Robust Embedded Deep K-means Clustering

Rui Zhang
Arizona State University
Tempe, Arizona, U.S.A.
ruizhang8633@gmail.com

Hanghang Tong*
University of Illinois at Urbana-Champaign
Urbana, Illinois, U.S.A.
htong@illinois.edu

Yinglong Xia
Facebook
Menlo Park, California, U.S.A.
yxia@fb.com

Yada Zhu
IBM T.J. Watson Research
Yorktown Heights, New York, U.S.A.
yzhu@us.ibm.com

ABSTRACT

Deep neural network clustering is superior to the conventional clustering methods due to deep feature extraction and nonlinear dimensionality reduction. Nevertheless, deep neural network leads to a rough representation regarding the inherent relationship of the data points. Therefore, it is still difficult for deep neural network to exploit the effective structure for direct clustering. To address this issue, we propose a robust embedded deep K-means clustering (RED-KC) method. The proposed RED-KC approach utilizes the δ-norm metric to constrain the feature mapping process of the auto-encoder network, so that data are mapped to a latent feature space, which is more conducive to the robust clustering. Compared to the existing auto-encoder networks with the fixed prior, the proposed RED-KC is adaptive during the process of feature mapping. More importantly, the proposed RED-KC embeds the clustering process with the auto-encoder network, such that deep feature extraction and clustering can be performed simultaneously. Accordingly, a direct and efficient clustering could be obtained within only one step to avoid the inconvenience of multiple separate stages, namely, losing pivotal information and correlation. Consequently, extensive experiments are provided to validate the effectiveness of the proposed approach.

KEYWORDS

embedded clustering, deep neural networks, robust k-means, auto-encoder

ACM Reference Format:
Rui Zhang, Hanghang Tong, Yinglong Xia, and Yada Zhu. 2019. Robust Embedded Deep K-means Clustering. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM '19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3357384.3357985

1 INTRODUCTION

Clustering [43] serves as the main task regarding grouping a set of objects such that the objects in the same group are more similar to each other than to those in the other groups [8, 45, 47]. Most conventional clustering algorithms perform the learning process according to the linear models [10, 20, 30, 32, 41, 46, 48, 49], which frequently fail to handle the data with irregular or nonlinear distributions. During the past decades, spectral-based clustering methods [28, 36, 38] and density-based clustering methods have achieved state-of-the-art results. The spectral-based clustering approaches perform the clustering in the following two steps. Firstly, it builds up an affinity matrix, i.e., similarity graph to represent the local structure of the data. Secondly, it clusters the data via grouping the eigenvectors of the graph Laplacian. The main idea of the density-based clustering [14] approach is to find the high-density regions that are segmented by the low-density regions. The density peak clustering algorithm (DPCA) is proposed by Alex Rodriguez [35]. The core idea of DPCA indicates that the center of the cluster is surrounded by certain points of low local density, which are segregated from the residual points of high local density. The DPCA incorporates the clustering process of non-clustered center points into a single process. Since the selection of the cluster center and the clustering of the non-cluster points are usually independent, the clustering precision is improved via DPCA.

To address the clustering problem concerning the nonlinear distributed data, the sparse subspace clustering (SSC) [7] algorithm is developed. The main contribution of SSC indicates that a sparse representation should tend to select the data points from the same subspace among the potential data representations. In fact, the SSC algorithm is developed by solving the sparse optimization within the framework of spectral clustering, where each cluster is projected to a low-dimensional subspace. Motivated by the similar idea of SSC, diverse sparse representation and low-rank approximation based methods for subspace clustering [19, 31, 32, 42] have attracted a lot of attentions in recent years. The key components of these methods are associated with a sparse and low-rank representation of the data by constructing a similarity graph upon the sparse coefficient matrix.

The spectral-based and the density-based clustering algorithms can effectively deal with the data of arbitrary distribution. However, only the superficial features of the data can be exploited [41]. Hence, it is tricky to further improve the clustering performance. On the other hand, deep neural networks can nonlinearly project the raw

[*] Hanghang Tong is the Corresponding Author. The work was partly done while Hanghang Tong was at Arizona State University.

CIKM '19, November 3–7, 2019, Beijing, China
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3357985

data to a new feature space for deep feature extraction and nonlinear dimensionality reduction. Therefore, in recent years, diverse deep subspace clustering algorithms have been developed.

Regardless of the connected or convoluted network structure, the core idea of deep neural network clustering [1, 6, 15, 17, 40] is to project the data to a new feature space, via which the clustering can be accomplished. Due to the nonlinear mapping, deep clustering has more powerful capabilities of both intrinsic feature extraction and data representation. More specifically, the auto-encoder clustering algorithms [1, 4, 37] are the deep clustering models, where a symmetric network structure is utilized to encode and decode the data simultaneously. Auto-encoder network is composed of two steps. Firstly, the code space of the data is achieved by reducing the dimensionality of the data in the latent subspace. Secondly, encoded data is reconstructed by a new generative decoded space. Based on the extension of the auto-encoder network, the idea of generative adversarial network [9, 12, 21, 39, 50] has been further introduced to enhance the efficiency of the deep clustering algorithms.

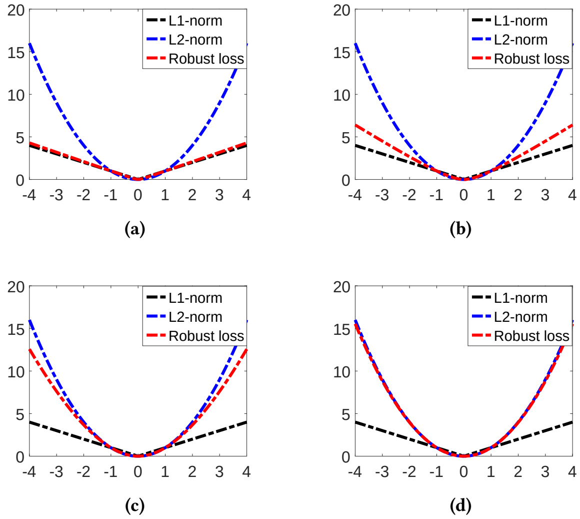
However, due to lack of prior knowledge, most of the current deep clustering algorithms [4, 19, 22, 26] obtain the rough representation of the data, such that it is often difficult to mine more effective information. To address the issue regarding the deep clustering algorithms, a robust embedded deep K-means clustering (RED-KC) approach is proposed in this paper. The proposed RED-KC approach embeds the robust K-means model with the auto-encoder network to obtain the data representation, which is more conducive to robust clustering. The proposed method has the following contributions:

1. The robust loss, namely, δ-norm metric is utilized so that the auto-encoder network can map the data to the feature space which is more conducive to robust clustering.
2. The indicator matrix is adaptively obtained. When indicator matrix degenerates to a prior label, we prove that the embedded robust K-means is equivalent to the within-class scatter under the specific condition.
3. The weighted cluster centroids can be achieved, such that a more clear grouping structure can be obtained for the data clustering.

Notations: All of uppercase italic boldface letters represent matrices, whereas lowercase italic boldface letters represent vectors. The uppercase curlicue letters represent the functions and the italic letters represent scalar values. $\boldsymbol{M}^T$ denotes the transpose of $\boldsymbol{M}$. $\boldsymbol{m}^i$ denotes the i-th row of matrix $\boldsymbol{M}$ and $\boldsymbol{m}_j$ denotes the j-th column of $\boldsymbol{M}$, where m_{ij} denotes the entry in the i-th row and the j-th column of $\boldsymbol{M}$. $|\boldsymbol{M}|$ denotes the absolute value of matrix $\boldsymbol{M}$, whereas $\|\boldsymbol{M}\|_F$ denotes the Frobenius norm of $\boldsymbol{M}$. $\mathbf{1} = [1, 1, \cdots, 1]^T \in \mathbb{R}^{N\times 1}$ and $\boldsymbol{I}$ is an identity matrix. For any matrix $\boldsymbol{M} \in \mathbb{R}^{D\times N}$, the $\ell_{2,1}$-norm is defined as

$$\|\boldsymbol{M}\|_{2,1} = \sum_{i=1}^{D} \sqrt{\sum_{j=1}^{N} m_{ij}^2} = \sum_{i=1}^{D} \left\|\boldsymbol{m}^i\right\|_2$$

where $\left\|\boldsymbol{m}^i\right\|_2$ denotes the ℓ_2-norm of vector $\boldsymbol{m}^i$.



(a)

(b)

(c)

(d)

Figure 1. Illustration of robust loss function with different δ. (a) $\delta = 0.1$. (b) $\delta = 1$. (c) $\delta = 10$. (d) $\delta = 100$.

