denote the source data label. Their objective function can be written as follows:

$$\min_{\mathbf{H}_S, \mathbf{H}_T, \mathbf{M}} \left\| \begin{bmatrix} \mathbf{Z}_S \\ \mathbf{Z}_T \end{bmatrix} - \begin{bmatrix} \mathbf{H}_S \\ \mathbf{H}_T \end{bmatrix} \mathbf{M} \right\|_{\mathrm{F}}^2 - \lambda U_c(\mathbf{H}_S, \mathbf{Y}_S), \tag{42}$$

where $\mathbf{Z}_S$, $\mathbf{Z}_T$, $\mathbf{Y}_S$ are input variables, $\mathbf{H}_S \in \mathbf{R}^{n_s \times K}$ and $\mathbf{H}_T \in \mathbf{R}^{n_t \times K}$ are the unknown assignment matrices for the source and target data, respectively. $\mathbf{M}$ is the corresponding centroid matrix for K-means clustering and λ is a positive trade-off parameter.

The aforementioned problem in Eq. (42) is similar to the one in Eq. (35), where a K-means-like solution can be achieved using our four key points.

Data Transformation.: The source and target data are put together with the source label information. For target data without labels, zeros are used to fill up the matrix. Then, we have the $(n_s + n_t) \times (d + K)$ concatenated matrix $\mathbf{D}$ as follows:

$$\mathbf{D} = \begin{bmatrix} \mathbf{Z}_S & \mathbf{Y}_S \\ \mathbf{Z}_T & \mathbf{0} \end{bmatrix}. \tag{43}$$

The corresponding distance function and centroid updating rule are similar to the ones in Eq. (40) and (39).

Distance Function.: As the concatenated matrix has two parts, the distance function also consists of two components, where the auxiliary zeros are not involved in the calculation. The corresponding distance function is similar to the one in Eq. (39).

Label Assignment.: Each data point is assigned to its nearest centroid according to the distance function in Eq. (39).

Centroid Updating.: The centroid updating rule is similar to the one in Eq. (40), where the auxiliary zeros do not contribute to the centroids.

4.5.2 Multi-source unsupervised domain adaptation

Problem Definition.: For the multi-source scenarios, we are given two source domains and one target domain, without loss of generality. Here a pre-learned shared space with different projections for these two source domains leads to $\mathbf{Z}_{S_1} \in \mathbf{R}^{n_{s_1} \times d_1}$, $\mathbf{Z}_{T_1} \in \mathbf{R}^{n_t \times d_1}$, $\mathbf{Z}_{S_2} \in \mathbf{R}^{n_{s_2} \times d_2}$ and $\mathbf{Z}_{T_2} \in \mathbf{R}^{n_t \times d_2}$. Then the optimization problem for a multi-source setting can be written as follows:

$$\begin{aligned} \min_{\mathbf{H}_{S_{1/2}}, \mathbf{H}_T, \mathbf{M}_{1/2}} & \left\| \begin{bmatrix} \mathbf{Z}_{S_1} \\ \mathbf{Z}_{T_1} \end{bmatrix} - \begin{bmatrix} \mathbf{H}_{S_1} \\ \mathbf{H}_T \end{bmatrix} \mathbf{M}_1 \right\|_{\mathrm{F}}^2 - \lambda U_c(\mathbf{H}_{S_1}, \mathbf{Y}_{S_1}) \\ & + \left\| \begin{bmatrix} \mathbf{Z}_{S_2} \\ \mathbf{Z}_{T_2} \end{bmatrix} - \begin{bmatrix} \mathbf{H}_{S_2} \\ \mathbf{H}_T \end{bmatrix} \mathbf{M}_2 \right\|_{\mathrm{F}}^2 - \lambda U_c(\mathbf{H}_{S_2}, \mathbf{Y}_{S_2}) . \end{aligned} \tag{44}$$

The target data have different representations as $\mathbf{Z}_{T_1}$ and $\mathbf{Z}_{T_2}$ with different d_1 and d_2 dimensions of common spaces, but share the same class structure $\mathbf{H}_T \in \mathbf{R}^{n_t \times K}$. λ is a positive trade-off parameter. For more than two source domains, Liu *et al.* [99], [97] aligned each of them to the target domain with the source label consistency constraint. To solve the problem in Eq. (44), we introduce an auxiliary matrix for data transformation.

Data Transformation.: The two source and target data are put together with the source label information according to different domains. Then, we have the $(n_{s_1} + n_{s_2} + n_t) \times (d_1 + K + d_2 + K)$ concatenated matrix $\mathbf{D}$ as follows:

$$\mathbf{D} = \begin{bmatrix} \mathbf{Z}_{S_1} & \mathbf{Y}_{S_1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{Z}_{S_2} & \mathbf{Y}_{S_2} \\ \mathbf{Z}_{T_1} & \mathbf{0} & \mathbf{Z}_{T_2} & \mathbf{0} \end{bmatrix}. \tag{45}$$

Unlike the auxiliary matrix in Eq. (43), zeros here are also employed to fill up the elements between two source domains, because the projection between source domains is not pre-learned.

Distance Function.: As the concatenated matrix has four parts, the distance function also consists of four components, where the auxiliary zeros are not involved in the calculation. Let $\mathbf{D}_1 = \{\mathbf{d}_l^{(1)}\} = \mathbf{Z}_{S_1} \cup \mathbf{Z}_{T_1}$ and $\mathbf{D}_2 = \{\mathbf{d}_l^{(2)}\} = \mathbf{Z}_{S_2} \cap \mathbf{Z}_{T_2}$, the distance function is as follows:

$$
\begin{aligned}
&f(\mathbf{d}_l, \mathbf{m}_k) = \\
&= \mathbf{1}(\mathbf{d}_l \in \mathbf{D}_1)\|\mathbf{d}_l^{(1)} - \mathbf{m}_k^{(1)}\|_2^2 + \lambda \mathbf{1}(\mathbf{d}_l \in \mathbf{Y}_{S_1})\|\mathbf{d}_l^{(2)} - \mathbf{m}_k^{(2)}\|_2^2 \\
&+ \mathbf{1}(\mathbf{d}_l \in \mathbf{D}_2)\|\mathbf{d}_l^{(3)} - \mathbf{m}_k^{(3)}\|_2^2 + \lambda \mathbf{1}(\mathbf{d}_l \in \mathbf{Y}_{S_2})\|\mathbf{d}_l^{(4)} - \mathbf{m}_k^{(4)}\|_2^2,
\end{aligned} \tag{46}
$$

where $\mathbf{d}_l^{(1)}$, $\mathbf{d}_l^{(2)}$, $\mathbf{d}_l^{(3)}$ and $\mathbf{d}_l^{(4)}$ are with d_1, K, d_2, K dimensions, respectively. $\mathbf{m}_l^{(1)}$, $\mathbf{m}_l^{(2)}$, $\mathbf{m}_l^{(3)}$ and $\mathbf{m}_l^{(4)}$ will be calculated by the following Eq. (47).

Label Assignment.: Each data point is assigned to its nearest centroid according to the distance function in Eq. (46).

Centroid Updating.: The centroid updating rule is similar to the one in Eq. (40), where the auxiliary zeros do not contribute to the centroids.

$$
\begin{aligned}
\mathbf{m}_k^{(1)} &= \frac{\sum \mathbf{x}_l \in \mathscr{C}_k \bigcap \mathbf{D}_1 \mathbf{d}_l^{(1)}}{\mid \mathscr{C}_k \bigcap \mathbf{D}_1 \mid}, \ \mathbf{m}_k^{(2)} = \frac{\sum \mathbf{x}_l \in \mathscr{C}_k \bigcap \mathbf{Y}_{S_1} \mathbf{d}_l^{(2)}}{\mid \mathscr{C}_k \bigcap \mathbf{Y}_{S_1} \mid}, \\
\mathbf{m}_k^{(3)} &= \frac{\sum \mathbf{x}_l \in \mathscr{C}_k \bigcap \mathbf{D}_2 \mathbf{d}_l^{(3)}}{\mid \mathscr{C}_k \bigcap \mathbf{D}_2 \mid}, \ \mathbf{m}_k^{(4)} = \frac{\sum \mathbf{x}_l \in \mathscr{C}_k \bigcap \mathbf{Y}_{S_2} \mathbf{d}_l^{(4)}}{\mid \mathscr{C}_k \bigcap \mathbf{Y}_{S_2} \mid}.
\end{aligned} \tag{47}
$$

After transforming the utility function into the Frobenius norm by Lemma 4.7, a total of seven unknown variables require updating. Thanks to the K-means-like solution, the complex problem can be elegantly solved by two-phase iterative optimization. The convergence of the K-means-like solution for the problems in Eq. (42) and (44) is also guaranteed.

The problems demonstrated in Sections 4.4 and 4.5 are solved by making the learned partition consistent with the given partial partition. Here, we call them structure-preserved learning, which consists of the K-means for the core clustering and the utility function as a regularizer. Beyond the squared Euclidean distance for the K-means clustering and categorical utility function, the aforementioned findings also hold for P2C distance in Eq. (12) and the KCC utility function in Eq. (31) for various applications. To obtain a K-means formulation, we fill zeros into the concatenated matrix $\mathbf{D}$, which are not involved in the distance calculation nor do they contribute to the centroid computing.

4.6 Clustering with Outlier Removal

In this subsection, we review clustering with outlier removal [95], a joint clustering and outlier detection algorithm, where the original feature space is transformed into partition space via several basic partitions, and Holoentropy is employed to enhance the compactness of each cluster with outliers removed. This method introduces an auxiliary binary matrix to ensure the problem is solved by K-means−− [27].

