Deep Embedding Clustering Driven by Sample Stability

Zhanwen Cheng[†] , **Feijiang Li**[†] , **Jieting Wang** and **Yuhua Qian**[*]
Institute of Big Data Science and Industry, Shanxi University, Taiyuan 030006, China.
zhanwen_cheng@126.com, fjli@sxu.edu.cn, jtwang@sxu.edu.cn, jinchengqyh@126.com

Abstract

Deep clustering methods improve the performance of clustering tasks by jointly optimizing deep representation learning and clustering. While numerous deep clustering algorithms have been proposed, most of them rely on artificially constructed pseudo targets for performing clustering. This construction process requires some prior knowledge, and it is challenging to determine a suitable pseudo target for clustering. To address this issue, we propose a deep embedding clustering algorithm driven by sample stability (DECS), which eliminates the requirement of pseudo targets. Specifically, we start by constructing the initial feature space with an autoencoder and then learn the cluster-oriented embedding feature constrained by sample stability. The sample stability aims to explore the deterministic relationship between samples and all cluster centroids, pulling samples to their respective clusters and keeping them away from other clusters with high determinacy. We analyzed the convergence of the loss using Lipschitz continuity in theory, which verifies the validity of the model. The experimental results on five datasets illustrate that the proposed method achieves superior performance compared to state-of-the-art clustering approaches.

1 Introduction

Clustering [Xu and Wunsch, 2005], one of the most crucial tasks in machine learning, aims to group similar samples into the same cluster while separating dissimilar ones into different clusters. Traditional clustering methods such as k-means [MacQueen, 1967b], spectral clustering [Ng *et al.*, 2001; Yang *et al.*, 2018], Gaussian mixture model [Bishop and Nasrabadi, 2006; Reynolds, 2009] and hierarchical clustering [Sneath and Sokal, 1962; Johnson, 1967; Koga *et al.*, 2007] have achieved tremendous success over the past decades. However, these methods depend on manually extracted features, making them impractical for high-dimensional and unstructured data. Benefiting from the development of deep representation learning, deep clustering arises and has attracted increasing attention recently.

The existing deep clustering methods can be roughly categorized into three types: First, the pseudo labeling deep clustering method [Niu *et al.*, 2020; Niu *et al.*, 2022] filters out a subset of samples with high confidence and trains in a supervised manner, yet, the performance of this method heavily relies on the quality of the filtered pseudo labels, which is susceptible to model capability and hyper-parameter tuning. Second, the self-training deep clustering method [Xie *et al.*, 2016; Guo *et al.*, 2017a] optimizes the distribution of cluster assignments by minimizing the KL-divergence between the assignment distribution and an auxiliary distribution, but the performance of this method is limited by the construction method of the auxiliary distribution. Third, the contrastive deep clustering method [Jaiswal *et al.*, 2020; Jing and Tian, 2020] aims to pull the positive pairs close while pushing the negative pairs far away, this method relies on the construction approach of positive and negative sample pairs. In summary, despite the proposal of numerous excellent deep clustering methods, most of them rely on artificially constructed pseudo targets that require prior knowledge and may heavily impact the clustering results.

In this paper, inspired by traditional clustering methods based on sample stability [Li *et al.*, 2019; Li *et al.*, 2020], we propose a deep embedding clustering algorithm driven by sample stability (DECS). Different from prior methods, our method eliminates the requirement of a pseudo target, and clustering using sample stability as a constraint.

Specifically, our method consists of two stages: representation learning and clustering. In the representation learning stage, we employ a convolutional autoencoder [Guo *et al.*, 2017b] to map the raw data into a latent embedding space that captures the underlying structure of the data. This is achieved by minimizing the reconstruction loss, ensuring that the learned feature representations preserve the essential information from the original input. Subsequently, in the clustering stage, we retain the encoder module of the autoencoder and compute the soft assignment probabilities of each sample to all cluster centroids based on the learned embedding representations, which we refer to as co-association probability. Then, we compute the level of determinacy for each sample with respect to all cluster centers and further calculate the stability of all samples by considering their determinacy levels

[*]The corresponding author

regarding each cluster center. To the best of our knowledge, our method is the first to utilize the deterministic relationship between samples and centroids for clustering, and innovatively employ the instability of samples as a loss to optimize the parameters of deep neural networks. In summary, the main contributions of this work are as follows:

- The concept of sample stability is extended into deep clustering, and a novel loss function that effectively captures both intra-class and inter-class relationships among the samples is proposed. This approach is then applied in a joint learning framework, which comprises an autoencoder and a clustering layer.
- The model's convergence is theoretically analyzed, providing evidence that clustering with internal sample relationship driven by sample stability can indeed converge.
- Experiments are conducted on five image datasets to validate the effectiveness of our method. The experimental results demonstrate that our approach outperforms the state-of-the-art methods.

2 Related Work

This work is closely related to convolutional autoencoder and sample stability, which are briefly introduced in this section.

2.1 Convolutional Autoencoder

Autoencoder is an unsupervised neural network model widely used for tasks like data dimensionality reduction and feature extraction. Deep Embedded Clustering (DEC) [Xie *et al.*, 2016] was the pioneer in utilizing denoising autoencoders for joint learning of feature representations and cluster assignments. Subsequent works, such as IDEC [Guo *et al.*, 2017a], FCDEC-DA [Guo *et al.*, 2018], SDEC [Ren *et al.*, 2019], and others, have built upon DEC's autoencoder framework, and achieved remarkable clustering results.

Due to the limited ability of fully connected layers in extracting features from high-dimensional data, such as images, convolutional autoencoder [Guo *et al.*, 2017b] was proposed by incorporating Convolutional Neural Networks (CNNs) into autoencoders, which showed improved adaptation to image-related tasks. DEPICT [Ghasedi Dizaji *et al.*, 2017], ConvDEC-DA [Guo *et al.*, 2018], DBC [Li *et al.*, 2018], StatDEC [Rezaei *et al.*, 2021], and so on, have adopted convolutional autoencoders instead of fully connected autoencoders in order to learn feature representations and achieve superior clustering results.

2.2 Sample Stability

The concept of sample stability [Li *et al.*, 2019] was first proposed in clustering ensembles. Clustering methods based on sample stability aim to explore the indeterminacy of sample relationships and identify sets of samples with stable relationships. These approaches leverage pairwise relationships between samples for clustering, which reduces the impact of indeterminate relationships among samples.

