MCL-GAN: Generative Adversarial Networks with Multiple Specialized Discriminators

Jinyoung Choi[1,2] **Bohyung Han**[1,2,3]
[1]ECE, [2]ASRI, [3]IPAI
Seoul National University, Korea
{jin0.choi,bhhan}@snu.ac.kr

Abstract

We propose a framework of generative adversarial networks with multiple discriminators, which collaborate to represent a real dataset more effectively. Our 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 a subset of the entire data and allow the generator to find reasonable correspondences between the latent and real data spaces automatically without extra supervision for training examples. Despite the use of multiple discriminators, the backbone networks are shared across the discriminators and the increase in training cost is marginal. We demonstrate the effectiveness of our algorithm using multiple evaluation metrics in the standard datasets for diverse tasks.

1 Introduction

With recent advances in deep generative models, Generative Adversarial Networks (GANs) [1] and Variational Autoencoders (VAEs) [2] have shown impressive achievements in generation of high-dimensional realistic images as well as various computer vision applications including image-to-image translation [3–5], image inpainting [6], image super-resolution [7], *etc*. GANs have particularly 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. Furthermore, compared to VAEs, GANs tend to produce acute, high-quality images. In theory, the generator learns the real data distribution by reaching an equilibrium point of the minimax game. 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 trivial solutions.

This paper focuses on the mode collapse problem in training GANs by adopting multiple collaborating discriminators. Each discriminator learns to be specialized in a subset of reference data space, which is identified automatically during training, so the ensemble of discriminators provides 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. To achieve these goals, we employ a Multiple Choice Learning (MCL) framework to learn multiple discriminators and update the generator via a set of expert discriminators, where each discriminator is associated with a subset of the true and generated examples. Our approach, based on a single generator and multiple discriminators, is called MCL-GAN, which is optimized by the standard objective of GANs combined with the objective of MCL on the discriminator side.

There are several GAN literature that adopts multiple discriminators [8–13]. Among them, GMAN [9] is closely related to our approach in the sense that it utilizes an 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 softened criteria. Meanwhile, there

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

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 in each discriminator for more informative feedback to the generator.

The training algorithm of the proposed approach is inspired by MCL [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 the production of diverse and accurate outputs in several applications. For instance, Mun *et al*. [16] propose an MCL-KD framework to come up with a visual question answering (VQA) system based on multiple models that are specialized in different types of visual reasoning. DiverseNet [17] introduces a control parameter as an input that diversifies the outputs of the network with an MCL loss. While these works generate multiple outputs explicitly and select one of them 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 for GANs, which has hardly been explored before. Our approach does not require any supervision for model specialization, such as class labels and other conditions, 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 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 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 small extra overhead and trains models efficiently 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 to the discriminator design. For example, LSGAN [21] substitutes the least square function for the binary cross-entropy loss as the discriminator objective. 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 optimized by the reconstruction loss as in autoencoders. BiGAN [26], ALI [27], VEEGAN [28], and Inclusive GAN [29] also learn reconstruction networks. In particular, VEEGAN [28] autoencodes the latent vectors to learn the inverse function of the generator, and maps both the true and generated data to the latent distribution, *e.g.*, a Gaussian distribution. Inclusive GAN [29] learns a generator by matching between real and fake examples in the feature space. The mode collapse and diversity issues of generated outputs have 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 a 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] introduces an augmented classifier as a discriminator, which predicts whether a sample is real and which generator the sample is drawn from, to encourage individual generators to

learn distinctive modes. MGAN [34] has similar strategies to MAD-GAN but constructs a separate branch in its 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 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 the Kullback-Leibler (KL) divergence and the inverse KL divergence between the true and generated data distributions. The balancing of the two losses plays a role in seeking desirable and diverse modes. Albuquerque *et al*. [12] propose a general multi-objective optimization framework in the scenario with multiple discriminators. They present a hypervolume maximization algorithm to obtain weighted gradients. Neyshabur *et al*. [13] train a GAN based on multiple projections. Each discriminator determines a 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 generated samples. Our approach, however, assigns each sample to the best-suited discriminator through interactions among the discriminators, and, consequently, each discriminator becomes an 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 $f_m(\cdot)$ 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 overconfidence issues. If non-specialized models make wrong predictions with high confidence in the score aggregation process, the average scores are misleading and the ensemble model may result in poor-quality outputs. To alleviate this 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 $f_m(\cdot)$ predicts the output distribution given data point $\mathbf{x}$, *i.e.*, $P_m(y|\mathbf{x})$, 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 extension with a generator $G(\cdot;\theta)$ and M discriminators $\{D_m(\cdot;\phi_m)\}_{m=1}^M$. Let p_z and p_d be the distributions in latent space and real data space, respectively. Given $\mathbf{z} \sim p_z$, the generator produces a sample $\tilde{\mathbf{x}} = G(\mathbf{z};\theta)$ and each of the M discriminators makes a prediction on a real example $\mathbf{x} \sim p_d$ or the 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. We now describe the loss functions $\mathcal{L}^D$ and $\mathcal{L}^G$ used to train the discriminators and the generator, respectively.

