

Appendix for Differential-Critic GAN

A ABLATION STUDY

The objective in our DiCGAN (equation 4) consists of two components, i.e., the WGAN loss, which serves as the cornerstone of DiCGAN, and the ranking loss, which serves as the correction for WGAN. Meanwhile, we introduce the operation of replacement (equation 8) during the model training.

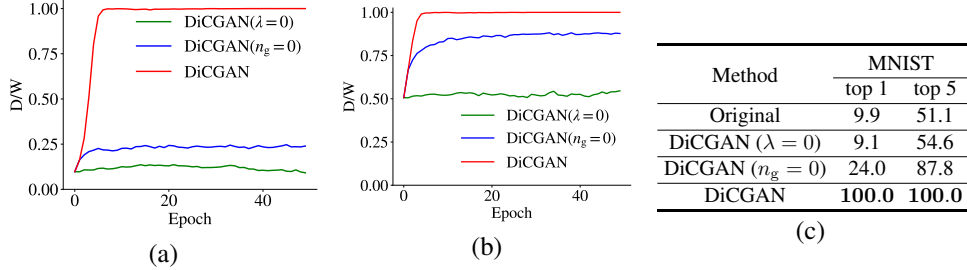


Figure 1: (a-b) The percentage of desired samples (D/W) versus epoch in DiCGAN ($\lambda = 0$), DiCGAN ($n_g = 0$) and DiCGAN. (a) plots the D/W of the digit zero. (b) plots the D/W of the digit zero to four. (c) The percentage of desired samples (D/W) from the original datasets, WGAN, DiCGAN ($\lambda = 0$), DiCGAN ($n_g = 0$) and DiCGAN.

To analyze the effects of the correction for WGAN (the third term in equation 5) and the replacement operation, we plot the percentage of desired samples (D/W) versus the training epoch for DiCGAN ($\lambda = 0$), DiCGAN ($n_g = 0$) and DiCGAN in Fig. 1a, 1b. Meanwhile, the converged percentage of desired samples (D/W) are reported in Fig. 1c. It can be seen that

1. **Without the correction term ($\lambda = 0$), DiCGAN cannot learn the desired data distribution.** The percentage of desired samples (D/W) from DiCGAN ($\lambda = 0$) remains constant during training on MNIST (Fig. 1a, 1b) compared with the original datasets (Fig. 1c). This is because that the remaining WGAN term in DiCGAN($\lambda = 0$) focuses on learning the training data distribution.
2. **Without the replacement ($n_g = 0$), DiCGAN makes a minor correction to the generated distribution.** In Fig. 1a, 1b, the D/W of DiCGAN ($n_g = 0$) slightly increases compared with the original datasets. This is consistent with our analysis that the correction term would drive the generation towards the desired distribution.
3. **DiCGAN learns the desired data distribution with a sequential minor correction.** The D/W of DiCGAN grows with training and reaches almost 100% when convergence. The correction term drives DiCGAN’s generation towards the desired data slightly at each epoch. With the iterative replacement, the minor correction sequentially accumulates and finally the generated distribution shifts to the desired data distribution.

B LEARNING THE DISTRIBUTION OF DESIRED OBJECTS

We consider cars as the desired objects and design the experiment to learn the distribution of cars in CIFAR.

Sample selection in FBGAN: A classifier, pretrained for classifying cars and planes, is adopted for selection. The generated objects, classified to car, are selected to replace the old training data.

Pairwise preferences construction in DiCGAN: Denoting label of CIFAR image as y , the pairwise preference between two images x_1 and x_2 are $x_1 \succ x_2$ when $y_1 = \text{“car”}$, $y_2 = \text{“plane”}$, and vice versa. At each iteration, we construct 32 pairs by random sampling pairs from the mini-batch 64 samples.

In Fig. 2, we visualize the generated CIFAR images randomly sampled from the generator of DiCGAN. It shows that DiCGAN gradually generates cars, as we desired.

