Learning the Precise Feature for Cluster Assignment

Yanhai Gan, Xinghui Dong, Huiyu Zhou, Feng Gao, and Junyu Dong

***Abstract*—Clustering is one of the fundamental tasks in computer vision and pattern recognition. Recently, deep clustering methods (algorithms based on deep learning) have attracted wide attention with their impressive performance. Most of these algorithms combine deep unsupervised feature learning and standard clustering together. However, the separation of feature extraction and clustering will lead to suboptimal solutions because the two-stage strategy prevents representation learning from adapting to subsequent tasks (e.g., clustering according to specific cues). To overcome this issue, efforts have been made in the dynamic adaption of representation and cluster assignment, whereas current state-of-the-art methods suffer from heuristically constructed objectives with representation and cluster assignment alternatively optimized. To further standardize the clustering problem, we formulate the objective of clustering as finding a precise feature as the cue for cluster assignment. Based on this, we propose a general-purpose deep clustering framework which radically integrate representation learning and clustering into an individual pipeline for the first time. The proposed framework exploits the powerful ability of recently progressed generative models for learning intrinsic features, and imposes an entropy minimization on the distribution of cluster assignment by a variational algorithm. Experimental results show that the performance of our method is superior, or at least comparable to, the state-of-the-art methods on the handwritten digit recognition, race recognition and object recognition benchmark datasets.**

***Index Terms*—Deep clustering, representation learning, generative models, entropy minimization, variational algorithm.**

I. Introduction

DEEP neural networks (DNNs) have demonstrated their powerful ability in computer vision tasks, such as object detection [1], classification [2], instance segmentation [3] and scene understanding [4]. However, the training of a robust and efficient DNN generally requires a large amount of annotated data. For example, over one million labeled images divided into 1000 categories are contained in the ImageNet dataset [5], and more than 375 million noisy labels are assigned to the 300 million images in the JFT-300M dataset [6]. As known, it is very time-consuming and labor-expensive to collect such a large-scale annotation set [7], [8]. On the other hand, large quantities of images, videos, and other types of data are produced every day. It is indeed impractical to manually annotate these data. Therefore, it is crucial to develop methods that can automatically exploit knowledge from unlabeled data.

Neuroscientists have shown that the naturalistic visual experience plays a fundamental role in learning invariant representations, and such an experience is important to developing a powerful visual system [9], [10]. This indicates that unsupervised learning happens constantly in the human perceptual system. Normally, unsupervised learning methods model the underlying structure or distribution of the input data without annotation [11]. As an unsupervised learning paradigm, clustering aims to divide the input data into a set of clusters according to the distributional attributes of the data [12]–[19]. However, standard clustering algorithms usually depend on some predefined distance metrics which are usually difficult to identify for high dimensional data [12]–[14], [20]–[24]. Furthermore, the time complexity of standard clustering algorithms will dramatically increase when large-scale datasets are encountered [25].

To mitigate these issues faced by standard clustering methods, researchers first embedded the input data into a new low-dimensional space and then implemented a standard clustering method in the embedding space [26]–[29]. Correspondingly, the problem is divided into two phases: representation learning and clustering. In this scheme, representation learning is agnostic to the following clustering task, and thus can hardly produce the representative features for a specific task. Therefore, some efforts have been made to dynamically adapt the representation and cluster assignment [30]–[36]. These methods generally assumes that the label of each cluster can be used as supervisory signals to learn representations and in turn the representations will be beneficial to image clustering. Consequently, the core idea of these methods is to apply a strategy to alternate between representation learning and clustering [34]–[39]. Although such kind of methods have produced promising clustering results, the heuristically constructed objective lacks a principled characterization of goodness of deep clustering, thus making the good performance of deep clustering models customized [40], [41].

Rather than conducting representation learning and clustering separately, humans tend to take into account these two tasks as a whole. For instance, one is likely to perform clustering according to the gender when he/she is asked to divide his/her colleagues into two groups. Nevertheless, he/she can also consider other characteristics, such as position, age and income, for clustering in terms of desired groups. That is to say, humans tend to discover the exactly matched features with regard to the desired number of groups and perform clustering accordingly. Inspired by this, we define the objective of deep clustering as finding a precise feature as the cue for cluster assignment. This objective provides a fresh avenue of exploration – how to optimally select a deep clustering

This work was supported in part by the National Natural Science Foundation of China under Grant 61271405, 60702014 and 61401413, and in part by the Ph. D. Program Foundation of Ministry of Education of China under Grant 20120132110018. (Corresponding author: Junyu Dong.)

Yanhai Gan, Feng Gao, and Junyu Dong are with the Department of Computer Science and Technology, Ocean University of China, Qingdao, Shandong Province, 266100 China e-mail: (see http://ai-ouc.cn/people.html).

Xinghui Dong is with the Centre for Imaging Sciences, The University of Manchester, United Kingdom (e-mail:xinghui.dong@manchester.ac.uk) and Huiyu Zhou is with the School of Informatics, University of Leicester, United Kingdom (e-mail: hz143@leicester.ac.uk).

All the codes and data will be publicly available in https://github.com/gyh5421/unified_deep_clustering when the paper is published.

architecture and how to best design the optimization objective. Meanwhile, this objective encourages the development of solutions for dealing with general-purpose clustering tasks.

Further insight into the decision-making mechanism of human provides us the intuition that representation learning and clustering are collaborative tasks and must work together to produce the desirable results. In the proposed framework, we integrate representation learning and clustering into an individual pipeline for joint optimization instead of alternating between them as in previous methods [34]–[39]. To the best of our knowledge, this is the first attempt which essentially couples representation learning and clustering. The main contributions of this work are summarized as follows:

- A principled deep clustering objective is proposed to find a precise feature as the cue for cluster assignment.
- A necessary and sufficient condition is postulated to enable a solution for accomplishing the stated objective.
- A general-purpose deep clustering framework that couples representation learning and clustering is introduced.
- The state-of-the-art clustering results obtained using the proposed framework on several public datasets provide the other researchers a set of benchmarks.

The remaining of this paper is organized as follows. In Section II, we review the existing related work. The core ideas of the proposed framework are introduced in Section III. In Section IV, we present the experimental results. Finally, our conclusions and future work are discussed in Section V.

II. Related Work

Since standard clustering algorithms, e.g., K-means, usually encounter difficulties when dealing with high-dimensional and large-scale datasets [42]–[44], enormous kinds of two-stage methods are explored [45]–[48]. These methods first projected the data into a low-dimensional manifold, and then applied standard clustering algorithms to the embedded representations [49]–[53]. However, these methods normally require the domain-specific architectural deliberation in order to learn discriminative representations [54]–[56]. Although such deliberation is necessary for obtaining the competitive clustering performance, it is harmful to choosing a suitable architecture for a given task. It makes state-of-the-art deep clustering architectures become increasingly domain-specific [27]–[29], [57]. In addition, after being optimized in the first stage, the learned representation is fixed, so it cannot be further improved to obtain better performance in the clustering stage.

In recent years, some efforts have been made in the dynamic adaptation of representation and cluster assignment [30]–[36]. As an early work, deep embedded clustering (DEC) [33] improves the clustering using an unsupervised algorithm that alternates between two steps: 1) computing a soft assignment between the embedded points and the cluster centroids, 2) updating the deep mapping and refine the cluster centroids by learning from current high confidence assignments using an auxiliary target distribution. Analogously, Yang *et al.* [37] formulate the successive operations in a clustering algorithm as the steps in a recurrent process. The proposed framework (JULE) works by alternating between two steps. One step updates the cluster labels using the current representation while another step updates the representation parameters based on the current clustering results. Lately, Chang *et al.* [34] propose a deep adaptive clustering (DAC) algorithm that recasts the clustering problem into a binary pairwise-classification problem for judging whether or not pairs of images belong to the same cluster. To further utilize the category information, Wu *et al.* [58] develop a deep comprehensive correlation mining (DCCM) method that is trained by selecting highly-confident information in a progressive way.

Kamran *et al.* [38] introduce a multinomial logistic regression method on top of a multi-layer convolutional autoencoder for the joint learning of representation and cluster assignment. This method was referred to as deep embedded regularized clustering (DEPICT). Similarly, Zhou *et al.* [35] form an adversarial deep embedded clustering by combining adversarial auto-encoder and k-means together, where the representation parameters and clustering results are iteratively fine-tuned in a form of self-training after the network has been pretrained. To overcome the shortcomings of traditional spectral clustering, Shaham *et al.* [39] propose a deep learning based method (SpectralNet) that learns a map to embed input data points into the eigenspace of their associated graph Laplacian matrix and then performs the clustering operation. Based on the same inspiration, Zhang *et al.* [36] combine convolutional networks, self-expression module and spectral clustering module into a joint optimization framework ($S^2ConvSCN$), where the current clustering results are used to self-supervise the training of feature learning and self-expression module.

Although these methods have devoted huge efforts to the dynamical adaptation of representation and cluster assignment, and have produced promising results, they usually employ an alternative optimization strategy for representation learning and clustering. As a result, these methods usually prefer certain datasets and incorporate many exotic designs for learning discriminative features. In contrast, we define the objective of deep clustering in a principle way. Specifically, we are committed to find a precise feature as the cue for cluster assignment. To this end, we radically integrate the representation learning and clustering into an individual pipeline rather than alternating between the two tasks. As a result, we discard those exotic designs for the representation learning, and come up with a general-purpose deep clustering framework that can be generalized to common clustering tasks.

As our implementation involve the generative adversarial networks (GANs) [59], we make a brief introduction to it. Although Jürgen claims that the idea of adversarial learning was introduced by his work in 1990s [60]–[62], most commonly, it is recognized that GANs was proposed by Ian J. et al. in 2014 [59]. Compared to the blurry and low-resolution outcome from other generative models [26], [63], GANs-based methods [64]–[67] generate more realistic results with richer local details and of higher resolution. However, training GANs is well acknowledged to be delicate and unstable, with most current papers dedicating to heuristically finding stable architectures [64], [68], [69]. The problem is that JS distance, which is essentially optimized by GANs, is not a continuous loss function on model's parameters under that the model

manifold and the true distribution's support do not have a non-negligible intersection, which is rather common situation where the true distribution is supported by low dimensional manifolds. WGANs cure the main training problems of GANs by continuously estimating the Earth Mover (Wasserstein) distance, which makes it possible to learn a probability distribution over a low dimensional manifold by doing gradient descent [70]. Whereas, WGANs sometimes still generate poor samples or fail to converge due to the use of weight clipping to enforce a Lipschitz constraint on the critic. To rescue WGANs from the pathological performance, Gulrajani *et al.* [71] penalized the norm of gradient of the critic with respect to its input as an alternative to clipping weights, which was called WGAN-GP. WGAN-GP performs much better than standard WGANs and enables stable training of a wide variety of GANs architectures with almost no hyperparameter tuning.

III. The Unified Deep Clustering Framework

In this section, we first formulate the objective of deep clustering as finding a precise feature as the cue for cluster assignment. To fulfil this objective, we make an assumption. Then, we introduce the discipline for constructing the unified deep clustering framework based on the assumption. Finally, we describe the implementation details of the proposed deep clustering framework.

A. Objective Formulation

Unlike supervised learning, where the learning objective can be straightforward defined as the closeness between the ground-truth annotation and the prediction [72]–[74], how to define the objective of unsupervised learning is still an open problem worth exploring [27]–[29], [51], [53]–[56], [75], [76]. Generally, the objective is defined to discover the most discriminative features of data points [27]–[29]. However, in the deep clustering context, discriminative features may be task-specific. For example, digit recognition and handwriting recognition must require different cues of the input image samples. There is no doubt that the optimal features for digit recognition are not necessarily suitable for handwriting recognition. In the common sense, digit type is the most discriminative attribute of a hand-writing digit dataset. Nonetheless, these digits may be sampled from different writers. Indeed, if we would like to cluster these digits according to their chirography, none of the existing clustering algorithms perform well because their objectives are all dedicated to digit recognition.

To address this issue, we define the objective of deep clustering as finding a precise feature as the cue for cluster assignment. This objective encourages the establishment of a framework for handling general-purpose clustering tasks. These tasks may include the handwriting clustering problem mentioned, and all that is required is just the prior knowledge of the number of clusters. Please note that an exactly matched feature means that the possible values of the feature can establish a one-to-one relationship with the designated clusters. It accounts for a maximum predictability of the cluster assignment for a given sample. Mathematically, maximization of predictability is equivalent to minimization of the entropy of a distribution. Therefore, we can fulfil the defined objective by minimizing the distribution entropy of the cluster assignment.

