StudioGAN: A Taxonomy and Benchmark of GANs for Image Synthesis

Minguk Kang, Member, IEEE, Joonghyuk Shin, Member, IEEE, and Jaesik Park, Member, IEEE

Abstract—Generative Adversarial Network (GAN) is one of the state-of-the-art generative models for realistic image synthesis. While training and evaluating GAN becomes increasingly important, the current GAN research ecosystem does not provide reliable benchmarks for which the evaluation is conducted consistently and fairly. Furthermore, because there are few validated GAN implementations, researchers devote considerable time to reproducing baselines. We study the taxonomy of GAN approaches and present a new open-source library named StudioGAN. StudioGAN supports 7 GAN architectures, 9 conditioning methods, 4 adversarial losses, 12 regularization modules, 3 differentiable augmentations, 7 evaluation metrics, and 5 evaluation backbones. With our training and evaluation protocol, we present a large-scale benchmark using various datasets (CIFAR10, ImageNet, AFHQv2, FFHQ, and Baby/Papa/Granpa-ImageNet) and 3 different evaluation backbones (InceptionV3, SwAV, and Swin Transformer). Unlike other benchmarks used in the GAN community, we train representative GANs, including BigGAN and StyleGAN series in a unified training pipeline and quantify generation performance with 7 evaluation metrics. The benchmark evaluates other cutting-edge generative models (e.g., StyleGAN-XL, ADM, MaskGIT, and RQ-Transformer). StudioGAN provides GAN implementations, training, and evaluation scripts with the pre-trained weights. StudioGAN is available at https://github.com/POSTECH-CVLab/PyTorch-StudioGAN.

Index Terms—Benchmark of generative models, realistic image synthesis, taxonomy of generative adversarial networks.

I. INTRODUCTION

Generative Adversarial Network (GAN) [1] is a well-known paradigm for approximating a real data distribution through an adversarial process. A GAN pipeline comprises a generator and a discriminator network. The discriminator tries to classify whether given samples are from the generator or the real data. On the other hand, the generator strives to generate realistic examples to induce the discriminator to misjudge the generated samples as real. By repeating the adversarial process, it is demonstrated that the generator can approximate the real data distribution [1].

Since Radford et al. [2] showed that deep convolutional GAN (DCGAN) can generate high-quality natural images, GAN has been actively adopted for a variety of computer vision applications, including natural image synthesis [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], image-to-image translation [12], [13], [14], [15], [16], [17], super-resolution imaging [18], [19], [20], [21], and generative neural radiance field [22], [23], [24], [25], [26]. Along with the increasing demand for developing a high-performance generative model, evaluating the quality of generated images has become an imperative procedure for GAN development. Inception score (IS) [27], and Fréchet Inception Distance (FID) [28] are the representative metrics for the generative model evaluation, and follow-up studies adopt the two metrics for the comparison.

While researchers put much effort into properly evaluating GAN, there are still inappropriate training configurations, and evaluation procedures are inconsistent between proposed approaches. For instance, some software libraries show unexpected behavior in data processing procedures. A quantization error occurs when converting training images to an hdf5 format file for fast data I/O. Similarly, buggy image resizing libraries can cause aliasing artifacts in images, affecting the quality of generated images [9], [29]. The lack of elaborately designed training and evaluation protocols followed by reliable benchmark, impede the community’s advancement in developing better generative models.

Generative model evaluation is mainly concerned with IS [27] and FID [28] where a pre-trained InceptionV3 [30] is used to extract the features of images. As stated by Kynkäänniemi et al. [31], FID can be hacked if information leakage from the InceptionV3 network takes place. This implies the importance of comprehensive benchmark with various metrics and evaluation backbones, such as IS, FID, Precision & Recall [32], and Density & Coverage [33], evaluated using different backbones, in order to thoroughly examine various aspects of models.

In this work, we introduce a new software library, named StudioGAN which has implementations of 30 representative GANs, covering from DCGAN [2] to StyleGAN3 [9] reproducing reported results in most cases. StudioGAN provides 7 GAN architectures, 9 conditioning methods, 4 adversarial losses, 12 regularization methods, and 7 evaluation metrics. StudioGAN allows training and evaluating representative GANs in the same environment, indeed with proper training and evaluation protocols for a fair comparison. Moreover, StudioGAN enables for Image Synthesis.
allows the change of evaluation backbone from InceptionV3 to modern networks, including SwAV [34], [35], DINO [36], and Swin-T [37]. As a result, we present an unparalleled amount of benchmark results instead of referring to the results from existing papers.

Note that there are several papers [38], [39], [40], [41], [42] that provide benchmark tables of generative models. In these papers, authors contain the evaluation results from previous papers without being fully aware that the original papers often utilize different training and evaluation protocols. Correspondingly, these practices still result in unfairness due to the disparities in data preprocessing, the employed machine learning library, and some unintentional mistakes by researchers. Recent research by Parmar et al. [29] established protocols to train and evaluate GANs in a more controlled setup. While their primary goal is to establish a protocol for fairer comparison between generative models, the benchmark they present is limited to models with open-source implementations and includes a re-evaluation of pre-trained models. This motivates us to establish an extensive benchmark that is controlled from the scratch-training to fair and clear evaluation. By providing identical training and evaluating conditions, we aim to provide a playground where researchers can easily implement their new ideas, followed by an extensive, fairly computed benchmark.

While the primary focus of our paper is to provide open-source implementations of GANs with an extensive benchmark, we also include a comparison between GANs and other generative models, such as diffusion models [43], [44], [45], [46], [47] and auto-regressive models [40], [41], [48] in Section VIII. GANs noticeably differ from diffusion and auto-regressive models regarding the number of feed-forward processes for generating a single sample. Therefore, we present the inference speed and the number of model parameters to help analyze the strengths and weaknesses of each model. For diffusion models or auto-regressive models, we use official implementations and apply the same evaluation protocol used for GAN evaluation.

In our extensive evaluation of GANs, we observe interesting findings on BigGAN, StyleGAN2, and StyleGAN3 characteristics. For example, we experimentally identify that StyleGAN2 tends to generate more diverse but lower quality images than BigGAN. However, we also witness that training StyleGAN2 and StyleGAN3 is more challenging than BigGAN if the targeted distribution has large inter-class variations. Furthermore, we discuss the potential hazards that can arise during a comparison with other models and add the future research direction at the end of the paper. To the best of our knowledge, this is the first attempt to provide a fairly established benchmark for generative modeling on such a substantial scale.

In summary, this paper has the following contributions.

- We present a taxonomy of GAN approaches based on five criteria (Section III).
- We pinpoint inappropriate practices that can cause poor generation results, reproducibility issues, and unfair evaluation (Section IV).
- We introduce StudioGAN, a software library that consists of 7 GAN architectures, 9 conditioning methods, 4 adversarial losses, 12 regularization modules, 3 differentiable augmentations, 7 evaluation metrics, and 5 evaluation backbones (Section VI). Through carefully assembling provided modules, researchers can design diverse and reproducible GANs, including 30 existing GANs.
- We provide large-scale evaluation benchmark for generative model development on standard and novel datasets (Section VII). Some interesting findings from modern generative models are presented (Section IV-A and VIII).

II. PRELIMINARY

Generative Adversarial Network: Goodfellow et al. [1] proposed a framework for generative modeling based on a two-player minmax game. Generative Adversarial Network (GAN) aims to estimate the distribution of an arbitrary data in high-dimensional space and consists of a generator and a discriminator network adversarial to each other.

Specifically, the generator $G: Z \rightarrow X$ strives to map a latent variable $z \sim p(z)$ into a sample $G(z)$ as if the sample is drawn from the real data distribution $p(x)$. The discriminator $D: X \rightarrow [0, 1]$, on the other hand, tries to discriminate whether the given sample comes from the real data distribution or the generator, becoming the adversary to the generator. Mathematically, the generative and discriminative processes can be expressed as follows:

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

(1)

The adversarial training between the two networks converges to Nash equilibrium under the two assumptions: (1) the two models have enough capacity, and (2) the discriminator is allowed to reach its optimum against the generator at each step. Under these conditions, implicit distribution of the generator $p_G(x) \rightarrow p(x)$, and generator becomes an efficient sampler for the true data distribution.

Evaluating GANs: To quantify the generated image’s quality without laborious human intervention, Inception Score (IS) [27] and Fréchet Inception Distance (FID) [28] were introduced and have been broadly adopted. Those metrics are computed on feature space, so all images involved in computing the metrics are mapped to the feature space using a pre-trained InceptionV3 [30] network. IS can be written as follows.

$$\text{IS}(x_t) = \text{Score}(x_t, F = \text{InceptionV3}) = \exp \left(\frac{1}{M} \sum_{i=1}^{M} D_{KL}(p(y|x_i)\|\hat{p}(y))\right),$$

(2)

where $X_t = \{x_1, \ldots, x_M\}$ is the image samples we target to evaluate, the posterior $p(y|x_i)$ is estimated by $F$ (InceptionV3), $D_{KL}$ is Kullback–Leibler divergence, and $\hat{p}(y)$ is an empirical class distribution: $\frac{1}{M} \sum_{i=1}^{M} p(y|x_i)$. FID is defined as follows.

$$\text{FID}(X_s, X_t) = \text{FD}(X_s, X_t, F = \text{InceptionV3}) = \|\mu_s - \mu_t\|_2 + \text{Tr} \left( \Sigma_s + \Sigma_t - 2(\Sigma_s \Sigma_t)^{\frac{1}{2}} \right),$$

(3)
where $\mu$ and $\Sigma$ are the mean vector and covariance matrix of the features from $F$, and the subscripts $s$ and $t$ indicate the source and target, respectively.

