TinyGAN: Distilling BigGAN for Conditional Image Generation

Ting-Yun Chang[1][0000−0003−2198−1170] and Chi-Jen Lu[1][0000−0003−0835−1190]

Institute of Information Science, Academia Sinica, Taiwan
r06922168@ntu.edu.tw, cjlu@iis.sinica.edu.tw.
https://www.iis.sinica.edu.tw/en/index.html

Abstract. Generative Adversarial Networks (GANs) have become a powerful approach for generative image modeling. However, GANs are notorious for their training instability, especially on large-scale, complex datasets. While the recent work of BigGAN has significantly improved the quality of image generation on ImageNet, it requires a huge model, making it hard to deploy on resource-constrained devices. To reduce the model size, we propose a black-box knowledge distillation framework for compressing GANs, which highlights a stable and efficient training process. Given BigGAN as the teacher network, we manage to train a much smaller student network to mimic its functionality, achieving competitive performance on Inception and FID scores with the generator having 16× fewer parameters.[1]

1 Introduction

Generative Adversarial Networks (GANs) [1] have achieved considerable success in recent years. The framework consists of a generator, which aims to produce a distribution similar to a target one, as well as a discriminator, which aims to distinguish these two distributions. The generator and the discriminator are trained in an alternative way, with the discriminator acting as an increasingly scrupulous critic of the current generator. Conditional GANs (cGANs) [2] are a type of GANs for generating samples based on some given conditional information. Different from unconditional GANs, the discriminator of cGANs is now asked to distinguish the two distributions given the conditional information.

Despite their success, GANs are also known to be hard to train, especially on large-scale, complex datasets such as ImageNet. The recent work of BigGAN [3], a kind of cGANs, demonstrates the benefit of scaling. More precisely, by scaling up both the model size and batch size, some of the training problems can be mitigated, and high-quality images can be generated. However, this also leads to high computational cost and memory footprint, even for inference in test time.

One may wonder if it is possible to compress such a large model into a much smaller one. For classification tasks, several techniques have been developed for

[1] The source code and the trained model are publicly available at https://github.com/terarachang/ACCV_TinyGAN.

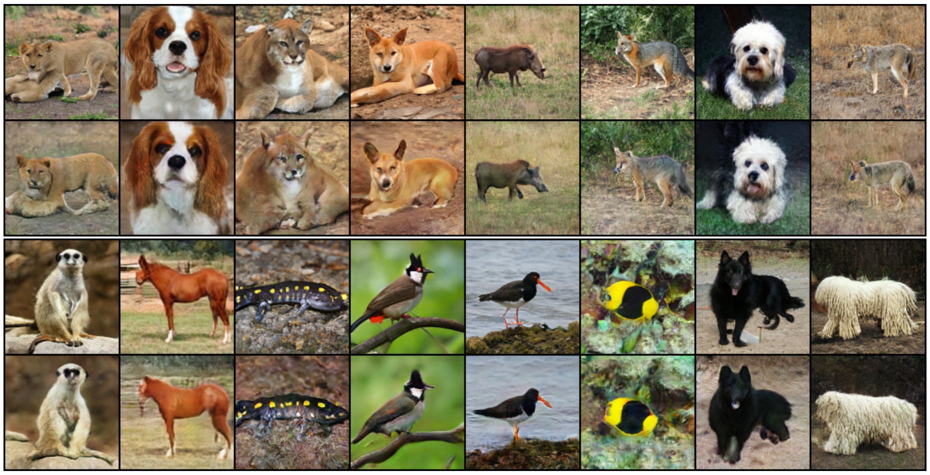

Fig. 1: A comparison between images generated by BigGAN and the proposed TinyGAN. Pictures in odd rows are produced by BigGAN, while those in even rows are by TinyGAN given the same input.

compressing classifiers, including *knowledge distillation* [4], *network pruning* [5], and *quantization* [6]. For compressing GANs, we find that the concept of knowledge distillation (KD) becomes especially appealing. Based on a *teacher-student framework*, it aims to impart knowledge encoded in a large, well-trained teacher network to a small student network. For GANs, we find it appropriate to consider the input-output relationship of the teacher generator as the knowledge to be distilled. Note that the difficulties of training GANs from scratch may be attributed mostly to the lack of supervision from paired training data. Not sure about what the ideal functionality it should have, the generator turns to chase a moving target provided by an evolving discriminator. On the other hand, having a well-trained generator such as BigGAN as a teacher, we can use it simply as a black box to generate its input-output pairs as training data, and train a student network in a supervised way. Such a supervised learning is typically much easier, with a much more stable and efficient training process. In contrast, training classifiers are usually done in a supervised way already, and hence KD on classifiers usually takes a white-box approach, requiring access to the internal of the teacher networks.

