Balanced Self-Paced Learning for Generative Adversarial Clustering Network

Kamran Ghasedi Dizaji[1], Xiaoqian Wang[1], Cheng Deng[2], and Heng Huang*[1,3]

[1]Electrical and Computer Engineering Department, University of Pittsburgh, PA, USA
[2]School of Electronic Engineering, Xidian University, Xi'an, Shaanxi, China
[3]JD Digits
kamran.ghasedi@gmail.com, xiaoqian.wang@pitt.edu, chdeng.xd@gmail.com, heng.huang@pitt.edu

Abstract

*Clustering is an important problem in various machine learning applications, but still a challenging task when dealing with complex real data. The existing clustering algorithms utilize either shallow models with insufficient capacity for capturing the non-linear nature of data, or deep models with large number of parameters prone to overfitting. In this paper, we propose a deep Generative Adversarial Clustering Network (*ClusterGAN*), which tackles the problems of training of deep clustering models in unsupervised manner.* ClusterGAN *consists of three networks, a discriminator, a generator and a clusterer (*i.e. *a clustering network). We employ an adversarial game between these three players to synthesize realistic samples given discriminative latent variables via the generator, and learn the inverse mapping of the real samples to the discriminative embedding space via the clusterer. Moreover, we utilize a conditional entropy minimization loss to increase/decrease the similarity of intra/inter cluster samples. Since the ground-truth similarities are unknown in clustering task, we propose a novel balanced self-paced learning algorithm to gradually include samples into training from easy to difficult, while considering the diversity of selected samples from all clusters. Our unsupervised learning framework makes it possible to efficiently train clusterers with large depth. Experimental results indicate that* ClusterGAN *achieves competitive results compared to the state-of-the-art models on several datasets.*

1. Introduction

Clustering is one of the essential active research topics in computer vision and machine learning communities with various applications. Clustering problem has been extensively studied in the literature by introducing numerous algorithms with unsupervised learning frameworks [51]. However, the existing methods that employ shallow or deep models suffer from different issues. The shallow clustering models may not capture the nonlinear nature of data due to their shallow and linear embedding functions, adversely affect their performance by using inflexible hand-crafted features, and have difficulties in scaling to large datasets. In contrast, the deep clustering methods have enough capacity to model the non-linear and complex data, and are able to deal with large-scale datasets. But they are prone to the overfitting issue leading to get stuck in bad local minima, since there is no reliable supervisory signal for training their large number of parameters.

In this paper, we propose a generative adversarial clustering network, called *ClusterGAN*, as a novel deep clustering model to address the aforementioned issues. *ClusterGAN* adopts the adversarial game in *GAN* for the clustering task, and employs an efficient self-paced learning algorithm to boost its performance. The standard *GAN* is formulated as an adversarial game between two networks, a discriminator and a generator [19]. In particular, the generator $\mathcal{G}$ is supposed to synthesize realistic images to fool the discriminator $\mathcal{D}$ by mapping the random input $\mathbf{z}$ into the data space, and the discriminator aims to distinguish the real data from the generated samples. The objective function in this two-player adversarial game between $\mathcal{D}$ and $\mathcal{G}$ is:

$$\min_{\mathcal{G}} \max_{\mathcal{D}} \; \mathbb{E}_{\mathbf{x}\sim P(\mathbf{x})}\big[\log \mathcal{D}(\mathbf{x})\big] + \mathbb{E}_{\mathbf{z}\sim P(\mathbf{z})}\big[\log\big(1-\mathcal{D}(\mathcal{G}(\mathbf{z}))\big)\big], \tag{1}$$

where $P(\mathbf{x})$ is the real data distribution, and $P(\mathbf{z})$ is the generator random input distribution. In this adversarial loss, $\mathcal{G}$ is trained to learn the conditional distribution of real data given the random variables, and $\mathcal{D}$ is trained to find the boundaries between samples drawn from the real and gen-

*Corresponding Author. K. Ghasedi, X. Wang, H. Huang were partially supported by U.S. NSF IIS 1836945, IIS 1836938, DBI 1836866, IIS 1845666, IIS 1852606, IIS 1838627, IIS 1837956.

erated data distributions.

Unlike the traditional *GAN*, *ClusterGAN* consists of three networks, a discriminator $\mathcal{D}$, a generator $\mathcal{G}$, and a clusterer $\mathcal{C}$ (*i.e.* a clustering network). The generator and clusterer are both conditional generative networks, where $\mathcal{G} : \mathbf{z} \rightarrow \hat{\mathbf{x}}$ generates the realistic data samples given the latent variables and $\mathcal{C} : \mathbf{x} \rightarrow \hat{\mathbf{z}}$ generates the discriminative latent variables given the real data. The discriminator $\mathcal{D}$ accepts a joint distribution of samples and features (*i.e.* latent variables) as the input, and tries to identify whether the paired samples belong to the generator $(\mathbf{z}, \hat{\mathbf{x}})$ or the clusterer $(\hat{\mathbf{z}}, \mathbf{x})$. Thus, training the generator and clusterer to fool the discriminator leads to generating synthesized samples similar to real data and estimating features similar to the generator latent variables. By considering a discriminative distribution for the generator inputs, we employ the adversarial game between $\mathcal{D}$, $\mathcal{G}$ and $\mathcal{C}$, and learn a discriminative embedding space in the output of the clusterer. Figure 1 illustrates the architecture of *ClusterGAN*.

Moreover, we introduce a novel clustering objective, which is directly applied on the output of the clusterer given the real samples. The basic idea is to impose a block diagonal constraint on the adjacency matrix of the real data. To do so, we first compute the similarity values between real samples using the cosine similarity function applied on the clusterer outputs. Then, a minimum entropy loss function is imposed to the similarity values to push them towards 0 (*i.e.* dissimilar) or 1 (*i.e.* similar). However, the main challenge is that the ground-truth similarities are unknown in unsupervised learning, which makes it difficult to train a deep clustering model from the scratch. In order to tackle this issue, we enhance the minimum entropy objective by utilizing a novel self-paced learning algorithm. Generally, the standard self-paced learning algorithm initiates the training process with easy samples, and then gradually takes more difficult samples into the training. Considering the difficulty level of samples based on their loss values, the self-paced learning is reported to alleviate the problem of getting stuck in bad local minima, and provides better generalization for the models [30]. In addition to this gradually learning approach, we take the prior of selected samples into consideration using an exclusive lasso regularization. This helps us to select a more diverse set of samples in each training step, and prevents learning from easy samples belonging only to a few clusters. We also provide a theoretical proof for our balanced self-paced learning algorithm in regard to achieving the global optimum closed form solution.

