Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Robust deep *k*-means: An effective and simple method for data clustering☆

Shudong Huang[a], Zhao Kang[b], Zenglin Xu[c,b,d], Quanhui Liu[a,*]

[a] *College of Computer Science, Sichuan University, Chengdu 610065, China*
[b] *School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China*
[c] *School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, China*
[d] *Centre for Artificial Intelligence, Peng Cheng Lab, Shenzhen 518055, China*

A R T I C L E I N F O

Article history:
Received 13 October 2020
Revised 10 March 2021
Accepted 18 April 2021
Available online 28 April 2021

Keywords:
k-means algorithm
Robust clustering
Deep learning

A B S T R A C T

Clustering aims to partition an input dataset into distinct groups according to some distance or similarity measurements. One of the most widely used clustering method nowadays is the *k*-means algorithm because of its simplicity and efficiency. In the last few decades, *k*-means and its various extensions have been formulated to solve the practical clustering problems. However, existing clustering methods are often presented in a single-layer formulation (i.e., shallow formulation). As a result, the mapping between the obtained low-level representation and the original input data may contain rather complex hierarchical information. To overcome the drawbacks of low-level features, deep learning techniques are adopted to extract deep representations and improve the clustering performance. In this paper, we propose a robust deep *k*-means model to learn the hidden representations associate with different implicit lower-level attributes. By using the deep structure to hierarchically perform *k*-means, the hierarchical semantics of data can be exploited in a layerwise way. Data samples from the same class are forced to be closer layer by layer, which is beneficial for clustering task. The objective function of our model is derived to a more trackable form such that the optimization problem can be tackled more easily and the final robust results can be obtained. Experimental results over 12 benchmark data sets substantiate that the proposed model achieves a breakthrough in clustering performance, compared with both classical and state-of-the-art methods.

1. Introduction

Clustering, the goal of which focuses on dividing a dataset into homogeneous groups, is no doubt one of the most fundamental techniques in statistic and machine learning [1,2]. Custering has been found to conduct surprisingly well, especially in unsupervised scenarios [3]. Myriads of applications can be formulated as a clustering problem, text mining [4], voice recognition [5], image segmentation [6], to name a few. In recent years, a number of clustering methods have been investigated based on different methodologies and statistical theories [7], such as *k*-means clustering [8], spectral clustering [9], nonnegative matrix factorization-based clustering [10], information theoretic clustering [11,12], multi-view clustering [13,14], etc.. Among them, *k*-means successfully attracted extensive attention because of its effectiveness and simplicity since it was presented in 1967 [8]. What's more, it has been widely recognized as one of the top ten data mining algorithms, for contributions to the usage and clustering performance in various practical problems [15].

Recent development on machine learning has illustrated that one can process explosively increase of data more effectively with deep learning techniques, especially in unsupervised settings. For example, the deep neural networks have been commonly used in clustering tasks [16,17]. Ji et al. [16] presented a deep neural network architecture for subspace clustering by introducing a self-expressive layer between the encoder and the decoder. Zhou et al. [17] further extended this architecture to a deep adversarial subspace clustering model by utilizing the adversarial learning. That is, a subspace clustering generator is used to learn the sample representations, while a quality-verifying discriminator is used to evaluate current clustering performance by estimating whether the re-sampled data have consistent properties. Guo et al. [18] proposed to jointly optimize cluster labels assignment and learn fea-

☆ This paper is an extended version of [1], which has been accepted for presentation at the International Conference on Neural Computing for Advanced Applications (NCAA-2020).

* Corresponding author.
E-mail address: quanhuiliu@scu.edu.cn (Q. Liu).

https://doi.org/10.1016/j.patcog.2021.107996
0031-3203/© 2021 Elsevier Ltd. All rights reserved.

tures by making use of local structure preservation and applying the under-complete autoencoder. Dizaji et al. [19] defined a clustering model using KL divergence minimization, which can map the raw data into a discriminative subspace and predict cluster assignments. Peng et al. [20] introduced a structured autoencoder for subspace clustering, where both the local and global subspace structure can be preserved by minimizing reconstruction error and incorporating a prior structured information. Peng et al. [21] discovered a common invariance based on the assumption that different distance metrics would result in similar clustering assignments on the manifold. Based on such a common invariance, a deep clustering method is designed by minimizing the discrepancy between pairwise sample assignments for each data point. Zhang et al. [22] proposed an end-to-end self-supervised convolutional clustering network, which can accomplish the convolutional network module, self-expression module, and spectral clustering module into a joint optimization framework. However, the training process of these methods are typically time-consuming and unstable. Besides, there are myriads of parameters need to be tuned which makes them impractical for unsupervised tasks. More important, these methods require a vast amount of data to train, which in turn needs extensive computational power. Combining deep learning architecture and classical clustering models into a unified framework provides a better potential solution for clustering tasks.

In the past decades, numerous variants of classical k-means algorithm have been studied to boost the clustering performance. Ding and He [23] proved that principal components essentially afford the continuous solutions, which can be seen as discrete indicators for k-means clustering. Buchta et al. [24] employed cosine similarities to conduct prototype-based partitioning, on which a spherical k-means algorithm was proposed for text clustering. Khanmohammadi et al. [25] tried to overcome the model sensitivity to initial cluster centroids by combining overlapping k-means and k-harmonic means algorithms. It uses the output of k-harmonic means method to initialize the cluster centers of overlapping k-means method. By locating the seed points at dense areas of the dataset, Kumar and Reddy [26] improved the performance of k-means filtering method and separated the data points well. The dense areas, unlike other methods, are identified by representing the data points in a kd-tree. Considering existing methods lack statistical guarantees when many features are irrelevant, Chakraborty et al. [27] addressed the problem by applying entropy regularization to learn feature relevance while annealing. The convergence and consistency of the model was guaranteed, and a scalable majorization-minimization algorithm was proposed to optimize the model. This model yields significant improvements over k-means, yet retains the same computational complexity. Capó et al. [28] intended to solve the bottleneck of massive data by introducing an efficient approximation to the k-means problem. This method partitions the dataset into a number of subsets, each of which is generally characterized by its representative and weight. The k-means algorithm is then performed on such local representation, which reduces the number of computed distances.

Despite the remarkable progress made by the aforementioned k-means approaches, these methods are conventionally devised in a single-layer formulation. Thus the mapping between the obtained low-dimensional representation and the original input data may contain rather complex hierarchical information. Considering the the development of deep learning that forces on adopting multiple processing layers to extract the hierarchical information of data [29], in this paper, a novel robust deep k-means model is proposed to exploit the hierarchical information of multiple level attributes. The overall framework of our model is shown in Fig. 1. As we can see, by using the deep structure to hierarchically perform k-means, the hierarchical semantics of data can be exploited in a layerwise way. That is, the data samples from the same class are gathered closer layer by layer, which is greatly beneficial for the clustering task.

The main contributions of this work are three aspects:

- A novel robust deep model is proposed to perform k-means hierarchically, thus the hierarchical semantics of data can be explored in a layerwise way. As a result, data samples from the same class are effectively gathered closer layer by layer, which provides a clear clustering structure.
- To solve the optimization problem of our model, the corresponding objective function is derived to a more trackable form and an alternative updating algorithm is presented to solve the optimization problem.
- Experiments over 12 benchmark data sets are conducted and show promising results, compared to both classical and state-of-the-art methods.

The groundwork of this paper is conceived as follows. We brief introduce the closely related works in Section 2. The details of our model are given in Section 3. Experimental results are described in Section 4. Finally, we give a conclusion of this paper in Section 5.

The differences between this work and our earlier paper [1] are threefold. First, we propose a general form of our robust deep k-kmeans model. In detail, we deploy a series of divergence functions to measure the reconstruction errors (Section 3), instead of merely the Frobenius norm which is sensitive to noisy data and outliers. Hence our earlier work [1] is simply a special case of this paper. Second, more comprehensive experiments over 12 benchmark datasets are conducted, substantiating the robustness and effectiveness of our model: (i) the clustering results on more datasets and advanced baselines are recorded (Section 4.3); (ii) adding the experiment results with respect to different parameter settings (Section 4.4); (iii) showcasing the experiments of convergence analysis (Section 4.5); (iv) investigating the influence of different divergence functions to clustering performance (Section 4.6). Third, we have introduced more closely related literatures (in Sections 1 and 2), clarifying the connections and differences with the state-of-the-arts. This helps better position the proposed work in the community.