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 each example. With the indicator variable $v_{i,m}$ for a real example $\mathbf{x}_i$, the discriminators are trained to minimize the following loss function:

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

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):

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

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

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

4.2 Non-expert training

The non-expert discriminators should not be overconfident to real examples 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 a certain weight. To be precise, we obtain the following non-expert loss term corresponding to (3):

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

where $v_{i,m}$ is the same indicator variable 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

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

The non-expert model training is effective to handle the overconfidence issue, but the model may still suffer from the data deficiency problem frequently happening in the standard MCL algorithms because each discriminator can observe only a subset of the whole dataset. To alleviate this limitation, our discriminators share the parameters of all layers in the feature extractor while branching the last layer only. This implementation is also sensible since the discriminators partially have the same objective to distinguish fake examples. The common representations of all real samples are prone to be learned in the earlier layers despite being clustered in different subsets, whereas the critical information for high-level understanding is often encoded in deeper layers. Moreover, the number of model parameters and training time are saved significantly while still 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 about the specialization target of a 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 a 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 equal 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 a temperature τ given a logit. To sum up, the balance loss is given by

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

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 constraint on the distribution of the generator's output facilitates balanced generation while preventing the statistics of generated samples from being skewed. For this case, we use the distribution of the discriminators' assignments $\mathbf{q}$ instead of arbitrarily chosen $\boldsymbol{\mu}$, *i.e.*,

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

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 losses for discriminators and the generator are summarized respectively as

$$\mathcal{L}^D = \mathcal{L}^D_{\text{e}} + \alpha\mathcal{L}^D_{\text{ne}} + \beta_d\mathcal{L}^D_{\text{bal}} \quad \text{and} \quad \mathcal{L}^G = \mathcal{L}^G_{\text{e}} + \alpha\mathcal{L}^G_{\text{ne}} + \beta_g\mathcal{L}^G_{\text{bal}}, \tag{10}$$

where α and $\{\beta_d, \beta_g\}$ are hyperparameters. 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 augment the ℓ_1 regularization loss to the discriminator side with a 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 subset of discriminators in practice. It is true that this strategy may not always lead to the optimal number of discriminators and have conflicts with the balance loss in (8). However, the balance loss fades away as training goes on, and our model identifies a proper number of clusters by deactivating a fraction of discriminators adaptively. This sparsity loss would be useful when we learn on real datasets, where the examples are 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.

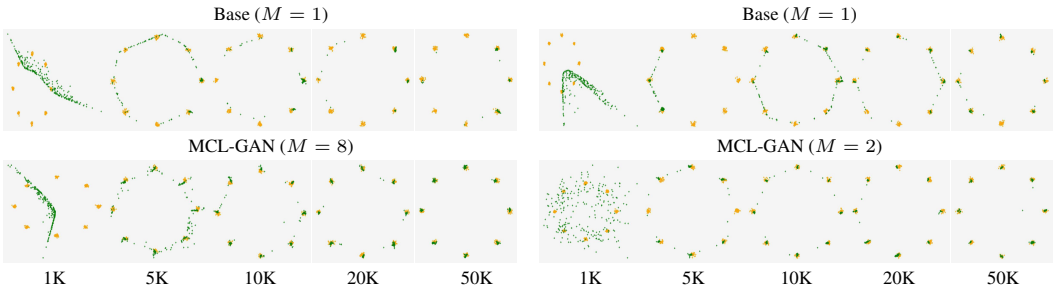


Figure 1: Snapshots of 256 random samples drawn from the generators of the baseline and MCL-GAN with (left) the standard GAN loss and (right) the Hinge loss after 1K, 5K, 10K, 20K and 50K steps. Data sampled from the true distribution are in orange while the generated ones are in green.

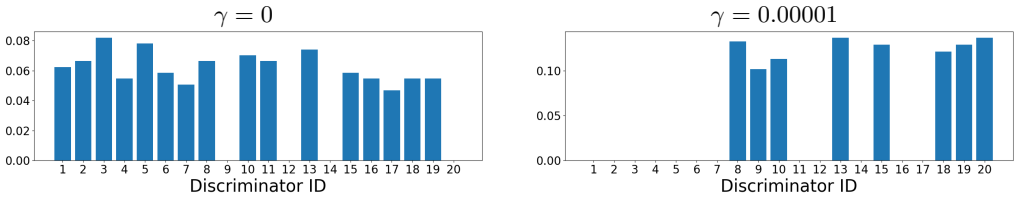


Figure 2: Effect of the ℓ_1 loss weight in MCL-GAN (γ). The graphs show the ratio of training examples associated with each discriminator.

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 visualize the detailed mapping between examples and discriminators in Appendix A. 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]. Appendix E describes more details of our setting.

We adopt Precision Recall Distribution (PRD) [46] and Frèchet Inception Distance (FID) [47] as our evaluation metrics. The PRD curve provides more credible assessment on generative models than

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. † denotes the usage of the different DCGAN architecture, which is adopted in [45].

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
Base (GAN)*†	1	1	31.90	Requires clustering / utilizes 100 clusters
Self-conditioned GAN*† [45]	1	1	18.70	
MCL-GAN†	5	1	**17.15**	

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

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.

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.

As presented in Table 2, MCL-GAN outperforms the compared GAN models (based on DCGAN) by large margins in terms of FID scores on CIFAR-10, while being competitive with MGAN, which relies on multiple generators. MCL-GAN also achieves better results than a clustering-based approach, *i.e*., self-conditioned GAN [45], which requires additional computation and model parameters due to clustering features during training and conditioning the generator's inputs with cluster label embedding. 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

(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**

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 Appendix D.1.

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.

5.2.4 Discriminators specialization

Qualitative results 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.

Analysis using attribute annotations To examine how effective the discriminators represent the real data, we analyze the discriminator specialization statistics of MCL-GAN for individual attributes on CelebA dataset, which has 40 binary attribute annotations per image. We first compute the normalized histogram of discriminator assignments for the images with a certain attribute, and then compute the cosine similarity between the histograms to obtain a 40×40 dimensional similarity matrix, where 40 is the number of attributes. Figure 4 illustrates a submatrix of the similarity matrix, where we choose a subset of attributes for better visualization.