However, when representation and cluster assignment are jointly optimized, direct entropy minimization is prone to getting stuck in the non-optimal local minima during training [38]. The reason is that practical samples usually contain a large number of variables that create many spurious correlations. To avoid the learning method from falling into these spurious correlations (e.g., division of the value space of a real-valued feature), we propose two constraints: 1) there is not an empty cluster, 2) the feature chosen as the cue for cluster assignment must be unique-valued in a cluster. Under these two constraints, the learning method can only select a precise (exactly matched) feature as the cue for cluster assignment. This is further formalized in the following assumption.

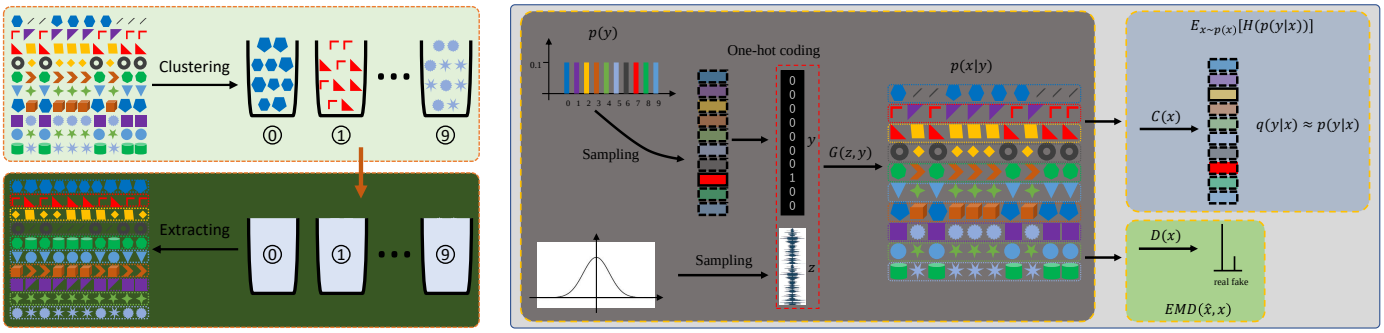
Assumption 1. *Minimization of the expectation of the distribution entropy of the cluster assignment is the necessary and sufficient condition for learning a precise feature as the cue for cluster assignment, under two constraints: 1) there is no empty cluster and 2) the feature chosen as the cue for cluster assignment must be unique-valued in a cluster.*

More explanation of this assumption can be found in the supplemental material. According to Assumption 1, we can endow a deep clustering algorithm with similar ability to humans by exploiting the most suitable features for different clustering tasks. In the next subsection, we will introduce the framework substantiating this assumption.

B. Framework Design

In Fig.1(a), 100 blocks of 10 colors, 20 shapes and 5 sizes are assigned to 10 baskets. As shown in the bottom, the assignment can be derived by fetching blocks from these baskets. If x and y denote a sample and a cluster respectively, then clustering is to assign x to y. Equivalently, this can be expressed as extracting x from y. Since the inverse formula naturally conforms to a generative paradigm from y to x, we design the deep clustering framework as shown in Fig.1(b), where G is an implicit generative model acting as the process of fetching samples from given cluster ids, D is a discriminator used to estimate the consistency between the generated samples and the real sample, C is a classifier that is used to implement the approximation of the real posterior distribution of the cluster assignment. In the following, we detail the motivation for each design choice of the framework.

The reason why we use G to reversely simulate the clustering process is because the generative paradigm helps to achieve the constraints of Assumption 1. In Fig.1(a), if shape is chosen as the cue for basket assignment, there will be at least one basket containing blocks of at least two different shapes (Pigeonhole principle). Consequently, if we use the identity of this basket as condition for generation, the produced samples will be of only one shape. This is due to the fact that there is no additional information for indicating what shapes exist in that basket. Finally, the generated samples drop into a subspace of the original sample space, since the cue used for cluster assignment is not an intrinsic feature, but the value space division of a certain feature.



(a) (b)

Fig. 1. Illustration of the overall framework. (a) is a simple sketch of the clustering process. In the first row, 100 blocks of 10 colors, 20 shapes and 5 sizes are clustered into 10 baskets. Equivalently, as depicted in the second row, the basket assignment can be derived by extracting blocks from these baskets. In the second row, the baskets are covered in gray because the assignment is unknown until all the blocks are extracted from these baskets. (b) is the flowchart of the proposed framework, where x denotes a sample and y denotes a cluster id. Besides, z is a random variable that obeys a multivariate normal distribution (with covariance matrix being an identity matrix), representing the features independent of y. In the framework, C is optimized to estimate the expectation of the distribution entropy of the cluster assignment, and D aims to estimate the Earth Mover distance (EMD) between the generated samples and the real samples. Afterwards, G is optimized to minimize the expectation of the distribution entropy of the cluster assignment and EMD simultaneously

In the framework, D is used to ensure the consistency between the generated samples and real samples. Under the pressure of D, the generative model G will be forced to put samples of the same feature value into one cluster. As a result, the second constraint of Assumption 1 (i.e., the feature chosen as the cue for cluster assignment must be unique-valued in one cluster) is satisfied. In this paper, we refer to features that are independent on each other and essential to composing the sample space as intrinsic features. In this sense, the generative formulation makes the deep model learn an intrinsic feature as the cue for cluster assignment.

For the first constraint of Assumption 1 (i.e., there is not an empty cluster), we implement it by assuming a uniform prior – denoted by $p(y)$ in Fig.1(b) – on the marginal distribution of cluster assignment. In this way, we virtually assume that samples are evenly distributed across clusters. In practice, the number of clusters is usually predetermined, but the marginal distribution of clusters is often unknown. Therefore, this is an over implementation of the first constraint, which limits the applicability of the deep clustering framework. Its specific impact will be further analyzed in the experimental part.

Since the two constraints of Assumption 1 have been satisfied, we are to deduce the entropy minimization objective required by the assumption. Although it is straightforward to perform an entropy minimization in a discriminative model, this is not the case for a generative model, because the distribution of the cluster assignment therein is posterior and always intractable. For this reason, we introduce a variational algorithm for indirect optimization of the distribution entropy of the cluster assignment. Specifically, we first calculate an approximation (output of C) of the real posterior distribution, and then induce an upper bound of the expectation of the distribution entropy. Afterwards, the expectation of the distribution entropy of the cluster assignment can be consistently minimized as we continue to lower the upper bound.

As illustrated in Fig.1(b), the conditional distribution implied by G is denoted as $p(x|y)$, the posterior distribution of the cluster assignment is denoted as $p(y|x)$, and the approximation of $p(y|x)$ is denoted as $q(y|x)$. Afterward, the expectation of the cross-entropy between the real posterior distribution and the approximation can be calculated as follows:

$$\begin{aligned} & E_{x\sim p(x)}[H(p(y|x), q(y|x))] \\ = & -\int p(x)\int p(y|x)\log q(y|x)dydx \\ = & -\int\int p(x)p(y|x)\log q(y|x)dydx \\ = & -\int\int p(x,y)\log q(y|x)dydx \\ = & E_{x,y\sim p(x,y)}[-\log q(y|x)]. \end{aligned} \tag{1}$$

Eq.1 pronounces that the expectation of the cross-entropy is equal to the expectation of the negative log-likelihood of the approximation on the joint distribution of x and y.

In addition, the expectation of the cross-entropy can be expressed as an addition of two terms:

$$\begin{aligned} & E_{x\sim p(x)}[H(p(y|x), q(y|x))] \\ = & E_{x\sim p(x)}[-\int p(y|x)\log q(y|x)dy] \\ = & E_{x\sim p(x)}[-\int p(y|x)\log(\frac{q(y|x)}{p(y|x)}p(y|x))dy] \\ = & E_{x\sim p(x)}[-\int p(y|x)(\log\frac{q(y|x)}{p(y|x)}+\log p(y|x))dy] \\ = & E_{x\sim p(x)}[\int p(y|x)\log\frac{p(y|x)}{q(y|x)}dy - \int p(y|x)\log p(y|x)dy] \\ = & E_{x\sim p(x)}[KL(p(y|x), q(y|x))] + E_{x\sim p(x)}[H(p(y|x))], \end{aligned} \tag{2}$$

where $KL(p(y|x), q(y|x))$ represents the Kullback-Leibler divergence between $p(y|x)$ and $q(y|x)$, and $H(p(y|x))$ denotes the distribution entropy of $p(y|x)$. Since $KL(p(y|x), q(y|x))$ is definitely positive, we have the following inequality:

$$E_{x\sim p(x)}[H(p(y|x))] \leq E_{x\sim p(x)}[H(p(y|x), q(y|x))]. \tag{3}$$

Eq.3 indicates that the expectation of the cross-entropy is an upper bound of the expectation of the distribution entropy of the cluster assignment, and the upper bound becomes tight

if and only if $KL(p(y|x), q(y|x))$ gets close to zero, which means that $q(y|x)$ is approaching $p(y|x)$ almost everywhere. Therefore, we will consistently minimize the expectation of the distribution entropy of the cluster assignment if we keep the approximation $q(y|x)$ accurate and continue to reduce the cross-entropy $E_{x\sim p(x)}[H(p(y|x), q(y|x))]$.

However, the solution illustrated by Eq.1 for the expectation of the cross-entropy involves an expectation on the joint distribution $p(x, y)$ that is implicit. Since direct calculation is not practical, we solve for the expectation by utilizing a Monte-Carlo algorithm. As we encode cluster ids in the one-hot fashion, according to the strong law of large numbers, the following equation can be obtained:

$$\begin{aligned} & E_{x,y\sim p(x,y)}[-\log q(y|x)] \\ = & \lim_{n\to\infty} -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} y_{ij}\log q(y_j|x_i), \end{aligned} \tag{4}$$

where y_{ij} represents the jth entry of the one-hot coding vector of the cluster id generating x_i, k is the number of clusters, and n denotes the number of samples. Eq.4 enables a Monte-Carlo solution for the expectation by first sampling y from the prior distribution and then sampling x from the likelihood $p(x|y)$ that is implicitly modeled by the generator G. The two-stage sampling process is equivalent to sampling (x, y) from their joint distribution $p(x, y)$. In practice, the Monte-Carlo approximation becomes more and more accurate with the value of n getting larger. Here we suppose that n is large enough, and the solution is basically accurate.

In the training stage, D and C are optimized first before each optimization of G. It is known that the discriminator D is dedicated to estimating a distance between the generated samples and real samples [59], [70], [71], [77]. Because C is used to approximate the real posterior distribution of the cluster assignment of a given sample, the Kullback-Leibler divergence between $p(y|x)$ and $q(y|x)$ will approach zero after C is optimized. This means that $q(y|x)$ asymptotically equals to $p(y|x)$ almost everywhere and the upper bound implied by Eq.3 hence becomes tight. In turn, the generative model G is optimized for two tasks: 1) minimization of the distance between the generated samples and real samples, 2) reducing the up bounder of the expectation of the distribution entropy of the cluster assignment. Consequently, the generative model G learns a mapping between the cluster ids and the samples, where the minimized distribution entropy of the cluster assignment ensures a one-to-one correspondence between the clusters and the discrete values of the cue feature.

In this subsection, we assume that samples and labels are continuous and perform derivation by calculus, whereas it should be noted that the conclusions still hold for discrete variables. In that case, the integration becomes summation and the probability densities become discrete probability masses.

C. Implementation

First, the uniform distribution $y \sim \mathcal{U}^{int}[1, k]$, which denotes a discrete distribution with the probability mass uniformly distributed on integers in the closed interval $[1, k]$, is employed as the marginal distribution of the cluster assignment to satisfy the first constraint of Assumption 1. Second, as WGAN-GP [71] achieves much better performance than other generative models [64], [68]–[70], we employ it as the backbone to realize the consistency between the generated samples and real samples. The consistency condition satisfies the second constraint of Assumption 1. Now, we formally present the loss functions for each component of the framework.

First of all, the loss function of the discriminator D is the same defined as that in WGAN-GP:

$$\begin{aligned} L_D = & E_{x\sim p_g(x)}[D(x)] - E_{x\sim p_r(x)}[D(x)] \\ & +\lambda E_{\hat{x}\sim p_{\hat{x}}(\hat{x})}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2], \end{aligned} \tag{5}$$

where $p_{\hat{x}}()$ represents the uniform sampling function which works along the straight lines between pairs of points sampled from both the data distribution p_r and the generator distribution p_g, $D(x)$ denotes the output of the discriminator when x is given, and λ is a hyperparameter for the gradient penalty term. After the discriminator D is sufficiently optimized, L_D will be approximately equal to the Earth Mover distance [71] between the generated samples and the real samples.