2. Preliminaries

Nonnegative Matrix Factorization (NMF) has received much attention in data clustering due to its intuitive parts-based interpretation [30,31]. Previous studies have shown that NMF is essentially equal to k-means with a relaxed condition [30]. Here we start with an introduction to NMF [32]. Suppose $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n] \in \mathbb{R}^{m\times n}$ is a nonnegative data matrix with n data samples and m features. NMF aims to find two nonnegative matrices $\mathbf{U} \in \mathbb{R}^{m\times c}$ and $\mathbf{V} \in \mathbb{R}^{n\times c}$ such that $\mathbf{X} \approx \mathbf{U}\mathbf{V}^T$, and the general form of NMF is

$$\mathcal{D}_\beta\left(\mathbf{X}|\mathbf{U}\mathbf{V}^T\right) = \sum_{i=1}^{m}\sum_{j=1}^{n} d_\beta\left(\mathbf{X}_{ij}|\left(\mathbf{U}\mathbf{V}^T\right)_{ij}\right) \quad \text{s.t.}\, \mathbf{U} \geq 0, \mathbf{V} \geq 0, \tag{1}$$

where $\mathcal{D}_\beta\left(\mathbf{X}|\widehat{\mathbf{X}}\right)$ denotes a scalar cost function (i.e., some measure of divergence between $\mathbf{X}$ and its reconstruction $\widehat{\mathbf{X}}$) and $\mathbf{X}_{ij}$ is the ijth element of $\mathbf{X}$. In Eq. (1), a family of divergence functions termed β-divergence [33] can be adopted. There are three divergence functions most widely used in NMF:

* $\beta = 2$ (Euclidean Distance): $d_2(a|b) = \frac{1}{2}(a-b)^2$
* $\beta = 1$ (Kullback–Leibler Divergence): $d_1(a|b) = a\log\frac{a}{b} - a + b$
* $\beta = 0$ (Itakura–Saito Divergence): $d_0(a|b) = \frac{a}{b} - \log\frac{a}{b} - 1$

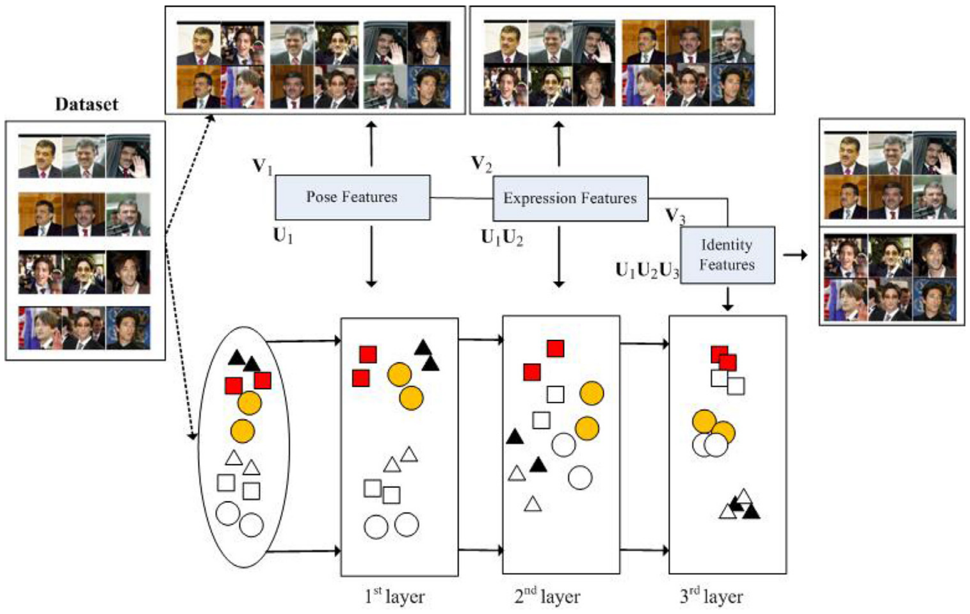

Fig. 1. A three-layer deep k-means structure is used to depict the framework of our model. Same shape indicates the data samples belong to the same class. It is clear the variability in face data can be distinguished based on the attributes such as the facial expression (with or without a smile) or the pose (eyes left, right or front) of the subject. The proposed deep model focuses on exploiting the hierarchical information layer by layer. As a result, a more discriminative representation can be expected.

[32] adopted the Euclidean distance in Eq. (1), i.e.,

$$J_{NMF} = \sum_{j=1}^{m}\sum_{i=1}^{n}\left(\mathbf{X}_{ij} - (\mathbf{U}\mathbf{V}^T)_{ij}\right)^2 = \|\mathbf{X} - \mathbf{U}\mathbf{V}^T\|_F^2 \quad \text{s.t.} \mathbf{U} \geq 0, \mathbf{V} \geq 0, \tag{2}$$

where $\|\cdot\|_F$ means the Frobenius norm. [32] further pointed out that Eq. (2) is a bi-convex formulation (convex in $\mathbf{U}$ or $\mathbf{V}$ only), and searched the local minimum by applying the updating rules as follows:

$$\begin{aligned}\mathbf{U}_{ij} &\leftarrow \mathbf{U}_{ij}\frac{(\mathbf{X}\mathbf{V})_{ij}}{(\mathbf{U}\mathbf{V}^T\mathbf{V})_{ij}},\\ \mathbf{V}_{ij} &\leftarrow \mathbf{V}_{ij}\frac{(\mathbf{X}^T\mathbf{U})_{ij}}{(\mathbf{V}\mathbf{U}^T\mathbf{U})_{ij}},\end{aligned}$$

where $\mathbf{V}$ can be seen as the cluster indicator matrix [30], $\mathbf{U}$ represents the centroid matrix, and c denotes the number of clusters. Usually, we have $c \ll n$ and $c \ll m$, which means Eq. (2) actually searches for a low-dimensional representation $\mathbf{V}$ of $\mathbf{X}$.

But in reality, datasets are often complex and always contain multiple hierarchical modalities (i.e., factors). Taking the face dataset as an example, it typically consists of some common modalities, such as expression, pose, scene, and so on. Thus it is obvious that single-layer-based NMF cannot fully exploit the hidden information with respect to different factors. To fill this gap, [34] studied a multi-layer deep model that innovatively explores the hierarchical information of data by conducting the semi-NMF hierarchically. And the basic formulation of the deep semi-NMF model is defined as

$$\begin{aligned}\mathbf{X} &\approx \mathbf{U}_1\mathbf{V}_1^T,\\ \mathbf{X} &\approx \mathbf{U}_1\mathbf{U}_2\mathbf{V}_2^T,\\ &\vdots\\ \mathbf{X} &\approx \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T,\end{aligned} \tag{3}$$

where r is the layer number, $\mathbf{U}_i$ and $\mathbf{V}_i$ are basis matrix and representation matrix of the ith layer, respectively. It is clear that deep semi-NMF also targets to search a low-dimensional embedding representation, i.e., the last layer $\mathbf{V}_r$. By hierarchically decomposing each layer $\mathbf{V}_i(i < r)$, Eq. (3) is able to discover the latent hierarchy. Compared with existing single-layer NMF models, deep semi-NMF allows for a better ability to uncover the hierarchical information of data, as different modalities can be can be recognized by the low-dimensional representations of different layers. Hence our model can fully achieve representations suitable for the subsequent clustering in terms of different modalities. For instance, as shown in Fig. 1, $\mathbf{U}_3$ corresponds to the characteristics of expressions, $\mathbf{U}_2\mathbf{U}_3$ corresponds to the characteristics of poses, and finally, $\mathbf{U} = \mathbf{U}_1\mathbf{U}_2\mathbf{U}_3$ corresponds to identities mapping of face images. In this way, a better high-identifiability, final-layer representation for clustering according to the characteristic with the lowest variability can be obtained.

3. The proposed model

In this section, we present a novel deep k-means model termed robust deep k-means (RDKM). We introduce an efficient updating algorithm to solve the corresponding optimization problem. The convergence of the proposed algorithm is also analysed.

3.1. Robust deep k-means

To explore the low-dimensional representations with respect to different modalities, a novel robust deep k-means model is investigated by utilizing the deep structure to conduct k-means hierarchically. In this paper, to enlarge the applicable range of our model (i.e., deal with both the negative and nonnegative data), the nonnegative constraint on $\mathbf{U}_i$ is omitted. Considering that the nonnegativity constraints on V_i make the optimization problem difficult to solve, we transform the objective function into a more trackable form by introduce new variables V_i^+. In this way, the nonnegativity constraints are separated and equivalently adopted, with the constraints $V_i = V_i^+$. Hence we not only extend the application, but also preserve the strong interpretability of our model. Here we use alternating direction method of multipliers (ADMM) [35] to solve the optimization problem. Mathematically, the proposed RDMK model is formulated as

$$\begin{aligned}&J = \mathcal{D}_\beta(\mathbf{X}|\mathbf{Y})\\ &\text{s.t.} \mathbf{Y} = \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T, (\mathbf{V}_r)_{.c} = \{0,1\}, \textstyle\sum_{c=1}^{C}(\mathbf{V}_r)_{.c} = 1,\\ &\mathbf{V}_i = \mathbf{V}_i^+, \mathbf{V}_i^+ \geq 0, i \in [1,\ldots,r-1].\end{aligned} \tag{4}$$

In Eq. (4), we can see that the 1-of-C coding scheme is employed on each row of $\mathbf{V}_r$. The primary goal of the 1-of-C coding scheme is to guarantee the uniqueness of $\mathbf{V}_r$. Moreover, based on $\mathbf{V}_r$, we can obtain the final discrete partition result directly without any postprocessing.

Similar to Eq. (2), if the Euclidean distance (i.e., $\beta = 2$) is adopted in Eq. (4), we have

$$\begin{aligned}&J = \|\mathbf{X} - \mathbf{Y}\|_F^2 \\ &\text{s.t.} \mathbf{Y} = \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T, (\mathbf{V}_r)_{.c} = \{0, 1\}, \textstyle\sum_{c=1}^C (\mathbf{V}_r)_{.c} = 1, \\ &\mathbf{V}_i = \mathbf{V}_i^+, \mathbf{V}_i^+ \geq 0, i \in [1, \ldots, r-1].\end{aligned} \tag{5}$$

However, it has been proven that Frobenius norm is sensitive to noisy data and outliers [36,37]. To enhance the robustness of the proposed model, the sparsity-inducing norm (i.e., $l_{2,1}$-norm), is employed in our model. According to [36], $l_{2,1}$-norm is able to reduce the influence of outliers as it performs the l_2-norm within a data point and the l_1-norm among data points. Finally, our robust deep k-means (RDKM) model is written as

$$\begin{aligned}&J_{RDKM} = \|\mathbf{X} - \mathbf{Y}\|_{2,1} \\ &\text{s.t.} \mathbf{Y} = \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T, (\mathbf{V}_r)_{.c} = \{0, 1\}, \textstyle\sum_{c=1}^C (\mathbf{V}_r)_{.c} = 1, \\ &\mathbf{V}_i = \mathbf{V}_i^+, \mathbf{V}_i^+ \geq 0, i \in [1, \ldots, r-1].\end{aligned} \tag{6}$$

As we can see, $\|\mathbf{X} - \mathbf{Y}\|_{2,1}$ is simple to minimize with respect to $\mathbf{Y}$, while $\|\mathbf{X} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T\|_{2,1}$ is not that simple to minimize with respect to $\mathbf{U}_i$ or $\mathbf{V}_i$. Multiplicative updating rules implicitly address the problem such that $\mathbf{U}_i$ and $\mathbf{V}_i$ decouple. In ADMM context, a natural formulation is to optimize $\|\mathbf{X} - \mathbf{Y}\|_{2,1}$ with the constraint $\mathbf{Y} = \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T$. **That's the reason why we consider solving the problem like** Eq. (6).