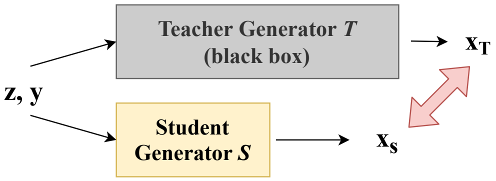

Fig. 2: Illustration of the problem formulation. z is the noise vector, and y is the class label. Our goal is to mimic the functionality of the teacher generator via black-box knowledge distillation.

Although KD has been successfully applied to classification tasks [7, 4], it is less studied for image generation. In our work, we leverage BigGAN trained on ImageNet as our teacher network and design a compact, lightweight student network to mimic the functionality of BigGAN. Given a noise vector and a class label as input, we would like the student network to generate a similar, high-quality image like that produced by BigGAN. In this paper, we focus on *black-box KD*, defined as having access to only the input-output functionality of the teacher network, instead of any internal knowledge such as its intermediate features as needed in works such as [4, 8]. We claim that this is a meaningful setting for several aspects. First, it allows one to utilize a model without needing the authority to access its model parameters, by simply collecting its input/output pairs. Next, it allows us to discard the teacher network (both generator and discriminator) in the training phase to save memory after collecting such pairs in the preprocessing step. Furthermore, it allows us to adopt a different architecture for the student network, which enables us to substantially reduce the model size from that of BigGAN. Figure 2 is an illustration of our problem formulation, and Figure 1 shows some sampled results.

We propose several training objectives for distilling BigGAN, including *pixel-level distillation*, *adversarial distillation*, and *feature-level distillation*. Given the same input, let x_T and x_S be the images generated by the teacher and the student networks respectively. The objective of *pixel-level distillation* is to minimize the distance between x_T and x_S, and here we use the pixel-wise L1 distance. We further utilize a small discriminator to help align our generated distribution to BigGAN's, with the *adversarial distillation* having a similar objective as in standard GAN training, but now taking BigGAN's output distribution as the target one. Finally, as pixel-level distance often leads to blurry images, we apply *feature-level distillation* to mitigate this problem. We achieve this without needing additional parameters, by taking the intermediate features in the discriminator and encouraging those derived from x_S to match those from x_T. In addition to the distillation objectives, we also include the standard cGANs loss, to push our generated distribution towards that of ImageNet as well. Our main contributions are summarized as follows.

- We identify a unique and advantageous property of compressing GANs via knowledge distillation, and initiate the study on the diverse ImageNet.
- We propose a black-box KD framework tailored for GANs, which requires little permission for the teacher networks and highlights an efficient training process.
- Our strategy greatly compresses BigGAN, while our model maintains competitive performance.

We see our contributions as more conceptual than technical. While the task of compressing classifiers has received much attention, to our knowledge, we are the first to explore black-box KD for compressing GANs. Moreover, we identify a unique property of KD on GANs, which enables us to apply rather simple techniques to achieve a substantial compression ratio, and we believe that it is possible to combine our approach with other compression techniques to further reduce the model size. Let us remark that the emphasis of our work is the realization of a simple and efficient strategy to obtain a generator with both good quality and a compact size. Whereas we do not rule out the possibility of training a small-sized, well-performed, and stable GANs from scratch, it is likely to be challenging except for very skilled and experienced experts. In fact, in our attempt to directly train a smaller GAN from scratch, we have encountered those notorious training problems as expected, while we have never experienced any issues of instability when taking our KD approach. Therefore, our work suggests a possibly more reliable way to obtain a lightweight, high-quality generator: instead of directly training one from scratch, one could first train a large generator and then distill from it a small one.

2 Related Work

Generative Adversarial Networks. GANs have excelled in a variety of image generation tasks [9–11]. Still, they are well known for problems such as training instability and sensitivity to hyperparameter choices, requiring great efforts in model tuning. Several works [12–16] have aimed to tackle such problems. Notably, the recent work of [17, 18] proposed to constrain the Lipschitz constant of the discriminator function by limiting the spectral norm of its weights, which makes possible high quality class-conditional image generation over large-scale, complex distributions.

