Neurocomputing 449 (2021) 38–47

Contents lists available at ScienceDirect

Neurocomputing

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

Adversarial self-supervised clustering with cluster-specificity distribution

Wei Xia [a], Xiangdong Zhang [a], Quanxue Gao [a,*], Xinbo Gao [b,c]

[a] State Key Laboratory of Integrated Services Networks, Xidian University, Shaanxi 710071, China
[b] School of Electronic Engineering, Xidian University, Shaanxi 710071, China
[c] Chongqing Key Laboratory of Image Cognition, Chongqing University of Posts and Telecommunications, Chongqing 400065, China

ARTICLE INFO

Article history:
Received 13 June 2020
Revised 18 February 2021
Accepted 29 March 2021
Available online 2 April 2021
Communicated by Zidong Wang

Keywords:
Deep clustering
Cluster-specificity distribution
Self-supervision
Adversarial learning

ABSTRACT

As we embark on the big data era, numerous unlabeled data has generated increasingly, and thus deep clustering analysis has become prevalent in artificial intelligence. Yet, most existing deep clustering methods still have the following demerits: 1) They fail to take cluster-specificity distribution into account, resulting in suboptimal latent representation. 2) They suffer from the scale issue of the distributions between the given sample and cluster centers, resulting in unstable clustering performance. 3) They fail to utilize the obtained clustering labels, resulting in suboptimal clustering performances. To fill these gaps, we propose a new deep clustering solution, namely Adversarial Self-supervised Clustering With Cluster-specificity Distribution (ASC2D). Specifically, by imposing the cluster-specificity constraint, which is measured by the $\ell_{1,2}$-norm, the learned latent representation well encodes the cluster structure. Meanwhile, by introducing the thought of adversarial learning, ASC2D well eliminates the gaps between distributions. Moreover, ASC2D utilize the clustering label to supervise the learning of representation, where the latter is used in turn to conduct the subsequent clustering. By this way, clustering and representation learning are seamlessly connected, with the aim to achieve better clustering performance. Extensive experimental results show that ASC2D is superior to 14 state-of-the-art baselines on six image datasets in terms of three evaluation metrics, especially on Fashion-MNIST datasets, ASC2D brings about 4.1% and 7.1% improvement over the best baseline in terms of ACC and NMI metrics.

1. Introduction

Deep learning based clustering is one of the active topic in the field of unsupervised learning due to its outstanding representative capacity and fast inference speed. One of the most representative deep clustering methods is deep embedded clustering (DEC) [1]. It learns a nonlinear mapping from a high-dimensional data space to a lower-dimensional feature space in which it iteratively solves a clustering objective function. Inspired by DEC, several deep clustering methods have been proposed for image clustering [2–4] and obtained promising preliminary results.

Recently, $\ell_{1,2}$-norm regularization has been proven to be an effective tool to characterize class-specificity distribution in dimension space [5]. Applying it to the classification tasks of supervised learning has been achieved impressive results, but to the best of our knowledge, similar investigations for deep unsupervised learning have been found lacking so far, which is one of the motivations behind this work. Meanwhile, existing methods only utilize Kullback–Leibler (KL) divergence to measure the mismatch of data distribution and target distribution, which neglects the scale issue[1] between the two distributions, resulting in unstable clustering performance. Moreover, existing methods do not make full use of the information embedded in the learned clustering labels, resulting in suboptimal latent representations.

Motivated by above insight analysis, we propose a novel deep clustering method, namely Adversarial Self-supervised Clustering With Cluster-specificity Distribution (ASC2D). More specifically, to well exploit the cluster structure, we learn latent representation with the $\ell_{1,2}$-norm constraint such that the latent representations with the same cluster to further have a common distribution in dimension space while representations with different clusters have different distribution in the intrinsic dimension space. To deal with the scale issue[1] in different distributions, we introduce the adver-

* Corresponding author.
E-mail address: qxgao@xidian.edu.cn (Q. Gao).

[1] **Scale issue:** Although the error value between the two distributions is small, the magnitude of the element values of the two distributions may differ significantly in terms of significant features.

https://doi.org/10.1016/j.neucom.2021.03.108
0925-2312/

sarial thought of Generative Adversarial Networks (GANs) [6] to eliminate the gaps between data distribution and target distribution. Finally, we make good use of the clustering labels to supervise the learning of latent representation, and the latter is employed in turn to conduct the subsequent clustering. By this way, the representation learning and clustering are seamlessly connected, such an incorporation enables the overall framework to be trained towards achieving better clustering results. The main contributions of this paper are as follows:

- We introduce the $\ell_{1,2}$-norm constraint to learn latent representation. Thus, the learned representation well exploits the cluster structure.
- We introduce an adversarial learning to complement distribution constraint, which makes the learned data distribution and representation more effective, thereby achieving superior clustering performance.
- We make full use of the clustering labels to guide the optimization of latent representation, which seamlessly encapsulates representations learning and clustering in a unified framework.

1.1. Notations

For convenience, we first introduce the notations used throughout the paper. We use bold upper case letters for matrices, *e.g.,* $\mathbf{Z}$, bold lower case letters for vectors,*e.g.,* $\mathbf{z}$, and upper case letters such as Z_{ij} for the entries of $\mathbf{Z}$. The Frobenius norm of $\mathbf{Z}$ is $||\mathbf{Z}||_F = \sqrt{\sum_{i=1}^{d}\sum_{j=1}^{k}\mathbf{Z}_{ij}^2}$. $||\mathbf{Z}||_{1,2}^2 = \sum_{i=1}^{n}||\mathbf{z}_i||_1^2 = \sum_{i=1}^{n}\left(\sum_{j=1}^{d}|Z_{ij}|\right)^2$ is the $\ell_{1,2}$-norm of a matrix $\mathbf{Z} \in \mathbb{R}^{N\times d}$.

2. Related work

Image clustering targets at partitioning a group of unlabeled images into several different clusters, such that the images in the same cluster have high correlation to each other. Owing to its superiority in processing unlabeled data, clustering has become an active topic in artificial intelligence field, and it has been widely applied in various data mining tasks, such as image retrieval [7], image segmentation [8], image annotation [9] and document analysis [10].

To date, studies have presented many shallow clustering methods [11–15]. Despite the promising preliminary results, the performance of these shallow clustering methods deteriorates with large-scale real-world datasets due to unreliable similarity metrics. To tackle this problem, a naive method is to transform high dimensional data to a low dimensional feature space by applying hand-crafted feature extraction or dimension reduction techniques. Then, clustering can be performed on the lower feature space. Since the feature learning and clustering are two separate processes, these hand-crafted features ignore the interconnection between them, resulting in suboptimal results.

Motivated by the powerful latent representation learning capabilities of deep learning, numerous deep clustering methods have been proposed, which can be roughly divided into three categories: *auto-encoder based methods*, *adversarial learning based methods* and *the variants of DEC*. Next, we will discuss the characteristics of each category and the corresponding methods.

2.1. Auto-encoder based methods

Auto-encoder (AE) is a classical deep neural network. It aims to extract low-dimensional latent representation of raw samples via an unsupervised manner. For example, Yang et al. [16] proposed deep clustering network (DCN) by joint deep auto-encoder and K-Means clustering objective. Huang et al. [17] proposed a deep embedding network for clustering (DEN) based on deep auto-encoder, in which the locality-persevering and group sparsity are simultaneously taken into account. Based on a multi-layer convolutional denoising autoencoder, Dizaji et al. [18] presented a deep embedded regularized clustering network (DEPICT) by relative entropy minimization. Pan et al. [19] proposed the deep subspace clustering networks (DSC-Net). However, they fail to take cluster-specificity distribution into account, which may result in the learned representation fail to describe cluster structure. In contrast, our proposed ASC2D imposes the cluster-specificity distribution constraint on latent representation to learn better representation.