We discover that the discriminator specialization statistics highly coincide with the latent semantic similarities of examples. Note that the four attributes more relevant to female—Arched_Eyebrows, Wearing_Earings, Wearing_Lipstick, and Heavy_Makeup—are clustered together, while another set of four attributes such as Goatee, Male, Mustache, and Sideburns construct a strong group. These results support that the individual discriminators learned by MCL-GAN are specialized to reasonable

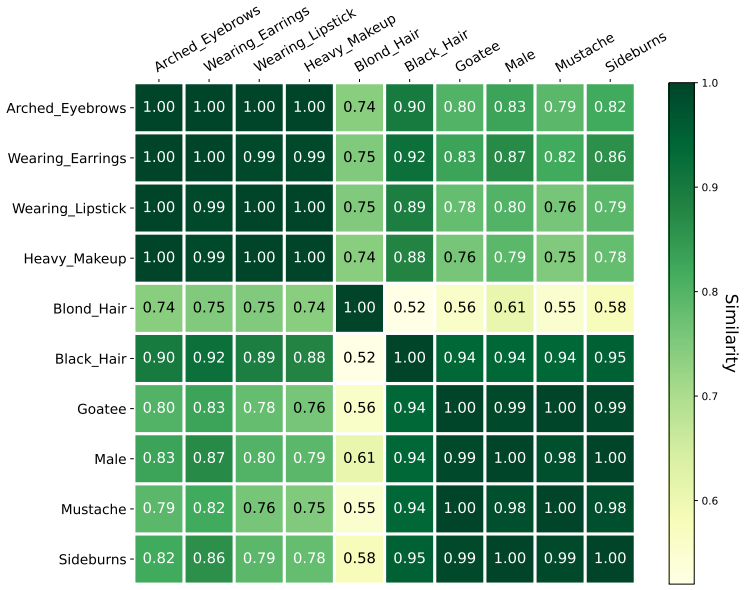


Figure 4: Cosine similarities among attribute representations obtained by the discriminators of MCL-GAN on CelebA dataset.

Table 5: Quantitative results on the Yosemitee (Summer $\rightleftharpoons$ Winter) [3] 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

subsets of the dataset, which is helpful to generate more realistic images congruent to the samples in the latent space.

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 [48], and LPIPS [49] following [11]. Note that NDB counts the number of statistically different bins based on the clusters made by k-means clustering.

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**	-	-

Image-to-image translation We choose DRIT [4, 5], 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.

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.

Acknowledgements This work was partly supported by Samsung Electronics Co., Ltd., and Institute of Information & communications Technology Planning & Evaluation (IITP) grant and the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) [No.2022-0-00959, (Part 2) Few-Shot Learning of Causal Inference in Vision and Language for Decision Making; No.2022R1A5A708390811].

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)

[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)

[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] 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)

[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)

[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)

[8] Nguyen, T., Le, T., Vu, H., Phung, D.: Dual Discriminator Generative Adversarial Nets. In NeurIPS. (2017)

[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 AAAI. (2019)

[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)

[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)

[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)

[15] Chen, Q., Koltun, V.: Photographic Image Synthesis with Cascaded Refinement Networks. In ICCV. (2017)

[16] Mun, J., Lee, K., Shin, J., Han, B.: Learning to Specialize with Knowledge Distillation for Visual Question Answering. In NeurIPS. (2018)

[17] Firman, M., Campbell, N.D., Agapito, L., Brostow, G.J.: DiverseNet: When One Right Answer is not Enough. In CVPR. (2018)

[18] Li, K., Zhang, T., Malik, J.: Diverse Image Synthesis from Semantic Layouts via Conditional IMLE. In ICCV. (2019)

[19] Lee, K., Hwang, C., Park, K.S., Shin, J.: Confident Multiple Choice Learning. In ICML. (2017)

[20] Tian, K., Xu, Y., Zhou, S., Guan, J.: Versatile Multiple Choice Learning and its Application to Vision Computing. In CVPR. (2019)

[21] Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z., Paul Smolley, S.: Least Squares Generative Adversarial Networks. In ICCV. (2017)

[22] Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein Generative Adversarial Networks. In ICML. (2017)

[23] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved Training of Wasserstein GANs. In NeurIPS. (2017)

[24] Lin, Z., Khetan, A., Fanti, G., Oh, S.: PacGAN: The Power of Two Samples in Generative Adversarial Networks. In 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. In ICLR. (2017)

[27] Dumoulin, V., Belghazi, I., Poole, B., Mastropietro, O., Lamb, A., Arjovsky, M., Courville, A.: Adversarially Learned Inference. In 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)

[29] Yu, N., Li, K., Zhou, P., Malik, J., Davis, L., Fritz, M.: Inclusive GAN: Improving Data and Minority Coverage in Generative Models. In ECCV. (2020)

[30] Liu, S., Zhang, X., Wangni, J., Shi, J.: Normalized Diversification. In CVPR. (2019)

[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)

[33] Ghosh, A., Kulharia, V., Namboodiri, V.P., Torr, P.H., Dokania, P.K.: Multi-Agent Diverse Generative Adversarial Networks. In CVPR. (2018)

[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)

[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)

[37] Guzman-Rivera, A., Batra, D., Kohli, P.: Multiple Choice Learning: Learning to Produce Multiple Structured Outputs. In NeurIPS. (2012)

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

[39] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-Based Learning Applied to Document Recognition. IEEE (1998)

[40] Xiao, H., Rasul, K., Vollgraf, R.: Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. arXiv preprint arXiv:1708.07747 (2017)

[41] Krizhevsky, A.: Learning Multiple Layers of Features from Tiny Images. Technical report (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)

[45] Liu, S., Wang, T., Bau, D., Zhu, J.Y., Torralba, A.: Diverse Image Generation via Self-Conditioned GANs. In CVPR. (2020)

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

[47] 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)

[48] Richardson, E., Weiss, Y.: On GANs and GMMs. (2018)

[49] Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In CVPR. (2018)

[50] Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., Metaxas, D.N.: StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks. TPAMI (2018)

[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)

[52] Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. In ICLR. (2015)