III. TAXONOMY OF GAN APPROACHES

Since the invention of the GAN framework, enormous efforts have been made to utilize GANs for realistic data generation. We present a GAN taxonomy based on five divisions: architecture, conditioning goal, adversarial loss, regularization, and data-efficient training. Fig. 1 shows the taxonomy of representative approaches.

A. GAN Architecture

When GAN was first introduced to the field of machine learning, training GAN was known to be extremely difficult. Dynamics between the generator and the discriminator often undergo a mode-collapse problem for unknown reasons, resulting in meaningless outputs. The first prominent results in GAN development came from adopting a deep convolutional neural network as a backbone architecture (DCGAN) [2] instead of multi-layer perceptron [1]. Although the quality is still far from being realistic, DCGAN is at least capable of producing recognizable images and significantly stabilizes the dynamics between the generator and discriminator. As deep learning and computing acceleration technologies advance, GAN benefits from larger and more complex network architectures [7], [8], [9], [48], [49], [50], [51], [52], [53], [54], [55], [56], [57], [58], [59], [60], [61], [62], [63], [64]. Gulrajani et al. [52] develop a ResNet-style generator and discriminator. Zhang et al. [49] show that adding self-attention layers into GAN can improve image generation performance (SAGAN). Brock et al. [7] scale up the ResNet-style architecture with hierarchical embedding (BigGAN and BigGAN-Deep). Apart from the above stream, Karras et al. have proposed a series of GAN architectures starting from Progressive GAN [53], to StyleGAN [54], StyleGAN2 [8], and StyleGAN3 [9]. The StyleGAN family specializes in generating images with a low inter-class variation using more inductive biases in the architecture structure than SAGAN and BigGAN. Beyond the narrow distribution modeling capabilities of StyleGAN-family models, Kang et al. [64] and Sauer et al. [65] develop GigaGAN and StyleGAN-T, featuring much bigger model capacity, and demonstrate GAN’s potential for a more generic image synthesis task, i.e., text-to-image synthesis.
and classification objective together. PD-GAN (Projection Discriminator GAN) [79] combines projected categorical embeddings with discriminator’s embeddings, so adversarial training is performed both on the discriminative process and on the categorical classification. ReACGAN [10] succeeds in the spirit of ACGAN and adopts data-to-data cross-entropy (D2D-CE) loss for class conditioning. The primary motivation for ReACGAN invention is ACGAN’s gradient exploding problem, and the exploding issue can be significantly alleviated by normalizing the discriminator’s features onto a unit hypersphere.

C. Adversarial Losses and Regularizations

The unstable nature of GAN dynamics is a major obstacle in high-quality image generation. To unearth the source of the instability, many researchers strive to provide interpretations of the unstable training issue and propose prescriptions in the form of adversarial loss and regularization. Arjovsky et al. [94], [95] state that this unstable characteristic attributes to the inherent gradient vanishing problem of the vanilla GAN loss. According to their papers, a vanilla GAN, which tries to minimize the Jensen-Shannon divergence between real and fake distributions, cannot provide helpful gradient signals to the discriminator if the distribution of fake images is far away from the real data distribution. To detour the gradient vanishing problem, Arjovsky et al. propose WGAN that minimizes 1-Wasserstein distance instead of the Jessen-Shannon criterion. Since training WGAN requires restricting the discriminator to a 1-Lipschitz function, several WGAN variants are proposed with the change of imposing the restriction: WGAN-GP [52], WGAN-DRA [96], and WGAN-LP [97]. On the other side, GANs based on Integral Probability Measure (IPM) [98] are proposed by many researchers. WGAN [94], McGAN [99], MMD GAN [100], Fisher GAN [101], Cramér GAN [102], and SphereGAN [103] are representative IPM-based GANs that require a restricted discriminator architecture. However, those models cannot produce successful results on large-scale image datasets, such as ImageNet [104]. Meanwhile, Miyato et al. [105] show successful results in ImageNet generation by devising spectral normalization, which bounds the gradient w.r.t an input image to 1.0. Consequently, modern GANs can be divided into two streams: (1) SNGAN [105], SAGAN [49], BigGAN [7], and ReACGAN [10] that require spectral normalization for stable training and (2) StyleGAN2 [8], and StyleGAN3 [9] that use R1 regularization [106] which also shrinks the norm of gradient significantly.

D. Techniques for Data-Efficient Training

Brock et al. [7] found that the discriminator of GAN tends to memorize the training dataset and provide an authenticity score for a given image, disregarding the realism of the generated images. This observation implies that if the training dataset is not enough, the discriminator can quickly memorize the training dataset, accelerating the training collapse. To this empirical discovery, data-efficient training has become one of the popular research trends in the GAN community. Researchers apply data augmentations on real and fake images to prevent the discriminator’s overfitting. Differentiable augmentation (DiffAug) [107], adaptive discriminator augmentation (ADA) [108], and adaptive pseudo augmentation (APA) [109] are some notable studies that deal with data augmentations for data-efficient learning. Unlike the augmentation-based approaches, Tseng et al. presents LeCam regularization [110], which transforms the WGAN training procedure to minimize LeCam divergence under mild assumptions. Since LeCam divergence is known to be more robust under the limited data situation than other f-divergence variants, GAN-imposed LeCam is expected to be data-efficient.

IV. INAPPROPRIATE PRACTICES

GAN approaches have been improved remarkably. However, there are inappropriate practices in data preprocessing and evaluation of GAN that can be boiled down to four points as follows:

- Adopting an improper image resizer and applying an unnecessary quantization process during the preprocessing step. (Section IV-A)
- The image resizer applied before the evaluation backbone (Feature Extractor, F) is not the same as the resizer used for training the evaluation backbone. (Section IV-B)
- Evaluating GAN only with IS and FID that are based on the pre-trained InceptionV3. (Sections IV-C and IV-D)
- Lacking the description of the type of normalization operation in batch normalization layers, although it can significantly affect the results. (Section IV-E)

Following the conclusions from these sections, we suggest protocols for training and evaluating GANs in Section V.

A. When Processing Training Images

The improper practices in preprocessing image data can be summarized into three points: (1) inherent aliasing artifacts in the training dataset due to improper resizing functions, (2) sampling error occurred by adopting a poor interpolation filter for both up- and down-sampling, and (3) quantization error that raises when converting a training dataset into other data I/O formats (i.e., HDF5).

We highlight less attention to avoiding aliasing artifacts in training images. As Parmar et al. [29] and Karras et al. [9] stated, processing training dataset using not only anti-aliasing but also a high-quality resizer is necessary for realistic image generation. However, most GAN approaches do not report what kind of resizer is used to preprocess images. By investigating original implementations, we identify that many methods use a bilinear interpolation for the image resizer, so resized images are occasionally of poor quality and exhibit aliasing artifacts. Furthermore, some studies utilize Tiny-ImageNet [111] dataset which contains aliasing and noise artifacts (Fig. A1).

This work proposes to use either bicubic [112] or Lanczos [113] interpolation for image up- and down-sampling to preprocess images. Interestingly, Parmar et al. [29] discovered that resizers implemented in popular TensorFlow [114], PyTorch [115], and OpenCV [116] libraries could cause aliasing artifacts in resulting images. According to this finding, we
Fig. 2. The schematic describes how real and fake images are processed during training and evaluation. Resizing and normalization operations are applied to the raw image in the (1) procedure. Then, the pre-processed and generated images are fed into the discriminator to perform adversarial training. When evaluating GAN, it is recommended to quantize both real and generated images to 8-bit representation to save images into a disk (2) and (3) as stated in [29]. Finally, the quantized images are resized and normalized before being fed to the evaluation backbone (4). We suggest using an architecture-friendly anti-aliasing resizer instead of directly adopting a high-quality resizing filter. Note that skipping the procedures (1) and (2) and processing the raw image using (5) procedure should not be allowed since it will make a discrepancy between the generator’s ideal distribution and the real data distribution so leading to biased evaluation.

B. When Processing Generated Images

We observe another hazard during the GAN evaluation. It attributes to the choice of image resizer applied to the generated images when feeding them to the evaluation backbone (i.e., TensorFlow-InceptionV3 to acquire FID and IS scores). Regarding this issue, Parmar et al. [29] confirmed that the improper resizing operation provokes aliasing artifacts, and they suggest clean-metric (clean-FID) by using the PIL.BICUBIC resizer [117].

In this work, we suggest using backbone friendly resizer for the evaluation. Our key finding is that the proper image resizer depends on the backbone network. For instance, feeding bicubic-interpolated images to the InceptionV3 network is improper since the InceptionV3 network was originally trained using a different resizer. Adopting the same resize filter used in training the TensorFlow InceptionV3 network cannot be the solution. This is because the InceptionV3 network was trained using the direct route (5) as depicted in Fig. 2 while real images for GAN should be resized using the anti-aliasing bicubic/Lanczos resizer through the route (1). This fact makes the entire data processing procedures of a recognition model and GAN inevitably different. Also, using a resizer implemented in TensorFlow for (5) method can induce aliasing artifacts as discovered by Parmar et al. [29], so both generated and real images may lose details. For that matter, we recommend using a resizer in the PIL library as suggested by Parmar et al. [29], but with a different interpolation filter applied.

