# Rebooting ACGAN: Auxiliary Classifier GANs with Stable Training

**Minguk Kang** **Woohyeon Shim** **Minsu Cho** **Jaesik Park**

Pohang University of Science and Technology (POSTECH), South Korea
{mgkang, wh.shim, mscho, jaesik.park}@postech.ac.kr

## Abstract

Conditional Generative Adversarial Networks (cGAN) generate realistic images by incorporating class information into GAN. While one of the most popular cGANs is an auxiliary classifier GAN with softmax cross-entropy loss (ACGAN), it is widely known that training ACGAN is challenging as the number of classes in the dataset increases. ACGAN also tends to generate easily classifiable samples with a lack of diversity. In this paper, we introduce two cures for ACGAN. First, we identify that gradient exploding in the classifier can cause an undesirable collapse in early training, and projecting input vectors onto a unit hypersphere can resolve the problem. Second, we propose the Data-to-Data Cross-Entropy loss (D2D-CE) to exploit fruitful relational information in the class-labeled dataset. On this foundation, we propose the Rebooted Auxiliary Classifier Generative Adversarial Network (ReACGAN). The experimental results show that ReACGAN achieves the state-of-the-art generation performances on CIFAR10, Tiny-ImageNet, CUB200, and ImageNet datasets. We also verify that ReACGAN benefits from differentiable augmentations and that D2D-CE is harmonized well with StyleGAN2 architecture. All the model weights and the source codes are available at https://github.com/POSTECH-CVLab/PyTorch-StudioGAN.

## 1 Introduction

Generative Adversarial Networks (GAN) [1] are known for the forefront approach to generating high-fidelity images of diverse categories [2, 3, 4, 5, 6, 7, 8, 9]. Behind the sensational generation ability of GANs, there has been tremendous effort to develop adversarial objectives free from vanishing gradient problem [10, 11, 12], regularizations for stabilizing adversarial training [11, 13, 14, 3, 15, 16, 17], and conditioning techniques to support the adversarial training using category information of dataset [18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28]. Subsequently, the conditioning techniques have become the *de facto* standard for high-quality image generation. The models with the conditioning methods are called conditional Generative Adversarial Networks (cGAN), and cGANs can be divided into two groups depending on the discriminator's conditioning way: classifier-based GANs [18, 20, 23, 24, 25, 27] and projection-based GANs [19, 3, 4, 28].

The classifier-based GANs facilitate an auxiliary classifier to generate class-specific images by penalizing the generator if the synthesized outputs are not consistent with the conditioned labels. ACGAN [18] has been one of the widely used classifier-based GANs for its simple design and satisfactory generation performance. While ACGAN can exploit class information by pushing and pulling classifier's weights (proxies) against image embeddings [23], it is well known that ACGAN training is prone to collapse at the early stage of training as the number of classes increases [19, 23, 25, 27]. In addition, the generator of ACGAN tends to generate easily classifiable images at the cost of reduced diversity [18, 19, 27]. Projection-based GANs, on the other hand, have shown cutting-edge generation results on datasets with a large number of categories. SNGAN [3],

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

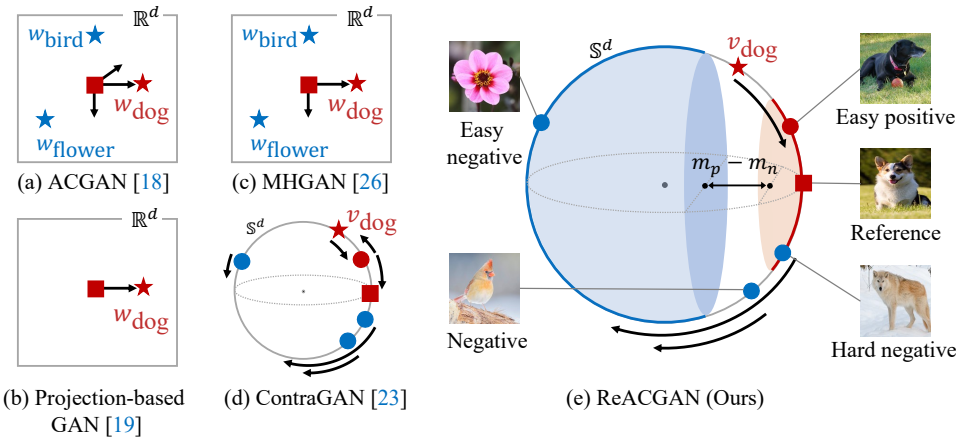



Figure 1: Schematics that depict how cGANs perform conditioning. The red color means positive samples/proxy, and the blue color indicates negative samples/proxies. Arrows represent push-pull forces based on the reference sample. The length of an arrow indicates the magnitude of the force.

SAGAN [29], and BigGAN [4] are representatives in this family and can generate realistic images on CIFAR10 [30] and ImageNet [31] datasets. However, projection-based GANs only consider a pairwise relationship between an image and its label proxy (data-to-class relationships). As the result, the projection-based GANs can miss an additional opportunity to consider relation information between data instances (data-to-data relationships) and between instance categories (class-to-class relationships) as discovered by [23].

In this paper, we analyze why ACGAN training becomes unstable as the number of classes increases and then propose remedies for (1) the instability and (2) the relatively poor generation performance of ACGAN compared with the projection-based models. First, we begin by analytically deriving the gradient of the softmax cross-entropy loss used in ACGAN. By examining the exact values of analytic gradients, we discover that the unboundedness of input feature vectors and poor classification performance in the early training stage can cause an undesirable gradient exploding problem. Second, with alleviating the instability, we propose the Rebooted Auxiliary Classifier Generative Adversarial Networks (ReACGAN) using the Data-to-Data Cross-Entropy loss (D2D-CE). ReACGAN projects image embeddings and proxies onto a unit hypersphere and computes similarities for data-to-data and data-to-class consideration. Additionally, we introduce two margin values for intra-class variations and inter-class separability. In this way, ReACGAN overcomes the training instability and can exploit additional supervisory signals by explicitly considering data-to-class and data-to-data relationships, and also by implicitly looking at class-to-class relationships in the same mini-batch.

To validate our model, we conduct image generation experiments on CIFAR10 [30], Tiny-ImageNet [32], CUB200 [33], and ImageNet [31]. Through the extensive experiments, we demonstrate that ReACGAN beats both classifier-based and projection-based GANs, improving over the state of the art by 2.5%, 15.8%, 5.1%, and 14.5% in terms of Fréchet Inception Distance (FID) [34] on the four datasets, respectively. We also verify that ReACGAN benefits from consistency regularization [16] and differentiable augmentations [9, 8]. Finally, we confirm that D2D-CE harmonizes with the StyleGAN2 architecture [7].