Second, because the cluster assignment of a given sample obeys a categorical distribution, the estimator C for the posterior distribution of cluster assignment adopts the conventional cross entropy as the loss function:

$$L_C = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} y_{ij}\log q(y_j|x_i), \tag{6}$$

where y_{ij} denotes the jth entry of the one-hot coding of the cluster id used as the input in G to generate the ith sample, and $q(y_j|x_i)$ stands for the jth output of the classifier when the ith sample is fed as input. The cluster ids fed into the generator are viewed as ground-truth when the cross-entropy loss is calculated, and we refer to them as fake labels in the following. It should be noted that the fake labels are autonomously generated rather than being annotated by humans.

Finally, the loss function of the generator G is defined as an addition of two terms:

$$L_G = L_{reality} + \eta L_{entropy}, \tag{7}$$

where $L_{reality}$ is the reality term, and $L_{entropy}$ is the entropy minimization term. According to the conventional practice of GANs [71], we make $L_{reality} = -E_{x\sim p_g(x)}[D(x)]$. Since L_C is defined as the cross entropy by Eq.6, we can readily set $L_{entropy} = L_C$ by referring to Eq.4. In addition, $\eta > 0$ is a trade-off parameter between the reality term and the entropy minimization term. During training, η is exponentially increased in a staircase function:

$$\eta \leftarrow \eta\gamma^{\lfloor t/\tau \rfloor}, \tag{8}$$

where t represents the current training step, and τ denotes the number of steps for every increase. In this manner, the generator will tend to focus on the entropy minimization objective in the later training stage, in which the quality of the generated samples has been significantly improved.

The training dynamics of these three components are further formalized in Algorithm 1, which also declares the configuration of the hyperparameters used in the experiment. After the optimization is completed, the estimator for the posterior distribution of the cluster assignment becomes accurate and is exploited for efficient clustering in the inference stage.

IV. Experiments

In this section, we conduct several experiments to verify the proposed method. Specifically, we experiment on a synthetic dataset, MNIST, Fashion-MNIST, Artifact-MNIST, ORL, USPS, Cifar-10 and ImageNet-10 to examine a practical and theoretically grounded direction towards solving the deep clustering problems. As popular measures in the literature, Clustering Accuracy (ACC), Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) are employed for evaluation. The value range of ACC and NMI is [0,1], and the value range of ARI is [-1,1]. It should be noted that the effectiveness of the framework strongly relies on the capability of the generative model to produce realistic samples. Therefore, the reported results can consistently get improved as the generative model evolves, which is now prospective [78]–[80].

A. Networks

Because our main purpose is to verify the utility of integrating representation learning and clustering into a unified framework, we do not carry out exhaustive architecture and hyperparameter search in all experiments, and the architectural choice and experimental configuration are similar to [71]. In particular, the generator and discriminator inherit the network structure in [71]. The classifier shares a similar structure with the discriminator, but the classifier has a different output layer to produce categorical probability masses. The training dynamics between these three components have been outlined in Algorithm 1, and will be explained in detail below.

The networks embodying our framework are illustrated in Fig.2, where all the convolutional or deconvolutional layers adopt a 5x5 kernel size and a 2x2 stride. All the experiments are conducted with this architecture, and only a few modifications are made to adapt to different datasets, except for experiments on the synthetic dataset, where three fully-connected networks are employed. We have also tried to combine the discriminator and classifier into a single network with two output heads for multi-task learning. However, this strategy causes performance degradation on some data sets (i.e., according to median statistics of ten runs, at least 10% performance degradation on MNIST and Fashion-MNIST, and completely crashed results on Artifact-MNIST).

It is likely that even though both the discriminator and the classifier learn discriminative features of the samples, they focus on different aspects. The discriminator looks for the difference between the generated samples and real samples. As the training progresses, this difference gradually changes. On the contrary, the classifier aims to discover the accurate features for clustering. Sometimes, these two prompts may be quite different, especially in the final stage of training, at this stage, the most effective features for distinguishing the generated samples from real samples may be invalid for the on-hand clustering task. In addition, the discriminator is constrained by the gradient penalty to realize a Lipschitz function [70], [71], which may impair the learning of clustering hints, whereas the separate classifier does not have to be Lipschitz.

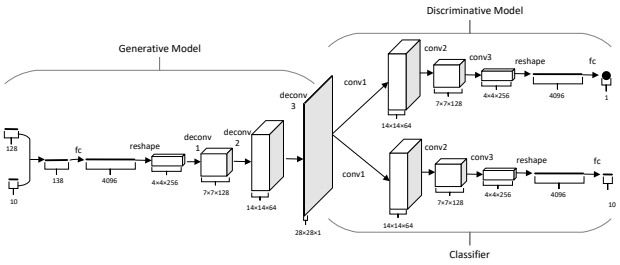


Fig. 2. Architecture used in our experiments. This architecture is used across experiments on MNIST, Fashion-MNIST, Artifact-MNIST, ORL, USPS, Cifar-10 and ImageNet-10. For MNIST, Fashion-MNIST and Artifact-MNIST, the size of the images is 28x28, we drop one pixel horizontally and vertically after the first deconvolutional layer. In the experiments on ORL and Cifar-10, the size of the feature maps outputted by the first deconvolutional layer should be 8x8. For Cifar-10, the output of the generator should be 32x32x3, which is formalized by a deconvolutional layer with 3 kernels rather than 1 for grayscale images. As USPS variants consist of images of 16x16, we adopt a 1x1 stride in the last deconvolutional layer of the generator and the first convolutional layer of the discriminator and classifier. For ImageNet-10, we append a deconvolutional layer to the generator and a convolutional layer to the discriminator and classifier to process 64x64 images. In this figure, we just write the output of the generator as 28x28x1 for compact. To disambiguate, we make it clear here for readers

B. Experiments on synthetic dataset

In order to study the effectiveness and characteristics of the proposed framework, we conduct experiments on a simple dataset consisting of eight isotropic Gaussian blobs of data. The centers of these Gaussian blobs are (1.414,0), (-1.414,0), (0,1.414), (0,-1.414), (1,1), (-1,1), (-1,-1), (1,-1), and the standard deviation of all the blobs is 0.014. The Gaussian mixture where samples come from is figured in the supplemental material. In the experiment, we view samples coming from the same Gaussian blob as in one cluster. The training and evaluating samples in the experiment are all randomly sampled from the Gaussian mixture. Therefore, there are indeed infinite training and evaluation samples.

In this experiment, three fully connected networks are adopted to embody the framework. The network structure of the generator acts like $x - 512 - 512 - 512 - 2$, the network structure of the discriminator acts like $2 - 512 - 512 - 512 - 1$, and the network structure of the classifier acts like $2 - 512 - 512 - 512 - 8$. Therein x denotes the number of inputting variables to the generator, which varies from 8 to 16 in the experiments. As one-hot coding is applied, one cluster id corresponds to 8 inputting variables. The other inputting variables are all noise variables. We use $relu$ as activation function in each hidden layer of these networks, and use non-activation function in the output layer of the generator and discriminator. The output of the classifier is activated by $softmax$ to realize a normalized probability distribution.

A total of six experiments are performed on the synthetic dataset, where the cluster id sampled from the uniform

Algorithm 1 We use default values of $\lambda = 100$, $\eta = 10$, $N_{critic} = 5$, $N_{class} = 4$, $N = 900000$, $\alpha = 0.0001$, $\beta_1 = 0.5$, $\beta_2 = 0.9$, $\tau = 30000$, $\gamma = 1.2$.

Input: The gradient penalty coefficient λ, the trade-off parameter η, the number of critic iterations N_{critic} and the number of classifier iterations N_{class}, the number of generator iterations N, the batch size n, Adam hyperparameters α, β_1, β_2
Input: Initial critic parameters θ_d, initial generator parameters θ_g, initial classifier parameters θ_c, the number of clusters k
Output: Classifier C

while $t \leq N$ **and** θ_g has not converged **do**
  repeat
    for $i = 1, \ldots, n$ **do**
      Sample a real data $\mathbf{x} \sim \mathbb{P}_r$, a cluster id $y \sim \mathcal{U}^{int}[1, k]$, a random noise vector $\mathbf{z} \sim \mathcal{N}(0, 1)$, a random number $\epsilon \sim \mathcal{U}[0, 1]$
      $\mathbf{y} \leftarrow$ one-hot coding of y
      $\tilde{\mathbf{x}} \leftarrow G_{\theta_g}(\mathbf{y}, \mathbf{z})$
      $\hat{\mathbf{x}} \leftarrow \epsilon\mathbf{x} + (1 - \epsilon)\tilde{\mathbf{x}}$
      $L^{(i)} \leftarrow D_{\theta_d}(\tilde{\mathbf{x}}) - D_{\theta_d}(\mathbf{x}) + \lambda(\|\nabla_{\hat{\mathbf{x}}} D_{\theta_d}(\hat{\mathbf{x}})\|_2 - 1)^2$
    end for
    $\theta_d \leftarrow Adam(\nabla_{\theta_d} \frac{1}{n} \Sigma_{i=1}^{n} L^{(i)}, \theta_d, \alpha, \beta_1, \beta_2)$
  until Reach the maximal iteration N_{critic}
  repeat
    for $i = 1, \ldots, n$ **do**
      Sample a cluster id $y \sim \mathcal{U}^{int}[1, k]$, a random noise vector $\mathbf{z} \sim \mathcal{N}(0, 1)$
      $\mathbf{y} \leftarrow$ one-hot coding of y
      $\hat{\mathbf{y}} \leftarrow C_{\theta_c}(G_{\theta_g}(\mathbf{y}, \mathbf{z}))$
      $L^{(i)} \leftarrow -\Sigma_{j=1}^{k} \mathbf{y}_j \ln \hat{\mathbf{y}}_j$
    end for
    $\theta_c \leftarrow Adam(\nabla_{\theta_c} \frac{1}{n} \Sigma_{i=1}^{n} L^{(i)}, \theta_c, \alpha, \beta_1, \beta_2)$
  until Reach the maximal iteration N_{class}
  Sample a batch of cluster ids $\{y^{(i)}\}_{i=1}^{n} \sim \mathcal{U}^{int}[1, k]$, a batch of random noise vectors $\{\mathbf{z}^{(i)}\}_{i=1}^{n} \sim \mathcal{N}(0, 1)$
  for all $y^{(i)}$ such that $i \in [1, n]$ **do**
    $\mathbf{y}^{(i)} \leftarrow$ one-hot coding of $y^{(i)}$
    $\hat{\mathbf{y}}^{(i)} \leftarrow C_{\theta_c}(G_{\theta_g}(\mathbf{y}^{(i)}, \mathbf{z}^{(i)}))$
  end for
  $L \leftarrow -\frac{1}{n} \Sigma_{i=1}^{n} (D_{\theta_d}(G_{\theta_g}(\mathbf{y}^{(i)}, \mathbf{z}^{(i)})) + \eta \Sigma_{j=1}^{k} \mathbf{y}_j^{(i)} \ln \hat{\mathbf{y}}_j^{(i)})$
  $\theta_g \leftarrow Adam(\nabla_{\theta_g} L, \theta_g, \alpha, \beta_1, \beta_2)$
  $t \leftarrow t + 1$
  $\eta \leftarrow \eta\gamma^{\lfloor t/\tau \rfloor}$
end while

categorical distribution on [0, 7] and different numbers (0, 1, 2, 3, 4, 8) of noise variables are used together as the input to the generator. Experimental results illustrate that when feeding 0, 1, 2 or 4 noise variables into the generator, the framework can exploit the centers of the gaussian blobs as the cue for cluster assignment. This demonstrate that, under appropriate experimental configuration, the framework can exploit the dominant feature of samples as the cue and give fascinating clustering results accordingly. However, when 3 or 8 noise variables are fed in, the generator begins to generate samples completely deviating from the true distribution, and the classifier falls into severe overfitting.

Because there are actually 3 intrinsic features (the centers of the gaussian blobs and the biases on x-axis and y-axis) that control the positions of samples in the plane, the experimental results declares that when the number of variables used as input deviates from the true number of intrinsic features, the performance of the framework becomes unstable. However, it's worth noting that, when the number of inputting variables decreases from the actual number of intrinsic features, the performance of the framework does not decrease as sharply as the number of inputting variables increases from them on. Thus, we preferentially use fewer inputting variables in the framework if the true number of intrinsic features is unknown, which is also desirable in practice for efficiency reasons. Furthermore, we can judge whether the number of inputting variables exceeds the true number of intrinsic features by plotting the learning curves. When too many variables are used as input, the Earth-Mover distance and the evaluated clustering accuracy will gradually diverge in the later stage of the training process. More details about the experimental results can be found in the supplemental material.

C. Experiments on MNIST and Fashion-MNIST

In academia, MNIST is often the first dataset researchers try. Members of the AI/ML/Data Science community love this dataset and use it as a benchmark to validate their algorithms.

