Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 3742536, 11 pages
https://doi.org/10.1155/2021/3742536

Research Article

Representation Learning Based on Autoencoder and Deep Adaptive Clustering for Image Clustering

Siquan Yu,[1,2,3] **Jiaxin Liu**,[4] **Zhi Han**,[2,3] **Yong Li**,[5] **Yandong Tang**,[2,3] **and Chengdong Wu**[6]

[1]*School of Information Science and Engineering, Northeastern University, Shenyang, Liaoning 110819, China*
[2]*State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, Liaoning 110016, China*
[3]*Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang, Liaoning 110016, China*
[4]*State Grid Liaoning Electric Power Research Institute, Shenyang 110006, China*
[5]*State Grid Shandong Electric Power Company, Jining, Shandong 250001, China*
[6]*Faculty of Robot Science and Engineering, Northeastern University, Shenyang, Liaoning, 110819, China*

Correspondence should be addressed to Zhi Han; hanzhi@sia.cn

Received 13 July 2020; Revised 29 December 2020; Accepted 31 December 2020; Published 25 February 2021

Academic Editor: Yumin Cheng

Image clustering is a complex procedure, which is significantly affected by the choice of image representation. Most of the existing image clustering methods treat representation learning and clustering separately, which usually bring two problems. On the one hand, image representations are difficult to select and the learned representations are not suitable for clustering. On the other hand, they inevitably involve some clustering step, which may bring some error and hurt the clustering results. To tackle these problems, we present a new clustering method that efficiently builds an image representation and precisely discovers cluster assignments. For this purpose, the image clustering task is regarded as a binary pairwise classification problem with local structure preservation. Specifically, we propose here such an approach for image clustering based on a fully convolutional autoencoder and deep adaptive clustering (DAC). To extract the essential representation and maintain the local structure, a fully convolutional autoencoder is applied. To manipulate feature to clustering space and obtain a suitable image representation, the DAC algorithm participates in the training of autoencoder. Our method can learn an image representation that is suitable for clustering and discover the precise clustering label for each image. A series of real-world image clustering experiments verify the effectiveness of the proposed algorithm.

1. Introduction

Clustering is a basic unsupervised learning problem whose purpose is to divide data into several subgroups. Generally speaking, the elements in the same subgroup are similar and different from the elements of the other subgroups [1]. Image clustering is one of the fundamental high-dimension data clustering tasks in computer vision and machine learning [2]. Despite decades of development, the reliable clustering method of image data is still an outstanding problem [3, 4].

From the perspective of image representation, there are two types of image clustering methods, which are the traditional image clustering methods and deep clustering methods [5]. The traditional image clustering methods group data on handcrafted features and treat feature extraction and clustering separately [6]. Based on this insight, many attempts have been dedicated to developing suitable clustering feature extracting techniques such as manually designed feature descriptors, including Bag of feature (BOW) [7], Histogram of Oriented Gradient (HOG) [8], Principal Component Analysis (PCA) [9], and Scale-Invariant Feature Transform (SIFT) [10]. However, the representation ability is limited by using handcrafted features that do not depend on the distribution of input data. How to establish an effective feature representation is a crucial problem that needs to be solved in image clustering.

In recent years, a deep neural network has been successfully applied in various supervised learning tasks [11, 12]. The reasons for the success of deep neural networks are to learn more essential representation of images by constructing a network with multiple hidden layers and train the network with a large number of data [13]. Motivated by the success of deep neural networks in supervised learning, some unsupervised deep learning methods have been used to image clustering. These methods are called deep clustering [14]. Most previous deep clustering studies are two-stage training schemes based on an autoencoder. First, they usually train an autoencoder to reduce the dimension of image data. Then, the encoder acts as a feature extractor and uses a clustering algorithm to train it simultaneously. The two-stage clustering methods have been widely studied and successfully applied in many works [15–19]. The reason for the effectiveness of autoencoder based methods is that it can preserve some properties of data by adding prior knowledge to subjective. Thus, the encoder constructs a feature representation that can comprehensively describe the image information. However, since no clustering-driven objection participates during the training two-stage clustering methods, the learned encoder may not be suitable for clustering.

Latter, one-stage clustering methods that jointly accomplish feature transformation and clustering come into being. Deep adaptive image clustering (DAC) is a typical one-stage image clustering algorithm [20]. It defines an effective objective and proposes a self-learning scheme to realize image clustering. The defined objective function is used to update the parameters of a convolutional network by selecting highly confident image pairs and the cluster assignment is integrated into classification labels. However, there are two crucial factors that affect the stability and effectiveness of the DAC algorithm. On the one hand, the initialization of the convolutional network is an important factor affecting the performance of DAC. On the other hand, with the training of DAC, the local structure preservation of representation cannot be guaranteed. Thus, the image representation in the distorted feature space will hurt the clustering performance.

To overcome the problems of DAC, we present an image clustering representation learning method based on autoencoder (AE) [21] and deep adaptive image clustering (DAC) [20]. Specifically, to obtain the essential features of the image and provide initial parameters for DAC, we incorporate a fully convolutional autoencoder into DAC algorithm. As a clustering algorithm, DAC helps to train the autoencoder to get clustering friendly features. Autoencoder can guarantee the feature space not to be distorted. The proposed method can learn the image representation suitable for clustering and simultaneously find the clustering labels of each image. Extensive experiments verify the effectiveness of the proposed algorithm.

The main contributions of this paper can be concluded in three aspects:

(1) We propose a novel system based on an AE and DEC and use it to learn an informative image representation

(2) Since AE and DEC can complement each other, we use the learned image representation to realize image clustering

(3) We conduct extensive experiments on four real-world datasets to verify the effectiveness of the proposed algorithm

The rest of the paper is organized as follows. The Section 2 will briefly introduce the related work of our paper. Section 3 proposes the clustering algorithm as well as some details of the algorithm. Section 4 provides a series of experiments to verify the effectiveness of the proposed algorithm. The last section briefly concludes our paper.

2. Related Work

2.1. Deep Clustering. Deep clustering refers to clustering with the related algorithm of deep neural networks [22]. Existing deep clustering algorithms are mainly to seek some effective ways to combine deep feature learning with traditional clustering methods, which are mainly divided into two categories: (I) a two-stage work that apply clustering after a representation is learned; (II) a one-stage work that jointly optimize the representation learning and clustering [23].