2.2. The variants of DEC

Considering the great progress of DEC in deep clustering, many variants of DEC have been proposed, *e.g.*, Guo et al. [2] proposed the improved deep embedded clustering model (IDEC) with local structure preservation. Guo et al. [20] proposed deep embedded clustering with data augmentation. Mrabah et al. [3] proposed adversarial deep embedded clustering to better trade-off between feature randomness and feature drift. Despite obtained good performances, they neglect the scale issue of the distributions between the given sample and cluster centers, resulting in unstable clustering performance. Our proposed ASC2D well minimizes the gaps between distributions by leveraging the thought of adversarial learning.

2.3. Adversarial learning based methods

The thought of adversarial learning is the core of the Generative Adversarial Network (GAN), which is one of the most classical deep neural networks. To date, studies proposed several adversarial learning based clustering method, *e.g.*, by employing gaussian mixture distribution, Harchaouiet al. [21] match the aggregated posterior of the latent representation, and proposed a deep adversarial gaussian mixture auto-encoder for clustering (DACEC). Springenberg proposed Categorial Generative Adversarial Network for clustering (CatGAN) [22]. Unlike classic GAN, CatGAN enforces discriminator to classify all samples into k classes, while being unknown of cluster assignments for samples generated by generator. By optimizing the mutual information between a fixed small subset of the GAN's noise variables and the observation, Chen et al. [23] proposed the information maximizing generative adversarial nets (InfoGAN). More recently, Zhou et al. [24] presented deep adversarial subspace clustering (DASC). Mukherjee et al. [25] proposed latent space clustering in generative adversarial networks for clustering (ClusterGAN). Although the impressive clustering results have been obtained by the above methods, they fail to utilize the obtained clustering labels, resulting in suboptimal performances. To handle this problem, our proposed ASC2D use the clustering labels to supervise the optimizing of latent representation, with the target to obtain better clustering performances.

For more detailed perspective of network architecture, please refer to [26], which provided a comprehensive review of deep clustering.

3. Methodology

3.1. Overall framework

Fig. 1 shows the overall framework of the proposed ASC2D. Given an image dataset $\{\mathbf{x}_i^{d\times d} \in \mathbf{X}\}_{i=1}^{N}$ with $\mathscr{C}$ clusters, where d

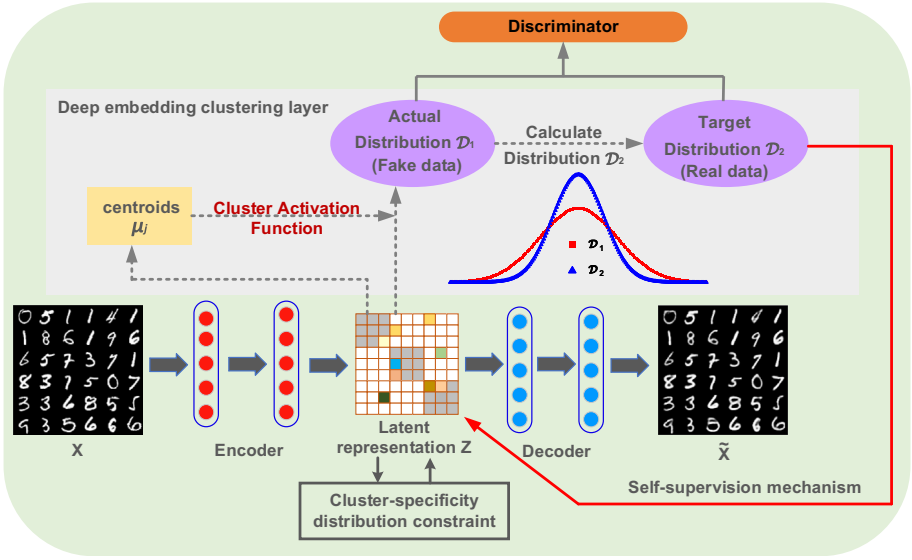


Fig. 1. The overall framework of our proposed ASC2D. It is made up of five sub-networks: 1) Convolutional auto-encoder. It aims to learn low-dimension latent representation. 2) Deep embedding clustering layer. It aims to carry out clustering and obtain clustering labels. 3) Cluster-specificity distribution constraint layer. 4) Self-supervision mechanism. 5) Discriminator. It helps to implement the thought of adversarial learning.

and N indicates the width of a image and the number of images. Inputting the i-th raw image to a multi-layer auto-encoder, we can obtain the corresponding latent representation $\left\{\mathbf{z}_i^{1\times d_z} \in \mathbf{Z}\right\}_{i=1}^{N}$, where d_z is dimension of latent representation. Motivated by a observation that the cluster assignment matrix can always be represented by a low dimensional mapping of the high dimensional data [13], we make full use of clustering labels to self-supervise the learning latent representation, where the clustering labels is obtained from the target distribution $\mathscr{D}_2$. Meanwhile, to make the learned representation can well exploit cluster structure, we imposed the cluster-specificity distribution constraint on latent representation. Moreover, to eliminate the gaps between distributions, *i.e.*, effectively solving the scale issue[1] between distributions, we introduce a distribution consistency adversarial regularization. For ease of explanation, we utilize $\mu_j(j = 1, 2, \ldots \mathscr{C})$ to represent the centroid of each cluster. The tradeoff parameters are donated by $\lambda = [\lambda_1, \lambda_2, \lambda_3]$.

3.2. Overall framework

• **Convolutional Encoder E_n and Convolutional Decoder D_e:** $\mathbf{R}^{d\times d} \rightarrow \mathbf{R}^{\mathscr{C}} \rightarrow \mathbf{R}^{d\times d}$. Encoder, which consists of three convolutional layers and two fully connected layers, aims to learn latent representations $\mathbf{Z}$ of raw data $\mathbf{X}$. We utilize auto-encoder based on the fact that auto-encoder consistently produces semantically meaningful and well-separated representations on real-world datasets in many unsupervised learning methods [19,24]. To be specific, the encoder projects the input data to a low-dimensional representation $\mathbf{Z}$ via a non-linear mapping $\mathbf{z}_i = e(\mathbf{x}_i; \theta)$, where $e(\cdot)$ refers to the non-linear mapping function and $\theta = \left\{\mathbf{W}^{(l)}, \mathbf{b}^{(l)}\right\}$ is the l_{th} layer's learnable parameters of encoder E_n. We attempt to enforce the latent representations to be an approximately continuous label, so the dimension of a representation is $1 \times \mathscr{C}$, where $\mathscr{C}$ is cluster number. Then, a decoder is exploited to reconstruct the input data $\mathbf{X}$ from low-dimensional representations. To ensure that the latent features obtained by the encoder are available, the network minimizes the least mean square loss between $\mathbf{X}$ and $\tilde{\mathbf{X}}$ to update the learnable parameters of E_n and D_e.

• **Deep embedding clustering layer and Self-supervision mechanism.** In this layer, we aim to find a reasonable continuous cluster assignment matrix to self-supervise the encoder to generate low-dimensional, compact, approximately continuous representations. Motivated by *t-SNE* [27], which captures the similarity between two data points, we use the learned latent feature $\mathbf{z}_i$ and the centroid μ_j to characterize the relationship between data point and cluster centroid. Based on this, we can find a reasonable continuous distribution $\mathscr{D}_2$ to supervise the learning of $\mathbf{Z}$, where $\mathscr{D}_2$ is calculated by the frequency of actual distribution $\mathscr{D}_1$ (See Section 3.3). Hence, we utilize this constraint $\|\mathbf{Z} - \mathscr{D}_2\|_F^2$ to guide the auto-encoder to generate more reasonable $\mathbf{Z}$ (See Eq. (6)). Furthermore, in order to improve the latent representations ability of E_n, and make the target distribution $\mathscr{D}_2$ can be adaptively updated according to current clustering results, we not only constrain the mismatch of representation $\mathbf{Z}$ and target distribution $\mathscr{D}_2$, but also minimize the difference between $\mathscr{D}_2$ and actual distribution $\mathscr{D}_1$ to update learnable parameters of E_n and the cluster centroid of each cluster.

• **Cluster-specificity distribution constraint layer.** To well characterize class structure, in this layer, we aim to make the learned latent representation with the same cluster to further have a common distribution in dimension space while representations with different clusters have different distribution in the intrinsic dimension space. To this end, we make full use of the good property of $\ell_{1,2}$-norm, and impose this constraint on the representation $\mathbf{Z}$.