2 ROBUST LOSS: δ-NORM

As for the metrics, the ℓ_2-norm is sensitive to the large data outliers with robustness to the small loss, while the ℓ_1-norm is sensitive to the small loss with the robustness to the large one. Therefore, we attempt to develop a robust loss, which is robust to outliers regardless of small or large losses. The proposed robust loss, i.e., δ-norm is formulated by

$$\|\boldsymbol{M}\|_{\delta}= \sum_{i} \frac{(1+\delta)\left\|\boldsymbol{m}^i\right\|_2^2}{\left\|\boldsymbol{m}^i\right\|_2+\delta} \tag{1}$$

where δ needs to be tuned. For better comprehension, the illustration of the robust loss function with different values of δ is demonstrated in Figure 1. Furthermore, the robust loss function has the following properties:

1. $\|\boldsymbol{M}\|_{\delta}$ is nonnegative and convex, which is suitable for loss function.
2. $\|\boldsymbol{M}\|_{\delta}$ is twice differentiable and easy for optimization.
3. When $\left\|\boldsymbol{m}^i\right\|_2 \ll \delta$, $\|\boldsymbol{M}\|_{\delta} \to \frac{1+\delta}{\delta}\|\boldsymbol{M}\|_F^2$
4. When $\left\|\boldsymbol{m}^i\right\|_2 \gg \delta$, $\|\boldsymbol{M}\|_{\delta} \to (1+\delta)\|\boldsymbol{M}\|_{2,1}$
5. When $\delta \to 0$, $\|\boldsymbol{M}\|_{\delta} \to \|\boldsymbol{M}\|_{2,1}$
6. When $\delta \to \infty$, $\|\boldsymbol{M}\|_{\delta} \to \|\boldsymbol{M}\|_F^2$

In sum, robust loss function interpolates between the ℓ_1-norm and ℓ_2-norm via tuning the parameter δ. To solve problem (1), we at first introduce a general robust loss function as

$$\min_{\mathbf{x}} f(\mathbf{x}) + \sum_{i} \frac{(1+\delta)\left\|h_i(\mathbf{x})\right\|_2^2}{\left\|h_i(\mathbf{x})\right\|_2+\delta} \tag{2}$$

where $h_i(\mathbf{x})$ is the vector output and the second term of problem (2) is the extension of the proposed loss function in problem (1). In particular, $f(\mathbf{x})$ is a smooth function. Accordingly, we attempt to solve problem (2) by an iterative re-weighted method. By taking the derivative of problem (2) with respect to $\mathbf{x}$ and setting it to zero, we have

$$\frac{\partial f\left(\mathbf{x}\right)+\sum_i \frac{(1+\delta)\left\|h_i(\mathbf{x})\right\|_2^2}{\left\|h_i(\mathbf{x})\right\|_2+\delta}}{\partial x}=0\Rightarrow 0=f'\left(\mathbf{x}\right)+$$

$$(1+\delta)\sum_i\frac{\left(\left\|h_i\left(\mathbf{x}\right)\right\|_2+\delta\right)\frac{\partial\left\|h_i(\mathbf{x})\right\|_2^2}{\partial x}-\left\|h_i\left(\mathbf{x}\right)\right\|_2^2\frac{\partial\left\|h_i(\mathbf{x})\right\|_2}{\partial x}}{\left(\left\|h_i\left(\mathbf{x}\right)\right\|_2+\delta\right)^2}$$

$$\Rightarrow 0=f'\left(\mathbf{x}\right)+$$

$$(1+\delta)\sum_i\frac{\left(2\left\|h_i\left(\mathbf{x}\right)\right\|_2+2\delta-\frac{2\left\|h_i(\mathbf{x})\right\|_2^2}{2\sqrt{\left\|h_i(\mathbf{x})\right\|_2^2}}\right)}{\left(\left\|h_i\left(\mathbf{x}\right)\right\|_2+\delta\right)^2}h_i\left(\mathbf{x}\right)h_i'\left(\mathbf{x}\right)$$

which further leads to

$$f'\left(\mathbf{x}\right)+2\left(1+\delta\right)\sum_i\frac{\left\|h_i\left(\mathbf{x}\right)\right\|_2+2\delta}{2(\left\|h_i\left(\mathbf{x}\right)\right\|_2+\delta)^2}h_i\left(\mathbf{x}\right)h_i'\left(\mathbf{x}\right)=0 \quad (3)$$

Moreover, we denote $\mathcal{D}_{ii}=(1+\delta)\,\frac{\|h_i(\mathbf{x})\|_2+2\delta}{2(\|h_i(\mathbf{x})\|_2+\delta)^2}$. Hence, Eq. (3) can be rewritten as

$$f'\left(\mathbf{x}\right)+2\sum_i\mathcal{D}_{ii}h_i\left(\mathbf{x}\right)h'_i\left(\mathbf{x}\right)=0 \quad (4)$$

By treating $\mathcal{D}_{ii}$ as a transitional weight, then problem (2) is equivalent to the following re-weighted problem

$$\min_{\mathbf{x}} f\left(\mathbf{x}\right)+\sum_i\mathcal{D}_{ii}\left\|h_i\left(\mathbf{x}\right)\right\|_2^2 \quad (5)$$

which shares the same KKT condition as represented in (4). We further provide the theoretical analysis between the original problem (2) and its re-weighted dual in (5) as follows.

Lemma 2.1. *For any vectors x, y with the same size, the following inequality holds:*

$$\frac{\|x\|_2^2}{\|x\|_2+\delta}-\frac{\|y\|_2+2\delta}{2(\|y\|_2+\delta)^2}\|x\|_2^2$$
$$\leq\frac{\|y\|_2^2}{\|y\|_2+\delta}-\frac{\|y\|_2+2\delta}{2(\|y\|_2+\delta)^2}\|\mathrm{y}\|_2^2$$

Proof.

$$(\|x\|_2-\|y\|_2)^2(\|x\|_2\|y\|_2+2\delta\|x\|_2+\delta\|y\|_2)\geq 0$$
$$\Rightarrow 2\,\|x\|_2^2\,\|y\|_2^2+3\delta\,\|x\|_2^2\,\|y\|_2\leq\|x\|_2\|y\|_2\,\|y\|_2^2+$$
$$\|x\|_2\|y\|_2\,\|x\|_2^2+2\delta\|x\|_2\,\|x\|_2^2+\delta\|y\|_2\,\|y\|_2^2$$
$$\Rightarrow 2\,\|x\|_2^2\,(\|y\|_2+\delta)^2$$
$$\leq(\|y\|_2\,\|y\|_2^2+\|y\|_2\,\|x\|_2^2+2\delta\,\|x\|_2^2)(\|x\|_2+\delta)$$
$$\Rightarrow\frac{\|x\|_2^2}{\|x\|_2+\delta}\leq\frac{\|y\|_2\,\|y\|_2^2+\|y\|_2\,\|x\|_2^2+2\delta\,\|x\|_2^2}{2(\|y\|_2+\delta)^2}$$
$$\Rightarrow\frac{\|x\|_2^2}{\|x\|_2+\delta}-\frac{\|y\|_2+2\delta}{2(\|y\|_2+\delta)^2}\|x\|_2^2\leq\frac{\|y\|_2\,\|y\|_2^2}{2(\|y\|_2+\delta)^2}$$
$$\Rightarrow\frac{\|x\|_2^2}{\|x\|_2+\delta}-\frac{\|y\|_2+2\delta}{2(\|y\|_2+\delta)^2}\|x\|_2^2$$
$$\leq\frac{\|y\|_2^2}{\|y\|_2+\delta}-\frac{\|y\|_2+2\delta}{2(\|y\|_2+\delta)^2}\|y\|_2^2$$

which completes the proof. □

Theorem 2.2. *The re-weighted problem* (5) *will monotonically decrease the objective of problem* (2) *by updating the transitional weight $\mathcal{D}_{ii}$ in each iteration.*

Proof. Suppose that x is updated by $\tilde{x}$ in the algorithm, then we have

$$f(\tilde{x})+\sum_i\mathcal{D}_{ii}\,\|h_i(\tilde{x})\|_2^2\leq f(x)+\sum_i\mathcal{D}_{ii}\,\|h_i(x)\|_2^2$$

Note that $\mathcal{D}_{ii}=(1+\delta)\,\frac{\|h_i(x)\|_2+2\delta}{2(\|h_i(x)\|_2+\delta)^2}$, we have

$$f(\tilde{x})+(1+\delta)\sum_i\frac{\|h_i(x)\|_2+2\delta}{2(\|h_i(x)\|_2+\delta)^2}\|h_i(\tilde{x})\|_2^2$$
$$\leq f(x)+(1+\delta)\sum_i\frac{\|h_i(x)\|_2+2\delta}{2(\|h_i(x)\|_2+\delta)^2}\|h_i(x)\|_2^2$$

Based on Lemma 2.1, then we substitute $x=h_i(\tilde{x})$ and $y=h_i(x)$ and obtain

$$\frac{\|h_i(\tilde{x})\|_2^2}{\|h_i(\tilde{x})\|_2+\delta}-\frac{\|h_i(x)\|_2+2\delta}{2(\|h_i(x)\|_2+\delta)^2}\|h_i(\tilde{x})\|_2^2$$
$$\leq\frac{\|h_i(x)\|_2^2}{\|h_i(x)\|_2+\delta}-\frac{\|h_i(x)\|_2+2\delta}{2(\|h_i(x)\|_2+\delta)^2}\|h_i(x)\|_2^2$$
$$\Rightarrow\sum_i\frac{(1+\delta)\|h_i(\tilde{x})\|_2^2}{\|h_i(\tilde{x})\|_2+\delta}-(1+\delta)\sum_i\frac{\|h_i(x)\|_2+2\delta}{2(\|h_i(x)\|_2+\delta)^2}\|h_i(\tilde{x})\|_2^2$$
$$\leq\sum_i\frac{(1+\delta)\|h_i(x)\|_2^2}{\|h_i(x)\|_2+\delta}-(1+\delta)\sum_i\frac{\|h_i(x)\|_2+2\delta}{2(\|h_i(x)\|_2+\delta)^2}\|h_i(x)\|_2^2$$

By combining the inequalities above, we have

$$f(\tilde{x})+\sum_i\frac{(1+\delta)\,\|h_i(\tilde{x})\|_2^2}{\|h_i(\tilde{x})\|_2+\delta}\leq f(x)+\sum_i\frac{(1+\delta)\,\|h_i(x)\|_2^2}{\|h_i(x)\|_2+\delta}$$

which completes the proof. □

Since the re-weighted dual (5) satisfies the same KKT condition of the original problem (2), the re-weighted problem (5) monotonically converges to a local optimal solution to the original problem (2) according to Theorem 2.2.