## 2 Background: Generative Adversarial Networks

Generative Adversarial Network (GAN) [1] is an implicit generative model that aims to generate a sample indistinguishable from the real. GAN consists of two networks: a *Generator* $G : \mathcal{Z} \longrightarrow \mathcal{X}$ that tries to map a latent variable $\mathbf{z} \sim p(\mathbf{z})$ into the real data space $\mathcal{X}$ and a *Discriminator* $D : \mathcal{X} \longrightarrow [0, 1]$ that strives to discriminate whether a given sample $\mathbf{x}$ is from the real data distribution $p_{\text{real}}(\mathbf{x})$ or the implicit distribution $p_{\text{gen}}(G(\mathbf{z}))$ derived from the generator $G(\mathbf{z})$. The objective of a vanilla GAN [1] can be expressed as follows:

$$\min_G \max_D \; \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\log(D(\mathbf{x}))] + \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))]. \tag{1}$$

While GANs have shown impressive results in the image generation task [2, 35, 11, 13], training GANs often ends up encountering a mode-collapse problem [36, 11, 12]. As one of the prescriptions for stabilizing and reinforcing GANs, training GANs with categorical information, named conditional Generative Adversarial Networks (cGAN), is suggested [37, 18, 19]. Depending on the presence of explicit classification losses, cGAN can be divided into two groups: Classifier-based GANs [18, 20, 23, 25, 27] and Projection-based GANs [19, 3, 4, 28]. ACGAN is one of the widely used classifier-based GANs with the softmax cross-entropy loss. ACGAN utilizes an auxiliary classifier and is trained to perform adversarial training and image classification jointly.

Although ACGAN has shown satisfactory generation results, training ACGAN becomes unstable as the number of classes in the training dataset increases [19, 23, 25, 27]. Besides, ACGAN tends to generate easily classifiable images at the cost of limited diversity [18, 19, 27]. To alleviate those problems, Zhou *et al.* [38] have proposed performing adversarial training on the classifier. Gong *et al.* [20] have deployed one more classifier to remove a conditional entropy which can hinder estimating the joint distribution of the real dataset. However, ACGAN training still suffers from the early-training collapse and reduced diversity problems when trained on datasets with a large number of class categories, such as Tiny-ImageNet [32] and ImageNet [31].

In this situation, Miyato *et al.* [19] have proposed a projection discriminator for cGAN and shown significant improvement in generating ImageNet dataset with a single pair of the generator and discriminator. Motivated by the promising results of the projection discriminator, many projection-based GANs [3, 29, 4, 16, 6, 17] are proposed and become the standard for conditional image generation. In this paper, we revisit ACGAN and unveil the source of the instability. Coping with the instability, we propose the Rebooted Auxiliary Classifier GANs for realistic image generation.

## 3 Rebooting Auxiliary Classifier GANs

### 3.1 Feature Normalization

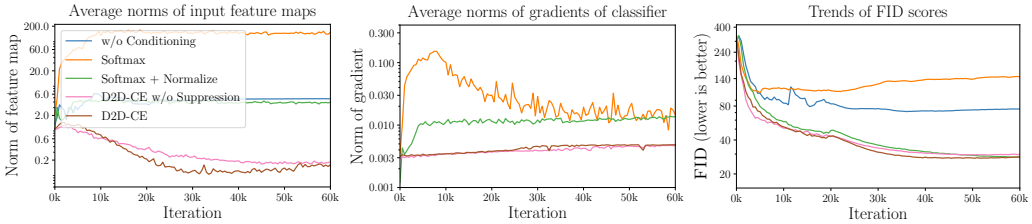
To uncover a nuisance that can cause the instability of ACGAN, we start by analytically deriving gradients of weight vectors in the softmax classifier. Let the part of the discriminator before the fully connected layer be a *Feature extractor* $F : \mathcal{X} \longrightarrow \mathcal{F} \in \mathbb{R}^d$ and let *Classifier* $C : \mathcal{F} \longrightarrow \mathbb{R}^c$ be a single fully connected layer parameterized by $\boldsymbol{W} = [\boldsymbol{w}_1, ..., \boldsymbol{w}_c] \in \mathbb{R}^{d \times c}$, where $c$ denotes the number of classess. We sample training images $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$ and integer labels $\boldsymbol{y} = \{y_1, ..., y_N\}$ from the joint distribution $p(\mathbf{x}, \mathbf{y})$. Using the above notations, we can express the empirical cross-entropy loss used in ACGAN [18] as follows:

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N}\sum_{i=1}^{N} \log\Big(\frac{\exp\left(F(\boldsymbol{x}_i)^\top \boldsymbol{w}_{y_i}\right)}{\sum_{j=1}^{c}\exp\left(F(\boldsymbol{x}_i)^\top \boldsymbol{w}_{y_j}\right)}\Big). \tag{3}$$

Based on this, we can calculate the derivative of the cross-entropy loss, w.r.t $\boldsymbol{w}_{k\in\{1,...,c\}}$ as follows:

$$\frac{\partial \mathcal{L}_{\text{CE}}}{\partial \boldsymbol{w}_k} = -\frac{1}{N}\sum_{i=1}^{N}\Big\{F(\boldsymbol{x}_i)\Big(\mathbf{1}_{y_i=k} - p_{i,k}\Big)\Big\}, \tag{4}$$

where $\mathbf{1}_{y_i=k}$ is an indicator function that will output 1 if $y_i = k$ is satisfied, and $p_{i,k}$ is a class probability that represents the probability of $i$-th sample belongs to class $k$, mathematically $\frac{\exp\left(F(\boldsymbol{x}_i)^\top \boldsymbol{w}_k\right)}{\sum_{j=1}^{c}\exp\left(F(\boldsymbol{x}_i)^\top \boldsymbol{w}_j\right)}$. The equation above implies that the norm of a gradient of the softmax cross-entropy loss is coupled with the norms and directions of each input feature map $F(\boldsymbol{x}_i)$ and the class probability $p_{i,k}$. In the early training stage, the classifier is prone to making incorrect predictions, resulting in low probabilities. This phenomenon occurs more frequently as the number of categories in the dataset increases. As the result, the norm of the gradient $|\frac{\partial \mathcal{L}_{\text{CE}}}{\partial \boldsymbol{w}_k}|$ begins to explode as the vector $F(\boldsymbol{x}_i)$ stretches out to $\boldsymbol{w}_k$ direction but being located close to the the other vectors $\boldsymbol{w}_{j\in\{1,...,c\}\setminus\{k\}}$. This often breaks the balance between adversarial learning and classifier training, leading to an early-training collapse. Once the early-training collapse occurs, ACGAN training concentrates on classifying categories of images instead of discriminating the authenticity of given samples. We experimentally demonstrate that the average norm of ACGAN's input feature maps increases as the training progresses (Fig. 2a). Accordingly, the average norm of the gradients increases sharply at the early training stage and decreases with the high class probabilities of the classifier (Fig. 2b, Fig.




(a) Feature norm (b) Gradient norm (c) FID score

Figure 2: Merits of integrating feature normalization and data-to-data relationship consideration. All experiments are conducted on Tiny-ImageNet [32] dataset. (a) Average norms of input feature maps $F(\cdot)$, (b) average norms of gradients of classification losses, and (c) trends of FID scores. Compared with ACGAN [18], the proposed ReACGAN does not experience the training collapse problem caused by excessively large norms of feature maps and gradients at the early training stage. In addition, ReACGAN can converge to better equilibrium by considering data-to-data relationships with easy positive and negative sample suppression.

A3 in Appendix). While the average norm of gradients decreases at some point, the FID value of ACGAN does not decrease, implying the collapse of ACGAN training (Fig. 2c).