Problem Definition.—Based on diverse assumptions, a number of unsupervised outlier detection methods have been proposed, including linear [141], [134], proximity [20], [149], [64], [51], and probability-based models [126]. Moreover, some studies pursue outlier detection by subspace learning [83], low-rank [184], matrix-completion [76], random walk

[121], and ensemble models [93], [93], [85]. More details on recent deep outlier detection can be found in this review [122].

As cluster analysis and outlier detection are closely coupled tasks [47], some work studies these two problems together [27], [163], [165]. Here, we review clustering with outlier removal (COR) [95], which jointly achieves cluster analysis and outlier detection, where o data points are detected as outliers, and the remainder is partitioned into K clusters. Generally speaking, the original space is transformed into a binary space by generating basic partitions to define clusters. Subsequently, a Holoentropy-based objective function [173] is employed to maximize the compactness of each cluster with a few outliers removed, which is efficiently solved by a unified K-means−−. In the following points, we present the objective function in [95] in terms of outlier detection, after which we demonstrate that only the partial problem can be easily formulated as the K-means optimization. Finally, by taking the original binary matrix and its counterpart matrix as the input, K-means−− is employed for the final solution.

First, the data matrix $\mathbf{X}$ is transformed in the original feature space into the binary matrix $\mathbf{B}$ in the partition level space with r basic partitions, where each partition has K_i clusters, $1 \le i \le r$. This process is similar to generating basic partitions in consensus clustering, which can be obtained by Eq. (27). Let $\mathcal{O}$ denote the set of o outliers. The goal of clustering with outlier removal is to maximize the compactness of each cluster in the partition level space with o outliers removed. We present the following objective function on the binary matrix $\mathbf{B} = \{\mathbf{b}_l \mid 1 \le l \le n\}$ as follows.

$$\min_{\mathcal{C}_k, \mathcal{O}} \sum_{k=1}^{K} p_{k+} HL(\mathcal{C}_k), \tag{48}$$

where $\mathcal{C}_k \cap \mathcal{C}_{k'} = \emptyset$ if $k \neq k'$ and $\mathcal{C}_1 \cup \cdots \cup \mathcal{C}_K = X \backslash \mathcal{O}$, $p_{k+} = |\mathcal{C}_k| / (n-o)$ and $HL(\cdot)$ denotes the Holoentropy on the categorical data [173], which is defined as the sum of the entropy and the total correlation of the random input. By the definition of Holoentropy, a detailed objective function is written as follows:

$$\begin{aligned} &\sum_{k=1}^{K} p_{k+} HL(\mathcal{C}_k) \propto \sum_{k=1}^{K} \sum_{\mathbf{b}_l \in \mathcal{C}_k} \sum_{i=1}^{r} \sum_{j=1}^{K_i} H(\mathcal{C}_{k,i,j}) \\ &= \sum_{k=1}^{K} \sum_{\mathbf{b}_l \in \mathcal{C}_k} \sum_{i=1}^{r} \sum_{j=1}^{K_i} \left(- p(\mathcal{C}_{k,i,j} = 0) \log p(\mathcal{C}_{k,i,j} = 0) \right. \\ &\left. - p(\mathcal{C}_{k,i,j} = 1) \log p(\mathcal{C}_{k,i,j} = 1) \right). \end{aligned} \tag{49}$$

Data Transformation.—The input is a set of basic partitions. Similar to consensus clustering, the binary matrix is derived from basic partitions in Eq. (27). When running K-means clustering on the binary matrix $\mathbf{B}$, the following lemma presents the centroids.

Lemma 4.10. For K-means clustering on the binary data set $\mathbf{B}$, the m-th centroid $\mathbf{m}_k$ satisfies

$$\begin{aligned}&\mathbf{m}_k = (\mathbf{m}_{k,1}, \cdots, \mathbf{m}_{k,i}, \cdots, \mathbf{m}_{k,r}), \text{ with}\\&\mathbf{m}_{k,i} = (m_{k,i,1}, \cdots, m_{k,i,j}, \cdots, m_{k,i,K_i}),\\&m_{k,i,K_i} = \sum_{b_l \in \mathscr{C}_k} b_{l,i,j} \,/\, |\,\mathscr{C}_k\,| = p(\mathscr{C}_{k,i,j} = 1)), \forall\, k, i, j\,.\end{aligned} \tag{50}$$

The k-th centroid of K-means on the binary matrix $\mathbf{B}$ is extactly the same with the $p(\mathscr{C}_{k,i,j} = 1)$. When replacing $p(\mathscr{C}_{k,i,j} = 1)$ in Eq. (49) with m_{k,i,K_i}, the second part is transformed into the objective function of a K-means clustering with entropy regarding on the centroids in Eq. (14). In the following points, we also demonstrate how the first part is transformed into the K-means framework. Note that for the binary matrix, $p(\mathscr{C}_{k,i,j} = 1) + p(\mathscr{C}_{k,i,j} = 0) = 1$. Therefore, we present another binary matrix $\widetilde{\mathbf{B}} = \{\widetilde{\mathbf{b}} \mid 1 \le l \le n\}$ as follows:

$$\begin{aligned}&\widetilde{\mathbf{b}}_l = (\widetilde{\mathbf{b}}_{l,1}, \cdots, \widetilde{\mathbf{b}}_{l,i}, \cdots, \widetilde{\mathbf{b}}_{l,r}), \text{ with}\\&\widetilde{\mathbf{b}}_{l,i} = (\widetilde{b}_{l,i,1}, \cdots, \widetilde{b}_{l,i,j}, \cdots, \widetilde{b}_{l,i,K_i}), \text{ and}\\&\widetilde{b}_{l,i,j} = \begin{cases} 0, \text{ if } L_{\pi_i}(\mathbf{x}_l) = j \\ 1, \text{ otherwise} \end{cases}.\end{aligned} \tag{51}$$

$\mathbf{B}$ and $\widetilde{\mathbf{B}}$ are the 1-of-K and $K-$1-of-K codings of the original data, respectively. $\widetilde{\mathbf{m}}_k$ is the centroid of $\widetilde{\mathbf{B}}$. Finally, we have the input for our generalized K-means as follows: $D = \left[\mathbf{B}\,\widetilde{\mathbf{B}}\right] \in \mathbf{R}^{n \times 2\sum_{i=1}^{r} K_i}$. With the data matrix $\mathbf{D}$, the following theorem is illustrated to solve this problem in Eq. (48) with K-means−−.

Theorem 4.11 ([95]). Given the data matrix $\mathbf{X}$, we generate several basic partitions $\boldsymbol{\pi}$ from $\mathbf{X}$ and transform them into binary matrices $\mathbf{B}$, $\widetilde{\mathbf{B}}$ by Eq. (27) and (51). Let $D = \left[\mathbf{B}\,\widetilde{\mathbf{B}}\right] = \{\mathbf{d}_l \mid 1 \le l \le n\}$, we have

$$\begin{aligned}&\min_{\mathscr{C}_k, \mathscr{O}} \sum_{k=1}^{K} p_{k+} HL(\mathscr{C}_k)\\&\Leftrightarrow \min_{\mathscr{C}_k, \mathscr{O}, \mathbf{m}_k, \widetilde{\mathbf{m}}_k} \sum_{k=1}^{K} \sum_{\mathbf{d}_l \in \mathscr{C}_k} f(\mathbf{d}_l, (\mathbf{m}_k, \widetilde{\mathbf{m}}_k)),\end{aligned} \tag{52}$$

where f is the distance function of the summation of KL-divergence on each dimension, and $\mathbf{m}_k$ and $\widetilde{\mathbf{m}}_k$ are the corresponding centroid by Eq. (50), which does not involve the outliers.

Distance Function.—According to Holoentropy, the summation of KL-divergence on each dimension calculates as the K-means distance.

Label Assignment.—Each data point is assigned to its nearest centroid according to the specified distance function. For this problem, however, some data points with large distances are regarded as outliers that are not assigned cluster labels. The non-exhaustive strategy in Eq. (15) applies to the assignment.

Centroid Updating.—The centroid is updated by the arithmetic average of each cluster (standard way). It is noteworthy that the outliers do not belong to any cluster and nor do they contribute to the centroid.

5 Experimental Results

5.1 Experimental Settings

Experimental problems.—In this section, we present experimental results to verify the effectiveness of the K-means solution in terms of consensus clustering, constrained clustering, image co-segmentation, domain adaptation, and outlier detection by different measurements. Iterative subspace projection and clustering is the cluster analysis problem, in essence, which is not included here due to that we focus on the non-clustering problems. We illustrate these experimental problems as follows:

- Consensus clustering. Given several basic partitions of the same data points as inputs, consensus clustering aims to fuse these partitions into one that is integrated.
- Constrained clustering. Constrained clustering seeks the intrinsic partition with the assistance of the partition level side information or pairwise must-link/ cannot-link constraints.
- Image co-segmentation. Given a collection of images containing similar objects, co-segmentation separates foreground objects from the background of each image.
- Unsupervised domain adaptation. Domain adaptation refers to the ability to apply an algorithm trained in one or more source domains to a different but related target domain. Here the “unsupervised” means that the source domain is associated with labels, while the target domain does not have annotated labels.
- Unsupervised outlier detection. An unsupervised outlier detection algorithm aims to detect a small portion of data points as outliers, which are different from other majority data points.

Data sets.—Several benchmark data sets are used to address the aforementioned experimental problems. They include *iris, wine, breast, ecoli, pendigits, satimage, yeast* from a UCI machine learning repository,[3] text data sets *cranmed, hitech, k*1*b, mm, cacmcisi, classic, la*12, *reviews, sports, fbis, re*1*, wap* from CLUTO package,[4] image classification data sets *caltech,*[5] image co-segmentation data sets *elephant, ferrari, gymnastics, kite, skating* [13], and domain adaptation data sets *Amazon*(*A*), *Caltech*(*C*), *Dsl*(*D*), *Webcam*(*W*).[6] For the outlier detection data sets, we treat the class with the smallest size as outliers.