3 METHODOLOGY

In this section, we elaborate the details of the proposed robust embedded deep K-means clustering approach (RED-KC). The framework of RED-KC is an auto-encoder network with embedding the robust K-means clustering. With the support of δ-norm distance, RED-KC extracts deep features of the data by mapping them from source space to a latent feature space, such that the weighted cluster centroids can be obtained, namely, a more clear grouping structure can be obtained for the data clustering.

3.1 Robust Embedded Deep K-means Clustering

The neural network of RED-KC consists of $M+1$ layers with M nonlinear transformations, where M is an even number. The first $\frac{M}{2}$ hidden layers are the encoders, which learn a set of compact representations, i.e., dimensionality reduction. The last $\frac{M}{2}$ layers are the

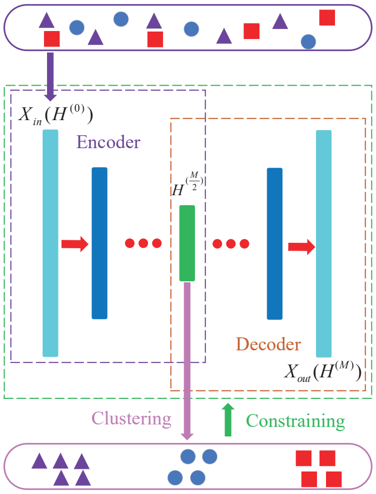


Figure 2. Framework of RED-KC.

decoders, which reconstruct the input. The framework of RED-KC is shown in Figure 2. Suppose $\boldsymbol{H}^{(0)} = \boldsymbol{X}_{in} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N] \in \mathbb{R}^{D \times N}$ as the input matrix of the first layer with N samples, then each data point $\boldsymbol{h}^{(0)}$ is the column of the matrix $\boldsymbol{H}^{(0)}$ with dimension D. As for the encoder, the output of the i-th layer is represented as

$$\boldsymbol{h}^{(i)} = \mathcal{F}_e\left(\boldsymbol{W}^{(i)}\boldsymbol{h}^{(i-1)} + \boldsymbol{b}^{(i)}\right) \in \mathbb{R}^{d_i} \tag{6}$$

where $i = 1, 2, \cdots, \frac{M}{2}$ indexes the layers of the encoder, $\boldsymbol{W}^{(i)} \in \mathbb{R}^{d_i \times d_{i-1}}$ denotes the weight matrix, and $\boldsymbol{b}^{(i)} \in \mathbb{R}^{d_i}$ denotes the bias of the i-th layer. $\mathbb{R}^{d_i}$ indicates that the $\boldsymbol{h}^{(i)}$ belongs to a d_i dimension feature space. The $\mathcal{F}_e(\cdot)$ is a nonlinear activation function. In particular, the $\frac{M}{2}$-th layer $\boldsymbol{h}^{\left(\frac{M}{2}\right)} \in \mathbb{R}^{d_{\frac{M}{2}}}$ is shared by the encoder and the decoder. For the purpose of dimensionality reduction, the dimensions of the layers in the encoder are designed as $D \geq d_{i-1} \geq d_i \geq d_{\frac{M}{2}}$. As for the decoder, the output of the j-th layer can be represented as

$$\boldsymbol{h}^{(j)} = \mathcal{F}_d\left(\boldsymbol{W}^{(j)}\boldsymbol{h}^{(j-1)} + \boldsymbol{b}^{(j)}\right) \in \mathbb{R}^{d_j} \tag{7}$$

where $j = \frac{M}{2} + 1, \frac{M}{2} + 2, \cdots, M$ indexes the layers of the decoder and the nonlinear activation function $\mathcal{F}_d(\cdot)$ can be the same as $\mathcal{F}_e(\cdot)$ or a different nonlinear function. For the purpose of the data reconstruction, the dimensions of the layers in the decoder are designed as $d_{\frac{M}{2}} \leq d_{i-1} \leq d_i \leq d_M = D$. Therefore, given a sample $\boldsymbol{h}^{(0)}$, (i.e., $\boldsymbol{x}_{in}$) as the input of the first layer of RED-KC, $\boldsymbol{h}^{(M)}$, (i.e., $\boldsymbol{x}_{out}$) is the reconstruction of $\boldsymbol{h}^{(0)}$ and the corresponding $\boldsymbol{h}^{\left(\frac{M}{2}\right)}$ is the deep representation of $\boldsymbol{x}_{in}$. Suppose the data matrix $\boldsymbol{H}^{(0)} = \left[\boldsymbol{h}_1^{(0)}, \boldsymbol{h}_2^{(0)}, \cdots, \boldsymbol{h}_N^{(0)}\right] \in \mathbb{R}^{D \times N}$ which denotes a collection of N given samples, then the output matrix of the decoder $\boldsymbol{H}^{(M)} = \left[\boldsymbol{h}_1^{(M)}, \boldsymbol{h}_2^{(M)}, \cdots, \boldsymbol{h}_N^{(M)}\right] \in \mathbb{R}^{D \times N}$ is the corresponding reconstruction of $\boldsymbol{H}^{(0)}$ and $\boldsymbol{H}^{\left(\frac{M}{2}\right)} = \left[\boldsymbol{h}_1^{\left(\frac{M}{2}\right)}, \boldsymbol{h}_2^{\left(\frac{M}{2}\right)}, \cdots, \boldsymbol{h}_N^{\left(\frac{M}{2}\right)}\right] \in \mathbb{R}^{d_{\frac{M}{2}} \times N}$ is the low-dimensional deep representation of $\boldsymbol{H}^{(0)}$.

The objective of RED-KC is to minimize the data reconstruction error and embed the robust K-means clustering with the corresponding deep representation $\boldsymbol{H}^{\left(\frac{M}{2}\right)}$ simultaneously. With the terms previously defined, the objective of RED-KC can be formulated as

$$\begin{aligned} \min_{\boldsymbol{W}^{(m)}, \boldsymbol{b}^{(m)}, \boldsymbol{F}, \boldsymbol{G} \in ind\{0,1\}^{N \times K}} & \underbrace{\frac{1}{2}\left\|\boldsymbol{H}^{(0)} - \boldsymbol{H}^{(M)}\right\|_F^2}_{\mathcal{J}_1} \\ & + \underbrace{\frac{\lambda_1}{2}\left\|\boldsymbol{H}^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{G}^T\right\|_\delta}_{\mathcal{J}_2} \\ & + \underbrace{\frac{\lambda_2}{2}\sum_{m=1}^{M}\left(\left\|\boldsymbol{W}^{(m)}\right\|_F^2 + \left\|\boldsymbol{b}^{(m)}\right\|_2^2\right)}_{\mathcal{J}_3} \end{aligned} \tag{8}$$

where λ_1 and λ_2 are the tradeoff parameters. The terms $\mathcal{J}_1$, $\mathcal{J}_2$, and $\mathcal{J}_3$ are specifically designed for different purposes. As for Eq. (8), the first term $\mathcal{J}_1$ is to preserve the information of the data via the minimization of the reconstruction error. In other words, the input serves as a supervisor of learning a compact representation $\boldsymbol{H}^{\left(\frac{M}{2}\right)}$. Due to the fact that objects in the same cluster tend to have similar features, the term $\mathcal{J}_2$ in (8) is designed to learn the clustering structure from the deep representation $\boldsymbol{H}^{\left(\frac{M}{2}\right)}$ by minimizing the δ-norm error regarding robust K-means, where $\boldsymbol{F} \in \mathbb{R}^{d_{\frac{M}{2}} \times K}$ denotes the matrix of clustering centroids and $\boldsymbol{G} \in \{0, 1\}^{N \times K}$ denotes the binary indicator matrix. In other words, each column of $\boldsymbol{F}$ represents a cluster centroid, while each row $\boldsymbol{g_i}, \forall i$ of $\boldsymbol{G}$ denotes a binary label. As for each row of $\boldsymbol{G}$, the elements of $\boldsymbol{g_i}, \forall i$ contain only one 100% with the others being 0%. The value K denotes the number of clusters. Finally, $\mathcal{J}_3$ serves as a regularization term to avoid over-fitting. Our neural network model utilizes the input as the self-supervisor to learn deep representation and constrain the nonlinear transformation, such that the intrinsic features can be extracted from the source data. Additionally, the δ-norm metric is utilized for the robust K-means clustering. Therefore, the weighted cluster centroids can be achieved in the next subsection, such that a more clear grouping structure can be obtained. Therefore, robust K-means clustering $\mathcal{J}_2$ is embedded with the deep auto-encoder networks such that RED-KC model is proposed in Eq. (8).