To verify the robustness and effectivenss of $l_{2,1}$-norm used in our model, the cases of different divergence functions (i.e., $\beta = 2$, $\beta = 1$, and $\beta = 0$) will be discussed in a later section. The optimization algorithms with respect to different divergence functions will also be described in Appendix A.

For Eq. (6), we introduce an effective optimization algorithm based on ADMM [35]. The Lagrangian function of Eq. (6) is

$$\begin{aligned}&\mathcal{L}(\mathbf{Y}, \mathbf{U}_i, \mathbf{V}_i, \mathbf{V}_i^+, \boldsymbol{\mu}, \boldsymbol{\lambda}_i) = \|\mathbf{X} - \mathbf{Y}\|_{2,1} + \langle \boldsymbol{\mu}, \mathbf{Y} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T \rangle \\ &+ \tfrac{\rho}{2}\|\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T\|_F^2 + \textstyle\sum_{i=1}^{r-1} \langle \boldsymbol{\lambda}_i, \mathbf{V}_i - \mathbf{V}_i^+ \rangle \\ &+ \frac{\rho}{2}\sum_{i=1}^{r-1} \|\mathbf{V}_i - \mathbf{V}_i^+\|_F^2,\end{aligned} \tag{7}$$

where ρ denotes a penalty parameter, $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}_i$ both represent the Lagrangian multipliers, and $\langle \cdot, \cdot \rangle$ means inner product operation.

The alternating algorithm for Eq. (7) is derived by minimizing $\mathcal{L}$ with respect to $\mathbf{Y}, \mathbf{U}_i, \mathbf{V}_i, \mathbf{V}_i^+$, one at a time while fixing others, i.e., the solution can be drawn by repeating the steps as follows,

$$\begin{aligned}\mathbf{U}_i^{t+1} &= \arg\min_{\mathbf{U}_i} \mathcal{L}\Big(\mathbf{Y}^t, \mathbf{U}_i, \mathbf{V}_i^t, \big(\mathbf{V}_i^+\big)^t, \boldsymbol{\mu}^t, \boldsymbol{\lambda}_i^t\Big), \\ \mathbf{V}_i^{t+1} &= \arg\min_{\mathbf{V}_i} \mathcal{L}\Big(\mathbf{Y}^t, \mathbf{U}_i^{t+1}, \mathbf{V}_i, \big(\mathbf{V}_i^+\big)^t, \boldsymbol{\mu}^t, \boldsymbol{\lambda}_i^t\Big), \\ \mathbf{Y}^{t+1} &= \arg\min_{\mathbf{Y}} \mathcal{L}\Big(\mathbf{Y}, \mathbf{U}_i^{t+1}, \mathbf{V}_i^{t+1}, \big(\mathbf{V}_i^+\big)^t, \boldsymbol{\mu}^t, \boldsymbol{\lambda}_i^t\Big), \\ \big(\mathbf{V}_i^+\big)^{t+1} &= \arg\min_{\mathbf{V}_i^+} \mathcal{L}\big(\mathbf{Y}^{t+1}, \mathbf{U}_i^{t+1}, \mathbf{V}_i^{t+1}, \mathbf{V}_i^+, \boldsymbol{\mu}^t, \boldsymbol{\lambda}_i^t\big),\end{aligned} \tag{8}$$

and then updating the multipliers $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}_i$ with the following formulas [35]:

$$\begin{aligned}\boldsymbol{\mu}^{t+1} &= \boldsymbol{\mu}^t + \rho\Big(\mathbf{Y}^{t+1} - \big(\mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T\big)^{t+1}\Big), \\ \boldsymbol{\lambda}_i^{t+1} &= \boldsymbol{\lambda}_i^t + \rho\Big(\mathbf{V}_i^{t+1} - \big(\mathbf{V}_i^+\big)^{t+1}\Big).\end{aligned} \tag{9}$$

It is noteworthy that our model is different from the deep semi-NMF model [34]. For one thing, deep semi-NMF is based on Frobenius norm, which is well-known to be sensitive to noisy data and outliers. For another, the target data representation in deep semi-NMF cannot directly assign the discrete clustering result, therefore requires a postprocessing to allocate the class label to each data sample. Thus it is difficult to achieve a stable result. Moreover, the optimization algorithm of our model is totally different from [34], which will be introduced below.

3.2. Optimization

In this paper, an iterative updating algorithm is introduced to solve the optimization problem. Specifically, we update Eq. (7) with respect to each distinct variable while keeping the other variables fixed.

Before the optimization, we adopt a pre-training by decomposing the input data matrix $\mathbf{X} \approx \mathbf{U}_1\mathbf{V}_1^T$, where $\mathbf{V}_1 \in \mathbb{R}^{n \times c_1}$ and $\mathbf{U}_1 \in \mathbb{R}^{m \times c_1}$. The obtained representation matrix $\mathbf{V}_1$ is then further decomposed as $\mathbf{V}_1 \approx \mathbf{U}_2\mathbf{V}_2^T$, where $\mathbf{V}_2 \in \mathbb{R}^{n \times c_2}$ and $\mathbf{U}_2 \in \mathbb{R}^{c_1 \times c_2}$. We respectively denote c_1 and c_2 as the dimensionalities of the first layer and the second layer.[1] Continue to do so, all layers will be pre-trained, which would greatly boost the effectiveness and reduce the training time of our model. This trick has been successfully employed in deep autoencoder networks [35].

With simple algebra, Eq. (7) can be rewritten as

$$\begin{aligned}&\mathcal{L}(\mathbf{Y}, \mathbf{U}_i, \mathbf{V}_i, \mathbf{V}_i^+, \boldsymbol{\mu}, \boldsymbol{\lambda}_i) = \mathrm{Tr}\Big((\mathbf{X} - \mathbf{Y})\mathbf{D}(\mathbf{X} - \mathbf{Y})^T\Big) \\ &+ \tfrac{\rho}{2}\mathrm{Tr}\Big(\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T\big)\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T\big)^T\Big) \\ &+ \langle \boldsymbol{\mu}, \mathbf{Y} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T \rangle + \sum_{i=1}^r \langle \boldsymbol{\lambda}_i, \mathbf{V}_i - \mathbf{V}_i^+ \rangle \\ &+ \tfrac{\rho}{2}\sum_{i=1}^r \mathrm{Tr}\Big(\big(\mathbf{V}_i - \mathbf{V}_i^+\big)\big(\mathbf{V}_i - \mathbf{V}_i^+\big)^T\Big),\end{aligned} \tag{10}$$

where $\mathbf{D}$ denotes a diagonal matrix with the jth diagonal element being

$$d_j = \frac{1}{2\|\mathbf{e}_j\|_2}. \tag{11}$$

and $\mathbf{e}_j$ is the jth column of $\mathbf{E} = \mathbf{X} - \mathbf{Y}$.

3.2.1. Updating $\mathbf{U}_i$

The optimization problem w.r.t. $\mathbf{U}_i$ is

$$\begin{aligned}&\mathcal{L}_U = \langle \boldsymbol{\mu}, \mathbf{Y} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T \rangle + \\ &\tfrac{\rho}{2}\mathrm{Tr}\Big(\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T\big)\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T\big)^T\Big).\end{aligned} \tag{12}$$

Setting $\frac{\partial \mathcal{L}_U}{\partial \mathbf{U}_i} = 0$, we have

$$\mathbf{U}_i = \big(\boldsymbol{\Phi}^T\boldsymbol{\Phi}\big)^{-1}\Big(\boldsymbol{\Phi}^T\mathbf{Y}\widetilde{\mathbf{V}}_i + \tfrac{\boldsymbol{\Phi}^T\boldsymbol{\mu}\widetilde{\mathbf{V}}_i}{\rho}\Big)\Big(\widetilde{\mathbf{V}}_i^T\widetilde{\mathbf{V}}_i\Big)^{-1}, \tag{13}$$

where $\boldsymbol{\Phi} = \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_{i-1}$, and $\widetilde{\mathbf{V}}_i$ is the reconstruction of the ith layer's centroid matrix.

3.2.2. Updating $\mathbf{V}_i$ $(i < r)$

The optimization problem w.r.t. $\mathbf{V}$ is

$$\begin{aligned}&\mathcal{L}_V = \langle \boldsymbol{\mu}, \mathbf{Y} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T \rangle + \\ &\tfrac{\rho}{2}\mathrm{Tr}\Big(\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T\big)\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2 \ldots \mathbf{U}_r\mathbf{V}_r^T\big)^T\Big) \\ &+ \sum_{i=1}^r \langle \boldsymbol{\lambda}_i, \mathbf{V}_i - \mathbf{V}_i^+ \rangle + \frac{\rho}{2}\sum_{i=1}^r \mathrm{Tr}\Big(\big(\mathbf{V}_i - \mathbf{V}_i^+\big)\big(\mathbf{V}_i - \mathbf{V}_i^+\big)^T\Big).\end{aligned} \tag{14}$$

Similarly, setting $\frac{\partial \mathcal{L}_V}{\partial \mathbf{V}_i} = 0$, we obtain

$$\mathbf{V}_i = \Big(\mathbf{Y}^T\boldsymbol{\Phi}\mathbf{U}_i + \mathbf{V}_i^+ + \tfrac{\boldsymbol{\mu}^T\boldsymbol{\Phi}\mathbf{U}_i}{\rho} - \tfrac{\boldsymbol{\lambda}_i}{\rho}\Big)\big(\mathbf{I} + \mathbf{U}_i^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{U}_i\big)^{-1}, \tag{15}$$

where $\mathbf{I}$ denotes an identity matrix.

