MCL-GAN: Generative Adversarial Networks with Multiple Specialized Discriminators

Anonymous Author(s)
Affiliation
Address
email

Abstract

We propose a generative adversarial network with multiple discriminators, which collaborate to represent a real dataset more effectively. This approach facilitates learning a generator consistent with the underlying data distribution based on real images and thus mitigates the chronic mode collapse problem. From the inspiration of multiple choice learning, we guide each discriminator to have expertise in the subset of the entire data and allow the generator to find reasonable correspondences between the latent and real data spaces automatically without the extra supervision for training examples. Despite the use of multiple discriminators, the backbone networks are shared across the discriminators and the increase of training cost is marginal. We demonstrate the effectiveness of our algorithm using multiple evaluation metrics in the standard datasets for diverse tasks.

1 Introduction

Generative models learn to represent a probability distribution of data. With recent advances of deep generative models, Generative Adversarial Networks (GANs) [1] and Variational Autoencoders (VAEs) [2] have shown impressive achievements in unconditional generation of high-dimensional realistic images as well as various conditional generation tasks including image-to-image translation [3–5], image inpainting [6], image super-resolution [7], *etc*.

GANs have received a lot of attention due to their interesting framework of minimax games, where two agents, a generator and a discriminator, compete against each other. Specifically, a discriminator distinguishes whether a sample comes from the real dataset or the generator while the generator attempts to deceive the discriminator. In theory, the generator learns the real data distribution by reaching an equilibrium point of the minimax game. It is known that GANs produce acute, high quality images compared to VAEs. However, in practice, the alternating training procedure does not guarantee the convergence to the optimal solution and often experiences mode collapsing, failing to cover the multiple modes of real data or, even worse, reaching at trivial solutions.

This paper focuses on the mode collapse problem in training GANs by adopting multiple collaborating discriminators. Each discriminator is learned to specialize in a subset of reference data space, which is identified automatically via the training procedure, so the ensemble of discriminators provide not only the differentiation of fake data, but also more accurate predictions over the clusters of real data. In this respect, a generator is encouraged to produce diverse modes that deceive a set of discriminators. We employ Multiple Choice Learning (MCL) to learn multiple discriminators that are trained on a subset of training data. The generator is updated via a set of expert models, each of which is associated with a subset of the true and generated examples closest to the expert. Our approach based on a single generator and multiple discriminators is called MCL-GAN, which is optimized by the standard objective of GAN combined with the objective for MCL in the discriminator side.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

There are several GAN literatures that employ multiple discriminators [8–13]. Among them, GMAN [9] is closely related with our approach in the sense that it utilizes the ensemble prediction of discriminators. It explores multi-discriminator extensions of GANs with diverse versions of the aggregated prediction of discriminators—from a harsh trainer to a lenient teacher with a softened criteria. Meanwhile, there exist significant differences in the method of ensembling from our approach. While GMAN focuses on the loss to the generator with parallel learning of discriminators, our strategy specializes each discriminator for more informative feedbacks to the generator.

The training algorithm of the proposed method is inspired by Multiple Choice Learning [14], which is known to be effective in learning specialized models with high oracle accuracy in recognition tasks. Encouraged by this benefit, [15–18] apply MCL or its variations [19, 20] to produce diverse and accurate outputs in several applications. For instance, Mun *et al*. [16] propose MCL-KD framework to come up with the visual question answering (VQA) systems based on multiple models that are specialized in different types of visual reasonings. DiverseNet [17] introduces the control parameter as an input that diversifies the outputs of networks with an MCL loss. While these works generate multiple outputs explicitly and select one of predictions or take their ensemble at inference time, our approach adopts a unique strategy for diversifying the mode with no additional cost at inference time.

The proposed method takes an advantage of MCL techniques for GANs, which has hardly been explored before. No supervision such as class labels or other conditions are assumed unlike the aforementioned works. Our main contributions are summarized as follows:

- We propose a single-generator multi-discriminator GAN training algorithm to alleviate the mode collapse problem. Our approach provides the simple yet effective learning objectives based on MCL to achieve the goal.
- We present a balanced discriminator assignment strategy to facilitate the robust convergence of models and preserve the multi-modality of training data, where the number of the discriminators is determined adaptively.
- The proposed method is applicable to many GAN variants since there is no constraint on network architectures or loss functions. Our method requires a small extra overhead and trains the model with computational efficiency via feature sharing in the discriminators.

2 Related work

This section discusses the mode collapse issue in GANs and the GAN models with multiple discriminators or generators.

2.1 Handling mode collapse for diversity

Many variations of GANs propose either novel metrics for the discriminator loss or better alternatives of the discriminator design. For example, LSGAN [21] substitutes the least square function for the binary cross-entropy function as the discriminator loss. WGAN [22] introduces a critic function based on the Earth-Mover's distance rather than a binary classifier, and WGAN-GP [23] improves WGAN by adding a gradient penalty term. PacGAN [24] augments the discriminator's input by packing samples for a single label. EBGAN [25] models the discriminator as an energy function, which is implemented by the reconstruction loss of the autoencoder. BiGAN [26], ALI [27], VEEGAN [28], Inclusive GAN [29] also learn reconstruction networks. In particular, VEEGAN [28] autoencodes the latent vectors to learn the inverse function of the generator and map both the true and generated data to the latent distribution, *i.e*. a Gaussian. Inclusive GAN [29] learns a generator by matching between real and fake examples in the feature space. The mode collapse and diversity issue of generated outputs has been addressed explicitly in [11, 30, 31]. They formulate the diversity metrics that encourage the mode exploration of the generators and derive the loss function using the metrics.