In our experiments, *ClusterGAN* achieves state-of-the-art results compared to the alternative clustering methods on several datasets. We also examine the effects of each component in our learning objective function using an ablation study. Moreover, we evaluate the performance of *ClusterGAN* representations in comparison with unsupervised hash functions on information retrieval tasks. The experimental results confirm the effectiveness of our learning framework in training unsupervised models with large depth. Therefore, the contribution of this paper can be summarized as the following points.

- We introduce a deep clustering model by adopting the generative adversarial network for clustering.
- We propose a novel balanced self-paced learning algorithm for clustering by gradually incorporating easy to more difficult samples into training steps, while keeping the prior of selected samples balanced in each step.
- Our proposed model achieves comparable results to the state-of-the-art methods on clustering and information retrieval tasks.

2. Related Works

2.1. Clustering Algorithms

Countless number of clustering methods have been proposed in the literature, which can be divided into shallow and deep models. In shallow clustering algorithms, K-*means* and Gaussian mixture model (*GMM*) [3] are two classical examples of distance-based clustering methods, which represent the clusters using geometric properties of the data points. The kernel-based algorithms, like max-margin methods [59, 50], attempt to model the non-linearity of data via the proper kernel functions. The connectivity-based algorithms, including spectral methods [37, 56], aim to partition the data points that are highly connected. However, these algorithms are not able to model the complex real-world data because of their shallow and linear models.

Recently, deep clustering models attract more attentions due to their capabilities in dealing with complex, high-dimension and large-scale datasets. A mutli-layer sparse coding network followed by a clustering algorithm is introduced in [47], where an alternative learning approach is used to update the code books and estimate the clustering assignments. Trigeorgis *et al.* stacked multiple semi non-negative matrix factorization layers to achieve discriminative representations at the top layer, and used K-*means* to get cluster assignments [44].

Autoencoder network is also adopted in multiple deep clustering models to build discriminative embedding space using the reconstruction task. Tian *et al.* trained a stacked autoencoder on the affinity matrix of a graph, and then obtained the clusters by running K-*means* at the top layer features [43]. Xie *et al.* introduced deep embedded clustering (*DEC*), which is first pre-trained using the reconstruction loss, and then fine-tuned via Kullback-Leibler divergence minimization [49]. Dizaji *et al.* proposed *DEPICT* as a deep clustering autoencoder network, that is trained using a joint reconstruction loss and relative entropy minimiza-

tion. *DEPICT* also benefits from a regularization term for balancing the prior probability of cluster assignments [14].

Moreover, *JULE* employs a convolutional neural network to represent the features, which are iteratively clustered using an agglomerative clustering algorithm [53]. Yu *et. al.* extended *GMM* to GAN mixture model by allocating a *GAN* model for each cluster [55]. Hu *et. al.* introduced a clustering algorithm, called *IMSAT*, by encouraging the predictions for augmented samples to be close to the original ones, and maximizing the mutual information of the predicted representations. *IMSAT* employs the virtual adversarial training [36] and geometric transformations as data augmentation approaches [22]. *ClusterGAN* differs from the previous models, because it adopts the adversarial game in *GAN* for unsupervised learning of discriminative representations, and employs a novel self-paced learning algorithm for clustering. Consequently, it is able to efficiently train deeper clusterers compared to alternative algorithms.

2.2. Self-Paced Learning Algorithms

Inspired by the human learning principle, curriculum learning starts learning with easier examples, and then gradually takes more complex examples into consideration [2]. But in order to avoid heuristic easiness measures, Kumar *et. al.* proposed self-paced learning algorithm that incorporates curriculum learning into the model optimization. It adds a regularization term to the objective function, and consequently defines easiness measures by the loss value regarding each sample [30]. Jiang *et. al.* extended self-paced learning to also consider the diversity of samples selected in each training step [25]. Many studies further adopted self-paced learning in their tasks to avoid getting stuck in bad local minima and improve the generalization of their models [57, 33, 32]. Our balanced self-paced learning approach differs with the existing methods, since it is applied to an unsupervised loss based on adjacency matrix. It is also specially different with the algorithm in [25], which uses the $\ell_{2,1}$-norm regularization and supervised class labels, but our approach utilizes the exclusive lasso regularization with no need to supervisory signals.

2.3. Generative Adversarial Networks

GAN [19] is a powerful class of deep generative models, and is able to generate realistic images with great details. Particularly, its effective approach is relied on a minimax game between a generator and a discriminator, which compete each other to synthesize more realistic samples and detect the real samples. Several studies further attempted to improve the quality of generated images, for instance by using Laplacian pyramid framework [11], strided convolution layers and batch normalization [40], and a generator conditioning on the class labels or text descriptions [35, 39]. In addition, *GAN* has been adopted in supervised, semi-supervised and unsupervised tasks, which have an inference model (*e.g.* classifier) [6, 10, 41, 16, 15, 46]. Among them, ALI [12] and Triple-GAN [7] are more close to our proposed model, where they are specifically designed for semi-supervised classification, but *ClusterGAN* is developed for clustering. In particular, our learning framework is unique by utilizing a novel self-paced learning algorithm and customized generative adversarial network for clustering.

3. Method

In this section, we first define the adversarial game regarding the minimax objective in *ClusterGAN*, and then explain our conditional entropy minimization loss, which is enhanced by the proposed balanced self-paced learning algorithm. Given n unlabeled samples $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]$ as the inputs, we aim to cluster them into c categories, where the ground-truth labels are represented by $\mathbf{y} = [y_1, ..., y_n]$. While *ClusterGAN* contains three networks, a discriminator, a generator, and a clusterer, our final goal is to construct a block diagonal adjacency matrix $\mathbf{A}$ based on the outputs of the clusterer, where $a_{ij} = 1$ if $y_i = y_j$ and $a_{ij} = 0$ otherwise. Achieving the proper block diagonal adjacency matrix leads to easy clustering assignments with no need to a complicated clustering algorithm. Since the output layer of the clusterer is sigmoid function, we simply use the cosine similarity function to compute the adjacency matrix as $a_{ij} = \hat{\mathbf{z}}_i^\intercal \hat{\mathbf{z}}_j / \|\hat{\mathbf{z}}_i\| \|\hat{\mathbf{z}}_j\|$, where $\hat{\mathbf{z}}_i$ is the clusterer output for the i-th sample, and $\|.\|$ represents the ℓ_2-norm function.