3.2 Optimization Procedure

In this subsection, the optimization with respect to (w.r.t.) $\boldsymbol{W}$ and $\boldsymbol{b}$ of the proposed RED-KC model (8) is derived via the gradient descent method, while the solutions w.r.t. $\boldsymbol{G}$ and $\boldsymbol{F}$ to the embedded robust K-means $\mathcal{J}_2$ in (8) are achieved via direct optimization. Since $\boldsymbol{G}$ and $\boldsymbol{F}$ of robust K-means are only involved with the layer $\boldsymbol{H}^{\left(\frac{M}{2}\right)}$,

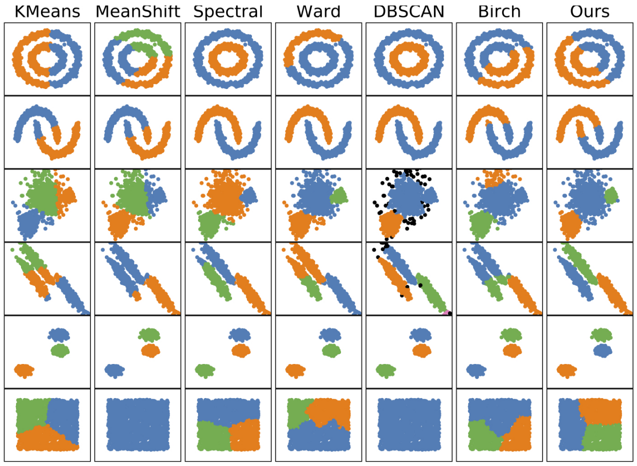


Figure 3. Toy examples

we present the gradient descent and the solutions of $\boldsymbol{G}$ and $\boldsymbol{F}$ separately. According to Eq. (5), the objective function of RED-KC in (8) can be reformulated into the following re-weighted form as

$$\begin{aligned}\mathcal{J} = \frac{1}{2}\sum_{i=1}^{N}\left(\left\|\boldsymbol{h}_i^{(0)} - \boldsymbol{h}_i^{(M)}\right\|_2^2 + \lambda_1 \mathcal{D}_{ii}\left\|\boldsymbol{h}_i^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{g}_i^T\right\|_2^2\right) \\ + \frac{\lambda_2}{2}\sum_{m=1}^{M}\left(\left\|\boldsymbol{W}^{(m)}\right\|_F^2 + \left\|\boldsymbol{b}^{(m)}\right\|_2^2\right)\end{aligned} \tag{9}$$

where the transitional weight $\mathcal{D}_{ii} \leftarrow (1+\delta)\dfrac{\left\|\boldsymbol{h}_i^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{g}_i^T\right\|_2 + 2\delta}{2\left(\left\|\boldsymbol{h}_i^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{g}_i^T\right\|_2 + \delta\right)^2}$.

According to the definitions of the encoder $\boldsymbol{h}^{(i)}$ in (6) and the decoder $\boldsymbol{h}^{(j)}$ in (7), the gradients of Eq. (9) w.r.t. $\boldsymbol{W}^{(m)}$ and $\boldsymbol{b}^{(m)}$ can be obtained via the chain rule as

$$\begin{cases}\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}^{(m)}} = \left(\boldsymbol{\Delta}^{(m)} + \lambda_1 \mathcal{D}_{ii}\boldsymbol{\Lambda}^{(m)}\right)\left(\boldsymbol{h}_i^{(m-1)}\right)^T + \lambda_2 \boldsymbol{W}^{(m)} \\ \frac{\partial \mathcal{J}}{\partial \boldsymbol{b}^{(m)}} = \boldsymbol{\Delta}^{(m)} + \lambda_1 \mathcal{D}_{ii}\boldsymbol{\Lambda}^{(m)} + \lambda_2 \boldsymbol{b}^{(m)}\end{cases} \tag{10}$$

where $\boldsymbol{\Delta}^{(m)}$ and $\boldsymbol{\Lambda}^{(m)}$ are denoted by

$$\begin{cases}\boldsymbol{\Delta}^{(m)} = \begin{cases}-\left(\boldsymbol{h}_i^{(0)} - \boldsymbol{h}_i^{(M)}\right) \odot \mathcal{G}'\left(\boldsymbol{z}_i^{(M)}\right) & m = M \\ \left(\boldsymbol{W}^{(m+1)}\right)^T \boldsymbol{\Delta}^{(m+1)} \odot \mathcal{G}'\left(\boldsymbol{z}_i^{(m)}\right) & \text{otherwise}\end{cases} \\ \boldsymbol{\Lambda}^{(m)} = \begin{cases}\left(\boldsymbol{W}^{(m+1)}\right)^T \boldsymbol{\Lambda}^{(m+1)} \odot \mathcal{G}'\left(\boldsymbol{z}_i^{(m)}\right) & m = 1, \cdots, \frac{M}{2} - 1 \\ \left(\boldsymbol{h}_i^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{g}_i^T\right) \odot \mathcal{G}'\left(\boldsymbol{z}_i^{\left(\frac{M}{2}\right)}\right) & m = \frac{M}{2} \\ \boldsymbol{0} & m = \frac{M}{2} + 1, \cdots, M\end{cases}\end{cases} \tag{11}$$

In Eq. (11), mark $\odot$ denotes the element-wise multiplication operator, $\boldsymbol{z}_i^{(m)} = \boldsymbol{W}^{(m)}\boldsymbol{h}_i^{(m-1)} + \boldsymbol{b}^{(m)}$, and $\mathcal{G}'(\cdot)$ is the derivative of the activation function $\mathcal{G}(\cdot)$ as

$$\mathcal{G}(\cdot) = \begin{cases}\mathcal{F}_e(\cdot) & m = 1, \cdots, \frac{M}{2} \\ \mathcal{F}_d(\cdot) & m = \frac{M}{2} + 1, \cdots, M\end{cases} \tag{12}$$

Via the gradient descent method, $\{\boldsymbol{W}^{(m)}, \boldsymbol{b}^{(m)}\}_{m=1}^{M}$ are further updated by

$$\begin{cases}\boldsymbol{W}^{(m)} \leftarrow \boldsymbol{W}^{(m)} - \mu\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}^{(m)}} \\ \boldsymbol{b}^{(m)} \leftarrow \boldsymbol{b}^{(m)} - \mu\frac{\partial \mathcal{J}}{\partial \boldsymbol{b}^{(m)}}\end{cases} \tag{13}$$

where $\mu > 0$ is the step weight, which can be set as different small values for certain scenario. As the output label of RED-KC model, the indicator matrix $\boldsymbol{G}$ is associated with the robust K-means clustering of $\boldsymbol{H}^{\left(\frac{M}{2}\right)}$. In other words, the binary label $\boldsymbol{G}$ is updated by solving the second clustering term in (9), namely, robust K-means problem. To further obtain the cluster centroid matrix $\boldsymbol{F}$, we rewrite the robust K-means problem in (9) as the following matrix form

$$\min_{\boldsymbol{F}, \boldsymbol{G} \in \{0,1\}^{N \times K}} \left\|\left(\boldsymbol{H}^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{G}^T\right)\mathcal{D}^{\frac{1}{2}}\right\|_F^2 \tag{14}$$

where the weight matrix $\mathcal{D}$ is diagonal with its (i,i)-th entry $\mathcal{D}_{ii}$ defined in (9). According to Eq. (14), the cluster centroid matrix $\boldsymbol{F}$ could be obtained as

$$\begin{aligned}&\frac{\partial\left\|\left(\boldsymbol{H}^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{G}^T\right)\mathcal{D}^{\frac{1}{2}}\right\|_F^2}{\partial \boldsymbol{F}} = 0 \\ &\Rightarrow \frac{\partial Tr\left(\boldsymbol{F}\boldsymbol{G}^T\mathcal{D}\boldsymbol{G}\boldsymbol{F}^T - 2\boldsymbol{H}^{\left(\frac{M}{2}\right)}\mathcal{D}\boldsymbol{G}\boldsymbol{F}^T\right)}{\partial \boldsymbol{F}} = 0 \\ &\Rightarrow \boldsymbol{F} = \boldsymbol{H}^{\left(\frac{M}{2}\right)}\mathcal{D}\boldsymbol{G}(\boldsymbol{G}^T\mathcal{D}\boldsymbol{G})^{-1}\end{aligned} \tag{15}$$

which implies the weighted cluster centroids. In particular, $\boldsymbol{G}$ will be obtained simultaneously via the optimization of the RED-KC model. In sum, the proposed RED-KC method can be summarized in Algorithm 1.