2.2 GAN with multiple generators

Another line of research is the integration of multiple generators [32–35]. This approach represents the data distribution with a mixture model enforcing each generator to cover a portion of the whole data space. It is naturally expected that mixture models approximate true distributions better than a single model especially in high-dimensional spaces with multiple modes.

MAD-GAN [33] introduce an augmented classifier as a discriminator, which predicts whether the sample is real and which generator the sample is drawn from, to encourage individual generators to learn distinctive modes. MGAN [34] has the similar strategies to MAD-GAN, but constructs a separate branch in the discriminator to perform the two tasks. MEGAN [35] adopts a gating network that produces a one-hot vector to select the generator creating the best example. P2GAN [36] sequentially adds a new generator to cover the missing modes of the real data.

2.3 GAN with multiple discriminators

Multiple discriminators are often employed to improve the performance of a single generator [8–10, 12]. D2GAN [8] conducts a three player minimax game, where two discriminators are trained for the completely opposite objectives, minimizing Kullback-Leibler (KL) divergence and the inverse KL divergence between the true and generated data distributions. The balancing of two losses plays a role for seeking desirable and diverse modes at the same time. Albuquerque *et al*. [12] propose a general multi-objective optimization framework in the scenario with multiple discriminators. They present the hypervolume maximization algorithm to obtain weighted gradients. Neyshabur *et al*. [13] train a GAN based on multiple projections. Each discriminator makes a decision for the random low-dimensional projection of a sample to address the instability of GAN training in high-dimensions. GMAN [9] presents diverse aggregation methods of multiple discriminators, where both hard and soft discriminator selection strategies are studied. Note that all the existing approaches learn the multiple discriminators independently and they may have strong correlations, which may not be appropriate for diversifying the generated samples. The proposed approach, however, assigns each sample to the best-suited discriminator through the interactions among the discriminators, and, consequently, each discriminator becomes the expert model for the assigned examples.

3 Multiple Choice Learning

We present the main idea of MCL [37] and its extensions briefly. Given a training dataset with N samples, $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, M models, $\{f_m\}_{m=1}^M$ and a task-specific loss function, $\ell(\cdot,\cdot)$, MCL minimizes the following oracle loss:

$$\mathcal{L}_{\text{MCL}}(\mathcal{D}) = \sum_{i=1}^{N} \min_m \ell(y_i, f_m(\mathbf{x}_i)). \tag{1}$$

In other words, only the model with the smallest error out of M candidates is selected for each example. This optimization process makes each model f_m become an expert for a subset of $\mathcal{D}$, thus leads to forming a natural cluster in $\mathcal{D}$.

A weakness of MCL is the possible mistakes caused by the overconfidence issues. If non-specialized models make wrong predictions with high confidences in the score aggregation process, the average scores are misleading and the ensemble model may result in poor quality outputs. To alleviate the limitation, Confident Multiple Choice Learning (CMCL) [19] adopts a confident oracle loss that enforces the predictions of a non-specialized model to be uniformly distributed using KL divergence, denoted by D_{KL}. Assuming that f_m predicts the output distribution given data point x, *i.e.*, $P_m(y|x)$, the loss is modified as

$$\mathcal{L}_{\text{CMCL}}(\mathcal{D}) = \sum_{i=1}^{N} \sum_{m=1}^{M} v_{i,m} \ell(y_i, P_m(y|\mathbf{x}_i)) + \beta(1 - v_{i,m}) D_{\text{KL}}(\mathcal{U}(y) \| P_m(y|\mathbf{x}_i)), \tag{2}$$

where $\mathcal{U}(y)$ is the uniform distribution and the flag variable $v_{i,m} \in \{0, 1\}$ allows the choices of the specialized models. Note that, if $\sum_{m=1}^{M} v_{i,m} = k$ ($k < M$), each example is assigned to k models.

4 MCL-GAN

We describe our GAN structure with a generator $G(\cdot;\theta)$ and M discriminators $\{D_m(\cdot;\phi_m)\}_{m=1}^M$ extended from the standard GAN. Let p_z and p_d be the distributions of the latent space and real data space, respectively. Given $\mathbf{z} \sim p_z$, the generator produces a sample $\tilde{\mathbf{x}} = G(z;\theta)$ and M predictions are made by the discriminators for each real example $\mathbf{x} \sim p_d$ and fake sample $\tilde{\mathbf{x}}$. Each prediction, $D_m(\mathbf{x};\phi_m)$, ranges in $[0, 1]$ and represents the probability that $\mathbf{x}$ belongs to the true data distribution.

4.1 Expert training

Assuming that we draw N_d real data and generate N_g examples in each training batch, denoted by $\mathbf{x}$ and $\tilde{\mathbf{x}}$, respectively, each network is trained as follows.