• **Discriminator** $\mathfrak{D} : \mathbf{R}^{N\times\mathscr{K}} \rightarrow \{0, 1\}$. The discriminator, consisting of three fully connected layers, aims to distinguish target distribution $\mathscr{D}_2$ (*real*) and actual distribution $\mathscr{D}_1$ (*fake*), and subsequently supervise the deep clustering layer to improve clustering performance. Although we minimize the error between $\mathscr{D}_2$ and $\mathscr{D}_1$ above, due to scale issue[1], we utilize the adversarial learning to maintain the diversity between $\mathscr{D}_2$ and $\mathscr{D}_1$. Motivated by *t-SNE*, if the data in two spaces are similar, their distribution should be the same. Hence, for discriminator, we hope it can distinguish that $\mathscr{D}_1$ is the actual distribution of latent representations and $\mathscr{D}_2$ is target distribution. Then we minimize the adversarial loss to update the parameters of the discriminator and auto-encoder until the actual distribution in the latent space and the target distribution get sim-

ilar, which also shows that the network has learned a satisfactory representation **Z**.

3.3. Objective function and implementation

The objective of our model contains five terms covering auto-encoder loss $\mathscr{L}_{\text{AE}}$, distribution consistency adversarial regularization loss $\mathscr{L}_{\mathfrak{D}}$, self-supervision loss $\mathscr{L}_{\text{S}}$, distribution consistent loss $\mathscr{L}_{\text{C}}$ and cluster-specificity distribution constraint $\mathscr{L}_{\text{CSD}}$. The overall objective of our method is given by

$$\mathscr{L} = \min_{\theta,\omega,\mu}\max_{\psi} \mathscr{L}_{\text{AE}} + \lambda_1 \mathscr{L}_{\mathfrak{D}} + \mathscr{L}_{\text{C}} + \lambda_2 \mathscr{L}_{\text{S}} + \lambda_3 \mathscr{L}_{\text{CSD}}. \tag{1}$$

where ω, θ and ψ are the parameters of the convolutional encoder, convolutional decoder and discriminator.

• **Auto-encoder Loss w.r.t. θ, ω.** The auto-encoder loss minimizes the least mean square error of reconstructed samples and the original samples, which is defined as

$$\mathscr{L}_{\text{AE}} = \min_{\theta,\omega} \frac{1}{N}(\mathbf{X} - g(e(\mathbf{X};\theta);\omega))^2 = \min_{\theta,\omega} \frac{1}{N}\left(\mathbf{X} - \widetilde{\mathbf{X}}\right)^2, \tag{2}$$

where the output of decoder is $\widetilde{\mathbf{X}} = g(e(\mathbf{X};\theta);\omega), \omega = \left\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\right\}$ represents learnable parameters of D_e in m_{th} layer's. This loss is used for training encoder and decoder. It encourages encoder to catch the essential structure for the latent representation from input data, and the latent representation recovers the real data exactly. The encoder E_n takes **X** as input and learns latent representations $\mathbf{Z} = e(\mathbf{X};\theta)$. The decoder reconstructs **X** from the latent representation **Z**.

We minimize the reconstruction error to optimize auto-encoder network in Eq. (2). Like other unsupervised learning methods, for getting a reasonably good initial representation, we pre-train the auto-encoder via **X**.

Self-supervision loss and distribution consistent loss w.r.t. θ, μ. Given an initial estimate of the non-linear mapping $\mathbf{z}_i \in \mathbf{Z}$, we get a latent representation **Z**. As shown on the left of Fig. 2, we find the learned latent representation is continuous. In order to characterize clustering relationship between representation $\mathbf{z}_i$ of sample i and cluster centroid μ_j, we herein define a distinctive activation function $\mathbb{A}(\cdot,\cdot)$ to characterize the distribution between **Z** and cluster centroids μ. Thus, we have(3)

$$\mathbb{A}\left(\mathbf{z}_i, \mu_j\right) = \frac{\left(1 + ||\mathbf{z}_i - \mu_j||^2\right)^{-1}}{\sum_{j'}\left(1 + ||\mathbf{z}_i - \mu_{j'}||^2\right)^{-1}}, \tag{3}$$

We define the result obtained by Eq. (3) as the actual distribution $\mathscr{D}_1 \in \mathbb{R}^{N\times\mathscr{C}}$. So, we have $d^1_{ij} = \mathbb{A}\left(\mathbf{z}_i, \mu_j\right)$, where $d^1_{ij} \in \mathscr{D}_1$ represents the probability that clusters sample i into j-*th* cluster. The centroid μ_j of each cluster is defined as the trainable variable. We employe the centroids calculated by K-means to initialize μ_j.

It's challenging to find a reasonable target distribution $\mathscr{D}_2$ to regularize representation **Z**. Suppose we map latent representation **Z** to a low-dimensional Ψ. According to *t-SNE*, we need to find a reasonable distribution $\mathscr{D}_2$ of Ψ, if representation **Z** is similar to low-dimensional Ψ, the actual distribution $\mathscr{D}_1$ should be the same as distribution $\mathscr{D}_2$. In our method, we consider the actual distribution $\mathscr{D}_1$ as low-dimensional map Ψ of latent representation **Z**. We hope target distribution $\mathscr{D}_2$ has the following properties: 1) It can further emphasize more on the nodes assigned with high confidence. 2) It can strengthen predictions. 3) It can prevent large clusters from distorting the latent representations of the nodes. Hence, following [1], $\mathscr{D}_2$ is defined as

$$d^2_{ij} = \frac{d^{1\,2}_{ij} / \sum_i d^1_{ij}}{\sum_{j'} d^{1\,2}_{ij'} / \sum_i d^1_{ij'}}, \tag{4}$$

where $\sum_i d^1_{ij}$ is soft cluster frequency. In fact, target distribution intends to further enhance the actual distribution, and it concentrates more on the assigned data with high confidence, so $\mathscr{D}_2$ is called ideal target distribution. We can gain the clustering results directly from the lasted optimized $\mathscr{D}_2$, and the label estimated for sample i can be calculated by $l_i = \max \mathbf{Index}\left(\mathbf{d}^2_i\right)$, where $\max \mathbf{Index}(\cdot)$ is set to find the index of max probability value in i-*th* row of $\mathscr{D}_2$.

Now we have obtained two distributions: target distribution $\mathscr{D}_2$ and actual distribution $\mathscr{D}_1$. Considering the observation that the cluster assignment matrix can always be represented by a low dimensional linear mapping of the high dimensional data [13], we hope the learned latent representation is a kind of approximate continuous label, we utilize the target distribution $\mathscr{D}_2$ to self-supervise the **Z**. To this end, we minimize the square of the F-norm between latent representations **Z** and continuous prior $\mathscr{D}_2$. Thus, the self-supervision loss is defined as

$$\mathscr{L}_{\text{S}} = \min_{\omega} \|\mathbf{Z} - \mathscr{D}_2\|^2_F. \tag{5}$$

In the experiments, we observe the loss of $\|\mathbf{Z} - \mathscr{D}_2\|^2_F$ term is much larger than $\|\mathscr{D}_1 - \mathscr{D}_2\|^2_F$ term, so we set λ_2 to a small value for reasonable optimization.

Going back to [27], we constrain the difference between distribution $\mathscr{D}_1$ and $\mathscr{D}_2$ for the purpose of guiding convolutional auto-encoder to generate more powerful **Z**. We hope that $\mathscr{D}_1$ can get close to $\mathscr{D}_2$ via iterative training. Thus the distribution consistent loss can be defined as

$$\mathscr{L}_{\text{C}} = \min_{\omega,\mu} \|\mathscr{D}_1 - \mathscr{D}_2\|^2_F, \tag{6}$$

Although the term $\|\mathscr{D}_1 - \mathscr{D}_2\|^2_F$ can tackle this problem well, it can only guarantee that the two distribution get similar in some information, which cannot guarantee the diversity of these two distributions due to scale issue[1]. Hence, we introduce the adversarial learning (see Eq. (8)) to supplement this defect.