3. https://archive.ics.uci.edu/ml/datasets.html
4. http://glaros.dtc.umn.edu/gkhome/cluto/cluto/download
5. http://www.vision.caltech.edu/Image_Datasets/Caltech101/
6. https://people.eecs.berkeley.edu/~jhoffman/domainadapt/

Competitive methods and implementation.—Next, we present the competitive methods according to different tasks.

- Consensus clustering. The Cluster-based Similarity Partitioning Algorithm (CSPA) [145] is pioneering work with graph partition along with consensus clustering. Hierarchical Consensus Clustering (HCC) [46] is the most representative of the link-based methods, which applies the agglomerative hierarchical clustering on the co-association matrix to find the consensus partition. Probability Trajectory Based Graph Partitioning (PTGP) [66] is based on the micro-cluster concept to summarize the basic partitions into a small core co-association matrix. To generate basic partitions, we apply K-means clustering with the cluster number sampled from $\left[K, \sqrt{n}\right]$ to generate 100 basic partitions, where K is the true clustering number. Here K-means-based Consensus Clustering (KCC) [169] is the K-means solution with entropy-based utility U_H, while Spectral Ensemble Clustering (SEC) [96] is the K-means solution with the co-association matrix.
- Constrained clustering. Constrained Non-negative Matrix Factorization (CNMF) [166] is a partition-level constrained clustering method based on NMF, while Linear-time Constrained Vector Quantization Error (LCVQE) [124] is a pair-wise constrained clustering method based on K-means. KCC with U_c combines the partition from the data and partition level side information for the final solution. PLCC [101] is the K-means solution for constrained clustering. Here we apply 50% partition level side information and transfer it to must-link and cannot-link for LCVQE.
- Image co-segmentation. Joulin [75], Vicente [157], and Rubio [132] are used for comparison, which directly take the images as input. For the K-means solution, Saliency Guided PLCC (SG-PLCC) [101] employs the cosine utility U_{cos}, where the unsupervised saliency prior [24] is obtained first as the partition level side information.
- Domain adaptation. For single-source domain adaptation, Transfer Component Analysis (TCA) [120], Transfer Subspace Learning (TSL) [142], and Joint Domain Adaptation (JDA) [107] are used for comparison. For multi-source domain adaptation, Robust visual Domain Adaptation with Low-rank Reconstruction (RDALR) [73] employs the low-rank construction and linear projection for the adaptation process; Fisher Discrimination Dictionary Learning (FDDL) [177] applies the fisher discrimination dictionary learning for sparse representation, and Shared Domain-adapted Dictionary Learning (SDDL) [138] employs the domain-adaptive dictionaries to learn the spare representation. Structured-Preserved Unsupervised Domain Adaptation (SP-UDA) [97], [99] is the K-means solution for this task, where the common space of source and target data is obtained by JDA [107].
- Outlier detection. Local Outlier Factor (LOF) [20], Fast Angle-Based Outlier Detector (FABOD) [126], and iForest [93] are three representative outlier

detection methods based on local density, angle, and random forest, respectively. Clustering with Outlier Removal (COR) [95] is the K-means solution for this task, which runs in the partition space. Similar to consensus clustering, we apply K-means clustering with the cluster number sampled from $[K, \sqrt{n}]$ to generate 100 basic partitions.

These tasks are all unsupervised, where the default parameters suggested by the authors are employed for fair comparison.

Metrics.—As labels are available for these data sets, external measurements are applied to evaluate the performance in terms of cluster validity and outlier detection.

Normalized Rand Index (R_n), Normalized Mutual Information (NMI), and $Accuracy$ are three widely used external measurements for cluster validity [171]. R_n measures the similarity between two partitions in a statistical manner; NMI measures mutual information between the resulting cluster and ground truth labels, followed by a normalization operation; $Accuracy$ comes from classification with the best mapping. These metrics can be computed as follows:

$$NMI = \frac{\sum_{i,j} n_{ij} \log \frac{n \cdot n_{ij}}{n_{i+} \cdot n_{+j}}}{\sqrt{(\sum_i n_{i+} \log \frac{n_{i+}}{n})(\sum_j n_{j+} \log \frac{n_{+j}}{n})}},$$

$$R_n = \frac{\sum_{i,j} \binom{n_{ij}}{2} - \sum_i \binom{n_{i+}}{2} \cdot \sum_j \binom{n_{+j}}{2} / \binom{n}{2}}{\sum_i \binom{n_{i+}}{2} / 2 + \sum_j \binom{n_{+j}}{2} / 2 - \sum_i \binom{n_{i+}}{2} \cdot \sum_j \binom{n_{+j}}{2} / \binom{n}{2}},$$

$$Accuracy = \sum_{i=1}^{n} \delta(s_i, map(r_i)) / n,$$

where $\delta(x, y)$ denotes the Kronecker delta function that equals to one if $x = y$; and equals to a zero, otherwise. $map(r_i)$ is the permutation mapping function that maps each cluster label r_i to the ground truth s_i. The best mapping is applied by the Kuhn-Munkres algorithms. n_{ij} is the number of co-occurrence instances in the cluster C_i and C_j of the obtained partition and ground truth, respectively. $n_{i+} = \sum_{j=1}^{K} n_{ij}$ and $n_{+j} = \sum_{i=1}^{K} n_{ij}$. Please refer to Table 3 for an in-depth understanding the meaning of n_{ij}, n_{i+}, and n_{+j}. Note that these four metrics are positive measurements, i.e., a larger value means better performance, whereas a negative R_n indicates a result poorer than random labeling.

Jaccard index is employed to evaluate the outlier detection, and can be computed as follows:

$$Jaccard = \frac{|O \bigcap O^*|}{|O \bigcup O^*|},$$

where O and O^* are the outlier sets by the algorithm and ground truth, respectively.

5.2 Discussions

The K-means solutions have simple mathematical formulation, efficient time complexity and deliver competitive performance in different tasks. Here we demonstrate the advantages of these K-means solutions in terms of effectiveness and efficiency.

Table 6 shows the experimental results of the K-means solution and other competitive methods in terms of consensus clustering, constrained clustering, image co-segmentation, domain adaptation, and outlier detection. For consensus clustering, the challenges predominantly lie in choosing the utility function and solving the combinational optimization efficiently. KCC and SEC are K-means or weighted K-means methods with a roughly linear time complexity to the number of instances and basic partitions. For cluster quality, CSPA and HCC obtain the best performance on *iris* and *k1b,* respectively; for other cases, KCC and SEC achieve the best results. It is noteworthy that HCC and PTGP deliver extremely worse partitions on *mm*. On the contrary, KCC and SEC have a robust performance on most data sets. For constrained clustering, PLCC, CNMF, and KCC are based on partition level side information, while LCVQE is based on pairwise constraints.

Although PLCC and LCVQE are K-means variants, the performance of PLCC outperforms LCVQE by a significant margin on *pendigits* and *satimages.* Compared with CNMF and KCC, PLCC remains competitive and delivers satisfactory results, indicating the utility function's effectiveness for structure-preserved learning. Based on PLCC, the authors extend constrained clustering for image co-segmentation. To obtain the partition level side information, they employ saliency prior to guide the image co-segmentation (SG-PLCC). From the experimental results, SG-PLCC gains improvements over other image co-segmentation methods. For domain adaptation, structure-preserved learning also assists in transfer learning. In some cases, SP-UDA employs the source structure to guide the target structure learning. Compared with the shared space learning methods, SP-UDA outperforms others in both single and multi-source domain adaptation. For outlier detection, in contrast to the methods, which are conducted in the original feature space, COR focuses on the partition space to better define the outliers. Compared with K-means−−, the benefits of COR result from the feature space transformation. Additional experimental results and impact factor analyses can be found in [169], [102], [101], [97], [95].

Beyond simplicity, another merit of K-means solutions based on Lloyd's algorithm is its high efficiency. Here we demonstrate the execution time of these K-means solutions in terms of consensus clustering, constrained clustering, and outlier detection in Table 7. For image co-segmentation and domain adaptation, there are different experimental settings or inputs compared with other competitive methods such that the execution time is not reported here. For consensus clustering, KCC and SEC are significantly faster than other methods, especially on large data sets. SEC is 18,000 times faster than HCC on *classic,* while KCC is 160 times faster than PTGP on *k1b.* For constrained clustering, PLCC is the fastest one among the competitive methods. It is noteworthy that LCVQE struggles when the number of pairwise constraints increases, where we employ 10% partition level side information in Table 7. For outlier detection, COR first transforms the original space into partition space by generating 100 basic partitions with binary codings and conducts joint clustering and outlier

detection. Although some time is required to generate 100 basic partitions, COR execution time is extremely fast, with no nearest neighbors or trees constructed.

In summary, K-means solutions have a simple, flexible, and elegant formulation and deliver promising results in terms of effectiveness and efficiency on several different tasks.

6 Conclusion

In this paper, we demonstrated how complex, challenging problems can be converted such that they can be solved by simple K-means optimization. Generally speaking, the objective functions and optimization algorithms for these problems were rewritten into a modified K-means version. In addition, we described how complex problems can be transformed into K-means by considering generalizing four aspects of a K-means formulation: data representation, distance function, non-exhaustive label assignment, and incomplete centroid updating. Finally, we illustrated how to convert and solve six applications, including iterative subspace projection and clustering, consensus clustering, constrained clustering, domain adaptation, and outlier detection.

Acknowledgement

This research is partially supported by NIH/NHLBI (U01HL089856).