Given a set of clustering results, the co-association probability between two samples can be represented by their frequency of belonging to the same cluster based on their similarity. A co-association probability value of one indicates high determinacy that the samples belong to the same cluster, while a value of zero indicates high determinacy that they do not belong to the same cluster. However, when the value falls between zero and one, it becomes difficult to definitively determine whether the two samples belong to the same cluster, resulting in a low determinacy. Due to the insufficiency of using co-association probabilities alone in reflecting the level of determinacy regarding whether two samples belong to the same cluster, the determinacy function [Li *et al.*, 2019] was proposed to evaluate the level of determinacy between two samples. Then, the stability of sample x_i is defined as the average level of determinacy among sample x_i and all other samples based on their co-association probability values:

$$sq\left(x_i\right) = \frac{1}{n}\sum_{j=1}^{n} fq\left(p_{ij}\right)$$

where n represents the number of samples in the dataset, and $p_{ij} \in P, P = \{p_{ij} | 1 \leq i \leq n, 1 \leq j \leq n\}$ denote the co-association probability of sample x_i and x_j, and $fq\left(\cdot\right)$ denote the determinacy function.

Subsequently, SSC [Li *et al.*, 2020] extended the concept of sample stability from cluster ensembles to clustering analysis and proposed a new function for measuring sample stability in cluster analysis, and the theoretical validity of sample stability was analyzed in this work.

Although the clustering methods based on sample stability [Li *et al.*, 2019; Li *et al.*, 2020] provide new ideas for clustering, these methods are heuristic approaches that only use sample stability as an evaluation function. However, utilizing sample's stability to guide clustering optimization has not been well studied.

3 Method

In this section, we present the proposed Deep Embedding Clustering Driven by Sample Stability (DECS) model. Our model first trains a convolutional autoencoder and then utilizes sample stability as guidance to accomplish clustering. Fig.1 illustrates the overall process.

3.1 Problem Formulation

In this paper, we aim to cluster a set of n samples $X = \{x_i\}_{i=1}^{n}$ from the input space $X \in R^d$ into k classes using a clustering network. Distinguishing with the prior works, we reconsider the problem of clustering in deep neural networks by introducing constraints on the relationships among samples, and make the first attempt to reduce the calculation of sample stability from n^2 to kn. Our method first employs an autoencoder to map the sample set X into a representation space, and then utilizes sample stability as a constraint to achieve sample clustering.

To this end, the objective function of our framework can be formulated as:

$$L = L_r + L_c, \tag{1}$$

where L_r and L_c represent the reconstruction loss and clustering loss, respectively.

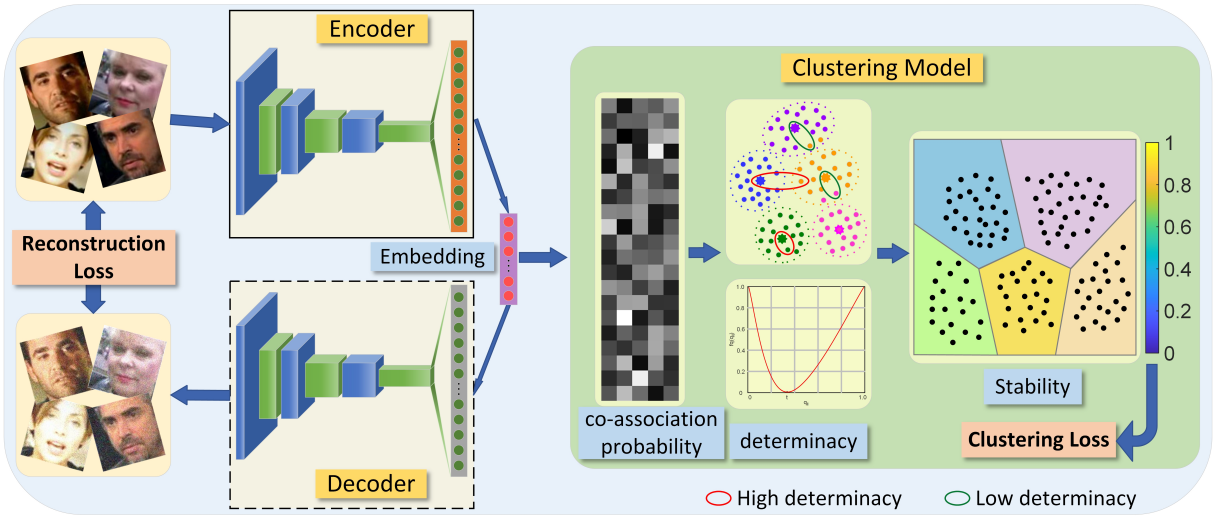


Figure 1: Pipeline of the proposed DECS. We first train an autoencoder consisting of an encoder and a decoder to embed the inputs into a latent space and reconstruct the input samples using their latent representations. The reconstruction loss is utilized to learn discriminative information from the inputs. Then, we discard the decoder and jointly optimize the encoder and clustering model to get the clustering results.

3.2 Extract Features with Convolutional Autoencoder

To accurately represent each sample with an embedding vector, we employ a convolutional autoencoder as the feature extractor, an encoder f_{θ_e} is used to map a sample $x_i \in X$ to its latent embedding vector $z_i \in Z$, while a decoder g_{θ_d} reconstructs x_i from its embedding vector z_i. To be specific, given a set of samples $X = \{x_i \in R^d\}_{i=1}^n$, a random transformation T_{random} is applied to each sample x_i to obtain the augmented sample $x_i' = T_{random}(x_i)$, and then serve as the inputs of convolutional autoencoder, which extracts the latent embedding representation of each sample from its internal layers by minimizing the reconstruction loss:

$$L_r = \frac{1}{n}\sum_{i=1}^{n} \left\| g_{\theta_d}\left(f_{\theta_e}\left(x_i'\right)\right) - x_i' \right\|_2^2, \tag{2}$$

where n represent the number of samples.

The convolutional encoder is utilized to capture the essential information of the samples, while a convolutional decoder is employed to validate and enhance the representation ability of the embedding vectors. This process can be expressed as:

$$f_{\theta_e} = \sigma\left(\sum_{i\in H} x' * W_i + b_i\right), \tag{3}$$

$$g_{\theta_d} = \sigma\left(\sum_{i\in H} z_i * \tilde{W}_i + c_i\right), \tag{4}$$

where θ_e and θ_d represent the parameters of the convolutional encoder and decoder, respectively, σ is the activation function such as ReLU, H denotes the group of latent feature maps, W_i and b_i correspond to the filter and bias of the i^{th} feature map in the encoder, similarly, $\tilde{W}$ and c_i are the corresponding parameter in the decoder, and $*$ denotes the convolution operation.