To identify which resizer is more suitable for the specific backbone, we experiment with the assumption that the resizer is better if it allows better image classification accuracy. If the evaluation backbone shows better recognition ability, the backbone is likely to extract more disentangled features. The assumption correlates with the reason why InceptionV3 network is widely applied as an evaluation backbone network. The authors of IS [27] selected InceptionV3 [30] as a feature extractor since it was one of the cutting-edge models at that time.

In Table I, we measure Inception Score [27], Top-1, and Top-5 image classification accuracies on ImageNet [104] recommend using python imaging library (PIL) [117], such as PIL.BICUBIC or PIL.LANCZOS resizer.

Lastly, we discover that some popular approaches have an unnecessary or improper quantization process during the data preprocessing step. For example, software platforms built upon PyTorch-BigGAN [118] convert an image from PNG or JPEG to HDF5 format for fast I/O. The software platforms perform unnecessary normalization and wrong quantization operations during the conversion, raising quantization errors. Practitioners should skip the unnecessary normalization process and use a proper quantization operation to avoid the quantization error.

For example, if we use NumPy [119] framework for uint8 quantization, it is necessary to add 0.5 to each pixel since NumPy adopts the round-down operation for uint8 quantization. We express one of the possible solutions for image quantization in the NumPy environment as follows:

$$x_q = \text{clip}(127.5x_n + 128, 0, 255), \quad (4)$$

where clip(·, lower bound, upper bound) is a clipping function, $x_q$ is an image quantized in unit8 type, and $x_n$ is an normalized image with a pixel range of [-1.0, 1.0].
TABLE I

| RESIZER  | TensorFlow-InceptionV3 [30] | PyTorch-SwAV [34] | PyTorch-Swin-T [37] |
|----------|-----------------------------|-------------------|---------------------|
|          | IS † ∙ Top-1 † ∙ Top-5 †  | SS † ∙ Top-1 † ∙ Top-5 †  | TS † ∙ Top-1 † ∙ Top-5 †  |
| NEAREST  | 167.06 ∙ 67.88 ∙ 88.07    | 153.46 ∙ 59.71 ∙ 82.80    | 106.32 ∙ 75.35 ∙ 92.79    |
| BILINEAR | 182.57 ∙ 72.03 ∙ 90.09    | 276.67 ∙ 69.78 ∙ 89.46    | 61.88 ∙ 78.21 ∙ 94.81    |
| BICUBIC [114] | 178.45 ∙ 71.84 ∙ 89.80    | 273.76 ∙ 69.26 ∙ 88.99    | 71.70 ∙ 78.93 ∙ 95.16    |
| LANCZOS [115] | 173.79 ∙ 71.42 ∙ 89.53    | 254.81 ∙ 68.36 ∙ 88.37    | 74.55 ∙ 79.65 ∙ 95.27    |

We use imagenet [104] valid dataset to measure the metrics. We resize a raw image using PIL.LANCZOS resizer (procedure ①) and quantize the image (procedure ②). The numbers shown on the left side of the table (128, 256, and 512) indicate the resolution of the quantized image, and each image is resized once again to adjust the resolution for the evaluation backbone (procedure ④). The final resolution depends on the types of evaluation backbones: 299 for tensorflow-inceptionv3 [30], 224 for pytorch-swav [34], and 224 for pytorch-swin-t [37]. We measure image score (IS), swav score (SS), swin-transformer score (TS), top-1 accuracy, and top-5 accuracy. SS and TS have the same conceptual scores as is, except that the inceptionv3 backbone is replaced with swav and swin-t, respectively. The numbers in bold-faced denote the best performances, and gray colored numbers indicate architecture-friendly resizers for each evaluation backbone.

validation dataset with various image resizers applied for the procedure ④ in Fig. 2. Interestingly, the results are sensitive depending on the choice of image resizer, which indicates GAN comparison should be carefully designed. Experimental results also show that PIL.BILINEAR exhibits the coherent Inception Score (IS) and recognition accuracies in most cases for TensorFlow-InceptionV3 [30]. To see the effects of image resizer in modern backbone networks, we perform the same experiment with PyTorch-SwAV [34] and PyTorch-Swin-T [37]. In the case of PyTorch-SwAV, PIL.BILINEAR is considered a reasonable choice. One worth mentioning point is that processed images using PIL.NEAREST resizer gives exceptionally high Swin Transformer Scores (TS). For this reason, we weight Top-1 and Top-5 accuracies to select the proper resizer for Swin-T and decide to use PIL.BICUBIC resizer.

C. Backbone Networks for GAN Evaluation

Conventionally, almost all scores for generative model evaluation are computed using the features extracted from the TensorFlow-InceptionV3 network. Since Fréchet Distance (FD) measured using TensorFlow-InceptionV3 network (FID) is sample-inefficient [35] and is biased towards ImageNet classification task [31], it sometimes does not capture the discrepancy between distributions well. Therefore, we investigate GAN evaluation using a modern recognition backbone (Swin-T [37] and self-supervised backbones (SwAV [34] and DINO [36]).

Let $X_S$ and $X_T$ be the source and target datasets, and $X_1 \subset X_T$ be a subset of the target dataset. Then, relative-FD can be written as $\text{FD}(X_S, X_1)$ to analyze the sample efficiency of FD metrics across different backbones $F$. Fig. 3 shows relative-FD computed using pre-trained BigGAN models [7] on CIFAR10 [120] and ImageNet [104]. Lower is better because it indicates we can evaluate small number of generated images reasonably. The figure implies that the evaluated FDs using the InceptionV3 network are sample-inefficient compared to the other networks, as stated in [121]. Note that SwAV, DINO, and Swin-T give relatively sample-efficient results on CIFAR10 and ImageNet datasets.

To explore the effectiveness of FD, we visualize trends of real-to-real-FDs as the number of samples increases in Fig. 4. Using various backbone networks, we measure the feature distance between real images and the fraction of real images. Using the evaluation backbones, we calculate real images’ moments (mean and variance). Then, we sample fractions of the real images and calculate FDs. The real-to-real-FD can be written as $\text{FD}(X_S, X_S)$, where $X_S \subset X_S$. Unlike the results in Figs. 3 and 4 shows that DINO does not measure the real-to-real-FD efficiently. We speculate this phenomenon attributes to the property of DINO capturing the layout of a scene. As a result, we believe that SwAV, and Swin-T are good backbone candidates for GAN evaluation in addition to the InceptionV3 network.
D. Evaluation Metrics

Currently, GAN evaluation hugely depends on the value of Fréchet Inception Distance (FID) [28]. While FID has been verified by subsequent papers [7, 8, 9, 105], it has several limitations that FID informs the general ability of a generative model instead of the model’s specialties [32], [33], [122]. Also, FID is sometimes not coherent with human judgment [35], [60], [123]. To complement FID, Precision & Recall metrics [32], [33], [122] are proposed. However, those metrics are not as widely used as FID, and the criterion for computing the metrics has not been established concretely [63]. Therefore, we emphasize that evaluating GAN using various metrics with concrete criteria is necessary to analyze the various aspect of GAN.

E. Normalization Operations

We experimentally discover that the normalization operation of batch normalization layers can significantly affect the generation quality in testing time if the generator contains a batch normalization layer. Specifically, the generation results vary depending on the way of calculating statistics (mean and variance of features) for the normalization operation: (1) using average statistics proposed by Brock et al. [72], (2) adopting batch statistics [124], and (3) standing statistics [7]. The moving average statistics are computed as the moving average of batch statistics during training time, exactly how the batch normalization layer works in testing time. On the other hand, batch statistics work in the same fashion as batch normalization in training time. Unlike the previous two approaches, standing statistics propose by Brock et al. [7] compute the statistics by accumulating statistics of multiple batches (number of repetition) whose batch size is randomly determined from 1 to maximum batch size. As shown in Table II, (1) and (2) give poor synthesis outputs compared to (3) based on various metrics, including IS, FID, and Precision & Recall. In addition, when we evaluate GAN using (2), the generation performance differs according to the batch size, resulting in inconsistent evaluation. We suggest using the standing statistics for GAN evaluation.

F. Reproducibility

In addition to the previous issues, we observe that open-sourced GANs have some mistakes and missing statements. For instance, when evaluating a generative model during training for monitoring purposes, it is necessary to switch the generative model to the evaluation mode that freezes the moving average moments in the batch normalization layers. This prevents the model from cheating information in the validation dataset. In addition, many GAN approaches do not mention which dataset split is used for the evaluation. The lack of description induces misleading comparison since the metrics are sensitive to the reference distribution. Therefore, GAN approaches should clarify the dataset type (train, validation, and test) and the number of reference data for a fair evaluation. If a subset of a dataset is used in the assessment, providing the subset is required for reproducibility.