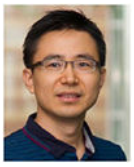
Biographies

Hongfu Liu received his bachelor and master degree in Management Information Systems from School of Economics and Management, Beihang University, in 2011 and 2014 respectively. He received the Ph.D. degree in computer engineering from Northeastern University, Boston MA, 2018. Currently he is a tenuretrack Assistant Professor affiliated with Michtom School of Computer Science at Brandeis University. His research interests generally focus on data mining and machine learning, with special interests in ensemble learning. He has served as the journal reviewers including TPAMI, TKDE, TNNLS, TIP, and TBD. He has also served on the program committee including KDD, AAAI and IJCAI. He is the Associate Editor of IEEE Computational Intelligence Magazine.

Junxiang Chen received his Bachelor of Engineering degree in Electronic and Information Engineering from Hong Kong Polytechnic University. He received the M.S. and Ph.D.

degrees in computer engineering from Northeastern University, Boston MA, in 2013 and 2018, respectively. Currently, he is a postdoctoral associate in the Department of Biomedical Informatics at University of Pittsburgh. His research interests generally focus on data mining and machine learning, and their application to biomedical imaging and health science, with particular interests in clustering and dimensionality reduction. He has served on the program committee for the conferences including AAAI, AISTATS, ICLR, ICML, NeurIPS, and UAI.

Jennifer Dy received the B.S. degree (magna cum laude) in electrical engineering from University of the Philippines, Quezon, Philippines, in 1993, and the M.S. and Ph.D. degrees from the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA, in 1997 and 2001, respectively. She joined the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA, USA, in 2002, where she is currently a Professor. Her current research interests include fundamental research in machine learning and data mining and their application to biomedical imaging, health, science, and engineering, with research contributions in clustering, multiple clustering, dimensionality reduction, feature selection and sparse methods, large margin classifiers, learning from crowds, and Bayesian nonparametric models. Dr. Dy was a recipient of the NSF Career Award in 2004. She served as an Action Editor/Editorial Board Member of Machine Learning, the Journal of Machine Learning Research, Organizing and/or Technical Program Committee Member of premier conferences in machine learning and data mining such as ICML, ACM SIGKDD, AAAI, IJCAI, AISTATS, SIAM SDM, and the Program Chair of SIAM SDM 2013.



Yun Fu (S’07-M’08-SM’11-F’19) received the B.Eng. degree in information engineering and the M.Eng. degree in pattern recognition and intelligence systems from Xian Jiaotong University, China, respectively, and the M.S. degree in statistics and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana Champaign, respectively. He is an interdisciplinary faculty member affiliated with College of Engineering and the College of Computer and Information Science at Northeastern University since 2012. He has extensive publications in leading journals, books/book chapters, and international conferences/workshops. He serves as associate editor, chairs, PC member and reviewer of many top journals and international conferences/workshops. He received seven Prestigious Young Investigator Awards from NAE, ONR, ARO, IEEE, INNS,

UIUC, Grainger Foundation; eleven Best Paper Awards from IEEE, ACM, IAPR, SPIE, SIAM; many major Industrial Research Awards from Google, Samsung, and Adobe, etc. He is currently an Associate Editor of IEEE T-IP. He is Member of Academia Europaea, Fellow of AAAS, IEEE, IAPR, OSA, SPIE, and AAIA, a lifetime Distinguished Member of ACM, Lifetime Senior Member of AAAI, and Institute of Mathematical Statistics, member of AAAS, ACM Future of Computing Academy, Global Young Academy, INNS and Beckman Graduate Fellow during 2007-2008.

References

[1]. Aghabozorgi S, Shirkhorshidi AS, and Wah TY. Time-series clustering–a decade review. Information Systems, 53:16–38, 2015.

[2]. Agrawal A and Gupta H. Global k-means (gkm) clustering algorithm: a survey. International Journal of Computer Applications, 79(2), 2013.

[3]. Ahmed M, Seraj R, and Islam SMS. The k-means algorithm: a comprehensive survey and performance evaluation. Electronics, 9(8):1295, 2020.

[4]. Aloise D, Deshpande A, Hansen P, and Popat P. Np-hardness of euclidean sum-of-squares clustering. Machine Learning, 75(2):245–248, 2009.

[5]. Arthur D and Vassilvitskii S. How slow is the k-means method? In Proceedings of Symposium on Computational Geometry, 2006.

[6]. Arthur D and Vassilvitskii S. k-means++: The advantages of careful seeding. In Proceedings of ACM-SIAM Symposium on Discrete Algorithms, 2007.

[7]. Bachem O, Lucic M, Hassani H, and Krause A. Fast and provably good seedings for k-means. Advances in Neural Information Processing Systems, 29:55–63, 2016.

[8]. Bachem O, Lucic M, Hassani SH, and Krause A. Approximate k-means++ in sublinear time. In Proceedings of AAAI Conference on Artificial Intelligence, 2016.

[9]. Balakrishnama S and Ganapathiraju A. Linear discriminant analysis-a brief tutorial. Institute for Signal and Information Processing, 18(1998):1–8, 1998.

[10]. Banerjee A, Merugu S, Dhillon I, and Ghosh J. Clustering with bregman divergences. Journal of Machine Learning Research, 6:1705–1749, 2005.

[11]. Basu S, Banerjee A, and Mooney RJ. Active semi-supervision for pairwise constrained clustering. In Proceedings of SIAM International Conference on Data Mining, pages 333–344, 2004.

[12]. Basu S, Davidson I, and Wagstaff K. Constrained clustering: Advances in algorithms, theory, and applications. CRC Press, 2008.

[13]. Batra D, Kowdle A, Parikh D, Luo J, and Chen T. Interactively co-segmentating topically related images with intelligent scribble guidance. International Journal of Computer Vision, 93(3):273–292, 2011.

[14]. Belkin M and Niyogi P. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation, 15(6):1373–1396, 2003.

[15]. Bengio Y, Lamblin P, Popovici D, and Larochelle H. Greedy layer-wise training of deep networks. In Advances in Neural Information Processing Systems, pages 153–160, 2007.

[16]. Bezdek JC, Ehrlich R, and Full W. Fcm: The fuzzy c-means clustering algorithm. Computers & Geosciences, 10(2-3):191–203, 1984.

[17]. Blömer J, Lammersen C, Schmidt M, and Sohler C. Theoretical analysis of the k-means algorithm–a survey. In Algorithm Engineering, pages 81–116. Springer, 2016.

[18]. Bottou L and Bengio Y. Convergence properties of the k-means algorithms. In Advances in Neural Information Processing Systems, 1995.

[19]. Bradley PS and Fayyad UM. Refining initial points for k-means clustering. In Proceedings of International Conference on Machine Learning, volume 98, pages 91–99. Citeseer, 1998.

[20]. Breunig MM, Kriegel H-P, Ng RT, and Sander J. Lof: identifying density-based local outliers. In Proceedings of ACM Sigmod Record, volume 29. ACM, 2000.

[21]. Buzo A, Gray A, Gray R, and Markel J. Speech coding based upon vector quantization. IEEE Transactions on Acoustics, Speech, and Signal Processing, 28(5):562–574, 1980.

[22]. Cai H, Zheng VW, and Chang KC-C. A comprehensive survey of graph embedding: Problems, techniques, and applications. IEEE Transactions on Knowledge and Data Engineering, 30(9):1616–1637, 2018.

[23]. Cai X, Nie F, and Huang H. Multi-view k-means clustering on big data. In Proceedings of International Joint Conference on Artificial Intelligence, 2013.

[24]. Cao X, Tao Z, Zhang B, Fu H, and Feng W. Self-adaptively weighted co-saliency detection via rank constraint. IEEE Transactions on Image Processing, 23(9):4175–4186, 2014. [PubMed: 24968170]

[25]. Celebi ME, Kingravi HA, and Vela PA. A comparative study of efficient initialization methods for the k-means clustering algorithm. Expert Systems with Applications, 40(1):200–210, 2013.

[26]. Chaturvedi A, Green PE, and Caroll JD. K-modes clustering. Journal of classification, 18(1):35–55, 2001.

[27]. Chawla S and Gionis A. k-means−−: A unified approach to clustering and outlier detection. In Proceedings of SIAM International Conference on Data Mining, 2013.

[28]. Chen M, Xu Z, Weinberger K, and Sha F. Marginalized denoising autoencoders for domain adaptation. arXiv preprint arXiv:1206.4683, 2012.

[29]. Chitta R, Jin R, Havens TC, and Jain AK. Approximate kernel k-means: Solution to large scale kernel clustering. In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 895–903, 2011.

[30]. Choudhary R and Shukla S. A clustering based ensemble of weighted kernelized extreme learning machine for class imbalance learning. Expert Systems with Applications, 164:114041, 2021.

[31]. Cicek S and Soatto S. Unsupervised domain adaptation via regularized conditional alignment. In Proceedings of IEEE/CVF International Conference on Computer Vision, pages 1416–1425, 2019.

[32]. Damodaran BB, Kellenberger B, Flamary R, Tuia D, and Courty N. Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation. In Proceedings of European Conference on Computer Vision, pages 447–463, 2018.

[33]. Dasgupta S and Freund Y. Random projection trees for vector quantization. IEEE Transactions on Information Theory, 55(7):3229–3242, 2009.

[34]. Datta S, Giannella C, and Kargupta H. Approximate distributed k-means clustering over a peer-to-peer network. IEEE Transactions on Knowledge and Data Engineering, 21(10):1372–1388, 2008.

[35]. de Amorim RC. A survey on feature weighting based k-means algorithms. Journal of Classification, 33(2):210–242, 2016.