3.3 Clustering with Sample Stability

In the clustering stage, we utilize the encoder trained in the previous stage as the feature extractor and then fine-tune the encoder using sample stability as guidance, which ensures that it learns cluster-oriented sample representations. For the sake of writing and understanding, we consider only a single sample and describe the computational processes in vector form in the following description. We introduce the notation $\mathbf{I} \in R^{1\times k}$ as a row vector with all ones, and T denotes the transpose of a vector.

In the clustering stage, we first perform k-means clustering in the embedding space to obtain initial centroids $\mathbf{m} = [m_1, m_2, \ldots, m_k] \in R^{k\times d}$, where k and d represent the number and dimension of centroids, respectively. Next, we calculate the assignment probability between a sample embedding z_i and the centers $\mathbf{m}$ of all clusters by a Student's t-distribution:

$$\mathbf{q_i} = \frac{\mathbf{I}\left[\mathbf{E} + \frac{1}{\alpha} diag\left(\left(\mathbf{Z_i} - \mathbf{m}\right)\left(\mathbf{Z_i} - \mathbf{m}\right)^T\right)\right]^{-\frac{\alpha+1}{2}}}{\mathbf{I}\left[\mathbf{E} + \frac{1}{\alpha} diag\left(\left(\mathbf{Z}_i - \mathbf{m}\right)\left(\mathbf{Z_i} - \mathbf{m}\right)^T\right)\right]^{-\frac{\alpha+1}{2}}\mathbf{I}^T}, \tag{5}$$

where $\mathbf{q_i} \in R^{1\times k}$, $\mathbf{Z_i} = \mathbf{I}^T \cdot \mathbf{z_i}$ represents the dimension broadcasting of the embedding representation for the i^{th} sample x_i, and $\mathbf{E}$ is an identity matrix.

Then, the determinacy among sample x_i and all centroids is determined by a mapping function as follow:

$$\mathbf{fq_i} = \frac{\mathbf{Q_i}\cdot diag\left(\mathbf{Q_i}\right)}{t^2}\mathbf{1}\left(\mathbf{Q_{ij}} < 0\right) + \frac{\mathbf{Q_i}\cdot diag\left(\mathbf{Q_i}\right)}{\left(1-t\right)^2}\mathbf{1}\left(\mathbf{Q_{ij}} \geq 0\right), \tag{6}$$

here, t represents the co-association probability at the lowest level of determinacy, which is adaptively determined using Otsu's method, $\mathbf{Q_i} = \mathbf{q_i} - t, \mathbf{Q_i} \in R^{1\times k}$ indicates the offset of $\mathbf{q_i}$ with respect to the threshold t, $diag$ denotes the construction of a diagonal matrix from a vector, and $\mathbf{1}(\cdot)$ repre-

Algorithm 1 Algorithm of DECS

Input: Dataset X, Number of cluster k, Maximum iterations MaxIter;
Output: Cluster center $\mathbf{m}$, Cluster assignment s;

```
Initialize autoencoder's weight by (2) with X;
Initialize m and s with k-means algorithm;
while iter ≤ MaxIter do
    compute the co-association probability matrix of samples x ∈ X and centers m by (5);
    compute the determinacy between samples x ∈ X and centers m by (6);
    compute the stability sq of n samples x ∈ X by (7);
    update encoder's weight and m by optimizing (16)(17);
end while
return Cluster center m, Cluster assignment s by maximizing sq.
```

sents an indicator function that equals one only when a certain condition is satisfied, and zero otherwise.

After obtaining the determinacy relationship between each sample-center pair, the stability of sample x_i can be calculated based on the following formula:

$$\mathbf{sq_i} = \frac{1}{k}\mathbf{fq_i}\cdot\mathbf{I}^T - \frac{\lambda}{k}tr\left[\left(\mathbf{fq_i} - \frac{1}{k}\mathbf{fq_i}\cdot\mathbf{I}^T\mathbf{I}\right)^T\left(\mathbf{fq_i} - \frac{1}{k}\mathbf{fq_i}\cdot\mathbf{I}^T\mathbf{I}\right)\right], \tag{7}$$

here, k represents the number of clusters, λ is a proportionality coefficient, tr denotes the trace of a matrix. That is, the first and second terms of Eq. (7) represent the mean and variance of $\mathbf{fq_i}$, respectively.

Based on the above discussion, we can obtain the stability of each sample. During the process of clustering, we utilize instability as the loss and optimize the network parameters by minimizing this loss. Thus, our clustering loss function can be formulated as:

$$L_c = 1 - \frac{1}{n}\mathbf{I}\cdot\mathbf{sq}, \tag{8}$$

where n denotes the number of samples, $\mathbf{sq} = \{\mathbf{sq_i}\}_{i=1}^{n}$, $\mathbf{sq} \in R^{n\times 1}$ represents the stability of the n samples.

By optimizing this objective function, we can gradually move each sample close to its corresponding cluster and farther away from other clusters. This results in a high level of stability of all samples, close to one. The training steps of the proposed DECS are shown in Algorithm 1.

3.4 Optimization and Convergence Analysis

At each epoch, our model jointly optimizes the cluster centers $\{m_j\}$ and neural network parameters θ using stochastic gradient descent with momentum. Firstly, the gradients of L_c with respect to $\mathbf{sq}$ can be expressed as follows:

$$\frac{\partial L_c}{\partial \mathbf{sq}} = -\frac{1}{n}\mathbf{I}^T. \tag{9}$$

Secondly, the gradient of stability $\mathbf{sq_i}$ of the i^{th} sample with respect to deterministic $\mathbf{fq_i}$ can be written as:

$$\frac{\partial \mathbf{sq_i}}{\partial \mathbf{fq_i}} = \frac{1}{k}\mathbf{I}^T - \frac{2\lambda}{k}\left[\left(\mathbf{fq_i}^T - \frac{1}{k}\mathbf{fq_i}^T\cdot\mathbf{I}\mathbf{I}^T\right)\right]. \tag{10}$$

We proof the correctness of Eq. (10) as follow:

Proof. According to Eq. (7),

$$\begin{aligned}
\mathbf{sq_i} =& \frac{1}{k}\mathbf{fq_i}\cdot\mathbf{I}^T - \frac{\lambda}{k}tr\left[\left(\mathbf{fq_i} - \frac{1}{k}\mathbf{fq_i}\cdot\mathbf{I}^T\mathbf{I}\right)^T\left(\mathbf{fq_i} - \frac{1}{k}\mathbf{fq_i}\cdot\mathbf{I}^T\mathbf{I}\right)\right] \\
=& \frac{1}{k}\mathbf{fq_i}\cdot\mathbf{I}^T - \frac{\lambda}{k}tr\left[\left(\mathbf{fq_i}^T\mathbf{fq_i} - \frac{1}{k}\mathbf{fq_i}^T\mathbf{fq_i}\cdot\mathbf{I}^T\mathbf{I}\right.\right. \\
& \left.\left. -\frac{1}{k}tr(\mathbf{I}^T\mathbf{I}\cdot\mathbf{fq_i}^T\mathbf{fq_i}) + \frac{1}{k^2}tr(\mathbf{I}^T\mathbf{I}\cdot\mathbf{fq_i}^T\mathbf{fq_i}\cdot\mathbf{I}^T\mathbf{I})\right)\right] \\
=& \frac{1}{k}\mathbf{fq_i}\cdot\mathbf{I}^T - \frac{\lambda}{k}\left[\left(tr(\mathbf{fq_i}^T\mathbf{fq_i}) - \frac{1}{k}tr(\mathbf{fq_i}^T\mathbf{fq_i}\cdot\mathbf{I}^T\mathbf{I})\right.\right. \\
& \left.\left. -\frac{1}{k}tr(\mathbf{fq_i}\mathbf{I}^T\mathbf{I}\cdot\mathbf{fq_i}^T) + \frac{1}{k^2}tr(\mathbf{fq_i}\cdot\mathbf{I}^T\mathbf{I}\cdot\mathbf{I}^T\mathbf{I}\cdot\mathbf{fq_i}^T)\right)\right] \\
=& \frac{1}{k}\mathbf{fq_i}\cdot\mathbf{I}^T - \frac{\lambda}{k}\left[\left(tr(\mathbf{fq_i}^T\mathbf{fq_i}) - \frac{1}{k}tr(\mathbf{fq_i}^T\mathbf{fq_i}\cdot\mathbf{I}^T\mathbf{I})\right.\right. \\
& \left.\left. -\frac{1}{k}tr(\mathbf{fq_i}\mathbf{I}^T\mathbf{I}\cdot\mathbf{fq_i}^T) + \frac{1}{k}tr(\mathbf{fq_i}\cdot\mathbf{I}^T\mathbf{I}\cdot\mathbf{fq_i}^T)\right)\right] \\
=& \frac{1}{k}\mathbf{fq_i}\cdot\mathbf{I}^T - \frac{\lambda}{k}\left[\left(tr(\mathbf{fq_i}^T\mathbf{fq_i}) - \frac{1}{k}tr(\mathbf{fq_i}^T\mathbf{fq_i}\cdot\mathbf{I}^T\mathbf{I})\right)\right].
\end{aligned}$$

So, we can easily conclude Eq. (10). □

Thirdly, the gradient of determinacy $fq\left(q_{ij}\right)$ with respect to q_{ij} can be written as follow:

$$\frac{\partial fq\left(q_{ij}\right)}{\partial q_{ij}} = \frac{2\left(q_{ij}-t\right)}{t^2}\mathbf{1}\left(q_{ij}<t\right) + \frac{2\left(q_{ij}-t\right)}{\left(1-t\right)^2}\mathbf{1}\left(q_{ij}\geq t\right), \tag{11}$$

and thus the gradient of determinacy $\mathbf{fq_i}$ with respect to $\mathbf{q_i}$ can be written as

$$\frac{\partial \mathbf{fq_i}}{\partial \mathbf{q_i}} = \left[\frac{\partial fq\left(q_{i1}\right)}{\partial q_{i1}}, \frac{\partial fq\left(q_{i2}\right)}{\partial q_{i2}}, \ldots, \frac{\partial fq\left(q_{ik}\right)}{\partial q_{ik}}\right], \tag{12}$$

where q_{ij} and $fq\left(q_{ij}\right)$ represents the j^{th} element of $\mathbf{q_i}$ and $\mathbf{fq_i}$, respectively.

Lastly, for simplicity, we let $\alpha = 1$ and

$$\begin{aligned}
\mathbf{A} &:= diag\left(\mathbf{E} + \mathbf{Z}\mathbf{Z}^T - 2\mathbf{Z}\mathbf{m}^T + \mathbf{m}\mathbf{m}^T\right); \\
\mathbf{B} &:= -diag\left(2\mathbf{Z} - 2\mathbf{m}\right); \\
\mathbf{C} &:= -diag\left(2\mathbf{m} - 2\mathbf{Z}\right).
\end{aligned} \tag{13}$$

The gradient of $\mathbf{q_i}$ with respect to $\mathbf{z_i}$ and $\mathbf{m}$ can be expressed separately as:

$$\frac{\partial \mathbf{q_i}}{\partial \mathbf{z_i}} = \frac{\mathbf{A}^{-2}\mathbf{B}\mathbf{I}^T\mathbf{I}\mathbf{A}^{-1}\mathbf{I}^T - \mathbf{A}^{-1}\mathbf{I}^T\mathbf{I}\mathbf{A}^{-2}\mathbf{B}\mathbf{I}^T}{\mathbf{I}\mathbf{A}^{-1}\mathbf{I}^T\mathbf{I}\mathbf{A}^{-1}\mathbf{I}^T}, \tag{14}$$

$$\frac{\partial \mathbf{q_i}}{\partial \mathbf{m}} = \frac{\mathbf{A}^{-2}\mathbf{C}\mathbf{I}^T\mathbf{I}\mathbf{A}^{-1}\mathbf{I}^T - \mathbf{A}^{-1}\mathbf{I}^T\mathbf{I}\mathbf{A}^{-2}\mathbf{C}\mathbf{I}^T}{\mathbf{I}\mathbf{A}^{-1}\mathbf{I}^T\mathbf{I}\mathbf{A}^{-1}\mathbf{I}^T}. \tag{15}$$

Proof. According to Eq. (5),

$$\begin{aligned}
\mathbf{q_i} &= \frac{\mathbf{I}\left[\mathbf{E} + diag\left(\left(\mathbf{Z_i}-\mathbf{m}\right)\left(\mathbf{Z_i}-\mathbf{m}\right)^T\right)\right]^{-1}}{\mathbf{I}\left[\mathbf{E} + diag\left(\left(\mathbf{Z}_i-\mathbf{m}\right)\left(\mathbf{Z_i}-\mathbf{m}\right)^T\right)\right]^{-1}\mathbf{I}^T} \\
&= \frac{\mathbf{I}\left[diag\left(\mathbf{E} + \mathbf{Z_i}\mathbf{Z_i}^T - 2\mathbf{Z_i}\mathbf{m}^T + \mathbf{m}\mathbf{m}^T\right)\right]^{-1}}{\mathbf{I}\left[diag\left(\mathbf{E} + \mathbf{Z_i}\mathbf{Z_i}^T - 2\mathbf{Z_i}\mathbf{m}^T + \mathbf{m}\mathbf{m}^T\right)\right]^{-1}\mathbf{I}^T}.
\end{aligned}$$