3.1. Cluster-GAN Adversarial Loss

As shown in Figure 1, *ClusterGAN* consists of a discriminator $\mathcal{D}$, a generator $\mathcal{G}$ and a clusterer $\mathcal{C}$, in which the generator and clusterer aims to fool the discriminator by synthesizing realistic samples by $\mathcal{G} : \mathbf{z} \rightarrow \hat{\mathbf{x}}$ and similar latent variable to the generator inputs by $\mathcal{C} : \mathbf{x} \rightarrow \hat{\mathbf{z}}$, and the discriminator tries to distinguish the joint distribution of samples $(\hat{\mathbf{z}}, \mathbf{x})$ and $(\mathbf{z}, \hat{\mathbf{x}})$ coming from the clusterer and generator respectively.

In order to assist constructing the block diagonal adjacency matrix $\mathbf{A}$, we require to set the random input vectors of generator $\mathbf{z}$ to be orthogonal or parallel. To do so, we consider a binary random variable with m/c elements equal to 1 and the remaining equal to 0, where m is the length of $\mathbf{z}$ vector. In this case, if the distribution of clusterer output $\hat{\mathbf{z}}$ becomes similar to the generator input variables $\mathbf{z}$, we achieve the goal of an adjacency matrix with block diagonal structure. But in order to represent the intra-cluster variations, we add small uniform random noise to the inputs of the generator. While this trick empirically helps to generate realistic samples with more diversity, it just has insignificant effect on the block diagonal adjacency matrix.

As mentioned, the discriminator in *ClusterGAN* tries to discriminate the two joint distributions $P(\mathbf{z}, \hat{\mathbf{x}}) =$

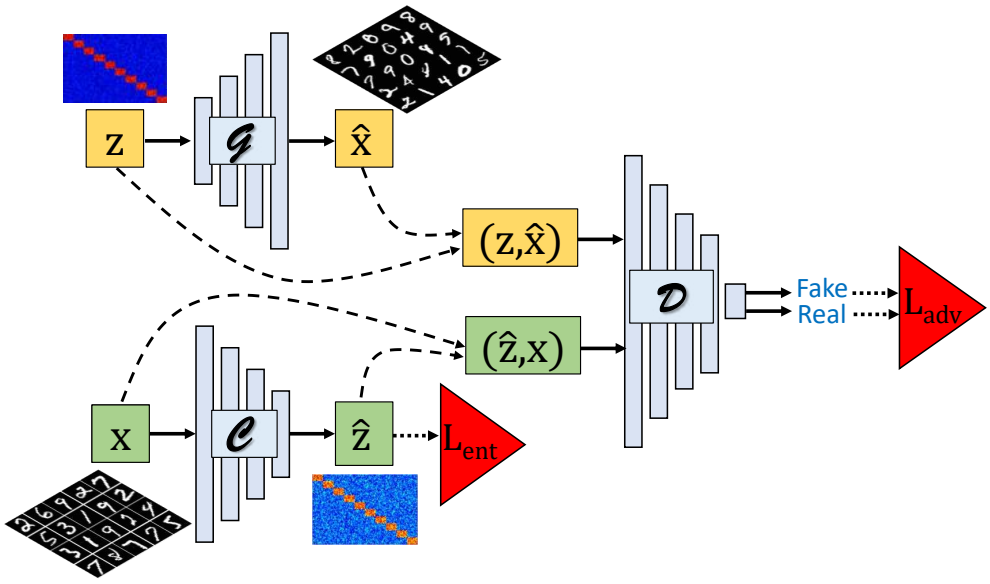


Figure 1: Architecture of *ClusterGAN* along with the applied loss functions. *ClusterGAN* consists of three networks, a generator $\mathcal{G}$, a clusterer $\mathcal{C}$ and a discriminator $\mathcal{D}$. The generator synthesizes the realistic samples given the discriminative random inputs. The clusterer maps the real images into the discriminative latent variables. The discriminator distinguishes whether its input pair belongs to the generator or the clusterer. The adversarial L_{adv} and minimum entropy L_{ent} loss functions are applied to the discriminator and clusterer outputs respectively.

$P(\mathbf{z})P_{\mathcal{G}}(\mathbf{x}|\mathbf{z})$ and $P(\hat{\mathbf{z}},\mathbf{x}) = P(\mathbf{x})P_{\mathcal{C}}(\mathbf{z}|\mathbf{x})$, which are coming form the generator and clusterer respectively. Since the generator random variable distribution $P(\mathbf{z})$ and the empirical distribution of real data $P(\mathbf{x})$ are known, our objective is to learn the conditional distribution of $P_{\mathcal{G}}(\mathbf{x}|\mathbf{z})$ and $P_{\mathcal{C}}(\mathbf{z}|\mathbf{x})$ to match the distributions $P(\mathbf{z},\hat{\mathbf{x}})$ and $P(\hat{\mathbf{z}},\mathbf{x})$. In order to acquire this condition, we employ the adversarial game between $\mathcal{D}$, $\mathcal{G}$ and $\mathcal{C}$ such that the discriminator is trained to identify whether joint pairs are sampled from $P(\mathbf{z},\hat{\mathbf{x}})$ or $P(\hat{\mathbf{z}},\mathbf{x})$, whereas the generator and clusterer are learned to fool the discriminator. Therefore, the objective function of this adversarial game for *ClusterGAN* is:

$$\min_{\mathcal{G},\mathcal{C}} \max_{\mathcal{D}} \; \mathbf{U}(\mathcal{D},\mathcal{G},\mathcal{C}) = \mathbb{E}_{\mathbf{x}\sim P(\mathbf{x})}\big[\log \mathcal{D}\big(\mathcal{C}(\mathbf{x}),\mathbf{x}\big)\big] + \mathbb{E}_{\mathbf{z}\sim P(\mathbf{z})}\big[\log\big(1-\mathcal{D}\big(\mathbf{z},\mathcal{G}(\mathbf{z})\big)\big)\big]. \quad (2)$$

Using this minimax objective function, we are able to alleviate the overfitting issue in training of a deep network with large complexity. This becomes more important in unsupervised clustering task, since there is no reliable supervised information to learn the deep clustering model. It can be shown that the optimal discriminator defined by this objective is balanced between the joint distribution of pairs belonging to the clusterer $P(\hat{\mathbf{z}},\mathbf{x})$ and generator $P(\mathbf{z},\hat{\mathbf{x}})$.