Meanwhile, we sample $10K$ samples from the generator and calculate the percentage of car images among the generated samples for quantitative evaluation. In Table 1, (1) almost all images generated by DiCGAN and FBGAN are car images; (2) the percentage of car images generated by WGAN is similar to the training dataset. (3) We calculate the IS and MS-SSIM of the cars in the training data (denoted as “original”). The IS and the diversity of FBGAN and DiCGAN does not exhibit a big difference as the original dataset, which means they have relatively good quality and diversity. Meanwhile, they achieve comparable IS and MS-SSIM.

Method	CIFAR (IS)		
	D/W	IS	MS-SSIM
Dataset	50.0	4.96	0.45
FBGAN	93.9	3.78	0.52
DiCGAN	95.4	3.67	0.49

Table 1: Results on CIFAR dataset.



Figure 2: Generated images of DiCGAN on CIFAR. DiCGAN aims to learn the distribution of car images of CIFAR. The training dataset is composed of car and plane images in CIFAR-10.

C EXPERIMENT SETTINGS

Hyperparameter The batch size b is set to 50 for MNIST, 64 for CIFAR and CelebA-HQ datasets. The #generated samples n_g is set to 50K for MNIST, 1K for CIFAR and 3,000 for CelebA-HQ, respectively. Other hyperparameters are adopted the same as in Gulrajani et al. (2017).

We construct pairwise preferences using the minibatch samples at each iteration based on the classification labels. We construct the pairs by randomly selecting two samples from the minibatch samples, respectively. The pairs, in which two samples belong to the same class, i.e., same digits or same objects, are removed.

CelebA-HQ training setting WGAN is only trained with the constructed desired dataset. CWGAN conditions on c to model a conditional data distribution $p(x|c)$. There are 6,632 samples labeled as desired and 23,368 samples labeled as undesired in the training data. A classifier, pre-trained for classifying young faces and old faces, is adopted for predicting the labels for the generated face images. At every training epoch, FBGAN generates 3,000 images and those classified as the old are selected to replace the old training data. As for DiCGAN, the generated face image classified with the old attribute is preferred over the face image classified with the young attribute. At each iteration, we construct 32 pairs by random sampling pairs from the mini-batch 64 samples.