As one of the remedies for the gradient exploding problem, we find that simply *normalizing the feature embeddings onto a unit hypersphere* $\frac{F(\boldsymbol{x}_i)}{||F(\boldsymbol{x}_i)||}$ effectively resolves ACGAN's early-training collapse (see Fig. 2). The motivation is that normalizing features onto the hypersphere restricts the norms of features under 1.0. Thus, the discriminator does not experience the gradient exploding problem. From the next section, we will deploy a projection layer $P$ on the feature extractor $F$ and will normalize both the embeddings from the projection layer and the weight vectors $W$ in the classifier since applying normalization to both the embeddings and the weight vectors does not degrade image generation performance, and it is convenient for future development. We denote the normalized embedding $\frac{P(F(\boldsymbol{x}_i))}{||P(F(\boldsymbol{x}_i))||}$ as $\boldsymbol{f}_i$ and the normalized weight vector $\frac{\boldsymbol{w}_{y_i}}{||\boldsymbol{w}_{y_i}||}$ as $\boldsymbol{v}_{y_i}$.

### 3.2 Data-to-Data Cross-Entropy Loss (D2D-CE)

We expand the feature normalized softmax cross-entropy loss described in Sec. 3.1 to the Data-to-Data Cross-Entropy (D2D-CE) loss. The motivations are summarized into two points: (1) replacing data-to-class similarities in the denominator of Eq. (3) with data-to-data similarities and (2) introducing two margin values to the modified softmax cross-entropy loss. We expect that point (1) will encourage the feature extractor to consider data-to-data as well as data-to-class relationships, and that the point (2) will guarantee inter-class separability and intra-class variations in the feature space while preventing ineffective gradient updates induced by easy negative and positive samples. To develop the feature normalized cross-entropy loss into D2D-CE, we replace the similarities between a sample embedding and all proxies except for the positive one in the denominator, $\sum_{j\in\{1,\dots,c\}\setminus\{y_i\}} \exp(\boldsymbol{f}_i^\top \boldsymbol{v}_j)$ with similarities between negative samples in the same mini-batch. The modified cross-entropy loss can be expressed as follows:

$$\mathcal{L}_{\text{CE}}^{'} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{\exp(\boldsymbol{f}_i^\top \boldsymbol{v}_{y_i}/\tau)}{\exp(\boldsymbol{f}_i^\top \boldsymbol{v}_{y_i}/\tau) + \sum_{j\in\mathcal{N}(i)}\exp(\boldsymbol{f}_i^\top \boldsymbol{f}_j/\tau)}\right), \tag{5}$$

where $\tau$ is a temperature, and $\mathcal{N}(i)$ is the set of indices that point locations of the negative samples whose labels are different from the reference label $\boldsymbol{v}_{y_i}$ in the mini-batch. The similarity matrix calculated by two mini-batches of samples is used to compute the similarities between negative samples with a false negative mask (see Fig.3). As the result, Eq. (5) enables the discriminator to contrastively compare visual differences between multiple images and can supply a more informative supervision for image conditioning. Finally, we introduce two margin values to $\mathcal{L}_{\text{CE}}^{'}$ and name it *Data-to-Data Cross-Entropy loss* (D2D-CE). The proposed D2D-CE can be expressed as follows:

$$\mathcal{L}_{\text{D2D-CE}} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{\exp\left([\boldsymbol{f}_i^\top \boldsymbol{v}_{y_i} - m_p]_-/\tau\right)}{\exp\left([\boldsymbol{f}_i^\top \boldsymbol{v}_{y_i} - m_p]_-/\tau\right) + \sum_{j\in\mathcal{N}(i)}\exp\left([\boldsymbol{f}_i^\top \boldsymbol{f}_j - m_n]_+/\tau\right)}\right), \tag{6}$$

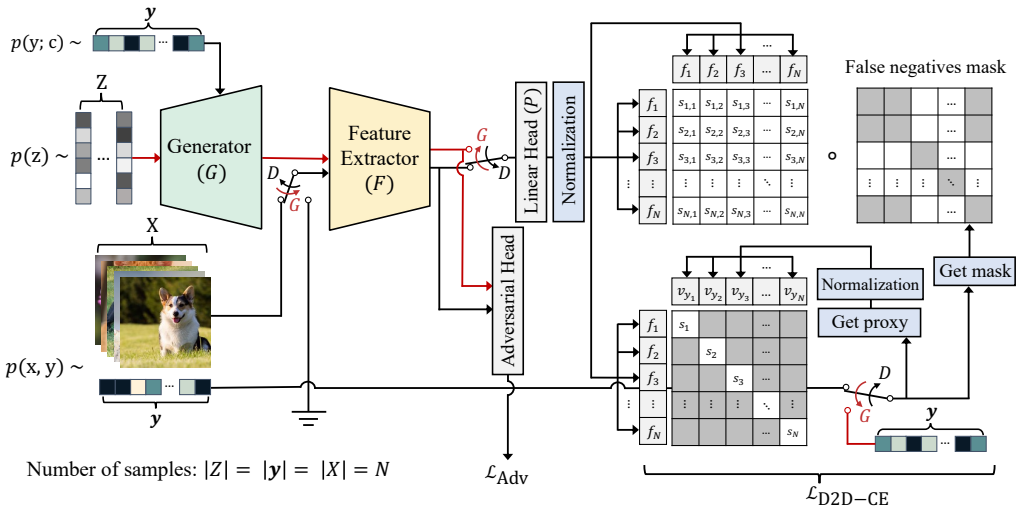



Figure 3: Overview of ReACGAN. ReACGAN performs adversarial training using the loss denoted as $\mathcal{L}_{\text{Adv}}$. At the same time, ReACGAN tries to minimize Data-to-Data Cross-Entropy loss (D2D-CE) on the linear projection head ($P$) to exploit relational information in the labeled dataset. $\circ$ means element-wise product. Note that *False negatives mask* operates on the similarity matrix between two batches of sample embeddings to compute negative similarities in the denominator of Eq. (5) and (6).

where $m_p$ is a margin value for the similarity between the reference sample $f_i$ and its corresponding proxy $\boldsymbol{v}_{y_i}$, $m_n$ is a margin value for similarities between negatives samples $\boldsymbol{f}_i^\top \boldsymbol{f}_j$. $[\cdot]_-$ and $[\cdot]_+$ denote min($\cdot$,0) and max($\cdot$,0) functions, respectively.

### 3.3 Useful Properties of Data-to-Data Cross-Entropy Loss

In this subsection, we explain four useful properties of D2D-CE. Let $s_q$ be a similarity between the normalized embedding $\boldsymbol{f}_q$ and the corresponding normalized proxy $\boldsymbol{v}_{y_q}$, $s_{q,r}$ be a similarity between $\boldsymbol{f}_q$ and $\boldsymbol{f}_r$, and $a$ and $b$ be arbitrary indices of negative samples, *i.e.*, $a, b \in \mathcal{N}(q)$. Then the properties of D2D-CE can be summarized as follows:

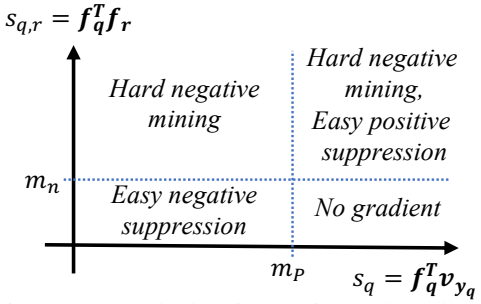



Figure 4: Graph showing regions where hard negative mining, easy positive and negative suppression, and no gradient update occur.