[1] For simplicity, the layer size (dimensionalities) of layer 1 to layer r is written as $[c_1, \ldots, c_r]$ in this paper.

3.2.3. Updating $\mathbf{V}_r$ (i.e., $\mathbf{V}_i$, $(i=r)$)

The optimization problem w.r.t $\mathbf{V}_r$ can be stated as

$$J_{V_r} = \min_{\mathbf{V}_r} \|\mathbf{Y}-\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\|_{2,1} \\ = \min_{\mathrm{v}} \sum_{j=1}^{n} d_j \|\mathrm{x}_j - \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathrm{v}_j\|_2^2 \\ \text{s.t.} (\mathbf{V}_r)_{.c} = \{0,1\}, \textstyle\sum_{c=1}^{C} (\mathbf{V}_r)_{.c} = 1, \tag{16}$$

where x_j is the jth data point of $\mathbf{X}$, and v_j denotes the jth column of $\mathbf{V}_r^T$.It is obvious that Eq. (16) is independent for each j. Accordingly, for a specific j, it can be independently solved by

$$\min_{\mathrm{v}} \left(d\|\mathrm{x}-\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathrm{v}\|_2^2\right) \\ \text{s.t.} v_c = \{0,1\}, \textstyle\sum_{c=1}^{C} v_c = 1. \tag{17}$$

Since v satisfies the 1-of-C coding scheme, there are obvious C possible solutions for Eq. (17). We can find that each individual solution is exactly the cth column of the identity matrix $\mathbf{I}_C = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_C]$. Hence we can obtain the optimal solution, v^*, by conducting an exhaustive search, i.e.,

$$\mathrm{v}^* = \mathbf{f}_c, \tag{18}$$

where

$$c = \arg\min_{\bar{c}} \left(d\|\mathrm{x}-\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{f}_{\bar{c}}\|_2^2\right). \tag{19}$$

3.2.4. Updating $\mathbf{Y}$

The optimization problem w.r.t. $\mathbf{Y}$ is

$$\mathcal{L}_Y = \mathrm{Tr}\Big((\mathbf{X}-\mathbf{Y})\mathbf{D}(\mathbf{X}-\mathbf{Y})^T\Big) + \\ \tfrac{\rho}{2}\mathrm{Tr}\Big(\big(\mathbf{Y}-\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)\big(\mathbf{Y}-\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)^T\Big) \\ + \langle\boldsymbol{\mu}, \mathbf{Y}-\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\rangle. \tag{20}$$

Setting $\frac{\partial\mathcal{L}_Y}{\partial\mathbf{Y}} = 0$, we get

$$\mathbf{Y} = \big(2\mathbf{X}\mathbf{D} + \rho\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T - \boldsymbol{\mu}\big)(2\mathbf{D}+\rho\mathbf{I})^{-1}. \tag{21}$$

Eq. (21) gives us an update rule for $\mathbf{Y}$ when $l_{2,1}$-norm is adopted to measure the reconstruction error. One may ask what if adopting other divergence functions. Here we give the updating rules for $\mathbf{Y}$ with different divergence functions. Thus we need to solve the general optimization problem as follows

$$\mathcal{L}_{Y_\beta} = \mathcal{D}_\beta(\mathbf{X}|\mathbf{Y}) + \langle\boldsymbol{\mu}, \mathbf{Y}-\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\rangle + \\ \tfrac{\rho}{2}\mathrm{Tr}\Big(\big(\mathbf{Y}-\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)\big(\mathbf{Y}-\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)^T\Big). \tag{22}$$

According to Eq. (22), the updating rules for $\mathbf{Y}$ are given by

$$\mathbf{Y}_{ij} = \begin{cases} \frac{(2\mathbf{X}+\rho\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T-\boldsymbol{\mu})_{ij}}{2+\rho} & \text{when } \beta = 2, \\ \frac{z+\sqrt{z^2+4\rho\mathbf{X}_{ij}}}{2\rho} & \text{when } \beta = 1, \\ \mathbf{S}_{ij} - \frac{1}{3}\mathbf{A}_{ij} & \text{when } \beta = 0. \end{cases} \tag{23}$$

The definition of z, $\mathbf{S}$, $\mathbf{A}$ as well as the specific details about Eq. (23) are given in Appendix A.

3.2.5. Updating $\mathbf{V}_i^+$

The optimization problem w.r.t. $\mathbf{V}_i^+$ is

$$\mathcal{L}_{V^+} = \sum_{i=1}^{r}\langle\boldsymbol{\lambda}_i, \mathbf{V}_i - \mathbf{V}_i^+\rangle + \frac{\rho}{2}\sum_{i=1}^{r}\mathrm{Tr}\Big(\big(\mathbf{V}_i-\mathbf{V}_i^+\big)\big(\mathbf{V}_i-\mathbf{V}_i^+\big)^T\Big). \tag{24}$$

Similarly, setting $\frac{\partial\mathcal{L}_{V^+}}{\partial\mathbf{V}_i^+} = 0$, we obtain

$$\mathbf{V}_i^+ = \mathbf{V}_i + \frac{\boldsymbol{\lambda}_i}{\rho}. \tag{25}$$

For clarity, the proposed algorithm is summarized step by step in Algorithm 1.

Algorithm 1 Robust deep k-means for data clustering (RDKM).

```
Input: Input data matrix $\mathbf{X}$, layer size $p$;
Output: $\mathbf{U}_i$ and $\mathbf{V}_i$ for each of the layers;
  Iteration:
  for all layers do
    $\mathbf{U}_i, \mathbf{V}_i \leftarrow k\text{-means}\big(\mathbf{V}_{i-1}, \text{layers}(i)\big)$
  end for
  Initialize $\mathbf{D}$ as defined in Eq.~(11).
  repeat
    for all layers do
      1. Update $\widetilde{\mathbf{V}}_i = \begin{cases}\mathbf{V}_r & \text{if } i=r,\\ \mathbf{U}_{i+1}\widetilde{\mathbf{V}}_{i+1}^T & \text{otherwise.}\end{cases}$
      2. Compute $\boldsymbol{\Phi} = \prod_{j=1}^{i-1}\mathbf{U}_j$.
      3. Update $\mathbf{U}_i$ according to Eq.~(13).
      4. Update $\mathbf{V}_i$ according to Eq.~(15).
      5. Update $\mathbf{Y}$ according to Eq.~(21) or Eq.~(23).
      6. Update $\mathbf{V}_i^+$ according to Eq.~(25).
      7. Compute $\mathbf{D}$ according to Eq.~(11).
      8. Update $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} + \rho\big(\mathbf{Y}-\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)$.
      9. Update $\boldsymbol{\lambda}_i \leftarrow \boldsymbol{\lambda}_i + \rho\big(\mathbf{V}_i - \mathbf{V}_i^+\big)$.
    end for
  until Converges
```

3.3. Convergence analysis

We prove the convergence of Algorithm 1 as follows: the objective function in Eq. (6) can be divided into four subproblems and each one is a convex problem with respect to the corresponding variable. By iteratively solving the subproblems, it can be guaranteed that we can search the optimal solution to each subproblem. Finally, Algorithm 1 will converge to a local minima.

For different divergence functions, the main difference of the corresponding optimization algorithms is the updating rules for $\mathbf{Y}$. As we introduced in Appendix A, the optimization subproblems for $\mathbf{Y}$ with respect to different divergences are all convex problems, and we can obtain the corresponding closed-form solutions as shown in Eqs. (A.2), (A.5) and (A.14). Thus the optimization algorithms with respect to different divergences will also converge to a local minima.

4. Experiments

In this paper, we experimentally evaluate the effectiveness of our method. We compare the proposed RDKM on 12 benchmark datasets against six baselines: the standard k-means [8], NMF [32], Orthogonal NMF (ONMF) [30], Semi-NMF (SNMF) [38], $l_{2,1}$-NMF [36] and the deep Semi-NMF (DeepSNMF) [34].

4.1. Data sets

In our experiment, we adopt 12 benchmark datasets including two gene expression datasets, four textual datasets, and six image datasets. As an illustration, Fig. 2 shows the sample images of datasets COIL and MNIST. Table 1 summarizes the specific details of all datasets, from which we can see the instance number is ranged from 102 to 7094, and the number of features is from 256 to 7511, covering a broad range of properties.