Two-stage methods usually train an autoencoder at the first stage. Then, the encoder acts as a feature extractor and uses a clustering algorithm to obtain the clustering results. Autoencoder (AE) is a classical feature learning method that is based on deep neural networks and image reconstruction loss function [18]. Recently, many image algorithms attempt to regularize the learning of image representation of autoencoder with the loss function of the traditional clustering algorithm. For instance, Deep Embedding Clustering (DEC) utilizes KL-divergence as a loss function to measure the distance between the distribution of image feature and the target distribution [24]. Ghasedi Dizaji et al. propose Stacked Auto-Encoder (SAE) to learn a deep learning-based latent feature representation and use it to improve classification [18]. Peng et al. propose a novel clustering method by minimizing the discrepancy between pairwise sample assignments for each data point [25]. Gaussian Mixture Variational Autoencoders (GMVAE) is a representative generation-based clustering algorithm that incorporates Gaussian distribution to variational autoencoder [16]. The advantage of AE is that it keeps the essential information of features in the process of clustering algorithm training, and the learned representation is more suitable for clustering tasks. Thus, it can avoid the degradation of the solution and improve clustering performance. The disadvantage of the two-stage method is the mismatch problem between image representation and clustering. Specifically, the clustering algorithm does not participate in representation learning, which will lead to the blindness of representation learning.

One-stage methods combine image representation with clustering learning. For instance, deep adaptive image clustering (DAC) is a typical one-stage image clustering algorithm [20]. It defines an effective objective and proposes an adaptive mechanism to realize image clustering. Guo et al. propose Improved Deep Embedded Clustering (IDEC) algorithm to take care of data local structure preservation [26]. IDEC trains AE and self-training simultaneously to realize local feature preservation. (CatGAN) uses general Generative Network Adversarial (GAN) and entropy as loss function to realize data clustering [27]. JULE proposes a recurrent framework for joint unsupervised learning of deep representations and image clusters [17]. The effectiveness of these learning schemes has been proved in theory and practical experiments. However, there are two crucial factors that affect the stability and effectiveness of these algorithms. On the one hand, the initialization of the convolutional network is an important factor. On the other hand, with the training going on, the local structure preservation of representation cannot be guaranteed.

3. Image Clustering Based on AE and DEC

3.1. Autoencoder. AE is a type of artificial neural network which are used to learn efficient data codings in an unsupervised manner [28]. Generally speaking, the objection of an AE is to extract a feature (encoding) of the input data. In the field of computer vision, it is usually used to learn image representations and reduce image dimension [16, 29].

Consider $X: = \{(x_i)\}_{i=1}^n$ being a set of images, where n denotes the number of images. An AE reduces the dimension of images from high-dimensional spaces $\mathbb{R}^D$ to a low dimensional space $\mathbb{T}^d$ and $d < D$. The embedding of the dataset $\mathbb{T}^d$ is denoted by $Y: = \{(y_i)\}_{i=1}^n$. The function that performs the embedding denotes as f_θ. Thus, $y_i = f_\theta(x_i), i = 1, 2, ..., n$. Generally, to guarantee the learned representation Y can adequately represent the input image information. The following reconstruction loss is used to train the autoencoder network:

$$\mathscr{L}_a(\theta) = \frac{1}{2n}\sum_{i=1}^{n}\left\|x_i - g_w(f_\theta(x_i))\right\|_2^2, \tag{1}$$

where $g_w(\cdot)$ is the decoder that maps the representation to the output.

In our algorithm, to extract essential features and preserve spatial locality of images, we adopt a fully convolutional autoencoder to realize the image feature extraction stream.

3.2. Deep Adaptive Image Clustering. DAC is a clustering algorithm that is realized by a convolutional neural network (CNN) and an adaptive training mechanism [20]. It employs some constraints on the classification output and generates a feature for image clustering.

Let us assume that x_i and y_j are two unlabeled images, r_{ij} denotes an unknown binary output received by the generated label, where $r_{ij} = 1$ if x_i and y_j belong to the same cluster and otherwise, $r_{ij} = 0$. In this case, $f'_\theta(x_i) \cdot f'_\theta(x_j) = z_i \cdot z_j$ is the dot product of z_i and z_j, and it indicates the similarity, where $f'_\theta(\cdot)$ is a classification network. Based on the similarity of the input image features, the binary labels are defined as follows:

$$r_{ij} = \begin{cases} 1, & z_i \cdot z_j \geq u(\xi), \\ 0, & z_i \cdot z_j \leq l(\xi), \\ \text{None}, & \text{otherwise}, \end{cases} \tag{2}$$

where ξ is an adaptive parameter, $u(\cdot)$ and $l(\cdot)$ are two learnable thresholds.

For network training, the objection function of DAC is defined as follows:

$$\min_{\theta,\xi}\sum_{i,j} v_{ij}\mathscr{L}_r\left(r_{ij}, s\left(x_i, x_j, \theta\right)\right) + u(\xi) - l(\xi), \tag{3}$$

where $s(\cdot)$ denote the estimated similarity of x_i and x_j with a classification network parameter θ, v_{ij} an indicator coefficient matrix to predict the training samples, which $v_{ij} = 1$ means that the sample is selected to train the network, and $v_{ij} = 0$ otherwise. v_{ij} is defined as follows:

$$v_{ij} = \begin{cases} 1, \text{ if } r_{ij} = 1, \\ 0, \text{ otherwise}, \end{cases} \quad i, j = 1, 2, ..., n. \tag{4}$$

$\mathscr{L}_r(r_{ij}, s(x_i, x_j, \theta))$ is the loss function and is defined as follows:

$$\mathscr{L}_r\left(r_{ij}, s\left(x_i, x_j, \theta\right)\right) = -r_{ij}\log\left(s\left(x_i, x_j, \theta\right)\right)\left(1 - r_{ij}\right)\log\left(1 - s\left(x_i, x_j, \theta\right)\right). \tag{5}$$

3.3. Network Architecture. The network architecture of the proposed clustering algorithm is shown in Figure 1. There are two streams in our network, the autoencoder stream, and the DAC stream. The autoencoder stream is realized by several fully convolutional layers and the DAC stream is composed of the autoencoder's encoder and several fully connected layers. Considering a dataset $D: = \{(x_i)\}_{i=1}^n$ with n input samples that need to be clustered. The number of clustering K is a priori knowledge. $Z: = \{(z_i)\}_{i=1}^n$ be the output of the encoder. Thus, the encoder can be defined as a nonlinear mapping $f_{\theta_1}: x_i \longrightarrow z_i$ and the decoder is $g_w: z_i \longrightarrow \tilde{x}_i$ where θ_1 and w are the parameters of encoder and decoder, respectively. $\tilde{x}_i$ denotes the output of the decoder and the output of DAC can be represented as $f_{\theta_1,\theta_2}(x_i)$, which is a classification network and θ_2 is the parameters of fully connected layers.