Lemma 1. *For any fixed $\mathcal{G}$ and $\mathcal{C}$, the optimal $\mathcal{D}$ defined by the utility function $\mathbf{U}(\mathcal{D},\mathcal{G},\mathcal{C})$ is:*

$$\mathcal{D}^*(\mathbf{z},\mathbf{x}) = \frac{P(\mathbf{x})P_{\mathcal{C}}(\mathbf{z}|\mathbf{x})}{P(\mathbf{x})P_{\mathcal{C}}(\mathbf{z}|\mathbf{x}) + P(\mathbf{z})P_{\mathcal{G}}(\mathbf{x}|\mathbf{z})}$$

Given $\mathcal{D}^*(\mathbf{x},\mathbf{z})$, we can further replace $\mathcal{D}$ in the utility function $\mathbf{U}(\mathcal{D},\mathcal{G},\mathcal{C})$ and reformulate the objective as $\mathbf{V}(\mathcal{G},\mathcal{C}) = \max_{\mathcal{D}} \mathbf{U}(\mathcal{D},\mathcal{G},\mathcal{C})$, whose optimal value is shown in the following Lemma.

Lemma 2. *The global optimum point of $\mathbf{V}(\mathcal{G},\mathcal{C})$ is achieved if and only if $P(\mathbf{z},\hat{\mathbf{x}}) = P(\hat{\mathbf{z}},\mathbf{x})$.*

Employing this adversarial game in *ClusterGAN*, we can attain the desired clusterer and generator for our problem. In fact, the generator is trained to synthesize the images similar to the real data distribution. The clusterer is trained to learn the inverse mapping function of the generator, estimating discriminative features for the real data. Thus, we can construct an almost block diagonal adjacency matrix from the clusterer outputs. In another point of view, this adversarial loss can be considered as a data-dependent regularization in training our deep clustering model, helping to avoid getting stuck in bad local minima. The proof for Lemma 1 and Lemma 2 are presented in the Appendix.

3.2. Cluster-GAN Entropy Minimization Loss

In addition to the adversarial loss, we introduce a clustering objective based on conditional entropy minimiza-

tion, which is directly applied to the adjacency matrix constructed from the real data. Maximizing the mutual information or minimizing the conditional entropy has been reported to have successful results in clustering [4, 27]. The conditional entropy minimization loss in our problem has the following form:

$$\min_{\mathcal{C}} \; -\sum_{i,j=1}^{n} \left[a_{ij} \log a_{ij} + (1-a_{ij}) \log(1-a_{ij}) \right], \quad (3)$$

in which the adjacency elements a_{ij} are pushed towards 0 or 1. Therefore, minimizing the conditional entropy is in favor of the block diagonal adjacency matrix. However, the similarity values computed from the clusterer features are not reliable especially at the first iterations of training. To tackle this issue, we can use the standard self-paced learning approach, which embeds gradual learning from easy to more difficult samples into model optimization as follows.

$$\min_{\mathcal{C},\boldsymbol{\nu}} \; \sum_{i=1}^{n} \nu_i l_i - \lambda_\nu \sum_{i=1}^{n} \nu_i \,, \quad s.t.\; \boldsymbol{\nu} \in [0,1]^n \quad (4)$$

where $l_i = -\sum_{j=1}^{n} a_{ij} \log a_{ij} - (1-a_{ij}) \log(1-a_{ij})$ is a loss related to the i-th sample, ν_i is the self-paced learning parameter, and λ_ν is a hyper-parameter for controlling the learning pace. The parameters of self-paced learning algorithm and clusterer are generally trained using an alternative learning strategy. Keeping the model parameters fixed, the globally optimum solution for the self-paced learning parameters is $\nu_i^* = 1$ if $l_i < \lambda_\nu$, and $\nu_i^* = 0$ otherwise. It is obvious that by increasing λ_ν throughout training, the self-paced learning algorithm allows more difficult samples into the training process. However, the standard self-paced learning does not consider selecting a balanced set of samples from all clusters, and may choose easy samples of only a few clusters. In order to address this issue, we propose balanced self-paced learning algorithm, which penalizes the lack of diversity using the following objective function:

$$\min_{\mathcal{C},\boldsymbol{\nu}} \; \sum_{k=1}^{c} \left[\sum_{i=1}^{n_k} \nu_{ki}(l_{ki} - \lambda_\nu) + \gamma \Big(\sum_{i=1}^{n_k} |\nu_{ki}| \Big)^2 \right] \quad s.t.\; \boldsymbol{\nu} \in [0,1]^n \,, \quad (5)$$

where γ is the regularization hyper-parameter, and ν_{ki} represents the self-paced learning parameter for the i-th sample of the k-th cluster, where the data are assumed to belong to c clusters as $\sum_{k=1}^{c} n_k = n$. The second term in the loss is also the exclusive lasso regularization $\|\boldsymbol{\nu}\|_e = \sum_{k=1}^{c} \big(\sum_{i=1}^{n_k} |\nu_{ki}| \big)^2$. Note that the balanced self-paced learning objective has two regularization terms, $-\|\boldsymbol{\nu}\|_1 = -\lambda_\nu \sum_{k=1}^{c} \sum_{i=1}^{n_k} \nu_{ki}$ that is in favor of selecting easier samples, and $\|\boldsymbol{\nu}\|_e$ that penalizes groups with more selected samples. Thus, the proposed balanced self-paced learning algorithm consider both the easiness and diversity of selected samples to ensure robust and unbiased training steps. In order to solve this objective function, we use an alternative learning approach, where the clusterer parameters are fixed while obtaining the self-paced learning parameters, and the self-paced parameters are assumed to be known while updating the clusterer parameters. Given the fixed $\mathcal{C}$, the objective function for estimating $\boldsymbol{\nu}$ is:

$$\min_{\boldsymbol{\nu}} \sum_{i=1}^{n} \nu_i l_i - \lambda_\nu \|\boldsymbol{\nu}\|_1 + \gamma \|\boldsymbol{\nu}\|_e \;\; s.t.\; \boldsymbol{\nu} \in [0,1]^n. \quad (6)$$

We derive the global optimum solution for this optimization problem as shown in the following theorem.

