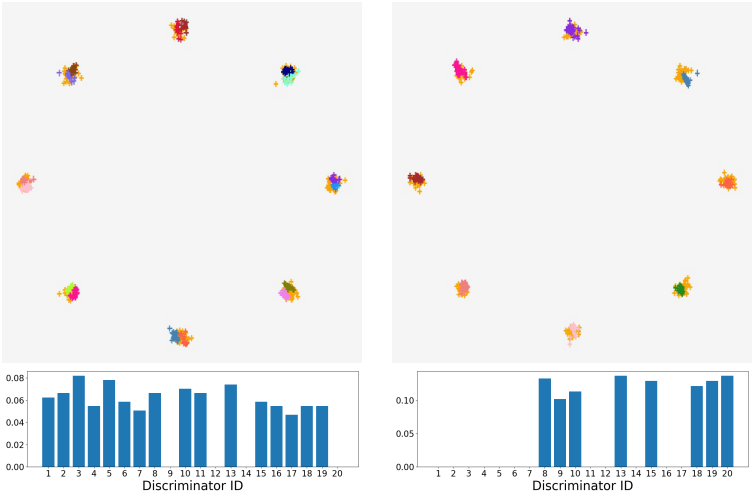
Appendix

A Analysis

A.1 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 coincides with 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.

A.2 Hyperparameters

A.2.1 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. While all positive values of α improve the precision scores, which indicates that the overconfidence is reduced in non-expert discriminators, the best score is obtained with $\alpha = 0.01$.

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
Rec.↑	0.983	0.984	0.978	0.982	0.983	0.976
Prec.↑	0.951	0.977	0.965	0.968	0.974	0.971

A.2.2 Balance loss weights

We conduct 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	Rec.↑	Prec.↑
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

A.2.3 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 value 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 with each other so the subclusters become less distinctive.

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

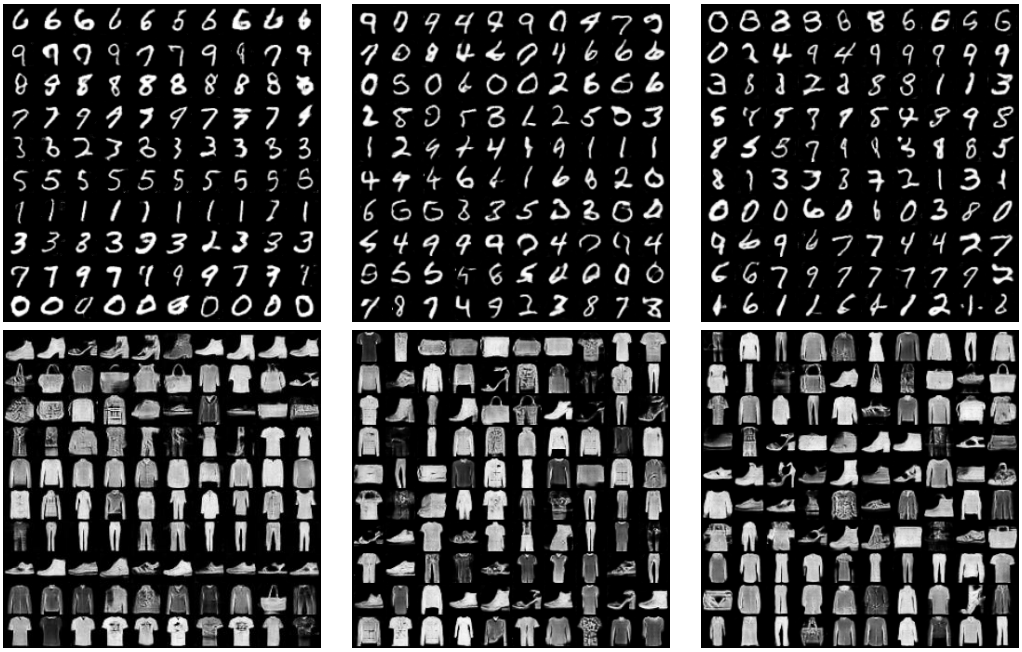
		MNIST		CIFAR-10	
M	k	Rec.↑	Prec.↑	Rec.↑	Prec.↑
5	1	0.983	0.975	0.903	0.942
5	3	0.983	0.981	0.896	0.948
10	1	0.973	0.973	0.902	0.937
10	3	0.973	0.969	0.913	0.946
10	5	0.975	0.964	0.917	0.948
10	7	0.960	0.927	0.912	0.951

A.2.4 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.