Discriminators Expert discriminators are the ones that predict the highest scores for an example. With the indicator variable $v_{i,m}$ for $\mathbf{x}_i$, the discriminators are trained to minimize the following loss function:

$$\mathcal{L}_{\text{e}}(\mathbf{x}) = -\sum_{i=1}^{N_d}\sum_{m=1}^{M} v_{i,m}\log(D_m(\mathbf{x}_i;\phi_m)), \tag{3}$$

where we choose k experts out of M discriminators for each example, *i.e.*, $\sum_{m=1}^{M} v_{i,m} = k$. For a fake sample, since all discriminators have to identify it correctly, the following loss is added to (3):

$$\mathcal{L}_{\text{e}}(\tilde{\mathbf{x}}) = -\sum_{j=1}^{N_g}\sum_{m=1}^{M} \log(1 - D_m(G(\mathbf{z}_j);\phi_m)). \tag{4}$$

Generator We train the generator using the gradients received from the expert models to encourage the generator to find the closest mode given $\mathbf{z}$. With another indicator variable $u_{j,m}$ for $\mathbf{z}_j$, the expert loss for the generator is given by

$$\mathcal{L}_{\text{e}}(\tilde{\mathbf{x}}) = \sum_{j=1}^{N_g}\sum_{m=1}^{M} u_{j,m}\log(1 - D_m(G(\mathbf{z}_j;\theta));\phi_m) \text{ and } \sum_{m=1}^{M} u_{j,m} = k. \tag{5}$$

4.2 Non-expert training

The non-expert discriminators should not be over-confident to real example while it is desirable to produce higher scores for real samples than fake ones. For this requirement, we give a uniform soft label, *e.g.*, $y = [0.5, 0.5]$ for non-expert discriminators and regularize them with some weight. To be precise, we obtain the following non-expert loss term corresponding to (3):

$$\mathcal{L}_{\text{ne}}(\mathbf{x}) = \sum_{i=1}^{N_d}\sum_{m=1}^{M} (1 - v_{i,m})\ell_{\text{ce}}(D_m(\mathbf{x}_i), y), \tag{6}$$

with the same $v_{i,m}$ defined in (3) and $\ell_{\text{ce}}(\cdot, y)$ is the cross-entropy loss function given a target label y. The other counterpart for (5) is derived similarly as

$$\mathcal{L}_{\text{ne}}(\tilde{\mathbf{x}}) = \sum_{j=1}^{N_g}\sum_{m=1}^{M} (1 - u_{j,m})\ell_{\text{ce}}(D_m(G(\mathbf{z}_j)), y). \tag{7}$$

The non-expert model training is effective to handle the overconfidence issue, but the model may still suffer from the data deficiency problem of the standard MCL framework because each discriminator can see only a subset of the whole dataset. To ameliorate this limitation, our discriminators share the parameters of all layers for feature extraction while branching the last layer only. This implementation is also sensible in that the discriminators partially have the same objective to distinguish the fake examples. The common representations of all real samples are likely to be learned in the earlier layers despite being clustered in the different subsets whereas the critical information for the high-level classification is often found in the last layer. Moreover, the number of training parameters and training time are saved significantly while taking advantage of ensemble learning.

4.3 Balanced assignment of discriminators

On top of the adversarial losses, we introduce another loss for balanced updates of discriminators. As there is no supervision for the specialized factor for certain discriminator, *e.g.*, class labels or feature embeddings, it may be difficult to reasonably distribute real samples to expert models from the beginning. Since the abilities of individual discriminators are severely off-balanced, they are

highly prone to assign all samples to few specific models. Especially at an early phase of training, the model's capability is more sensitive to the number of updates in the discriminators.

To tackle this challenge, we propose another loss, called the balance loss, which gives discriminators balanced chances for updates. Let the selection of expert discriminators approximately follow a categorical distribution with a parameter $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_M]$. Then the loss is computed by the KL divergence of the probability distribution of discriminators for being selected as experts from $\boldsymbol{\mu}$. To obtain the probability for discriminator selection, we apply the `softmax` function to the vector of M predictions from discriminators for each example since the discriminator with the highest score is guaranteed to be chosen as an expert. We average these probability vectors over the training batch. *i.e.*, $\mathbf{q} = \frac{1}{N_d}\sum_{i=1}^{N_d} \mathtt{s}([D_1(\mathbf{x}_i), \ldots, D_M(\mathbf{x}_i)]; \tau)$, where $\mathtt{s}(\cdot;\tau)$ denotes a vector-valued `softmax` function with temperature τ given an input vector. To sum up, the balance loss is given by

$$\mathcal{L}_{\text{bal}}(\mathbf{x}) = D_{\text{KL}}(\boldsymbol{\mu} \| \mathbf{q}). \tag{8}$$

In practice, we set $\mu_m = \frac{1}{M}, \forall m$ to update the discriminators evenly, which is because the true distribution is unavailable. This assumption may not be congruent to the real distribution of the dataset and excessively forced assignment would not result in an optimal clustering for specialization. We, therefore, decrease the weight for the balance loss gradually during training. Eventually, each example will be naturally assigned to its best model with a very small weight of the balance loss. This adjustment helps stabilize training and naturally cluster the reference data. Note that the models are balanced within a few epochs and the weight reduction helps generate higher quality samples.

Likewise, a small enforcement on the distribution of the generator's output facilitates balanced generation when the statistics of generated samples are skewed. For this case, we use the distribution of the discriminators' assignments instead of arbitrarily chosen $\boldsymbol{\mu}$, *i.e.*,

$$\mathcal{L}_{\text{bal}}(\tilde{\mathbf{x}}) = D_{\text{KL}}(\mathbf{q} \| \mathbf{o}). \tag{9}$$

where $\mathbf{o} = \frac{1}{N_g}\sum_{j=1}^{N_g} \mathtt{s}([D_1(G(\mathbf{z}_j)), \ldots, D_M(G(\mathbf{z}_j))]; \tau)$.