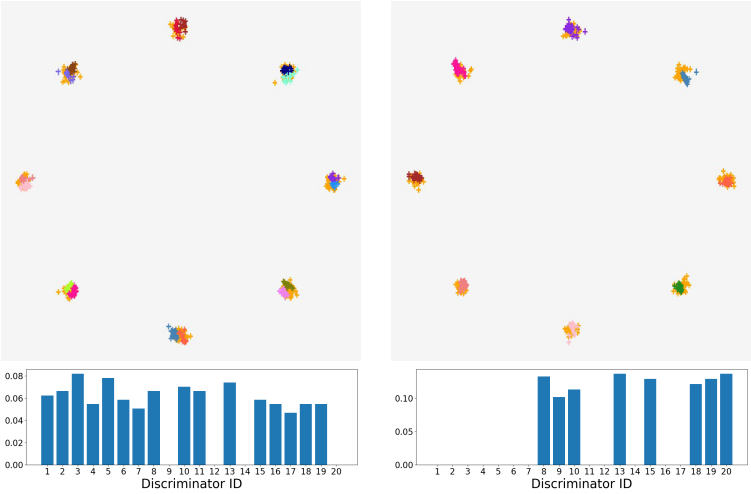
Checklist

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 Appendix G.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Appendix H.
   (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 Appendix E, F and the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 2 and Appendix E.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Partly. See 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 Appendix C.
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]

Appendix

A Mapping between examples and discriminators on synthetic data

Figure 5 illustrates how the generated samples from a synthetic dataset based on a mixture with 8 Gaussians are associated with 20 discriminators in the proposed MCL-GAN framework. Note that this figure is an extended version of Figure 2 in the main paper. The true data are represented in orange while other colors denote the expert discriminator incides for individual examples. Without the ℓ_1 loss ($\gamma = 0$), 16 discriminators are utilized to cluster the true distribution where exactly two discriminators are responsible for each mode; two distinct colors appear in each mode except the one (orange) corresponding to true data. By adding the ℓ_1 loss with a small coefficient, we discover only 8 discriminators are active in training, one for each mode.



(a) $\gamma = 0$ (b) $\gamma = 0.00001$

Figure 5: Visualization of Mapping between examples and discriminators with different ℓ_1 loss weights (γ). Each sample is colored by its expert discriminator, and real data are in orange. The bar graphs show the ratio of training examples associated with each discriminator which are identical to those in Figure 2 of the main paper.

B Analysis on hyperparameters

B.1 Non-expert loss weight

Table 8 shows the effect of non-expert training regularization when training 10 discriminators with the standard GAN loss on MNIST. The overall performance is robust to the variation of the hyperparameter α. The best score is obtained with $\alpha = 0.01$ while all positive α improves the precision scores, which reduces the overconfidence issue of non-expert discriminators.

Table 8: Effect of non-expert loss weight (α) when $M = 10$ and $k = 1$ on MNIST.

α	0	0.01	0.1	0.2	0.5	1
Recall	0.983	0.984	0.978	0.982	0.983	0.976
Precision	0.951	0.977	0.965	0.968	0.974	0.971

B.2 Balance loss weights

Let β_d and β_g be two different versions of β in Eq. (10) of the main paper; the former is for the discriminator and the latter is for the generator. We conduct the ablation studies on several (β_d, β_g) combinations for 10 discriminators with Hinge loss on MNIST. Table 9 shows that β_d plays an

important role in boosting the performance of GAN. This is mainly because β_d is responsible for distributing the chances of being an expert to multiple discriminators; only a few discriminators are utilized in training if β_d is zero or too small. β_g has a smaller effect on the performance than β_d. However, it helps improve recall scores without sacrificing precision and leads to the best score when $(\beta_d, \beta_g) = (0.5, 10)$. Note that the performance does not change drastically for all cases with the positive value of β_d, which surpass a single discriminator GAN by large margins.

Table 9: Effect of the balance loss weights (β_d and β_g) when $M = 10$ and $k = 1$ on MNIST.

β_d	β_g	Recall	Precision
vanilla ($M = 1$)		0.803	0.765
0	0	0.926	0.856
0.2	0	0.973	0.966
0.5	0	0.971	0.970
0	5	0.931	0.894
0.2	5	0.978	0.963
0.5	5	0.977	0.967
0	10	0.949	0.883
0.2	10	0.978	0.966
0.5	10	0.981	0.972

B.3 Number of experts per example

We evaluate our model by varying the number of experts k with $M = 5$ and 10, and present the results on MNIST and CIFAR-10 in Table 10. The number of optimal k may be different in each dataset, however, choosing too many experts tend to drop the recall scores.

Figure 6 shows how the specialization characteristics of discriminators differ by the number of experts per sample, *i.e.*, $k \in \{1, 3, 5\}$, when there are 10 discriminators on MNIST and Fashion-MNIST. As k increases, the models get less specialized by sharing more data each other so the subclusters become less distinctive.

Table 10: Comparisons by number of experts per sample (k).