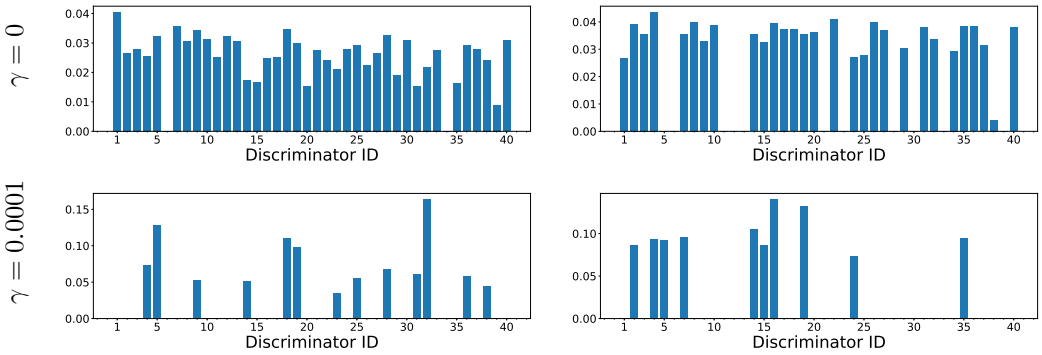
A.2.5 Stability to hyperparameter setting

We conduct experiments on MNIST with $M \in \{10, 20, 40\}$ 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).

M	ℓ_1 loss	MNIST Rec.↑	MNIST Prec.↑	Fashion-MNIST Rec.↑	Fashion-MNIST Prec.↑
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 for $M \in \{10, 20, 40\}$ and $\gamma = 0.0002$ on MNIST.

M	ℓ_1 loss	Rec.↑	Prec.↑	# 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

A.3 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 increase by several folds for GMAN [10] 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 a 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	-	-

B Qualitative Results

B.1 Unconditional GAN on image datasets

We investigate the quality of images generated by MCL-GAN and compare its performance with GMAN [10] 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 generates 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.

B.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 [12] 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.

C Implementation Details

C.1 Synthetic data

We reuse the experimental design and implementation[1] following [24].

C.2 Unconditional GANs 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 variants, 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.

C.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.

D Evaluation Details

D.1 Unconditional GANs 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 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 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.

D.2 Conditioned image synthesis

We measure FID, NDB/JSD[9] and LPIPS[10] using their official implementations. We follow all evaluation details in MSGAN [12] which is referenced for comparison.

E 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 A.2, 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.

F Potential Societal Impact

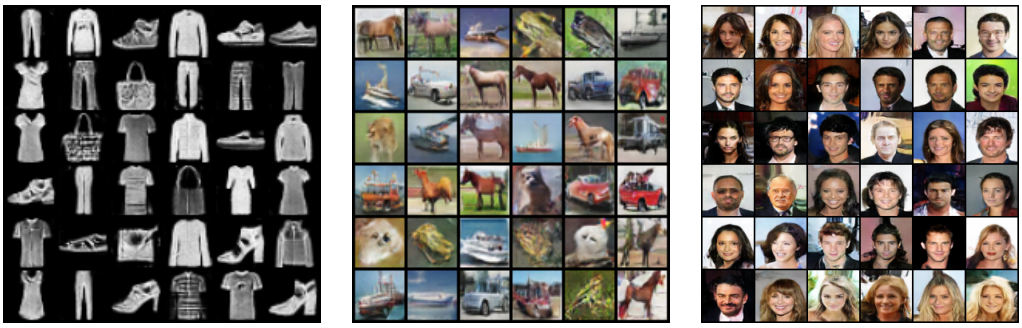
Deep generative models have some potentials to be used for adverse or abusive applications. Although our work involves unconditional image generations based on face datasets, this is rather a generic framework based on GANs to mitigate the mode collapse and dropping problems hampering sample diversity. 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.