Cluster-specificity distribution constraint w.r.t. ω. For the learned latent representation, we hope it can it can well characterize the cluster structure. Specifically, we hope the learned latent representation can obtain the intrinsic feature distribution of different clusters. To this end, we impose the cluster-specificity distribution constraint, which is measured by $\ell_{1,2}$-norm, on the latent representation, thus, we have

$$\mathscr{L}_{\text{CSD}} = \min_{\mathbf{Z},\omega} ||\mathbf{Z}||^2_{1,2} = \min_{\mathbf{Z},\omega} \sum_{i=1}^{n} ||\mathbf{z}_i||^2_1 = \min_{\mathbf{Z},\omega} \sum_{i=1}^{n} \left(\sum_{j=1}^{d_z} |Z_{ij}|\right)^2. \tag{7}$$

By minimizing Eq. (7), different elements in squared ℓ_1-norm of i-th row $\mathbf{z}_i$ are competing with each other to survive, and at least one element in row $\mathbf{z}_i$ survives (remaining non-zero). By doing so, some discriminative features are survived for each cluster to provide certain flexibility in the learned latent representation, *i.e.*, making **Z** well preserve the cluster structure.

Algorithm 1. ASC2D

Input: Data $\mathbf{x}_i^{d\times d} \in \mathbf{X}_{i=1}^{N}$, tradeoff parameters λ and cluster number $\mathcal{C}$.

Output: Clustering label l_i of $\mathbf{x}_i \in \mathbf{X}$.

Randomly initialize the parameters of E_n, D_e;
for *not converged* **do**
  // **Step 1** $\longrightarrow$ Pre-train auto-encoder of $\mathcal{G}$
  Updating E_n and D_e by Eq. (2);
end
Use the pre-trained parameters of E_n and D_e to project raw sample and gain $\mathbf{Z}$;
Implement K-means on feature space $\mathbf{Z}$, obtain the initial clustering centroids $\{\mu_j\}_{j=1}^{\mathcal{C}}$;
Calculate initial $\mathcal{D}_1$ and $\mathcal{D}_2$ via Eqs. (3, 4);
Input $\mathcal{D}_1$ and $\mathcal{D}_2$ to discriminator networks;
for *not converged* **do**
  // **Step 2** $\longrightarrow$ Jointly training overall networks.
  Alternately updating $\mathfrak{D}$ and other sub-networks by Eq. (1);
end
for *all* $\mathbf{x}_i \in \mathbf{X}$ **do**
  // **Step 3** $\longrightarrow$ Calculating the clusters
  $\mathrm{l}_i := \mathbf{maxIndex}(\mathbf{d}_i^2), \mathbf{d}_i^2 \in \mathcal{D}_2^{N\times\mathcal{C}}$;
end
return: Cluster label $\mathbf{L}$.

Distribution consistency adversarial regularization loss w.r.t. ω, μ and ψ. We leverage discriminator to make sure the diversity between the actual distribution $\mathscr{D}_1$ and the target distribution $\mathscr{D}_2$ when optimizing Eq. (6). By feeding back the evaluation information, the discriminator can supervise the generator to improve clustering performance. Specifically, we combine the encoder with a discriminator to apply a GAN-*alike* model's idea in our method. The loss function of generator, *i.e.*, **the encoder and the clustering layer**, minimizes the likelihood that actual distribution $\mathscr{D}_1$ assigns to the fake source, while the discriminator $\mathfrak{D}$ maximizes the likelihood that $\mathscr{D}_1$ is assigned to the fake source, so the loss function of adversarial learning is

$$\mathscr{L}_{\mathfrak{D}} = \min_{\omega,\mu}\max_{\psi}\sum_{i=1}^{N}\mathbb{E}\left[\log\left(\mathfrak{D}\left(\mathbf{d}_i^2\right)\right)\right] + \mathbb{E}\left[\log\left(1-\mathfrak{D}\left(\mathbf{d}_i^1\right)\right)\right]. \tag{8}$$

For all datasets, we employ a three-layer fully connected neural network as discriminator.

The generator is trained to generate actual distribution $\mathscr{D}_1$ which is similar to target distribution $\mathscr{D}_2$. The $\mathfrak{D}$ is trained to distinguish the mismatch between $\mathscr{D}_1$ and $\mathscr{D}_2$, which can not only make sure the diversity of $\mathscr{D}_1$ and $\mathscr{D}_2$, but also help to evaluate the quality of clustering results. They play a min–max game until convergence. The adversarial loss can assist the encoder E_n in mapping a given sample $\mathbf{X}$ to a desired output $\mathbf{Z}$. Thus, the combination of adversarial learning and above constraint further ensures the encoder project the input to a desired output, *i.e.*, generating effective sample representation and satisfactory performance. Due to the adversarial training may usually suffer from the stability issue as widely discussed in GAN models, and the network architecture of discriminator $\mathfrak{D}$ is shallower than auto-encoder, in our method, we alternately update $\mathfrak{D}$ and auto-encoder for 3 times and 1 time within each epoch respectively. We report a brief implementation of ASC2D in Algorithm 1.

4. Experiments

4.1. Datasets and experimental settings

4.1.1. Datasets

Two handwritten digit datasets, *i.e.*, MNIST [28] and USPS[2], one products dataset, *i.e.*, Fashion-MNIST [29], and two face datasets, *i.e.*, FRGC-v2.0[3] and Youtube-Face (YTF) [30] are used to verify the effectiveness of our method with three frequently-used measures, *i.e.*, Accuracy (ACC), Normalized Mutual Information (NMI) [31] and running time (T). Brief statistics of the datasets are shown in Table 1. **Note that** image clustering aims to partition a group of unlabeled image into several different clusters. Therefore, for all datasets, all image data are assigned to train the overall network.

4.1.2. Experimental settings

In our experiment, all network parameters and their values are listed in Table 2. For the proposed method, we implement it in Ten-

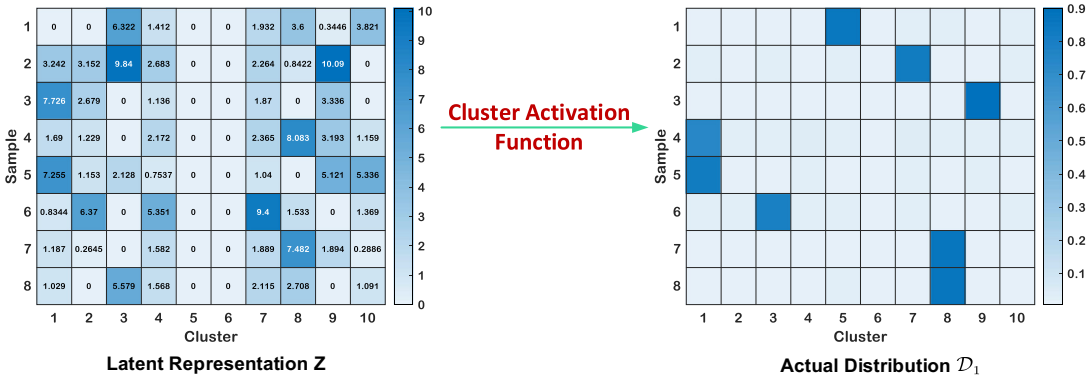


Fig. 2. The illustration shows the cluster activation function. We aims to utilize this cluster activation function to obtain a reasonable and continuous actual distribution of latent representation, in which the corresponding values between 0 and 1.

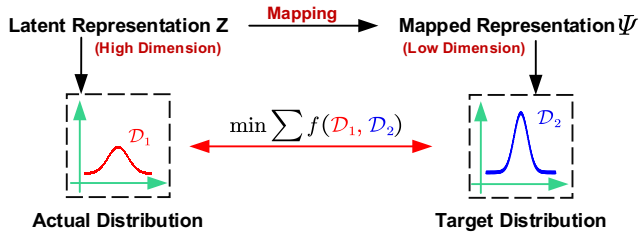


Fig. 3. The illustration shows the method to calculate $\mathscr{D}_2$.