4.4 Total loss

Altogether, the total loss is summarized as follows:

$$\mathcal{L} = \mathcal{L}_{\text{e}} + \alpha\mathcal{L}_{\text{ne}} + \beta\mathcal{L}_{\text{bal}}, \tag{10}$$

where β is different for discriminators and generator. Note that the proposed approach is applicable to any GAN formulations with the corresponding adversarial losses.

4.5 Choice of number of discriminators

A remaining concern is how to find the optimal number of discriminators while such information is not available in general as in many clustering tasks. If the number of discriminators is much larger than the optimal one, it is more desirable to focus on training a subset of discriminators than dividing the dataset into many minor clusters forcefully.

To ease this issue, we can augment ℓ_1 regularization loss to the discriminator with weight γ and encourage the sparsity in the discriminator selection for more desirable clustering results. Hence, even in the case that we are given an excessively large number of discriminators, our algorithm converges at good points by using a small number of discriminators in practice. It is true that this strategy may not always lead to the optimal number of discriminators and has conflict with the balance loss in (8). However, the balance loss fades away as training goes, and our model identifies a proper number of clusters by deactivating a subset of discriminators. This sparsity loss may be useful when we learn on the examples drawn from unknown distributions.

5 Experiments

We evaluate the performance of MCL-GAN on unconditional and conditional image generation. Note that the asterisk (*) denotes the copied results from other papers throughout this section.

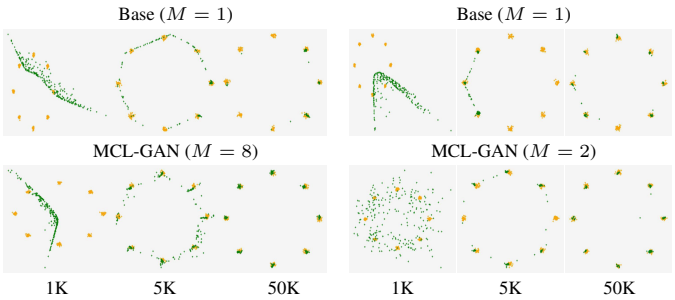

Figure 1: Snapshots of 256 random samples drawn from the generators of the baseline and MCL-GAN with (left) the standard GAN loss and (right) the Hinge loss after 1K, 5K, and 50K iterations. Data sampled from the true distribution are in orange while the generated ones are in green.

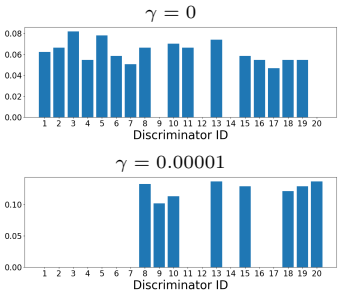

Figure 2: Effect of the ℓ_1 loss weight in MCL-GAN (γ). The graphs show the ratio of training examples associated with each discriminator.

5.1 Synthetic dataset

We first perform toy experiments to verify the main idea of MCL-GAN intuitively. We consider a mixture of 8 2D isotropic Gaussians whose centers are located on the circumference of a circle with a radius of $\sqrt{2}$ while their standard deviation in each dimension is 0.05. We employ 8 discriminators for training with the standard GAN loss while utilizing 2 discriminators with the Hinge loss [38]. We choose one expert discriminator for each sample ($k = 1$) in all experiments.

Figure 1 illustrates the snapshots of generated examples through iterations by the baselines and MCL-GANs. Unlike the base models ($M = 1$) that fail to cover all 8 modes, MCL-GANs learn to identify diverse modes quickly and produce the samples at all modes eventually. Note that the Hinge loss tends to generate more diverse examples even with a single discriminator and MCL-GAN also requires less discriminators in the Hinge loss case to reconstruct the original data distribution. Figure 2 shows how MCL-GANs adaptively select discriminators, given an excessive number of discriminators, *e.g.*, $M = 20$. They mostly utilize 8 or 16 expert discriminators when $\gamma = 0$ and 10^{-5}, respectively; it turns out that each mode is associated with 1 or 2 discriminators depending on the value of γ. We visualizes the detailed mapping between examples and discriminators in the supplementary document. This observation implies that MCL-GAN covers all modes effectively while the ℓ_1 loss helps identify the proper number of discriminators to generate high-fidelity data.

5.2 Unconditional GAN on image dataset

5.2.1 Experiment setup and evaluation protocol

We run the unconditional GAN experiment on four distinct datasets including MNIST [39], Fashion-MNIST [40], CIFAR-10 [41] and CelebA [42], where two types of network architectures are employed—DCGAN [43] and StyleGAN2 [44]. The images are resized to 32×32 except for CelebA dataset, for which DCGAN and StyleGAN2 adopt 64×64 and 128×128 images, respectively. For the StyleGAN2 experiments on CelebA, we use the first and the last 30K images from the "align&cropped" version for the train and validation splits following [29]. We apply our method to the DCGAN architecture with three different GAN loss functions: the vanilla GAN loss [1], LSGAN [21] and Hinge loss [38]. The supplementary document describes more details of our setting.