BigGAN. BigGAN [3] further scales up GANs by training with considerable model size and batch size on complex datasets. It basically follows previous SOTA architectures [17–19], and proposes two variants, BigGAN and BigGAN-deep, to incorporate the input noises and class labels. Utilizing the truncation trick, i.e., training a model with $z \sim N(0, I)$ but sampling z from a truncated normal (with values falling outside two standard deviations being re-sampled) in test time, BigGAN is able to trade off variety and fidelity. BigGAN demonstrates that GANs benefit dramatically from scaling.

Knowledge Distillation on GANs. Perhaps the work most related to ours is [20], which to our knowledge is the first to apply knowledge distillation on GANs. However, their experiments are conducted on MNIST, CIFAR-10, and CelebA, which are relatively simple with much less image diversity compared to ImageNet. Besides, there are several differences in the settings. First, they do not experiment on conditional generation. Second, they explore teacher-student generators only on the DCGAN architectures, which might be less general and seems easier for the student to mimic a teacher with a similar architecture. Finally, accessing to and updating the teacher discriminator are allowed in their work, while we focus on black-box knowledge distillation, which is more memory efficient during training as we do not need to keep the large teacher network. (Once we synthesized the dataset from BigGAN's generator during the preprocessing phase, we do not need it anymore). To sum up, in this work, we study knowledge distillation on GANs in a more general framework and a harder setting.

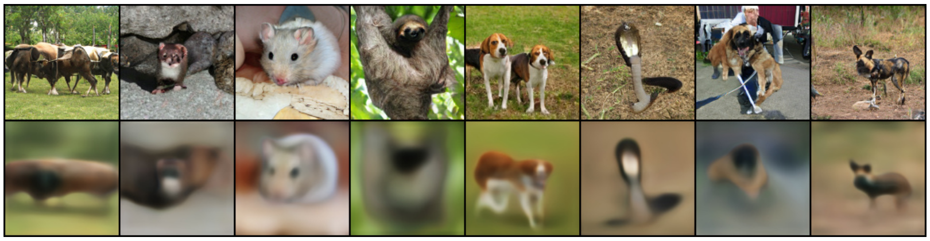

Fig. 3: Examples generated by TinyGAN trained with pixel-level distillation loss (Eq. 1) alone, shown in the second row. The first row shows corresponding images produced by BigGAN given the same input.

3 Tiny Generative Adversarial Networks

We first describe how our proposed framework, TinyGAN, distills knowledge from BigGAN. Then, we discuss how TinyGAN incorporates real images from the ImageNet dataset, which further improves the performance.

3.1 BigGAN Distillation

We propose a black-box KD method specifically designed for GANs, which does not need to access the parameters of the teacher network or share a similar network structure. We use BigGAN as the teacher network and train our student network, TinyGAN, with much fewer parameters to mimic its input-output behavior. We will elaborate on several proposed objectives for knowledge distillation in this subsection.

Pixel-Level Distillation Loss. To mimic the functionality of BigGAN, a naive method is to minimize the pixel-level distance between the images generated by BigGAN and TinyGAN given the same input. Formally, let

$$L_{\text{KD\_pix}} = \mathbb{E}_{z\sim p(z), y\sim q(y)}[\|T(z,y) - S(z,y)\|_1], \tag{1}$$

where T is the frozen teacher network (BigGAN's generator), S is our student network, $z \in \mathbb{R}^{128}$ is a latent variable drawn from the truncated normal distribution $p(z)$, and y is the class label sampled from some categorical distribution $q(y)$. However, we found that using such a pixel-level distance alone is not sufficient for modeling complex datasets such as ImageNet, resulting in blurry images as shown in Figure 3. Thus, we propose the following additional objectives to mitigate this problem.