Table 1
Descriptions of datasets, where ♯ means the number of.

Dataset	Attribute dim	♯ Categories	♯ Samples
MNIST-full [28]	$28 \times 28 \times 1$	10	70,000
MNIST-test [28]	$28 \times 28 \times 1$	10	10,000
Fashion-MNIST [29]	$28 \times 28 \times 1$	10	70,000
USPS[2]	$16 \times 16 \times 1$	10	11,000
FRGC[31]	$32 \times 32 \times 3$	20	2,462
YTF [30]	$55 \times 55 \times 3$	41	10,000

sorFlow 1.13.1 platform based on Python 3.6. All the experiments (including some baseline algorithms) are conducted on a machine with two NVIDIA Tesla P100-PICE GPUs, the Intel(R) Xeon(R) Gold 6230 CPU and 128 GB RAM. The visualizations of experimental results are processed on MATLAB R2020a. The Adam optimization algorithm [32] is adopted to optimize the proposed ASC2D. The encoder consist of three convolutional layers and a fully connected layer, and their kernel size of convolutional layers are always set to $5 \times 5, 5 \times 5$ and 3×3 with stride $= 2$, respectively. A symmetrical structure to encoder is set for decoder. We utilize ReLU activation function for all layers of convolutional auto-encoder. For the discriminator $\mathfrak{D}$, we utilize three fully connected layers with $2000 \rightarrow 2000 \rightarrow 1$ output neurons, where ReLU activation function is employed for the first two hidden layers and Softmax activation function is employed for output layer. For all datasets, in order to get reasonably good initial representation parameters, we pre-train the auto-encoder network via Eq. (2).

4.2. Comparison methods

We compare the proposed method with 14 clustering methods including K-means [11], spectral embedded clustering (SEC[4]) [13], DEC[5] [1], IDEC[6] [2], joint unsupervised learning (JULE) [33], DEPICT [18], DSC-Net[7] [19], deep Spectral Net [34], deep spectral clustering using dual auto-encoder network (DSCDAN[8]) [35], DACEC [21], CatGAN[9] [22], InfoGAN [23], ADEC [3] and ClusterGAN[10] [25].

For fair comparison, we utilize the same datasets provided by [33,18] with same pre-processing method. For all datasets except Fashion-MNIST, the results of K-means, JULE and DEPICT are obtained from [18]. For SEC, the results on MNIST-full and MNIST-test datasets are also obtained from [18]. For DEC, the results on MNIST-full, MNIST-test and USPS datasets are obtained from [18]. For IDEC and DSCDAN, the results of all datasets are obtained from [35], except FRGC and YTF datasets. The results of SpectralNet are obtained from [35]. The results of InfoGAN and ClusterGAN on MNIST-full and Fashion-MNIST are obtained from [25]. For Fashion-MNIST dataset, the results of K-means, JULE and DEPICT are obtained from [35]. The results of ADEC is used from [3]. Besides, for aforementioned compared methods, if the clustering results of their methods on some datasets are not reported in their corresponding paper, we run their corresponding released code with hyper-parameters mentioned in their papers, and the results are marked by (★) in the upper right corner. When the code is not publicly available, or running the released code is not practical, we put (–) instead of the corresponding results. We report the mean values of the two metrics for each algorithm after executing 10 times in Table 3, the running time results of some typical deep clustering methods are reported in Table 4.

4.3. Experiment results and analysis

Tables 3 and 4 list the results of baselines on six datasets. From Tables 3 and 4, we have the following interesting observations:

- In general, the proposed ASC2D achieves satisfactory clustering performance on three clustering tasks, *i.e.,* handwritten digits recognition, products recognition and face recognition. This shows that ASC2D has the ability to handle large-scale and complex image data in reality.
- Comparing the experimental results with the shallow methods, *i.e.,* K-means and SEC, ASC2D achieves best clustering results. The reason is our method utilizes deep convolutional auto-encoder, which can better learn image representation so that the encoder of generator can easily deal with the complex hand-written images (shifting, rotation and so on).
- Comparing with DEC and IDEC, our proposed achieves the best results on all datasets in both ACC and NMI metrics. This is probably because that ASC2D constrains the latent representations to be a kind of compact, continuous label via self-supervision. Moreover, we introduce the adversarial learning between two distributions, which can help deal with scale issue.
- On the MNIST-test and USPS dataset, our method does not seem to perform as well as ADEC on ACC metric, the reason may be that ADEC employed the good property of data augmentation. Previous works [20,36] have proved that data augmentation based on prior knowledge leads to better clustering results. However, from Table 4, it can be seen that our proposed ASC2D is more efficient than ADEC.
- For a comprehensive comparison, ASC2D achieves remarkable improvements comparing with these GAN-based clustering approaches. For example, on the Fashion-MNIST datasets, the accuracy of our method is 7.3% at least higher than that of the best GAN-based approach Cluster-GAN, this is because when optimizing latent representation, our proposed ASC2D take cluster-specificity distribution into account, which helps to make the learned representation well preserve the cluster structure.
- As reported in Table 4, we can see that our method is faster than other comparison methods when dealing with large-scale image data. This performance again demonstrates the practicality of our method for real-world image clustering tasks.

4.3.1. Sensitivity analysis

In ASC2D, there are three tradeoff parameters λ_1, λ_2 and λ_3. For convenience, we herein test the sensitivity of ASC2D *w.r.t.* the parameter λ_1 of adversarial learning and ratio λ_2 of self-supervision constraint term. We first analyze the sensitivity of the parameter λ_1. The tested range is $[0, 1]$. The ACC and NMI metric

[4] urlhttps://github.com/xdweixia/Spectral-embedded-clustering.
[5] urlhttps://github.com/HaebinShin/dec-tensorflow.
[6] urlhttps://github.com/XifengGuo/IDEC.
[7] urlhttps://github.com/panji1990/Deep-subspace-clustering-networks.
[8] urlhttps://github.com/xdxuyang/Deep-Spectral-Clustering-using-Dual-Autoencoder-Network.
[9] urlhttps://github.com/xinario/catgan_pytorch.
[10] urlhttps://github.com/sudiptodip15/ClusterGAN.

Table 2
Details of networks parameters, where ♯ means the number of.

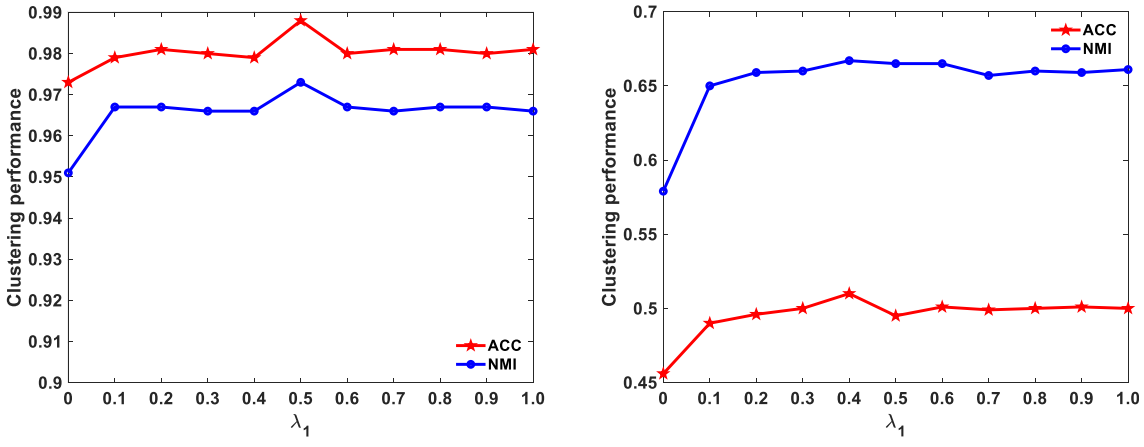
Parameters	Value	Parameters	Value	♯ channel of **G**	Value
λ_1	0.5	Learning rate of pre-train	3×10^{-3}	Encoder-1/decoder-3	32
λ_2	10^{-4}	Learning rate of auto-encoder	3×10^{-3}	Encoder-2/decoder-2	64
λ_3	10^{-5}	Learning rate of $\mathfrak{D}$	10^{-3}	Encoder-3/decoder-1	128

Table 3
Clustering results of various methods on six datasets. Best results are highlighted in **bold**. " - -" means the results are unavailable from the corresponding paper or code. The data marked with ☆ in the upper right corner is obtained by running the code provided by the author.