Moreover, training results dramatically change depending on the choice of hyperparameters, hidden details, and even the specific release version of machine learning framework (Caffe [125], TensorFlow [114], PyTorch [115], etc.). As a result, it is hard to reproduce the reported performance of GAN even with the authors’ official implementation. For example, the authors’ implementation of BigGAN [118] is not guaranteed to reproduce the experiment on ImageNet dataset [126], [127]. Moreover, image generation benchmarks on CIFAR10 [120] and ImageNet [104] are not compared carefully, so the reported performances in BigGAN [7], TACGAN [81], OmniGAN [84], ReACGAN [10], and ADCGAN [85] are not consistent, resulting in an unfair and improper comparison between models.

V. PROTOCOL SUGGESTION

With the discussions and observations in Section IV, it is obvious to build a standard protocol before making a fair comparison. Therefore, we propose the training and evaluation protocols to train and evaluate a variety of GAN models.

**Training Protocol:** When training a GAN model for high-quality image generation, it is important to keep the quality of training images as good as possible:

1. If an image resizer is required in training, use a high-quality resizer (e.g., PIL.BICUBIC and PIL.LANCZOS) that minimizes aliasing artifacts.
2. If there is a need to save processed images into a disk, keep the images with lossless compression (e.g. png) or with little lossy compression (e.g. high-quality jpeg).
3. Avoid applying any unnecessary quantization operation to the training dataset.

| Normalization          | IS ↑ | FID ↓ | Precision ↑ | Recall ↑ |
|------------------------|------|-------|-------------|----------|
| Moving average [126]   | 9.92 | 4.65  | 0.74        | 0.64     |
| Batch statistics [126] | 9.93 | 4.12  | 0.74        | 0.65     |
| Standing statistics [7] | 9.96 | 4.16  | 0.74        | 0.65     |
| Moving average [126]   | 100.69 | 10.23 | 0.74        | 0.58     |
| Batch statistics [126] | 87.62 | 10.66 | 0.73        | 0.60     |
| Standing statistics [7] | 98.51 | 8.54  | 0.75        | 0.60     |

We utilize a pre-trained biggan [7] and evaluate the model by changing a normalization operation.
Fig. 5. Generated images from GANs implemented in StudioGAN library. We select the best performing GAN on each dataset based on the average rank and generate fake images using those GANs. StyleGAN2 [8] is used for FFHQ [54], StyleGAN3-t + ADA [9] is used for AFHQv2 [9, 74], ReACGAN [10] is used for ImageNet, and StyleGAN2 + ADA [108] is used for CIFAR10 [120].

4) When evaluating a generative model during training, switch the generative model to freeze the batch normalization layers in the generative model.

**Evaluation Protocol:** When evaluating a GAN model, reproducibility and fairness are the crucial factors. Therefore, we present an evaluation protocol as follows:

1) Save generated images in the lossless png format to a disk using a proper image quantization operation.
2) Resize real and fake images using the backbone-friendly resizer described in Section IV-B.
3) Evaluation of a generative model should include FD (e.g. FID) and metrics that can quantify fidelity (e.g. Precision) and diversity (e.g. Recall) of generated samples [32, 33].
4) If the generator contains batch normalization layers, apply the standing statistics technique proposed by [7].
5) Clarify the dataset name and split (training, validation, and test) and the number of reference data for a fair evaluation.
6) Specify the version of the deep learning framework and the included library to reproduce the results.

**VI. STUDIOGAN**

This section introduces StudioGAN, a software library that provides reproducible implementations of representative GANs. StudioGAN is motivated by practical needs, where reproducibility has been a ball and chain around the ankle of GAN development. StudioGAN supports over 30 popular GANs and GAN-related modules based on the PyTorch framework (refer to the appendix for the full list, available online).

In short, StudioGAN has the following key features:

- **Coverage:** StudioGAN is a self-contained library that provides 7 GAN architectures, 9 conditioning methods, 4 adversarial losses, 13 regularization modules, 3 augmentation modules, 8 evaluation metrics, and 5 evaluation backbones. Among these configurations, we formulate 30 GANs as representatives (Fig. 5).

- **Flexibility:** Each modularized option is managed through a configuration system that works through a YAML file, so users can train a large combination of GANs by mismatching distinct options.

- **Reproducibility:** With StudioGAN, users can compare and debug various GANs with the unified computing environment without concerning about hidden details and tricks. Table III shows that StudioGAN successfully reproduces the most representative GANs on four benchmark datasets.

- **Versatility:** StudioGAN supports 5 types of acceleration methods with synchronized batch normalization for training: a single GPU training, data-parallel training,


### TABLE III

| Method          | Paper | Our StudioGAN Implementation |
|-----------------|-------|------------------------------|
| WGAN-GP [52, 107] | 6.68  | 40.1 | IS ↑ | FID ↓ | 6.71 (± 0.03) | 36.64 (± 3.26) |
| PD-GAN [81]     | 8.62  | 17.5 | 7.36 (± 1.36) | 29.79 (± 12.29) |
| SN-DGCAN [107]  | 7.82  | 29.3 | 8.20 (± 0.78) | 15.66 (± 13.64) |
| SNResGAN [81]   | 8.22  | 21.7 | 8.99 (± 0.76) | 9.51 (± 12.19) |
| BigGAN [7]      | 9.22  | 14.3 | 9.97 (± 0.75) | 4.16 (± 10.57) |
| StyleGAN [8]    | 9.53  | 6.96 | 10.28 (± 0.75) | 3.69 (± 3.27) |
| MEGAN [54, 110] | 9.58  | 6.40 | 10.13 (± 0.55) | 3.93 (± 2.47) |
| StyleGAN + ADA [10A] | 10.14 | 2.42 | 10.46 (± 0.32) | 2.31 (± 0.11) |
| CRGAN [38]      | -     | 11.4 | 10.38 | 7.18 (± 4.3) |
| IRCGAN [134]    | -     | 9.21 | 10.15 | 7.43 (± 1.78) |
| BigGAN-DiffAug [109] | 9.25  | 8.59 | 9.78 (± 0.53) | 7.16 (± 1.43) |
| BigGAN-LeCam [112] | 9.31  | 8.31 | 10.13 (± 0.82) | 7.36 (± 0.95) |
| ACGAN [79, 80]  | 8.25  | -    | 7.33 (± 0.92) | 31.72 |
| LOGAN [133]     | 8.67  | 17.7 | 7.66 (± 1.01) | 20.65 (± 2.95) |

#### VII. BENCHMARK

### A. Reproducibility of StudioGAN

Before presenting a benchmark, we examine whether our software library can reproduce evaluation results of well-known GAN papers. We utilize CIFAR10, ImageNet, AFHQv2, and FFHQ (FQ) datasets, along with Inception Score (IS) and Fréchet Inception Distance (FID), for the reproducibility check. StudioGAN generally presents superior image synthesis results in most cases, with the exceptions of PD-GAN [79], ACGAN [77], LOGAN [131], and SAGAN [49]. Investigating the performance discrepancies between original implementations and StudioGAN would give researchers insights into successful GAN training; however, pinpointing exact causes is extremely demanding, as implemented libraries, used datasets, and minor details vary from paper to paper. For example, StudioGAN seems to successfully replicate the results of SNGAN [105] on ImageNet but struggles to reproduce the results of SAGAN [49]. The only differences between SNGAN and SAGAN are the self-attention layers [134] and changed learning rate as suggested by the SAGAN paper [49]. We notice such small difference results in the reproducibility issue.

BigGAN [7] and StyleGAN-family models [8, 9, 54, 63, 108] have become the standard for GAN developments. Since these models have shown limited metrics and have little flexibility to mix-match diverse GAN training; however, pinpointing exact causes is extremely demanding, as implemented libraries, used datasets, and minor details vary from paper to paper. For example, StudioGAN uses different GAN backbones and recent training techniques.

### TABLE IV

| Backbone | Score | Prediction | Recall | Density | Coverage |
|----------|-------|------------|--------|---------|----------|
| InceptionV3 [30] | IS | FID | Precision | Recall | Density | Coverage |
| SwAV [34] | SS | FID | Precision | Recall | Density | Coverage |
| Swin-T [36] | TS | FID | T-Precision | T-Recall | T-Density | T-Coverage |

We provide a comprehensive GAN benchmark using the StudioGAN library. We train and evaluate GANs spanning primitive DCGAN [2] to recent StyleGAN3 [9], built upon four different types of backbone for the generator and the discriminator (DCCGAN, ResNetGAN, BigGAN, and StyleGAN). Some GANs (such as CRGAN [38] and DiffAug [107]) can be applied to multiple GAN backbones, but we use the same generator and discriminator architectures described in the original paper to provide evaluation results close to the original study.