4.2. Parameter setting

For k-means algorithm, on all datasets, we conduct k-means until it convergence. For a fair comparison, the results of k-means are also used as the initialization of other compared methods. For the compared methods, we set the parameters just as reported in each

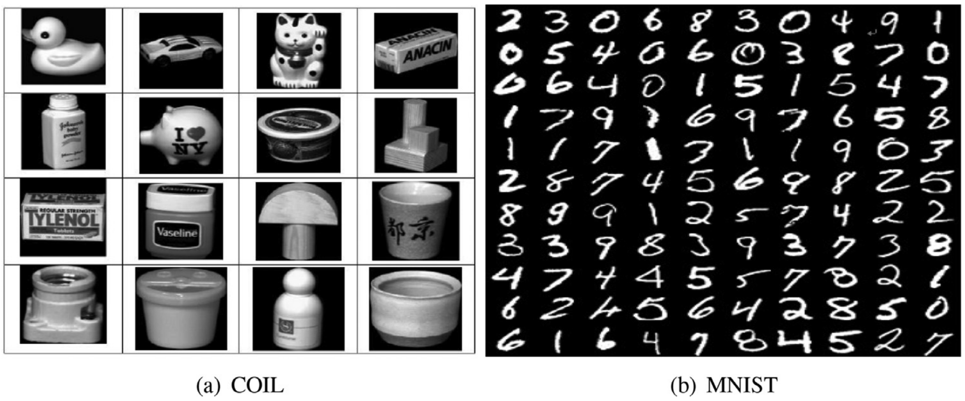
Fig. 2. Sample images of (a) COIL and (b) MNIST.

Table 1
The details of our experimental data sets.

ID	Data sets	# samples	# features	# classes	# type
1	PROSTATEML	102	5966	2	gene
2	Yale32	165	1024	15	image
3	Lungml	203	3312	5	gene
4	ORL32	400	1024	40	image
5	COIL	1440	1024	20	image
6	Semeion	1593	256	10	image
7	MSRA	1799	256	12	image
8	Text	1946	7511	2	text
9	Cranmed	2431	462	2	text
10	MNIST05	3495	784	10	image
11	Cacmcisi	4663	348	2	text
12	Classic	7094	462	4	text

paper. If there are no suggested values, we searched the parameters exhaustively, and used the ones producing the best performance. For our RDKM, the layer sizes (as described in 3.2) are set as [50 C], [100 C] and [100 50 C] according to [14,39]. As for the parameter ρ, we search it from {1e5, 1e4, 1e3, 1e2, 0.1, 1, 10, 100}.

To reduce the influence of the initialization, we repeat the experiments 20 times, and reported the average performance over the 20 repetitions.

4.3. Results and analysis

Table 2 displays the clustering performance in terms of clustering accuracy (ACC), normalized mutual information (NMI) and purity of all methods over 12 datasets. It can be seen that the proposed method outperforms other algorithms in most cases. In detail, for ACC, our model achieves the best results 11 times among the 12 datasets. For NMI, our model achieves the best results 10 times. For purity, the number is also 10. In a word, the clustering performance is sufficient to validate the effectiveness of the proposed model. The superiority of RDKM demonstrates that it is beneficial to discover a better cluster structure by exploring the hierarchical semantics of data. The reason is that, by applying the deep framework to perform k-means hierarchically, the hierarchical information of data can be exploited in a layerwise way, and finally, a better high-identifiability, final-layer representation is obtained for clustering task. By skillfully combining the deep framework and k-means model, our model is able to boost the clustering performance in general cases.

According to the theoretical analysis and empirical results reported in this paper, it can be concluded that combining the deep structure learning and classical machine learning models into a unified framework would be an interesting research trend.

4.4. Parameter discussion

There are two parameters, the layer size and the penalty parameter ρ, need to be tuned in our model. Here we study the clustering performance with respect to different parameter settings. As shown in Fig. 3, we can find that the clustering performance is stable with respect to different settings of layer size, while the performance is a little sensitive with respect to the penalty parameter ρ. For image datasets (e.g., Yale32, ORL32, COIL and MSRA), better results can be obtained when ρ is in the range [1e3,1e1]. For gene expression dataset (e.g., Lungml), better results are obtained when ρ is in the range [1e2,1]. While for textual datase (e.g., Cranmed), searching ρ in the range [1e5,1e3] would be a better choice. In general, we can search the parameter ρ in the range [1e3,1] for a relatively good performance.

4.5. Convergence analysis

In this subsection, we empirically show how fast our method will converge. Fig. 4 shows the convergence curves of our RDKM, where x-axis denotes the number of iterations, while y-axis denotes the objective value. It can be observed that the updating rules for our RDKM converge very fast, usually within 100 iterations. For data set MSRA, it even converges within 10 iterations, which further demonstrates the effectiveness of the proposed optimization algorithm.

4.6. Divergence function selection

As mentioned in Section 2, several widely used divergence functions can be adopted for residue calculation. We use $l_{2,1}$-norm in our previous experiment. In this section, we empirically investigate the influence of different divergence functions to clustering performance.

As can be seen in Eqs. (6) and (7), only the step of updating $\mathbf{Y}$ will change when the divergence function is changed. That is, the updating of all variables except $\mathbf{Y}$ are still the same with respect to different divergence functions. We give the updating rules for $\mathbf{Y}$ when $\beta = 2$ (Euclidean Distance), $\beta = 1$ (Kullback–Leibler Divergence), and $\beta = 0$ (Itakura–Saito Divergence) in Appendix A.

The clustering performance with respect to different divergence functions is shown in Fig. 5. For comparison, the results of our model (i.e., $l_{2,1}$-norm based divergence function) are also recorded. It is clear that $l_{2,1}$-norm outperforms other divergence functions on all datasets, which once again verifies the robustness of our model. For the three cases that $\beta = 2$, $\beta = 1$, and $\beta = 0$, we can see that $\beta = 1$ obtains good results on dataset Yale32, $\beta = 0$ obtains good results on dataset Lungml, while $\beta = 2$ achieves better results on

Table 2
Clustering performance measured by Accuracy/NMI/Purity (mean ± std) of the compared methods. The best performance on each data set is bolded. •/○ indicates our method is significantly better/worse than compared methods (paired *t*-tests at 95% significance level). The win/tie/loss (w/t/l) counts for our method are summarized in the last row.