**Property 1.** *Hard Negative Mining. If the value of $s_{q,a}$ is greater than $s_{q,b}$, the derivative of $\mathcal{L}_{\text{D2D-CE}}$ w.r.t $s_{q,a}$ is greater than or equal to the derivative w.r.t $s_{q,b}$; that is $\frac{\partial \mathcal{L}_{\text{D2D-CE}}}{\partial s_{q,a}} \geq \frac{\partial \mathcal{L}_{\text{D2D-CE}}}{\partial s_{q,b}} \geq 0$.*

**Property 2.** *Positive suppression. If $s_q - m_p > 0$, the derivative of $\mathcal{L}_{\text{D2D-CE}}$ w.r.t $s_q$ is 0.*

**Property 3.** *Negative suppression. If $s_{q,r} - m_n < 0$, the derivative of $\mathcal{L}_{\text{D2D-CE}}$ w.r.t $s_{q,r}$ is 0.*

**Property 4.** *$\mathcal{L}_{\text{D2D-CE}}$ has the global minima of $\frac{1}{N}\sum_{i=1}^{N} \log\left(1 + |\mathcal{N}(i)|\right)$ when there does not exist an arbitrary $q$ and $r$ such that $s_q - m_p < 0$ or $s_{q,r} - m_n > 0$.*

We put proofs of the above properties in Appendix D. Property 1 indicates that D2D-CE implicitly conducts hard negative mining and benefits from comparing samples with each other. Also, Properties 2 and 3 imply that samples will not affect gradient updates if the samples are trained sufficiently. Consequently, the classifier concentrates on pushing and pulling hard negative and hard positive examples without being dominated by easy samples.

### 3.4 Rebooted Auxiliary Classifier Generative Adversarial Networks (ReACGAN)

With the proposed D2D-CE objective, we propose the *Rebooted Auxiliary Classifier Generative Adversarial Networks* (ReACGAN). As ACGAN does, ReACGAN jointly optimizes adversarial

and classification objectives. Specifically, the discriminator, which consists of the feature extractor, adversarial head, and projection head, of ReACGAN tries to discriminate whether a given image is sampled from the real distribution or not. At the same time, the discriminator tries to maximize similarities between real positive samples and corresponding proxies while minimizing similarities between real negative samples using D2D-CE loss. After predetermined numbers of the discriminator updates end, the generator strives to deceive the discriminator by generating realistic images that will give a low D2D-CE objective value. By alternatively training the discriminator and generator until convergence, ReACGAN generates high-quality images of diverse categories without early-training collapse. We attach the algorithm table for ReACGAN training in Appendix A.

**Differences between ReACGAN and ContraGAN.** The authors of ContraGAN [23] propose a conditional contrastive loss (2C loss) to cover data-to-data relationships when training cGANs. The main differences between 2C loss and D2D-CE are summed up into three points: (1) while 2C loss is derived from NT-Xent loss [39], which is popularly used in self-supervised learning, our D2D-CE is developed to resolve the early-training collapsing problem and poor generation performance of ACGAN, (2) 2C loss holds false-negative samples in the denominator, and it causes unexpected forces to repulse positive samples out of the reference embedding, and (3) 2C loss contains similarities between all positive samples in the numerator, and it will give rise to large gradients on easy positive samples. As a result, GANs with 2C loss tend to synthesize images of unintended classes as reported in the author's software document [40]. However, D2D-CE loss does not contain the false negatives in the denominator and only considers the similarity between data and its proxy in the numerator. Therefore, ReACGAN does not conduct unnecessary easy positive mining and is free from the unwanted repelling forces. More detailed explanations are attached to Appendix F.

**Consistency Regularization and D2D-CE Loss.** Zhang *et al.* [16] propose a consistency regularization to let the discriminator output consistent predictions if given two images are visually close to each other. They create a visually similar image pair by augmenting the reference image with predefined augmentations. After that, they force the discriminator to minimize the L2 distance between the logits of the reference and augmented images. While ReACGAN locates an image embedding nearby to its corresponding proxy and far apart from multiple image embeddings according to class labels, consistency regularization only pulls the original and augmented images towards each other. Since consistency regularization and D2D-CE can be applied together, we will show that ReACGAN benefits from consistency regularization in the experiments section (Table 1).

# 4 Experiments

## 4.1 Datasets and Evaluation Metrics

To verify the effectiveness of ReACGAN, we conduct conditional image generation experiments using five datasets: CIFAR10 [30], Tiny-ImageNet [32], CUB200 [33], ImageNet [31], and AFHQ [41] datasets and four evaluation metrics: Inception Score (IS) [42], Fréchet Inception Distance (FID) [34], and $F_{0.125}$ (Precision) and $F_8$ (Recall) [43]. The details on the training datasets are in Appendix C.1.

**Inception Score (IS)** [42] **and Fréchet Inception Distance (FID)** [34] are commonly used metrics for evaluating generative models. We utilize IS and FID together because some studies [4, 6, 25] have shown that IS has a tendency to measure the fidelity of images better while FID tends to weight capturing the diversity of images.

**Precision ($F_{0.125}$) and Recall ($F_8$)** [43] are metrics for estimating precision and recall of the approximated distribution $p(G(\mathbf{z}))$ against the true data distribution $p(\mathbf{x})$. Instead of evaluating generative models using one-dimensional scores, such as IS and FID, Sajjadi *et al.* [43] suggest using a two-dimensional score $F_{0.125}$ and $F_8$ that can quantify how precisely the generated images are and how well the generated images cover the reference distribution.

## 4.2 Experimental Details

The implementation of ReACGAN basically follows our PyTorch-StudioGAN library [40][1] that supports various experimental setups from ACGAN [18] to StyleGAN2-ADA [7, 8] with different

[1]PyTorch-StudioGAN is an open-source library under the MIT license (MIT) with the exception of StyleGAN2 and StyleGAN2-ADA related implementations, which are under the NVIDIA source code license.

Table 1: Comparison with classifier-based GANs [18, 23] and projection-based GANs [3, 4, 29] on CIFAR10 [30], Tiny-ImageNet [32], and CUB200 [33] datasets using IS [42], FID [34], $F_{0.125}$ and $F_8$ [43] metrics. For baselines, both the numbers in the original paper (denoted as * in method) and from our experiments using StudioGAN library [23] are reported. The numbers in bold-faced denote the best performance and in underline indicate the values are in one standard deviation from the best.