D MORE VISUAL RESULTS

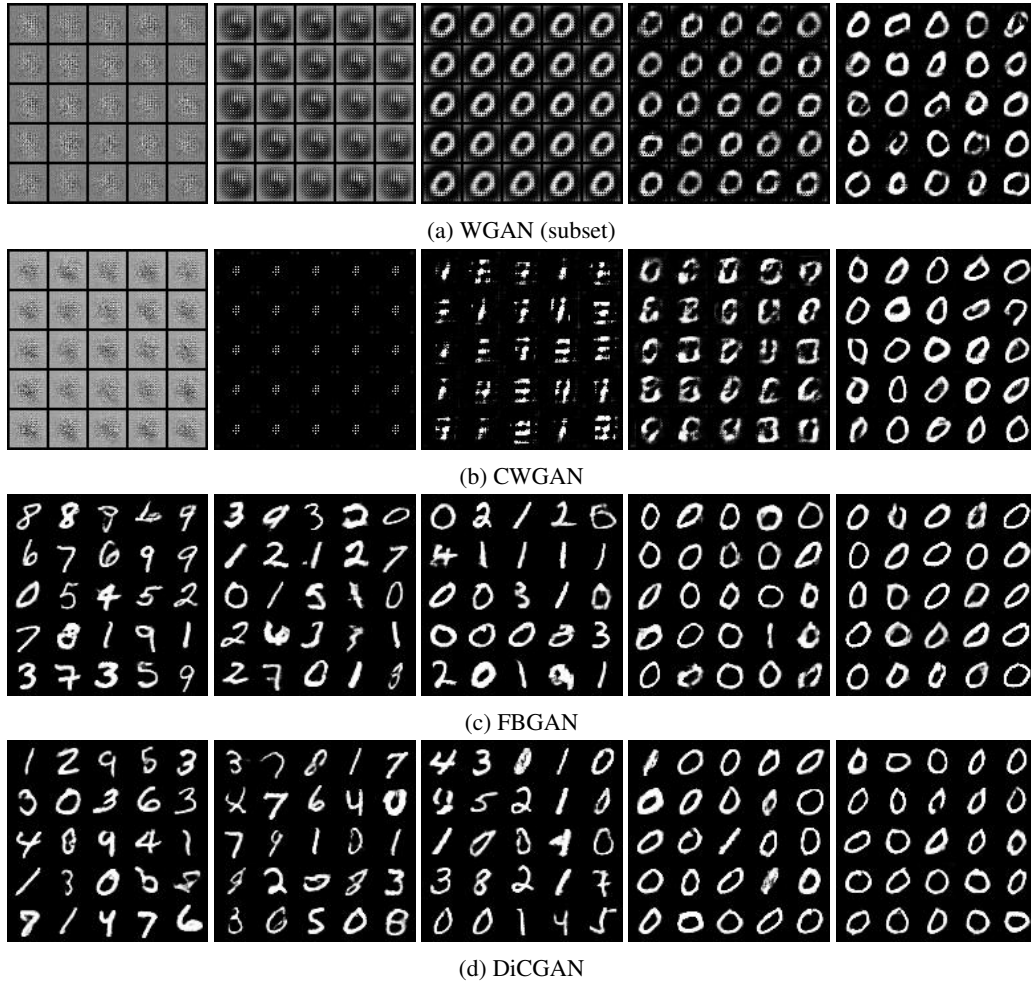


Figure 3: Generated digits of DiCGAN on MNIST during the training process.

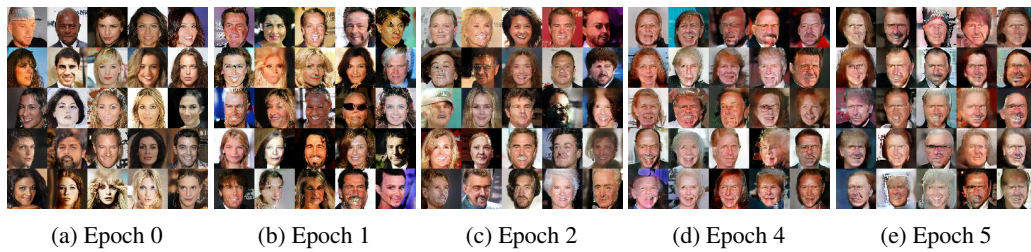


Figure 4: Generated images of DiCGAN on CelebA-HQ. DiCGAN learns the distribution of old faces. DiCGAN gradually generates more old face images.

REFERENCES

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

Algorithm 1 Training algorithm of DiCGAN

```
1: input: training data  $X$ , pairwise preferences  $S$ 
2: initilization: balance factor  $\lambda$ , #generated samples  $n_g$ , #pairs  $n_s$ , batchsize  $b$ , #iterations per epoch  $n_i$ ,
   #critic iterations per generator iteration  $n_{\text{critic}}$ 
3: Pretrain  $D$  and  $G$ 
4: repeat
5:   % Shift to the user-preferred distribution
6:   Generate samples using equation 7
7:   Replace old samples in  $X$  with  $X_g$  using equation 8
8:   Obtain pairwise preferences  $R$  using equation 2
9:   % Training of  $D$  and  $G$  at an epoch
10:  for  $i = 1, \dots, n_i$  do
11:    for  $t = 1, \dots, n_{\text{critic}}$  do
12:      Sample  $\{x^i\}_{i=1}^b$  from  $X$ ,  $\{z^i \sim p(z)\}_{i=1}^b$ 
13:      Sample  $\{s^j\}_{j=1}^{n_s}$  from  $S$ .
14:      Train the differential critic  $D$  using  $L_D$  in equation 5
15:    end for
16:    Train the generator  $G$  using  $L_G$  in equation 5
17:  end for
18: until converge
```