Theorem 1. *For any fixed $\mathcal{C}$, the optimal $\boldsymbol{\nu}^*$ defined by the objective function in Eq. (6) is:*

$$\begin{cases} \nu_{kq}^* = 1, & if \quad l_{kq} < \lambda_\nu - 2\gamma q \\ \nu_{kq}^* = \frac{\lambda_\nu - l_{kq}}{2\gamma} - q, & if \quad \lambda_\nu - 2\gamma q \le l_{kq} < \lambda_\nu - 2\gamma(q-1) \\ \nu_{kq}^* = 0, & if \quad l_{kq} \ge \lambda_\nu - 2\gamma(q-1) \end{cases}$$

where $q \in \{1, ..., n_k\}$ is the sorted index based on the loss values $\{l_{k1}, ..., l_{kn_k}\}$ in the k-th group.

This solution intuitively makes sense, since the samples with loss greater/less than the threshold $\lambda_\nu - 2\gamma(q-1)$ are considered as the difficult/easy samples, and are not-involved/involved in the current training step. Interestingly, the threshold is also a function of the ordered loss in each group, and consequently is increased as the number of samples in a cluster increases. Hence, the balanced self-paced learning algorithm considers both the easiness and diversity of selected samples in our learning framework. The proof for Theorem 1 is presented in Appendix.

In addition to the adversarial loss and the minimum entropy loss, we utilize a consistency loss to train the clusterer

Algorithm 1: *ClusterGAN* Algorithm

for *number of training iterations* **do**
- Sample a batch of pairs $(\mathbf{x}, \mathcal{C}(\mathbf{x}))$ and $(\mathcal{G}(\mathbf{z}), \mathbf{z})$ using the clusterer and generator
- Update the discriminator parameters by
$$\max_{\mathcal{D}} \; \sum_{i=1}^{n} \log \mathcal{D}\big(\mathcal{C}(\mathbf{x}_i), \mathbf{x}_i\big) + \sum_{j=1}^{n} \log\big(1 - \mathcal{D}\big(\mathbf{z}_j, \mathcal{G}(\mathbf{z}_j)\big)\big)$$
- Update the generator parameters by
$$\min_{\mathcal{G}} \; \sum_{j=1}^{n} \log\big(1 - \mathcal{D}\big(\mathbf{z}_j, \mathcal{G}(\mathbf{z}_j)\big)\big)$$
- Update the clusterer parameters by
$$\min_{\mathcal{C}} \; \sum_{j=1}^{n} \log \mathcal{D}\big(\mathcal{C}(\mathbf{x}_i), \mathbf{x}_i\big) + \nu_i l_i + \|\mathcal{C}(\mathbf{x}_i) - \mathcal{C}(\tilde{\mathbf{x}}_i)\|^2$$
- Update the self-paced learning parameters by
$$\min_{\boldsymbol{\nu}} \sum_{i=1}^{n} \nu_i l_i - \lambda_\nu \|\boldsymbol{\nu}\|_1 + \gamma \|\boldsymbol{\nu}\|_e \;\; s.t.\; \boldsymbol{\nu} \in [0,1]^n$$

end

parameters. The consistency loss encourages the clusterer to have similar outputs for each samples $\mathbf{x}$ and its variations $\tilde{\mathbf{x}}$ augmented by image transformations or noise as follows:

$$\min_{\mathcal{C}} \sum_{i=1}^{n} \|\mathcal{C}(\mathbf{x}_i) - \mathcal{C}(\tilde{\mathbf{x}}_i)\|^2 . \quad (7)$$

The minimum entropy loss function in Eq. (3) is defined on the full-batch, and has quadratic complexity *w.r.t.* the number of samples. However, we practically alleviate this scalability issue by applying the loss only to the samples of each mini-batch. Algorithm 1 shows the training steps for *ClusterGAN*, where all of the networks are trained using our alternative leaning framework.

4. Experiments

We perform several experiments to evaluate the performance of *ClusterGAN* in clustering and information retrieval tasks on several datasets. We also examine the effect of each component in our learning framework using an ablation study.

Datasets: We examine *ClusterGAN* clustering performance in comparison with alternative algorithms on *MNIST* [31], *USPS*, *FRGC* [53], *CIFAR-10* [28] and *STL-10* [8] datasets. Moreover, we compare *ClusterGAN* with unsupervised hash functions in the image retrieval task on *MNIST* and *CIFAR-10* datasets. Table 1 provides the summary of datasets statistics.

Implementation details: We mainly use the architectures of *Triple-GAN* in [7] for *ClusterGAN* except the last layer of clusterer, which is set as same as the size of generator input with the sigmoid non-linearity. For image preprocessing, we only normalize the image intensities to be in the range of $[-1, 1]$ on *CIFAR-10* and *STL-10* and $[0, 1]$ for the others, and consequently use the tangent-hyperbolic and sigmoid functions in the last layer of the generator. The added noise to the generator inputs has uniform distribution with range $[0, 0.5]$ which is linearly shrinking to $[0, 0.1]$ throughout training. Moreover, we set the learning rate to 10^{-4} and linearly decrease it to 10^{-5}, and adopt Adam [26] as our optimization method with the hyper-parameters $\beta_1 = 0.5$, $\beta_2 = 0.999$, $\epsilon = 1e-08$. In order to avoid manually setting λ_ν and γ for different datasets, we choose them based on the loss values of samples such that we start training with only 1% of samples at the first iteration, and then linearly increase λ_ν to include all samples in $3/4$ of the maximum epoch. We run *K-means* on the clusterer outputs for clustering, and use the indicator function $\mathbf{1}(. > 0.5)$ to binarize the clusterer outputs for hashing. The reported results are all the average of 5 experimental outcomes. We use Theano toolbox [1] for writing our code, and run the algorithm on a machine with one Titan X Pascal GPU.

Dataset	# Samples	# Classes	# Dimensions
MNIST	70,000	10	1×28×28
USPS	11,000	10	1×16×16
FRGC	2,462	20	3×32×32
CIFAR-10	60,000	10	3×32×32
STL-10	13,000	10	3×96×96

Table 1: Dataset Descriptions

4.1. Image Clustering

Alternative Models: We compare our clustering model with several baselines and state-of-the-art clustering algorithms, including *K-means*, normalized cuts (*N-Cuts*) [42], large-scale spectral clustering (*SC-LS*) [5], agglomerative clustering via path integral (*AC-PIC*) [58], spectral embedded clustering (*SEC*) [38], local discriminant models and global integration (*LDMGI*) [54], *NMF* with deep model (*NMF-D*) [44], deep embedded clustering (*DEC*) [49], joint unsupervised learning (*JULE-RC*) [53], *DEPICT* [14] and *IMSAT* [22].