It is widely believed that if an algorithm doesn't work on MNIST, it won't work at all. However, the reality is that even if an algorithm works well with MNIST, it doesn't necessarily work well with others. As MNIST is too easy and overused, some researchers think MNIST can not represent modern CV tasks and call for people to move away from MNIST. Fashion-MNIST is intended to serve as a direct drop-in replacement for the original MNIST dataset for benchmarking machine learning algorithms. It shares the same image size and structure of training and testing splits as MNIST [81].

On these two datasets, we use the original training and testing splits to train and evaluate our framework and all the other comparison methods, where label information is not used during the training phase. In order to ensure the fairness of the comparison, all the comparison methods use the best hyperparameters reported in the literature to retrain on the datasets. Our experiments on MNIST and Fashion-MNIST adopt the same configuration, including architecture selection and hyperparameter settings. On each dataset, we run each method ten times with the same configuration, and report the minimum, maximum, and median statistics of the three metrics. The results are summarized in Table I and Table II, where the results marked with * are reported in the literature.

We rarely find clear statements in the relevant papers that the reported results are median. Most of the time, researchers report the best results found in experiments. However, we consciously treat all reported results as medians to allow a sound analysis of the improvement of our method. Table I and Table II demonstrate that our method obtains state-of-the-art clustering results on the MNIST and Fashion-MNIST datasets. On MNIST, from the median statistics of the three metrics, our method is competitive with state-of-the-art methods. From the maximum statistics, our method is superior to all the other methods. On Fashion-MNIST, our method outperforms all the other comparison methods with a surprising advantage. The only flaw of our method is that the minimum statistics of the performance on MNIST is relatively poor compared to state-of-the-art due to the unstable performance that has been quantitatively reflected in Fig.4a as the considerable standard deviation. More detailed comparison of the performance of ten runs can be found in the supplemental material.

Fig.3 illustrates the learning dynamics on MNIST and Fashion-MNIST. In Fig.3(a)(e), the cross-entropy loss decreases quickly as the training begins. Correspondingly, the fake classifying accuracy obtained by treating the fake labels as ground truth is rapidly improved. Later in the training process, the cross-entropy loss is kept small, and the fake classifying accuracy remains close to 1. According to the derivation in section III-B, since the classifier is fully optimized, the objective is actually to guide the generator to minimize the expectation of the distribution entropy of the cluster assignment. In the experiment, the expectation of the distribution entropy quickly reaches its optimum at the beginning of the training process, so the generator is actually optimized to produce realistic samples while keeping the distribution entropy to a minimum. In this case, as the quality of the generated samples improves (Earth Mover distance converges as in Fig.3(b)(f)), the evaluated clustering accuracy will continue to increase (as shown in Fig.3(c)(g)). Finally, when the generator produces high-quality samples, the framework obtains encouraging clustering results. In fact, the samples generated in Fig.3(d) and Fig.3(h) show that the generated samples from the same cluster are similar in perception – basically the same in digit or apparel type. This proves that the framework has discovered the digit or apparel type in the image as a clue for cluster assignment.

In the experiments, we initialize η to 10 and then multiply it by 1.2 every 30,000 iterations. A total of 900,000 iterations of this optimization are performed. It should be noted that in Fig.3(b) and Fig.3(f), the plots of the trade-off parameter are scaled by 10. In addition, the hyperparameter λ is set to 100.

D. Learning a precise feature as the cue for clustering

We have argued in section III that our objective is to lean an exactly matched characteristic as the cue for cluster assignment, which differs the proposed framework from previous methods. However, the above experiments cannot provide proof of this statement, because the clustering tasks are still focused on finding the most dominant feature (centers of the gaussian blobs in synthetic dataset, digit and apparel types in MNIST and Fashion-MNIST) of the data points, and performing grouping accordingly. In this section, we plan to empirically demonstrate the ability of the proposed framework to learn a precise feature as the cue for cluster assignment.

1) Experiments on Artifact-MNIST: To this end, we make an artifact version of the original MNIST dataset and call it Artifact-MNIST. Specifically, we randomly set the first pixel of each image in the MNIST dataset to 0.1, 0.5, and 0.9 with the same probability. Since the artifact is so subtle, it must not be the dominant feature of the samples, and can be regarded as an analogy of chirography. In addition, because of the sample-level and independent (of other features) nature, the artifact is one of the intrinsic characteristics of these samples. In this experiment, we want to investigate whether our method can find such delicate artifacts as clues for cluster assignment when three categories are specified for the clustering task.

We run our method and all the comparison methods ten times on Artifact-MNIST. The experimental results are listed in Table III, where the zero values of NMI and ARI represent meaningless (totally random) cluster assignment. A more detailed comparison of ten runs is given in the supplementary material. Although our method does not always provide ideal results, all the other clustering algorithms can not accomplish this task. The reason for the failure cases of our method may be that the dataset contains other intrinsic features that are also ternary. The intelligent agent tries to capture an intrinsic feature to perfectly solve this three-category clustering problem, but it will arbitrarily select one from multiple candidates, and does not always encounter the desired one. However, the persistent success results (over 80% of the time) indicate that our framework can indeed solve such general-purpose clustering problems. In practice, a small evaluation set is needed to check whether the intelligent agent has captured the desirable feature as the cue for cluster assignment.

2) Experiments on the ORL dataset: ORL [85] is a widely used dataset in the context of face recognition [36], [86].

TABLE I
PERFORMANCE COMPARISON ON MNIST. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD**. * INDICATES THAT THE RESULT IS REPORTED IN LITERATURE

Method	ACC			NMI			ARI		
	Min	*Max*	*Med*	*Min*	*Max*	*Med*	*Min*	*Max*	*Med*
NMF [82]	-	-	0.545*	-	-	0.608*	-	-	0.430*
K-means [13]	0.534	0.571	0.563	0.479	0.521	0.499	0.347	0.374	0.352
SC [83]	-	-	0.696*	-	-	0.663*	-	-	0.521*
AC [23]	-	-	0.695*	-	-	0.609*	-	-	0.481*
DeCNN [84]	-	-	0.818*	-	-	0.758*	-	-	0.669*
GAN [64]	-	-	0.828*	-	-	0.764*	-	-	0.736*
DDC [48]	-	-	0.965*	-	-	0.916*	-	-	-
DDC-DA [48]	-	-	**0.970***	-	-	0.927*	-	-	-
DAC [34]	0.745	0.813	0.804	0.782	0.836	0.820	0.678	0.756	0.728
DEC [33]	0.862	0.864	0.864	0.833	0.835	0.835	0.797	0.801	0.800
JULE [37]	0.948	0.964	0.960	0.901	0.926	0.912	0.913	0.927	0.922
SpectralNet [39]	**0.967**	0.971	0.969	0.920	0.924	0.921	**0.931**	0.934	**0.933**
DCCM [58]	0.641	0.921	0.780	0.651	0.905	0.785	0.499	0.815	0.650
Ours	0.915	**0.984**	0.958	**0.922**	**0.978**	**0.944**	0.855	**0.951**	0.912

TABLE II
PERFORMANCE COMPARISON ON FASHION-MNIST. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD**. THE IMPROVEMENT OF OUR METHOD HAS BUILT A LARGE MARGIN ON THREE METRICS REGARDLESS OF THE USED STATISTICS

Method	ACC			NMI			ARI		
	Min	*Max*	*Med*	*Min*	*Max*	*Med*	*Min*	*Max*	*Med*
DDC [48]	-	-	0.619*	-	-	0.682*	-	-	-
DDC-DA [48]	-	-	0.609*	-	-	0.661*	-	-	-
K-means [13]	0.254	0.354	0.331	0.172	0.307	0.256	0.181	0.271	0.249
DEC [33]	0.469	0.478	0.477	0.492	0.504	0.501	0.320	0.331	0.330
SpectralNet [39]	0.488	0.523	0.505	0.519	0.529	0.523	0.329	0.347	0.337
JULE [37]	0.423	0.505	0.486	0.594	0.652	0.639	0.342	0.421	0.390
DAC [34]	0.435	0.591	0.531	0.487	0.584	0.552	0.371	0.459	0.414
DCCM [58]	0.406	0.593	0.544	0.315	0.515	0.449	0.301	0.416	0.375
Ours	**0.685**	**0.754**	**0.721**	**0.689**	**0.749**	**0.719**	**0.524**	**0.589**	**0.555**

TABLE III
PERFORMANCE COMPARISON ON ARTIFACT-MNIST. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD**. THE ZERO VALUES OF NMI AND ARI INDICATE COMPLETELY RANDOM CLUSTER ASSIGNMENT

Method	ACC			NMI			ARI		
	Min	*Max*	*Med*	*Min*	*Max*	*Med*	*Min*	*Max*	*Med*
K-means [13]	0.347	0.352	0.350	0.000	0.000	0.000	0.000	0.000	0.000
DEC [33]	0.335	0.337	0.336	0.000	0.000	0.000	0.000	0.000	0.000
SpectralNet [39]	0.338	0.342	0.340	0.000	0.000	0.000	0.000	0.000	0.000
JULE [37]	0.345	0.362	0.359	0.000	0.000	0.000	0.000	0.000	0.000
DAC [34]	0.348	0.379	0.368	0.000	0.000	0.000	0.000	0.000	0.000
DCCM [58]	0.322	0.349	0.332	0.000	0.000	0.000	0.000	0.000	0.000
Ours	**0.467**	**1.000**	**1.000**	**0.051**	**1.000**	**1.000**	**0.036**	**1.000**	**1.000**

TABLE IV
PERFORMANCE OF VARIOUS CLUSTERING METHODS ON ORL. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD**. IN EACH CELL, THE LEFT VALUE IS EXAMINED BY VIEWING IDENTITY AS GROUND TRUTH, AND THE RIGHT VALUE IS EXAMINED BY VIEWING GENDER AS GROUND TRUTH

Method	ACC						NMI						ARI					
	Min		*Max*		*Med*		*Min*		*Max*		*Med*		*Min*		*Max*		*Med*	
DAC [34]	0.098	0.265	0.143	0.690	0.133	0.404	0.286	0.027	0.379	0.082	0.344	0.049	0.009	-0.020	0.052	0.150	0.040	0.008
K-means [13]	0.715	0.900	0.800	0.900	0.751	0.900	0.842	0.018	0.885	0.020	0.864	0.019	0.570	-0.009	0.673	-0.005	0.621	-0.007
DEC [33]	0.025	0.503	0.395	0.900	0.191	0.569	0.000	0.000	0.619	0.058	0.422	0.006	0.000	-0.039	0.203	0.036	0.073	0.000
SpectralNet [39]	0.025	0.888	0.448	0.892	0.025	0.890	0.000	0.007	0.670	0.009	0.000	0.008	0.000	-0.020	0.293	-0.012	0.000	-0.016
JULE [37]	0.560	0.900	0.625	0.900	0.597	0.900	0.758	0.006	0.805	0.093	0.786	0.030	0.371	-0.084	0.480	0.099	0.428	-0.025
$S^2ConvSCN - l_2$ [36]	-	-	-	-	0.888*	-	-	-	-	-	-	-	-	-	-	-	-	-
$S^2ConvSCN - l_1$ [36]	-	-	-	-	0.895*	-	-	-	-	-	-	-	-	-	-	-	-	-
DCCM [58]	0.735	0.502	0.825	0.610	0.775	0.540	**0.883**	0.000	**0.921**	0.163	**0.902**	0.025	0.651	-0.016	0.764	0.047	0.705	0.002
Ours	**0.893**	**0.973**	**0.922**	**0.980**	**0.910**	**0.978**	0.875	**0.615**	0.902	**0.840**	0.893	**0.688**	**0.875**	**0.615**	**0.902**	**0.840**	**0.893**	**0.688**