| Method | CIFAR10 [30] | | | | Tiny-ImageNet [32] | | | | CUB200 [33] | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ |
| SNGAN* [3] | 8.22 | 21.7 | - | - | - | - | - | - | - | - | - | - |
| BigGAN* [4] | 9.22 | 14.73 | - | - | - | - | - | - | - | - | - | - |
| ContraGAN* [23] | - | 10.60 | - | - | - | 29.49 | - | - | - | - | - | - |
| ACGAN [18] | 9.84 | 8.45 | 0.993 | **0.992** | 6.00 | 96.04 | 0.656 | 0.368 | **6.09** | 60.73 | 0.726 | 0.891 |
| SNGAN [3] | 8.67 | 13.33 | 0.985 | 0.976 | 8.71 | 51.15 | 0.900 | 0.702 | 5.41 | 47.75 | 0.754 | 0.912 |
| SAGAN [29] | 8.66 | 14.31 | 0.983 | 0.973 | 8.74 | 49.90 | 0.872 | 0.712 | 5.48 | 54.29 | 0.728 | 0.882 |
| BigGAN [4] | 9.81 | 8.08 | 0.993 | **0.992** | 12.78 | 32.03 | 0.948 | 0.868 | 4.98 | 18.30 | 0.924 | **0.967** |
| ContraGAN [23] | 9.70 | 8.22 | 0.993 | 0.991 | 13.46 | 28.55 | **0.974** | 0.881 | 5.34 | 21.16 | 0.935 | 0.942 |
| ReACGAN | **9.89** | **7.88** | **0.994** | **0.992** | **14.06** | **27.10** | 0.970 | **0.894** | 4.91 | **15.40** | **0.970** | 0.954 |
| CR-BigGAN* [16] | - | 11.48 | - | - | - | - | - | - | - | - | - | - |
| CR-BigGAN [16] | 9.97 | **7.18** | 0.995 | 0.993 | 15.94 | 19.96 | 0.972 | **0.950** | **5.14** | 11.97 | 0.978 | **0.981** |
| CR-ContraGAN [16, 23] | 9.59 | 8.55 | 0.992 | 0.972 | 15.81 | **19.21** | 0.983 | 0.941 | 4.90 | 11.08 | 0.984 | 0.967 |
| CR-ReACGAN [16] | **10.11** | 7.20 | **0.996** | **0.994** | **16.56** | 19.69 | **0.984** | 0.940 | 4.87 | **10.72** | **0.985** | 0.971 |
| DiffAug-BigGAN* [9] | 9.17 | 8.49 | - | - | - | - | - | - | - | - | - | - |
| DiffAug-BigGAN [9] | 9.94 | 7.17 | 0.995 | 0.992 | 18.08 | 15.70 | 0.980 | **0.972** | **5.53** | 12.15 | 0.967 | 0.981 |
| DiffAug-ContraGAN [9, 23] | 9.95 | 7.27 | 0.995 | 0.992 | 18.20 | 15.40 | 0.986 | 0.963 | 5.39 | 11.02 | 0.978 | 0.970 |
| DiffAug-ReACGAN [9] | **10.22** | **6.79** | **0.996** | **0.993** | **20.60** | **14.25** | **0.988** | **0.972** | 5.22 | **9.27** | **0.985** | **0.983** |
| Real Data | 11.54 | | | | 34.11 | | | | 5.49 | | | |

scales of datasets [30, 32, 33, 31, 41] and architectures [2, 13, 4, 7]. For a fair comparison, we use the same backbones over all baselines, except otherwise noted, for both the discriminator and generator. For stable training, we apply spectral normalization (SN) [3] to the generator and discriminator except for experiments using SNGAN (in this case, we apply SN to the discriminator only) and StyleGAN2 [7, 8]. We also use the same conditioning method for generators with conditional batch normalization (cBN) [44, 45, 19]. This allows us to investigate solely the conditioning methods of the discriminator, which are the main interests to validate our proposed method. If not specified, we use hinge loss [46] as a default for the adversarial loss.

Before conducting main experiments, we perform grid search for hyperparameter selection with candidates of a temperature $\tau \in \{0.125, 0.25, 0.5, 0.75, 1.0\}$ and a negative margin value $m_n \in \{0.5, 0.75, 0.9, 0.95, 0.98, 1.0\}$. For convenience, we set a positive margin value $m_p$ as $1-m_n$. Through extensive experiments with 3 runs per each setting, we choose $\tau$ with $\{0.5, 0.75, 0.25, 0.5\}$ and $m_n$ with $\{0.98, 1.0, 0.95, 0.98\}$ for CIFAR10, Tiny-ImageNet, CUB200 and ImageNet, respectively. A low temperature seems to work well on fine-grained image generation task, but generally, ReACGAN is robust to the choice of hyperparameters. The results of the hyperparameter search and other hyperparameter setups are provided in Appendix C.2.

We evaluate all methods through the same protocol of [9, 16, 17], which use the same amounts of generated images from the reference splits specialized for each dataset.[2] Besides, we run all the experiments three times with random seeds and report the averaged best performances for reliable evaluation with the lone exception of ImageNet and StyleGAN2 related experiments. Please refer to Appendix C.2 for other experimental details. The numbers in bold-faced denote the best performance and in underline indicate that the values are in one standard deviation from the best.

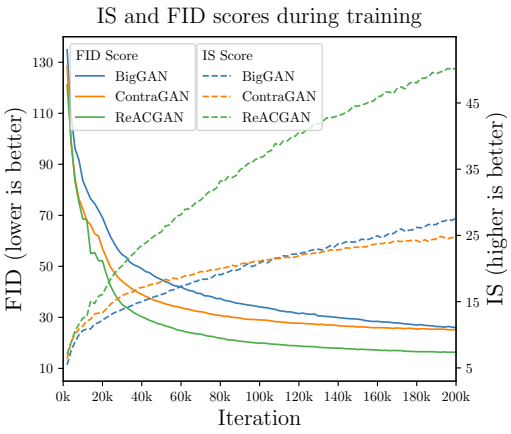
### 4.3 Evaluation Results

**Comparison with other cGANs.** We compare ReACGAN with previous state-of-the-art cGANs in Tables 1 and 2. We employ the implementations of GANs in PyTorch-StudioGAN library as they provide improved results on standard benchmark datasets [30, 31]. For comparison, we provide results from each original paper (denoted as * in method) as well as those from StudioGAN

[2] We use the validation split as the default reference set, but we use the test split of CIFAR10 and the training split of CUB200 and AFHQ due to the absence or lack of the validation dataset of such datasets.

Table 2: Experiments using ImageNet [31] dataset. B.S. means batch size for training. (Left): Comparisons with previous cGAN approaches. (Right): Learning curves of BigGAN [4], ContraGAN [23], and ReACGAN (ours) which are trained using batch size of 256.

| | Method | ImageNet [31] | | | |
|---|---|---|---|---|---|
| | | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ |
| B.S.= 256 | ACGAN* [18, 20] | 7.26 | 184.41 | - | - |
| | SNGAN* [3] | 36.80 | 27.62 | - | - |
| | SAGAN* [29] | 52.52 | 18.28 | - | - |
| | BigGAN* [4, 20] | 38.05 | 22.77 | - | - |
| | TAC-GAN* [20] | 28.86 | 23.75 | - | - |
| | ContraGAN* [23] | 31.10 | 19.69 | 0.951 | 0.927 |
| | ACGAN [18] | 62.99 | 26.35 | 0.935 | 0.963 |
| | SNGAN [3] | 32.25 | 26.79 | 0.938 | 0.913 |
| | SAGAN [29] | 29.85 | 34.73 | 0.849 | 0.914 |
| | BigGAN [4] | 43.97 | 16.36 | 0.964 | 0.955 |
| | ContraGAN [23] | 25.25 | 25.16 | 0.947 | 0.855 |
| | ReACGAN | **68.27** | **13.98** | **0.976** | **0.977** |
| | DiffAug-BigGAN [9] | 36.97 | 18.57 | 0.956 | 0.941 |
| | DiffAug-ReACGAN [9] | **69.74** | **11.95** | **0.977** | **0.975** |
| B.S.= 2048 | BigGAN* [4] | 99.31 | 8.51 | - | - |
| | BigGAN* [40] | 99.71 | **7.89** | 0.985 | 0.989 |
| | BigGAN* [25] | **104.57** | 9.18 | - | - |
| | ReACGAN | 92.74 | 8.23 | **0.991** | **0.990** |
| | Real Data | 173.33 | | | |




library [40]. We also conduct experiments with popular augmentation-based methods: consistency regularization (CR) [16] and differentiable augmentation (DiffAug) [9].