Evaluation metrics: To compare the clustering performance of our model with previous studies, we rely on the two popular metrics used to evaluate clustering: *normalized mutual information* (NMI), and *accuracy* (ACC). NMI provides a measure of similarity between two data with the same label, which is normalized between 0 (lowest similarity) to 1 (highest similarity) [52]. To calculate ACC, we find the best map between the predicted clusters and the true labels [29].

Performance comparison: Table 2 shows the clustering results of *ClusterGAN* and the alternative models on five datasets. As it is expected, the deep clustering models mostly have better results than their shallow alternatives. Among the deep models, *ClusterGAN* outperforms the other methods almost on all datasets. Note that the *IMSAT* results on *CIFAR-10* and *STL-10* are obtained using the 50-layer pre-trained deep residual networks on ImageNet dataset [9], and cannot be compared to the results of other models trained with no supervisory signals. It is worth mentioning that *ClusterGAN* is able to train deeper clustering networks (*e.g.* 9 hidden layers on *CIFAR-10*) compared to the other deep models (*e.g.* 3 or 4 hidden layers on *CIFAR-10*). This effective learning framework could be the reason for *ClusterGAN*'s better performances on the more complex datasets like *CIFAR-10* and *STL-10*. This experiment confirms the efficiency *ClusterGAN* discriminative representations in clustering of different datasets with various sizes, dimensions and complexities.

4.2. Ablation Study

We perform an ablation study to examine the contribution of the adversarial loss (*GAN*), the balanced self-

	Dataset	*MNIST*		*USPS*		*FRGC*		*CIFAR-10*		*STL-10*	
	Model	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC
Shallow	*K-means*	0.500	0.534	0.450	0.460	0.287	0.243	0.102*	0.239*	0.209*	0.284*
	N-Cuts [42]	0.411	0.327	0.675	0.314	0.285	0.235	-	-	-	-
	SC-LS [5]	0.706	0.714	0.681	0.659	0.550	0.407	0.114*	0.258*	0.105*	0.168*
	AC-PIC [58]	0.017	0.115	0.840	0.855	0.415	0.320	0.118*	0.264*	0.235*	0.329*
	SEC [38]	0.779	0.804	0.511	0.544	-	-	0.107*	0.249*	0.245*	0.307*
	LDMGI [54]	0.802	0.842	0.563	0.580	-	-	0.109*	0.253*	0.260*	0.331*
Deep	*NMF-D* [44]	0.152	0.175	0.287	0.382	0.259	0.274	-	-	-	-
	DEC [49]	0.816	0.844	0.586	0.619	0.505	0.378	0.267*	0.312*	0.284*	0.359*
	JULE-RC [53]	0.913	0.964	0.913	0.950	0.574	0.461	0.194*	0.275*	0.204*	0.288*
	DEPICT [14]	0.917	0.965	0.927	0.964	0.610	0.470	0.274*	0.326*	0.303*	0.371*
	IMSAT [22]	-	**0.984**	-	-	-	-	-	0.456†	-	0.941†
	ClusterGAN	**0.921**	0.964	**0.931**	**0.970**	**0.615**	**0.476**	**0.323**	**0.412**	**0.335**	**0.423**

Table 2: Clustering performance of *ClusterGAN* and several alternative models on several datasets based on ACC and NMI. The results of other models are reported from the reference papers, except for the ones marked by $(*)$ on top, which are obtained by us running the released code. The result with † sign are for the models with supervised pre-training.

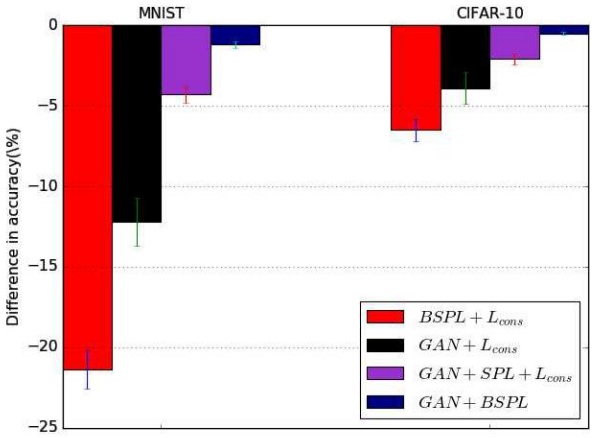


Figure 2: The difference in clustering accuracy, when *ClusterGAN* is trained using some components of the original objective function.

paced learning algorithm (*BSPL*), and the consistency loss (L_{cons}). To do so, we train *ClusterGAN* without *GAN* architecture and adversarial loss (*BSPL*+L_{cons}), without the balanced self-paced learning algorithm (*GAN*+L_{cons}), and without the consistency loss (*GAN+BSPL*). Moreover, we explore the effect of exclusive lasso regularization in *BSPL* by training *ClusterGAN* using the standard self-paced learning algorithm (*GAN+SPL+*L_{cons}). Figure 2 illustrates the difference in accuracy between each scenario and the original *ClusterGAN* on *MNIST* and *CIFAR-10* datasets.

The first observation is that all of the terms contribute in improving the results. Moreover, the figure shows the strong effect for *GAN* as a key components to avoid getting stuck in bad local minima. It also demonstrates that the balanced self-paced learning is important in stable training, and also has better results compared to standard self-paced learning approach. Furthermore, the relative analysis of the results in both dataset demonstrates that consistency loss is more effective on *CIFAR-10* than on *MNIST*. This is expected as we only use noise for image transformation on MNIST since the images are centered and scaled, but employ extra transformations including translations and horizontal flipping on *CIFAR-10*.

Moreover, we visualize the embedding subspace of a few clustering models on *USPS dataset* in Figure 3. The figure shows the 2D visualization of clusterer outputs for *BSPL*+L_{cons} and *ClusterGAN* using principle component analysis (PCA). In addition, we also illustrate the raw data in the input space. As shown in the figure, *ClusterGAN* provides a significantly more discriminative embedding subspace compared to the other model and raw data.