BigGAN and StyleGANs are representative GANs for natural image synthesis. While those models are being actively studied, BigGAN and StyleGANs tend to be studied separately in the literature because StyleGANs are designed for unconditional image generation, whereas BigGAN is designed for conditional image generation. To compare those models in a fair way, we adopt the class conditioning version of StyleGAN2 [108], StyleGAN3-t [9], and StyleGAN3-r [9] for all experiments except for the cases on AFHQv2 and FFHQ datasets in Tables V, A3, and A4. Because of the poor convergence issue, StyleGAN3-r results on ImageNet are omitted from Tables V.
| Tensorflow-InceptionV3 [30] | IS ↑ | FID ↓ | Precision ↑ | Recall ↑ | Density ↑ | Coverage ↑ | IFID ↓ |
|-----------------------------|------|-------|--------------|---------|-----------|-----------|--------|
| DCGAN [11] | 6.11 | 48.29 | 0.61 | 0.21 | 0.52 | 0.28 | 117.00 |
| LSGAN [147] | 6.55 | 40.22 | 0.60 | 0.34 | 0.49 | 0.35 | 111.31 |
| Geometric GAN [148] | 6.61 | 43.16 | 0.58 | 0.25 | 0.48 | 0.31 | 114.59 |
| ACGAN-Mod [79] | 7.19 | 33.39 | 0.61 | 0.21 | 0.52 | 0.36 | 72.75 |
| WGAN [97] | 3.59 | 107.68 | 0.45 | 0.02 | 0.29 | 0.08 | 158.11 |
| DRAGAN [98] | 6.59 | 34.69 | 0.63 | 0.47 | 0.56 | 0.36 | 106.46 |
| WGAN-GP [52] | 5.22 | 53.98 | 0.68 | 0.26 | 0.67 | 0.27 | 119.22 |
| FD-GAN [81] | 7.25 | 31.54 | 0.61 | 0.28 | 0.53 | 0.39 | 65.04 |
| SNGAN [107] | 8.83 | 90.01 | 0.70 | 0.62 | 0.79 | 0.75 | 22.85 |
| SAGAN [49] | 8.64 | 10.35 | 0.69 | 0.63 | 0.73 | 0.71 | 25.00 |
| TACGAN [83] | 7.68 | 29.51 | 0.62 | 0.28 | 0.57 | 0.41 | 64.52 |
| LOGAN [133] | 7.89 | 20.45 | 0.65 | 0.63 | 0.61 | 0.54 | 95.82 |

| CIFAR10 | StyleGAN2 [8] | 10.17 | 3.78 | 0.72 | 0.67 | 0.97 | 0.89 | 14.04 |
| Neighbor | StyleGAN2 + DiffAug [8], [109] | 10.55 | 3.29 | 0.74 | 0.68 | 1.02 | 0.92 | 11.56 |
| Baby-ImageNet | StyleGAN2 + ADA [110] | 10.53 | 3.21 | 0.75 | 0.69 | 1.04 | 0.93 | 11.82 |
| Papa-ImageNet | StyleGAN2 + LeCam [8], [112] | 10.04 | 4.65 | 0.74 | 0.66 | 0.90 | 0.86 | 15.66 |
| StyleGAN2 + APA [111] | 10.05 | 4.04 | 0.70 | 0.68 | 0.89 | 0.87 | 54.40 |
| StyleGAN2 + D2D-CE [10] | 10.43 | 3.44 | 0.74 | 0.64 | 1.02 | 0.90 | 14.14 |
| StyleGAN2+ + ADA [9] | 9.72 | 10.83 | 0.68 | 0.53 | 0.84 | 0.75 | 27.32 |

| Baby-ImageNet | SNGAN [107] | 18.21 | 36.31 | 0.54 | 0.58 | 0.38 | 0.39 | 93.10 |
| Grandpa-ImageNet | SAGAN [49] | 16.47 | 42.10 | 0.50 | 0.53 | 0.33 | 0.32 | 111.11 |
| Papa-ImageNet | BigGAN [7] | 27.47 | 18.64 | 0.63 | 0.47 | 0.59 | 0.57 | 62.22 |
| ImageNet | ContraGAN [85] | 23.14 | 24.67 | 0.62 | 0.47 | 0.53 | 0.45 | 220.72 |
| FQ | ReACGAN [10] | 26.18 | 19.75 | 0.63 | 0.43 | 0.54 | 0.50 | 77.95 |
| FQ | StyleGAN2 [8] | 23.40 | 22.21 | 0.62 | 0.59 | 0.58 | 0.62 | 70.83 |
| FQ | StyleGAN2+ + ADA [9] | 9.99 | 102.24 | 0.37 | 0.01 | 0.17 | 0.06 | 235.20 |

| ImageNet | SNGAN [107] | 15.04 | 41.06 | 0.52 | 0.55 | 0.38 | 0.37 | 107.13 |
| Baby-ImageNet | SAGAN [49] | 12.86 | 51.55 | 0.48 | 0.53 | 0.31 | 0.27 | 130.10 |
| Grandpa-ImageNet | BigGAN [7] | 21.62 | 24.99 | 0.60 | 0.47 | 0.54 | 0.53 | 79.35 |
| Papa-ImageNet | ContraGAN [85] | 19.22 | 23.33 | 0.62 | 0.48 | 0.54 | 0.46 | 203.33 |
| ImageNet | ReACGAN [10] | 21.80 | 21.74 | 0.62 | 0.47 | 0.55 | 0.51 | 91.76 |
| FQ | StyleGAN2 [8] | 19.28 | 23.28 | 0.58 | 0.61 | 0.52 | 0.58 | 76.06 |
| FQ | StyleGAN2+ + ADA [9] | 9.45 | 48.74 | 0.47 | 0.00 | 0.34 | 0.11 | 234.71 |

| ImageNet | SNGAN-256 [107] | 31.42 | 31.49 | 0.56 | 0.67 | 0.43 | 0.39 | 112.68 |
| Baby-ImageNet | BigGAN-256 [7] | 40.78 | 20.37 | 0.66 | 0.65 | 0.63 | 0.52 | 121.32 |
| Grandpa-ImageNet | ContraGAN-256 [85] | 25.23 | 29.59 | 0.59 | 0.52 | 0.62 | 0.31 | 162.80 |
| ImageNet | ReACGAN-256 [10] | 66.67 | 15.65 | 0.77 | 0.38 | 0.88 | 0.53 | 127.31 |
| FQ | StyleGAN2 [8] | 22.54 | 3.40 | 0.54 | 0.63 | 0.40 | 0.34 | 135.19 |
| FQ | StyleGAN2+ + ADA [110] | 11.94 | 6.08 | 0.85 | 0.49 | 1.32 | 0.81 | 6.02 |
| FQ | StyleGAN2+ + ADA [9] | 11.70 | 4.75 | 0.83 | 0.58 | 1.20 | 0.83 | 4.73 |
| FQ | StyleGAN2+ + ADA [9] | 11.63 | 4.33 | 0.81 | 0.68 | 1.05 | 0.84 | 4.39 |
| FQ | StyleGAN2 [8] | 11.74 | 4.62 | 0.81 | 0.66 | 1.04 | 0.84 | 4.35 |

| FQ | StyleGAN2 [8] | 5.27 | 3.16 | 0.77 | 0.60 | 1.09 | 0.90 | - |

The resolutions of CIFAR10 [120], baby/papa/grandpa imagenet, imagenet [104], AFHQv2 [9], [74], and FQ [54] datasets are 32, 64, 128, 512, and 1024, respectively.
A3, and A4. Here, we utilize InceptionV3, SwAV, and Swin-T for the evaluation backbone to avoid biased conclusions.

C. Datasets

We use four standard datasets for the conditional image generation: CIFAR10 [120], ImageNet [104], AFHQv2 [9], [74], and FFHQ [54] and three of our proposed ImageNet subsets: Baby/Papa/Grandpa-ImageNet. We create the subsets of ImageNet for a small-scale ImageNet experiment because training GAN on the full ImageNet requires extensive computational resources. The proposed subsets of the ImageNet are made according to the classification difficulty, enabling researchers to analyze behaviors of conditional GANs based on the level of classification difficulty. Baby-ImageNet contains the easiest images to classify, whereas the Grandpa-ImageNet contains the hardest ones. We attach a detailed explanation of datasets in the appendix, available online. Tiny-ImageNet [111] is also intended for the small-scale ImageNet experiment, but we decided not to use it for the benchmark because Tiny-ImageNet is known to have severe aliasing artifacts and degradation.

D. Evaluation Metrics

We utilize 7 evaluation metrics: Inception Score (IS) [27] (2), Fréchet Inception Distance (FID) [28] (3), Precision & Recall [32], Density & Coverage [33], and Intra-class FID (IFID) [49] that is the average of class-wise FID. Each metric is computed using TensorFlow-InceptionV3 network in which weights are borrowed from Improved-GAN GitHub [136]. We use PyTorch implementations of IS (we implemented), FID [137], Precision & Recall [138], and Density & Coverage [138] to construct seamless training and evaluation procedures.

Precision & Recall are used to quantify the fidelity and diversity of generated images based on the estimated support of real and fake distributions. Precision is defined as the portion of randomly generated images that falls within the support of real data distribution. On the other hand, Recall is the portion of real images falling within the support of fake data distribution. High Precision and Recall mean that the generated images are high-quality and diverse. Mathematically, Precision & Recall can be written as follows:

\[
\text{Precision}(X_s, X_t, F, k) := \frac{1}{M} \sum_{i=1}^{M} \mathbb{1} \left( f_{t,i} \in \mathcal{M}(F(X_s), k) \right),
\]

(5)

\[
\text{Recall}(X_s, X_t, F, k) := \frac{1}{N} \sum_{i=1}^{N} \mathbb{1} \left( f_{s,i} \in \mathcal{M}(F(X_t), k) \right),
\]

(6)

where \( \mathcal{M}(F, k) := \bigcup_{f \in F} B(f, NN_k(F, f, k)) \) is a collection of features from \( F \), \( X_s = \{x_{s,1}, \ldots, x_{s,N}\} \) is a source dataset (real images), \( X_t = \{x_{t,1}, \ldots, x_{t,M}\} \) is a target dataset (generated images), \( B(f, r) \) is the n-dimensional sphere in which \( f = F(x) \in \mathbb{R}^n \) is the center and \( r \) is the radius, \( \mathbb{1}(\cdot) \) is an indicator function.