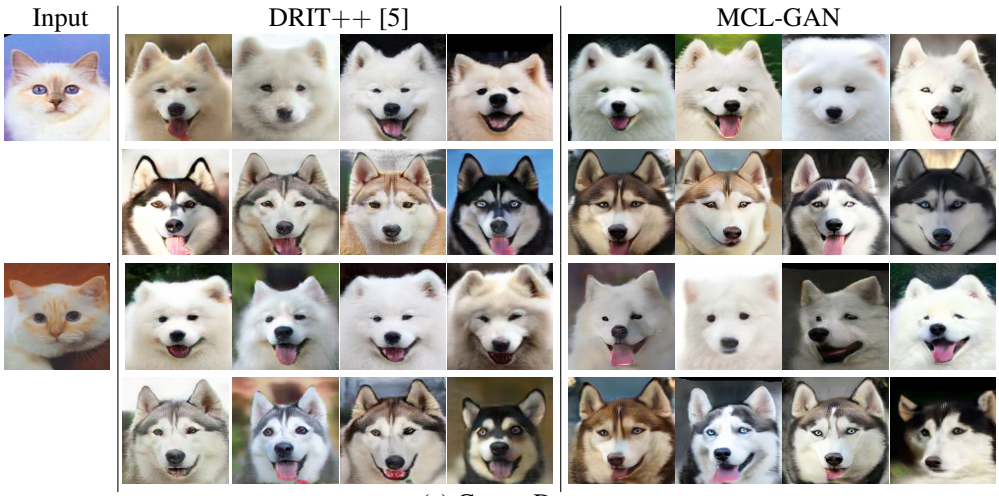


(a) Fashion-MNIST (b) CIFAR-10 (c) CelebA

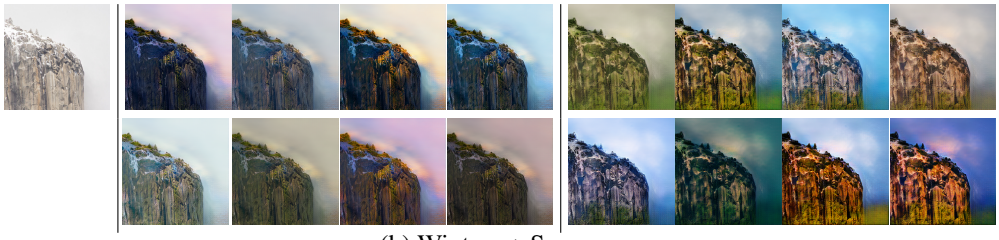
Figure 9: Selected samples generated by MCL-GAN with DCGAN architecture.