Adversarial Distillation Loss. To sharpen the generated images, we incorporate a discriminator to help make the images generated by TinyGAN indistinguishable from those by BigGAN. We adopt an adversarial loss

$$L_{\text{KD\_S}} = -\mathbb{E}_{z,y}[D(S(z,y),y)] \tag{2}$$

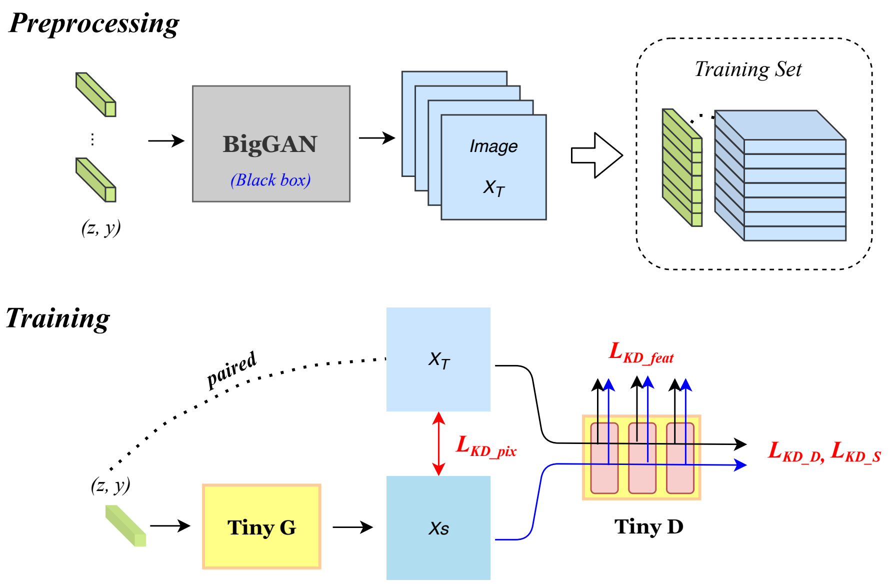

Fig. 4: Illustration of the proposed pipeline and the distillation objectives.

for the generator, and the loss

$$L_{\mathrm{KD\_D}} = \mathbb{E}_{z,y}[\max(0, 1 - D(T(z, y), y)) + \max(0, 1 + D(S(z, y), y))] \quad (3)$$

for the discriminator, where z is the noise vector, y is the class label, $T(z, y)$ is the image generated by BigGAN, while S and D are respectively the generator and discriminator of our TinyGAN, which are alternatively trained as in usual GAN training. We trained our small-sized discriminator D from scratch, and have experimentally found that projection discriminator with *hinge* adversarial loss proposed by [17] works the best.

Feature-Level Distillation Loss. To further mitigate the problem of generating blurry images using pixel-level distance, we propose a feature-level distillation loss, which does not require any additional parameter. We believe that as the discriminator needs to distinguish the source of images, it must learn some useful features. Hence, we take the features computed at each convolutional layer in the discriminator, and ask TinyGAN to generate images with similar features as those from BigGAN. Formally, let

$$L_{\mathrm{KD\_feat}} = \mathbb{E}_{z,y}[\Sigma_i \alpha_i \| D_i(T(z, y), y) - D_i(S(z, y), y) \|_1], \quad (4)$$

where D_i is the feature vector extracted from the ith-layer of our discriminator, and α_i is the corresponding weight. We put more emphasis on higher-level features and assign larger weights to them.

This objective is similar to the feature matching loss proposed by [21], which encourages the generator to generate images containing intermediate representations similar to those from the real images in order to fool the discriminator. We have also tried to incorporate different kinds of feature-level loss, such as perceptual loss from VGG network [22], but got worse results. Figure 4 illustrates all the proposed distillation objectives.

3.2 Learning from Real Distribution

We also allow our model to learn from real images in ImageNet dataset, attempting to ameliorate the mode dropping problem of BigGAN we observed in some classes. Specifically, we use the *hinge* version of the adversarial loss [23]

$$L_{\mathrm{GAN\_D}} = \mathbb{E}_{x,y}[\max(0, 1 - D(x, y))] + \mathbb{E}_{z,y}[\max(0, 1 + D(S(z, y), y))], \tag{5}$$

where x is now the real image sampled from ImageNet. The generator loss $L_{\mathrm{GAN\_S}}$ is the same as $L_{\mathrm{KD\_S}}$ in Equation (2).