E. Findings From the Evaluation Results

Tables V, A3, and A4 show quantitative results using InceptionV3, SwAV, and Swin-T, respectively. Thin solid lines are used to group GANs with the similar training backbone (DCGAN, ResNet, BigGAN, and StyleGAN). We also visualize the quantitative results in Figs. A3, A4, A5, A6, A7, A8, A9, A10, and A11 to analyze results better. For the details about the experiment setup, please refer to the appendix, available online.

We provide our key findings from the experiments below.

Rooms for Improvement in CIFAR10: CIFAR10 experiment is a starting point for generative model development and has been highly explored by numerous GANs. Interestingly, we find that the generation performances of representative GANs

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
can still be improved with our StudioGAN implementations. Although FID has several drawbacks mentioned in Section IV-D, it has long been conventionally adopted as a primary metric for evaluating generative models. Thus we decide to compare FID in this section, as a lot of existing papers only present FID scores. As can be seen in Tables III and IV, FID values of SNGAN, BigGAN, StyleGAN2, MHGAN, CRGAN, ICRGAN, and BigGAN-DiffAug are significantly lower than the originally reported numbers. We believe StudioGAN achieves better training dynamics with numerically stable spectral normalization implemented in PyTorch and dedicated hyperparameter selection.

**StyleGAN2 Can Generate More Diverse Images Compared to BigGAN**: Experiments using Baby/Papa/Grandpa-ImageNet shown in Table V and A3 present that StyleGAN2 models are apt to exhibit higher recall and coverage values than BigGAN counterparts. Figs. A9 and A10 in Appendix, available online verify the statement again by showing the orange markers (StyleGAN2) are located above (high recall and coverage) the red markers (BigGAN). However, based on a few examples, we notice that the recall and coverage values of StyleGAN2 are sometimes lower than those of BigGAN. For example, StyleGAN2 trained using ImageNet shows lower density (BigGAN: 0.65 → StyleGAN2: 0.63) and coverage (BigGAN: 0.52 → StyleGAN2: 0.34) than BigGAN as can be seen in Table V. We speculate that this is because StyleGAN’s generator design is too restrictive to model complex data distributions like ImageNet.

**StyleGAN Family Struggles With ImageNet**: As shown in Tables V and A3, StyleGAN2 and StyleGAN3-t present FID values of 33.4 and 36.51 and FSD values of 5.61 and 6.07 on ImageNet generation experiments, respectively, which are worse than those of BigGAN, ContraGAN, and ReACGAN. In contrast, StyleGAN2 has a better synthesis ability than BigGAN (refer to Table A1) with AFHQv2 dataset. Such results imply that StyleGAN2 and StyleGAN3-t are tailored to center-aligned and high-resolution images, and the StyleGAN-family models are not good at generating images having high inter-class variations. This observation becomes more evident when comparing the FID values of cGAN models trained on subsets of ImageNet (Baby/Papa/Grandpa-ImageNet) and the original ImageNet. StyleGAN2 models trained on Baby-, Papa-, and Grandpa-ImageNet achieve FID scores of 22.21, 23.28, and 21.88, respectively, which are comparable to or even better than those achieved by BigGAN. However, when StyleGAN2 is trained on the full ImageNet, it exhibits a significantly higher FID score (33.40) compared to other cGAN models (BigGAN: 20.57, ContraGAN: 29.59, and ReACGAN: 15.65). This indicates that training conditional StyleGAN2 becomes increasingly challenging as the number of classes in the training data increases.

**Fréchet SwAV Distance Favor StyleGAN2 Over BigGAN**: As can be seen in Tables V and A3, we discover that FID and FSD show different preferences across BigGAN, ContraGAN, ReACGAN, and StyleGAN2. Specifically, SwAV tends to take the side of StyleGAN2 while InceptionV3 is apt to take the side of BigGAN and ReACGAN. We believe that StyleGAN variants can generate more realistic images because Morozov et al. [35] demonstrates that FSD and Precision & Recall measured through SwAV provide more consistent evaluation results with human evaluation. However, given that FID [28] is still broadly adopted, we think that further analysis of FID, FSD, Precision & Recall, and S-Precision & S-Recall is necessary.

**Image Generation Quality versus Image Classification Accuracy**: As shown in Table V, evaluation results using the InceptionV3 network on Baby/Papa/Grandpa-ImageNet imply no strong correlation between classification difficulty and generation quality. However, Table A3 shows interesting trends that the FSD value of the generated images gets lower if training images are easily classifiable. For instance, FSD values of BigGAN, ContraGAN, ReACGAN, and StyleGAN on Baby-ImageNet are 4.49, 4.47, 4.18, and 3.43, while FSD values of those models on Papa-ImageNet are 5.3, 4.62, 4.3, and 3.7. This relationship holds for Grandpa-ImageNet too. Choosing the most suitable evaluation backbone is an open question, so we do not judge that well-classifiable images are well-generated images. However, we value the correlation between FSD and classification difficulty and believe it is worth exploring in the future.

**F. Potential Hazards**

So far, we have described an extensive benchmark based on our newly suggested evaluation protocol. However, as we look over the results and inspect the details of each metric, we found some unclear points and the potential danger of misleading metrics. In this section, we elaborate few potential hazards for GAN community to be aware of in future research.

**Evaluation Results are Highly Affected by Evaluation Backbone**: As shown in Tables V, A3, and A4, the rank of each metric varies as InceptionV3 [30], SwAV [34], or Swin-T [37] is used as an evaluation backbone. InceptionV3 network was adopted as an evaluation backbone because there was a belief that a cutting-edge recognition model is useful for extracting good image features. Still, there is no guarantee that InceptionV3 is the best choice for generative model evaluation. Therefore, we encounter a potential hazard that the evaluation using Inception Score (IS), Fréchet Inception Distance (FID), Precision & Recall, and Density & Coverage can be biased toward the pre-trained InceptionV3 model. Therefore, future studies should be conducted to identify which type of an evaluation backbone is appropriate for generative model evaluation.

**IFID May Lead to Wrong Conclusion**: Intra-class FID (IFID) is the average of class conditional FIDs and is used to measure intra-class fidelity, diversity, and class conditioning performance in a scalar value. As Sajjadi et al. [122] stated, quantifying multiple characteristics into a scalar leads to confusing conclusions, and independently measuring each characteristic is required. For example, ContraGAN exhibits a lower FID value (21.14) than BigGAN (24.4) in the Grandpa-ImageNet generation experiment. However, the IFID value of ContraGAN (193.93) is exceptionally high compared to the IFID value of BigGAN (78.11). In Table VI, we notice that the exceptionally high IFID...
While conditional GAN (cGAN) is devised to is proposed to quantify class-conditional [144].

VI. APPROACHES

VII. COMPARATIVE STUDIES

The expectation that classifier-based GANs would perform better in class conditioning than ReACGAN, which differs from our experiment. While StyleGAN-XL achieves the lowest FID value compared to other models, ADM-G-U presents higher precision and recall scores than StyleGAN-XL. This contradicts our expectation that higher precision and recall scores should be correlated with a lower FID. We leave the analysis on this matter as a future work because a more careful and thorough analysis should take place to understand this corner case of evaluation metrics.

StyleGAN-XL: StyleGAN-XL [63] tackles the limitation of the StyleGAN approaches that struggle with generating a diverse class of images, such as ImageNet. In the CIFAR10 generation experiment, StyleGAN-XL exhibits the FID of 1.88, being first to break the wall of 2. Despite the record, StyleGAN-XL shows the lowest recall value of 0.59 compared to the other competitive models. This phenomenon is unnatural since images with a low recall are expected to exhibit a high FID.

A similar phenomenon is observed in the ImageNet-256 experiment. While StyleGAN-XL achieves the lowest FID value compared to other models, ADM-G-U presents higher precision and recall scores than StyleGAN-XL. This contradicts our expectation that higher precision and recall scores should be correlated with a lower FID. We leave the analysis on this matter as a future work because a more careful and thorough analysis should take place to understand this corner case of evaluation metrics.

Auto-Regressive (AR) and Diffusion Models: In Table VII, RQ-Transformer [41] shows the second best FID on ImageNet-256 followed by diffusion models ADM-G and ADM-G-U [46]. While StyleGAN-XL and RQ-Transformer exhibit lower FID values than ADM-G-U, we empirically confirm that ADM-G-U synthesizes more visually plausible images than StyleGAN-XL and RQ-Transformer. As can be seen in Fig. 6 and Figs. A12, A13, and A14 in Appendix, available online, the generated images from ADM-G-U appear to reflect the overall structure of objects more faithfully than StyleGAN-XL and RQ-Transformer. To our knowledge, ADM-G-U is the first model to generate reasonable facial images among ImageNet-trained generative models.