Denote $\mathcal{X}_i$ as the dataset of the i-th class and n_i as the number of data points in the i-th class, then the within-class scatter matrix $\boldsymbol{S}_w$, the between-class scatter matrix $\boldsymbol{S}_b$, and the total-class scatter matrix $\boldsymbol{S}_t$ [44] are defined as

$$\begin{cases}\boldsymbol{S}_w = \sum_{i=1}^{K}\sum_{\boldsymbol{x} \in \mathcal{X}_i}(\boldsymbol{x} - \bar{\boldsymbol{x}}_i)(\boldsymbol{x} - \bar{\boldsymbol{x}}_i)^T \\ \boldsymbol{S}_b = \sum_{i=1}^{K} n_i (\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})^T \\ \boldsymbol{S}_t = \sum_{i=1}^{N}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T\end{cases} \tag{16}$$

where $\bar{\boldsymbol{x}}_i = \frac{1}{n_i}\sum_{\boldsymbol{x}_j \in \mathcal{X}_i}\boldsymbol{x}_j$ is the class-specific mean of the i-th class and $\bar{\boldsymbol{x}} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{x}_i$ is the global mean. According to the definitions in (16), we have the following theorem to illustrate the relationship between robust K-means and within-class scatter.

Theorem 3.1. *If the deep representation of the $\frac{M}{2}$ layer is the centralized data, i.e., $\boldsymbol{H}^{\left(\frac{M}{2}\right)} = \boldsymbol{X}_{in}\boldsymbol{P}$ and the binary label $\boldsymbol{G}$ is known as a prior, i.e., the supervised learning, then the embedded robust K-means in* (14) *is equivalent to $Tr(\boldsymbol{S}_w)$ when $\delta \longrightarrow \infty$.*

Proof. According to the definition of the diagonal matrix $\mathcal{D}$ in (14), its arbitrary (i, i)-th entry can be deduced as

$$\begin{aligned}\lim_{\delta\to\infty} \mathcal{D}_{ii} &= \lim_{\delta\to\infty} (1+\delta)\, \frac{\left\|\boldsymbol{h}_i^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{g}_i^T\right\|_2 + 2\delta}{2\left(\left\|\boldsymbol{h}_i^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{g}_i^T\right\|_2 + \delta\right)^2} \\
&= \lim_{\delta\to\infty} \left(\frac{1}{\delta}+1\right) \frac{\frac{\left\|\boldsymbol{h}_i^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{g}_i^T\right\|_2}{\delta} + 2}{2\left(\frac{\left\|\boldsymbol{h}_i^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{g}_i^T\right\|_2}{\delta} + 1\right)^2} \\
&= (0+1)\frac{0+2}{2(0+1)^2} \\
&= 1\end{aligned}$$

which leads to the conclusion that $\mathcal{D} = \boldsymbol{I}$ when $\delta \to \infty$. Besides, since $\boldsymbol{G}$ is fixed as the prior binary label and $\boldsymbol{H}^{\left(\frac{M}{2}\right)}$ is the the centralized data $\boldsymbol{X}_{in}\boldsymbol{P}$, the embedded robust K-means in (14) degenerates to

$$\min_{\boldsymbol{F}} \left\|\left(\boldsymbol{H}^{\left(\frac{M}{2}\right)} - \boldsymbol{F}\boldsymbol{G}^T\right)\mathcal{D}^{\frac{1}{2}}\right\|_F^2 = \left\|\boldsymbol{X}_{in}\boldsymbol{P} - \boldsymbol{X}_{in}\boldsymbol{P}\boldsymbol{G}\left(\boldsymbol{G}^T\boldsymbol{G}\right)^{-1}\boldsymbol{G}^T\right\|_F^2 \tag{17}$$

where $\boldsymbol{P}$ is the centering matrix. We further define the least squares loss function as

$$\varepsilon = \|\boldsymbol{T}_1 - \boldsymbol{T}_2\|_F^2 \tag{18}$$

and define $\boldsymbol{A}^{(t)} = \frac{1}{N}\boldsymbol{1}\boldsymbol{1}^T$ and $\boldsymbol{A}_{ij}^{(w)} = \begin{cases} \frac{1}{n_i} & i = j \\ 0 & \text{otherwise} \end{cases}$. By substituting $\boldsymbol{T}_1 = \boldsymbol{X}_{in}$ and $\boldsymbol{T}_2 = \boldsymbol{X}_{in}\boldsymbol{A}^{(w)}$ in (18), we have

$$\left\|\boldsymbol{X}_{in} - \boldsymbol{X}_{in}\boldsymbol{A}^{(w)}\right\|_F^2 = Tr\left(\boldsymbol{X}_{in}\left(\boldsymbol{I} - \boldsymbol{A}^{(w)}\right)^2 \boldsymbol{X}_{in}^T\right) = Tr\left(\boldsymbol{S}_w\right) \tag{19}$$

Similarly, by setting $\boldsymbol{T}_1 = \boldsymbol{W}^T\boldsymbol{X}$ and $\boldsymbol{T}_2 = \boldsymbol{W}^T\boldsymbol{X}\boldsymbol{A}^{(t)}$ in (18), we get

$$\left\|\boldsymbol{X}_{in} - \boldsymbol{X}_{in}\boldsymbol{A}^{(t)}\right\|_F^2 = Tr\left(\boldsymbol{X}_{in}\left(\boldsymbol{I} - \boldsymbol{A}^{(t)}\right)^2 \boldsymbol{X}_{in}^T\right) = Tr\left(\boldsymbol{S}_t\right) \tag{20}$$

According to Eqs. (19) and (20), the between-class scatter matrix could be reformulated into

$$\begin{aligned}Tr\left(\boldsymbol{S}_b\right) &= Tr\left(\boldsymbol{S}_t - \boldsymbol{S}_w\right) \\
&= Tr\left(\boldsymbol{X}_{in}\left(\boldsymbol{A}^{(w)} - \boldsymbol{A}^{(t)}\right)\boldsymbol{X}_{in}^T\right) \\
&= Tr\left(\boldsymbol{X}_{in}\left(\boldsymbol{A}^{(w)} - \boldsymbol{A}^{(t)} - \boldsymbol{A}^{(w)}\boldsymbol{A}^{(t)} + \left(\boldsymbol{A}^{(t)}\right)^2\right)\boldsymbol{X}_{in}^T\right) \\
&= Tr\left(\boldsymbol{X}_{in}\left(\boldsymbol{A}^{(w)} - \boldsymbol{A}^{(t)}\right)\left(\boldsymbol{I} - \boldsymbol{A}^{(t)}\right)\boldsymbol{X}_{in}^T\right) \\
&= Tr\left(\boldsymbol{X}_{in}\left(\boldsymbol{A}^{(w)} - \boldsymbol{A}^{(t)}\boldsymbol{A}^{(w)}\right)\left(\boldsymbol{I} - \boldsymbol{A}^{(t)}\right)\boldsymbol{X}_{in}^T\right) \\
&= Tr\left(\boldsymbol{X}_{in}\left(\boldsymbol{I} - \boldsymbol{A}^{(t)}\right)\boldsymbol{A}^{(w)}\left(\boldsymbol{I} - \boldsymbol{A}^{(t)}\right)\boldsymbol{X}_{in}^T\right) \\
&= Tr\left(\boldsymbol{X}_{in}\boldsymbol{P}\boldsymbol{G}\left(\boldsymbol{G}^T\boldsymbol{G}\right)^{-1}\boldsymbol{G}^T\boldsymbol{P}\boldsymbol{X}_{in}^T\right)\end{aligned} \tag{21}$$

Algorithm 1: Robust Embedded Deep K-means Clustering (RED-KC) method

```
Input: data matrix $X_{in}$, (i.e., $H^{(0)}$), parameter $\delta$, and
        number of clusters $K$.
Output: indicator matrix $G$.
Initialize $\mathcal{D} = I$ and random pseudo label matrix $G$;
for $m = 1 : M$ do
    Initialize $W^{(m)}$ and $b^{(m)}$;
end
while not convergence do
    for $i = 1 : \frac{M}{2}$ do
        $h^{(i)} \leftarrow \mathcal{F}_e\left(W^{(i)}h^{(i-1)} + b^{(i)}\right)$;
    end
    for $j = \left(\frac{M}{2} + 1\right) : M$ do
        $h^{(j)} \leftarrow \mathcal{F}_d\left(W^{(j)}h^{(j-1)} + b^{(j)}\right)$;
    end
    Update $F$ by (15);
    for $i = 1 : N$ do
        Update $\mathcal{D}_{ii} \leftarrow (1+\delta)\frac{\left\|h_i^{\left(\frac{M}{2}\right)} - Fg_i^T\right\|_2 + 2\delta}{2\left(\left\|h_i^{\left(\frac{M}{2}\right)} - Fg_i^T\right\|_2 + \delta\right)^2}$;
    end
    Perform robust K-means of $H^{\left(\frac{M}{2}\right)}$ and update $G$;
    for $m = 1 : M$ do
        Update $W^{(m)}$ and $b^{(m)}$ by (13);
    end
end
return $G$
```

where the centering matrix $\boldsymbol{P} = \boldsymbol{I} - \boldsymbol{A}^{(t)}$. According to (21), we have

$$Tr\left(\boldsymbol{S}_w\right) = Tr\left(\boldsymbol{S}_t - \boldsymbol{S}_b\right) = \left\|\boldsymbol{X}_{in}\boldsymbol{P}\left(\boldsymbol{I} - \boldsymbol{G}\left(\boldsymbol{G}^T\boldsymbol{G}\right)^{-1}\boldsymbol{G}^T\right)\right\|_F^2 \tag{22}$$

which equals to Eq. (17). Proof is completed here. □

4 EXPERIMENTS

In this section, we compare the proposed RED-KC approach with the state-of-the-art clustering methods on 4 image datasets in terms of 2 evaluation metrics. In addition, the effectiveness of RED-KC is investigated under different coefficients and activation functions.