3.3 Full Objective

Finally, the objective to optimize our student generator and discriminator, S and D, are written respectively as

$$L_{\mathrm{S}} = L_{\mathrm{KD\_feat}} + \lambda_1 L_{\mathrm{KD\_pix}} + \lambda_2 L_{\mathrm{KD\_S}} + \lambda_3 L_{\mathrm{GAN\_S}}, \text{ and} \tag{6}$$

$$L_{\mathrm{D}} = L_{\mathrm{KD\_D}} + \lambda_4 L_{\mathrm{GAN\_D}}. \tag{7}$$

Empirically, we gradually decay the weight of the pixel-level distillation loss λ_1 to zero, relying on the discriminator to provide useful guidance. Note that pixel-level distillation loss is still an important term, since it provides stable supervision in the early training phase while discriminator might still be quite naive at that time.

4 Network Architecture

Now we describe the architectures of our generator and discriminator in detail.

4.1 Generator

We have tried different generator architectures and experimentally found that ResNet [24] based generator with class-conditional BatchNorm [25, 26] works better. To keep a tight computation budget, our student generator does not adopt attention-based [19] or progressive-growing mechanisms [27]. To substantially reduce the model size, we mainly rely on using fewer channels and replacing standard convolution by depthwise separable convolution. In addition, we adopt a simpler way to introduce class conditions which also helps the reduction. Overall, our generator has 16× fewer parameters than BigGAN's, while still capable of generating satisfying images of 128×128 resolution.

Shared Class Embedding. We provide class information to the generator with class-conditional BatchNorm [25, 26]. To reduce computation and memory costs, similar to BigGAN, we use shared class embedding for different layers, which is linearly transformed to produce the BatchNorm affine parameters [28]. Different from BigGAN, we design a simpler architecture to incorporate the class label. Specifically, we only input the noise vector z to the first layer, and then for each conditional BatchNorm layer, we linearly transform the class embedding $E(y)$ to the gains and biases. Figure 5 is the illustration of our generator architecture.

Depthwise Separable Convolution. To further reduce the model size, we replace all the 3×3 standard convolutional layers in our generator with depthwise separable convolution [29], which factorizes a standard convolution into a depthwise convolution and a pointwise convolution, by first applying a single filter to each input channel (depthwise), and then utilizing a 1×1 convolution to combine the outputs (pointwise). Depthwise separable convolution uses $\frac{1}{O} + \frac{1}{k \times k}$ fewer parameters than the standard one, where O is the number of output channels and k is the kernel size. We denote TinyGAN using standard conv. layers as TinyGAN-std, and the variant with depth-wise conv. layer as TinyGAN-dw.[2]

4.2 Discriminator

With the supervision from BigGAN, the difficulties of training is greatly reduced and we found that a simple discriminator architecture already works well. Following [18, 17], we use spectral normalized discriminator and introduce the class condition via projection. But instead of utilizing complicated residual blocks, we found that simply stacking multiple convolutional layers with stride as DC-GAN [30] works well enough, which greatly reduces the number of parameters. In fact, our discriminator is 10× smaller than that of BigGAN's.

5 Experiments

5.1 Datasets

ImageNet. The ImageNet ILSVRC 2012 dataset [31] consists of 1,000 image classes, each having approximately 1,300 images. We compressed each image to 128x128 pixels, using the source code released by [17].

Images Generated by BigGAN. We view BigGAN, our teacher network, as a black-box model and collect its input-output pairs to train our student network. For each class, we randomly sample 3,000[3] noise vectors from the truncated normal distribution and collect the corresponding output generated by BigGAN using the official API.[4]

[2] Note that all the figures in this paper are generated by the TinyGAN-dw variant.

[3] We also tried 1000 instances per class, which already achieves good results; however, no significant improvement was observed when we increased to 4000.