[36]. De la Torre F and Kanade T. Discriminative cluster analysis. In Proceedings of International Conference on Machine Learning, pages 241–248, 2006.

[37]. Dhillon I, Guan Y, and Kulis B. Kernel k-means: Spectral clustering and normalized cuts. In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004.

[38]. Dhillon IS, Guan Y, and Kulis B. Kernel k-means: spectral clustering and normalized cuts. In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 551–556, 2004.

[39]. Dhillon IS, Mallela S, and Kumar R. A divisive information theoretic feature clustering algorithm for text classification. Journal of machine learning research, 3:1265–1287, 2003.

[40]. Ding C and He X. K-means clustering via principal component analysis. In Proceedings of International Conference on Machine Learning, 2004.

[41]. Ding C, He X, and Simon H. On the equivalence of nonnegative matrix factorization and spectral clustering. In Proceedings of SIAM International Conference on Data Mining, 2005.

[42]. Ding C and Li T. Adaptive dimension reduction using discriminant analysis and k-means clustering. In Proceedings of International Conference on Machine Learning, pages 521–528, 2007.

[43]. Elkan C. Using the triangle inequality to accelerate k-means. In Proceedings of International Conference on Machine Learning, pages 147–153, 2003.

[44]. Fern XZ and Brodley CE. Solving cluster ensemble problems by bipartite graph partitioning. In Proceedings of International Conference on Machine Learning, 2004.

[45]. Fodor IK. A survey of dimension reduction techniques. Technical report, Lawrence Livermore National Lab., CA (US), 2002.

[46]. Fred A and Jain A. Combining multiple clusterings using evidence accumulation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(6):835–850, 2005. [PubMed: 15943417]

[47]. Gan G and Ng MK-P. K-means clustering with outlier removal. Pattern Recognition Letters, 90:8–14, 2017.

[48]. Ganin Y, Ustinova E, Ajakan H, Germain P, Larochelle H, Laviolette F, Marchand M, and Lempitsky V. Domain-adversarial training of neural networks. Journal of Machine Learning Research, 17(1):2096–2030, 2016.

[49]. Ghuli P, Prabhakar M, and Shettar R. A comprehensive survey on centroid selection strategies for distributed k-means clustering algorithm. International Journal of Computer Applications, 125(5), 2015.

[50]. Golay X, Kollias S, Stoll G, Meier D, Valavanis A, and Boesiger P. A new correlation-based fuzzy logic clustering algorithm for fmri. Magnetic resonance in medicine, 40(2):249–260, 1998. [PubMed: 9702707]

[51]. Goldstein M and Dengel A. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. KI-2012: Poster and Demo Track, pages 59–63, 2012.

[52]. Goyal P and Ferrara E. Graph embedding techniques, applications, and performance: A survey. Knowledge-Based Systems, 151:78–94, 2018.

[53]. Guo C, Jia H, and Zhang N. Time series clustering based on ica for stock data analysis. In International Conference on Wireless Communications, Networking and Mobile Computing, 2008.

[54]. Guo R, Sun P, Lindgren E, Geng Q, Simcha D, Chern F, and Kumar S. Accelerating large-scale inference with anisotropic vector quantization. In Proceedings of International Conference on Machine Learning, pages 3887–3896. PMLR, 2020.

[55]. Hamerly G. Making k-means even faster. In Proceedings of SIAM International Conference on Data Mining, pages 130–140, 2010.

[56]. Hamerly G and Drake J. Accelerating lloyd's algorithm for k-means clustering. In Partitional clustering algorithms. 2015.

[57]. Hamerly G and Elkan C. Alternatives to the k-means algorithm that find better clusterings. In Proceedings of International Conference on Information and Knowledge Management, pages 600–607, 2002.

[58]. Hamerly G and Elkan C. Learning the k in k-means. Advances in Neural Information Processing Systems, 16:281–288, 2003.

[59]. Hartigan JA. Clustering algorithms. John Wiley & Sons, Inc., 1975.

[60]. Hartigan JA and Wong MA. Algorithm as 136: A k-means clustering algorithm. Journal of Royal Statistical Society. Series C (Applied Statistics), 28(1):100–108, 1979.

[61]. Hautamaki V, Nykanen P, and Franti P. Time-series clustering by approximate prototypes. In 2008 19th International conference on pattern recognition, pages 1–4. IEEE, 2008.

[62]. He L and Zhang H. Kernel k-means sampling for nyström approximation. IEEE Transactions on Image Processing, 27(5):2108–2120, 2018. [PubMed: 29432094]

[63]. He X, Ding CH, Zha H, and Simon HD. Automatic topic identification using webpage clustering. In Proceedings of IEEE International Conference on Data Mining, pages 195–202, 2001.

[64]. He Z, Xu X, and Deng S. Discovering cluster-based local outliers. Pattern Recognition Letters, 24(9-10):1641–1650, 2003.

[65]. Holmes M, Gray A, and Isbell C. Fast svd for large-scale matrices. In Workshop on Efficient Machine Learning at NIPS, volume 58, pages 249–252, 2007.

[66]. Huang D, Lai J-H, and Wang C-D. Robust ensemble clustering using probability trajectories. IEEE Transactions on Knowledge and Data Engineering, 28(5):1312–1326, 2016.

[67]. Huang Z. Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery, 2(3):283–304, 1998.

[68]. Hunt L and Jorgensen M. Clustering mixed data. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 1(4):352–361, 2011.

[69]. Jagannathan G and Wright RN. Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pages 593–599, 2005.

[70]. Jain A. Data clustering: 50 years beyond k-means. Pattern recognition letters, 31(8):651–666, 2010.

[71]. Jain AK and Dubes RC. Algorithms for clustering data. Prentice-Hall, Inc., 1988.

[72]. Jamel AA and Akay B. A survey and systematic categorization of parallel k-means and fuzzy-c-means algorithms. Comput. Syst. Sci. Eng, 34(5):259–281, 2019.

[73]. Jhuo I-H, Liu D, Lee D, and Chang S-F. Robust visual domain adaptation with low-rank reconstruction. In Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2012.

[74]. Jolliffe I. Principal component analysis. Encyclopedia of statistics in behavioral science, 2005.

[75]. Joulin A, Bach F, and Ponce J. Discriminative clustering for image co-segmentation. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2010.

[76]. Kannan R, Woo H, Aggarwal CC, and Park H. Outlier detection for text data. In Proceedings of SIAM International Conference on Data Mining, pages 489–497, 2017.

[77]. Kaufman L and Rousseeuw PJ. Partitioning around medoids (program pam). Finding groups in data: an introduction to cluster analysis, 344:68–125, 1990.

[78]. Khan SS and Ahmad A. Cluster center initialization algorithm for k-means clustering. Pattern Recognition Letters, 25(11):1293–1302, 2004.

[79]. Khandare A and Alvi A. Survey of improved k-means clustering algorithms: improvements, shortcomings and scope for further enhancement and scalability. In Information Systems Design and Intelligent Applications, pages 495–503. Springer, 2016.

[80]. Kodinariya TM and Makwana PR. Review on determining number of cluster in k-means clustering. International Journal, 1(6):90–95, 2013.

[81]. Korn F, Jagadish HV, and Faloutsos C. Efficiently supporting ad hoc queries in large datasets of time sequences. Acm Sigmod Record, 26(2):289–300, 1997.

[82]. Kövesi B, Boucher J-M, and Saoudi S. Stochastic k-means algorithm for vector quantization. Pattern Recognition Letters, 22(6-7):603–610, 2001.

[83]. Kriegel H-P, Kröger P, Schubert E, and Zimek A. Outlier detection in axis-parallel subspaces of high dimensional data. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 831–838. Springer, 2009.

[84]. Kulis B and Jordan MI. Revisiting k-means: New algorithms via bayesian nonparametrics. In Proceedings of International Conference on Machine Learning, 2012.

[85]. Lazarevic A and Kumar V. Feature bagging for outlier detection. In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pages 157–166, 2005.

[86]. Lee C-Y, Batra T, Baig MH, and Ulbricht D. Sliced wasserstein discrepancy for unsupervised domain adaptation. In Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10285–10295, 2019.

[87]. Li T, Ding C, and Jordan MI. Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization. In Proceedings of IEEE International Conference on Data Mining, pages 577–582, 2007.

[88]. Li T, Ma S, and Ogihara M. Document clustering via adaptive subspace iteration. In Proceedings of Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 218–225, 2004.

[89]. Li Z, Liu J, and Tang X. Constrained clustering via spectral regularization. In Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 421–428, 2009.

[90]. Liao T, Bolt B, Forester J, Hailman E, Hansen C, Kaste R, and O'May J. Understanding and projecting the battle state. In 23rd Army Science Conference, Orlando, FL, volume 25, 2002.

[91]. Likas A, Vlassis N, and Verbeek J. The global k-means clustering algorithm. Pattern Recognition, 36(2):451–461, 2003.

[92]. Linde Y, Buzo A, and Gray R. An algorithm for vector quantizer design. IEEE Transactions on Communications, 28(1):84–95, 1980.

[93]. Liu FT, Ting KM, and Zhou Z-H. Isolation forest. In Proceedings of IEEE International Conference on Data Mining, 2008.

[94]. Liu H and Fu Y. Clustering with partition level side information. In Proceedings of International Conference on Data Mining, 2015.

[95]. Liu H, Li J, Wu Y, and Fu Y. Clustering with outlier removal. IEEE Transactions on Knowledge and Data Engineering, 33(6):2369–2379, 2021.

[96]. Liu H, Liu T, Wu J, Tao D, and Fu Y. Spectral ensemble clustering. In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015.

[97]. Liu H, Shao M, Ding Z, and Fu Y. Structure-preserved unsupervised domain adaptation. IEEE Transactions on Knowledge and Data Engineering, 31(4):799–812, 2018.