We adopt Precision Recall Distribution (PRD) [45] and Frèchet Inception Distance (FID) [46] as our evaluation metrics. The PRD curve provides more credible assessment on generative models than single-valued metrics since it quantifies the model in two folds—mode coverage (recall) and quality of generated samples (precision). We measure the recall and precision of each model by computing the F_8 and $F_{1/8}$ scores from the PRD curve, respectively. We prefer higher scores and they are equal to 1 when the generated data distribution is identical to the reference. FID is a popular evaluation metric of generative models, based on the distance between two datasets with multivariate Gaussian assumption. Lower FID scores mean that generated samples are closer to the reference data.

Table 1: Precision and recall scores from PRD curves on MNIST, Fashion-MNIST and CelebA datasets with the DCGAN architecture. Note that GMAN fails to converge with the Hinge loss.

Method	Loss	M	MNIST		Fashion-MNIST		CelebA		
			Rec.↑	Prec.↑	Rec.↑	Prec.↑	FID ↓	Rec.↑	Prec.↑
Base (DCGAN) [43]	GAN	1	0.896	0.778	0.936	0.900	30.93	0.834	0.839
GMAN [9]		5	0.968	0.976	0.909	**0.955**	31.66	0.888	0.873
GMAN [9]		10	0.964	**0.977**	0.928	0.946	22.45	0.921	0.923
MCL-GAN		5	**0.985**	**0.977**	**0.977**	0.929	**16.88**	**0.955**	**0.957**
MCL-GAN		10	0.976	0.975	0.964	0.914	21.18	0.940	0.938
Base (DCGAN) [43]	LSGAN [21]	1	0.977	0.957	0.928	0.866	19.87	0.923	0.943
GMAN [9]		10	0.966	0.973	0.953	**0.952**	22.72	0.934	0.906
MCL-GAN		10	**0.983**	**0.980**	0.**963**	0.911	**17.81**	**0.950**	**0.952**
Base (DCGAN) [43]	Hinge [38]	1	0.790	0.785	0.936	0.853	23.56	0.905	0.883
MCL-GAN		5	0.957	0.965	**0.959**	**0.916**	**20.49**	0.914	0.925
MCL-GAN		10	**0.978**	**0.968**	0.949	0.885	21.23	**0.928**	**0.931**

Table 2: FID scores on CIFAR-10 with the DCGAN architecture.

Method	# Disc.	# Gen.	FID ↓	Remark
Base (DCGAN)* [43]	1	1	37.7	
GMAN [9]	10	1	37.11	
Albuquerque *et al*. [12]	10	1	30.26	
MCL-GAN	10	1	**26.87**	
MGAN* [34]	1	10	26.7	Requires multiple generators for inference
MSGAN* [11]	1	1	28.73	Requires class labels for training

Table 3: KL divergence and LPIPS using pretrained classifiers.

Method	M	MNIST		Fashion-MNIST		CIFAR-10	
		KL ↓	LPIPS↑	KL ↓	LPIPS ↑	KL ↓	LPIPS ↑
Base (DCGAN) [43]	1	0.0268	0.0257	0.0437	0.0233	0.0532	0.0562
GMAN [9]	5	**0.0072**	0.0238	0.0587	0.0289	0.1084	**0.0594**
MCL-GAN	5	0.0127	**0.0259**	**0.0194**	**0.0294**	**0.0474**	0.0562

5.2.2 DCGAN backbone

Table 1 summarizes the precision and recall scores of our methods compared to the baseline models with three different GAN objectives, when the number of discriminators is set to 5 or 10 ($M = 5$ or 10) and the number of experts is 1 ($k = 1$). MCL-GAN achieves outstanding performance in terms of both metrics compared to the baseline and GMAN, which is an existing approach based on multiple discriminators, on MNIST and CelebA. For Fashion-MNIST, we observe that MCL-GAN focuses on the mode coverage (diversity) while GMAN cares about the image quality rather than the diversity.

MCL-GAN outperforms all the compared GAN models based on DCGAN by large margins except MGAN and MSGAN in terms of the FID scores on CIFAR-10, as presented in Table 2. Note that MGAN relies on multiple generators, 10 in this case, and MSGAN requires class labels for training. The results imply that MCL-GAN is effective to maintain the multi-modality in the underlying distribution with relatively small memory footprint and without extra supervision.

To analyze the semantic quality of generated images, we present their classification results given by the pretrained classifiers in Table 3. We measure how much the predicted label distribution in each tested dataset deviates from the true (uniform) distribution using the KL divergence. We also calculate the average of class-wise LPIPS to measure the intra-class diversity of generated images. The overall results are favorable to MCL-GAN but there exist some misleading points. For example, GMAN tends to achieve overly high class-wise LPIPS scores due to many out-of-distribution examples that confuse the classifier, as illustrated in the supplementary document.

5.2.3 StyleGAN2 backbone

Table 4 presents that MCL-GAN is also effective in the state-of-the-art backbone model and outperforms StyleGAN2 and its variation, Inclusive GAN [29], in terms of all metrics. Note that Inclusive GAN uses the sample-wise reconstruction loss by regarding each image as a mode. This strategy appears to improve recall but the model may suffer from sampling bias and scalability issue by estimating overly complex distributions. The results imply that MCL-GAN enhances the convergence and the reproducibility of large GAN models, practically leading to improved performance.