[4] `https://tfhub.dev/deepmind/biggan-deep-128/1`

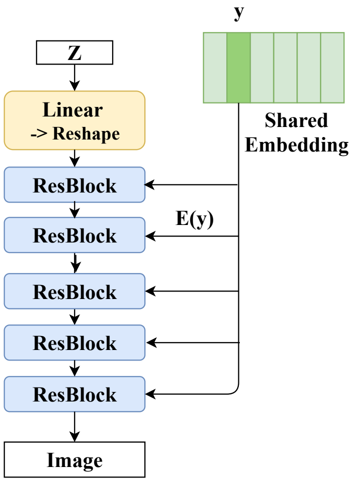

Fig. 5: Student Generator S

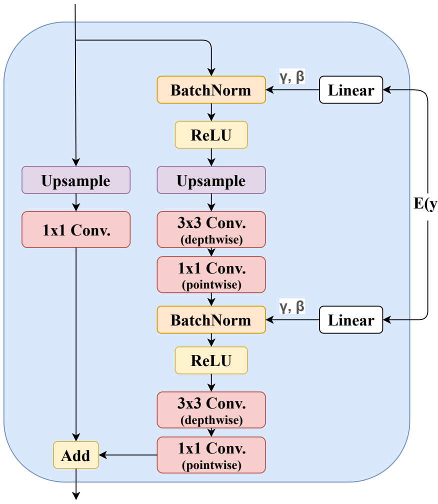

Fig. 6: A Residual Block in S

As we found TinyGAN unable to model some complicated objects well enough, we only report in Table 1 the IS/FID/intra-FID scores measured on *all* animal classes (398 classes in total). It shows that our approach can work well for a large set of homogeneous classes, and we focus on animals as they may have more downstream applications than other classes. Further discussion about experiments on all 1000 classes can be found in the supplementary material.

5.2 Evaluation Metrics

Inception score (IS). IS [32] measures the KL-divergence between the conditional class distribution $p(y|x)$ and the marginal class distribution $p(y)$. Formally,

$$\mathrm{IS} = \exp\left(\mathbb{E}_x[\mathrm{KL}(p(y|x)\|p(y))]\right). \tag{8}$$

A higher Inception score suggests a better performance. Despite the limitations of IS [33], we still adopt it as it is widely used in prior works.

Fréchet Inception Distance (FID). FID score [34] computes the 2-Wasserstein distance between the two distributions r and g, and is given by

$$\mathrm{FID}(r, g) = \|\mu_r - \mu_g\|_2 + \mathrm{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}). \tag{9}$$

Here, μ_r and μ_g are the means of the final feature vectors extracted from the inception model [35] with input from the real and generated samples respectively, while Σ_r and Σ_g are the corresponding covariance matrices, and Tr is the trace. We also compute the intra-FID score [17], which measures the average FID score within each class.

Unlike Inception score, FID is able to detect intra-class mode dropping. It is considered a more consistent estimator [33], as a model that generates only a single image per class can score a perfect IS but not a good FID. Here we follow prior works and use `TensorFlow toolkit` to calculate IS and FID scores.

5.3 Baseline Models

SNGAN-Projection. Spectral Normalization GAN (SNGAN) [18] proposes spectral normalization to stabilize the training of the discriminator. [17] further proposes a projection-based discriminator, which incorporates the class labels via inner product instead of concatenation. The combined model, denoted as SNGAN-Projection, has shown significant improvements on ImageNet Dataset, and we consider it as a strong baseline model. Statistics in Table 1 are reported using the source code and the pretrained generator released by the authors.[5]

SAGAN. Self-Attention GAN (SAGAN) [19] is built atop SNGAN-projection, and introduces a self-attention mechanism [36, 37] into convolutional GANs, in order to model long-range dependencies across image regions. As the authors do not provide a pretrained model and we are unable to train it from scratch due to limits of computation, its scores are left blank in Table 1, and we only compare its model size and computation cost to our TinyGAN. For reference, the IS/FID/intra-FID scores reported in the original paper, evaluated on all 1000 classes, are **52.52/18.7/83.7** respectively.

TinyGAN trained from scratch. To justify the effectiveness of knowledge distillation on GANs, we also experimented on training TinyGAN from scratch, without the guidance of a teacher network. That is, we use an identical architecture of TinyGAN, but trained it using only the adversarial loss L_{GAN} (Eq. 5).

5.4 Training

The proposed TinyGAN are trained using Adam [38] with $\beta_1 = 0.0$ and $\beta_2 = 0.9$. The learning rates of generator and discriminator are both set to 0.0002 with linear decay. We perform one generator update after 10 discriminator updates. With the stable guidance from the teacher network, no special tricks for training GANs are needed. While BigGAN notes that using a large batch size boosts the performance, in our TinyGAN, we found that a smaller batch size (32 or 16) works as well. Training takes about 3 days on a single NVIDIA 2080Ti GPU.