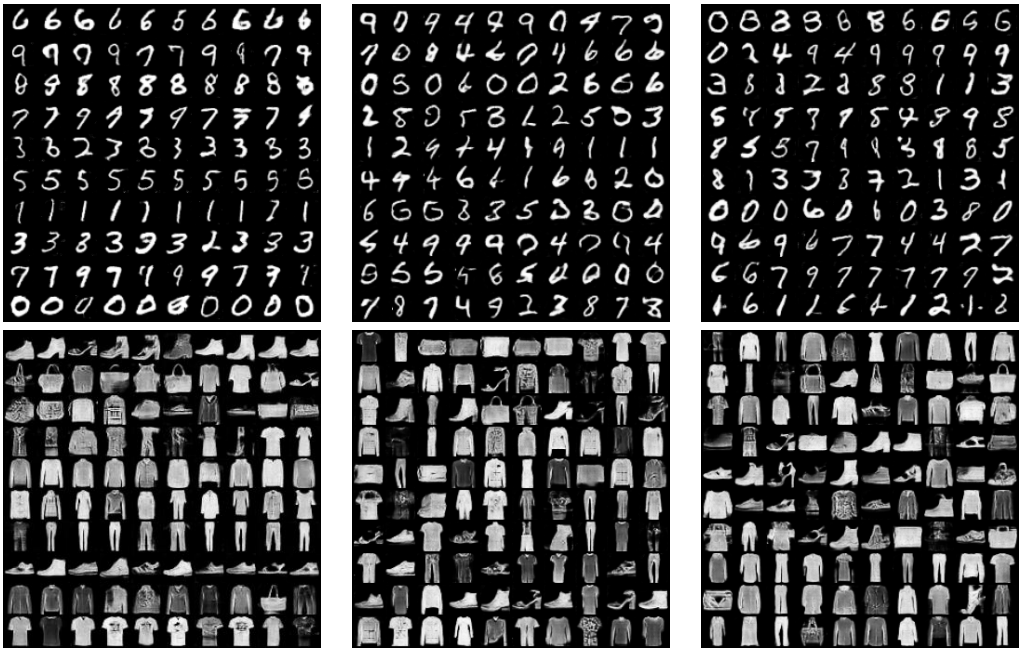
		MNIST		CIFAR-10	
M	k	Recall	Precision	Recall	Precision
5	1	0.983	0.975	0.903	0.942
5	3	0.983	0.981	0.896	0.948
10	1	0.973	0.973	0.902	0.937
10	3	0.973	0.969	0.913	0.946
10	5	0.975	0.964	0.917	0.948
10	7	0.960	0.927	0.912	0.951

B.4 Number of discriminators and ℓ_1 loss weight

We conduct the experiment under a various number of discriminators and illustrate the results in Table 11. It turns out that the performance of the proposed method is fairly robust to the number of discriminators quantitatively and adding the ℓ_1 loss does not incur noticeable differences in terms of precision/recall measure. However, interestingly, the ℓ_1 loss plays a crucial role in finding modes in the underlying distribution. Figure 7 illustrates the impact of the ℓ_1 loss on MNIST and Fashion-MNIST when we train the model with 40 discriminators. According to our results, only a fraction of the discriminators are specialized to data, and the number of active discriminators is coherent to the number of classes in the dataset.

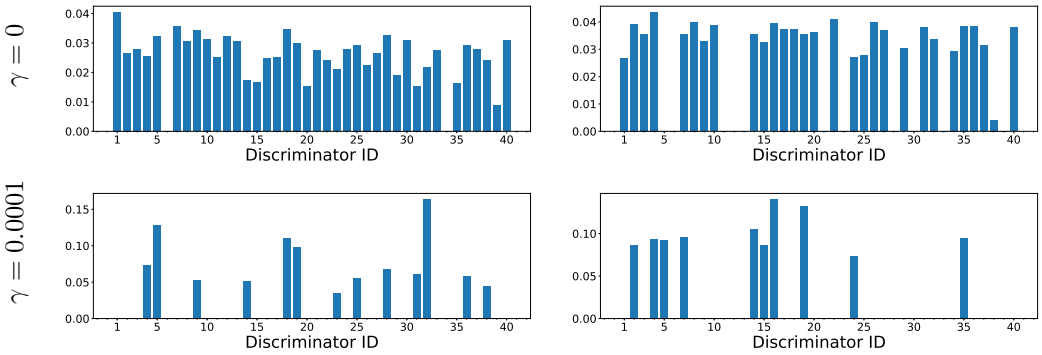
B.5 Stability to hyperparameter change

We conduct experiments on MNIST with $M = 10, 20, 40$ using ℓ_1 regularization multiple times and observe that the accuracies (precision and recall) are very stable regardless of the number of discriminators for expert training as in Table 12. Note that we performed each experiment 4 times with random initialization.



$k = 1$ $k = 3$ $k = 5$

Figure 6: Specialization results for $k \in \{1, 3, 5\}$ with 10 discriminators on MNIST and Fashion-MNIST. The images in each row correspond to the same discriminators.



(a) MNIST (b) Fashion-MNIST

Figure 7: Effect of ℓ_1 loss weight (γ). The update statistics of individual discriminators when 40 discriminators are used for training on MNIST and Fashion-MNIST datasets.

Table 11: Comparisons by number of discriminators (M).

		MNIST		Fashion-MNIST	
M	ℓ_1 loss	Recall	Precision	Recall	Precision
1		0.883	0.795	0.928	0.904
5		0.979	0.976	0.974	0.929
10		0.974	0.972	0.965	0.934
20		0.972	0.958	0.958	0.922
40		0.977	0.970	0.974	0.938
20	✓	0.967	0.964	0.967	0.939
40	✓	0.973	0.960	0.966	0.914

Table 12: Stability of model performances and the number of active discriminators when $M = 10, 20, 40$ and $\gamma = 0.0002$ on MNIST.

M	ℓ_1 loss	Recall	Precision	# active disc.
10	✓	0.965 ± 0.004	0.965 ± 0.006	8.3 ± 1.0
20	✓	0.966 ± 0.004	0.964 ± 0.002	12.0 ± 1.2
40	✓	0.968 ± 0.005	0.963 ± 0.009	10.0 ± 1.4

C Computational overheads

Table 13 compares the training time per iteration and memory usage with the DCGAN [43] and StyleGAN2 [44] backbone when training on CIFAR-10 dataset (32×32 sized) with the batch size of 64. While required resources and training time increases by several folds for GMAN [9] with 5 or 10 discriminators, additional overheads are marginal for MCL-GAN compared to the single-discriminator baseline setting due to feature sharing. Note that the increases in overheads become more negligible in case of MCL-GAN with larger capacity of base architecture such as StyleGAN2 unlike GMAN that suffers from the scalability issue. We used a machine with a Titan Xp GPU for the measurement.