Data sets	Metrics	Kmeans	NMF	ONMF	L21NMF	SNMF	DeepSNMF	RDKM
Prostateml	ACC	58.82 ± 0.5•	58.63 ± 0.6•	57.65 ± 0.9•	58.14 ± 0.5•	58.82 ± 0.5•	59.98 ± 6.9•	**62.78 ± 2.1**
	NMI	12.39 ± 0.3•	12.33 ± 0.3•	11.78 ± 0.5•	11.98 ± 0.3•	12.39 ± 0.2•	11.93 ± 1.0•	**13.36 ± 1.8**
	Purity	58.82 ± 0.5•	58.63 ± 0.6•	57.65 ± 0.9•	58.14 ± 0.5•	58.82 ± 0.5•	59.98 ± 6.9•	**62.78 ± 2.1**
Yael32	ACC	37.39 ± 3.1•	35.88 ± 3.3•	35.58 ± 3.3•	38.67 ± 2.7•	37.64 ± 2.9•	29.09 ± 1.6•	**40.06 ± 2.8**
	NMI	43.05 ± 3.1•	42.42 ± 3.1•	41.04 ± 3.0	45.02 ± 1.6•	43.44 ± 2.3•	28.92 ± 1.1•	**48.22 ± 1.9**
	Purity	39.76 ± 3.1•	37.88 ± 2.7•	37.82 ± 3.1•	40.48 ± 1.7•	39.82 ± 2.6•	32.12 ± 1.8•	**43.52 ± 1.7**
Lungml	ACC	68.92 ± 11.1	62.27 ± 7.2•	68.92 ± 11.1	62.17 ± 5.9•	68.92 ± 11.1	58.62 ± 7.0•	**69.01 ± 1.5**
	NMI	52.10 ± 8.1	47.22 ± 4.4•	52.10 ± 8.1	47.07 ± 3.3•	52.10 ± 8.1	16.17 ± 1.8•	**52.72 ± 5.2**
	Purity	87.04 ± 3.0•	85.32 ± 4.0•	87.04 ± 3.0•	84.63 ± 3.2•	87.04 ± 3.0•	72.17 ± 2.4•	**89.41 ± 2.9**
ORL32	ACC	50.30 ± 2.2•	51.97 ± 2.8•	49.90 ± 3.1•	53.40 ± 4.1	51.78 ± 3.5•	49.86 ± 2.0•	**54.50 ± 1.2**
	NMI	71.06 ± 1.3•	72.10 ± 1.3•	70.11 ± 1.7•	72.70 ± 1.8•	71.76 ± 1.9•	68.83 ± 1.3•	**72.94 ± 1.5**
	Purity	56.06 ± 2.4•	56.37 ± 2.2•	55.15 ± 2.9•	58.07 ± 3.5•	56.07 ± 3.1•	57.18 ± 2.1•	**63.33 ± 2.0**
COIL	ACC	59.43 ± 6.8•	62.24 ± 3.1•	58.35 ± 6.0•	63.49 ± 4.4•	63.78 ± 5.9•	66.36 ± 6.2•	**68.03 ± 3.8**
	NMI	74.53 ± 2.8•	73.12 ± 1.7•	72.84 ± 2.6•	74.04 ± 2.3•	74.91 ± 3.0•	77.52 ± 7.4•	**78.99 ± 1.6**
	Purity	64.62 ± 5.1•	66.24 ± 2.7•	62.95 ± 4.7•	67.03 ± 3.5•	67.10 ± 4.8•	66.94 ± 6.7•	**71.01 ± 3.6**
Semeion	ACC	57.34 ± 4.7•	49.64 ± 5.6•	53.87 ± 5.8•	47.88 ± 2.7•	49.72 ± 5.4•	56.07 ± 2.0•	**62.07 ± 5.9**
	NMI	51.93 ± 2.8•	42.46 ± 3.5•	48.50 ± 2.8•	42.47 ± 2.0•	43.95 ± 2.8•	44.77 ± 1.2•	**53.97 ± 3.3**
	Purity	59.69 ± 3.8•	51.65 ± 5.0•	56.69 ± 4.4•	50.89 ± 2.0•	52.32 ± 4.7•	58.27 ± 2.6•	**67.93 ± 4.6**
MSRA	ACC	48.73 ± 4.4•	48.93 ± 2.7•	48.73 ± 5.8•	51.63 ± 5.2	49.24 ± 4.4•	49.19 ± 1.5•	**52.24 ± 2.2**
	NMI	55.85 ± 5.1•	55.86 ± 5.0•	55.64 ± 2.8•	59.06 ± 4.2	55.44 ± 4.9•	**60.10 ± 0.2**	59.83 ± 2.6
	Purity	52.42 ± 3.8•	51.86 ± 3.3•	52.30 ± 4.4•	55.14 ± 4.2	51.77 ± 3.8•	54.75 ± 0.2•	**55.79 ± 2.4**
Text	ACC	91.84 ± 2.1•	93.85 ± 3.9	92.47 ± 2.9•	90.21 ± 4.0•	90.99 ± 2.0•	90.67 ± 4.5•	**93.88 ± 5.7**
	NMI	61.31 ± 6.5	61.21 ± 1.5•	60.88 ± 1.9•	60.81 ± 3.0•	57.85 ± 5.2•	60.01 ± 4.2•	**61.55 ± 5.4**
	Purity	91.84 ± 2.1•	**94.08 ± 3.6** ○	92.71 ± 2.7•	90.62 ± 3.2•	90.99 ± 2.0•	90.67 ± 4.5•	93.88 ± 5.7
Cranmed	ACC	74.58 ± 0.1•	80.13 ± 8.8•	77.31 ± 1.2•	77.39 ± 2.5•	76.49 ± 4.9•	80.23 ± 3.1•	**82.31 ± 3.9**
	NMI	18.79 ± 0.3•	31.67 ± 5.6	20.74 ± 0.3•	20.84 ± 1.2•	24.66 ± 5.4•	25.05 ± 2.4•	**32.89 ± 2.3**
	Purity	74.58 ± 0.1•	80.38 ± 8.0•	77.78 ± 3.4•	77.67 ± 3.2•	79.10 ± 6.1•	**82.51 ± 3.0** ○	82.31 ± 3.9
MINIST	ACC	54.84 ± 4.1•	52.30 ± 3.3•	**58.84 ± 4.7** ○	54.00 ± 5.1•	48.65 ± 2.2•	56.18 ± 1.2○	55.46 ± 3.7
	NMI	49.44 ± 1.6•	43.21 ± 2.1•	49.92 ± 1.4○	45.26 ± 3.0•	41.26 ± 1.8•	**49.81 ± 2.0** ○	49.53 ± 2.4
	Purity	58.38 ± 2.1•	54.22 ± 2.4•	59.84 ± 2.3•	57.18 ± 4.5•	52.89 ± 2.7•	60.73 ± 2.4•	**62.10 ± 2.8**
Cacmcisi	ACC	91.99 ± 0.2•	89.75 ± 5.4•	94.96 ± 0.6•	95.37 ± 0.8•	92.22 ± 0.3•	92.80 ± 3.5•	**97.12 ± 7.7**
	NMI	58.47 ± 0.1•	60.01 ± 2.5•	70.42 ± 0.2•	72.05 ± 0.4•	70.52 ± 0.2•	70.07 ± 3.3•	**73.06 ± 2.4**
	Purity	91.99 ± 0.2•	91.70 ± 3.2•	94.96 ± 0.6•	95.37 ± 0.8•	93.69 ± 0.7•	94.86 ± 4.1•	**97.12 ± 7.7**
Classic	ACC	67.45 ± 0.3•	70.31 ± 3.2•	76.52 ± 6.6•	76.19 ± 6.4•	74.44 ± 2.0•	72.40 ± 1.3•	**76.98 ± 5.9**
	NMI	46.76 ± 1.3•	50.12 ± 2.2•	47.91 ± 6.4•	57.45 ± 7.9•	55.30 ± 2.6•	56.27 ± 2.5•	**59.61 ± 4.1**
	Purity	70.48 ± 1.0•	74.09 ± 3.3•	77.74 ± 5.2•	79.73 ± 4.2•	76.42 ± 1.2•	78.41 ± 1.9•	**81.07 ± 4.1**
RDKM: w/t/l	ACC	11/1/0	11/1/0	10/1/1	10/2/0	11/1/0	11/0/1	
	NMI	10/2/0	11/1/0	9/2/1	11/1/0	11/1/0	10/1/1	
	Purity	12/0/0	11/0/1	12/0/0	11/1/0	12/0/0	11/0/1	

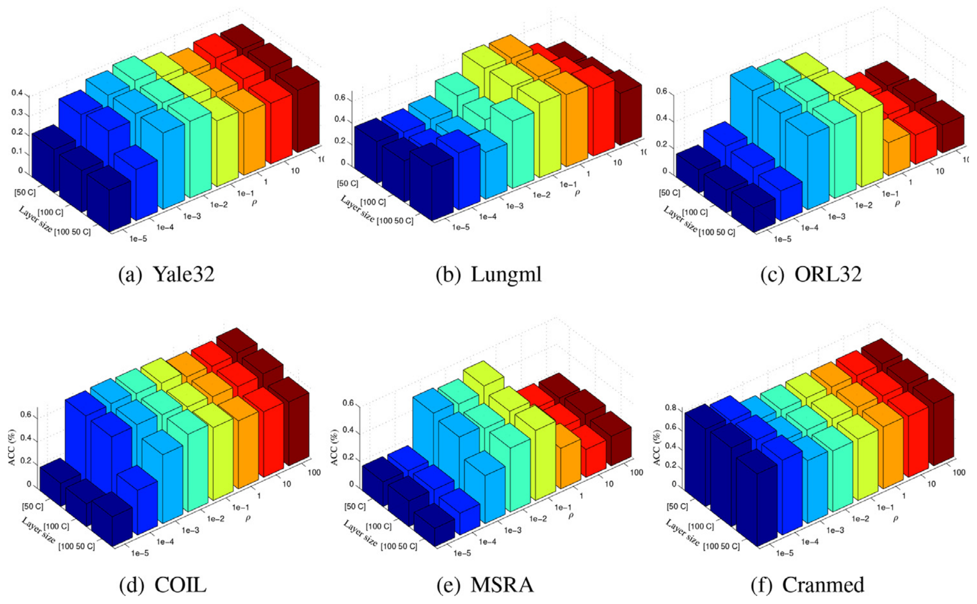

Fig. 3. Clustering results of RDKM w.r.t. layer size and parameter ρ.

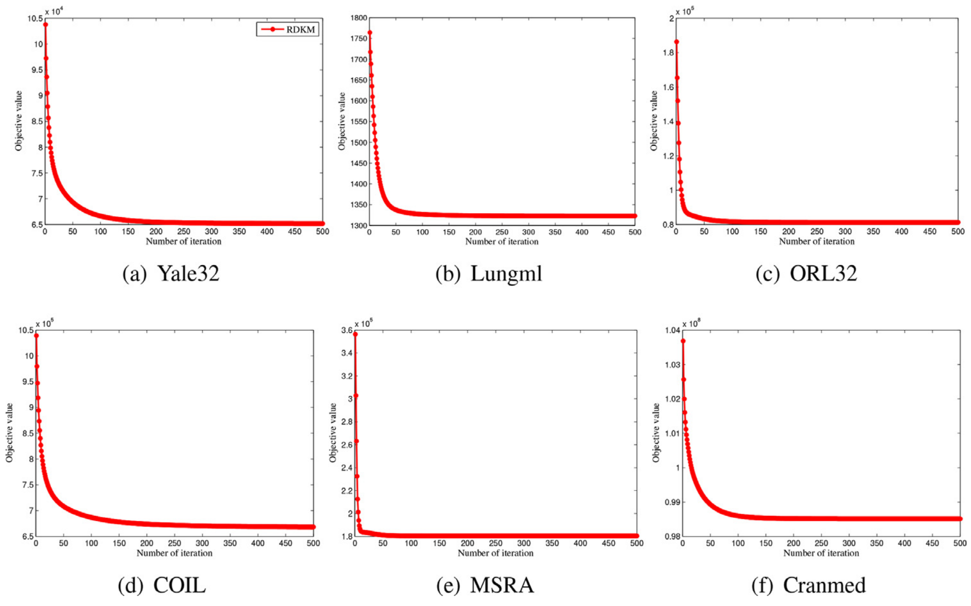

Fig. 4. Convergence speed of RDKM.

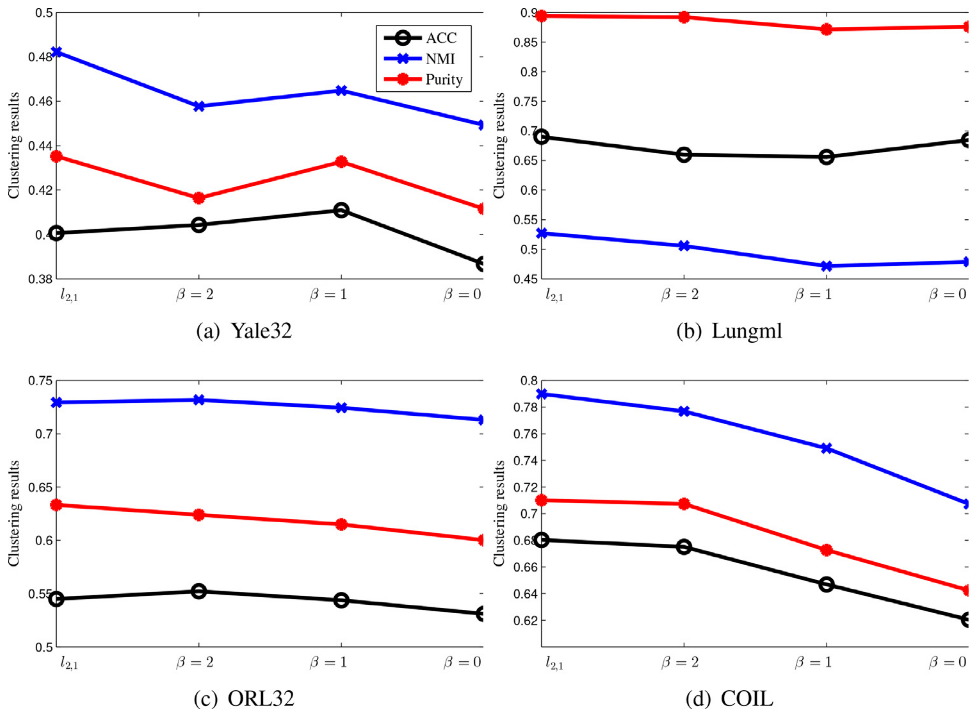

Fig. 5. Clustering performance of RDKM w.r.t. divergence functions.

datasets ORL32 and COIL. That is, different divergence functions get better results on different datasets. Generally, $l_{2,1}$-norm may be a better choice as it consistently achieves good performance.