[98]. Liu H, Shao M, and Fu Y. Consensus guided unsupervised feature selection. In Proceedings of AAAI Conference on Artificial Intelligence, 2016.

[99]. Liu H, Shao M, and Fu Y. Structure-preserved multi-source domain adaptation. In Proceedings of International Conference on Data Mining, 2016.

[100]. Liu H, Tao Z, and Ding Z. Consensus clustering: An embedding perspective, extension and beyond. arXiv preprint arXiv:1906.00120, 2019.

[101]. Liu H, Tao Z, and Fu Y. Partition level constrained clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(10):2469–2483, 2017. [PubMed: 29053445]

[102]. Liu H, Wu J, Liu T, Tao D, and Fu Y. Spectral ensemble clustering via weighted k-means: Theoretical and practical evidence. IEEE Transactions on Knowledge and Data Engineering, 29(5):1129–1143, 2017.

[103]. Liu X, Hu B, Liu X, Lu J, You J, and Kong L. Energy-constrained self-training for unsupervised domain adaptation. In Proceedings of International Conference on Pattern Recognition, pages 7515–7520, 2021.

[104]. Liu X, Xing F, Stone M, Zhuo J, Reese T, Prince JL, El Fakhri G, and Woo J. Generative self-training for cross-domain unsupervised tagged-to-cine mri synthesis. In Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention, pages 138–148, 2021.

[105]. Lloyd S. Least squares quantization in pcm. IEEE Transactions on Information Theory, 28(2):129–137, 1982.

[106]. Long M, Cao Y, Wang J, and Jordan M. Learning transferable features with deep adaptation networks. In Proceedings of International Conference on Machine Learning, pages 97–105, 2015.

[107]. Long M, Wang J, Ding G, Sun J, and Philip SY. Transfer feature learning with joint distribution adaptation. In Proceedings of IEEE International Conference on Computer Vision, 2013.

[108]. Long M, Zhu H, Wang J, and Jordan MI. Unsupervised domain adaptation with residual transfer networks. Advances in Neural Information Processing Systems, 29:136–144, 2016.

[109]. Lourenco A, Bulò S, Rebagliati N, Fred A, Figueiredo M, and Pelillo M. Probabilistic consensus clustering using evidence accumulation. Machine Learning, 98(1-2):331–357, 2015.

[110]. Lu Z, Peng Y, and Xiao J. From comparing clusterings to combining clusterings. In Proceedings of AAAI Conference on Artificial Intelligence, pages 665–670, 2008.

[111]. MacQueen J et al. Some methods for classification and analysis of multivariate observations. In Proceedings of Berkeley Symposium on Mathematical Statistics and Probability, 1967.

[112]. Mahajan GS and Bhagat KS. Survey on medical image segmentation using enhanced k-means and kernelized fuzzy c-means. International Journal of Advances in Engineering & Technology, 6(6):2531, 2014.

[113]. Manthey B and Röglin H. Worst-case and smoothed analysis of k-means clustering with bregman divergences. In International Symposium on Algorithms and Computation, 2009.

[114]. Melnykov I and Melnykov V. On k-means algorithm with the use of mahalanobis distances. Statistics & Probability Letters, 84:88–95, 2014.

[115]. Mirkin B. Reinterpreting the category utility function. Machine Learning, 45(2):219–228, November 2001.

[116]. Monti S, Tamayo P, Mesirov J, and Golub T. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. Machine Learning, 52(1-2):91–118, 2003.

[117]. Nassar CR. Telecommunications demystified. Elsevier, 2013.

[118]. Ng AY, Jordan MI, and Weiss Y. On spectral clustering: Analysis and an algorithm. In Advances in neural information processing systems, pages 849–856, 2002.

[119]. Nguyen T, Le T, Dam N, Tran QH, Nguyen T, and Phung D. Tidot: A teacher imitation learning approach for domain adaptation with optimal transport. In Proceedings of International Joint Conference on Artificial Intelligence, pages 2862–2868, 2021.

[120]. Pan SJ, Tsang IW, Kwok JT, and Yang Q. Domain adaptation via transfer component analysis. IEEE Transactions on Neural Networks, 22(2):199–210, 2011. [PubMed: 21095864]

[121]. Pang G, Cao L, and Chen L. Outlier detection in complex categorical data by modelling the feature value couplings. In Proceedings of International Joint Conference on Artificial Intelligence, 2016.

[122]. Pang G, Shen C, Cao L, and Hengel AVD. Deep learning for anomaly detection: A review. ACM Computing Surveys, 54(2):1–38, 2021.

[123]. Parsons L, Haque E, and Liu H. Subspace clustering for high dimensional data: a review. Acm Sigkdd Explorations Newsletter, 6(1):90–105, 2004.

[124]. Pelleg D and Baras D. K-means with large and noisy constraint sets. In Proceedings of European Conference on Machine Learning, 2007.

[125]. Pelleg D, Moore AW, et al. X-means: Extending k-means with efficient estimation of the number of clusters. In Proceedings of International Conference on Machine Learning, 2000.

[126]. Pham N and Pagh R. A near-linear time approximation algorithm for angle-based outlier detection in high-dimensional data. In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data mining, 2012.

[127]. Phillips SJ. Acceleration of k-means and related clustering algorithms. In Workshop on Algorithm Engineering and Experimentation, pages 166–177, 2002.

[128]. Policker S and Geva AB. Nonstationary time series analysis by temporal clustering. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 30(2):339–343, 2000.

[129]. Pugazhenthi A and Kumar LS. Selection of optimal number of clusters and centroids for k-means and fuzzy c-means clustering: A review. In Proceedings of International Conference on Computing, Communication and Security, pages 1–4, 2020.

[130]. Rani S and Sikka G. Recent techniques of clustering of time series data: a survey. International Journal of Computer Applications, 52(15), 2012.

[131]. Roweis ST and Saul LK. Nonlinear dimensionality reduction by locally linear embedding. Science, 290(5500):2323–2326, 2000. [PubMed: 11125150]

[132]. Rubio JC, Serrat J, López A, and Paragios N. Unsupervised co-segmentation through region matching. In Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2012.

[133]. Saito K, Watanabe K, Ushiku Y, and Harada T. Maximum classifier discrepancy for unsupervised domain adaptation. In Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 3723–3732, 2018.

[134]. Schölkopf B, Platt JC, Shawe-Taylor J, Smola AJ, and Williamson RC. Estimating the support of a high-dimensional distribution. Neural Computation, 13(7):1443–1471, 2001. [PubMed: 11440593]

[135]. Schölkopf B, Smola A, and Müller K-R. Nonlinear component analysis as a kernel eigenvalue problem. Neural computation, 10(5):1299–1319, 1998.

[136]. Schölkopf B, Smola AJ, Bach F, et al. Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press, 2002.

[137]. Schütze H, Manning CD, and Raghavan P. Introduction to information retrieval, volume 39. Cambridge University Press Cambridge, 2008.

[138]. Shekhar S, Patel VM, Nguyen HV, and Chellappa R. Generalized domain-adaptive dictionaries. In Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2013.

[139]. Shen J, Qu Y, Zhang W, and Yu Y. Wasserstein distance guided representation learning for domain adaptation. In Proceedings of AAAI Conference on Artificial Intelligence, 2018.

[140]. Shi J and Malik J. Normalized cuts and image segmentation. IEEE Transactions on pattern analysis and machine intelligence, 22(8):888–905, 2000.

[141]. Shyu M-L, Chen S-C, Sarinnapakorn K, and Chang L. A novel anomaly detection scheme based on principal component classifier. Technical report, University of Miami, 2003.

[142]. Si S, Tao D, and Geng B. Bregman divergence-based regularization for transfer subspace learning. IEEE Transactions on Knowledge and Data Engineering, 22(7):929–942, 2010.

[143]. Slonim N, Aharoni E, and Crammer K. Hartigan's k-means versus lloyd's k-means—is it time for a change? In Proceedings of International Joint Conference on Artificial Intelligence, 2013.

[144]. Śmieja M and Geiger BC. Semi-supervised cross-entropy clustering with information bottleneck constraint. Information Sciences, 421:254–271, 2017.

[145]. Strehl A and Ghosh J. Cluster ensembles — a knowledge reuse framework for combining partitions. Journal of Machine Learning Research, 3:583–617, 2002.

[146]. Sun B and Saenko K. Deep coral: Correlation alignment for deep domain adaptation. In Proceedings of European conference on computer vision, pages 443–450, 2016.

[147]. Tan P-N, Steinbach M, and Kumar V. Introduction to data mining. Pearson Education India, 2016.

[148]. Tang C and Monteleoni C. Convergence rate of stochastic k-means. In Artificial Intelligence and Statistics, pages 1495–1503. PMLR, 2017.

[149]. Tang J, Chen Z, Fu AW-C, and Cheung DW. Enhancing effectiveness of outlier detections for low density patterns. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 535–548. Springer, 2002.

[150]. Tao Z, Liu H, Fu Z, and Fu Y. Image co-segmentation via saliency-guided constraint clustering with cosine similarity. In Proceedings of AAAI Conference on Artificial Intelligence, 2017.

[151]. Topchy A, Jain AK, and Punch W. Combining multiple weak clusterings. In Proceedings of IEEE International Conference on Data Mining, pages 331–338. IEEE, 2003.

[152]. Topchy A, Jain AK, and Punch W. A mixture model for clustering ensembles. In Proceedings of SIAM International Conference on Data Mining, pages 379–390. SIAM, 2004.

[153]. Tzeng E, Hoffman J, Saenko K, and Darrell T. Adversarial discriminative domain adaptation. In Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7167–7176, 2017.