3.4. Loss Function. Since we aim to seek an encoder that makes the extracted feature more suitable for clustering. The reconstruction loss of the autoencoder is added to the initialization and training process of the DAC algorithm. On the one hand, the reconstructive loss function is used to assist the learning of image representation as it can learn the essential feature of input images and avoid the distortion of feature space in the training process of DAC. On the other hand, the loss of an autoencoder is only focused on image

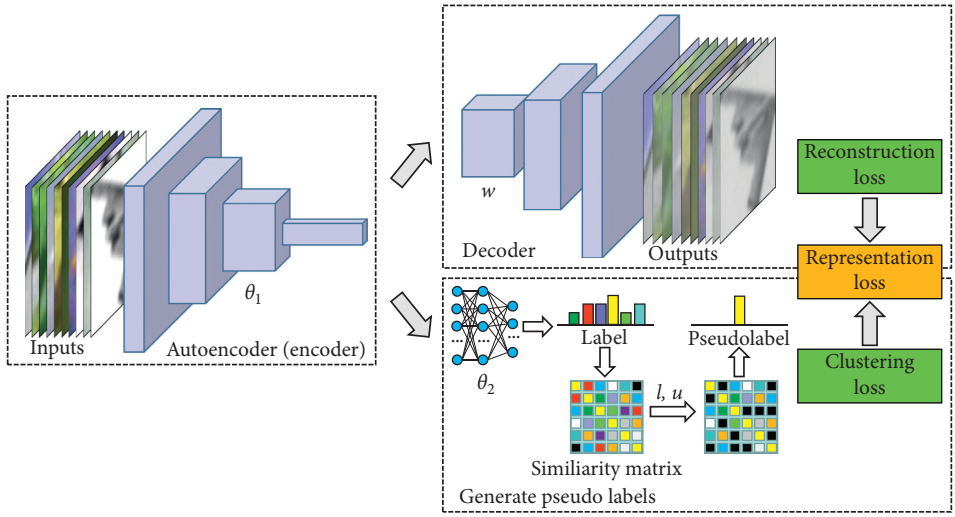

Figure 1: The network architecture of the proposed clustering algorithm.

reconstruction, which loses the useful information needed for clustering. DAC loss can guide it to obtain a better representation suitable for clustering. Thus, we define the complete loss function as follows:

$$\mathscr{L} = \alpha \mathscr{L}_r + \beta \mathscr{L}_c, \tag{6}$$

where α is a balance coefficient. $\mathscr{L}_r$ and $\mathscr{L}_c$ are reconstruction loss and clustering loss, respectively. The final objective function is as follows:

$$\min_{\theta_1,\theta_2,w,\xi} \alpha \mathscr{L}_r + \beta \mathscr{L}_c + u(\xi) + l(\xi), \tag{7}$$

where the definitions of $\mathscr{L}_r$ and $\mathscr{L}_c$ are as follows:

$$\mathscr{L}_r(\theta_1) = \frac{1}{2n} \sum_{i=1}^{n} \left\| x_i - g_w\left(f_{\theta_1}(x_i)\right) \right\|_2^2,$$

$$\mathscr{L}_c\left(r_{ij}, s\left(x_i, x_j, \theta_1, \theta_2\right)\right)$$

$$= -r_{ij} \log\left(s\left(x_i, x_j, \theta_1, \theta_2\right)\right)\left(1 - r_{ij}\right)\log\left(1 - s\left(x_i, x_j, \theta_1, \theta_2\right)\right). \tag{8}$$

3.5. Network Training. In this section, we present the whole training process of our algorithm. To minimize the loss function proposed in (7), we first abandon the DAC stream and pretrain a convolutional autoencoder by using loss L_r. The trained encoder can provide initial parameters for the DAC algorithm. Thus, the DAC algorithm will select more accurate labeled image pairs in the initial stage. Then, we simultaneously train the autoencoder stream and the DAC stream by minimizing (7). The detailed algorithm is formalized as Algorithm 1.

When the autoencoder is trained, it seems to get a clustering friendly representation by finetuning the autoencoder's encoder in the DAC algorithm. The encoder is connected with a number of fully connected layers to form a classification network, and the labels are calculated by the algorithm proposed in [20]. However, we suppose that this kind of finetuning can distort the representation space, which may weaken the expressive power and thereby hurt clustering results. For this reason, in the process of training the DAC stream, the autoencoder also needs to be trained to maintain the DAC algorithm to obtain highly confidence labeled image pairs.

4. Experiments

In this section, we carry out a series of experiments to verify the effectiveness of our algorithm. All the experiments are carried out in Tensorflow and Keras environment running Ubuntu14.04, Inter(R) Core i7-4790 CPU 3.6 GHz and Titan X GPU 12 GB.

4.1. Datasets. In this part, fore challenging image datasets including Fashion-MNIST, Cifar-10, Cifar-100, and STL-10 datasets are selected to verify the effectiveness of our algorithm. We first briefly introduce datasets.

4.1.1. Fashion-MNIST. Fashion-MNIST is a dataset of Zalando's article images, which includes a training set of 60,000 examples and a test set of 10,000 examples [30]. In the Fashion-MNIST dataset, each example is a 28×28 grayscale image, associated with a label from 10 classes.

4.1.2. Cifar-10 and Cifar-100. CIFAR-10 contains 50,000 training images and 10,000 test images from 10 classes [31]. Each image has a size of 32×32. Cifar-100 is similar to Cifar-10, except it has 10 times fewer images per class.

4.1.3. STL-10. The STL-10 dataset is an image dataset used to develop unsupervised feature learning, deep learning, and self-supervised learning algorithms [32]. It is inspired by the CIFAR-10 dataset but with some modifications. The high-resolution dataset (96×96) will make it a challenging benchmark to develop more scalable unsupervised learning methods.

Input: input images D: $= \{(x_i)\}_{i=1}^{n}$; number of clusters K; the threshold $u(\xi) = 0.95 - \xi$ and $l(\xi) = 0.5 + 0.1 \cdot \xi$; the batch size $b_s = 128$ and the learning rate $\eta = 0.005$ and $\rho = 0.0001$; the balance coefficient $\alpha = 0.5$, $\beta = 0.3$.
Output: reconstruction images $\tilde{D}$: $= \{(\tilde{x}_i)\}_{i=1}^{n}$; clustering labels of input images.
(1) Pertain a fully convolutional autoencoder;
(2) for t in *epochs* do
(3) Extract the images feature $f_{\theta_1}(x_i)$ in the batch X_{bs};
(4) Compute the similarity r_{ij} and pseudolabel based on (2);
(5) Calculate the indicator coefficient v_{ij} based on (4);
(6) Updata θ_1, θ_2 and w by minimizing (7);
(7) end for
(8) Updata ξ by minimizing (7).
(9) for x_i in X do
(10) I_i: $= f_{\theta_1,\theta_2}(x_i)$ and c_i: $= \arg\max_h (I_{ih})$;
(11) end for

Algorithm 1

In our experiments, the training set and validation set of each dataset are jointly utilized. In particular, the 20 superclasses of Cifar-100 dataset are considered in all the experiments. We summarize the detailed information of each dataset in Table 1.