Compared with other cGANs, ReACGAN performs best on most of the benchmarks, surpassing the previous methods by 2.5%, 5.1%, 15.8%, and 14.5% in FIDs on CIFAR10, Tiny-ImageNet, CUB200, and ImageNet (256 B.S.), respectively. ReACGANs also harmonize with augmentation-based regularizations, bringing incremental improvements on all the metrics. For the ImageNet experiments using a batch size of 256, ReACGAN reaches higher IS and lower FID with fewer iterations than other models in comparison. Finally, we demonstrate that ReACGAN learns with a larger batch size on ImageNet. While some recent methods [20, 38, 27] have been built on ACGAN to improve the generation performance, large-scale image generation experiments with the batch size of 2048 have never been reported. Table 2 shows that our ReACGAN reaches FID score of 8.23 on ImageNet using the batch size of 2048, being comparable with the value of 7.89 from BigGAN implementation of StudioGAN. ReACGAN also provides better synthesis results than other implementations of BigGAN [4, 25]. Note that our results on ImageNet are obtained in only two runs while the training setup and architecture of BigGAN have been extensively searched and finely tuned.

**Comparison with other Conditioning Methods.** We investigate how the generation qualities vary with different conditioning losses while keeping the other configurations fixed. We compare our D2D-CE loss with cross-entropy loss of ACGAN (AC) [18], loss of projection discriminator (PD) [19], multi-hinge loss (MH) [26], and conditional contrastive loss (2C) [23]. The results are shown in Table 3. AC- and MH-based models present decent performances on CIFAR10, but undergo early-training collapse on Tiny-ImageNet and CUB200 datasets. Replacing them with PD, 2C, and D2D-CE losses produce satisfactory results across all datasets, where PD loss makes the best $F_8$ (recall) on CUB200 dataset while giving third $F_{0.125}$ (precision) value. The noticeable point is that the proposed D2D-CE shows consistent results across all datasets, showing the lowest FID and the highest $F_8$ and $F_{0.125}$ values in most cases. This means ReACGAN can generate high-fidelity images and is relatively free from the precision and recall trade-off [43] than the others.

**Consistent Performance of ReACGAN on Adversarial Loss Selection.** We confirm the consistent performance of ReACGAN on four adversarial losses: non-saturation loss [1], least square loss [10], wasserstein loss with gradient penalty regularization (W-GP) [13], and hinge loss [46] on CIFAR10 and Tiny-ImageNet datasets in Table 4. The experimental results show that BigGAN + D2D-CE (ReACGAN) consistently outperforms the projection discriminator (PD) and conditional contrastive loss (2C) counterparts over three adversarial losses [1, 13, 46]. However, for the experiments using the least square loss [10], ReACGAN exhibits inferior generation performances to the projection discriminator. We speculate that minimizing the least square distance between an adversarial logit and a target scalar (1 or 0) might affect the norms of feature maps and spoil the conditioning training performed by D2D-CE loss.

Table 3: Experiments on the effectiveness of D2D-CE loss compared to other conditioning losses.

| Conditioning Method | CIFAR10 [30] | | | | Tiny-ImageNet [32] | | | | CUB200 [33] | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ |
| BigGAN w/o Condition [4] (Abbreviated to Big) | 9.46 | 12.21 | 0.987 | 0.982 | 7.38 | 76.15 | 0.804 | 0.576 | 5.16 | 35.17 | 0.852 | 0.936 |
| Big + AC [18] | 9.84 | 8.45 | 0.993 | **0.992** | 6.00 | 96.04 | 0.656 | 0.368 | **6.09** | 60.73 | 0.726 | 0.891 |
| Big + PD [19] | 9.81 | 8.08 | 0.993 | **0.992** | 12.78 | 32.03 | 0.948 | 0.868 | 4.98 | 18.30 | 0.924 | **0.967** |
| Big + MH [26] | **10.05** | 7.94 | **0.994** | 0.990 | 4.37 | 140.74 | 0.282 | 0.156 | 5.18 | 245.69 | 0.625 | 0.832 |
| Big + 2C [23] | 9.70 | 8.22 | 0.993 | 0.991 | 13.46 | 28.55 | **0.974** | 0.881 | 5.34 | 21.16 | 0.935 | 0.942 |
| Big + D2D-CE (ReACGAN) | 9.89 | **7.88** | **0.994** | **0.992** | **14.06** | **27.10** | 0.970 | **0.894** | 4.91 | **15.40** | **0.970** | 0.954 |

Table 4: Experiments to identify the consistent performance of D2D-CE on adversarial loss selection.

| Adversarial Loss | Conditioning Method | CIFAR10 [30] | | | | | Tiny-ImageNet [32] | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ | Better? | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ | Better? |
| Non-saturation [1] | PD [19] | 9.75 | 8.29 | **0.993** | **0.991** | ✓ | 8.27 | 58.85 | 0.816 | 0.713 | |
| | 2C [23] | 9.30 | 10.47 | 0.990 | 0.959 | | 6.57 | 84.27 | 0.745 | 0.556 | |
| | D2D-CE | **9.79** | **8.27** | **0.993** | **0.991** | ✓ | **11.76** | **39.32** | **0.942** | **0.852** | ✓ |
| Least square [10] | PD [19] | **9.94** | **8.26** | **0.993** | **0.992** | ✓ | **12.74** | **37.14** | **0.920** | **0.900** | ✓ |
| | 2C [23] | 8.66 | 12.18 | 0.986 | 0.941 | | 9.58 | 53.10 | 0.916 | 0.706 | |
| | D2D-CE | 9.70 | 9.56 | 0.991 | 0.987 | | 9.50 | 57.67 | 0.848 | 0.692 | |
| W-GP [13] | PD [19] | 5.71 | 64.75 | 0.792 | 0.652 | | 6.67 | 84.16 | 0.696 | 0.498 | |
| | 2C [23] | 5.93 | 55.99 | 0.842 | 0.709 | | 6.89 | 74.45 | 0.812 | 0.536 | |
| | D2D-CE | **7.30** | **35.94** | **0.942** | **0.847** | ✓ | **8.92** | **52.74** | **0.856** | **0.689** | ✓ |
| Hinge [46] | PD [19] | 9.81 | 8.08 | 0.993 | **0.992** | | 12.78 | 32.03 | 0.948 | 0.868 | |
| | 2C [23] | 9.70 | 8.22 | 0.993 | 0.991 | | 13.46 | 28.55 | **0.974** | 0.881 | |
| | D2D-CE | **9.89** | **7.88** | **0.994** | **0.992** | ✓ | **14.06** | **27.10** | 0.970 | **0.894** | ✓ |

Table 5: FIDs [34] of conditional StyleGAN2 (cStyleGAN2) [7] and StyleGAN2 + D2D-CE (ReACGAN) on CIFAR10 [30] and AFHQ [41] datasets. ADA means the adaptive discriminator augmentation [8]. FID is exceptionally computed using the training split of the dataset for the reference moments since FID value of StyleGAN2 is often calculated using the moments of the training dataset.

| Conditioning method | CIFAR10 [30] | AFHQ [41] |
|---|---|---|
| cStyleGAN2 [7] | 3.88 | - |
| StyleGAN2 + D2D-CE | **3.34** | - |
| cStyleGAN2 + ADA [8] | 2.67 | 4.99 |
| StyleGAN2 + ADA + D2D-CE | **2.60** | **4.95** |
| StyleGAN2 + DiffAug + D2D-CE + Tuning | **2.26** | - |

**Effect of D2D-CE for Different GAN Architectures.** We study the effect of D2D-CE loss with different GAN architectures. In Table 5, we validate that D2D-CE is effective for StyleGAN2 [7] backbone and also fits well with the adaptive discriminator augmentation (ADA). StyleGAN2 with D2D-CE loss represents better generation results than the conditional version of StyleGAN2 (cStyleGAN2) by 13.9% on CIFAR10. Moreover, StyleGAN2 with D2D-CE can be reinforced with ADA or DiffAug when train StyleGAN2 + D2D-CE under the limited data situation. Among GANs, StyleGAN2 + DiffAug + D2D-CE + Tuning achieves the best performance on CIFAR10, outperforming some diffusion-based methods [47, 48]. Additional results with other architectures, i.e., a deep convolutional network [2] and a resnet style backbone [13], are provided in Appendix E.