Comparing GANs With AR and Diffusion Models: Our experimental results suggest that AR and diffusion models show comparable or slightly worse FIDs with StyleGAN-XL (Table VII). However, the number of model parameters in popular GAN models, such as BigGAN-Deep and StyleGAN-XL, is much smaller than AR and diffusion models. For instance, the number of parameters of BigGAN-Deep is 112 M, but ADM-G and RQ-Transformer’s parameters are 608 M and 3.9B, respectively. This implies that GANs are parameter-efficient and can be scaled up to overcome the poor structural coherency [64]. Furthermore, unlike AR and diffusion models, GANs do not require a consecutive inference process, enabling very fast sampling. For example, while it takes only 0.040 seconds to generate one sample with...
1. **StyleGAN-XL [63] on ImageNet (256 × 256)**

StyleGAN-XL, it takes up to 49.609 seconds with ADM-G-U (1000 DDPM). While there have been efforts to reduce denoising steps [148], [149], these models require a well-trained diffusion model for distillation. Moreover, the synthesis ability of the few-step diffusion models still lags behind the state-of-the-art GAN approach [64]. Hence, we emphasize that GANs are highly efficient regarding model size, inference speed, and synthesis ability.

2. **RQ-Transformer [41] on ImageNet (256 × 256)**

3. **ADM-G-U [46] on ImageNet (256 × 256)**

Fig. 6. Generated images by StyleGAN-XL (GAN) [63], RQ-Transformer (Auto Regressive Model) [41], and ADM-G-U (Denoising Diffusion Probabilistic Model) [46]. More qualitative results are displayed in Figs. A12, A13, and A14.

### TABLE VII

**Benchmark Table for GANS [7], [9], [10], [48], [63], Autoregressive Models [40], [41], [48], and Diffusion Probabilistic Models [43], [44], [45], [46], [47]**

| Model                     | # Param. | IS † | FID ‡ | Precision † | Recall † | Density † | Coverage † | Inf. (s) |
|---------------------------|----------|------|-------|-------------|----------|-----------|------------|---------|
| ReACGAN + DiffAug (Ours) [10] | 9.4M     | 10.15 | 2.64  | 0.75        | 0.65     | 0.96      | 0.90       | 0.009   |
| StyleGANZ2-ADA [85]       | 20.2M    | 10.31 | 2.41  | 0.74        | 0.64     | 1.02      | 0.92       | 0.008   |
| StyleGANZ2-ADA (Ours) [63]| 20.2M    | 10.53 | 2.31  | 0.75        | 0.69     | 1.04      | 0.93       | 0.008   |
| StyleGANZ2 + DiffAug + D2A-CE (Ours) [10] | 20.2M | 10.46 | 2.30  | 0.76        | 0.68     | 1.03      | 0.93       | 0.007   |
| DDPM [43]                 | 35.2M    | 9.73  | 3.23  | 0.78        | 0.67     | 1.10      | 0.93       | 15.22   |
| DDPM++ [44]               | 106.6M   | 9.90  | 2.49  | 0.78        | 0.69     | 1.12      | 0.94       | 46.69   |
| NCSN++ [44]               | 107.6M   | 10.08 | 2.27  | 0.77        | 0.70     | 1.07      | 0.94       | 99.304  |
| LSCM [45]                 | -        | 10.04 | 2.80  | 0.80        | 0.70     | 1.15      | 0.95       | -       |
| LSCM-ODE [45]             | -        | 10.07 | 2.09  | 0.77        | 0.71     | 1.03      | 0.94       | -       |
| CLD-SGM [47]              | -        | 9.88  | 2.38  | 0.78        | 0.69     | 1.12      | 0.94       | -       |
| StyleGAN-XL [63]          | 18.0M    | 11.03 | 1.88  | 0.77        | 0.59     | 1.08      | 0.94       | 0.010   |

**ImageNet-256**

| BigGAN [7]†               | 70M      | 128.51 | 7.66   | 0.80        | 0.51     | 1.16      | 0.84       | 0.139   |
| BigGAN (Ours) [7]†        | 70M      | 98.51  | 8.54   | 0.75        | 0.60     | 0.96      | 0.82       | 0.015   |
| BigGAN-Deep [7]†          | 50M      | 161.73 | 5.90   | 0.89        | 0.45     | 1.66      | 0.91       | 0.086   |
| ReACGAN (Ours) [10]†      | 70M      | 94.06  | 8.19   | 0.80        | 0.42     | 1.07      | 0.72       | 0.012   |
| StyleGAN-XL [63]†‡†       | 159M     | 221.04 | 1.94   | 0.82        | 0.62     | 1.17      | 0.94       | 0.030   |
| ADM-G [46]†               | 464M     | 156.22 | 3.05   | 0.81        | 0.66     | 1.13      | 0.95       | 23.513  |

**ImageNet-128**

| BigGAN [7]†               | 164M     | 145.52 | 7.75   | 0.83        | 0.47     | 1.34      | 0.85       | 0.324   |
| BigGAN-Deep [7]†          | 112M     | 224.46 | 6.95   | 0.89        | 0.36     | 1.47      | 0.88       | 0.177   |
| VOGAN [48]                | 1.3B     | 314.61 | 5.20   | 0.81        | 0.57     | 1.03      | 0.85       | 6.551   |
| StyleGAN-XL [63]†‡†       | 164M     | 297.62 | 2.32   | 0.82        | 0.61     | 1.16      | 0.93       | 0.040   |
| ADM-G [46]†               | 608M     | 207.86 | 4.49   | 0.84        | 0.62     | 1.19      | 0.94       | 50.574  |
| ADM-G-U [46]†             | 679M     | 240.24 | 4.01   | 0.85        | 0.62     | 1.22      | 0.95       | 49.609  |
| MarkGIT [40]†             | 227M     | 216.38 | 5.40   | 0.87        | 0.60     | 1.37      | 0.94       | 5.313   |
| RQ-Transformer [41]†       | 3.9B     | 339.41 | 3.83   | 0.85        | 0.60     | 1.18      | 0.91       | 3.816   |

We evaluate those generative models using tensorflow-inceptionv3 [30] and the architecture-friendly resizer: PIL.BILINEAR for the postprocessing step. In the case of imagenet-128 and imagenet-256 experiments, we preprocess images using the best performing resizer among PIL.BILINEAR, PIL.BICUBIC, and PIL.LANCZOS based on fréchet distance. We mark † if images are preprocessed using PIL.BILINEAR for evaluation, † † for the case of PIL.BICUBIC, and † † † for the case of PIL.LANCZOS. Inference time is computed using a single A100 GPU.

### IX. Future Research Direction

**Generative Model Evaluation:** Our paper primarily focuses on the training and evaluation protocols, as well as providing a comprehensive benchmark. However, it is still very difficult to explicitly specify which metrics researchers should use to develop and evaluate generative models. While widely adopted metrics like IS and FID have served as guiding principles in...
visual generative AI, several cases were reported where they do not perfectly align with the human evaluation of realism [150]. As a result, human evaluation is a crucial component for the accurate evaluation of generative models. Recently, several attempts, including Zhou et al. [151]’s Human eYe Perceptual Evaluation (HYPE), try to introduce human-in-the-loop evaluation based on psychophysics. However, the time and cost involved in human evaluations limit the active use of this method in the community, and as a consequence, only a few selected models like SNGAN [105], BigGAN [7], and StyleGAN [54] offer human evaluation results. Given these circumstances, we believe developing more reliable human/automatic procedures will be critical in the future.

*Image Synthesis on Open-World Visual Images:* Recently, due to the amazing results from text-to-image synthesis models through a diffusion process, the generative modeling community is experiencing a rapid shift in technological trends from GANs to diffusion or auto-regressive (AR) models. With the development of text-to-image diffusion models, such as GLIDE [152], DALL-E 2 [90], and Imagen [91], an enormous amount of GPUs are being devoted to training diffusion or AR models. GANs have yet to be actively utilized for text-to-image synthesis for their brittle nature of adversarial dynamics and their inability to synthesize structural details of objects within images faithfully. However, recent models, such as GiganGAN [64] and StyleGAN-T [65] have demonstrated that GANs can be also scaled up and used for text-to-image synthesis on open-world datasets, such as LAION2b-en [153] and COYO-700M [154]. This reveals that GANs can potentially serve as a breakthrough alternative to existing text-to-image models, which have suffered from slow inference time.

On the other hand, there exist efforts to reduce the inference time of diffusion models through techniques like consistency distillation [149] and progressive distillation [148], [155]. These approaches aim to make the synthesis process more efficient. We anticipate that these single-step generative models, including GANs, will be applied to various real-world applications. For example, one-step diffusion models can be effectively employed for customization tasks [156] that involve fine-tuning the model, while GANs can be utilized for image manipulation tasks [157] by leveraging the linear latent space property.

**X. CONCLUSION**

In this article, we have introduced improper practices that arise when training and evaluating GAN. By correcting the above practices, we have presented training and evaluation protocols focusing on high-quality image generation and fair evaluation. Since current image synthesis benchmarks are not constructed using the proposed protocols, we have implemented extensive types of GANs based on our proposed GAN taxonomy to train and evaluate GANs from scratch. We provide StudioGAN, a software library of various GAN and relevant modules. Using the training and evaluation protocols and StudioGAN, we have offered a large-scale GAN evaluation benchmark and explained intriguing findings and potential hazards, which cast important questions for generative model development. In addition, we have evaluated recent image generation approaches such as ADM and RQ-Transformer. We hope StudioGAN serves as a cornerstone to explore GAN approaches and boosts the development of a next-generation generative AI.