Based on Eq. (13), we can easily conclude Eq. (14) and Eq. (15). □

Therefore, the gradients of L_c with respect to the latent embedding $\mathbf{z_i}$ and cluster centroid $\mathbf{m}$ are be computed as:

$$\frac{\partial L_c}{\partial \mathbf{z_i}} = \frac{\partial L_c}{\partial \mathbf{sq_i}} \cdot \frac{\partial \mathbf{sq_i}}{\partial \mathbf{fq_i}} \cdot \frac{\partial \mathbf{fq_i}}{\partial \mathbf{q_i}} \cdot \frac{\partial \mathbf{q_i}}{\partial \mathbf{z_i}}, \tag{16}$$

$$\frac{\partial L_c}{\partial \mathbf{m}} = \frac{\partial L_c}{\partial \mathbf{sq_i}} \cdot \frac{\partial \mathbf{sq_i}}{\partial \mathbf{fq_i}} \cdot \frac{\partial \mathbf{fq_i}}{\partial \mathbf{q_i}} \cdot \frac{\partial \mathbf{q_i}}{\partial \mathbf{m}}. \tag{17}$$

Then, the gradients $\frac{\partial L_c}{\partial z_i}$ are propagated to the neural network and used in backpropagation to compute the network's parameter gradient $\frac{\partial L_c}{\partial \theta}$. Through iterating these updates, the model finds the optimal clustering result. The training process is repeated until the convergence condition is met.

To validate the correctness of the optimization process, Fig.2 presents the graphs of functions and their derivatives involved in computing sample stability for the case of two classes.

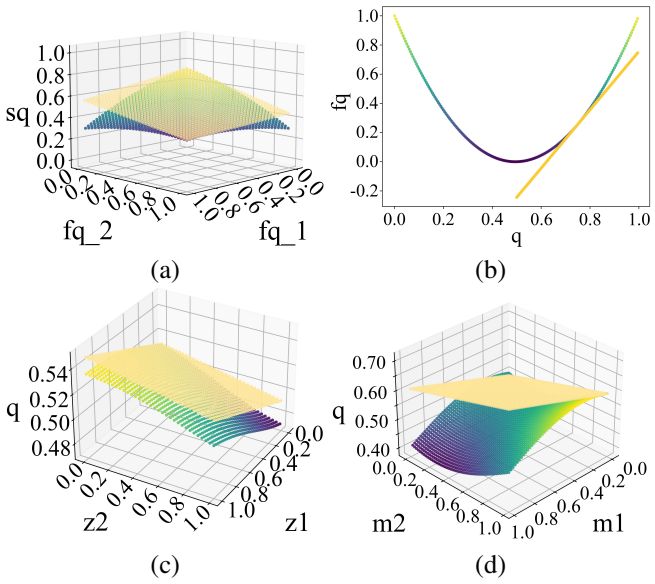


Figure 2: Visualization of the functions and their derivatives involved in sample stability clustering. (a) shows the function of sq w.r.t fq and its derivative in the case of two centroids. (b) represents the function of fq w.r.t q and its derivative. (c) demonstrates the function of q w.r.t two dimensions vector z and its derivative. (d) depicts the function of q w.r.t two cluster centers m and its derivative.

Furthermore, We have theoretically proven the convergence of the loss L_c with respect to the centroids $\mathbf{m}$.

Theorem 1. *There exists $M > 0$ such that $|| \bigtriangledown L_c|| \leq M$, where $M = \frac{2(1+2\lambda)(\alpha+1)}{4nkt^2\alpha} max(||z_i - m_j||)$.*

Proof.

Lemma 1. *[Nesterov, 1998] Let f be Lipschitz continuous on the ball $B_2(x^*, R)$ with the constant M and $||x_0 - x^*|| \leq R$. Then*

$$f_k^* - f^* \leq M \frac{R^2 + \sum_{i=0}^{k} h_i^2}{2\sum_{i=0}^{k} h_i}, \tag{18}$$

where h_i represents the step size and M is termed as the Lipschitz constant.

According to Lemma 1, its convergence can be determined by the initial solution and step size when the loss function satisfies Lipschitz continuity. Consequently, we can conclude based on Lemma 1 that the objective function L_c is convergence $i.f.f.$ $|| \bigtriangledown L_c|| \leq M$. In other words, to verify the convergence of L_c, it is necessary to prove the existence of an upper bound for its derivative.

For the sake of simplicity, a centroid m_j is used as an example for the convergence analysis, and we only consider the case where $q_{ij} < t$, while the case of $q_{ij} \geq t$ can be similarly treated. We can know:

$$\left|\left|\frac{\partial L_c}{\partial m_j}\right|\right| = \left|\left|\frac{\partial L_c}{\partial sq(x_i)} \frac{\partial sq(x_i)}{\partial fq(q_{ij})} \frac{\partial fq(q_{ij})}{\partial q_{ij}} \frac{\partial q_{ij}}{\partial m_j}\right|\right|, \tag{19}$$

where $sq(x_i)$, $fq(q_{ij})$ and q_{ij} represents the stability of i^{th} sample, determinacy and co-association probability of the i^{th} sample with respect to the j^{th} centroid.

For the first term of the Eq.(19), it is clear that $\frac{\partial L_c}{\partial sq(x_i)} = -\frac{1}{n}$, and as for the second item, we know that:

$$\left|\left|\frac{\partial sq(x_i)}{\partial fq(q_{ij})}\right|\right| = \frac{1}{k} - \frac{2\lambda}{k}\left(fq(q_{ij}) - \mu\right) \leq \frac{1+2\lambda}{k}, \tag{20}$$

here, μ represents the mean of $\{fq(q_{ij})\}_{j=1}^{k}$ and $0 \leq fq(q_{ij}) \leq 1$.

For the third term of the Eq.(19), in the case of $q_{ij} < t$:

$$\left|\left|\frac{\partial fq(q_{ij})}{\partial q_{ij}}\right|\right| = \frac{2\left(q_{ij} - t\right)}{t^2} \geq \frac{-2}{t^2}, \tag{21}$$

and for the last term of the Eq.(19), we know that:

$$\begin{aligned}\left|\left|\frac{\partial q_{ij}}{\partial m_j}\right|\right| &= \left|\left|\frac{\alpha+1}{\alpha} \frac{z_i - m_j}{1 + ||z_i - m_j||^2/\alpha} q_{ij}\left(1 - q_{ij}\right)\right|\right| \\ &\leq \frac{\alpha+1}{4\alpha}\left|\left|\left(z_i - m_j\right)\right|\right|,\end{aligned} \tag{22}$$

where, z_i represents the i^{th} sample embedding, $1 + ||z_i - m_j||^2/\alpha \geq 1$, and $q_{ij}\left(1 - q_{ij}\right) \leq \frac{1}{4}$ due to $0 \leq q_{ij} \leq 1$.