4.1 Experimental Settings

4.1.1 Datasets. Four datasets are utilized including COIL20-DSIFT, COIL20-HOG, YaleB-DSIFT, and YaleB-HOG. The COIL20-DSIFT and COIL20-HOG datasets are derived from the DSIFT and HOG feature extraction of the raw COIL20 dataset, respectively. Similarly, the YaleB-DSIFT and YaleB-HOG datasets are generated from the YaleB dataset. COIL-20 is a database of gray-scale images of 20 objects [27]. The objects were placed on a motorized turntable

Table 1. Performance comparison on COIL20 dataset

Features	DSIFT		HOG	
Methods	Accuracy(%)	NMI(%)	Accuracy(%)	NMI(%)
RED-KC	**89.2±2.6**	**93.8±1.9**	**90.6±1.6**	**93.4±2.1**
PARTY	85.7±2.3	91.1±1.8	85.5±1.9	91.9±1,6
AESSC	87.1±2.1	89.9±1.0	84.1±1.9	89.0±1.1
SAEg	65.3±1.2	77.0±1.1	74.9±1.0	89.2±1.6
SAEs	56.5±1.5	65.0±0.4	71.9±1.4	87.0±1.8
SSC	84.3±2.2	91.0±0.7	81.0±1.5	90.1±1.2
KSSC1	82.4±1.2	90.3±1.1	70.9±0.5	84.0±0.4
KSSC2	76.4±2.6	90.1±0.2	75.1±0.8	86.5±0.4
LS3C	30.9±3.3	49.2±1.8	30.3±2.1	40.5±0.7
LRR	79.0±1.4	89.7±1.2	58.4±3.2	76.9±1.6
KLRR1	70.2±1.8	81.4±0.6	73.7±4.0	81.2±1.3
KLRR2	78.5±1.3	83.8±0.9	74.1±0.9	83.8±0.6
LRSC	71.1±1.7	78.3±0.7	44.0±1.2	57.2±1.4
LSR1	61.5±1.3	71.2±0.8	64.7±1.4	73.0±1.0
LSR2	64.7±2.0	72.7±0.2	61.7±1.5	71.1±0.9
SMR	80.4±1.9	89.4±0.5	74.8±2.6	84.4±1.2

Table 2. Performance comparison on YaleB dataset

Features	DSIFT		HOG	
Methods	Accuracy(%)	NMI(%)	Accuracy(%)	NMI(%)
RED-KC	**89.7±2.3**	**92.7±2.6**	**94.2±1.0**	**98.4±2.9**
PARTY	88.5±2.5	90.8±0.8	92.0±1.1	96.9±1.5
AESSC	74.8±2.6	78.3±0.9	88.8±0.6	94.4±0.5
SAEg	82.3±0.8	87.5±0.9	84.7±0.4	93.4±0.8
SAEs	80.7±1.1	85.9±0.5	81.4±0.6	92.4±0.4
SSC	83.7±1.7	90.0±0.4	85.1±1.1	92.8±1.1
KSSC1	91.4±1.2	89.0±0.4	80.5±1.2	88.6±0.3
KSSC2	77.6±1.0	84.4±0.6	75.3±0.8	80.3±0.4
LS3C	49.9±1.4	59.8±0.5	49.1±0.4	53.5±0.2
LRR	81.6±0.3	89.1±0.4	81.0±0.1	93.0±0.5
KLRR1	69.9±0.6	74.7±0.2	78.9±1.3	86.1±0.5
KLRR2	66.1±1.1	72.3±0.4	60.1±0.6	68.9±0.5
LRSC	68.2±1.3	73.4±0.2	68.6±0.5	73.2±0.5
LSR1	72.8±0.6	77.6±0.7	76.5±1.0	81.0±0.6
LSR2	73.3±1.2	77.4±0.5	76.0±1.2	80.4±0.5
SMR	81.4±1.3	85.2±0.8	87.9±0.9	92.7±0.8

against a black background. The turntable was rotated through 360 degrees to vary object pose with respect to a fixed camera. The COIL20 dataset contains 1,440 samples where each image is with the size of 32 × 32. The YaleB dataset [19] consists of 5760 samples from 38 human subjects under 9 poses and 64 illumination conditions, where each image is with size of 192 × 168. More detail information of datasets can be found in Table 3. For the computational efficiency, we use PCA to reduce the feature dimension to 300.

4.1.2 Evaluation Criteria. Two metrics are adopted to evaluate the clustering quality: Accuracy and normalized mutual information(NMI).

Clustering Accuracy reflects the relationship between clusters and classes by measuring the degree that each cluster contains the number of data samples from the related class. The clustering accuracy is calculated by $\boldsymbol{Accuracy} = \frac{\sum_{i=1}^{N} \xi(\boldsymbol{map}(\boldsymbol{r}_i), \boldsymbol{l}_i)}{N}$, where $\boldsymbol{r}_i$ represents the pseudo-cluster label of $\boldsymbol{x}_i$, $\boldsymbol{l}_i$ represents the true class label, $\xi(x, y)$ is the delta function, and $\boldsymbol{map}(\cdot)$ is the optimal map function. Note $\xi(x, y) = 1$, if $x = y$; $\xi(x, y) = 0$, otherwise. The map function $\boldsymbol{map}(\cdot)$ projects each cluster label to the true label. A larger $\boldsymbol{Accuracy}$ implies a better clustering performance.

The Normalized Mutual Information serves as an index to determine the consistent quality of cluster, which is defined as $\boldsymbol{NMI} = \frac{\sum_{i=1}^{K} \sum_{j=1}^{K} n_{ij} \log \frac{n_{ij}}{n_i \hat{n}_j}}{\sqrt{(\sum_{i=1}^{K} n_i \log \frac{n_i}{N})(\sum_{j=1}^{K} \hat{n}_j \log \frac{\hat{n}_j}{N})}}$, where n_i denotes the number of data in the cluster $\boldsymbol{C}_i$ $(1 \leq i \leq K)$, $\hat{n}_j$ denotes the number of data belonging to the class L_j $(1 \leq j \leq K)$, and n_{ij} is the number of data which are in the intersection between cluster $\boldsymbol{C}_i$ and class $\boldsymbol{L}_j$. Similarly, a larger $\boldsymbol{NMI}$ represents a more consistent clustering performance.

4.1.3 Toy results. In Figure 3, 6 types of toy data are compared under 7 different clustering approaches. From Figure 3, we could observe that our method has much better performance on multi-class toy dataset instead of binary-class toy dataset. When dealing with multi-class toy dataset, our method outperforms other comparative methods.

Table 3. Information of the datasets.

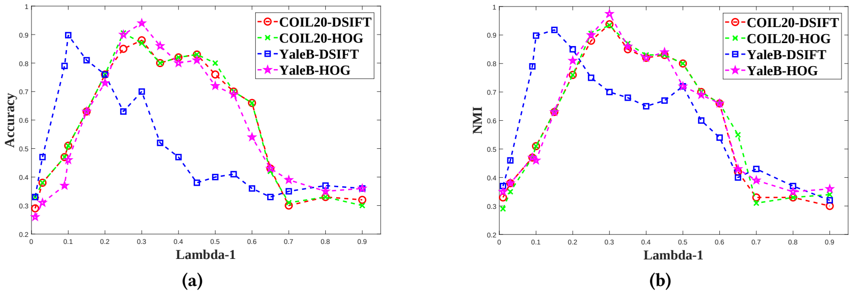
Dataset	size	dimensionality	class
COIL20	1440	1024	20
YALEB	5760	32256	38

4.1.4 Baseline Algorithms. The proposed RED-KC is compared with the clustering algorithms on four datasets as COIL20-DSIFT, COIL20-HOG, YaleB-DSIFT, and YaleB-HOG. The comparative methods include auto-encoder based subspace clustering algorithms (AESSC), deep subspace clustering with sparsity Prior (PARTY), sparse subspace clustering (SSC), low rank based subspace clustering (LRSC), least square regression (LSR), smooth representation clustering (SMR), kernel SSC (KSSC), kernel LRR (KLRR), latent subspace sparse subspace clustering (LS3C), and stacked sparse auto-encoder (SAE), where AESSC and PARTY are the deep clustering methods.