4.2. Evaluation Metrics. Three commonly clustering metrics including accuracy (ACC), normalized mutual information (NMI), and adjusted rand index (ARI) are adopted to evaluate the performance of clustering algorithms. These metrics reflect the cluster performance from different perspectives. ACC measures the best matching between the clustering labels and ground truth labels. NMI measures the similarity between pairs of clusters [33]. ARI establishes a baseline by using the expected similarity of all pairwise comparisons between clusters specified by a random model [34].

4.3. Competitors. We compare the performance to traditional clustering methods and deep clustering methods. Specifically, traditional clustering methods include K-means [35], Self-tuning Spectral Clustering (SSC) [36]. Deep clustering methods include Greedy Layer-Wise Training of Deep Networks (GLWTDN) [37], Consistent Inference of Latent Representations (CILR) [15], Gaussian Mixture Variational Autoencoders (GMVAE) [16], Categorical Generative Adversarial Networks (CatGAN) [27], Joint Unsupervised Learning (JULE) [17], Deep embedding clustering (DEC) [24], and DAC [20]. We summarize the clustering results of the mentioned methods on all datasets in Table 2. Next, we will briefly introduce the competitors.

4.3.1. Traditional Image Clustering Methods. K-means++ and SSC: these methods first use Bag of Wording (BOW) to encode the images, and then, the image features are clustered to achieve image clustering.

4.3.2. Deep Clustering Methods. GLWTDN: it first trains an AE model to extract image features and then uses the K-means algorithm to cluster the image features [37].

CILR: it first adopts consistent inference of latent representations (CILR) to generate latent labeled data points of the inputs. Then, CILR is derived to pretrain DNNs by minimizing the distance between latent labeled data points to realize image clustering [15].

GMVAE: it uses the Gaussian mixture model as a prior distribution to improve the traditional variational autoencoder. It uses the improved latent vector as image representation and then clusters representation to realize image clustering [16].

CatGAN: it uses general Generative Network Adversarial (GAN) and entropy as loss function to realize data clustering [27].

JULE: it proposes a recurrent framework for joint unsupervised learning of deep representations and image clustering [17].

DEC: it first learns image representations from an AE. Then, clusters are obtained by utilizing a typical K-means algorithm [24].

DAC: it formulates image clustering as a binary pairwise classification problem and identifies these pairs of images which should belong to the same cluster [20].

4.4. Experimental Settings. Following the setting in DAC, we set the initial thresholds which construct highly confident pseudolabel to $u(\xi) = 0.95 - \xi$ and $l(\xi) = 0.5 + 0.1\xi$, respectively. The initialization of the parameter ξ is set to 0. Considering the convergence of our algorithm, we set the learning rate of ξ to $\eta = 0.005$. The balance coefficient is 0.5. For training, we adopt the well-known Adam optimizer with an initial learning rate 0.0001. In addition, the batch size is set to $b_s = 128$ in all the experiments. The detailed network architecture used in each dataset is shown in Table 3.

4.5. Results

4.5.1. Clustering Performance Comparison. In this part, we first compare our method with many state-of-the-art methods including K-means [35], Self-Tuning Spectral

Table 1: Information of the datasets.

Data sets	Samples	Clusters	Dimensions
Fashion-MNIST	70000	10	28×28
Cifar-10	60000	10	$32 \times 32 \times 3$
Cifar-100	60000	20	$32 \times 32 \times 3$
STL-10	13000	10	$96 \times 96 \times 3$

Table 2: Clustering performance and comparison, ACC (%) and NMI (%) and ARI (%) on all datasets.

Data sets	Fashion-MNIST			Cifar-10			Cifar-100			STL-10		
Metric	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC
K-means [35]	18.55	14.37	22.89	8.71	4.87	22.89	8.39	2.80	12.97	12.45	6.08	19.20
SSC [36]	13.35	14.23	20.81	10.28	8.53	24.67	9.01	2.18	13.60	9.78	4.79	15.88
GLWTDN [37]	20.31	18.72	25.74	23.93	16.89	31.35	10.04	4.76	16.45	24.96	16.10	30.30
CILR [15]	25.68	20.71	30.10	11.05	6.12	25.83	9.43	4.53	15.54	13.98	6.52	24.52
GMVAE [16]	25.68	16.07	30.10	28.47	18.37	33.64	13.66	4.92	15.84	23.65	15.72	31.77
CatGAN [27]	26.31	17.11	32.24	26.46	17.57	31.52	12.0	4.53	15.10	21.00	13.90	29.84
JULE [17]	43.79	33.99	47.78	19.23	13.77	27.15	10.26	3.27	13.67	18.15	16.43	27.69
DEC [24]	60.54	58.43	65.25	25.68	16.07	30.10	13.58	4.95	18.52	27.60	18.61	35.90
DAC [20]	<u>65.38</u>	<u>62.41</u>	<u>70.52</u>	<u>43.79</u>	<u>33.99</u>	<u>47.78</u>	<u>21.24</u>	<u>11.00</u>	<u>25.52</u>	<u>40.33</u>	<u>28.64</u>	<u>48.18</u>
Ours	**70.66**	**69.43**	**75.21**	**46.39**	**42.25**	**52.25**	**25.46**	**16.10**	**30.47**	**44.31**	**33.08**	**52.12**

Most of the results are excerpted from [20]. The best and second-best results are marked in bold and underlined, respectively.

Clustering (SPC) [36], Greedy Layer-Wise Training of Deep Networks (GLWTDN) [37], Consistent Inference of Latent Representations (CILR) [15], Gaussian Mixture Variational Autoencoders (GMVAE) [16], Categorical Generative Adversarial Networks (CatGAN) [27], JULE-SF [17], JULE-RC [17], Deep embedding clustering (DEC) [24], and DAC [20]. We summarize the clustering results of the mentioned methods on all datasets in Table 2.