(a) MNIST (DCGAN) (b) Fashion-MNIST (DCGAN) (c) CIFAR-10 (StyleGAN2)

Figure 3: Generated image clusters by MCL-GAN. Each row represents the cluster associated with each discriminator. Note that the images in the same row often have similar shapes and semantics but are not necessarily the same class.

Table 4: FID, precision and recall scores on CIFAR-10 and CelebA datasets with the StyleGAN2 architecture, where 10 and 5 discriminators are adopted, respectively, while $k = 1$.

Method	CIFAR-10			CelebA30K*					
	FID↓	Rec.↑	Prec.↑	FID↓		Rec.↑		Prec.↑	
	-	-	-	Train	Val	Train	Val	Train	Val
Base (StyleGAN2) [44]	9.06	0.979	0.984	9.37	9.49	0.730	0.741	0.855	0.844
Inclusive GAN [29]	-	-	-	11.56	11.28	0.849	0.848	0.927	0.941
MCL-GAN	**7.13**	**0.985**	**0.989**	**8.41**	**8.61**	**0.988**	**0.990**	**0.985**	**0.983**

5.2.4 Discriminators specialization

Figure 3 qualitatively presents how successfully the discriminators in MCL-GAN are specialized to subsets of datasets. We learn the model with 10 discriminators, and illustrate the generated images with their membership to discriminators; the images in the same row belong to the same discriminator. We observe semantic consistency of images within the same row in both MNIST and Fashion-MNIST clearly. We further analyze discriminators specialization using attribute annotations of CelebA in the supplementary document.

5.3 Conditioned image synthesis

We apply MCL-GAN to image-to-image translation and text-to-image synthesis tasks, which require complex architectures to generate high-resolution images. In this experiment, we adopt MSGAN [11], a technique with a mode-seeking regularizer, as an additional component which alleviates the mode collapse in conditional GANs. Then, we observe whether the mode seeking and our multiple choice learning create synergy, based on FID, NDB/JSD [47], and LPIPS [48] following [11]. The definitions and properties of these metrics are discussed in the supplementary file.

Image-to-image translation We choose DRIT [4, 49], an unpaired image-to-image translation method based on the cycle consistency, as our baseline. We employ MCL-GAN with $M = 3$ and $k = 1$ for distinguishing real and translated images. As shown in Table 5, MCL-GAN significantly improves the diversity measure, LPIPS, while achieving high-fidelity data generation performance in terms of other metrics. In particular, our approach works better on a more challenging task, cat $\rightleftharpoons$ dog, since it effectively handles the changes in both object shape and texture across domains by specializing discriminators to a subset of modes. Note that the integration of MCL is mostly beneficial and the addition of the mode-seeking module often leads to extra performance gains.

Text-to-image synthesis This experiment is based on StackGAN++ [50] trained on CUB-200-2011 [51] with a mode-seeking regularizer. StackGAN++ has a hierarchical structure with a specialized pair of a discriminator and a generator to a certain image resolution. We adopt its 3-stage version and trains an MCL-GAN with $M = 3$ and $k = 1$ only at the last stage, which handles images with size 256×256. Table 6 illustrates that the integration of MCL improves performance consistently, especially in terms of the diversity measure, LPIPS, where we observe the merit of the mode-seeking regularizer via the combination with MCL.

Table 5: Quantitative results on the Yosemitee (Summer $\rightleftharpoons$ Winter) [5] and Cat $\rightleftharpoons$ Dog [4] dataset.

Dataset	Metric	DRIT*	+MS (DRIT++)* [11]	+MCL	+MCL+MS
Summer → Winter	FID ↓	57.24 ± 2.03	51.85 ± 1.16	53.77 ± 1.36	**49.74 ± 2.74**
	NDB ↓	25.60 ± 1.14	**22.80 ± 2.96**	25.40 ± 1.14	30.00 ± 2.55
	JSD ↓	0.066 ± 0.005	0.046 ± 0.006	**0.036 ± 0.004**	0.044 ± 0.005
	LPIPS ↑	0.115 ± 0.000	0.147 ± 0.001	0.199 ± 0.002	**0.263 ± 0.003**
Winter → Summer	FID ↓	47.37 ± 3.25	46.23 ± 2.45	49.41 ± 1.29	**41.94 ± 1.43**
	NDB ↓	30.60 ± 2.97	27.80 ± 3.03	**23.40 ± 1.52**	24.20 ± 3.27
	JSD ↓	0.049 ± 0.009	0.038 ± 0.004	0.033 ± 0.002	**0.030 ± 0.005**
	LPIPS ↑	0.097 ± 0.000	0.118 ± 0.001	0.153 ± 0.001	**0.248 ± 0.001**
Cat → Dog	FID ↓	22.74 ± 0.28	16.02 ± 0.30	20.64 ± 0.13	**15.36 ± 0.16**
	NDB ↓	42.00 ± 2.12	27.20 ± 0.84	29.80 ± 1.10	**22.20 ± 2.77**
	JSD ↓	0.127 ± 0.003	0.084 ± 0.002	0.048 ± 0.002	**0.031 ± 0.002**
	LPIPS ↑	0.245 ± 0.002	0.280 ± 0.002	0.511 ± 0.000	**0.553 ± 0.000**
Dog → Cat	FID ↓	62.85 ± 0.21	29.57 ± 0.23	**20.61 ± 0.05**	27.16 ± 0.20
	NDB ↓	41.00 ± 0.71	31.00 ± 0.71	**16.40 ± 0.89**	20.20 ± 1.48
	JSD ↓	0.272 ± 0.002	0.068 ± 0.001	**0.024 ± 0.001**	0.031 ± 0.001
	LPIPS ↑	0.102 ± 0.001	0.214 ± 0.001	0.429 ± 0.001	**0.482 ± 0.000**

Table 6: Quantitative results on CUB-200-2011.