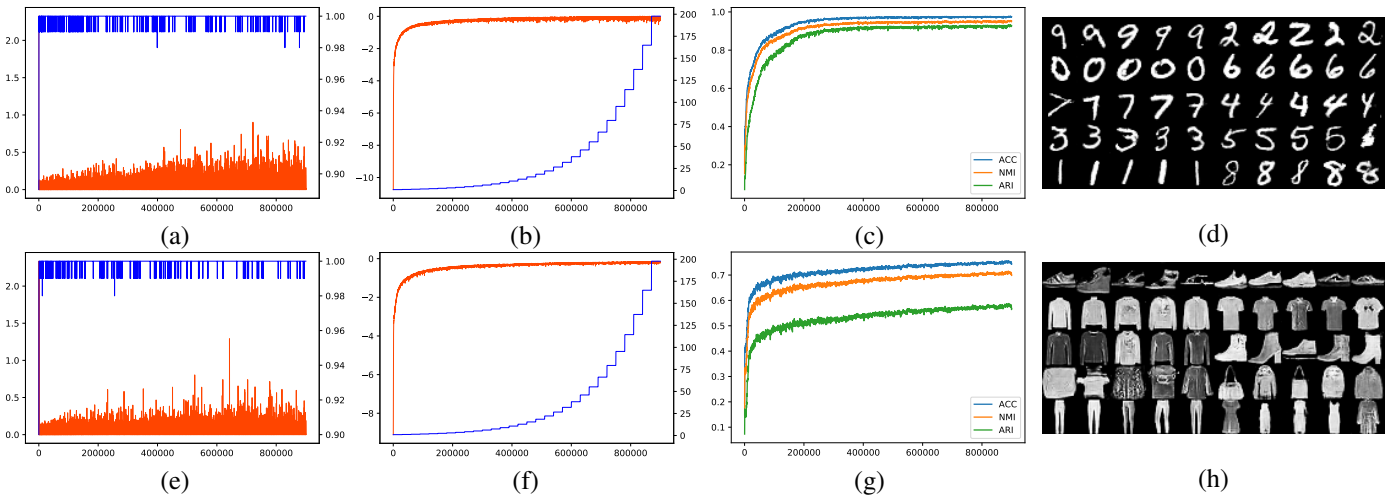


Fig. 3. Learning curves on MNIST and Fashion-MNIST. The subfigures in the first line depicts the learning dynamics on MNIST, and the subfigures in the second line depicts the learning dynamics on Fashion-MNIST. (a)(e) illustrate the changes of the expectation of distribution entropy of the cluster assignment (orange, ticks on the left axis), and the fake classifying accuracy (blue, ticks on the right axis) during training. (b)(f) describe the Earth Mover distance evaluated on the test data (orange, ticks on the left axis), and the trade-off parameter η (blue, ticks on the right axis) at each iteration. (c)(g) display the evolution of the ACC, NMI and ARI metrics evaluated on the test data at each iteration. Finally, the samples generated at the last optimization iteration are given in (d)(h). The generated samples are arranged according to the cluster id. Specifically, the first five or last five samples of each line in (d)(h) are generated from the same cluster id. A total of 900,000 iterations of this optimization are performed

Images in the dataset are taken under varying illumination conditions, facial expressions, and facial occlusions (with or without glasses). The ORL dataset consists of 400 images, 10 each of 40 different subjects. There are 4 female and 36 male subjects in the dataset. The ORL dataset can be used for verifying the capability of finding precise feature as the cue for cluster assignment. In the dataset, clustering can be performed according to identities or genders of the facial images.

In this experiment, each facial image is resized to 32x32 pixels. Since there are only 400 samples in the dataset, we optimize the generator for 30,000 iterations. Correspondingly, we initialize η to 10 and then multiply it by 1.2 every 1,000 iterations (reduced in proportion to the number of optimization iterations). All the comparison methods are also trained on the dataset using the default configuration on MNIST. We run each method for ten times, and the statistics are listed in Table IV. In each cell of Table IV, the left value is obtained by performing clustering according to identity, and the right value is obtained by performing clustering according to gender. When we want the framework to perform clustering according to gender, we provide the framework not only the categorical information, but also the distribution of the clusters, since we know that there are only four females. In spite of this, when we regard gender as the ground truth, the clustering performance is still less satisfying, because there is a hard subject in the dataset (the 12th subject), for which it is difficult for even humans to judge his gender. If we remove this subject from the dataset, we can get a higher accuracy.

It should be noted that clustering according to gender is a binary clustering task with extremely unbalanced distribution, so completely random assignment means an ACC around 0.5, and putting all samples into one cluster corresponds to an ACC of 0.9. Therefore, when performing clustering according to gender, the indicators NMI and ARI are of more reference value than ACC. With reference to Table IV, it can be found that, except our method, all methods fail to cluster by gender. This experiment on a real-world face recognition dataset demonstrates that our method does possess the ability to find precise cue (identity or gender) of samples according to the specified number of categories for cluster assignment.

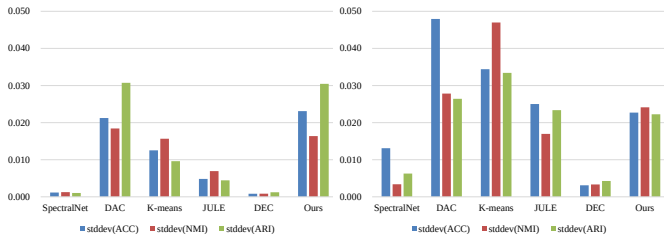
E. Stability analysis

Stability is one of many indicators to evaluate the quality of an algorithm. In order to compare the stability of our method and other methods, we calculated the standard deviations of three metrics in ten runs for all algorithms, and the results obtained on MNIST and Fashion-MNIST are shown in Fig.4. It can be seen that our method performs more unstable than SpectralNet [39] and DEC [33].

We have tried to provide fixed seeds to the random number generators of Python and Tensorflow (we use Tensorflow for implementation). However, the resulted clustering performance still fluctuate. We attribute the instability of the clustering performance to the random behavior of the cuDNN library employed by Tensorflow. In any case, our method is too sensitive to the initialization of the network parameters. This becomes the main flaw of our method. We hope to make the method more robust to random initialization in the future.

F. Dealing with non-uniform distributions

In section III-B, we adopt a uniform prior for the marginal distribution of the clusters to realize the first constraint of Assumption 1, which limits the application of our method to problems where samples are evenly distributed across clusters. To concretely see how the distribution of clusters affects the clustering performance, we construct five variants of the USPS dataset and conduct experiments on these variants. USPS is



(a) (b)

Fig. 4. Comparison of stability between different methods. (a) and (b) illustrate the results evaluated on MNIST and Fashion-MNIST respectively, where the standard deviation of the performance (ACC, NMI, and ARI) in ten runs is calculated and displayed

a handwritten digit dataset, which consists of 7,291 training images and 2,007 test images. The images in USPS are 16x6 grayscale pixels. The samples in USPS are unevenly distributed across 10 classes, with the largest class owning 1,553 samples, and the smallest class owning 708 samples. In this experiment, we reproduce the experimental configuration of section IV-C, including architecture and hyperparameters.

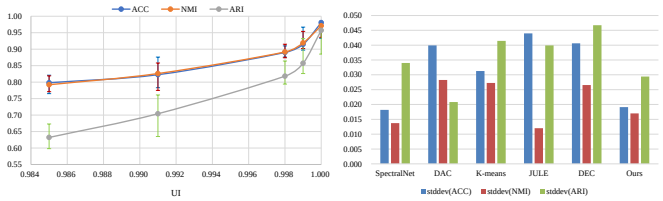
To quantitatively evaluate how the cluster distribution affects the performance of our method, we defined a metric to measure the uniformity of a dataset:

$$UI(X) = \frac{-\sum_{i=1}^{n} p_i \ln p_i}{\ln n}, \tag{9}$$

where UI is the defined metric, X is a dataset, p_i denotes the ratio of the samples of the ith class in all samples, and n denotes the number of classes. The more evenly distributed the classes are, the greater the value of the metric will be. The maximum value of the metric is one which indicates that the samples in the dataset are evenly distributed across all classes, and the minimum value of the metric is zero which indicates that the samples in the dataset all come from one class.

Please note that in this experiment, we specify 10 categories for the clustering task, so the class distribution of the dataset is also the cluster distribution. In addition, the annotations are only necessary for theoretical analysis of clustering performance but not in practical applications. To create variants of the USPS dataset, we iteratively reduce the largest classes to the second largest class by removing samples. In particular, we only select an integer multiple of 100 samples from each class. Finally, we obtained five datasets with different UI values. Detailed information about these datasets can be found in the supplementary material. We evaluate our method on each of these datasets and calculate the ACC, NMI and ARI metrics. Fig.5a illustrates how the clustering performance is affected by the UI value. It can be seen that, as the UI value increases, the performance of our method keeps improving. The reported results are median statistics in ten runs. Here, we use the same samples for training and evaluation, with label information only used in the evaluation phase.

Finally, we make a comparison between our method and other methods on the USPS700. The performance comparison is summarized in Table V, and the stability comparison is illustrated in Fig.5b. We use the experimental configuration reported in the literature to run each comparison method. For methods that did not carry out experiments on the USPS dataset, we resize the image to 28x28 and use the MNIST's configuration for the experiment. It can be seen that on the USPS700, our method performs much better and more stable than all the comparison methods. The stable performance is really a surprise as we haven't expected it since we found the instability of our method in section IV-C. This pronounces that the stability of performance depends on specific datasets.



(a) (b)

Fig. 5. (a) illustrates that as the UI value increases, clustering performance will also improve. The lower boundary of the error line represents the minimum of the performance in ten runs, and the upper boundary represents the maximum of the performance in ten runs. (b) illustrates the comparison of the stability of different methods on USPS700. The standard deviation of the performance (ACC, NMI, and ARI) of each method in ten runs is plotted

G. When intrinsicness is corrupted

Cifar-10 [87] is a common benchmark for verifying object recognition algorithms. ImageNet-10 [34] is a tiny version of the origianl ImageNet dataset [5]. In this subsection, we conduct experiments on these two datasets for examining if our method can deal with real-world object recognition tasks. We use the same architecture and hyperparameter configuration as section IV-C for experiments, except that we append a deconvolutional layer to the generator (correspondingly, a convolutional layer is appended to the discriminator and classifier respectively) to process 64x64 images (we resize the images in ImageNet-10 to 64x64). The experimental results are shown in Table VI. The results on Cifar-10 and ImageNet-10 are far from satisfactory. However, we can see that all the comparison methods (except for DCCM [58]) have also failed on these two dataset. This illustrates that there is still a long way to go to apply unsupervised methods in object recognition tasks.

We blame three reasons for the poor performance of deep clustering in object recognition: First, object recognition itself is a challenging task. Understanding the class structure on these dataset requires a lot of out-of-domain knowledge. For examples, although the appearance of chickens, ostriches and canaries varies greatly, they are all considered birds in the dataset. The same happens on freighters, cruise ships and motorboats. In fact, if you ignore the background information, motorboats look more like cars, but are classified as ships along with freighters and cruise ships. Most of the time, we use auxiliary knowledge, such as biology and the usage, to assist in object classification. Second, the quality of the generated samples are still poor, which undermines the criterion that the samples fetched from all clusters are equal to the original samples. This may be due to the insufficient capacity of the generator. Third, the images vary greatly in appearance. There

TABLE V
PERFORMANCE OF VARIOUS CLUSTERING METHODS ON USPS700. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD**. OUR METHOD COMPREHENSIVELY OUTPERFORMS ALL THE COMPARISON METHODS ON THREE METRICS

Method	ACC			NMI			ARI		
	Min	*Max*	*Med*	*Min*	*Max*	*Med*	*Min*	*Max*	*Med*
DDC [48]	-	-	0.967*	-	-	0.918*	-	-	-
DDC-DA [48]	-	-	0.977*	-	-	0.939*	-	-	-
DAC [34]	0.364	0.483	0.391	0.301	0.389	0.342	0.261	0.329	0.288
K-means [13]	0.469	0.564	0.534	0.367	0.451	0.433	0.259	0.368	0.344
DEC [33]	0.605	0.748	0.696	0.626	0.717	0.682	0.464	0.626	0.572
SpectralNet [39]	0.827	0.877	0.835	0.860	0.898	0.865	0.788	0.876	0.799
JULE [37]	0.856	0.954	0.877	0.862	0.893	0.888	0.802	0.908	0.840
DCCM [58]	0.153	0.328	0.293	0.134	0.246	0.201	0.056	0.332	0.119
Ours	**0.933**	**0.985**	**0.981**	**0.936**	**0.978**	**0.971**	**0.885**	**0.967**	**0.957**

TABLE VI
PERFORMANCE COMPARISON ON CIFAR-10 AND IMAGENET-10. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD**. ALL THE RESULTS OF THE COMPARISON METHODS ARE REPORTED IN LITERATURE. FOR OUR METHOD, THE MEDIAN STATISTICS IN 10 RUNS ARE REPORTED

Datasets	Cifar-10			ImageNet-10		
Method	ACC	NMI	ARI	ACC	NMI	ARI
K-means [13]	0.229	0.087	0.049	0.241	0.119	0.057
SC [83]	0.247	0.103	0.085	0.274	0.151	0.076
AC [23]	0.228	0.105	0.065	0.242	0.138	0.067
NMF [82]	0.190	0.081	0.034	0.230	0.132	0.065
AE [88]	0.314	0.239	0.169	0.317	0.210	0.152
SAE [89]	0.297	0.247	0.156	0.325	0.212	0.174
DAE [75]	0.297	0.251	0.163	0.304	0.206	0.138
DeCNN [84]	0.282	0.240	0.174	0.313	0.186	0.142
SWWAE [76]	0.284	0.233	0.164	0.324	0.176	0.160
GAN [64]	0.315	0.265	0.176	0.346	0.225	0.157
JULE [37]	0.272	0.192	0.138	0.300	0.175	0.138
DEC [33]	0.301	0.257	0.161	0.381	0.282	0.203
DAC [34]	0.522	0.396	0.306	0.527	0.394	0.302
DCCM [58]	**0.623**	**0.496**	**0.408**	**0.710**	**0.608**	**0.555**
Ours	0.330	0.315	0.014	0.368	0.377	0.030
Ours*	0.440	0.421	0.223	0.487	0.492	0.310

may be many features that can be easily exploited as cues for cluster assignment, such as style and hue, not just the type of object. Therefore, when we introduce knowledge, such as invariance to translation, rotation, resize, brightness, contrast, saturation, hue, noise and etc., into the training process, we can further improve the clustering performance. That has been illustrated as Ours* in Table VI. Because this paper aims to build a general-purpose deep clustering framework, elaborating on exotic designs dedicated to extracting effective features for specific clustering tasks is beyond the scope of discussion. Here, we just make an indicative specification about incorporating any orthogonal techniques into the framework.