According to the above analysis, we can conclude that the upper bound for the loss L_c:

$$\left|\left|\frac{\partial L_c}{\partial m_j}\right|\right| \leq \frac{2\left(1+2\lambda\right)\left(\alpha+1\right)}{4nkt^2\alpha}\left|\left|z_i - m_j\right|\right|. \tag{23}$$

That is to say, there exists $M > 0$ such that $|| \bigtriangledown L_c|| \leq M$, where $M = \frac{2(1+2\lambda)(\alpha+1)}{4nkt^2\alpha} max(||z_i - m_j||)$. In fact, there exists an upper boundary of $||z_i - m_j||$ for any real-world dataset. □

4 Experiments

In this section, we evaluate the effectiveness of the proposed DECS method on five benchmark datasets. We also present the visualization of sample distribution and analyze how these hyperparameters impact the performance.

methods	MNIST		MNIST-test		USPS		Fashion		YTF	
	ACC	*NMI*	*ACC*	*NMI*	*ACC*	*NMI*	*ACC*	*NMI*	*ACC*	*NMI*
k-means	0.532	0.499	0.542	0.500	0.668	0.626	0.474	0.512	0.601	0.776
GMM	0.433	0.366	0.540	0.593	0.551	0.530	0.556	0.557	0.348	0.411
SC-Ncut	0.656	0.731	0.660	0.704	0.649	0.794	0.508	0.575	0.510	0.701
SC-LS	0.714	0.706	0.740	0.756	0.746	0.755	0.496	0.497	0.544	0.759
AC-GDL	0.113	0.017	0.933	0.864	0.725	0.825	0.112	0.010	0.430	0.662
DEC	0.863	0.834	0.856	0.830	0.762	0.767	0.518	0.546	0.371	0.446
IDEC	0.881*	0.867*	0.846	0.802	0.761*	0.785*	0.529	0.557	0.400*	0.483*
DCEC	0.890	0.885	0.852	0.809	0.790	0.862	-	-	-	-
VaDE	0.944	0.891	0.944*	0.885*	0.566*	0.512*	0.629	0.611	0.601*	0.753*
JULE	0.964	0.913	0.961	0.915	0.950	0.913	0.563*	0.608*	0.684	0.848
DEPICT	0.965*	0.917*	0.963*	0.915*	0.899	0.906	0.392	0.392	0.621	0.802
IDCEC	0.948	0.906	0.923	0.853	0.812	0.858	-	-	0.632	0.793
TELL	0.952	0.888	0.776*	0.751*	0.865*	0.786*	0.584*	0.658*	-	-
AdaGAE	0.929	0.853	-	-	0.920	0.848	-	-	-	-
MI-ADM	0.969	0.922	0.871	0.885	0.979	0.948	-	-	0.606	0.801
DSCDA	0.978	0.941	0.980	0.946	0.869	0.857	**0.662**	0.645	0.691	0.857
DynAE	0.987	0.964	**0.987**	**0.963**	0.981	0.948	0.591	0.642	-	-
ASPC-DA	**0.988**	**0.966**	0.973	0.936	**0.982**	**0.951**	0.591	0.654	-	-
DeepDPM	0.980	0.950	-	-	0.940	0.900	0.610	0.500	0.821	**0.930**
TDEC	0.985	0.957	0.975	0.935	0.976	0.935	**0.645**	**0.693**	**0.950**	**0.980**
DECS	**0.990**	**0.973**	**0.990**	**0.971**	**0.992**	**0.976**	0.642	**0.716**	**0.827**	0.911

Table 1: Comparison of clustering performance on five datasets in terms of ACC and NMI. The bolded font represents the best and second results.

4.1 Datasets and Evaluation Metrics

In order to validate the performance and generality of the proposed method, we perform experiments on five image datasets, as shown in Table 2. Considering that clustering tasks are fully unsupervised, the training and test split are merged in all our experiments.

Dataset	samples	Classes	Dimensions
MNIST-full	70,000	10	1x28x28
MNIST-test	10,000	10	1x28x28
USPS	9,298	10	1x16x16
Fashion-Mnist	70,000	10	1x28x28
YTF	12,183	41	3x55x55

Table 2: Description of Datasets

Two widely-used unsupervised evaluation metrics including cluster accuracy(ACC) and normalized mutual information(NMI) are used to validate the performance of the proposed model. Higher values of these metrics indicate better clustering performance.

4.2 Baseline Methods

In the comparative experiments, our proposed method was compared with several representative conventional baseline, including: k-means [MacQueen, 1967a], GMM [Reynolds, 2009], sc-Ncut [Shi and Malik, 2000], SC-LS [Chen and Cai, 2011] and AC-GDL [Zhang *et al.*, 2013]. In addition, our method was compared with several state-of-the-art deep clustering algorithms. To ensure the fairness of our experiments, we have selected comparative methods that utilize autoencoders as feature extractors, including: DEC [Xie *et al.*, 2016], IDEC [Guo *et al.*, 2017a], DCEC [Guo *et al.*, 2017b], VaDE [Jiang *et al.*, 2016], JULE [Yang *et al.*, 2016], DEPICT [Ghasedi Dizaji *et al.*, 2017], IDCEC [Lu *et al.*, 2022], TELL [Peng *et al.*, 2022], AdaGAE [Li *et al.*, 2021], MI-ADM [Jabi *et al.*, 2019], DSCDA [Yang *et al.*, 2019], DynAE [Mrabah *et al.*, 2020], ASPC-DA [Guo *et al.*, 2019], TDEC [Zhang *et al.*, 2023] and DeepDPM [Ronen *et al.*, 2022].

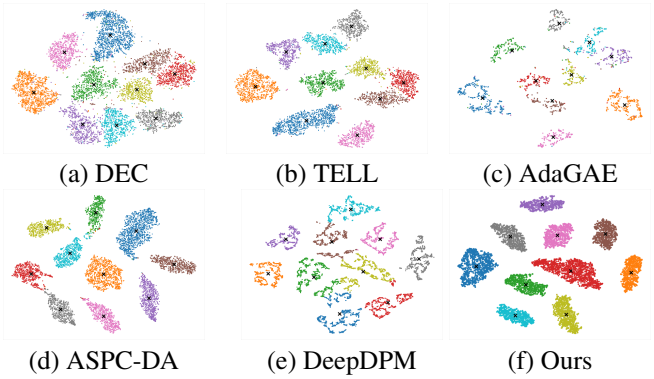
4.3 Experiment Results

Table 1 presents the performance of our method and other comparative methods. For the compared methods, if their results on some datasets were not reported, we ran the released code with hyperparameters mentioned in their papers, and the results are marked by (*) on top. When the code is not publicly available or running the released code is not practical, we replaced the corresponding results with dashes (-).