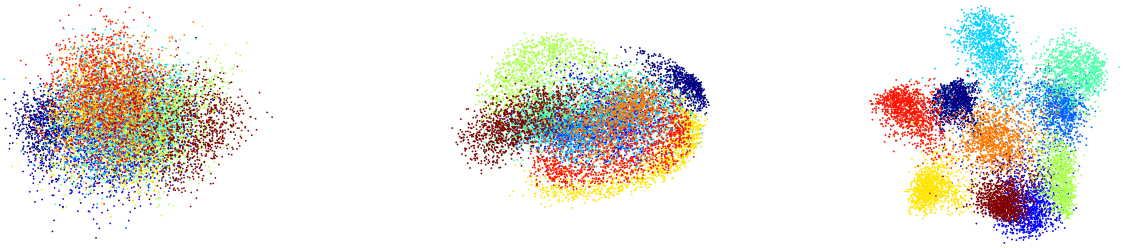
4.3. Image Retrieval

Alternative models: For image retrieval, we compare our method with the previous unsupervised hash functions including K-means hashing (*KMH*) [20], spherical hashing (*SphH*) [21], spectral hashing (*SpeH*) [48], PCA-based hashing (*PCAH*) [45], locality sensitivity hashing (*LSH*) [17], iterative quantization (*ITQ*) [18], deep hashing (*DH*) [13], discriminative attributes representations (*DAR*) [23], *DeepBit* [34] and unsupervised triplet hashing (*UTH*) [24].

Evaluation metrics: We evaluate the performance of *ClusterGAN* compared to the other unsupervised hashing functions using precision and mean average precision (mAP). We follow the standard protocol for both *MNIST* and *CIFAR-10* datasets, and randomly sample 1000 images as the query set and use the remaining data as the gallery set. In particular, we report the results of the image retrieval in terms of precision@1000, mAP, and mAP@1000.

	Dataset	*CIFAR-10*						*MNIST*					
		precision@1000		mAP		mAP@1000		precision@1000		mAP		mAP@1000	
	Model	32	64	32	64	32	64	32	64	32	64	32	64
Shallow	*KMH* [20]	19.72	20.16	13.93	14.46	23.56*	25.19*	53.82*	54.13*	33.29	35.78	70.32*	67.62*
	SphH [21]	20.91*	23.25*	14.58	15.38	24.16*	26.09*	54.74*	62.50*	30.77	34.75	65.45*	65.45*
	SpeH [48]	18.83	19.72	12.42	12.56	21.79*	21.97*	53.75*	54.13*	25.72	24.10	64.37*	67.60*
	PCAH [45]	19.35	18.73	12.60	12.10	21.62*	20.54*	51.90*	48.36*	24.85	21.47	64.47*	63.31*
	LSH [17]	19.10	22.25	13.76	15.07	16.31*	18.00*	45.05*	55.31*	25.83	31.71	50.45*	66.23*
	ITQ [18]	25.30	27.09	16.20	16.64	27.41*	28.93*	68.80*	71.00*	43.82	45.37	76.86*	80.23*
Deep	*DH* [13]	16.62	16.96	16.62	16.96	-	-	-	-	44.97	46.74	-	-
	DAR [23]	26.62	28.06	17.01	17.21	-	-	-	-	-	-	-	-
	DeepBit [34]	-	-	-	-	24.86†	27.73†	-	-	-	-	32.02†	44.53†
	UTH [24]	-	-	-	-	30.66†	32.41†	-	-	-	-	46.58†	49.88†
	ClusterGAN	**40.62**	**42.51**	**29.47**	**30.53**	**43.34**	**45.12**	**90.83**	**91.60**	**88.70**	**89.93**	**91.48**	**92.37**

Table 3: Image retrieval results (%) of *ClusterGAN* and unsupervised hash functions on CIFAR-10 and MNIST datasets, when the number of hash bits are 32 and 64. The results of other models are reported from the reference papers, except for the ones marked by (∗) on top, which are obtained by us running the released code. The result with † sign are for the models with supervised pre-training.



(a) Raw data (b) *BSPL+*L_{cons} (c) *ClusterGAN*

Figure 3: Visualization of different data representations on *USPS* dataset using principle component analysis (PCA) . (a) The space of raw data. (b) The embedding subspace of *ClusterGAN* without *GAN* architecture and adversarial loss denoted by *BSPL+*L_{cons}. (c) The embedding subspace of *ClusterGAN*.

Performance comparison: Another way to measure the effectiveness of *ClusterGAN* discriminative representation is to evaluate its performance in hashing tasks. As shown in Table 3, *ClusterGAN* consistently outperforms alternative models with significant margins across different number of bits, datasets and metrics. Interestingly, the unsupervised deep hash functions, which use supervised pre-training via ImageNet dataset, show better results on *CIFAR-10* dataset compared to the shallow models, but have relatively lower performance on *MNIST* dataset due to the difference in transfer data distribution. With no need to supervised pre-training, our model is not affected by pre-training bias and achieves superior results on both datasets.

5. Conclusion

In this paper, we proposed a generative adversarial clustering network, denoted by *ClusterGAN*, as a new deep clustering model. *ClusterGAN* consists of three networks, a generator, a discriminator, and a clusterer. In order to efficiently train the deep clusterer without any supervised information, we introduced an adversarial game between the three networks, such that the generator synthesizes the realistic images given the discriminative random inputs, the clusterer inversely maps the real samples into the discriminative features. We further proposed a minimum entropy loss on the real data along with a balanced self-paced learning algorithm to enhance the training of the clusterer. The balanced self-paced learning algorithm improves the generalization of our clusterer by gradually decreasing the easiness of included samples in the training process, while considering the diversity of selected samples. Experimental results demonstrated that *ClusterGAN* achieves state-of-the-art results in the clustering and information retrieval tasks, and confirmed the effectiveness of each component in our learning framework using an ablation study.