(a) CIFAR-10 (b) CelebA30K

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

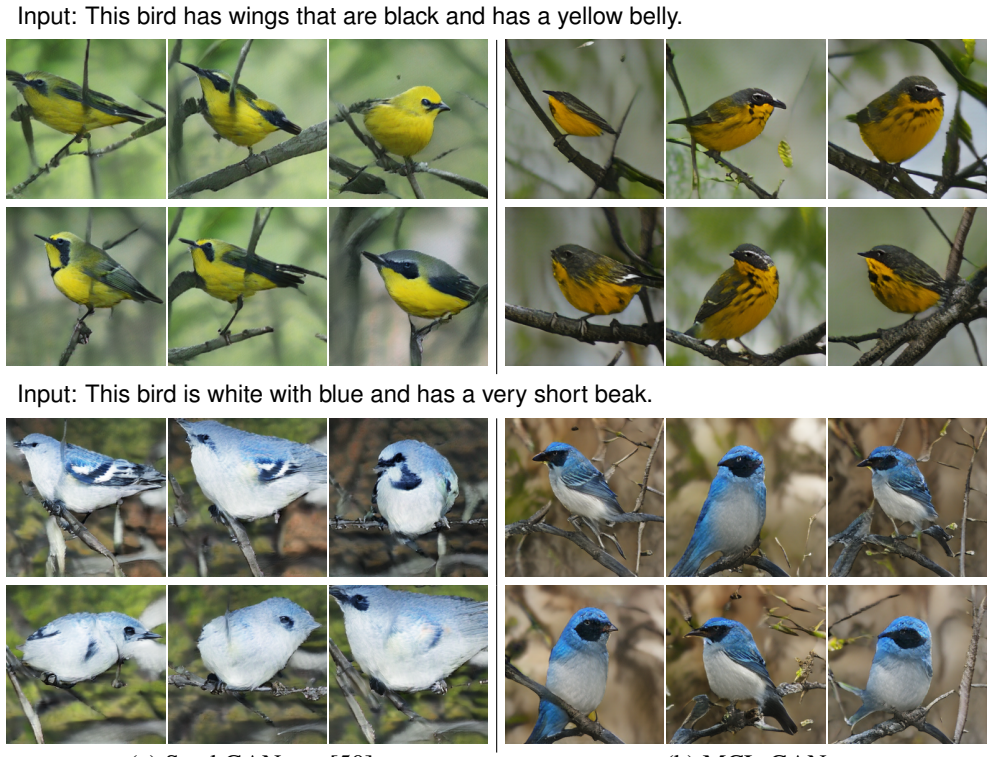


(a) Cat → Dog



(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.

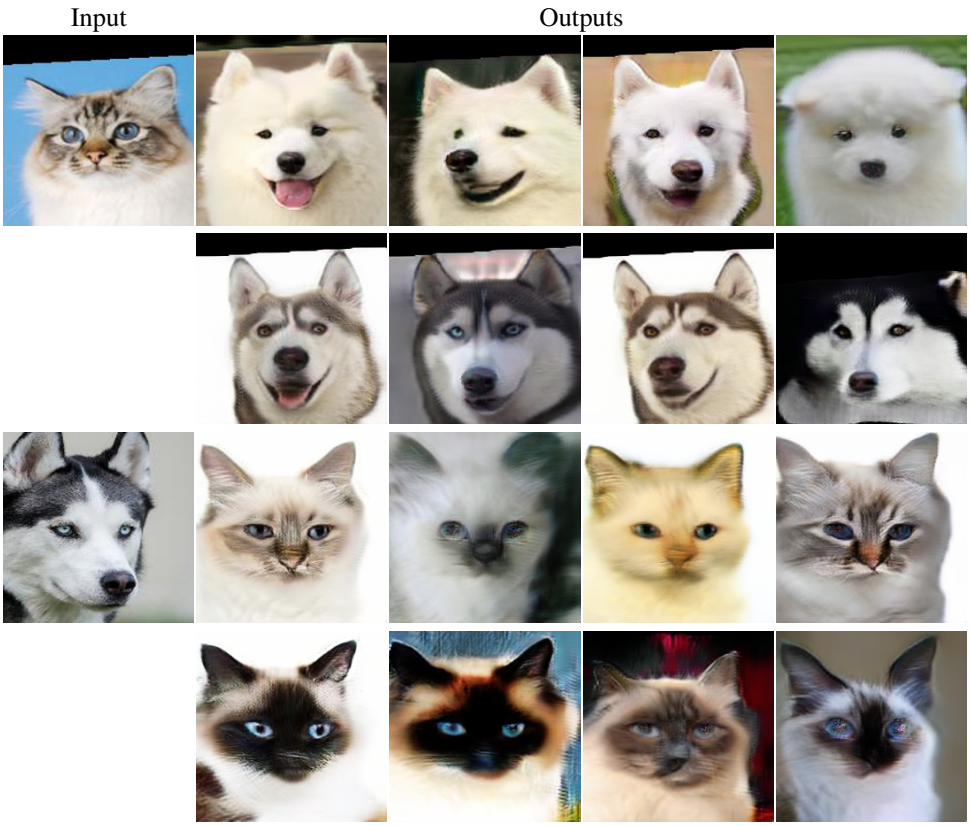


Figure 13: More image-to-image translation results by MCL-GAN on Cat⇌Dog.

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.