**REFERENCES**

[1] I. Goodfellow et al., “Generative adversarial nets,” in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014.

[2] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” 2016, arXiv 1511.06434.

[3] T. Karras, L. A. Nyberg, and R. Tomioka, “f-GAN: Training generative neural samplers using variational divergence minimization,” in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016.

[4] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, “Unrolled generative adversarial networks,” in *Proc. Int. Conf. Learn. Representations*, 2017.

[5] J. J. Zhao, M. Mathieu, and Y. LeCun, “Energy-based generative adversarial network,” in *Proc. Int. Conf. Learn. Representations*, 2017.

[6] M. Lin, “Softmax GAN,” 2017, arXiv:1704.06191.

[7] A. Brock, J. Donahue, and K. Simonyan, “Large scale GAN training for high fidelity natural image synthesis,” in *Proc. Int. Conf. Learn. Representations*, 2019.

[8] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of styleGAN,” in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8107–8116.

[9] T. Karras et al., “Alias-free generative adversarial networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 852–863.

[10] M. Kang, W. Shim, M. Cho, and J. Park, “Rebooting ACGAN: Auxiliary classifier GANs with stable training,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 23505–23518.

[11] N. Kumari, R. Zhang, E. Shechtman, and J.-Y. Zhu, “Ensembling off-the-shelf models for GAN training,” in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10641–10652.

[12] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proc. Proc. Int. Conf. Comput. Vis.*, 2017, pp. 2223–2232.

[13] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, “High-resolution image synthesis and semantic manipulation with conditional GANs,” in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8798–8807.

[14] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, “StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation,” in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8789–8797.

[15] M.-Y. Liu et al., “Few-shot unsupervised image-to-image translation,” in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 10551–10560.

[16] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu, “Contrastive learning for unpaired image-to-image translation,” in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 746–755.

[17] E. Richardson et al., “Encoding in style: A StyleGAN encoder for image-to-image translation,” in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 2287–2296.

[18] C. Ledig et al., “Photo-realistic single image super-resolution using a generative adversarial network,” in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 105–114.

[19] X. Wang et al., “ESRGAN: Enhanced super-resolution generative adversarial networks,” in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2018, pp. 63–79.

[20] Y. Yuan, S. Liu, J. Zhang, Y. Zhang, C. Dong, and L. Lin, “Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2018, pp. 814–81409.

[21] J. Guan, C. Pan, S. Li, and D. Yu, “SRDGAN: Learning the noise prior for off-the-shelf models for GAN training,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 10551–10560.

[22] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu, “Contrastive learning for unpaired image-to-image translation,” in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 746–755.

[23] E. Richardson et al., “Encoding in style: A StyleGAN encoder for image-to-image translation,” in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 2287–2296.
[24] M. Niemeyer and A. Geiger, “Giraffe: Representing scenes as compositional generative neural feature fields,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., 2021, pp. 11448–11459.

[25] J. Gu, L. Liu, P. Wang, and C. Theobalt, “StyleNeRF: A style-based 3D-aware generator for high-resolution image synthesis,” 2021, arXiv:2110.09895.

[26] P. Zhou, L. Xie, B. Ni, and Q. Tian, “CIPS-3D: A 3D-Aware generator of GANs based on conditionally-independent pixel synthesis,” 2021, arXiv:2110.09788.

[27] T. Salimans et al., “Improved techniques for training GANs,” in Proc. Adv. Neural Inf. Process. Syst., 2016, pp. 2234–2243.

[28] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “GANs trained by a two time-scale update rule converge to a local nash equilibrium,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 6629–6640.

[29] G. Parmar, R. Zhang, and J.-Y. Zhu, “On buggy resizing libraries and surprising subtleties in FID calculation,” 2021, arXiv:2104.11222.

[30] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., 2016, pp. 2818–2826.

[31] T. Kynkäneni, T. Karras, M. Aittala, T. Aila, and J. Lehtinen, “The role of ImageNet classes in frechet inception distance,” 2021, arXiv:2203.02026.

[32] T. Kynkäneni, T. Karras, S. Laine, J. Lehtinen, and T. Aila, “Improved precision and recall metric for assessing generative models,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, 2019.

[33] M. F. Naeem, S. J. Oh, Y. Uh, Y. Choi, and J. Yoo, “Reliable fidelity and diversity metrics for generative models,” in Proc. Int. Conf. Mach. Learn., 2020, pp. 7176–7185.

[34] M. Caron, J. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, “Unsupervised learning of visual features by contrasting cluster assignments,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 9912–9924.

[35] S. Morozov, A. Voynov, and A. Babenko, “On self-supervised image representations for GAN evaluation,” in Proc. Int. Conf. Learn. Representations, 2021.

[36] M. Caron et al., “Emerging properties in self-supervised vision transformers,” in Proc. Int. Conf. Comput. Vis., 2021, pp. 9650–9660.

[37] Z. Liu et al., “Swin transformer: Hierarchical vision transformer using shifted windows,” in Proc. Int. Conf. Comput. Vis., 2021, pp. 10012–10022.

[38] H. Zhang, Z. Zhang, A. Odena, and H. Lee, “Consistency regularization for generative adversarial networks,” in Proc. Int. Conf. Learn. Representations, 2020.

[39] S. Gu et al., “Vector quantized diffusion model for text-to-image synthesis,” 2021, arXiv:2111.14822.

[40] H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman, “MaskGIT: Masked generative image transformer,” 2022, arXiv:2202.04200.

[41] D. Lee, C. Kim, S. Kim, M. Cho, and W. Han, “Autoregressive image generation using residual quantization,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., 2022, pp. 11513–11522.

[42] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., 2022, pp. 10674–10685.

[43] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 6840–6851.

[44] Y. Song, J. Sohild-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, “Score-based generative modeling through stochastic differential equations,” in Proc. Int. Conf. Learn. Representations, 2021.

[45] A. Vahdat, K. Kreis, and J. Kautz, “Score-based generative modeling in latent space,” in Proc. Adv. Neurol Inf. Process. Syst., 2021, pp. 11287–11302.

[46] P. Dhariwal and A. Nichol, “Diffusion models beat GANs on image synthesis,” 2021, arXiv:2202.05997.

[47] T. Dockhorn, A. Vahdat, and K. Kreis, “Score-based generative modeling with critically-damped langevin diffusion,” in Proc. Int. Conf. Learn. Representations, 2022.

[48] P. Esser, R. Rombach, and B. Ommer, “Taming transformers for high-resolution image synthesis,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., 2021, pp. 12868–12878.

[49] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, “Self-attention generative adversarial networks,” in Proc. Int. Conf. Mach. Learn., 2019, pp. 7354–7363.

[50] E. L. Denton et al., “Deep generative image models using a laplacian pyramid of adversarial networks,” in Proc. Adv. Neurol Inf. Process. Syst., vol. 28, 2015.
[137] mseitzer, “FID score for PyTorch.” 2018. [Online]. Available: https://github.com/mseitzer/pytorch-fid
[138] Clovai, “Reliable fidelity and diversity metrics for generative models,” 2020. [Online]. Available: https://github.com/clovai/generative-evaluation-prdc
[139] FacebookResearch, “Unsupervised learning of visual features by contrasting cluster assignments,” 2020. [Online]. Available: https://github.com/facebookresearch/swav
[140] microsoft, “Swin transformer,” 2021. [Online]. Available: https://github.com/microsoft/Swin-Transformer

Minguk Kang (Member, IEEE) received the BS degree from Pusan National University, South Korea, in 2019. He is currently working toward the PhD degree from the Department of Computer Science and Engineering, POSTECH, in 2023. He is currently working toward the graduation degree with the Department of Computer Science and Engineering, Seoul National University, South Korea. Along with Minguk, he is one of the core contributors of the StudioGAN library. His research interests include generative modeling, with applications in computer vision. He has authored three academic papers in NeurIPS and CVPR.

Joonghyuk Shin (Member, IEEE) received the BS degree from the Department of Computer Science and Engineering, POSTECH, in 2023. He is currently working toward the graduation degree with the Department of Computer Science and Engineering, Seoul National University, South Korea. Along with Minguk, he is one of the core contributors of the StudioGAN library. His research interests include generative modeling and its diverse applications in computer vision.

Jaesik Park (Member, IEEE) received the bachelor’s degree from Hanyang University, in 2009, and the master’s and Ph.D. degrees from KAIST, in 2011 and 2015, respectively. He is an assistant professor with the Department of Computer Science and Engineering, Seoul National University. He worked at Intel as a research scientist (2015-2019), and he was a faculty member at POSTECH (2019-2023). His research interests include image synthesis and 3D scene understanding.

[160] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, “Neural photo editing with introspective adversarial networks,” in Proc. Int. Conf. Learn. Representation, 2017.
[161] S. Sinha, Z. Zhao, A. Goyal, C. Raffel, and A. Odena, “Top-k training of GANs: Improving GAN performance by throwing away bad samples,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 14638–14649.