References

[1] R. Al-Rfou, G. Alain, A. Almahairi, C. Angermueller, D. Bahdanau, N. Ballas, F. Bastien, J. Bayer, A. Belikov, A. Belopolsky, et al. Theano: A python framework for fast computation of mathematical expressions. *arXiv preprint*, 2016. 6

[2] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009. 3

[3] 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*, pages 719–725, 2000. 2

[4] J. S. Bridle, A. J. Heading, and D. J. MacKay. Unsupervised classifiers, mutual information and'phantom targets. In *Advances in Neural Information Processing Systems*, pages 1096–1101, 1992. 5

[5] X. Chen and D. Cai. Large scale spectral clustering with landmark-based representation. In *AAAI*, 2011. 6, 7

[6] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016. 3

[7] L. Chongxuan, T. Xu, J. Zhu, and B. Zhang. Triple generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 4091–4101, 2017. 3, 6

[8] A. Coates, A. Y. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics*, pages 215–223, 2011. 6

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE, 2009. 6

[10] Z. Deng, H. Zhang, X. Liang, L. Yang, S. Xu, J. Zhu, and E. P. Xing. Structured generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 3902–3912, 2017. 3

[11] E. L. Denton, S. Chintala, R. Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pages 1486–1494, 2015. 3

[12] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016. 3

[13] V. Erin Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou. Deep hashing for compact binary codes learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2475–2483, 2015. 7, 8

[14] K. Ghasedi Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5736–5745, 2017. 3, 6, 7

[15] K. Ghasedi Dizaji, X. Wang, and H. Huang. Semi-supervised generative adversarial network for gene expression inference. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1435–1444. ACM, 2018. 3

[16] K. Ghasedi Dizaji, F. Zheng, N. Sadoughi, Y. Yang, C. Deng, and H. Huang. Unsupervised deep generative adversarial hashing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3664–3673, 2018. 3

[17] A. Gionis, P. Indyk, R. Motwani, et al. Similarity search in high dimensions via hashing. In *VLDB*, volume 99, pages 518–529, 1999. 7, 8

[18] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *In Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition*, 2011. 7, 8

[19] 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*, pages 2672–2680, 2014. 1, 3

[20] K. He, F. Wen, and J. Sun. K-means hashing: An affinity-preserving quantization method for learning binary compact codes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2938–2945, 2013. 7, 8

[21] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon. Spherical hashing. In *Computer Vision and Pattern Recognition , 2012 IEEE Conference on*, pages 2957–2964. IEEE, 2012. 7, 8

[22] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, and M. Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *International Conference on Machine Learning*, pages 1558–1567, 2017. 3, 6, 7

[23] C. Huang, C. Change Loy, and X. Tang. Unsupervised learning of discriminative attributes and visual representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5175–5184, 2016. 7, 8

[24] S. Huang, Y. Xiong, Y. Zhang, and J. Wang. Unsupervised triplet hashing for fast image retrieval. *arXiv preprint arXiv:1702.08798*, 2017. 7, 8

[25] L. Jiang, D. Meng, S.-I. Yu, Z. Lan, S. Shan, and A. Hauptmann. Self-paced learning with diversity. In *Advances in Neural Information Processing Systems*, pages 2078–2086, 2014. 3

[26] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[27] A. Krause, P. Perona, and R. G. Gomes. Discriminative clustering by regularized information maximization. In *Advances in neural information processing systems*, pages 775–783, 2010. 5

[28] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Computer Science Department, University of Toronto, Tech. Rep*, page 7, 2009. 6

[29] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics (NRL)*, pages 83–97, 1955. 6

[30] M. P. Kumar, B. Packer, and D. Koller. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, pages 1189–1197, 2010. 2, 3

[31] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, pages 2278–2324, 1998. 6

[32] H. Li, M. Gong, D. Meng, and Q. Miao. Multi-objective self-paced learning. In *AAAI*, pages 1802–1808, 2016. 3

[33] J. Liang, Z. Li, D. Cao, R. He, and J. Wang. Self-paced cross-modal subspace matching. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 569–578. ACM, 2016. 3

[34] K. Lin, J. Lu, C.-S. Chen, and J. Zhou. Learning compact binary descriptors with unsupervised deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1183–1192, 2016. 7, 8

[35] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 3

[36] T. Miyato, S.-i. Maeda, M. Koyama, K. Nakae, and S. Ishii. Distributional smoothing with virtual adversarial training. *arXiv preprint arXiv:1507.00677*, 2015. 3

[37] A. Y. Ng, M. I. Jordan, Y. Weiss, et al. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, pages 849–856, 2002. 2

[38] F. Nie, Z. Zeng, I. W. Tsang, D. Xu, and C. Zhang. Spectral embedded clustering: A framework for in-sample and out-of-sample spectral clustering. *IEEE Transactions on Neural Networks*, pages 1796–1808, 2011. 6, 7

[39] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016. 3

[40] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 3

[41] 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*, pages 2234–2242, 2016. 3

[42] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000. 6, 7

[43] F. Tian, B. Gao, Q. Cui, E. Chen, and T.-Y. Liu. Learning deep representations for graph clustering. In *AAAI*, pages 1293–1299, 2014. 2

[44] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, and B. Schuller. A deep semi-nmf model for learning hidden representations. In *International Conference on Machine Learning*, pages 1692–1700, 2014. 2, 6, 7

[45] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *Computer Vision and Pattern Recognition , 2010 IEEE Conference on*, pages 3424–3431. IEEE, 2010. 7, 8

[46] X. Wang, K. Ghasedi Dizaji, and H. Huang. Conditional generative adversarial network for gene expression inference. *Bioinformatics*, 34(17):i603–i611, 2018. 3

[47] Z. Wang, S. Chang, J. Zhou, M. Wang, and T. S. Huang. Learning a task-specific deep architecture for clustering. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 369–377. SIAM, 2016. 2

[48] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Advances in neural information processing systems*, pages 1753–1760, 2009. 7, 8

[49] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International Conference on Machine Learning*, 2016. 2, 6, 7

[50] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In *Advances in neural information processing systems*, pages 1537–1544, 2004. 2

[51] R. Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on neural networks*, pages 645–678, 2005. 1

[52] W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267–273. ACM, 2003. 6

[53] 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*, pages 5147–5156, 2016. 3, 6, 7

[54] Y. Yang, D. Xu, F. Nie, S. Yan, and Y. Zhuang. Image clustering using local discriminant models and global integration. *IEEE Transactions on Image Processing*, 19(10):2761–2773, 2010. 6, 7

[55] Y. Yu and W.-J. Zhou. Mixture of gans for clustering. In *IJCAI*, 2018. 3

[56] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Advances in neural information processing systems*, pages 1601–1608, 2005. 2

[57] D. Zhang, D. Meng, C. Li, L. Jiang, Q. Zhao, and J. Han. A self-paced multiple-instance learning framework for co-saliency detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 594–602, 2015. 3

[58] W. Zhang, D. Zhao, and X. Wang. Agglomerative clustering via maximum incremental path integral. *Pattern Recognition*, 46(11):3056–3065, 2013. 6, 7

[59] B. Zhao, F. Wang, and C. Zhang. Efficient multiclass maximum margin clustering. In *Proceedings of the 25th international conference on Machine learning*, pages 1248–1255. ACM, 2008. 2