5.5 Results

We evaluate TinyGAN on all the 398 animal classes in the ImageNet dataset, and the results are shown in Table 1. We compare the computation cost of

[5] `https://github.com/pfnet-research/sngan_projection`

Model	Ch.	#Par.	G Par.	FLOPs	IS ↑	FID ↓	intra-FID ↓
SNGAN-proj	64	72.0 M	42.0 M	9.10 B	31.4 ± 0.7	29.0	84.1
SAGAN	64	81.5 M	42.0 M	9.18 B	-	-	-
BigGAN-deep	128	85.0 M	50.4 M	8.32 B	**146.1 ± 1.7**	**19.8**	**55.6**
TinyGAN-std	32	12.6 M	9.3 M	2.29 B	94.0 ± 1.2	21.6	70.6
TinyGAN-std	16	6.2 M	2.9 M	0.58 B	68.25 ± 1.0	27.4	88.1
TinyGAN-dw	32	6.4 M	3.1 M	0.44 B	79.19 ± 1.6	24.2	79.1

Table 1: Inception Score (IS, higher is better) and Fréchet Inception Distance (FID, lower is better). Ch. is the channel multiplier representing the number of units in each layer. #Par. is total number of parameters. We highlight the generator's parameters G Par. since the discriminator is not required for inference. M denotes million and B is billion.

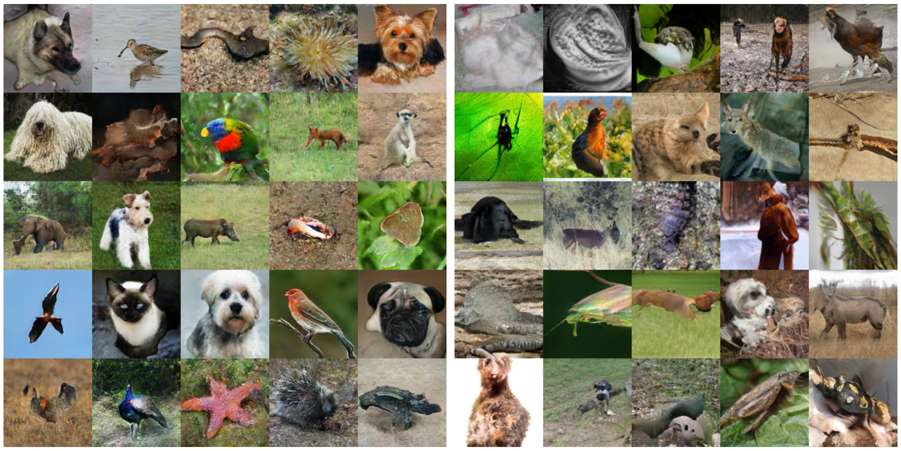

Fig. 7: A comparison between randomly sampled images generated by TinyGAN-dw (left) and SNGAN-projection (right).

TinyGAN with the teacher network (BigGAN-deep) and two strong baseline models discussed before.

Note that our proposed model uses much fewer parameters and floating-point operations than all the other frameworks. We also study the trade-off between model size and performance of different variants of TinyGAN in the last three rows in Table 1. Experiments show that TinyGAN with standard conv. layers (TinyGAN-std) achieves the best performance but uses more parameters. To reduce the model size, we can either reduce the channel multiplier or adopting depth-wise separable conv. layers (TinyGAN-dw). The result shows that aggressively reducing channels leads to a noticeable drop in performance. On the other hand, it is much less significant in TinyGAN-dw, making it a suitable choice under a tight computation budget. Specifically, our generator of

Model	FID ↓	intra-FID ↓
TinyGAN-dw	24.2	79.1
$-L_{\text{KD\_feat}}$	54.4	149.2
$-L_{\text{GAN}}$	28.8	89.9
$-L_{\text{KD\_S}}, L_{\text{KD\_D}}$	60.5	157.0
$L_{\text{KD\_pix}}$	107.9	216.0

Table 2: Ablation Study

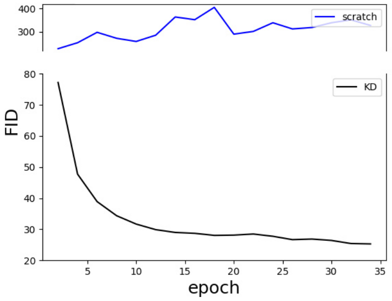

Fig. 8: FID scores during the training phase of TinyGAN-dw (trained from scratch v.s. trained with KD losses).

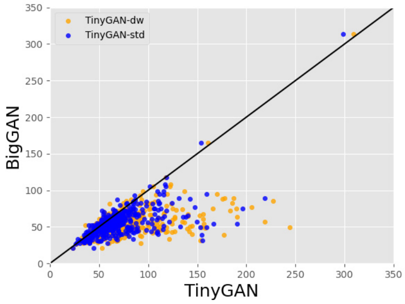

Fig. 9: Comparing intra-FID scores of TinyGAN and BigGAN. Each dot corresponds to a class.