### 4.4 Ablation Study

We study how the components of ReACGAN affect ACGAN training. By adding or ablating each part of ReACGAN, as shown in Table 6, we identify four major observations. (1) Feature and weight normalizations greatly stabilize ACGAN training and improve generation performances on Tiny-ImageNet and CUB200 datasets. (2) D2D-CE loss can enhance the generation performance by considering data-to-data relationships and by performing easy positive and negative suppression (4th row). (3) the suppression technique does not work well on the feature normalized cross-entropy loss (5th row). (4) While ACGAN shows high Inception score on ImageNet experiment, it shows

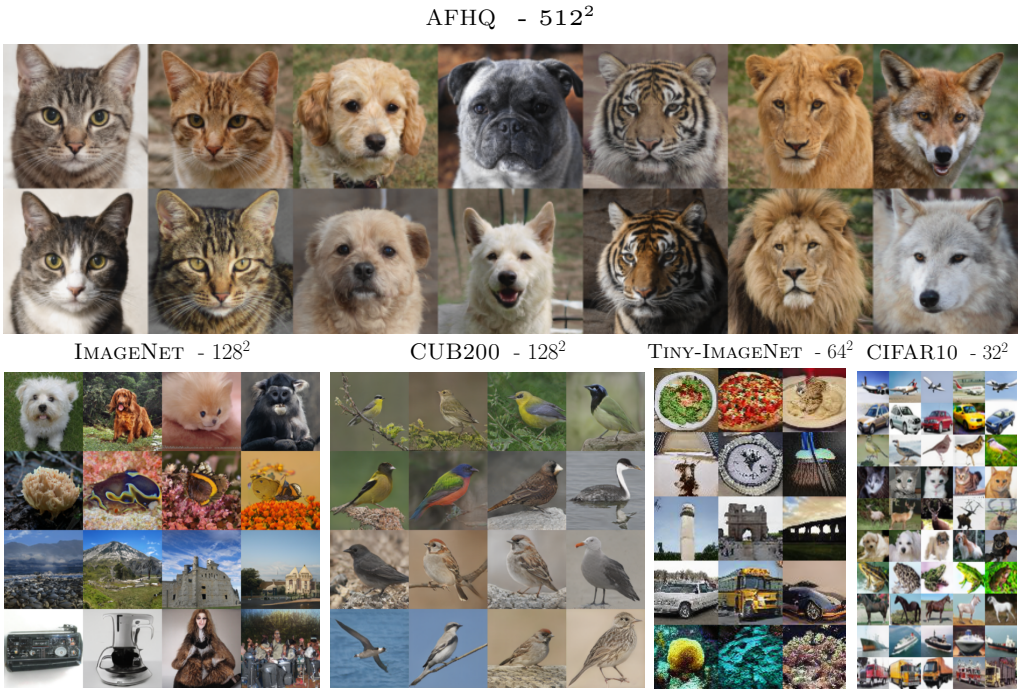



Figure 5: Curated images generated by the proposed ReACGAN. More qualitative results on ACGAN [18], BigGAN [4], ContraGAN [23], and our ReACGAN are attached in Appendix J.

Table 6: Ablation study on normalizing feature maps, data-to-data consideration, and easy positive and negative sample suppression.

| Ablation | Tiny-ImageNet [32] | | | | CUB200 [33] | | | | ImageNet [49] | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ | IS ↑ | FID ↓ | $F_{0.125}$ ↑ | $F_8$ ↑ |
| ACGAN [18] | 6.00 | 96.04 | 0.656 | 0.368 | **6.09** | 60.73 | 0.726 | 0.891 | 62.99 | 26.35 | 0.935 | 0.963 |
| + Normalization | 13.46 | 30.33 | 0.955 | 0.889 | 4.78 | 25.54 | 0.883 | 0.952 | 18.16 | 36.40 | 0.879 | 0.787 |
| + Data-to-data (Eq. 5) | 12.96 | 28.71 | 0.967 | 0.863 | 5.08 | 25.12 | 0.894 | 0.946 | - | - | - | - |
| + Suppression (Eq. 6) | **14.06** | **27.10** | **0.970** | **0.894** | 4.91 | **15.40** | **0.970** | **0.954** | **63.16** | **14.59** | **0.974** | **0.974** |
| - Data-to-data | 12.96 | 30.79 | 0.960 | 0.857 | 5.39 | 30.36 | 0.863 | 0.947 | - | - | - | - |

relatively low FID, $F_{0.125}$, and $F_8$ values compared with the model trained with normalization, data-to-data consideration, and the suppression technique. This is consistent with the qualitative results on ImageNet, where the images from ACGAN are easily classifiable, but images from ReACGAN are high quality and diverse. We attribute the improvement to the discriminator that successfully leverages informative data-to-data and data-to-class relationships with easy sample suppression.

## 5 Conclusion

In this paper, we have analyzed why training ACGAN becomes unstable as the number of classes in the dataset increases. By deriving the analytic form of gradients of the classifier and numerically checking the gradient values, we have discovered that the training instability comes from a gradient exploding problem caused by the unboundedness of input feature vectors and poor classification ability of the classifier in the early training stage. To alleviate the instability and reinforce ACGAN, we have proposed the data-to-data cross-entropy loss (D2D-CE) and Rebooted Auxiliary Classifier Generative Adversarial Network (ReACGAN). The experimental results verify the superiority of ReACGAN compared with the existing classifier- and projection-based GANs on five benchmark datasets. Moreover, exhaustive experiments demonstrate that ReACGAN is robust to hyperparameter selection and harmonizes with various architecture and differentiable augmentations.

## Acknowledgments and Disclosure of Funding

This work was supported by the IITP grants (No.2019-0-01906: AI Graduate School Program - POSTECH, No.2021-0-00537: Visual Common Sense, No.2021-0-02068: AI Innovation Hub) funded by Ministry of Science and ICT, Korea.

## References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2672–2680, 2014.

[2] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv preprint arXiv 1511.06434*, 2016.

[3] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[5] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4401–4410, 2019.

[6] Yan Wu, Jeff Donahue, David Balduzzi, Karen Simonyan, and Timothy P. Lillicrap. LOGAN: Latent Optimisation for Generative Adversarial Networks. *arXiv preprint arXiv 1912.00953*, 2019.

[7] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8110–8119, 2020.

[8] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training Generative Adversarial Networks with Limited Data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[9] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. *arXiv preprint arXiv 2006.10738*, 2020.

[10] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, Zhixiang Wang, and Stephen Paul Smolley. Least Squares Generative Adversarial Networks. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 2813–2821, 2017.

[11] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. *arXiv preprint arXiv 1701.07875*, 2017.

[12] Martín Arjovsky and Léon Bottou. Towards Principled Methods for Training Generative Adversarial Networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[13] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved Training of Wasserstein GANs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5767–5777, 2017.