As shown in Table 2, for each dataset, the performance of deep clustering methods is superior to that of traditional clustering algorithms. Autoencoder based method such as AE outperforms traditional algorithm K-means with a large margin, which justifies the fascinating potential of autoencoder in clustering task. Furthermore, note that the proposed method outperforms the other algorithms on all datasets. In addition, the clustering accuracy of our algorithm outperforms all competitive baselines, with significant margins of 4.79%, 4.47%, 4.95%, and 3.94% in the case of Fashion-MNIST, Cifar-10, Cifar-100, and STL-10, respectively. These results verify the effectiveness of our method in image clustering tasks.

Figure 2 shows the confusion matrixes of the clustering results for Cifar-10 and STL-10 datasets. The values along the diagonal represent the percentage of samples correctly classified into the corresponding categories. We can find that all clustering accuracy is average and stable for these two datasets. This proves that our method does not aggregate samples into a few categories and can effectively avoid the degenerate solution problem.

4.5.2. Visualization. In this part, we use two methods to visualize the clustering results of our algorithm. The first visualization experiment is conducted on the Fashion-MNIST dataset. We randomly sampled 10000 samples of the representation $z_i$and mapped them to a 2-dimension vector by using t-SNE [30]. The experiment results are shown in Figure 3. In Figures 3(a)–3(f), different colors indicate different clusters and the corresponding clustering accuracies are reported as follows. The experimental results show that the proposed algorithm can effectively improve the separability of data, which is helpful to improve the clustering accuracy of the image.

In the second visualization experiment, we qualitatively analyze the cluster results by the proposed Cifar-10 dataset. For each category, we randomly select an image as the original image at the first stage. Then, we pick up several samples, which are the smallest Euclidean distance between original images from the same cluster. Finally, we pick the samples which are closest to the original image in the incorrect cluster. All the picked images are shown in Figure 4, and we mark the incorrect samples with red boxes. Form the visualization results, we can find that the successful cases not only depend on texture information but also contain some semantic information of categories. The failure cases also contain a lot of texture information similar to the source images. It implies that our method not only captures image appearance information but also captures some abstract image information for image clustering. This is the reason why the proposed method can precisely discover cluster assignments.

4.5.3. On the Effect of Number of Clusters. In this part, we study the effect of the number of clusters on our algorithm and compare the results with the DAC algorithm. For each dataset, we conduct 6 experiments on different training sets. For Cifar-10 and STL-10 dataset, the number of training sets varies in the range of [5, 10] at equal intervals. For Cifar-100 dataset, the number of training samples varies in the range of

Table 3: Network architecture in experiments.

Fashion-MNIST			
Encoder		3 × 3 conv., stride 2, 8 BN Relu 3 × 3 conv., stride 2, 16 BN Relu 3 × 3 conv., stride 2, 32 BN Relu 3 × 3 conv., stride 2, 64 BN Relu	
Decoder	3 × 3 deconv., stride 2, 32 BN Relu 3 × 3 deconv., stride 2, 16 BN Relu 3 × 3 deconv., stride 2, 8 BN Relu 3 × 3 deconv., stride 2, 1	Clustering	(256,64) fc., BN Relu (64,10) fc., BN Relu Softmax
Cifar-10 and Cifar-100 dataset			
Encoder		3 × 3 conv., stride 2, 8 BN Relu 3 × 3 conv., stride 2, 16 BN Relu 3 × 3 conv., stride 2, 32 BN Relu 3 × 3 conv., stride 2, 64 BN Relu	
Decoder	3 × 3 deconv., stride 2, 32 BN Relu 3 × 3 deconv., stride 2, 16 BN Relu 3 × 3 deconv., stride 2, 8 BN Relu 3 × 3 deconv., stride 2, 3	Clustering	(256,64) fc., BN Relu (64,k) fc., BN Relu Softmax
STL-10 dataset			
Encoder		5 × 5 conv., stride 4, 16 BN Relu 3 × 3 conv., stride 2, 32 BN Relu 3 × 3 conv., stride 2, 64 BN Relu 3 × 3 conv., stride 2, 128 BN Relu 3 × 3 conv., stride 1, 256 BN Relu	
Decoder	3 × 3 deconv., stride 2, 128 BN Relu 3 × 3 deconv., stride 2, 64 BN Relu 3 × 3 deconv., stride 2, 32 BN Relu 3 × 3 deconv., stride 2, 16 BN Relu 3 × 3 deconv., stride 2, 3	Clustering	FC (256,64) fc., BN Relu (64,10) fc., BN Relu Softmax

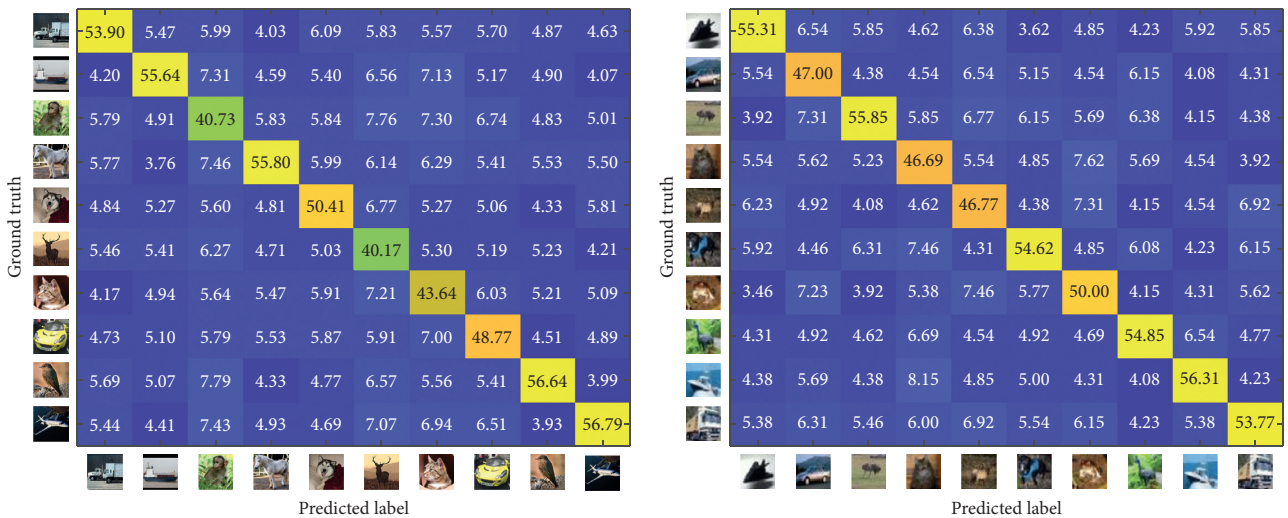

Figure 2: Confusion matrixes of Cifar-10 and STL-10 datasets. (a) Cifar-10. (b) STL-10.

[10, 20] with an interval of 2. We report the variation curves of clustering accuracy with the number of clusters in Figure 5. The detailed numerical results are shown in Table 4.