H. Complexity analysis

The training of the framework consumes a relatively long time (generally, 38 hours for MNIST, Fashion-MNIST, Artifact-MNIST and USPS, 1.2 hours for ORL, 56 hours for Cifar-10, 220 hours for ImageNet-10), but after training, the framework outputs the clustering result for an instance (32x32 grayscale/color image or 64x64 color image) within 0.03/0.034/0.09 milliseconds (average over a batch size of 50)[1]. It should be noted that since we used similar experimental configurations for all datasets, the reported performance must be below its maximum value. In this case, when a better hyperparameter is selected, the measured training latency can be reduced by cutting the number of iterations.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we define the objective of deep clustering as finding a precise feature as the cue for cluster assignment. To achieve this objective, we propose a general-purpose deep clustering framework that integrates representation learning and clustering into an individual pipeline for joint optimization. We apply the proposed framework to a synthetic dataset and several real-world image benchmarks. The results showed that the framework performed better than, or comparably to, the baselines. We attribute the promising results to the fact that our framework captures the intrinsic characteristics of samples and learns to select one whose discrete value space exactly matches the specified categories as the cue for cluster assignment. In essence, the proposed framework works in the similar manner as that humans behave in clustering tasks.

However, there are still some limitations for the proposed framework. First, the uniform prior imposed on the clusters only benefits when the samples are approximately uniformly distributed across clusters. Second, the failure on the object recognition datasets suggests that pure statistical methods are difficult to solve complex recognition problems, and it is necessary to introduce additional knowledge or visual mechanisms into the unsupervised framework. In the future, we aim to build more robust methods in order to cater for various clustering scenarios, including adopting a learnable prior to fit more general distributions and using the commonsense provided by humans to induce the learning procedure.

REFERENCES

[1] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.
[2] Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng, "Dual path networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 4467–4475.

[1]All the experiments are performed on computer with Ubuntu 18.04.1 LTS, Intel(R) Xeon(R) CPU E5-1620 v4 @ 3.50GHz, NVIDIA GeForce GTX 1080 Ti, CUDA 10.0, Python 3.6, Tensorflow 1.2.

[3] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, 2017, pp. 2980–2988.
[4] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 640–651, 2017.
[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. Ieee, 2009, pp. 248–255.
[6] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, pp. 843–852.
[7] J. Liu, Y. Gan, J. Dong, L. Qi, X. Sun, M. Jian, L. Wang, and H. Yu, "Perception-driven procedural texture generation from examples," *Neurocomputing*, vol. 291, pp. 21–34, 2018.
[8] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," in *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, 2014, pp. 740–755.
[9] J. N. Wood and S. M. Wood, "The development of invariant object recognition requires visual experience with temporally smooth objects," *Cognitive science*, vol. 42, no. 4, pp. 1391–1406, 2018.
[10] M. J. Arcaro, P. F. Schade, J. L. Vincent, C. R. Ponce, and M. S. Livingstone, "Seeing faces is necessary for face-domain formation," *Nature neuroscience*, vol. 20, no. 10, p. 1404, 2017.
[11] N. Zhou, Y. Xu, H. Cheng, Z. Yuan, and B. Chen, "Maximum correntropy criterion-based sparse subspace learning for unsupervised feature selection," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 29, no. 2, pp. 404–417, 2019.
[12] C. Biernacki, G. Celeux, and G. Govaert, "Assessing a mixture model for clustering with the integrated completed likelihood," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 7, pp. 719–725, July 2000.
[13] J. Wang, J. Wang, J. Song, X. 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, Jan 2015.
[14] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*, 1996, pp. 226–231.
[15] R. LaLonde, D. Zhang, and M. Shah, "Clusternet: Detecting small objects in large scenes by exploiting spatio-temporal information," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 2018, pp. 4003–4012.
[16] H. Yu, A. Wu, and W. Zheng, "Cross-view asymmetric metric learning for unsupervised person re-identification," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 994–1002.
[17] K. Zhao, W. Chu, and A. M. Martinez, "Learning facial action units from web images with scalable weakly supervised clustering," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 2018, pp. 2090–2099.
[18] P. Li and S. Chen, "Shared gaussian process latent variable model for incomplete multiview clustering," *IEEE Trans. Cybernetics*, vol. 50, no. 1, pp. 61–73, 2020.
[19] Y. Pang, J. Xie, F. Nie, and X. Li, "Spectral clustering by joint spectral embedding and spectral rotation," *IEEE Trans. Cybernetics*, vol. 50, no. 1, pp. 247–258, 2020.
[20] B. Zhao, F. Wang, and C. Zhang, "Efficient multiclass maximum margin clustering," in *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, 2008, pp. 1248–1255.
[21] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.
[22] R. Gomes, A. Krause, and P. Perona, "Discriminative clustering by regularized information maximization," in *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada.*, 2010, pp. 775–783.
[23] K. C. Gowda and G. Krishna, "Agglomerative clustering using the concept of mutual nearest neighbourhood," *Pattern recognition*, vol. 10, no. 2, pp. 105–112, 1978.
[24] W. Zhang, D. Zhao, and X. Wang, "Agglomerative clustering via maximum incremental path integral," *Pattern Recognition*, vol. 46, no. 11, pp. 3056–3065, 2013.
[25] X. Chen, J. Z. Huang, F. Nie, R. Chen, and Q. Wu, "A self-balanced min-cut algorithm for image clustering," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 2080–2088.
[26] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.
[27] A. Tacchetti, S. Voinea, and G. Evangelopoulos, "Trading robust representations for sample complexity through self-supervised visual experience," in *Advances in Neural Information Processing Systems*, 2018, pp. 9640–9650.
[28] Z. Ren and Y. J. Lee, "Cross-domain self-supervised multi-task feature learning using synthetic imagery," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 5, 2018.
[29] S. Jenni and P. Favaro, "Self-supervised feature learning by learning to spot artifacts," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 2018, pp. 2733–2742.
[30] P. Zhou, Y. Hou, and J. Feng, "Deep adversarial subspace clustering," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 2018, pp. 1596–1604.
[31] W. Lin, J. Chen, C. D. Castillo, and R. Chellappa, "Deep density clustering of unconstrained faces," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 2018, pp. 8128–8137.
[32] J. Chen, E. S. Azer, and Q. Zhang, "A practical algorithm for distributed clustering and outlier detection," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, 2018, pp. 2253–2262.
[33] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*, 2016, pp. 478–487.
[34] J. Chang, L. Wang, G. Meng, S. Xiang, and C. Pan, "Deep adaptive image clustering," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 5880–5888.
[35] W. Zhou and Q. Zhou, "Deep embedded clustering with adversarial distribution adaptation," *IEEE Access*, vol. 7, pp. 113 801–113 809, 2019.
[36] J. Zhang, C. Li, C. You, X. Qi, H. Zhang, J. Guo, and Z. Lin, "Self-supervised convolutional subspace clustering network," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, 2019, pp. 5473–5482.
[37] 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*, 2016, pp. 5147–5156.
[38] K. G. Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang, "Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 5747–5756.
[39] U. Shaham, K. P. Stanton, H. Li, R. Basri, B. Nadler, and Y. Kluger, "Spectralnet: Spectral clustering using deep neural networks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.
[40] P. K. Mishro, S. Agrawal, R. Panda, and A. Abraham, "A novel type-2 fuzzy c-means clustering for brain mr image segmentation," *IEEE Transactions on Cybernetics*, pp. 1–12, 2020, early access.
[41] J. Wang, G. Zhang, K. Zhang, Y. Zhao, Q. Wang, and X. Li, "Detection of small aerial object using random projection feature with region clustering," *IEEE Transactions on Cybernetics*, pp. 1–14, 2020, early access.
[42] X. Wang, R. Chen, Z. Zeng, C. Hong, and F. Yan, "Robust dimension reduction for clustering with local adaptive learning," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 30, no. 3, pp. 657–669, 2019.
[43] T. Yellamraju and M. Boutin, "Clusterability and clustering of images and other "real" high-dimensional data," *IEEE Trans. Image Processing*, vol. 27, no. 4, pp. 1927–1938, 2018.
[44] P. M. Domingos, "A few useful things to know about machine learning," *Commun. ACM*, vol. 55, no. 10, pp. 78–87, 2012.

[45] F. Tian, B. Gao, Q. Cui, E. Chen, and T. Liu, "Learning deep representations for graph clustering," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, 2014, pp. 1293–1299.

[46] J. Xie, R. B. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, 2016, pp. 478–487.

[47] X. Peng, S. Xiao, J. Feng, W. Yau, and Z. Yi, "Deep subspace clustering with sparsity prior," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, 2016, pp. 1925–1931.

[48] Y. Ren, N. Wang, M. Li, and Z. Xu, "Deep density-based image clustering," *Knowl. Based Syst.*, vol. 197, p. 105841, 2020.

[49] T. Bouwmans, C. Silva, C. Marghes, M. S. Zitouni, H. Bhaskar, and C. Frélicot, "On the role and the importance of features for background modeling and foreground detection," *Computer Science Review*, vol. 28, pp. 26–91, 2018.

[50] D. G. Lowe, "Object recognition from local scale-invariant features," in *ICCV*, 1999, pp. 1150–1157.

[51] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2536–2544.

[52] E. Denton, S. Gross, and R. Fergus, "Semi-supervised learning with context-conditional generative adversarial networks," *arXiv preprint arXiv:1611.06430*, 2016.

[53] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2794–2802.

[54] M. Noroozi, H. Pirsiavash, and P. Favaro, "Representation learning by learning to count," in *The IEEE International Conference on Computer Vision (ICCV)*, 2017.

[55] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *European Conference on Computer Vision*. Springer, 2016, pp. 69–84.

[56] R. Arandjelovic and A. Zisserman, "Look, listen and learn," in *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 609–617.

[57] S. Wang, Y. Xin, D. Kong, and B. Yin, "Unsupervised learning of human pose distance metric via sparsity locality preserving projections," *IEEE Trans. Multimedia*, vol. 21, no. 2, pp. 314–327, 2019.

[58] J. Wu, K. Long, F. Wang, C. Qian, C. Li, Z. Lin, and H. Zha, "Deep comprehensive correlation mining for image clustering," in *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, 2019, pp. 8149–8158.

[59] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[60] J. Schmidhuber, "Learning factorial codes by predictability minimization," *Neural Computation*, vol. 4, no. 6, pp. 863–879, 1992.

[61] J. Schmidhuber, M. Eldracher, and B. Foltin, "Semilinear predictability minimization produces well-known feature detectors," *Neural Computation*, vol. 8, no. 4, pp. 773–786, 1996.

[62] J. Schmidhuber, "Unsupervised minimax: Adversarial curiosity, generative adversarial networks, and predictability minimization," *CoRR*, vol. abs/1906.04493, 2019.

[63] A. Dosovitskiy, J. T. Springenberg, and T. Brox, "Learning to generate chairs with convolutional neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, 2015, pp. 1538–1546.

[64] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[65] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 2813–2821.

[66] J. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative visual manipulation on the natural image manifold," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V*, 2016, pp. 597–613.

[67] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[68] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, 2016, pp. 2234–2242.

[69] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[70] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[71] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, 2017, pp. 5767–5777.

[72] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, 2018, pp. 8792–8802.

[73] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, 2017.

[74] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[75] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of machine learning research*, vol. 11, no. Dec, pp. 3371–3408, 2010.

[76] J. Zhao, M. Mathieu, R. Goroshin, and Y. Lecun, "Stacked what-where auto-encoders," *arXiv preprint arXiv:1506.02351*, 2015.