Dataset	MNIST-full		MNIST-test		Fashion-MNIST		USPS		FRGC		YTF	
Method \ Metric	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
K-means [11]	0.534	0.500	0.547	0.501	0.474	0.512	0.460	0.450	0.243	0.287	0.560	0.752
SEC [13]	0.804	0.779	0.815	0.790	–	–	0.732∗	0.691∗	0.443∗	0.544∗	0.591∗	0.757∗
JULE [33]	0.964	0.913	0.961	0.915	0.563	0.608	0.950	0.913	0.461	0.574	0.684	0.848
DEC [1]	0.844	0.816	0.859	0.827	0.574∗	0.627∗	0.619∗	0.586∗	0.425∗	0.561∗	0.422∗	0.602∗
IDEC [2]	0.881	0.867	0.846	0.802	0.586∗	0.624∗	0.759∗	0.777∗	0.453∗	0.576∗	0.445∗	0.626∗
DEPICT [18]	0.965	0.917	0.963	0.915	0.529	0.557	0.964	0.927	0.470	0.610	0.621	0.802
DSC-L2 [19]	0.715∗	0.704∗	0.717∗	0.700∗	0.578	0.572	0.689∗	0.735∗	0.429∗	0.512∗	0.568∗	0.741∗
SpectralNet [34]	0.971	0.924	0.773	0.760	–	–	–	–	–	–	0.685	0.798
DACEC [21]	0.965	0.952	0.953	0.932	–	–	0.934	0.903	–	–	–	–
CatGAN [22]	0.957∗	0.936∗	0.949∗	0.924∗	0.625∗	0.643∗	0.921∗	0.893∗	–	–	–	–
InfoGAN [23]	0.870	0.840	0.852∗	0.834∗	0.610∗	0.590∗	0.840	0.813	0.440∗	0.564∗	0.640∗	0.835∗
ADEC [3]	0.986	0.961	**0.985**	0.957	0.586	0.662	**0.981**	0.948	–	–	–	–
ClusterGAN [25]	0.950	0.890	0.946	0.919	0.630∗	0.640∗	0.934	0.903	0.450∗	0.569∗	0.650∗	0.837∗
DSCDAN [35]	0.978	0.941	0.980	0.946	0.662	0.645	0.869	0.857	0.356∗	0.519∗	0.691	0.857
ASC2D	**0.988**	**0.973**	0.980	**0.969**	**0.703**	**0.716**	0.969	**0.986**	**0.510**	**0.667**	**0.780**	**0.896**

Table 4
Comparison of the execution times (in seconds) of different deep clustering approaches on six datasets.

Dataset	MNIST-full	MNIST-test	Fashion-MNIST	USPS	FRGC	YTF
JULE [33]	28800.59	1350.27	26332.58	1224.92	1105.65	5987.25
DEC [1]	1674.46	1606.9	1621.57	1474.08	1750.82	2425.62
IDEC [2]	1914.61	1673.26	1806.32	1488.46	1773.68	2535.73
DEPICT [18]	11045.32	1325.43	9954.61	1257.64	850.72	2155.29
ADEC [3]	10249.55	9983.41	10156.32	8216.20	–	–
ClusterGAN [25]	33285.71	–	34045.11	–	–	–
DSCDAN [35]	2047.4	2100.22	2003.77	1977.76	1310.45	**1215.53**
ASC2D	**1288.05**	**1037.48**	**1098.61**	**821.82**	**760.84**	2935.80



(a) MNIST-full (b) FRGC

Fig. 4. Sensitivity analysis of parameter λ_1 in ASC2D.

values on MNIST-full and FRGC datasets of different $\lambda_1 \in [0, 1]$ are shown in Fig. 4, from which we can observe that our method performs stably in a wide range of λ_1. When we make this experiment, the parameter λ_2 is a constant (10^{-4}). Next, we test the sensitivity of the parameter λ_2, in which we set $\lambda_1 = 0.5$. Due to the fact that the self-supervision constraint term $\|\mathbf{Z} - \mathscr{D}_2\|_F^2$ in Eq. (5) is a huge

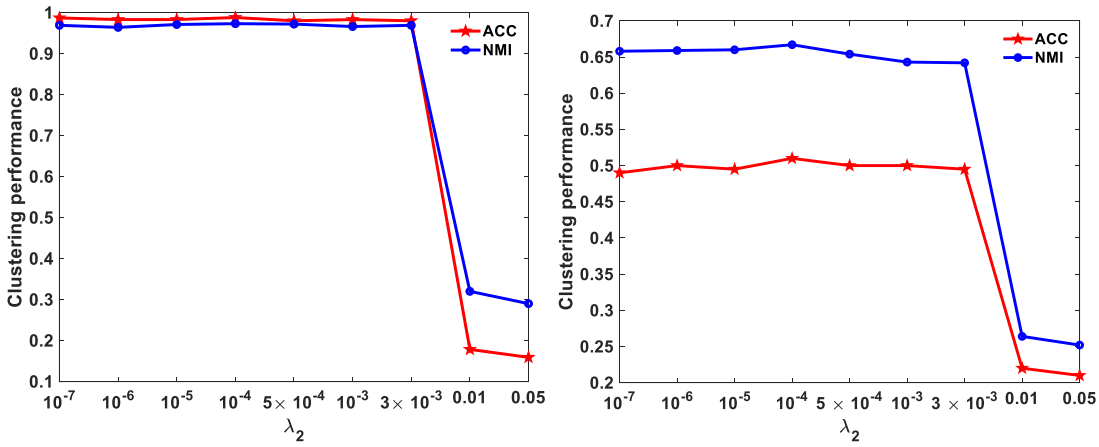


Fig. 5. Sensitivity analysis of parameter λ_2 in ASC2D.

Table 5
The clustering results of different versions of ASC2D on six datasets, where × denote the network does not contain discriminator network $\mathfrak{D}$ and ✓ is exactly the opposite.

Dataset	MNIST-full		MNIST-test		Fashion-MNIST		USPS		FRGC		YTF	
Metric	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
×	0.976	0.962	0.965	0.953	0.688	0.665	0.951	0.979	0.456	0.579	0.733	0.865
✓	**0.988**	**0.973**	**0.980**	**0.969**	**0.703**	**0.716**	**0.969**	**0.986**	**0.510**	**0.667**	**0.780**	**0.896**

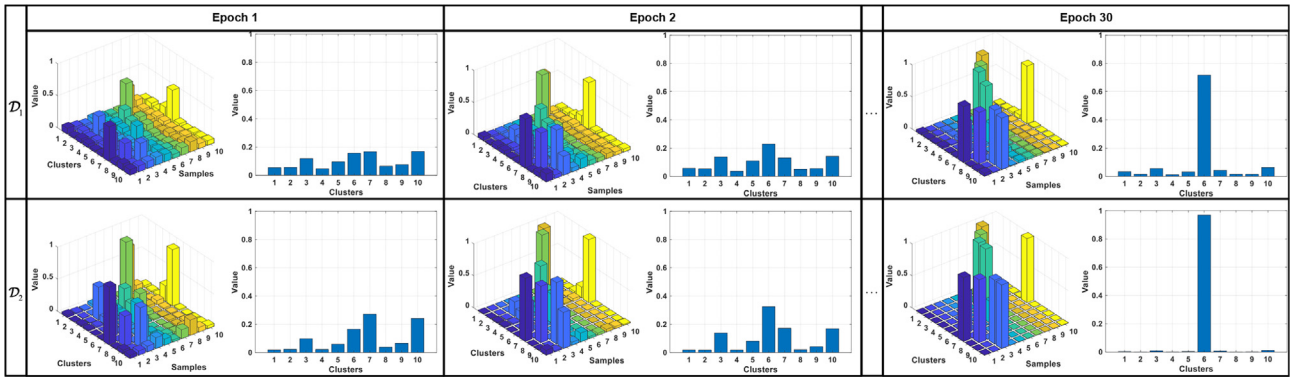


Fig. 6. We visualize the actual distribution and target distribution in adversarial learning phase of a random ten samples of fashion-MNIST dataset. To make the results more obvious, we take the fourth sample as an example to visualize the changes with training epochs.

number, so we set $\lambda_2 \in \left[10^{-7}, 5\times 10^{-2}\right]$ to keep the objective balanced. As shown in Fig. 5, our method achieves stably performance in a wide range of λ_2. When λ_{21} is bigger than 5×10^{-3}, the clustering loss can't maintain balance among these terms in total objective function, which lead to bad clustering results. Therefore, the default values of the parameter λ_1 and λ_2 are recommended to be set to 0.5 and 10^{-4}, respectively.