Particularly, the proposed RED-KC is designed as a five layer neural network structure, which consists of 300−200−100−200−300 neurons. To ensure a fair comparison, we report the best results of all the comparative methods with setting K as the class number of each dataset. As for the tradeoff parameters λ_1 and λ_2 in the proposed RED-KC method, we tune them via grid search in the set of $\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10^1, 10^2, 10^3\}$. As for the δ in the robust loss, we tune it in the grid of $\{10^{-2}, 10^{-1}, 1, 10^1, 10^2\}$.

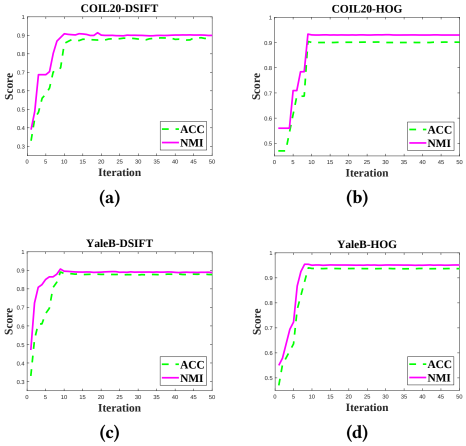
4.2 Comparison with the Evaluated Methods

In this subsection, we evaluate the performance of RED-KC by comparing with the baseline algorithms. In both Table 1 and Table



(a) (b)

Figure 4. Accuracy and NMI of RED-KC under different values of λ_1.



(a) (b) (c) (d)

Figure 5. Variations of Accuracy and NMI by increasing the iteration numbers for RED-KC.

2, the bolded names denote the state-of-the-art deep clustering methods, while the bolded scores denote the best results. From Table 1, the proposed RED-KC has the better performance. The Accuracy of RED-KC is at least 4.46% and 5.06% higher than other methods on COIL20-DISFT and COIL20-HOG, respectively. From Table 2, RED-KC achieves the best results, where the Accuracy is 1.24% and 1.94% higher than the runner-up method on YaleB-DSIFT and YaleB-HOG, respectively.

Besides that, the results demonstrate that deep clustering methods perform much better due to the deep feature extraction. Figure 5 shows the Accuracy and NMI of RED-KC under different iteration numbers. We could observe that the performance is enhanced rapidly within the first ten iterations, which indicates the efficiency of our method. After several iterations, both Accuracy and NMI remain stable.

4.3 Influence of Tradeoff Coefficient

As for the tradeoff coefficient λ_1, we investigate the variations of Accuracy and NMI under different values of λ_1 as shown in Figure 4. Since the coefficient λ_2 is to prevent over-fitting of RED-KC, i.e., insensitive to the clustering accuracy and NMI indexes, λ_2 is fixed as 1 in this case. From Figure 4, we notice that the evaluation indexes fluctuate according to λ_1. Moreover, the optimal performance is achieved near the value $\lambda_1 = 0.3$ with much larger probability.

4.4 Influence of Activation Functions

In this subsection, we report the performance of RED-KC under four different activation functions including *Tanh*, *Sigmoid*, *Nssigmoid*, and *Softplus*. From Figure 6, we can see that the *Tanh* function outperforms the other three activation functions and the *Nssigmoid* function achieves the runner-up results, which are very close to *Tanh*.

5 RELATED WORKS

In this section, we briefly introduce the related works regarding unsupervised deep learning and subspace clustering respectively.

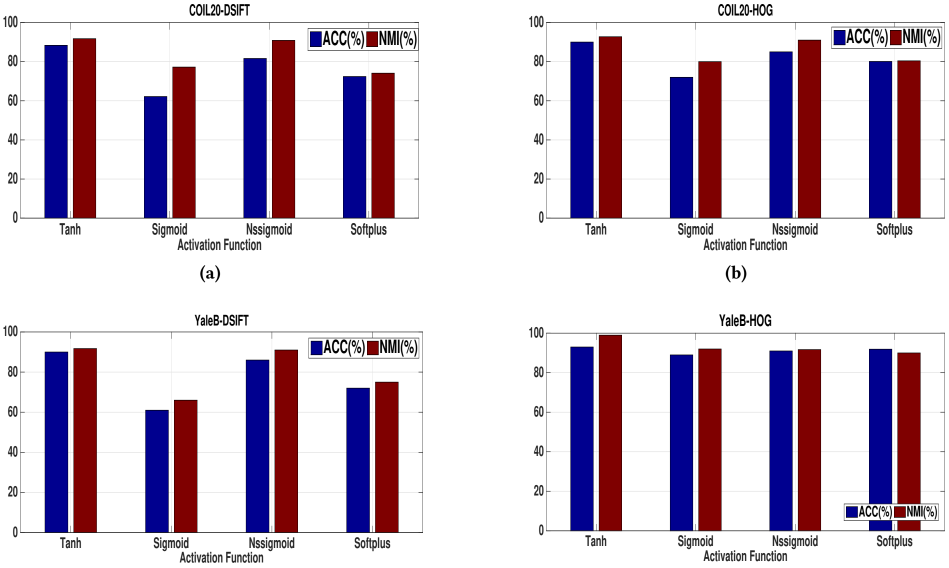
5.1 Auto-encoder Network

With impressive learning capabilities, deep learning techniques have achieved great success in diverse areas, especially in the field of supervised learning [29], such as image classification [11, 16], metric learning [13, 25], super-resolution reconstruction [5, 18], and image segmentation [2, 3, 24]. Meanwhile, the unsupervised deep learning is still under the development. Auto-encoder and generative adversarial network are the state-of-the-art methods for unsupervised deep learning. In this subsection, we mainly introduce the auto-encoder network.

In general, auto-encoder [37] serves as a network which consists of both encoder and decoder, where the structure of auto-encoder is symmetric. If the auto-encoder contains multiple hidden layers, then the number of hidden layers of the encoder equals to the number of hidden layers of the decoder. In other words, the purpose of the basic auto-encoder is to reconstruct the input data at the output layer. In particular, the encoding and decoding process can be described as

$$\begin{aligned} &\text{Encoding} \qquad \boldsymbol{h}^{(i+1)} = \mathcal{F}_e\left(\boldsymbol{W}^{(i)}\boldsymbol{h}^{(i)} + \boldsymbol{b}^{(i)}\right) \\ &\text{Decoding} \qquad \boldsymbol{h}^{(j+1)} = \mathcal{F}_d\left(\boldsymbol{W}^{(j)}\boldsymbol{h}^{(j)} + \boldsymbol{b}^{(j)}\right) \end{aligned} \tag{23}$$

where *Sigmoid*, *Tanh*, and *Relu* are the common activation functions for $\mathcal{F}_e$. As for $\mathcal{F}_d$, it could be the same as the encoding function. Therefore, the loss function of the basic auto-encoder is to minimize the error between $\boldsymbol{X}_{in}$ and $\boldsymbol{X}_{out}$. Specifically speaking, the encoder



(a)

(b)

(c)

(d)

Figure 6. The performance of RED-KC under four different activation functions.

converts the input signal into codes via the nonlinear mapping, while the decoder is to reconstruct the codes to the input signal.

It is easy to observe that both the encoding and decoding process would not depend on the label information. Therefore, the auto-encoder serves as an unsupervised learning method [33]. Moreover, the hidden layers of the automatic coding network can be categorized into three classes including the compressed structure, the sparse structure, and the equivalent-dimensional structure. When the number of input layer neurons is greater than the number of hidden layer neurons, it is known as the compressed structure. Conversely, when the number of input layer neurons is smaller than the number of hidden layer neurons, it is named the sparse structure. If the input layer and the hidden layer have the same number of neurons, it is called the equivalent-dimensional structure.

5.2 Deep Subspace Clustering

Diverse subspace clustering algorithms [7, 23, 32, 34, 41] are the linear models, which are unable to cope with the nonlinearity of the data in the practical scenarios. Benefited from the powerful capability of nonlinear modeling and feature extraction of the deep neural network, multiple deep clustering approaches have been developed in recent years. For instance, Song et al. [37] integrated an auto-encoder network with K-means to learn the latent features. Since the feature mapping and clustering are independent, the K-means algorithm is frequently separated from the feature mapping process. Therefore, the features extracted from the deep network may not be suitable for clustering. To address this issue, some deep clustering algorithms [40, 50] incorporated the discriminant and adversarial ideas. Due to lack of prior knowledge constraints, the feature mapping of these algorithms is weakened. In addition, a deep subspace clustering with sparsity prior (PARTY) [33] is developed by Peng et al. Based on an auto-encoder network, PARTY learns the deep representation of the input data via reconstruction error minimization and utilizes a prior information to preserve the sparse reconstruction. However, PARTY method has the following drawbacks: 1) The sparsity prior matrix needs to be pre-trained, which might not be optimal for clustering. 2) The graph matrix is pre-given as a prior such that data structure is fixed in the network.

Different from the existing works, our method embeds the robust K-means with an auto-encoder network, where the deep feature extraction and clustering can be performed simultaneously. The proposed RED-KC approach utilizes δ-norm distance to constrain the feature mapping so that the deep features extracted from the source space are more conducive to robust clustering.