As shown in Figure 5 and Table 4, with the increase in the number of clusters, the accuracy of clustering decreases gradually. For all datasets, the clustering accuracy of our method is always higher than that of the DAC algorithm in the different number of clusters. In addition, the other two metrics results also show the superiority of the proposed algorithm. This is because autoencoder can reduce the

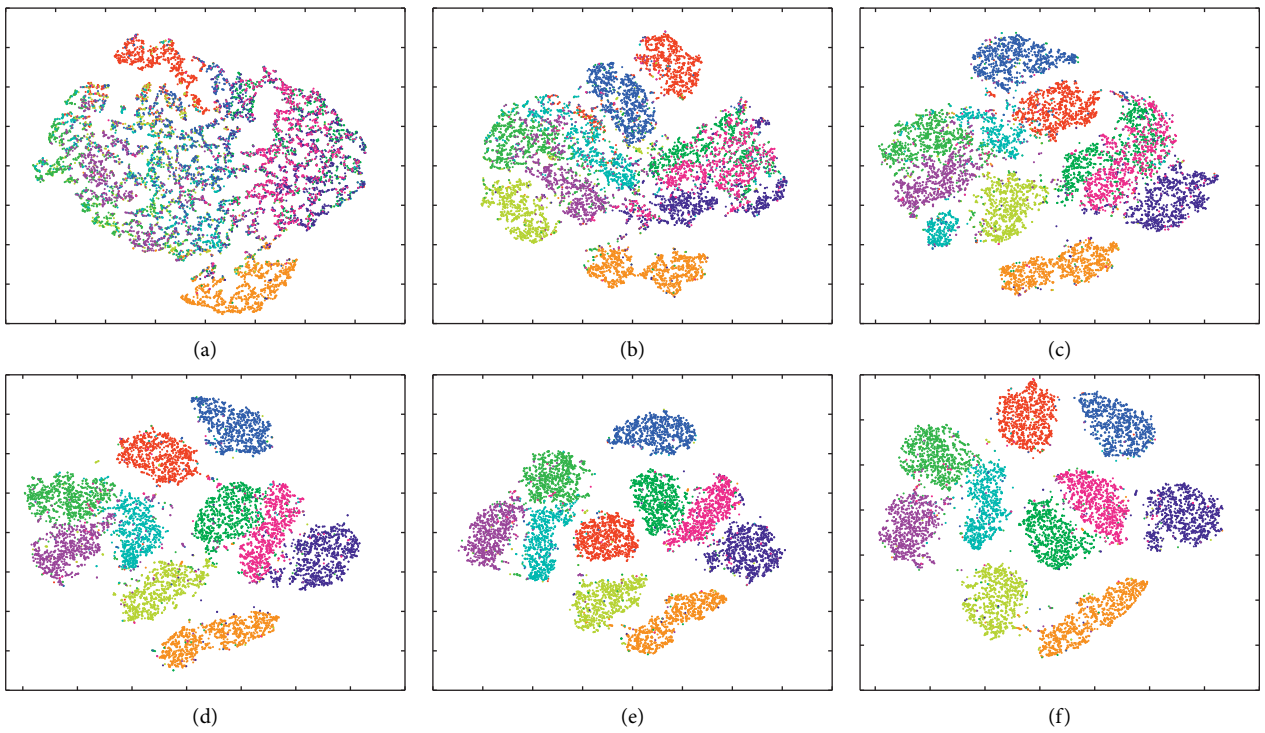

Figure 3: t-SNE visualization of image representation for Fashion-MNIST dataset. (a) 5.21%. (b) 33.43%. (c) 55.41%. (d) 60.35%. (e) 74.56%. (f) 75.21%.

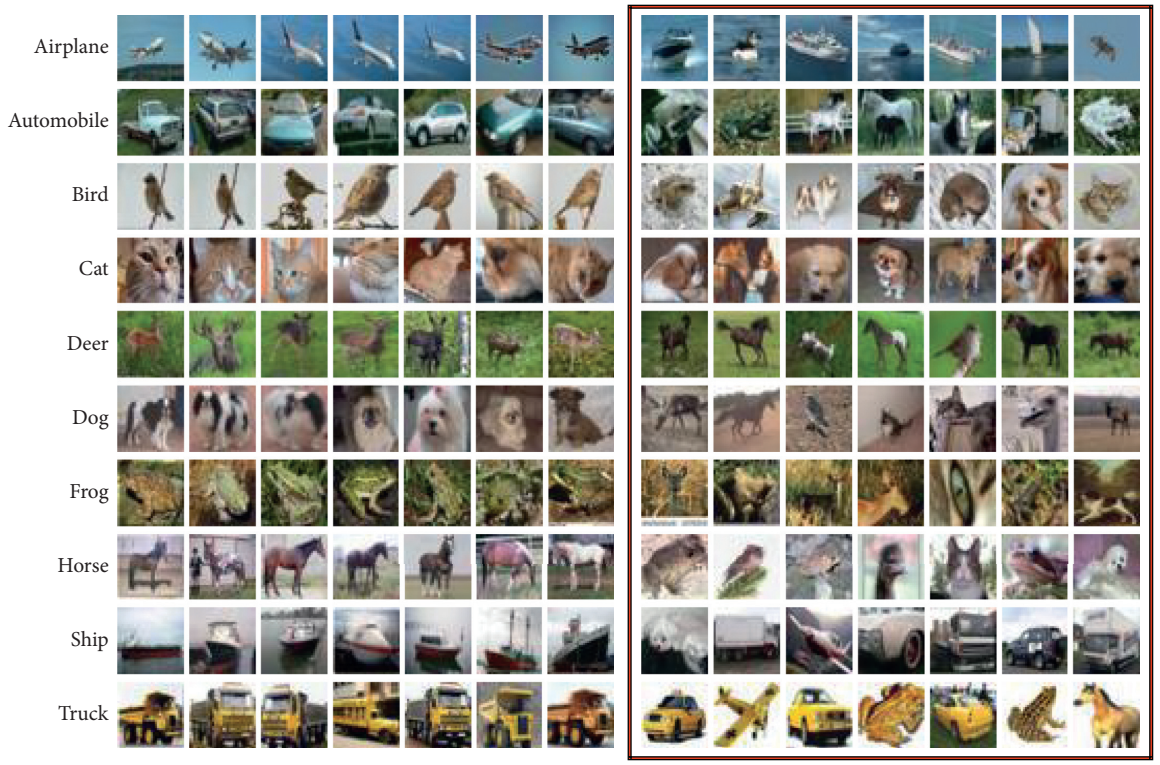

Figure 4: Qualitative analysis of clustering results.