It be seen from Table 1 that the proposed DECS algorithm achieves superior clustering results across all datasets. Specifically, on the USPS dataset, our algorithm achieves a clustering accuracy of over 99%. It outperforms the second-best ASPC-DA by 1.0% and 2.5% on ACC and NMI, respectively. Furthermore, our method significantly outperforms several classical shallow baselines, which can be attributed to the utilization of a multi-layer convolutional autoencoder as the feature extractor.

Furthermore, we also performed t-SNE visualization to compare the cluster structures obtained using different clustering algorithms on the USPS dataset, as shown in Fig.3. Specifically, Fig.3 (a)-(e) represent the clustering results obtained by algorithms DEC, TELL, AdaGAE, ASPC-DA, and DeepDPM, respectively, while Fig.3 (f) represents the clustering result of our proposed algorithm. It is evident that our proposed algorithm is able to achieve clearer and more accu-

rate cluster structures, which further proves the effectiveness of the proposed algorithm.



(a) DEC (b) TELL (c) AdaGAE

(d) ASPC-DA (e) DeepDPM (f) Ours

Figure 3: T-SNE visualization comparing the cluster structures obtained from different clustering algorithms on the USPS datasets. Distinct colors represent different digits, and the cluster centers are indicated by black 'x' symbols.

In addition, we have investigated the sensitivity of our model to the parameters α and λ using the USPS, MNIST and Fashion datasets, and the results are shown in Fig.4. Specifically, Fig.4 (a), (c) and (e) display the results of ACC from different parameter settings on USPS, MNIST and Fashion datasets, and Fig.4 (b), (d) and (f) show the results of NMI. According to the figure, we can observe that the variation of hyperparameters has little effect on the clustering performance, which indicates that our model is not sensitive to the initialization of hyperparameters.

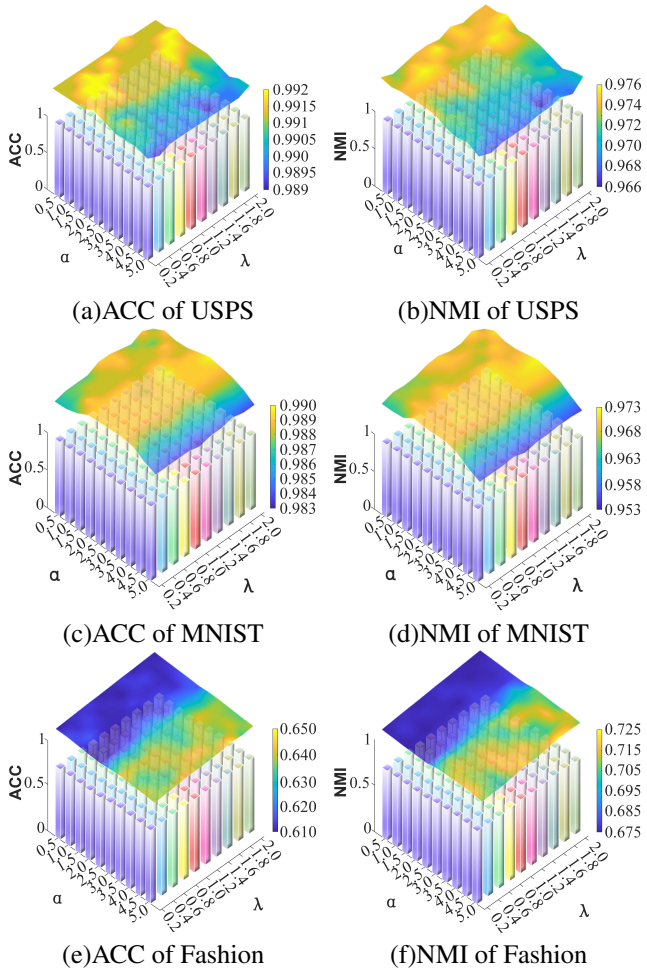
4.4 Implementation

For all datasets, we specify that the encoder to consists of four convolutional layers with channel sizes of 32, 64, 128, and 256, respectively. Each convolutional kernel has a size of 3×3 and uses a stride of 2. Furthermore, batch normalization and max pooling layers are added after each convolutional layer. The decoder uses a network that mirrors the encoder's structure. Additionally, ReLU is utilized as the activation function for all convolutional layers in the model.

During the training process, data augmentation techniques such as random rotation, translation, and cropping are applied to improve the neural network's generalization ability. In addition, the autoencoder is trained end-to-end for 500 epochs using the Adam optimizer with default parameters in Keras. Then, the encoder is further trained for 10000 iterations with a batch size of 256. The coefficient λ for variance is set to 0.8 during the calculation of sample stability. Our source code is publicly available at: https://github.com/ChengZhanwen/DECS.

5 Conclusion

In this paper, we proposed a deep embedding clustering algorithm driven by sample stability. The algorithm combines a convolutional autoencoder model with a clustering layer that relies on sample stability. Unlike previous methods, our method constrains the sample using sample stability, eliminating the need for artificially constructed pseudo targets. This mitigated the clustering biases caused by inappropriate pseudo targets and significantly improved the reliability of the clustering results. We analyzed the convergence of the proposed DECS model, and the experimental results on five image datasets indicate that our algorithm achieve superior clustering performance. In the future, incorporating more complex representation learning models and applying our approach to a wider range of real-world datasets may be an intriguing and practical avenue for research.



(a)ACC of USPS (b)NMI of USPS

(c)ACC of MNIST (d)NMI of MNIST

(e)ACC of Fashion (f)NMI of Fashion

Figure 4: ACC and NMI of our method with different α and λ on USPS, MNIST and Fashion datasets, respectively.

Acknowledgments

This work was supported by the National Science and Technology Major Project under Grant (No.2021ZD0112400), the National Natural Science Foundation of China (Nos. 62136005, 62106132, 62306170, 62306171), the Science and Technology Major Project of Shanxi (No. 202201020101006), the Special Fund for Science and Technology Innovation Teams of Shanxi Province (No. 202304051001001).

Contribution Statement

All authors conceived this paper. Zhanwen Cheng and Feijiang Li contributed equally to this work.

References

[Bishop and Nasrabadi, 2006] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[Chen and Cai, 2011] Xinlei Chen and Deng Cai. Large scale spectral clustering with landmark-based representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, pages 313–318, 2011.