5. Conclusion

In this paper, we introduced a robust deep k-means model to learn the hidden representations associate with different implicit lower-level attributes. By using the deep structure to hierarchically perform k-means, the hierarchical semantics of data can be exploited in a layerwise way. Data samples from the same class are forced to be closer layer by layer, which is beneficial for clustering task. The objective function of our model is derived to a more trackable form such that the optimization problem can be tackled more easily and the final robust results can be obtained. Experimental results over 12 benchmark data sets substantiate that: (i) the proposed model achieves a breakthrough in clustering performance, compared with both classical and state-of-the-art methods; (ii) the clustering performance is robust with respect to different settings of layer size as well as the divergence functions; (iii) the proposed optimization algorithm is effective and converges very fast. In our future work, it would be interesting to combine our deep model and other machine learning models (e.g., kernel learning and classification methods) into a unified framework.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was partially supported by the State Key Program of the National Science Foundation of China under Grant 61836006, the National Natural Science Fund for Distinguished Young Scholar under Grant 61625204, Sichuan Science and Technology Program under Grant 2020YFG0323, and the Fundamental Research Funds for the Central Universities under Grant 1082204112364.

Appendix A. Updating rules for Y w.r.t different divergence functions

For different divergence functions, the main difference of the corresponding optimization algorithms is the updating rules for **Y**. Here we introduce the updating rules for **Y** in terms of different divergence functions, particularly, when $\beta = 2$, $\beta = 1$, and $\beta = 0$.

When $\beta = 2$**(Euclidean Distance),** the objective function w.r.t **Y** is

$$\mathcal{L}_{Y_{\beta=2}} = \mathcal{D}_2(\mathbf{X}|\mathbf{Y}) + \langle \boldsymbol{\mu}, \mathbf{Y} - \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\rangle + \frac{\rho}{2}\mathrm{Tr}\Big(\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)^T\Big). \tag{A.1}$$

Calculating the derivative of $\mathcal{L}_{Y_{\beta=2}}$ w.r.t. **Y** and setting it to 0, we have

$$\mathbf{Y} = \big(2\mathbf{X} + \rho\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T - \boldsymbol{\mu}\big)/(2+\rho). \tag{A.2}$$

When $\beta = 1$ **(Kullback–Leibler Divergence),** the objective function w.r.t **Y** is

$$\mathcal{L}_{Y_{\beta=1}} = \mathcal{D}_1(\mathbf{X}|\mathbf{Y}) + \langle \boldsymbol{\mu}, \mathbf{Y} - \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\rangle + \frac{\rho}{2}\mathrm{Tr}\Big(\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)^T\Big). \tag{A.3}$$

Calculating the derivative of $\mathcal{L}_{Y_{\beta=1}}$ w.r.t. **Y** in an element-wise way, we have

$$-\frac{\mathbf{X}_{ij}}{\mathbf{Y}_{ij}} + 1 + \boldsymbol{\mu}_{ij} + \rho\Big((\mathbf{Y})_{ij} - \big(\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)_{ij}\Big) = 0. \tag{A.4}$$

Thus the solution for **Y** is

$$\mathbf{Y}_{ij} = \frac{z + \sqrt{z^2 + 4\rho\mathbf{X}_{ij}}}{2\rho}. \tag{A.5}$$

where $z = \rho\big(\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)_{ij} - \boldsymbol{\mu}_{ij} - 1$.

When $\beta = 0$ **(Itakura–Saito Divergence),** the objective function w.r.t **Y** is

$$\mathcal{L}_{Y_{\beta=0}} = \mathcal{D}_0(\mathbf{X}|\mathbf{Y}) + \langle \boldsymbol{\mu}, \mathbf{Y} - \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\rangle + \frac{\rho}{2}\mathrm{Tr}\Big(\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)\big(\mathbf{Y} - \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)^T\Big). \tag{A.6}$$

It is clear that $\mathbf{Y}^*$ would be a minimizer if and only if the gradient vanishes at $\mathbf{Y}^*$ and the the Hessian matrix is positive definite:

$$g_{ij}\big(\mathbf{Y}^*_{ij}\big) = 0, g'_{ij}\big(\mathbf{Y}^*_{ij}\big) = 0, \forall i, j, \tag{A.7}$$

where g_{ij} represents the derivative w.r.t $\mathbf{Y}_{ij}$ and can be defined as

$$g_{ij}(y) = -\frac{\mathbf{X}_{ij}}{y^2} + \frac{1}{y} + \boldsymbol{\mu}_{ij} + \rho\Big(y - \big(\mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T\big)_{ij}\Big). \tag{A.8}$$

Denote $p_{ij}(y) = \left(\frac{y^2}{\rho}\right)g_{ij}(y)$ such that p_{ij} is a cubic polynomial. It is clear that p_{ij} has the same roots as g_{ij}, and p'_{ij} has the same sign as g'_{ij}. p_{ij} can be explicitly expressed as

$$p_{ij}(y) = y^3 + \mathbf{A}_{ij}y^2 + \frac{1}{\rho}y - \frac{1}{\rho}\mathbf{X}_{ij}, \tag{A.9}$$

where $\mathbf{A} = \frac{1}{\rho}\boldsymbol{\mu} - \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_r\mathbf{V}_r^T$. Substituting $y = s - \frac{1}{3}\mathbf{A}_{ij}$ in Eq. (A.9), a depressed cubic can be obtained

$$q(s) = s^3 + 3\mathbf{B}_{ij}s - 2\mathbf{R}_{ij}, \tag{A.10}$$

where $\mathbf{B}_{ij} = \frac{1}{3}\rho - \frac{1}{9}\mathbf{A}_{ij}^2$, $\mathbf{R}_{ij} = -\frac{1}{27}\mathbf{A}_{ij}^3 + \frac{1}{6\rho}\mathbf{A}_{ij} + \frac{1}{2\rho}\mathbf{X}_{ij}$. We want to search a positive root $s_0 > 0$ of $q(s)$ such that $q'(s_0) < 0$. When dealing with nonnegative data (i.e., $\mathbf{X}_{ij} \geq 0$), we can search at least one such root as $p(0) \leq 0$ and $p(\infty) \to \infty$. The roots of q_{ij} are related to the roots of p_{ij} by $y_t = s_t - \frac{1}{3}\mathbf{A}_{ij}$ ($t = 0, 1, 2$, i.e., there are at most three roots for Eq. (A.10)). The discriminant **H** of q_{ij} can be defined as

$$\mathbf{H}_{ij} = \mathbf{B}_{ij}^3 + \mathbf{R}_{ij}^2, \tag{A.11}$$

and there are three cases:

1) $\mathbf{H}_{ij} > 0$, one real root:

$$s_0 = \Big(\mathbf{B}_{ij} + \sqrt{\mathbf{H}_{ij}}\Big)^{\frac{1}{3}} + \Big(\mathbf{B}_{ij} - \sqrt{\mathbf{H}_{ij}}\Big)^{\frac{1}{3}}. \tag{A.12}$$

Correspondingly, the root y_0 of p_{ij} must be positive and the minimizer of Eq. (A.6).

2) $\mathbf{H}_{ij} = 0$, two distinct real roots:

A double root $s_1 = s_2 = -\frac{1}{2}s_0$ can be derived. However, the double roots correspond to the point of inflections of q_{ij}, i.e., $g'_{ij}(y_1) = 0$. Thus y_1 is not a minimizer of Eq. (A.6). As a result, the relevant root is still s_0.

3) $\mathbf{H}_{ij} < 0$, three distinct real roots:

$$s_t = 2\sqrt{-\mathbf{B}_{ij}}\cos\left(\frac{1}{3}\cos^{-1}\frac{\mathbf{R}_{ij}}{\sqrt{-\mathbf{B}_{ij}^3}} - t\frac{2\pi}{3}\right), \tag{A.13}$$

for $t = 0, 1, 2$. When there are three distinct roots, p'_{ij} can only be positive at the smallest and largest roots. Since $s_0 \geq s_1 \geq s_2$, we only need to check y_0 and y_2 (the latter only if $y_2 > 0$). For simplicity, we always take y_0, which is guaranteed to be at least a local minima of Eq. (A.6). We omit the solution derivation and for more details please refer to [40].