[154]. Tzeng E, Hoffman J, Zhang N, Saenko K, and Darrell T. Deep domain confusion: Maximizing for domain invariance. arXiv preprint arXiv:1412.3474, 2014.

[155]. Ulutagay G and Nasibov E. Fuzzy and crisp clustering methods based on the neighborhood concept: A comprehensive review. Journal of Intelligent & Fuzzy Systems, 23(6):271–281, 2012.

[156]. Vert J-P, Tsuda K, and Schölkopf B. A primer on kernel methods. Kernel Methods in Computational Biology, 47:35–70, 2004.

[157]. Vicente S, Rother C, and Kolmogorov V. Object co-segmentation. In Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2011.

[158]. Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol P-A, and Bottou L. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research, 11(12), 2010.

[159]. Vlachos M, Lin J, Keogh E, and Gunopulos D. A wavelet-based anytime algorithm for k-means clustering of time series. In In proc. workshop on clustering high dimensionality data and its applications. Citeseer, 2003.

[160]. Von Luxburg U. A tutorial on spectral clustering. Statistics and computing, 17(4):395–416, 2007.

[161]. Wang F, Ding C, and Li T. Integrated kl (k-means–laplacian) clustering: a new clustering approach by combining attribute data and pairwise relations. In Proceedings of SIAM International Conference on Data Mining, pages 38–48. SIAM, 2009.

[162]. Wang S, Gittens A, and Mahoney MW. Scalable kernel k-means clustering with nyström approximation: relative-error bounds. The Journal of Machine Learning Research, 20(1):431–479, 2019.

[163]. Whang J, Dhillon I, and Gleich D. Non-exhaustive, overlapping k-means. In Proceedings of SIAM International Conference on Data Mining, 2015.

[164]. Whang JJ, Du R, Jung S, Lee G, Drake B, Liu Q, Kang S, and Park H. Mega: multi-view semi-supervised clustering of hypergraphs. Proceedings of the VLDB Endowment, 13(5):698–711, 2020.

[165]. Whang JJ, Hou Y, Gleich DF, and Dhillon IS. Non-exhaustive, overlapping clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(11):2644–2659, 2018. [PubMed: 30080141]

[166]. Wu H and Liu Z. Non-negative matrix factorization with constraints. In Proceedings of AAAI Conference on Artificial Intelligence, 2010.

[167]. Wu J. Advances in K-means clustering: a data mining thinking. Springer Science & Business Media, 2012.

[168]. Wu J, Liu H, Xiong H, and Cao J. A theoretic framework of k-means-based consensus clustering. In Proceedings of International Joint Conference on Artificial Intelligence, 2013.

[169]. Wu J, Liu H, Xiong H, Cao J, and Chen J. K-means-based consensus clustering: A unified view. IEEE Transactions on Knowledge and Data Engineering, 27(1):155–169, 2015.

[170]. Wu J, Wu Z, Cao J, Liu H, Chen G, and Zhang Y. Fuzzy consensus clustering with applications on big data. IEEE Transactions on Fuzzy Systems, 25(6):1430–1445, 2017.

[171]. Wu J, Xiong H, and Chen J. Adapting the right measures for k-means clustering. In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009.

[172]. Wu J, Xiong H, Liu C, and Chen J. A generalization of distance functions for fuzzy-means clustering with centroids of arithmetic means. IEEE Transactions on Fuzzy Systems, 20(3):557–571, 2012.

[173]. Wu S and Wang S. Information-theoretic outlier detection for large-scale categorical data. IEEE Transactions on Knowledge and Data Engineering, 25(3):589–602, 2013.

[174]. Xia S, Peng D, Meng D, Zhang C, Wang G, Giem E, Wei W, and Chen Z. A fast adaptive k-means with no bounds. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020.

[175]. Xie J, Girshick R, and Farhadi A. Unsupervised deep embedding for clustering analysis. In Proceedings of International Conference on Machine Learning, pages 478–487. PMLR, 2016.

[176]. Xie S, Gao J, Fan W, Turaga D, and Yu PS. Class-distribution regularized consensus maximization for alleviating overfitting in model combination. In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 303–312, 2014.

[177]. Yang M, Zhang L, Feng X, and Zhang D. Fisher discrimination dictionary learning for sparse representation. In Proceedings of IEEE/CVF International Conference on Computer Vision, 2011.

[178]. Ye J, Zhao Z, and Liu H. Adaptive distance metric learning for clustering. In Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1–7. IEEE, 2007.

[179]. Ye J, Zhao Z, and Wu M. Discriminative k-means for clustering. Advances in Neural Information Processing Systems, 20:1649–1656, 2007.

[180]. Yiu ML and Mamoulis N. Iterative projected clustering by subspace mining. IEEE Transactions on Knowledge and Data Engineering, 17(2):176–189, 2005.

[181]. Yu W, Ding Z, Hu C, and Liu H. Knowledge reused outlier detection. IEEE Access, 7:43763–43772, 2019.

[182]. Zhang Y, Liu T, Long M, and Jordan M. Bridging theory and algorithm for domain adaptation. In Proceedings of International Conference on Machine Learning, pages 7404–7413. PMLR, 2019.

[183]. Zhang Y, Tang H, Jia K, and Tan M. Domain-symmetric networks for adversarial domain adaptation. In Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019.

[184]. Zhao H, Liu H, Ding Z, and Fu Y. Consensus regularized multi-view outlier detection. IEEE Transactions on Image Processing, 27(1):236–248, 2017. [PubMed: 28945594]

[185]. Zhexue Huang JC and Ng MK. A note of k-modes clustering. Journal of Classification, 20(2), 2003.

[186]. Zou Y, Yu Z, Liu X, Kumar BV, and Wang J. Confidence regularized self-training. In Proceedings of IEEE International Conference on Computer Vision, 2019.

TABLE 1

Instances of Bregman Divergence and Point-to-Centroid Distance

Distance	Domain	$\phi(\mathbf{x})$	$f(\mathbf{x}, \mathbf{y})$
Logistic loss	$\{0, 1\}$	$x \log x$	$x \log(\frac{x}{y}) - (x - y)$
Itakura-Saito distance	$\mathbf{R}_{++}$	$-\log x$	$\frac{x}{y} - \log(\frac{x}{y}) - 1$
Squared Euclidean distance	$\mathbf{R}^d$	$\|\mathbf{x}\|_2^2$	$\|\mathbf{x} - \mathbf{y}\|_2^2$
Mahalanobis distance	$\mathbf{R}^d$	$\mathbf{xAx}$	$(\mathbf{x} - \mathbf{y})\mathbf{A}(\mathbf{x} - \mathbf{y})^\intercal$
KL-divergence	d-Simplex	$\sum_{j=1}^d x_j \log x_j$	$\sum_{j=1}^d x_j \log(\frac{x_j}{y_j})$
Generalized I-divergence	$\mathbf{R}_+^d$	$\sum_{j=1}^d x_j \log x_j$	$\sum_{j=1}^d x_j \log(\frac{x_j}{y_j}) - \sum_{j=1}^d (x_j - y_j)$
Cosine similarity	$\mathbf{R}^d$	$\|\mathbf{x}\|_2$	$\|\mathbf{x}\|_2 - \sum_{j=1}^d x_j y_j \,/\, \|\mathbf{y}\|_2$

Note: $\mathbf{A}$ is a $d \times d$ inverse of the covariance matrix.

TABLE 2

Six Applications with K-means solutions

Application	Data transformation	Distance function	Label assignment	Centroid updating
Iterative subspace projection and clustering [179]	✓			
K-means-based consensus clustering [151], [168], [169]	✓	✓		
Spectral ensemble clustering [96], [102]	✓			✓
Partition level constrained clustering [94], [101]	✓	✓		✓
Structure-preserved domain adaptation [99], [97]	✓	✓		✓
Clustering with outlier removal [95]	✓	✓	✓	

TABLE 3

The Contingency Matrix

$\boldsymbol{\pi}$		$\boldsymbol{\pi}_i$				
		$\mathscr{C}_1^{(i)}$	$\mathscr{C}_2^{(i)}$	$\cdots$	$\mathscr{C}_{K_i}^{(i)}$	$\sum$
	$\mathscr{C}_1$	$n_{11}^{(i)}$	$n_{12}^{(i)}$	$\cdots$	$n_{1K_i}^{(i)}$	n_{1+}
	$\mathscr{C}_2$	$n_{21}^{(i)}$	$n_{22}^{(i)}$	$\cdots$	$n_{2K_i}^{(i)}$	n_{2+}
	.	.	.	$\cdots$	.	.
	$\mathscr{C}_K$	$n_{K1}^{(i)}$	$n_{K2}^{(i)}$	$\cdots$	$n_{KK_i}^{(i)}$	n_{K+}
	$\sum$	$n_{+1}^{(i)}$	$n_{+2}^{(i)}$	$\cdots$	$n_{+K_i}^{(i)}$	n