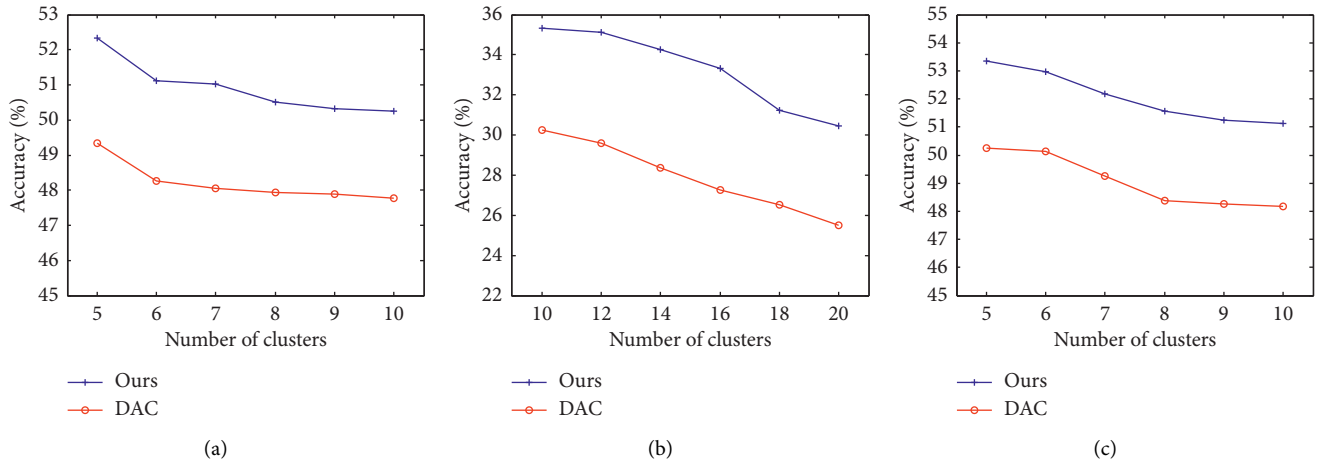

Figure 5: Accuracy with a different number of clusters. (a) Cifar-10. (b) Cifar-100. (c) STL-10.

Table 4: Clustering performance, ACC (%) and NMI (%) and ARI (%), with different number of clusters.

Method	DAC			Ours		
Clusters	NMI	ARI	ACC	NMI	ARI	ACC
Cifar-10						
5	44.89	35.58	49.35	48.12	42.53	52.33
6	44.35	35.03	48.27	47.82	42.02	51.12
7	44.21	34.69	48.05	47.39	41.79	51.02
8	44.02	34.52	47.93	46.64	41.52	50.51
9	43.92	34.21	47.88	46.52	41.27	50.32
10	43.79	33.99	47.78	46.39	40.25	50.25
Cifar-100						
10	22.16	11.89	30.25	26.67	21.72	35.33
12	22.05	11.68	29.58	26.03	21.47	35.10
14	21.97	11.56	28.37	25.81	21.32	34.27
16	21.88	11.35	27.25	25.72	21.04	33.43
18	21.65	11.21	26.54	25.67	20.53	31.25
20	21.24	11.00	25.52	25.46	20.10	30.47
STL-10						
5	41.43	29.68	50.30	45.01	34.82	53.35
6	41.22	29.35	50.12	44.96	34.51	52.98
7	41.05	29.06	49.26	44.72	34.03	52.17
8	40.76	28.96	48.37	44.56	33.87	51.55
9	40.55	28.72	48.27	44.33	33.53	51.23
10	40.33	28.64	48.18	44.31	33.08	51.12

randomness of searching label pair and improve clustering performance of DAC algorithm. The experimental results also show the stability of the algorithm.

4.5.4. On the Effect of the Parameter α and β. In this experiment, we mainly study the effect of the parameter α and β. The range of parameters (α, β) is selected by the grid search of the region $[0.1, 0.9] \times [0.1, 0.9]$ in a step size of 0.1. In Figure 6, we report the clustering accuracies with different (α, β). From the results, we can find that when α and β tend to be close, the clustering accuracy is the highest. This is mainly because the autoencoder can guarantee the local structure of image representation and prevent the distortion of feature space. It also means that autoencoder can promote the clustering performance of DAC, which explains the reason why our algorithm is effective.

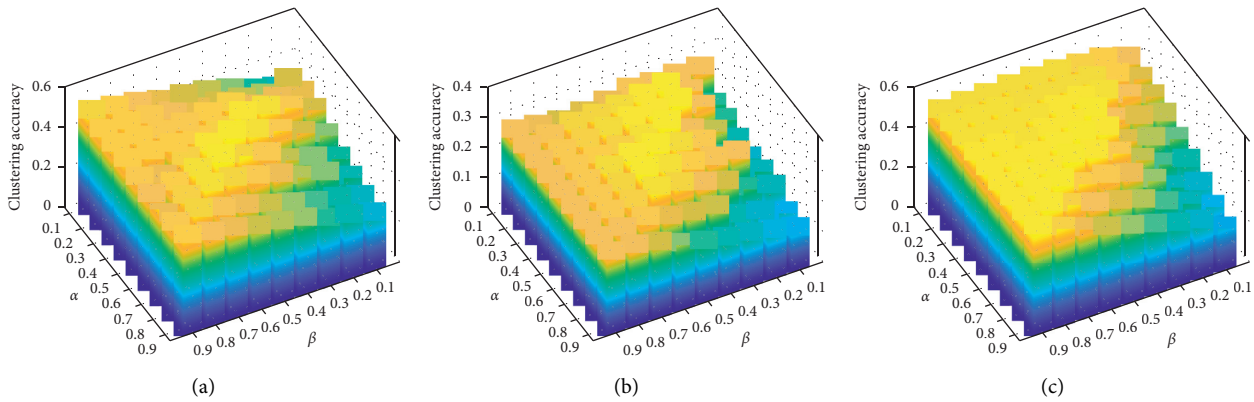

Figure 6: The clustering accuracies with different (α, β) pairs. (a) Cifar-10. (b) Cifar-100. (c) STL-10.

5. Conclusion

In this paper, we present a novel representation learning method and use it to solve the image clustering problem. To generate more informative representations for clustering, we borrow the DAC algorithm and incorporate it to train a fully convolutional autoencoder. The proposed algorithm was evaluated on unsupervised clustering tasks using popular datasets, achieving competitive results compared to the current state of the art. Furthermore, we may improve the proposed algorithm by applying some deep feature extraction models, e.g., Variational AutoEncoder (VAE) and Generative Adversarial Networks (GANs). To improve our method, it is an interesting direction to learn the distribution of image data instead of reconstructing the image. We will see this for future work.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the National Key Research and Development Program of China (no. 2018YFB1307400), the Science and Technology Project of the State Grid Corporation of China (no. SGSDDK00KJJS2000090). The authors appreciate the following research: https://ieeexplore.ieee.org/document/9184041.