Table 13: Comparisons of computational overheads on CIFAR-10.

		DCGAN		StyleGAN2	
Method	M	Time (s/iteration)	Memory (MiB)	Time (s/iteration)	Memory (MiB)
Base	1	0.0394	1443	0.7478	7971
MCL-GAN	5	0.0408	1443	0.7604	7971
MCL-GAN	10	0.0431	1445	0.7620	7971
GMAN	5	0.2156	2953	-	-
GMAN	10	0.4371	5513	-	-

D Qualitative results

D.1 Unconditional GAN on image datasets

We investigate the quality of images generated by MCL-GAN and compare its performance with GMAN [9] together with real data on MNIST [39], Fashion-MNIST [40], CIFAR-10 [41] and CelebA [42] in Figure 8. The samples are drawn randomly rather than cherry-picked. We observe clear differences between MCL-GAN and GMAN on MNIST and Fashion-MNIST. For MNIST, the generated images by GMAN are sometimes hard to recognize or look artificial (too thin and crisp) compared to real ones. The images in Fashion-MNIST are lacking in diversity; the types of generated bags and shoes are rather simple. Meanwhile, MCL-GAN generates the images that are faithful to the true distribution in semantics and diversity and are indistinguishable from real examples. For CIFAR-10, MCL-GAN generate relatively clear images and some of them are recognizable as vehicles or animals (see Figure 9) whereas most images obtained from GMAN look incomplete and noisy. For CelebA, GMAN produces high quality images, but we discover more distorted and unnatural images than MCL-GAN. Some selected examples from MCL-GAN with DCGAN backbone on Fashion-MNIST, CIFAR-10, and CelebA are displayed in Figure 9. Figure10 shows random samples generated by MCL-GAN with StyleGAN2 backbone on CIFAR-10 and CelebA30K.

D.2 Conditioned image synthesis

Figure 11 and 12 qualitatively compare the diversity of the generated images between the baselines and MCL-GANs. For all methods including the baselines, mode-seeking regularization [11] is applied. As shown in Figure 11(a), images generated by MCL-GAN have more variations in edges and expressions of dogs. Regarding Yosemitee results (Figure 11(b)) which the shapes of the contents are fixed, colors are more diverse and vivid in MCL-GAN results. For Figure 12, we fix the text code for each text description to remove the diversity effect of text embedding and produce images with the same set of latent vectors. MCL-GAN produces more diverse bird images, in terms of shape,

orientation and size with high quality. We present more qualitative results of MCL-GAN in Figure 13 and 14.

E Implementation details

E.1 Synthetic data

We reuse the experimental design and implementation[1] following [23].

E.2 Unconditional GAN on image datasets

DCGAN backbone We mostly follow the training convention proposed in DCGAN [43]. We use Adam optimizer [52] with $\beta = (0.5, 0.999)$ and set 64 and 128 as size of the mini-batch for real data and latent vectors, respectively. We use the same learning rate and temperature in balance loss for all networks, *i.e.*, $lr = 0.0001, \tau = 0.1$ for LSGAN experiments and $lr = 0.0002, \tau = 1.0$ for the others. The weights for balance loss of discriminators, β_d, is chosen in the range $[0.05, 1.0]$ and we choose the best performance. For the weights for balance loss of generator, $\beta_g = 0$ produces fairly good results on all cases while positive β_g gives particularly significant improvement in some LSGAN and Hinge loss experiments. We choose $\beta_g \in \{1.0, 2.0\}$ for LSGAN experiments on all datasets and $\beta_g \in \{5.0, 10.0\}$ for Hinge loss experiments on MNIST.

StyleGAN2 backbone We adopt the configuration E architecture among the StyleGAN2 variations and use default hyperparameters for training using the official implementation[2] without applying data augmentation option. We set the batch size at 64 and 16 for CIFAR-10 and CelebA30K, respectively.

GMAN settings We used the official implementation[3] of GMAN. Among its varients, we use three versions that use the arithmetic mean of softmax, *i.e.*, GMAN-1, GMAN-0 and GMAN*, and choose the best scores among them to report in Table 1 and 2. For differentiating discriminators, we apply different dropout rates in $[0.4, 0.6]$ and split of mini-batches for the input of discriminators while adopting the same architectures as DCGAN.

E.3 Conditioned image synthesis

We apply the MCL components to the official codes of DRIT++[4], StackGAN++[5] and MSGAN[6] and use the default settings of their original implementations.

F Evaluation details

F.1 Unconditional GAN on image datasets

Evaluation metrics We measure precision/recall based on Precision Recall Distribution (PRD) [46]. We adopt F_8 and $F_{1/8}$ scores from the PRD curve as a recall and precision of each model, respectively. We use the official implementations of PRD[7] and FID[8] for the measurement.

DCGAN backbone We run the the experiment on MNIST [39], Fashion-MNIST [40], CIFAR-10 [41] and CelebA [42] until 40, 50, 150 and 30 epochs, respectively. We generate 60K random examples for MNIST and Fashion-MNIST and 50K random samples for the other datasets, and then compare them with the reference datasets with the same number of examples.

[1] https://github.com/caogang/wgan-gp
[2] https://github.com/NVlabs/stylegan2-ada-pytorch
[3] https://github.com/iDurugkar/GMAN
[4] https://github.com/HsinYingLee/DRIT
[5] https://github.com/hanzhanggit/StackGAN-v2
[6] https://github.com/HelenMao/MSGAN
[7] https://github.com/msmsajjadi/precision-recall-distributions
[8] https://github.com/bioinf-jku/TTUR

StyleGAN2 backbone We run the the experiment on CIFAR-10 [41] and CelebA30K [42] until 300 epochs and choose the best model in terms of FID. We generate 50K and 30K random examples for CIFAR-10 and CelebA30K, respectively, and then compare them with the whole train (/validation) set. We do not use the truncation trick when generating samples for quantitative evaluations.