Dataset	Metric	StackGAN++*	+MS* [11]	+MCL	+MCL+MS
CUB-200-2011	FID ↓	25.99±4.26	25.53±1.83	**22.91±0.80**	25.44±0.41
	NDB ↓	38.20±2.39	30.60±2.51	28.80±3.63	**23.20±3.03**
	JSD ↓	0.09±0.01	0.07±0.00	0.08±0.00	**0.05±0.00**
	LPIPS ↑	0.36±0.00	0.37±0.01	**0.63±0.00**	0.62±0.00

Table 7: Comparison with other strategies of discriminator assignment such as Minimum, Random, and GT-Assign. The model denoted by GT-Assign links an expert discriminator with a real sample using the ground-truth class label under our multi-discriminator framework. This option is unrealistic due to the requirement of the ground-truth class labels.

Strategy	M	k	MNIST		Fashion-MNIST		CelebA	
			Rec.↑	Prec.↑	Rec.↑	Prec.↑	Rec.↑	Prec.↑
Base (DCGAN) [43]	1	1	0.896	0.778	0.936	0.900	0.834	0.839
Minimum	5	1	0.913	0.904	0.943	0.906	0.945	0.893
Random	5	1	0.971	0.954	0.930	0.917	0.930	0.946
MCL-GAN	5	1	**0.985**	**0.977**	**0.977**	0.929	**0.955**	**0.957**
GT-Assign	10	1	0.978	0.966	0.969	**0.935**	-	-

5.4 Discussion

MCL-GAN is a model-agnostic ensemble algorithm with multiple discriminators. Our experiments show that the specialized discriminators to the subsets of training data are helpful compared to independent training of the discriminators on the whole dataset (as in GMAN) or different strategies of discriminator assignments, *i.e.*, minimum-score discriminator selection, which is an opposite criterion to MCL-GAN, or random selection. Also, the performance of MCL-GAN is as competitive as the method assigning discriminators based on the ground-truth class labels, which exhibits the reliability of the discriminator specialization by MCL. Table 7 presents that MCL-GAN achieves outstanding performance compared to other strategies especially for the recall metric, even compared with GT-Assign without relying on class labels for discriminator specialization. The proposed approach is also efficient since it is free from any time-consuming clustering procedure for sample assignment to discriminators during training, and incurs marginal extra cost despite the use of multiple discriminators because the discriminators share the feature extractor.

6 Conclusion

We presented a generative adversarial network framework with multiple discriminators, where each discriminator behaves as an expert classifier and covers a separate mode in the underlying distribution. This idea is implemented by incorporating the concept of multiple choice learning. The combination of generative adversarial network and multiple choice learning turns out to be effective to alleviate the mode collapse problem. Also, the integration of the sparsity loss encourages our model to identify the proper number of discriminators and estimate a desirable distribution. We demonstrated the effectiveness of the proposed algorithm on various GAN models and datasets.

References

[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In NeurIPS. (2014) 2672–2680

[2] Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In ICLR. (2014)

[3] Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In ICCV. (2017) 2223–2232

[4] Lee, H.Y., Tseng, H.Y., Huang, J.B., Singh, M.K., Yang, M.H.: Diverse image-to-image translation via disentangled representations. In ECCV. (2018)

[5] Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In ICCV. (2017)

[6] Yeh, R.A., Chen, C., Yian Lim, T., Schwing, A.G., Hasegawa-Johnson, M., Do, M.N.: Semantic image inpainting with deep generative models. In CVPR. (2017) 5485–5493

[7] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In CVPR. (2017) 4681–4690

[8] Nguyen, T., Le, T., Vu, H., Phung, D.: Dual discriminator generative adversarial nets. In NeurIPS. (2017) 2670–2680

[9] Durugkar, I., Gemp, I., Mahadevan, S.: Generative multi-adversarial networks. (2017)

[10] Doan, T., Monteiro, J., Albuquerque, I., Mazoure, B., Durand, A., Pineau, J., Hjelm, R.D.: On-line adaptative curriculum learning for gans. In Proceedings of the AAAI Conference on Artificial Intelligence. Volume 33. (2019) 3470–3477

[11] Mao, Q., Lee, H.Y., Tseng, H.Y., Ma, S., Yang, M.H.: Mode seeking generative adversarial networks for diverse image synthesis. In CVPR. (2019) 1429–1437

[12] Albuquerque, I., Monteiro, J., Doan, T., Considine, B., Falk, T., Mitliagkas, I.: Multi-objective training of generative adversarial networks with multiple discriminators. In ICML. (2019) 202–211

[13] Neyshabur, B., Bhojanapalli, S., Chakrabarti, A.: Stabilizing gan training with multiple random projections. arXiv preprint arXiv:1705.07831 (2017)