4.4. Discussions of adversarial learning strategy

According to model (1), we discard the discriminator networks $\mathfrak{D}$ for proving the effectiveness of adversarial learning strategy. Hence, for the version without discriminator network, its objective can be represented as

$$\mathscr{L} = \min_{\omega,\theta,\mu} \mathscr{L}_{\mathrm{AE}} + \mathscr{L}_{\mathrm{C}} + \lambda_2 \mathscr{L}_{\mathrm{S}} + \lambda_3 \mathscr{L}_{\mathrm{CSD}}, \tag{9}$$

where λ_2 and λ_3 are two tradeoff parameters. For a fair comparison, for model (9), we tune λ_2 and λ_3 in range of $\left[10^{-7}, 5\times 10^{-2}\right]$ to gain the best clustering results. We run the version with discriminator network and the version without discriminator network with 10 random trials and report the average performance in Table 5.

4.4.1. Performances

As reported in Table 5, *w.r.t.* different training strategies, we find that the performance when adding $\mathscr{D}_2, \mathscr{D}_1$ adversarial learning outperforms the methods without $\mathscr{D}_2, \mathscr{D}_1$ adversarial learning on both two clustering quality measures, especially in Fashion-MNIST, FRGC and YTF dataset. As they utilize the same network for latent representation learning, the better performance of the version with discriminator network is benefited from adversarial learning between actual distribution $\mathscr{D}_1$ and target distribution

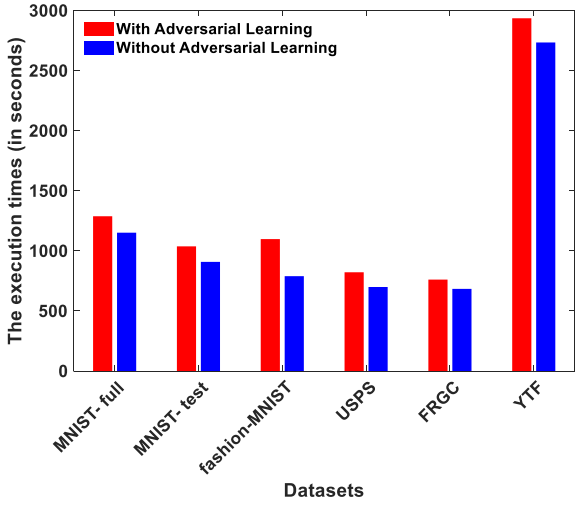


Fig. 7. The time costs of our method with and without the adversarial learning.

$\mathscr{D}_2$, which is effective at learning better latent representation **Z** thereby improving clustering performance.

4.4.2. Visualization

As reported in Fig. 6, we visualize the target distribution and actual distribution of a random 10 samples obtained from fashion-MNIST dataset during adversarial learning, where the left color *3-D* bar shows the 10 samples cluster assignment $\mathscr{D}_1$ (above) and target distribution $\mathscr{D}_2$ (below). The right *2-D* bar clearly shows the distributions $\mathscr{D}_1$ and $\mathscr{D}_2$ of the **sixth** sample. It can be seen that the distribution difference of $\mathscr{D}_1$ and $\mathscr{D}_2$ becomes smaller and smaller. As aforementioned, we hope the scale issue[1] can be tackled. That is to say, actual distribution can get close to target distribution with diversity, that means the clustering performance is excellent. Comparing with each training epoch, we find that our method performs well. Hence, the $\mathscr{D}_2, \mathscr{D}_1$ adversarial learning in Eq. (1) is advantageous in the process of learning latent representations.

4.4.3. Execution times

Finally, to more comprehensively analysis our proposed with and without the adversarial learning, we evaluate the time costs of our proposed method with and without the discriminator network. As reported in Fig. 7, *w.r.t.* different strategies on six datasets, time costs caused by discriminator network is acceptable. The reason may be that the discriminator network is composed of three-layer fully connected network, in this case, the number of parameters in the discriminator network is much smaller than the deep convolutional auto-encoder in our method. By contrast, the time costs of optimizing discriminator network is relatively little.

4.5. Discussions of cluster-specificity distribution constraint

We empirically analyze the effectiveness of cluster-specificity distribution constraint. The clustering performances for six dataset are reported in Table 6. We observe that the clustering performances are improved by the cluster-specificity distribution constraint. These results convey that cluster-specificity distribution constraint is a key technical choice for representation learning and clustering.

5. Conclusion and future works

We propose a adversarial self-supervised network with cluster-specificity distribution (ASDCN) for image clustering tasks. Different from previous works, to well characterize the cluster structure, we introduce the cluster-specificity distribution to constrain the learning of latent representation. Meanwhile, a reasonable adversarial regularization has been adopted to eliminate the gaps between actual distribution and target distribution. Moreover, with the aim to achieve better clustering results, ASDCN seamlessly connects the clustering and representation learning via self-supervision. Extensive experiments results demonstrate the superiority of ASDCN. In the future, we will take data augmentation into account in deep clustering.

CRediT authorship contribution statement

Wei Xia: Conceptualization, Methodology, Software, Investigation, Writing - original draft. **Xiangdong Zhang:** Validation, Formal analysis, Visualization, Software. **Quanxue Gao:** Conceptualization, Methodology, Supervision, Writing - review & editing. **Xinbo Gao:** Supervision, Writing - review & editing.

Declaration of Competing Interest

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

Acknowledgements

The authors would like to thank the anonymous reviewers and AE for their constructive comments and suggestions. This work is supported by the National Natural Science Foundation of China under Grants 61773302, Natural Science Basic Research Plan in Shaanxi Province under Grant 2020JZ-19 and 2020JQ-327, the Initiative Postdocs Supporting Program BX20190262, the Fundamental Research Funds for the Central Universities and the Innovation Fund of Xidian University.

References

[1] J. Xie, R. Girshick, A. Farhadi, Unsupervised deep embedding for clustering analysis, Proc. ICML (2016) 478–487.
[2] X. Guo, L. Gao, X. Liu, J. Yin, Improved deep embedded clustering with local structure preservation, Proc. IJCAI (2017) 1753–1759.
[3] N. Mrabah, M. Bouguessa, R. Ksantini, Adversarial deep embedded clustering: on a better trade-off between feature randomness and feature drift, IEEE Trans. Knowl. Data Eng. (2020) 1–17, https://doi.org/10.1109/TKDE.2020.2997772.
[4] X. Peng, H. Zhu, J. Feng, C. Shen, H. Zhang, J.T. Zhou, Deep clustering with sample-assignment invariance prior, IEEE Trans. Neural Networks Learn. Syst. 31 (11) (2020) 4857–4868.
[5] S. Deng, W. Xia, Q. Gao, X. Gao, Cross-view classification by joint adversarial learning and class-specificity distribution, Pattern Recognit. 110. doi:10.1016/j.patcog.2020.107633..

Table 6
Clustering results of our proposed ASC2D with and without cluster-specificity distribution constraint.