[77] J. J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial networks," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[78] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *arXiv preprint arXiv:1809.11096*, 2018.

[79] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[80] H. Zhang, I. J. Goodfellow, D. N. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, 2019, pp. 7354–7363.

[81] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[82] D. Cai, X. He, X. Wang, H. Bao, and J. Han, "Locality preserving nonnegative matrix factorization." in *IJCAI*, vol. 9, 2009, pp. 1010–1015.

[83] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," in *Advances in neural information processing systems*, 2005, pp. 1601–1608.

[84] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2010, pp. 2528–2535.

[85] F. Samaria and A. Harter, "Parameterisation of a stochastic model for human face identification," in *Proceedings of Second IEEE Workshop on Applications of Computer Vision, WACV 1994, Sarasota, FL, USA, December 5-7, 1994*, 1994, pp. 138–142.

[86] Y. Wang, Y. P. Tan, Y. Y. Tang, H. Chen, C. Zou, and L. Li, "Generalized and discriminative collaborative representation for multiclass classification," *IEEE Transactions on Cybernetics*, pp. 1–12, 2020, early access.

[87] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[88] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in neural information processing systems*, 2007, pp. 153–160.

[89] A. Ng, "Sparse autoencoder, vol. 72 of," *CS294A Lecture Notes*, 2011.

Supplemental Material for Learning the Precise Feature for Cluster Assignment

Yanhai Gan, Xinghui Dong, Huiyu Zhou, Feng Gao, and Junyu Dong

I. Human's decision-making mechanism

Thought experiment is a very important studying method in physics and plays pivotal role in both science and philosophy [1], [2]. A thought experiment is a device with which one performs an intentional, structured process of intellectual deliberation in order to speculate, within a specifiable problem domain, about potential consequents (or antecedents) for a designated antecedent (or consequent) [3], [4]. Here, we would like to introduce thought experiment to unsupervised learning researches to study how to construct human's decision-making mechanism inspired architecture design. Perceptual grouping has been thoroughly studied by psychologists [5], [6], and even further, Vickery and Jiang empirically demonstrate that visual statistical learning may form an important component of perceptual organization, which alters the perceptual grouping of distinct visual elements [7]. In spite of these studies focusing on how humans perceive objects as organized patterns and objects [8], we emphasize that the designated categories can induce the intelligent agent to learn the precisely matched intrinsic feature as the cue for cluster assignment.

A. Necessary and sufficient condition to achieve perfect clustering result

Supposing there are 100 blocks of 10 colors with 10 blocks for each color, and these blocks are undistinguishable in any other aspect, one is tasked to put the 100 blocks into 10 baskets with the requirement that, when a block is fetched, one should definitely tell which basket it lies in after that all the blocks have been distributed. This simplest toy task has been depicted in the first row of Fig.1. There are many strategies that can be adopted to distribute these blocks. Supposing one chooses strategy to put blocks with the same color into different baskets, it will become impossible for him to guess the correct basket when a block is fetched. To meet the requirement, one would like to put blocks with the same color into one basket. However, a very trivial solution can be made by putting all blocks into one basket. To avoid this, we make the first constraint - there should be no empty basket after the distribution has been finished. Under this constraint, each basket will be filled with ten blocks of the same color, which corresponds to perfect cluster assignment. One can prove that any difference from this assignment will lead to suboptimal solution. This toy task forces one to find the only discriminative feature of blocks and use it as the cue for cluster assignment. If one has found the discriminative feature, i.e., color, he can make cluster assignment according to it. On the contrary, when confronted with the obligation to accomplish this toy task, one would try to discover the only discriminative cue. That is to say, discriminative feature learning and cluster assigning are cooperative tasks, and must work together to produce perfect clustering result.

Now, we make an analysis about the relationship between the toy task and entropy minimization objective. For description clarity, we make some notations. We use y to denote the basket, use x to denote a block, use $y = \mathbf{g}(x)$ to denote putting x into y, and use $\hat{y} = \mathbf{f}(x)$ to denote guessing the basket when a block is fetched. The toy task can thus be depicted as optimizing $\mathbf{g}$ such that $\hat{y} = \mathbf{f}(x)$ can be as accurate as possible under the constraint that the marginal distribution of y covers the entire range of all plausible baskets. According to information theory, reducing the difficulty of assignment conjecture is equivalent to making the distribution of y given x informative, which corresponds to a low information entropy of distribution [9]. Therefore, it is straightforward to formally define the objective of the toy task as minimization of the average of entropies of the distributions of cluster assignment of given examples. In the optimal solution, where each basket is filled with ten blocks of the same color, the entropy of the distribution of the basket assignment of certain block becomes zero. On the contrary, it is easy to deduce that blocks of each color must be put into one basket in order to minimize the average of entropies of the distributions of basket assignment under the constraint that each basket is not empty. That is to say, the entropy minimization objective is the necessary and sufficient condition to perfectly achieve the requirement of this toy task.

B. Robustness of the entropy minimization objective

The second row of Fig.1 provides the 100 blocks with additional 3 different shapes. As one can choose shape as the cue for cluster assignment now, if he does so, he will be faced with the misalignment between shapes and baskets. If he puts blocks of the same shape into one basket, empty basket becomes inevitable. On the contrary, if he distributes blocks of the same shape into different baskets, the requirement of the toy task can't be satisfied. The misalignment between shapes and baskets prevents one from choosing shape as the cue for cluster assignment. This further demonstrates that discriminative feature learning and cluster assigning must work together to produce good result. The third row of Fig.1 provides the blocks with 10 colors, 2 shapes and 5 sizes. Different from before, one is able to choose the combination of shape and size as the cue to accomplish

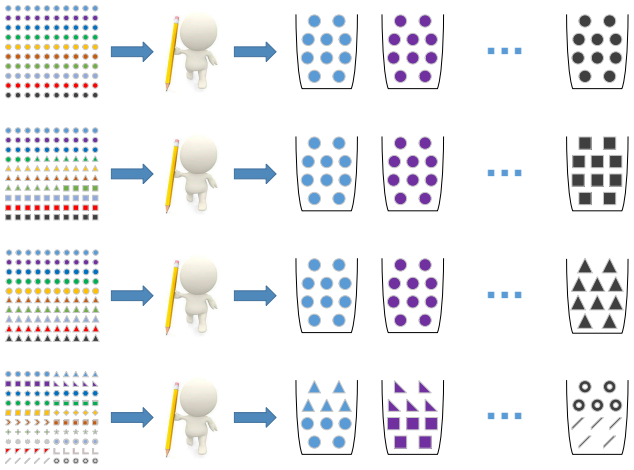


Fig. 1. Sketches of toy tasks. There are 100 blocks in each toy task. The first row depicts the simplest toy task, in which only color of the blocks is varied. The second row gives additional 3 different shapes to the blocks. The third row changes the blocks to be of 10 colors, 2 shapes and 5 sizes. The forth row gives the blocks 10 colors and 20 shapes as features. In all toy tasks, the intelligent agent ultimately chooses color as the cue for cluster assignment, as color is the only feature whose possible values precisely match the number of designated categories.

TABLE I
VARIANTS OF USPS DATASET.

Dataset	Number of samples in each category										UI
	0th class	1th class	2th class	3th class	4th class	5th class	6th class	7th class	8th class	9th class	
USPS1500	1500	1200	900	800	800	700	800	700	700	800	0.985
USPS1200	1200	1200	900	800	800	700	800	700	700	800	0.991
USPS900	900	900	900	800	800	700	800	700	700	800	0.998
USPS800	800	800	800	800	800	700	800	700	700	800	0.999
USPS700	700	700	700	700	700	700	700	700	700	700	1.000

the toy task. Although this strategy does work, we can safely assume that most people, even intelligent agents, prefer simpler solution of choosing color as the cue for cluster assignment, just like scientists use Occam's razor as an abductive heuristic in the development of theoretical models [10]. The fourth row of Fig.1 provides the blocks with 10 colors, 20 shapes. In this configuration, one is able to choose shape as the cue for cluster assignment by dividing these 20 shapes into 10 different subsets. To avoid it, we make the second constraint that the feature chosen as the cue for clustering must be unique-valued in a cluster.

Because samples in practice, especially images, contain a large amount of variables, many of which own an enormous value space, clustering would become meaningless if the intelligent agent is permitted to divide the value space of intrinsic feature into different subsets as a new cue for cluster assignment. That's the reason why we establish the second constraint. Under the first and second constraint, the intelligent agent has to choose color as the cue for cluster assignment in every task, as color is the only intrinsic feature whose possible values match the number of designated categories. If there are several such features, one intelligent agent may arbitrarily choose one from these alternatives. Since any practical case about feature learning and clustering can be boiled down into one of these toy tasks, the derived entropy minimization objective is robust. Till now, we have clearly pointed out the destiny of the entropy minimization objective, which guides the intelligent agent to learn intrinsic feature and simply use it for cluster assignment. It should be noted that, the correctness of the consequence is built upon the premise - the intelligent agent is capable enough to learn intrinsic features of samples, which refer to sample-level features independent of each other.

II. EXPERIMENTAL RESULTS

Fig.2 illustrates the dynamics learned by the framework on synthetic dataset. Fig.3 illustrates the results of ten runs on MNIST and Fashion-MNIST. Fig.4 is the result on Artifact-MNIST, and Fig.5 is the result on USPS700. Table II summarizes the variants of USPS, including the distribution of samples across classes and the uniformness index of each variant.