[14] Lee, S., Prakash, S.P.S., Cogswell, M., Ranjan, V., Crandall, D., Batra, D.: Stochastic multiple choice learning for training diverse deep ensembles. In NeurIPS. (2016) 2119–2127

[15] Chen, Q., Koltun, V.: Photographic image synthesis with cascaded refinement networks. In ICCV. (2017) 1511–1520

[16] Mun, J., Lee, K., Shin, J., Han, B.: Learning to specialize with knowledge distillation for visual question answering. In NeurIPS. (2018) 8081–8091

[17] Firman, M., Campbell, N.D., Agapito, L., Brostow, G.J.: Diversenet: When one right answer is not enough. In CVPR. (2018) 5598–5607

[18] Li, K., Zhang, T., Malik, J.: Diverse image synthesis from semantic layouts via conditional imle. In ICCV. (2019) 4220–4229

[19] Lee, K., Hwang, C., Park, K.S., Shin, J.: Confident multiple choice learning. In ICML. (2017) 2014–2023

[20] Tian, K., Xu, Y., Zhou, S., Guan, J.: Versatile multiple choice learning and its application to vision computing. In CVPR. (2019) 6349–6357

[21] Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z., Paul Smolley, S.: Least squares generative adversarial networks. In ICCV. (2017) 2794–2802

[22] Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. Volume 70 of PMLR., International Convention Centre, Sydney, Australia (06–11 Aug 2017) 214–223

[23] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In NeurIPS. (2017) 5767–5777

[24] Lin, Z., Khetan, A., Fanti, G., Oh, S.: Pacgan: The power of two samples in generative adversarial networks. NeurIPS (2018)

[25] Zhao, J., Mathieu, M., LeCun, Y.: Energy-based generative adversarial networks. In ICLR. (2017)

[26] Donahue, J., Krähenbühl, P., Darrell, T.: Adversarial feature learning. (2017)

[27] Dumoulin, V., Belghazi, I., Poole, B., Mastropietro, O., Lamb, A., Arjovsky, M., Courville, A.: Adversarially learned inference. ICLR (2017)

[28] Srivastava, A., Valkov, L., Russell, C., Gutmann, M.U., Sutton, C.: Veegan: Reducing mode collapse in gans using implicit variational learning. In NeurIPS. (2017) 3308–3318

[29] Yu, N., Li, K., Zhou, P., Malik, J., Davis, L., Fritz, M.: Inclusive gan: Improving data and minority coverage in generative models. In European Conference on Computer Vision, Springer (2020) 377–393

[30] Liu, S., Zhang, X., Wangni, J., Shi, J.: Normalized diversification. In CVPR. (2019) 10306–10315

[31] Yang, D., Hong, S., Jang, Y., Zhao, T., Lee, H.: Diversity-sensitive conditional generative adversarial networks. In ICLR. (2018)

[32] Tolstikhin, I.O., Gelly, S., Bousquet, O., Simon-Gabriel, C.J., Schölkopf, B.: Adagan: Boosting generative models. In NeurIPS. (2017) 5424–5433

[33] Ghosh, A., Kulharia, V., Namboodiri, V.P., Torr, P.H., Dokania, P.K.: Multi-agent diverse generative adversarial networks. In CVPR. (2018) 8513–8521

[34] Hoang, Q., Nguyen, T.D., Le, T., Phung, D.: Mgan: Training generative adversarial nets with multiple generators. In ICLR. (2018)

[35] Park, D.K., Yoo, S., Bahng, H., Choo, J., Park, N.: Megan: Mixture of experts of generative adversarial networks for multimodal image generation. In IJCAI. (2018) 878–884

[36] Trung Le, Q.H., Vu, H., Nguyen, T.D., Bui, H., Phung, D.: Learning generative adversarial networks from multiple data sources. In IJCAI. (2019) 2823–2829

[37] Guzman-Rivera, A., Batra, D., Kohli, P.: Multiple choice learning: Learning to produce multiple structured outputs. In NeurIPS. (2012) 1799–1807

[38] Lim, J.H., Ye, J.C.: Geometric gan. arXiv preprint arXiv:1705.02894 (2017)

[39] LeCun, Y., Cortes, C.: MNIST handwritten digit database. (2010)

[40] Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms (2017)

[41] Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images. (2009)

[42] Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In ICCV. (2015)

[43] Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. In ICLR. (2016)

[44] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In CVPR. (2020) 8110–8119

[45] Sajjadi, M.S.M., Bachem, O., Lučić, M., Bousquet, O., Gelly, S.: Assessing Generative Models via Precision and Recall. In NeurIPS. (2018) 5228–5237

[46] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Klambauer, G., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a nash equilibrium. In NeurIPS. (2017)

[47] Richardson, E., Weiss, Y.: On gans and gmms. arXiv preprint arXiv:1805.12462 (2018)

[48] Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In CVPR. (2018)

[49] Lee, H.Y., Tseng, H.Y., Mao, Q., Huang, J.B., Lu, Y.D., Singh, M.K., Yang, M.H.: Drit++: Diverse image-to-image translation via disentangled representations. IJCV (2020) 1–16

[50] Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., Metaxas, D.N.: Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In ICCV. (2017) 5907–5915

[51] Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology (2011)

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 4.5 and supplementary material.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See supplementary material.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 4.
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 2 and supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Partly. Section 5.3.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See supplementary material
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]