[14] Naveen Kodali, James Hays, Jacob D. Abernethy, and Zsolt Kira. On Convergence and Stability of GANs. *arXiv preprint arXiv 1705.07215*, 2018.

[15] Zhiming Zhou, Jiadong Liang, Yuxuan Song, Lantao Yu, Hongwei Wang, Weinan Zhang, Yong Yu, and Zhihua Zhang. Lipschitz generative adversarial nets. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 7584–7593, 2019.

[16] Han Zhang, Zizhao Zhang, Augustus Odena, and Honglak Lee. Consistency Regularization for Generative Adversarial Networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[17] Zhengli Zhao, Sameer Singh, Honglak Lee, Zizhao Zhang, Augustus Odena, and Han Zhang. Improved Consistency Regularization for GANs. *arXiv preprint arXiv 2002.04724*, 2020.

[18] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional Image Synthesis with Auxiliary Classifier GANs. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 2642–2651, 2017.

[19] Takeru Miyato and Masanori Koyama. cGANs with Projection Discriminator. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[20] Mingming Gong, Yanwu Xu, Chunyuan Li, Kun Zhang, and Kayhan Batmanghelich. Twin Auxilary Classifiers GAN. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[21] Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening and Coloring Batch Transform for GANs. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[22] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. Few-Shot Unsupervised Image-to-Image Translation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019.

[23] Minguk Kang and Jaesik Park. ContraGAN: Contrastive Learning for Conditional Image Generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[24] Woohyeon Shim and Minsu Cho. CircleGAN: Generative Adversarial Learning across Spherical Circles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[25] Peng Zhou, Lingxi Xie, Bingbing Ni, and Qi Tian. Omni-GAN: On the Secrets of cGANs and Beyond. *arXiv preprint arXiv:2011.13074*, 2020.

[26] Ilya Kavalerov, Wojciech Czaja, and Rama Chellappa. A multi-class hinge loss for conditional gans. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021.

[27] Liang Hou, Qi Cao, Huawei Shen, and Xueqi Cheng. cGANs with Auxiliary Discriminative Classifier. *arXiv preprint arXiv:2107.10060*, 2021.

[28] Han, Ligong and Min, Martin Renqiang and Stathopoulos, Anastasis and Tian, Yu and Gao, Ruijiang and Kadav, Asim and Metaxas, Dimitris N. Dual Projection Generative Adversarial Networks for Conditional Image Generation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[29] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-Attention Generative Adversarial Networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 7354–7363, 2019.

[30] Alex Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. PhD thesis, University of Toronto, 2012.

[31] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[32] Johnson et al. Tiny ImageNet Visual Recognition Challenge. https://tiny-imagenet.herokuapp.com.

[33] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical report, California Institute of Technology, 2010.

[34] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 6626–6637, 2017.

[35] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 271–279, 2016.

[36] Akash Srivastava, Lazar Valkov, Chris Russell, Michael U Gutmann, and Charles A Sutton. VEEGAN: Reducing Mode Collapse in GANs using Implicit Variational Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[37] Mehdi Mirza and Simon Osindero. Conditional Generative Adversarial Nets. *arXiv preprint arXiv 1411.1784*, 2014.

[38] Zhiming Zhou, Han Cai, Shu Rong, Yuxuan Song, Kan Ren, Weinan Zhang, Jun Wang, and Yong Yu. Activation Maximization Generative Adversarial Nets. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[39] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A Simple Framework for Contrastive Learning of Visual Representations. *arXiv preprint arXiv 2002.05709*, 2020.

[40] Minguk Kang and Jaesik Park. Pytorch-studiogan. https://github.com/POSTECH-CVLab/PyTorch-StudioGAN, 2020.

[41] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. StarGAN v2: Diverse Image Synthesis for Multiple Domains. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[42] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved Techniques for Training GANs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2234–2242, 2016.

[43] Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[44] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A Learned Representation For Artistic Style. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[45] Harm de Vries, Florian Strub, Jeremie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C Courville. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 6594–6604, 2017.

[46] Jae Hyun Lim and Jong Chul Ye. Geometric GAN. *arXiv preprint arXiv 1705.02894*, 2017.

[47] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

[48] Dongjun Kim, Seungjae Shin, Kyungwoo Song, Wanmo Kang, and Il-Chul Moon. Score Matching Model for Unbounded Data Score. *arXiv preprint arXiv:2106.05527*, 2021.

[49] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

## 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] The main contributions of our paper can be summarized into two points: (1) analyzing why ACGAN training becomes unstable as the number of classes in the training dataset increases, (2) developing a new cGAN model: Rebooted Auxiliary Classifier Generative Adversarial Networks (ReACGAN). Both claims are well reflected in the abstract and introduction.
   (b) Did you describe the limitations of your work?
   [Yes] The proposed ReACGAN generates high-quality images using informative data-to-data and data-to-class relationships in the dataset. However, it requires additional hyperparameters tuning for temperature and margins values. Also we analyze the weakness of ReACGAN in Appendix F.
   (c) Did you discuss any potential negative societal impacts of your work?
   [Yes] Generative Adversarial Networks can be used to synthesize indistinguishable images from the real. This might cause fake news, sexual harnesses, and hacking machine vision applications. Although the potential negative social impacts are not discussed in our main paper, we deal with this issue in the Appendix G.
   (d) Have you read the ethics review and ensured that your paper conforms to them?
   [Yes] We carefully read the ethics review guidelines and comply with them.

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results?
   [Yes] We state all the assumptions that arise from the theoretical results, i.e., Property 1-4 in Sec. 3.3.
   (b) Did you include complete proofs of all theoretical results?
   [Yes] We provide the complete proofs in the Appendix D.
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] We make all the model weights and the source code available at https://github.com/POSTECH-CVLab/PyTorch-StudioGAN.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)?
   [Yes] We state the details for training setups in Sec. 4.2, which include the evaluation procedure, hyperparameter selection, and controlled environment for a fair comparison. In addition to this, we add how we pre-process the training dataset in the Appendix C.1.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)?
   [Yes] For reliable evaluation, we run all the experiments three times with different random seeds. We provide standard deviations of Table 1 in the Appendix I.
   (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] We include this information in the Appendix H.
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] We cite the papers and the GitHub repository that we used for our experiments in Sec 4.2.
   (b) Did you mention the license of the assets?
   [No] We utilize five datasets to validate our method: CIFAR10, Tiny-ImageNet, CUB200, ImageNet, and AFHQ. These are widely used public datasets, and we cite them in the paper. However, we could not find any license information for CIFAR10, Tiny-ImageNet, and CUB200. The ImageNet and AFHQ datasets can be used for non-commercial purposes only. In addition, we mention the license of PyTorch-StudioGAN in the main paper, which is used for our experiments.
   (c) Did you include any new assets either in the supplemental material or as a URL?
   [Yes] We provide the URL of our source code in the abstract.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating?
   [Yes] The CIFAR10, Tiny-ImageNet, CUB200, ImageNet, and AFHQ datasets are properly curated, and they are popular datasets for research purposes. To our best knowledge, there is no inappropriate content in the datasets.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content?
   [Yes] The benchmark datasets we used are widely adopted by others and do not contain any harmful or troublesome contents.
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] Our research has no relevance to the crowdsourcing or human subjects.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable?
   [N/A] Our research has no relevance to the crowdsourcing or human subjects.
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation?
   [N/A] Our research has no relevance to the crowdsourcing or human subjects.