F.2 Conditioned image synthesis

We measure FID, NDB/JSD[9] and LPIPS[10] using their official implementations. We follow all evaluation details in MSGAN [11] which is referenced for comparision.

G Limitations

The proposed method carries additional hyperparameters including the weights for several loss terms and the number of discriminators, and one might question the robustness of MCL-GAN with respect to the variations of the hyperparameters. From our analysis of the hyperparameter setting, presented in Appendix B, the performance of the proposed method improves significantly by the expert training and the balanced assignment of discriminators while the rest of the loss terms make stable contributions over a wide range of their weights. Also, since MCL-GAN adjusts the number of active discriminators that participate in learning as experts, its performance is robust to the number of discriminators. Although not included in the scope of this paper, there are some settings that need further investigation. For example, how to extend the multi-discriminator environment with an extremely small dataset and compatibility with recent data augmentation techniques are not discovered but are worth studying.

H Potential societal impact

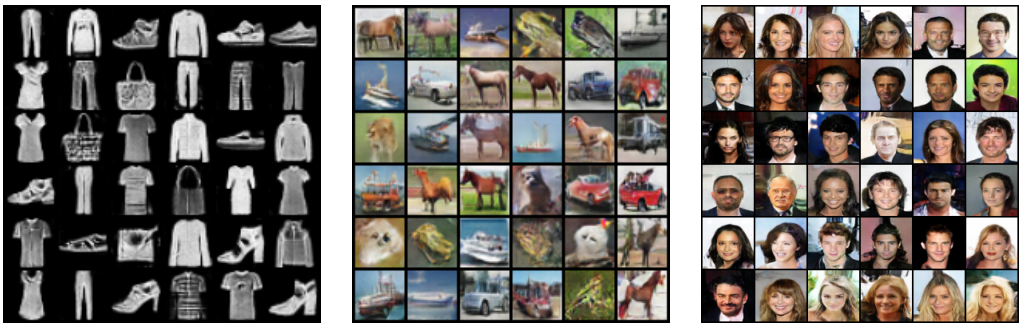
Deep generative models have some potentials to be used for adverse or abusive applications. Although our work involves unconditional image generations based on face datasets, this is rather a generic framework based on GANs to mitigate the mode collapse and dropping problems hampering sample diversity. Our algorithm is not directly related to particular applications with ethical issues, and we believe that the proposed approach can alleviate the bias and fairness issues by identifying the minority groups in a dataset effectively.

[9] https://github.com/eitanrich/gans-n-gmms

[10] https://github.com/richzhang/PerceptualSimilarity

(a) MCL-GAN (b) Real (c) GMAN

Figure 8: Qualitative comparison between MCL-GAN and GMAN on MNIST, Fashion-MNIST, CIFAR-10 and CelebA (from top to bottom). MCL-GAN generates more semantically faithful and diverse images with less failure cases compared to GMAN.

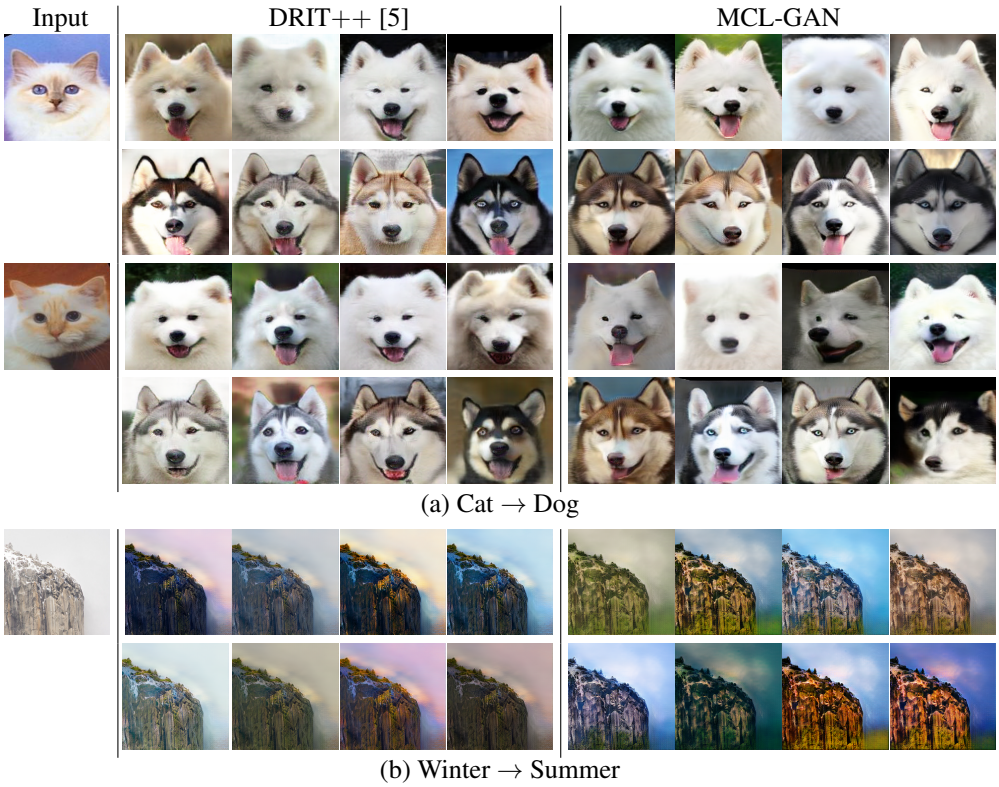


(a) Fashion-MNIST (b) CIFAR-10 (c) CelebA

Figure 9: Selected samples generated by MCL-GAN with DCGAN architecture.