In summary, we update **Y** by

$$\mathbf{Y}_{ij} = \mathbf{S}_{ij} - \frac{1}{3}\mathbf{A}_{ij}, \tag{A.14}$$

where

$$\mathbf{S}_{ij} = \begin{cases} \Big(\mathbf{B}_{ij} + \sqrt{\mathbf{H}_{ij}}\Big)^{\frac{1}{3}} + \Big(\mathbf{B}_{ij} - \sqrt{\mathbf{H}_{ij}}\Big)^{\frac{1}{3}} & \mathbf{H}_{ij} \geq 0, \\ 2\sqrt{-\mathbf{B}_{ij}}\cos\left(\frac{1}{3}\cos^{-1}\frac{\mathbf{R}_{ij}}{\sqrt{-\mathbf{B}_{ij}^3}} - t\frac{2\pi}{3}\right) & \mathbf{H}_{ij} < 0. \end{cases} \tag{A.15}$$

References

[1] S. Huang, Z. Kang, Z. Xu, Deep *k*-means: a simple and effective method for data clustering, in: Proceedings of the International Conference on Neural Computing for Advanced Applications, Springer, 2020, pp. 272–283.
[2] A.K. Jain, Data clustering: 50 years beyond *k*-means, Pattern Recognit. Lett. 31 (8) (2010) 651–666.
[3] A. Banerjee, S. Merugu, I.S. Dhillon, J. Ghosh, Clustering with Bregman divergences, J. Mach. Learn. Res. 6 (2005) 1705–1749.
[4] V. Tunali, T. Bilgin, A. Camurcu, An improved clustering algorithm for text mining: multi-cluster spherical *k*-means, Int. Arab J. Inf. Technol. 13 (1) (2016).
[5] S.V. Ault, R.J. Perez, C.A. Kimble, J. Wang, On speech recognition algorithms, Int. J. Mach. Learn. Comput. 8 (6) (2018) 518–523.
[6] L. Wang, C. Pan, Robust level set image segmentation via a local correntropy-based *k*-means clustering, Pattern Recognit. 47 (5) (2014) 1917–1925.
[7] Y. Ren, C. Domeniconi, G. Zhang, G. Yu, Weighted-object ensemble clustering: methods and analysis, Knowl. Inf. Syst. 51 (2) (2017) 661–689.
[8] J. Macqueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.
[9] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, in: Advances in Neural Information Processing Systems, 2002, pp. 849–856.
[10] D. Cai, X. He, J. Han, T.S. Huang, Graph regularized nonnegative matrix factorization for data representation, IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2010) 1548–1560.

[11] E. Gokcay, J.C. Principe, Information theoretic clustering, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2) (2002) 158–171.
[12] F. Gullo, G. Ponti, A. Tagarelli, S. Greco, An information-theoretic approach to hierarchical clustering of uncertain data, Inf. Sci. 402 (2017) 199–215.
[13] S. Huang, I. Tsang, Z. Xu, J.C. Lv, Measuring diversity in graph learning: a unified framework for structured multi-view clustering, IEEE Trans. Knowl. Data Eng. (2021) 1–15.
[14] H. Zhao, Z. Ding, Y. Fu, Multi-view clustering via deep matrix factorization, in: Proceedings of the 21st AAAI Conference on Artificial Intelligence, 2017, pp. 2921–2927.
[15] X. Wu, V. Kumar, J.R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, S.Y. Philip, et al., Top 10 algorithms in data mining, Knowl. Inf. Syst. 14 (1) (2008) 1–37.
[16] P. Ji, T. Zhang, H. Li, M. Salzmann, I. Reid, Deep subspace clustering networks, in: Advances in Neural Information Processing Systems, 2017, pp. 24–33.
[17] P. Zhou, Y. Hou, J. Feng, Deep adversarial subspace clustering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1596–1604.
[18] X. Guo, L. Gao, X. Liu, J. Yin, Improved deep embedded clustering with local structure preservation, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 1753–1759.
[19] K. Ghasedi Dizaji, A. Herandi, C. Deng, W. Cai, H. Huang, Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5736–5745.
[20] X. Peng, J. Feng, S. Xiao, W.-Y. Yau, J.T. Zhou, S. Yang, Structured autoencoders for subspace clustering, IEEE Trans. Image Process. 27 (10) (2018) 5076–5086.
[21] X. Peng, H. Zhu, J. Feng, C. Shen, H. Zhang, J.T. Zhou, Deep clustering with sample-assignment invariance prior, IEEE Trans. Neural Netw. Learn. Syst. (2019) 1–12.
[22] J. Zhang, C.-G. Li, C. You, X. Qi, H. Zhang, J. Guo, Z. Lin, Self-supervised convolutional subspace clustering network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 5473–5482.
[23] C. Ding, X. He, K-means clustering via principal component analysis, in: Proceedings of the 21st International Conference on Machine Learning, 2004, pp. 29–37.
[24] C. Buchta, M. Kober, I. Feinerer, K. Hornik, Spherical *k*-means clustering, J. Stat. Softw. 50 (10) (2012) 1–22.
[25] S. Khanmohammadi, N. Adibeig, S. Shanehbandy, An improved overlapping *k*-means clustering method for medical applications, Expert Syst. Appl. 67 (2017) 12–18.
[26] K.M. Kumar, A.R.M. Reddy, An efficient *k*-means clustering filtering algorithm using density based initial cluster centers, Inf. Sci. 418 (2017) 286–301.
[27] S. Chakraborty, D. Paul, S. Das, J. Xu, Entropy weighted power *k*-means clustering, in: International Conference on Artificial Intelligence and Statistics, 2020, pp. 691–701.
[28] M. Capó, A. Pérez, J.A. Lozano, An efficient approximation to the *k*-means clustering for massive data, Knowledge-Based Syst. 117 (2017) 56–69.
[29] Y. Bengio, Learning deep architectures for AI, Found. Trends® Mach. Learn. 2 (1) (2009) 1–127.
[30] C. Ding, T. Li, W. Peng, H. Park, Orthogonal nonnegative matrix *t*-factorizations for clustering, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006, pp. 126–135.
[31] S. Huang, H. Wang, T. Li, T. Li, Z. Xu, Robust graph regularized nonnegative matrix factorization for clustering, Data Min. Knowl. Discov. 32 (2) (2018) 483–503.
[32] D.D. Lee, H.S. Seung, Algorithms for non-negative matrix factorization, in: Advances in Neural Information Processing Systems, 2001, pp. 556–562.
[33] C. Févotte, J. Idier, Algorithms for nonnegative matrix factorization with the β-divergence, Neural Comput. 23 (9) (2011) 2421–2456.
[34] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, B.W. Schuller, A deep matrix factorization method for learning attribute representations, IEEE Trans. Pattern Anal. Mach. Intell. 39 (3) (2017) 417–429.
[35] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., Distributed optimization and statistical learning via the alternating direction method of multipliers, Found. Trends® Mach. learn. 3 (1) (2011) 1–122.
[36] D. Kong, C. Ding, H. Huang, Robust nonnegative matrix factorization using l21-norm, in: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, 2011, pp. 673–682.
[37] H. Gao, F. Nie, W. Cai, H. Huang, Robust capped norm nonnegative matrix factorization: capped norm NMF, in: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, 2015, pp. 871–880.
[38] C. Ding, T. Li, M.I. Jordan, Convex and semi-nonnegative matrix factorizations, IEEE Trans. Pattern Anal. Mach. Intell. 32 (1) (2010) 45–55.
[39] S. Huang, Z. Kang, Z. Xu, Auto-weighted multi-view clustering via deep matrix decomposition, Pattern Recognit. 97 (2020) 1–11.
[40] D.L. Sun, R. Mazumder, Non-negative matrix completion for bandwidth extension: a convex optimization approach, in: Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing, 2013, pp. 1–6.

Shudong Huang: received his Ph.D. degree in computer science and engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China. He is currently an associate research professor with the College of Computer Science, Sichuan University, Chengdu, China. His research interests include machine learning, deep learning, pattern recognition and data mining. He has published nearly 20 research papers in several related conferences and journals, including IEEE TCYB, Pattern Recognition, Data Mining and Knowledge Discovery, Information Sciences, Knowledge-Based Systems, Neurocomputing, IJCAI, IJCNN, etc. He has been a PC member or reviewer to a number of top conferences and journals such as AAAI, IJCAI, IEEE TNNLS, IEEE TFS, Neurocomputing, etc.

Zhao Kang: Dr. Kang received his Ph.D. degree in Computer Science from Southern Illinois University Carbondale, USA, in 2017. Currently, he is an assistant professor at the School of Computer Science and Engineering at University of Electronic Science and Technology of China (UESTC). His research interests are machine learning, data mining, pattern recognition, and deep learning. He has published over 40 research papers in top-tier conferences and journals, including AAAI, IJCAI, ICDE, CVPR, SIGKDD, ICDM, CIKM, SDM, ACML, IEEE Transactions on Cybernetics, ACM TIST, ACM TKDD, Pattern Recognition, Neurocomputing, Knowledge-Based Systems. He has been a PC member or reviewer to a number of top conferences such as AAAI, IJCAI, CVPR, ICCV, MM, ICDM, CIKM, etc. He regularly servers as a reviewer to IEEE TNNLS, IEEE Transactions on Cybernetics, IEEE TKDE, Neurocomputing, etc.

Zenglin Xu: received the Ph.D. degree in computer science and engineering from the Chinese University of Hong Kong. He is currently a full professor in University of Electronic Science & Technology of China. He has been working at Michigan State University, Cluster of Excellence at Saarland University and Max Planck Institute for Informatics, and later Purdue University. Dr. Xu's research interests include machine learning and its applications in information retrieval, health informatics, and social network analysis. He has been elected in the 2013's China Youth 1000-talent Program. He is the recipient of the outstanding student paper honorable mention of AAAI 2015, the best student paper runner up of ACML 2016, and the 2016 young researcher award from APNNS.

Quanhui Liu: received his Ph.D. degree in computer science and engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China. He is currently an associate research professor with the College of Computer Science, Sichuan University, Chengdu, China. His research interests include social networks, deep learning, and complex network.