Dataset	MNIST-full		MNIST-test		Fashion-MNIST		USPS		FRGC		YTF	
Method \ Metric	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
ASC2D-without ℓ_{12}-norm	0.981	0.967	0.970	0.954	0.689	0.695	0.969	0.984	0.460	0.625	0.750	0.881
ASC2D	**0.988**	**0.973**	**0.980**	**0.969**	**0.703**	**0.716**	**0.969**	**0.986**	**0.510**	**0.667**	**0.780**	**0.896**

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Proc. NeurIPS (2014) 2672–2680.
[7] P. Xie, E.P. Xing, Integrating image clustering and codebook learning, Proc. AAAI (2015) 1903–1909.
[8] J. Shi, J. Malik, Normalized cuts and image segmentation, Departmental Papers (CIS) (2000) 107.
[9] J. Li, J.Z. Wang, Real-time computerized annotation of pictures, IEEE Trans. Pattern Anal. Mach. Intell. 30 (6) (2008) 985–1002.
[10] S. Huang, Z. Xu, J. Lv, Adaptive local structure learning for document co-clustering, Knowl.-Based Syst. 148 (2018) 74–84.
[11] J. MacQueen, et al., Some methods for classification and analysis of multivariate observations, in: Proc. Mathematical Statistics and Probability, 1967, pp. 281–297..
[12] L. Kaufman, P.J. Rousseeuw, Partitioning around medoids (program pam), Finding Groups in Data: An Introduction to Cluster Analysis 344 (1990) 68–125.
[13] F. Nie, Z. Zeng, I.W. Tsang, D. Xu, C. Zhang, Spectral embedded clustering: a framework for in-sample and out-of-sample spectral clustering, IEEE Trans. Neural Networks 22 (11) (2011) 1796–1808.
[14] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, Y. Ma, Robust recovery of subspace structures by low-rank representation, IEEE Trans. Pattern Anal. Mach. Intell. 35 (1) (2013) 171–184.
[15] Q. Gao, W. Xia, Z. Wan, D. Xie, P. Zhang, Tensor-svd based graph learning for multi-view subspace clustering, Proc. AAAI (2020) 3930–3937.
[16] B. Yang, X. Fu, N.D. Sidiropoulos, M. Hong, Towards k-means-friendly spaces: simultaneous deep learning and clustering, in: Proceedings of the 34th International Conference on Machine Learning, PMLR, 2017, pp. 3861–3870.
[17] P. Huang, Y. Huang, W. Wang, L. Wang, Deep embedding network for clustering, in: 22nd International Conference on Pattern Recognition, ICPR, 2014, pp. 1532–1537.
[18] K. Ghasedi Dizaji, A. Herandi, C. Deng, W. Cai, H. Huang, Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization, in: Proc. IEEE ICCV, 2017, pp. 5736–5745..
[19] P. Ji, T. Zhang, H. Li, M. Salzmann, I. Reid, Deep subspace clustering networks, Proc. NeurIPS (2017) 24–33.
[20] X. Guo, E. Zhu, X. Liu, J. Yin, Deep embedded clustering with data augmentation, Proc. ACML (2018) 550–565.
[21] W. Harchaoui, P. Mattei, C. Bouveyron, Deep adversarial gaussian mixture auto-encoder for clustering, in: Proc. ICLR Workshop, 2017, pp. 1–5..
[22] J.T. Springenberg, Unsupervised and semi-supervised learning with categorical generative adversarial networks, Proc. ICLR (2016).
[23] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, P. Abbeel, Infogan: Interpretable representation learning by information maximizing generative adversarial nets, Proc. NeurIPS (2016) 2172–2180.
[24] P. Zhou, Y. Hou, J. Feng, Deep adversarial subspace clustering, in: Proc. IEEE CVPR, 2018, pp. 1596–1604..
[25] S. Mukherjee, H. Asnani, E. Lin, S. Kannan, Clustergan: latent space clustering in generative adversarial networks, Proc. AAAI (2019) 4610–4617.
[26] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, J. Long, A survey of clustering with deep learning: from the perspective of network architecture, IEEE Access 6 (2018) 39501–39514.
[27] L. v. d. Maaten, G. Hinton, Visualizing data using t-sne, JLMR 9 (2008) 2579–2605..
[28] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al., Gradient-based learning applied to document recognition, Proc. of the IEEE 86 (11) (1998) 2278–2324.
[29] H. Xiao, K. Rasul, R. Vollgraf, Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, CoRR abs/1708.07747..
[30] L. Wolf, T. Hassner, I. Maoz, Face recognition in unconstrained videos with matched background similarity, in: Proc. IEEE CVPR, 2011, pp. 529–543..
[31] Y. Luo, T. TIAN, J. Shi, J. Zhu, B. Zhang, Semi-crowdsourced clustering with deep generative models, in: Proc. NeurIPS, 2018, pp. 3212–3222..
[32] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, Proc. ICLR (2015).
[33] J. Yang, D. Parikh, D. Batra, Joint unsupervised learning of deep representations and image clusters, in: Proc. IEEE CVPR, 2016, pp. 5147–5156..
[34] U. Shaham, K.P. Stanton, H. Li, R. Basri, B. Nadler, Y. Kluger, Spectralnet: spectral clustering using deep neural networks, ICLR (2018).
[35] X. Yang, C. Deng, F. Zheng, J. Yan, W. Liu, Deep spectral clustering using dual autoencoder network, CVPR (2019) 4066–4075.
[36] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, M. Sugiyama, Learning discrete representations via information maximizing self-augmented training, in: Proc. ICML, 2017, pp. 1558–1567..

Wei Xia received the B.Eng. degree in Communication Engineering from Lanzhou University of Technology, Lanzhou, China, in 2018. He is currently pursuing the Ph.D. degree in communication and information system in Xidian University, Xi'an, China. His research interests include pattern recognition, machine learning and deep learning.

Xiangdong Zhang received the B.Eng. Degree, the M.S. degree and the Ph.D. degree from Xidian University, Xi'an, China, in 1992, 1995, and 1998, respectively. He is currently an associate professor with the school of Telecommunications Engineering, Xidian University. His current research interests include pattern recognition and machine learning.

Quanxue Gao received the B. Eng. degree from Xi'an Highway University, Xi'an, China, in 1998, the M.S. degree from the Gansu University of Technology, Lanzhou, China, in 2001, and the Ph.D. degree from Northwestern Polytechnical University, Xi'an China, in 2005. He was an associate research with the Biometrics Center, The Hong Kong Polytechnic University, Hong Kong from 2006 to 2007. From 2015 to 2016, he was a visiting scholar with the department of computer science, The University of Texas at Arlington, Arlington USA. He is currently a professor with the School of Telecommunications Engineering, Xidian University, and also a key member of State Key Laboratory of Integrated Services Networks. He has authored 60 technical articles in refereed journals and proceedings, including IEEE Transactions on Image Processing, IEEE Transactions on Neural Networks and Learning Systems, IEEE Transactions on Cybernetics, CVPR, AAAI, and IJCAI. His current research interests include pattern recognition and machine learning.

Xinbo Gao received the B.Eng., M.Sc. and Ph.D. degrees in electronic engineering, signal and information processing from Xidian University, Xi'an, China, in 1994, 1997, and 1999, respectively. From 1997 to 1998, he was a research fellow at the Department of Computer Science, Shizuoka University, Shizuoka, Japan. From 2000 to 2001, he was a post-doctoral research fellow at the Department of Information Engineering, the Chinese University of Hong Kong, Hong Kong. Since 2001, he has been at the School of Electronic Engineering, Xidian University. He is currently a Cheung Kong Professor of Ministry of Education of PR China, a Professor of Pattern Recognition and Intelligent System of Xidian University and a Professor of Computer Science and Technology of Chongqing University of Posts and Telecommunications. His current research interests include Image processing, computer vision, multimedia analysis, machine learning and pattern recognition. He has published six books and around 300 technical articles in refereed journals and proceedings. Prof. Gao is on the Editorial Boards of several journals, including Signal Processing (Elsevier) and Neurocomputing (Elsevier). He served as the General Chair/Co-Chair, Program Committee Chair/Co-Chair, or PC Member for around 30 major international conferences. He is a Fellow of the Institute of Engineering and Technology and a Fellow of the Chinese Institute of Electronics.