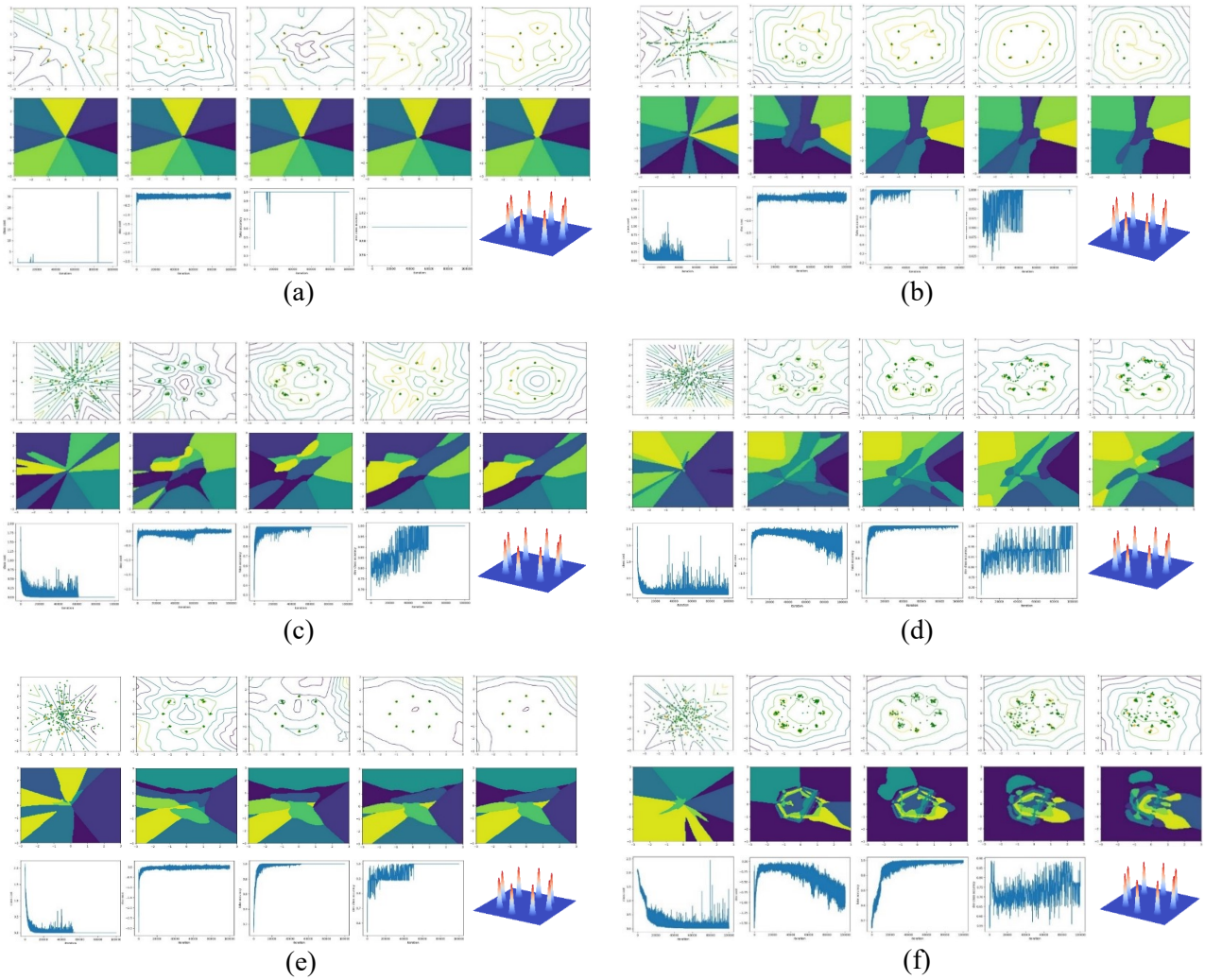


Fig. 2. Experimental results on synthetic dataset. (a),(b),(c),(d),(e),(f) are training dynamics of the framework, in which cluster id, cluster id plus one gaussian noise, cluster id plus two noises, cluster id plus three noises, cluster id plus four noises, cluster id plus eight noises are respectively used as inputs to the generator. The first row in each subfigure depicts the contour lines of the critic values, the generated samples and real samples. The generated samples are drawn as green pluses, the real samples are drawn as orange pluses. The second row in each subfigure depicts the clustering surface, where samples in regions covered by the same color are viewed as in one category by the model. The columns formed by the first two rows in each subfigure are drawn from the 99th, 24999th, 49999th, 74999th and 99999th iteration of the optimization dynamics for the generator. The third row in each subfigure depicts the learning curves of entropy expectation, Wasserstein distance, fake classifying accuracy, evaluated clustering accuracy and the gaussian mixture distribution of true samples. In (a),(b),(c),(e), the generated samples reach the real samples in the end, and the final clustering result is fascinating. In (d),(f), the Wasserstein distance diverges in the later stage of training, and the generated samples correspondingly bias from real samples. As a consequence, the finally leaned clustering surface looks weird, and the evaluated clustering accuracy severely deteriorates. In particular, when using 8 noises besides the cluster id as input to the generator, subfigure (f) presents serious overfitting, in which the final evaluated clustering accuracy gets really terrible without surprise. These phenomenons pronounce that the quality of the generated samples is vital important for the clustering performance of the framework. In this synthetic dataset, the samples are drawn from a mixture of 8 gaussian distributions. Therefore, there are actually 3 intrinsic features existing in the samples. Specifically, the three intrinsic features are the center of each blob, the bias in x-axis, and the bias in y-axis respectively. Among these intrinsic features, the biases in x-axis and y-axis can also be transformed as a distance and an angle from fixed point and direction. Whatever formalization, the degree of freedom insist. Hence, subfigure (c) indeed uses the accurate number of inputting variables identical to that of the intrinsic features for generating, and reasonably produces good clustering results. While the number of variables used as input increases, the performance of the framework quickly becomes unstable. However, when there are less variables than actual intrinsic features in the input, the clustering performance of the framework will not sharply degrade. Based on this observation, we'd better use less variables as input in the framework. In spite of these, subfigure (e) comes into a distinct phenomenon, in which the generated samples and the final evaluated clustering accuracy both get a desired result. To be audacious, we pronounce that this may be caused by the duality of the designated inputting noises, and the framework cleverly learns to combine two of them as one intrinsic feature.

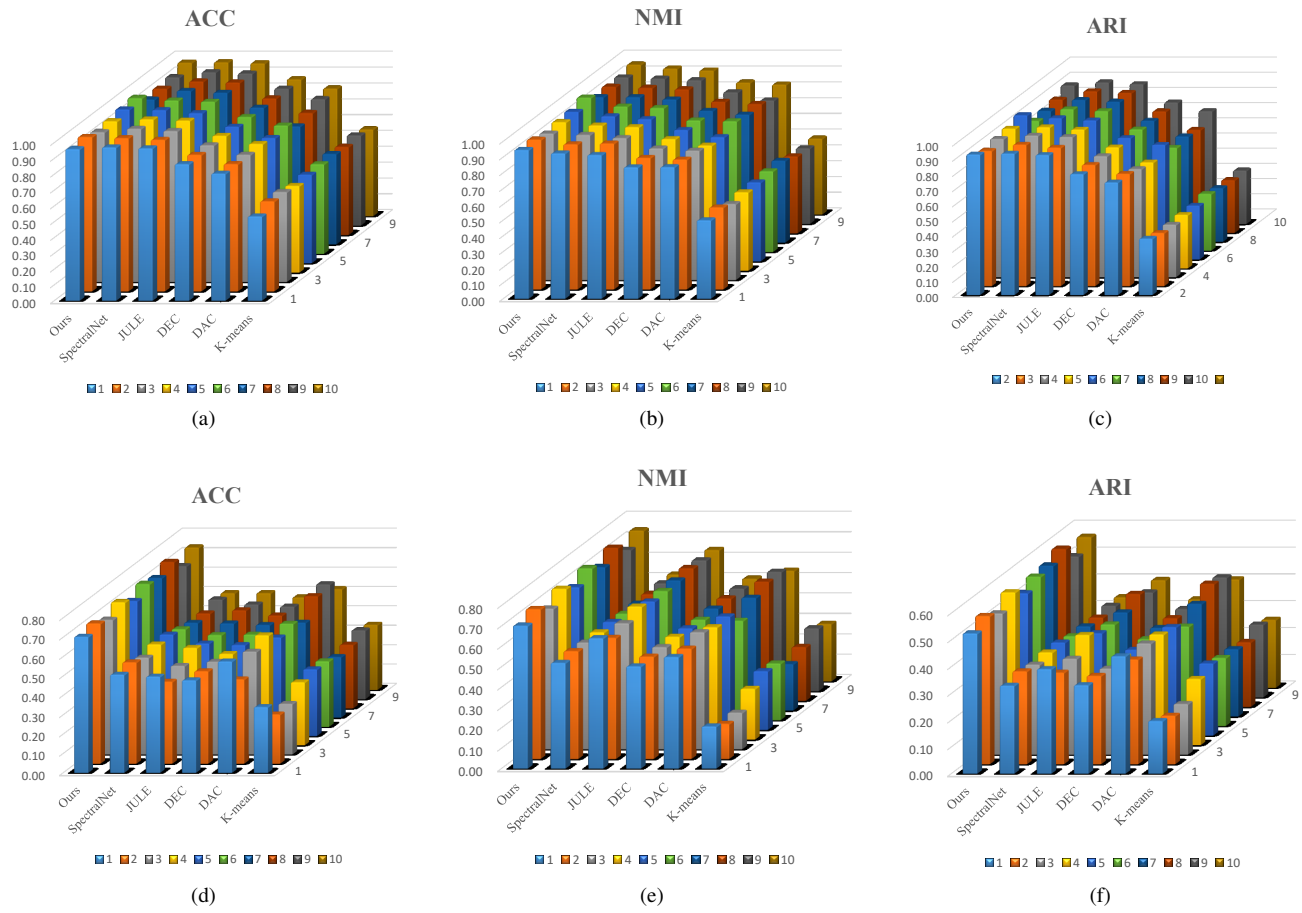


Fig. 3. Performance comparison of different methods on MNIST and Fashion-MNIST. (a)(b)(c) are evaluated on MNIST, (c)(d)(e) are evaluated on Fashion-MNIST. In each subfigure, each method is trained and evaluated by the same training and test splits of the datasets. Specifically, each method on each dataset is trained and evaluated for ten times, and the evaluated ACC, NMI and ARI metrics for each run are illustrated.

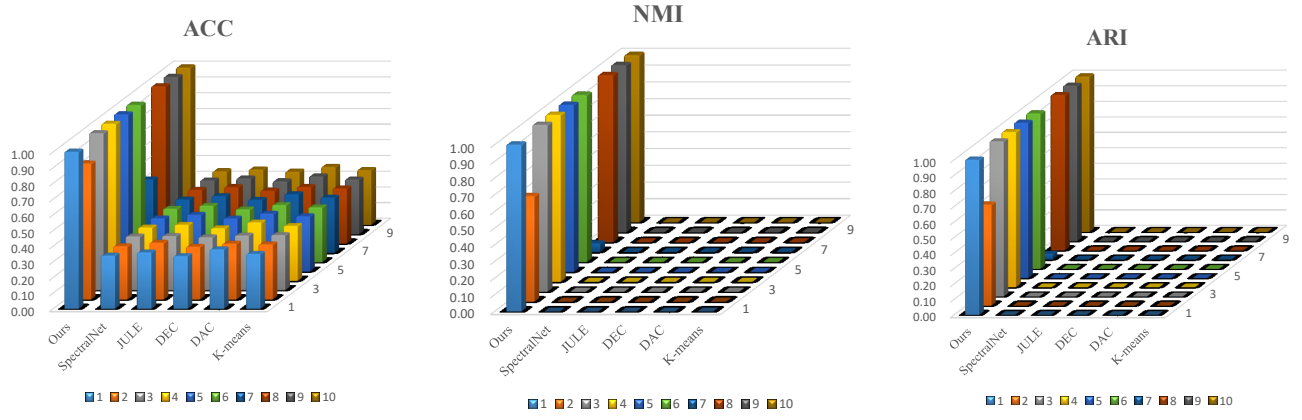
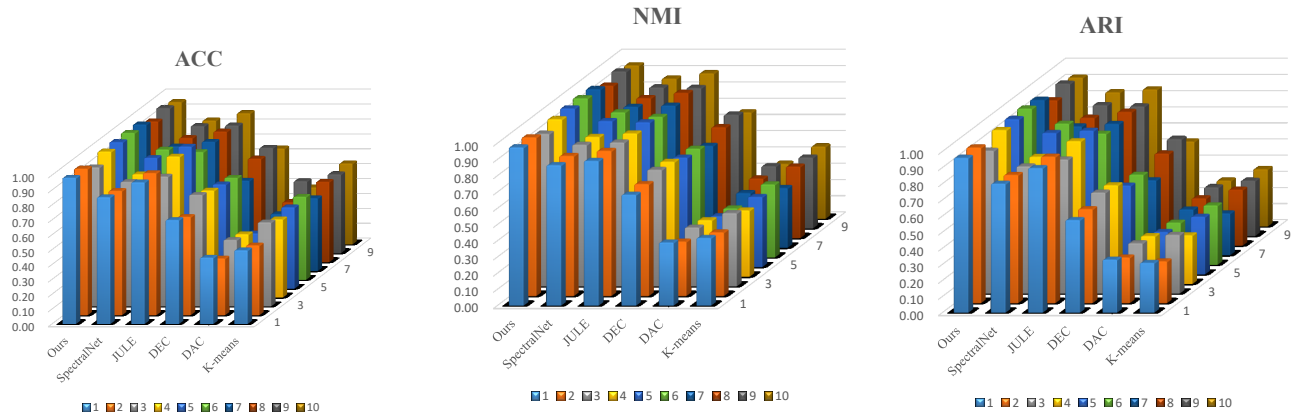


Fig. 4. Performance comparison of different methods on Artifact-MNIST. (a)(b)(c) illustrate the performance comparison of different methods measured by ACC, NMI and ARI. Each method is trained and evaluated by the same training and test split of the dataset for 10 times. It should be noted that all the comparison methods fail on this task.

REFERENCES

[1] C. G. Knott, "Quote from undated letter from maxwell to tait," *Life and Scientific Work of Peter Guthrie Tait. Cambridge University Press*, p. 215, 1911.
[2] M. Cohen, *Wittgenstein's beetle and other classic thought experiments*. John Wiley & Sons, 2008.
[3] L. Yeates, "Thought experimentation: A cognitive approach," *Graduate Diploma in Arts (By Research) dissertation, University of New South Wales*, 2004.
[4] E. Schrödinger, "Die gegenwärtige situation in der quantenmechanik," *Naturwissenschaften*, vol. 23, no. 49, pp. 823–828, 1935.



(a) (b) (c)

Fig. 5. Performance comparison of different methods on USPS700. (a)(b)(c) illustrate the performance comparison of different methods measured by ACC, NMI and ARI. Each method is trained and evaluated by the same training and test split of the dataset for 10 times.

[5] J. H. Elder and R. M. Goldberg, "Ecological statistics of gestalt laws for the perceptual organization of contours," *Journal of Vision*, vol. 2, no. 4, pp. 5–5, 2002.
[6] J. T. Wixted and J. Serences, *Stevens' Handbook of Experimental Psychology and Cognitive Neuroscience, Sensation, Perception, and Attention*. John Wiley & Sons, 2018, vol. 2.
[7] T. J. Vickery and Y. V. Jiang, "Associative grouping: perceptual grouping of shapes by association," *Attention Perception and Psychophysics*, vol. 71, no. 4, pp. 896–909, 2009.
[8] K. Koffka, *Principles of Gestalt psychology*. Routledge, 2013.
[9] C. E. Shannon, "A mathematical theory of communication," *Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.
[10] H. G. Gauch and H. G. Gauch Jr, *Scientific method in practice*. Cambridge University Press, 2003.