TABLE 4

Sample KCC Utility Functions

	$\mu(\mathbf{m}_{k,i})$	$U_\mu(\boldsymbol{\pi}, \boldsymbol{\pi}_i)$	$f(\mathbf{bl}, \mathbf{m}_k)$
U_c	$\|\mathbf{m}_{k,i}\|_2^2 - \|\mathbf{P}^{(i)}\|_2^2$	$\sum_{k=1}^{K} p_k + \|\mathbf{P}_k^{(i)}\|_2^2 - \|\mathbf{P}^{(i)}\|_2^2$	$\sum_{i=1}^{r} w_i \|\mathbf{b}_{l,i} - \mathbf{m}_{k,i}\|_2^2$
U_H	$(-H(\mathbf{m}_{k,i})) - (-H(\mathbf{P}^{(i)}))$	$\sum_{k=1}^{K} p_k + (-H(\mathbf{P}_k^{(i)})) - (-H(\mathbf{P}^{(i)}))$	$\sum_{i=1}^{r} w_i D(\mathbf{b}_{l,i} \|\mathbf{m}_{k,i})$
$U_{\cos}$	$\|\mathbf{m}_{k,i}\|_2 - \|\mathbf{P}^{(i)}\|_2$	$\sum_{k=1}^{K} p_k + \|\mathbf{P}_k^{(i)}\|_2 - \|\mathbf{P}^{(i)}\|_2$	$\sum_{i=1}^{r} w_i (1 - \cos(\mathbf{b}_{l,i}, \mathbf{m}_{k,i}))$
U_{L_p}	$\|\mathbf{m}_{k,i}\|_p - \|\mathbf{P}^{(i)}\|_p$	$\sum_{k=1}^{K} p_k + \|\mathbf{P}_k^{(i)}\|_p - \|\mathbf{P}^{(i)}\|_p$	$\sum_{i=1}^{r} w_i (1 - \frac{\sum_{j=1}^{K_i} b_{l,ij} m_{k,i,j}^{p-1}}{\|\mathbf{m}_{k,i}\|_p^{p-1}}$

Note: $\mathbf{P}_k^{(i)} = \mathbf{m}_{k,i} = (p_{k1}^{(i)} / p_{k+}, \cdots, p_{kj}^{(i)} / p_{k+}, \cdots, p_{kK_i}^{(i)} / p_{k+})$, $\mathbf{P}^{(i)} = (n_{+1}^{(i)} / n, \cdots, n_{+j}^{(i)} / n, \cdots, p_{+K_i}^{(i)} / n)$.

TABLE 5

Statistics of data sets.

Data set	Type	#instance	#feature	#cluster	#outlier
Amazon(A)	Image	958	800	10	0
breast	Tabular	699	9	2	0
Caltech(C)	Image	1123	800	10	0
caltech	Image	1415	4096	4	67
cacmcisi	Text	4463	14409	2	0
classic	Text	7094	41681	4	0
cranmed	Text	2431	41681	2	0
Dslr(D)	Image	157	800	1	0
ecoli[*]	Tabular	331	7	6	0
elephant	Image	#superpixel	3	2	0
fbis	Text	2463	2000	10	332
ferrari	Image	#superpixel	3	2	0
gymnastics	Image	#superpixel	3	2	0
hitech	Text	2301	126321	6	0
iris	Tabular	150	4	3	0
k1b	Text	2340	21839	6	0
kite	Image	#superpixel	3	2	0
la12	Text	6279	31472	6	0
mm	Text	2521	126373	2	0
pendigits	Tabular	10992	16	10	0
satimage	Tabular	4435	36	6	0
skating	Image	#superpixel	3	2	0
re1	Text	1657	3758	6	527
reviews	Text	4069	126373	5	0
Webcam(W)	Image	295	800	10	0
wap	Text	1560	8460	10	251
wine [+]	Tabular	178	13	3	0
yeast	Tabular	1484	8	4	185

Note:

*(1) **:* two clusters containing only two objects are deleted.

(2) +: the last attribute is normalized by a scaling factor 1000.

TABLE 6

Experimental results of the K-means solutions and other competitive methods in different tasks.

Problem	Data set	Metric	Competitive methods			K-means solution
			CSPA	**HCC**	**PTGP**	**KCC**
Consensus clustering	*iris*	R_n / *NMI*/ *Accuracy*	**0.94/0.91/0.98**	0.73/0.79/0.89	0.75/0.80/0.90	0.75/0.80/0.90
	cranmed		0.71/0.66/0.92	**0.99/0.98/0.99**	**0.99/0.98/0.99**	**0.99/0.98/0.99**
	hitech		0.18/0.23/0.43	0.27/0.33/0.50	0.27/0.33/0.50	**0.30/0.34/0.51**
	k1b		0.20/0.43/0.45	0.37/0.60/0.62	0.37/0.60/0.62	**0.41/0.61/0.64**
	mm		0.44/0.35/0.83	−0.01/0.00/0.53	−0.01/0.00/0.53	**0.62/0.52/0.89**
			CSPA	**HCC**	**PTGP**	**SEC**
Consensus clustering	*cacmcisi*	R_n / *NMI*/ *Accuracy*	0.35/0.37/0.79	−0.01/0.00/0.68	−0.01/0.00/0.68	**0.61/0.54/0.89**
	classic		0.42/0.52/0.66	0.55/0.69/0.76	0.00/0.00/0.45	**0.68/0.72/0.88**
	*la*12		0.37/0.43/0.57	0.51/0.58/0.69	0.51/0.58/0.69	**0.59/0.59/0.75**
	reviews		0.33/0.38/0.58	0.46/0.52/0.70	0.46/0.52/0.70	**0.58/0.59/0.75**
	wine		0.15/0.16/0.51	0.15/0.17/0.52	0.19/0.25/0.60	**0.33/0.39/0.65**
			CNMF	**LCVQE**	**KCC**	**PLCC**
Constrained clustering	*breast*	R_n / *NMI*/ *Accuracy*	0.90/0.82/0.97	**0.92/0.85/0.98**	**0.92/0.85/0.98**	**0.92/0.85/0.98**
	ecoli		0.71/0.65.0.80	0.80/0.75/0.85	0.64/0.68/0.79	**0.82/0.80/0.91**
	iris		0.83/0.82/**0.94**	0.83/0.81/**0.94**	0.82/0.81/0.93	**0.85/0.86/0.94**
	pendigits		0.41/0.61/0.56	0.29/0.47/0.46	0.61/0.71/0.76	**0.75/0.79/0.87**
	satimage		0.33/0.38/0.51	0.29/0.40/0.53	0.46/0.57/**0.71**	**0.53/0.61**/0.68
			Joulin	**Vicente**	**Rubio**	**SG-PLCC**
Image co-segmentation	*elephant*	*Accuracy*	0.70	0.43	0775	**0.90**
	ferrari		0.85	**0.90**	0.84	**0.90**
	gymnastics		0.91	0.92	0.87	**0.97**
	kite		0.87	0.90	0.90	**0.98**
	skating		**0.82**	0.78	0.77	**0.82**
	Single-source		**TCA**	**TSL**	**JDA**	**SP-UDA**
Domain adaptation	$C \rightarrow W$	*Accuracy*	0.39	0.34	0.37	**0.54**
	$A \rightarrow D$		0.33	0.26	0.29	**0.40**
	$W \rightarrow A$		0.30	0.30	0.36	**0.44**
	$D \rightarrow A$		0.32	0.28	0.30	**0.45**
	Multi-source		**RDALR**	**FDDL**	**SDDL**	**SP-UDA**
	$A, D \rightarrow W$	*Accuracy*	0.37	0.41	038	**0.76**
	$A, W \rightarrow D$		0.31	0.38	0.57	**0.74**
	$D, W \rightarrow A$		0.21	0.19	0.24	**0.44**
			LOF	**FABOD**	**iForest**	**COR**
Outlier detection	*caltech*	*Jaccard*	0.02	0.08	0.28	**0.97**
	fbis		0.08	0.06	0.05	**0.34**
	*re*1		0.22	0.19	0.17	**0.28**

Problem	Data set	Metric	Competitive methods			K-means solution
			CSPA	HCC	PTGP	KCC
	wap		0.11	0.13	0.11	**0.22**
	yeast		0.11	0.14	0.24	**0.50**

Note: For the first three cluster analysis related tasks, we report cluster validationin terms of R_n, *NMI*, and *Accuracy* for evaluation. For image co-segmentation, the competitive methods only report the *Accuracy* performance. For the reminder of non-cluster analysis tasks, we employ task-specific measurements for evaluation.

TABLE 7

Execution time of the K-means solutions and other competitive methods in seconds.

Task	Data set	Competitive methods			K-means solution	
		CSPA	**HCC**	**PTGP**	**KCC**	
Consensus clusteringwith utility function	*iris*	1.34	4.57	285	**0.29**	
	cranmed	5.27	105.41	35.76	**0.44**	
	hitech	4.77	102.93	36.93	**0.43**	
	k1b	5.66	119.36	35.27	**0.21**	
	mm	6.57	112.34	10.61	**0.14**	
		CSPA	**HCC**	**PTGP**	**SEC**	
Consensus clusteringwith co-association matrix	*cacmcisi*	18.55	543.20	117.69	**0.25**	
	classic	32.14	1640.71	524.76	**0.62**	
	*la*12	21.48	1148.17	44.27	**0.17**	
	reviews	10.97	397.16	26.89	**0.12**	
	wine	0.83	4.57	2.85	**0.05**	
		CNMF	**LCVQE**	**KCC**	**PLCC**	
Constrained clustering	*breast*	0.43	0.05	0.27	**0.01**	
	ecoli	0.19	0.03	0.22	**0.01**	
	iris	0.13	**0.01**	0.07	**0.01**	
	pendigits	195.38	76.73	4.98	**0.45**	
	satimage	0.05	**0.01**	0.10	**0.01**	
		LOF	**FABOD**	**iForest**	**BP**	**COR**
Outlier detection	*caltech*	13.69	140.68	6.91	12.86	**0.14**
	fbis	17.54	1319.21	12.60	15.66	**0.38**
	*re*1	16.86	738.22	8.50	20.03	**0.10**
	wap	26.81	1811.28	8.53	36.58	**0.19**
	yeast	0.09	3.44	4.35	0.28	**0.03**

Note: All the algorithms in the above table were implemented by MATLAB and run on a Ubuntu 14.04 platform with Intel Core i7-6900K @ 3.2GHz and 64 GB RAM.