6 CONCLUSION

In this paper, we proposed a robust embedded deep K-means clustering approach, which utilizes robust δ-norm metric to constrain the feature mapping process of the auto-encoder network, so that data are mapped to a latent feature space for the robust clustering. More importantly, the proposed method embeds the clustering process with the auto-encoder network, such that we can perform deep feature extraction and clustering simultaneously. Therefore, the proposed method accomplished the clustering within only one step to avoid losing pivotal information and correlation. In other words, a more clear grouping structure can be achieved for the data clustering with obtaining the weighted cluster centroids. Eventually, extensive experiments are provided to show that our method outperforms the state-of-the-art clustering methods.

7 ACKNOWLEDGEMENT

We acknowledge that *Arizona State University* and *University of Illinois at Urbana-Champaign* are partially supported by NSF (IIS-1651206, IIS-1715385) and DHS (2017-ST-061-QA0001).

REFERENCES

[1] Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. 2017. Deep adaptive image clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 5879–5887.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. 2018. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 4 (2018), 834–848.

[3] Bin Cheng, Guangcan Liu, Jingdong Wang, Zhongyang Huang, and Shuicheng Yan. 2011. Multi-task low-rank affinity pursuit for image segmentation. In *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2439–2446.

[4] Nat Dilokthanakul, Pedro AM Mediano, Marta Garnelo, Matthew CH Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. 2016. Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648* (2016).

[5] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. 2014. Learning a deep convolutional network for image super-resolution. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 184–199.

[6] Aysegul Dundar, Jonghoon Jin, and Eugenio Culurciello. 2015. Convolutional clustering for unsupervised learning. *arXiv preprint arXiv:1511.06241* (2015).

[7] Ehsan Elhamifar and Rene Vidal. 2013. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 11 (2013), 2765–2781.

[8] Adil Fahad, Najlaa Alshatri, Zahir Tari, Abdullah Alamri, Ibrahim Khalil, Albert Y. Zomaya, Sebti Foufou, and Abdelaziz Bouras. 2014. A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE Transactions on Emerging Topics in Computing* 2, 3 (2014), 267–279.

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*. 2672–2680.

[10] Muhan Guo, Rui Zhang, Feiping Nie, and Xuelong Li. 2018. Embedding fuzzy k-means with nonnegative spectral clustering via incorporating side information. In *ACM International Conference on Information and Knowledge Management (CIKM)*. 1567–1570.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778.

[12] John R. Hershey, Zhuo Chen, Jonathan Le Roux, and Shinji Watanabe. 2016. Deep clustering: Discriminative embeddings for segmentation and separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 31–35.

[13] Junlin Hu, Jiwen Lu, and Yap-Peng Tan. 2014. Discriminative deep metric learning for face verification in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1875–1882.

[14] Pan Ji, Mathieu Salzmann, and Hongdong Li. 2014. Efficient dense subspace clustering. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 461–468.

[15] Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian Reid. 2017. Deep subspace clustering networks. In *Advances in Neural Information Processing Systems (NIPS)*. 24–33.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*. 1097–1105.

[17] Marc T. Law, Raquel Urtasun, and Richard S. Zemel. 2017. Deep spectral clustering learning. In *International Conference on Machine Learning (ICML)*. 1985–1994.

[18] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, and Zehan Wang. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 105–114.

[19] Kuang-Chih Lee, Jeffrey Ho, and David J. Kriegman. 2005. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27, 5 (2005), 684–698.

[20] Xuelong Li, Han Zhang, Rui Zhang, Yun Liu, and Feiping Nie. 2019. Generalized Uncorrelated Regression with Adaptive Graph for Unsupervised Feature Selection. *IEEE Transactions on Neural Networks and Learning Systems* 30, 5 (2019), 1587–1595.

[21] Jie Liang, Jufeng Yang, Hsin-Ying Lee, Kai Wang, and Ming-Hsuan Yang. 2018. Sub-GAN: An unsupervised generative model via subspaces. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 698–714.

[22] Wei-An Lin, Jun-Cheng Chen, Carlos D. Castillo, and Rama Chellappa. 2018. Deep density clustering of unconstrained faces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 8128–8137.

[23] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. 2013. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 1 (2013), 171–184.

[24] Can-Yi Lu, Hai Min, Zhong-Qiu Zhao, Lin Zhu, De-Shuang Huang, and Shuicheng Yan. 2012. Robust and efficient subspace segmentation via least squares regression. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 347–360.

[25] Benjamin J. Meyer, Ben Harwood, and Tom Drummond. 2018. Deep metric learning and image classification with nearest neighbour gaussian kernels. In *IEEE International Conference on Image Processing (ICIP)*. IEEE, 151–155.

[26] Erxue Min, Xifeng Guo, Qiang Liu, Gen Zhang, Jianjing Cui, and Jun Long. 2018. A survey of clustering with deep learning: From the perspective of network architecture. *IEEE Access* 6 (2018), 39501–39514.

[27] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. 1996. Columbia object image library (coil-20). (1996).

[28] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. 2002. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems (NIPS)*. 849–856.

[29] Feiping Nie, Zhanxuan Hu, and Xuelong Li. 2018. Calibrated multi-task learning. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. 2012–2021.

[30] Feiping Nie, Sheng Yang, Rui Zhang, and Xuelong Li. 2019. A General Framework for Auto-Weighted Feature Selection via Global Redundancy Minimization. *IEEE Transactions on Image Processing* 28, 5 (2019), 2428–2438.

[31] Vishal M. Patel, Hien Van Nguyen, and René Vidal. 2013. Latent space sparse subspace clustering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 225–232.

[32] Vishal M. Patel and René Vidal. 2014. Kernel sparse subspace clustering. In *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2849–2853.

[33] Xi Peng, Shijie Xiao, Jiashi Feng, Wei-Yun Yau, and Zhang Yi. 2016. Deep subspace clustering with sparsity prior.. In *International Joint Conferences on Artificial Intelligence (IJCAI)*. 1925–1931.

[34] Xi Peng, Zhang Yi, and Huajin Tang. 2015. Robust subspace clustering via thresholding ridge regression.. In *AAAI Conference on Artificial Intelligence (AAAI)*. 3827–3833.

[35] Alex Rodriguez and Alessandro Laio. 2014. Clustering by fast search and find of density peaks. *Science* 344, 6191 (2014), 1492–1496.

[36] Jianbo Shi and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, 8 (2000), 888–905.

[37] Chunfeng Song, Feng Liu, Yongzhen Huang, Liang Wang, and Tieniu Tan. 2013. Auto-encoder based data clustering. In *Iberoamerican Congress on Pattern Recognition*. Springer, 117–124.

[38] Ulrike Von Luxburg. 2007. A tutorial on spectral clustering. *Statistics and Computing* 17, 4 (2007), 395–416.

[39] Wutao Wei, Bowei Xi, and Murat Kantarcioglu. 2018. Adversarial clustering: A grid based clustering algorithm against active adversaries. *arXiv preprint arXiv:1804.04780* (2018).

[40] Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *International Conference on Machine Learning (ICML)*. 478–487.

[41] Ming Yin, Yi Guo, Junbin Gao, Zhaoshui He, and Shengli Xie. 2016. Kernel sparse subspace clustering on symmetric positive definite manifolds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 5157–5164.

[42] Hongyang Zhang, Zhouchen Lin, Chao Zhang, and Junbin Gao. 2014. Robust latent low rank representation for subspace clustering. *Neurocomputing* 145 (2014), 369–373.

[43] Rui Zhang, Feiping Nie, Muhan Guo, Xian Wei, and Xuelong Li. 2019. Joint learning of fuzzy k-means and nonnegative spectral clustering with side information. *IEEE Transactions on Image Processing* 28, 5 (2019), 2152–2162.

[44] Rui Zhang, Feiping Nie, and Xuelong Li. 2017. Embedded clustering via robust orthogonal least square discriminant analysis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2332–2336.

[45] Rui Zhang, Feiping Nie, and Xuelong Li. 2017. Self-weighted spectral clustering with parameter-free constraint. *Neurocomputing* 241 (2017), 164–170.

[46] Rui Zhang, Feiping Nie, and Xuelong Li. 2018. Self-weighted supervised discriminative feature selection. *IEEE Transactions on Neural Networks and Learning Systems* 29, 8 (2018), 3913–3918.

[47] Rui Zhang, Feiping Nie, and Xuelong Li. 2019. Semisupervised learning with parameter-free similarity of label and side information. *IEEE Transactions on Neural Networks and Learning Systems* 30, 2 (2019), 405–414.

[48] Rui Zhang, Feiping Nie, Xuelong Li, and Xian Wei. 2019. Feature selection with multi-view data: A survey. *Information Fusion* 50 (2019), 158–167.

[49] Rui Zhang, Feiping Nie, Haiyang Wang, and Xuelong Li. 2019. Unsupervised Feature Selection via Adaptive Multi-Measure Fusion. *IEEE Transactions on Neural Networks and Learning Systems* (2019).

[50] Pan Zhou, Yunqing Hou, and Jiashi Feng. 2018. Deep adversarial subspace clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1596–1604.