References

[1] O. R. Battaglia, B. Di Paola, and C. Fazio, "Cluster analysis of educational data: an example of quantitative study on the answers to an open-ended questionnaire," http://arxiv.org/abs/1512.08998.

[2] S. A. Shah and V. Koltun, "Deep continuous clustering," http://arxiv.org/abs/1803.01449.

[3] Y. Yang, D. Xu, F. Nie, S. Yan, and Y. Zhuang, "Image clustering using local discriminant models and global integration," *IEEE Transactions on Image Processing*, vol. 19, no. 10, pp. 2761–2773, 2010.

[4] J. Lim, J. Ho, M.-H. Yang, K.-c. Lee, and D. Kriegman, "Image clustering with metric, local linear structure, and affine symmetry," in *European Conference on Computer Vision*, pp. 456–468, Springer, Berlin, Germany, 2004.

[5] M. Maheshwari, S. Silakari, and M. Motwani, "Image clustering using color and texture," in *Proceedings of the 2009 First International Conference on Computational Intelligence, Communication Systems and Networks*, pp. 403–408, Indore, India, July 2009.

[6] P. Antonopoulos, N. Nikolaidis, and I. Pitas, "Hierarchical face clustering using sift image features," in *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Image and Signal Processing*, pp. 325–329, Honolulu, HI, USA, April 2007.

[7] K. Firuzi, M. Vakilian, V. P. Darabad, B. T. Phung, and T. R. Blackburn, "A novel method for differentiating and clustering multiple partial discharge sources using s transform and bag of words feature," *IEEE Transactions on Dielectrics and Electrical Insulation*, vol. 24, no. 6, pp. 3694–3702, 2017.

[8] N. Ahmed and A. Jalil, "Multimode image clustering using optimal image descriptor," *IEICE Transactions on Information and Systems*, vol. E97.D, no. 4, pp. 743–751, 2014.

[9] D. Feldman, M. Schmidt, and C. Sohler, "Turning big data into tiny data: constant-size coresets for k-Means, PCA, and projective clustering," *SIAM Journal on Computing*, vol. 49, no. 3, pp. 601–657, 2020.

[10] Y. Zhang and H. Zhang, "Image clustering based on sift-affinity propagation," in *Proceedings of the 2014 11th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pp. 358–362, Xiamen, China, August 2014.

[11] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital Signal Processing*, vol. 73, pp. 1–15, 2018.

[12] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[13] S. Suthaharan, "Machine learning models and algorithms for big data classification," *Integrated Series in Information Systems*, vol. 36, pp. 1–12, 2016.

[14] E. Aljalbout, V. Golkov, Y. Siddiqui, M. Strobel, and D. Cremers, "Clustering with deep learning: taxonomy and new methods," http://arxiv.org/abs/1801.07648.

[15] J. Chang, L. Wang, G. Meng, S. Xiang, and C. Pan, "Deep unsupervised learning with consistent inference of latent representations," *Pattern Recognition*, vol. 77, pp. 438–453, 2018.

[16] N. Dilokthanakul, P. A. Mediano, M. Garnelo et al., "Deep unsupervised clustering with gaussian mixture variational autoencoders," http://arxiv.org/abs/1611.02648.

[17] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5147–5156, Las Vegas, NV, USA, June 2016.

[18] K. Ghasedi Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang, "Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5736–5745, Venice, Italy, October 2017.

[19] Y. Chen, L. Zhang, and Z. Yi, "Subspace clustering using a low-rank constrained autoencoder," *Information Sciences*, vol. 424, pp. 27–38, 2018.

[20] J. Chang, L. Wang, G. Meng, S. Xiang, and C. Pan, "Deep adaptive image clustering," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5879–5887, Venice, Italy, October 2017.

[21] R. Sang, P. Jin, and S. Wan, "Discriminative feature learning for action recognition using a stacked denoising autoencoder," in *Intelligent Data analysis and its Applications*, vol. I, pp. 521–531, Springer, Berlin, Germany, 2014.

[22] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long, "A survey of clustering with deep learning: from the perspective of network architecture," *IEEE Access*, vol. 6, pp. 39501–39514, 2018.

[23] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders," in *Proceedings of the International Conference on Neural Information Processing*, pp. 373–382, Springer, Guangzhou, China, November 2017.

[24] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proceedings of the International Conference on Machine Learning*, pp. 478–487, New York, NY, USA, June 2016.

[25] X. Peng, H. Zhu, J. Feng, C. Shen, and J. T. Zhou, "Deep clustering with sample assignment invariance prior," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 11, 2020.

[26] X. Guo, L. Gao, X. Liu, and J. Yin, "Improved deep embedded clustering with local structure preservation," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pp. 1753–1759, Melbourne, Australia, August 2017.

[27] J. T. Springenberg, "Unsupervised and semi-supervised learning with categorical generative adversarial networks," http://arxiv.org/abs/1511.06390.

[28] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE Journal*, vol. 37, no. 2, pp. 233–243, 1991.

[29] K. G. Lore, A. Akintayo, and S. Sarkar, "Llnet: a deep autoencoder approach to natural low-light image enhancement," *Pattern Recognition*, vol. 61, pp. 650–662, 2017.

[30] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[31] T. Ho-Phuoc, "Cifar10 to compare visual recognition performance between deep neural networks and humans," http://arxiv.org/abs/1811.07270.

[32] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 215–223, Fort Lauderdale, FL, USA, April 2011.

[33] W. Xu, X. Liu, and Y. Gong, "Document clustering based on non-negative matrix factorization," in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 267–273, Toronto Canada, July 2003.

[34] S. Zhang and H.-S. Wong, "Arimp: a generalized adjusted rand index for cluster ensembles," in *Proceedings of the 2010 20th International Conference on Pattern Recognition*, pp. 778–781, IEEE, Istanbul, Turkey, August 2010.

[35] J. Wang, J. Wang, J. Song, X.-S. Xu, H. T. Shen, and S. Li, "Optimized cartesian k-means," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 1, pp. 180–192, 2014.

[36] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," in *Advances in Neural Information Processing Systems*, pp. 1601–1608, MIT Press, Cambridge, MA, USA, 2005.

[37] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in Neural Information Processing Systems*, pp. 153–160, MIT Press, Cambridge, MA, USA, 2007.