[Ghasedi Dizaji *et al.*, 2017] Kamran Ghasedi Dizaji, Amirhossein Herandi, Cheng Deng, Weidong Cai, and Heng Huang. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *Proceedings of the IEEE international conference on computer vision*, pages 5736–5745, 2017.

[Guo *et al.*, 2017a] Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. Improved deep embedded clustering with local structure preservation. In *Ijcai*, volume 17, pages 1753–1759, 2017.

[Guo *et al.*, 2017b] Xifeng Guo, Xinwang Liu, En Zhu, and Jianping Yin. Deep clustering with convolutional autoencoders. In *Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part II 24*, pages 373–382. Springer, 2017.

[Guo *et al.*, 2018] Xifeng Guo, En Zhu, Xinwang Liu, and Jianping Yin. Deep embedded clustering with data augmentation. In *Asian conference on machine learning*, pages 550–565. PMLR, 2018.

[Guo *et al.*, 2019] Xifeng Guo, Xinwang Liu, En Zhu, Xinzhong Zhu, Miaomiao Li, Xin Xu, and Jianping Yin. Adaptive self-paced deep clustering with data augmentation. *IEEE Transactions on Knowledge and Data Engineering*, 32(9):1680–1693, 2019.

[Jabi *et al.*, 2019] Mohammed Jabi, Marco Pedersoli, Amar Mitiche, and Ismail Ben Ayed. Deep clustering: On the link between discriminative models and k-means. *IEEE transactions on pattern analysis and machine intelligence*, 43(6):1887–1896, 2019.

[Jaiswal *et al.*, 2020] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020.

[Jiang *et al.*, 2016] Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. Variational deep embedding: An unsupervised and generative approach to clustering. *arXiv preprint arXiv:1611.05148*, 2016.

[Jing and Tian, 2020] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):4037–4058, 2020.

[Johnson, 1967] Stephen C Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

[Koga *et al.*, 2007] Hisashi Koga, Tetsuo Ishibashi, and Toshinori Watanabe. Fast agglomerative hierarchical clustering algorithm using locality-sensitive hashing. *Knowledge and Information Systems*, 12:25–53, 2007.

[Li *et al.*, 2018] Fengfu Li, Hong Qiao, and Bo Zhang. Discriminatively boosted image clustering with fully convolutional auto-encoders. *Pattern Recognition*, 83:161–173, 2018.

[Li *et al.*, 2019] Feijiang Li, Yuhua Qian, Jieting Wang, Chuangyin Dang, and Liping Jing. Clustering ensemble based on sample's stability. *Artificial Intelligence*, 273:37–55, 2019.

[Li *et al.*, 2020] F Li, Y Qian, J Wang, J Liang, and W Wang. Clustering method based on samples stability. *Sci. Sin. Inf.*, 50(8):1239–1254, 2020.

[Li *et al.*, 2021] Xuelong Li, Hongyuan Zhang, and Rui Zhang. Adaptive graph auto-encoder for general data clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):9725–9732, 2021.

[Lu *et al.*, 2022] Hu Lu, Chao Chen, Hui Wei, Zhongchen Ma, Ke Jiang, and Yingquan Wang. Improved deep convolutional embedded clustering with re-selectable sample training. *Pattern Recognition*, 127:108611, 2022.

[MacQueen, 1967a] J MacQueen. Classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[MacQueen, 1967b] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[Mrabah *et al.*, 2020] Nairouz Mrabah, Naimul Mefraz Khan, Riadh Ksantini, and Zied Lachiri. Deep clustering with a dynamic autoencoder: From reconstruction towards centroids construction. *Neural Networks*, 130:206–228, 2020.

[Nesterov, 1998] Yurii Nesterov. Introductory lectures on convex programming volume i: Basic course. *Lecture notes*, 3(4):5, 1998.

[Ng *et al.*, 2001] Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14, 2001.

[Niu *et al.*, 2020] Chuang Niu, Jun Zhang, Ge Wang, and Jimin Liang. Gatcluster: Self-supervised gaussian-attention network for image clustering. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*, pages 735–751. Springer, 2020.

[Niu *et al.*, 2022] Chuang Niu, Hongming Shan, and Ge Wang. Spice: Semantic pseudo-labeling for image

clustering. *IEEE Transactions on Image Processing*, 31:7264–7278, 2022.

[Peng *et al.*, 2022] Xi Peng, Yunfan Li, Ivor W Tsang, Hongyuan Zhu, Jiancheng Lv, and Joey Tianyi Zhou. Xai beyond classification: Interpretable neural clustering. *The Journal of Machine Learning Research*, 23(1):227–254, 2022.

[Ren *et al.*, 2019] Yazhou Ren, Kangrong Hu, Xinyi Dai, Lili Pan, Steven CH Hoi, and Zenglin Xu. Semi-supervised deep embedded clustering. *Neurocomputing*, 325:121–130, 2019.

[Reynolds, 2009] Douglas Reynolds. *Gaussian Mixture Models*, pages 659–663. Springer US, Boston, MA, 2009.

[Rezaei *et al.*, 2021] Mina Rezaei, Emilio Dorigatti, David Ruegamer, and Bernd Bischl. Learning statistical representation with joint deep embedded clustering. *arXiv preprint arXiv:2109.05232*, 1, 2021.

[Ronen *et al.*, 2022] Meitar Ronen, Shahaf E Finder, and Oren Freifeld. Deepdpm: Deep clustering with an unknown number of clusters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9861–9870, 2022.

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

[Sneath and Sokal, 1962] Peter HA Sneath and Robert R Sokal. Numerical taxonomy. *Nature*, 193:855–860, 1962.

[Xie *et al.*, 2016] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR, 2016.

[Xu and Wunsch, 2005] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678, 2005.

[Yang *et al.*, 2016] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5147–5156, 2016.

[Yang *et al.*, 2018] Xu Yang, Cheng Deng, Xianglong Liu, and Feiping Nie. New l2, 1-norm relaxation of multi-way graph cut for clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[Yang *et al.*, 2019] Xu Yang, Cheng Deng, Feng Zheng, Junchi Yan, and Wei Liu. Deep spectral clustering using dual autoencoder network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4066–4075, 2019.

[Zhang *et al.*, 2013] Wei Zhang, Deli Zhao, and Xiaogang Wang. Agglomerative clustering via maximum incremental path integral. *Pattern Recognition*, 46(11):3056–3065, 2013.

[Zhang *et al.*, 2023] Ruilin Zhang, Haiyang Zheng, and Hongpeng Wang. Tdec: Deep embedded image clustering with transformer and distribution information. In *Proceedings of the 2023 ACM International Conference on Multimedia Retrieval*, pages 280–288, 2023.