(a) CIFAR-10 (b) CelebA30K

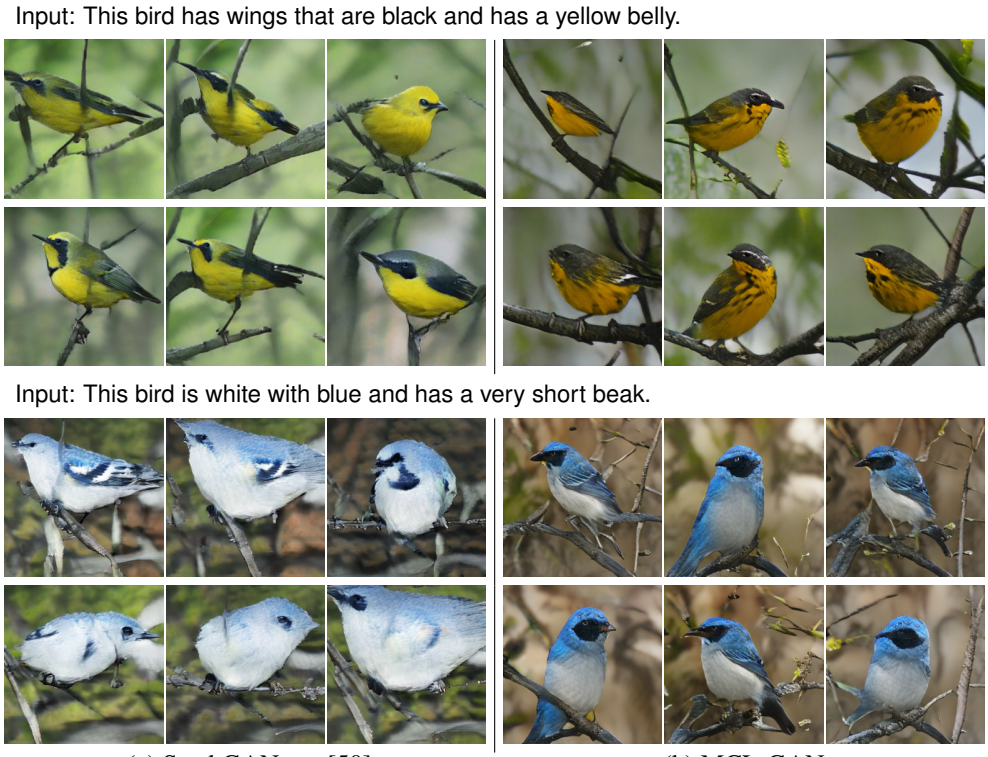
Figure 10: Random samples generated by MCL-GAN with StyleGAN2 architecture. For generation, truncation $\psi = 0.8$ is applied.



(a) Cat → Dog

(b) Winter → Summer

Figure 11: Diversity comparison of image-to-image translation on Yosemitee (Summer⇌Winter) and Cat⇌Dog dataset.



(a) StackGAN++ [50]

(b) MCL-GAN

Figure 12: Diversity comparison of text-to-image synthesis on CUB-200-2011.

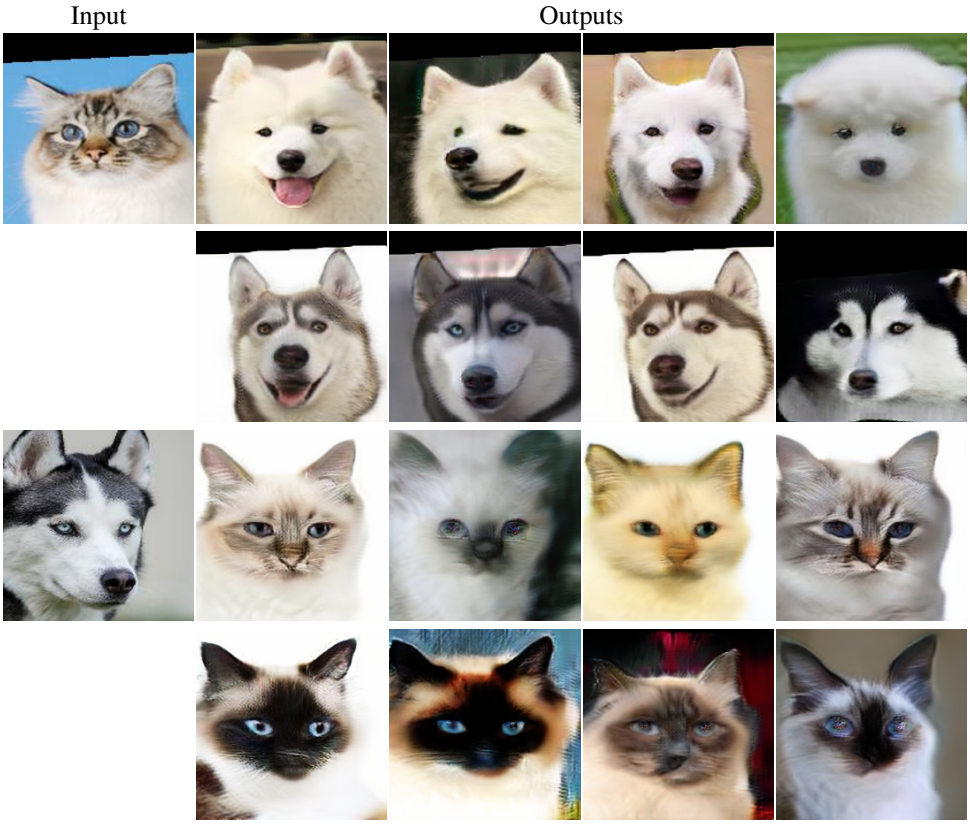


Figure 13: More image-to-image translation results by MCL-GAN on Cat⇌Dog.

Input: This bird has a pointed beak, yellow breast and belly, brown wings and yellow neck.

Input: This bird is white with black and has a very short beak.

Figure 14: More text-to-image synthesis results by MCL-GAN on CUB-200-2011.