TinyGAN-std/TinyGAN-dw has ∼ 18%/6% parameters and ∼ 28%/5% FLOPs when compared with the teacher network, and we also have similar reductions from the other two baseline models.

Although there is a performance gap between our TinyGAN-dw and the teacher network, we claim that it is tolerable considering its compact model size, and its better performance over SNGAN-projection in all the metrics. We further compare the image quality of TinyGAN-dw and SNGAN-projection in Figure 7, where all images are randomly sampled within animal classes. We found that while SNGAN-projection is able to produce sharper images with clear details, perhaps due to its larger model complexity, our TinyGAN focuses on the intended class itself and generates more realistic images with less distortion.[6]

Finally, let us stress that the main point of our work is not to claim how small our network is, but to propose an easy way to train such one. Figure 8 shows the learning curve of TinyGAN-dw, demonstrating a smooth, stable and efficient training process it has. In fact, with our knowledge distillation losses, we have never experienced any training collapse, and most of our effort has been spent on finding the right balance between model size and image quality. On the other hand, we have also experimented on training TinyGAN from scratch, and

[6] More randomly sampled images for comparisons between TinyGAN and SNGAN-projection can be found in the supplementary material.

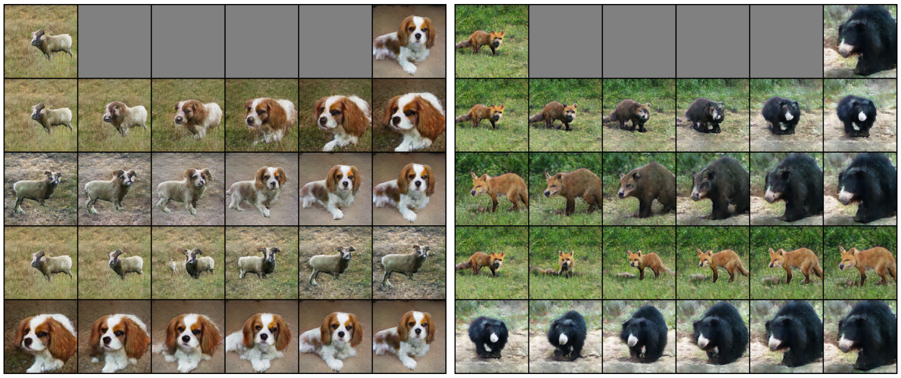

Fig. 10: Interpolations between z, y pairs. The second and third rows interpolate between y with z fixed, while the last two rows interpolate between z with y fixed. Observe that semantics are maintained between two endpoints.

the blue line in Figure 8 shows a typical training failure we often encountered. Although we do not rule out the possibility of training a small-size GAN from scratch, based on the network architecture of either TinyGAN, BigGAN, or other baselines, a successful training is likely to be hard without considerable efforts for overcoming those well-known training problems.

5.6 Analysis

Ablation Study. We conduct ablation study to validate those objectives proposed in Section 3.1. Table 2 shows the results of omitting $L_{\text{KD\_feat}}$ (Eq. 4), L_{GAN} (Eq. 5), and $L_{\text{KD\_S}}, L_{\text{KD\_D}}$ (Eq. 2,3) respectively, as well as that of using $L_{\text{KD\_pix}}$ (Eq. 1) alone without the discriminator.

The fifth row in Table 2 shows that adding a discriminator, which only costs a few parameters, is very crucial for the performance. Because the discriminator is trained to discern real from fake images, it guides the generator to produce sharper and more realistic images. Similarly, feature-level distillation loss $L_{\text{KD\_feat}}$ significantly improves the performance as the generator learns to match the informative features extracted from the discriminator. In addition, omitting adversarial distillation loss $L_{\text{KD\_S}}, L_{\text{KD\_D}}$ and keeping the others is equivalent to training a standard cGANs (with the real distribution from ImageNet) while incorporating supervision from the teacher via pixel-wise and feature-wise losses. The notable drop in performance in the fourth row indicates the importance of leveraging the discriminator to push the student's output distribution to the teacher's.

In addition to the distillation objectives which provide stable supervision from the teacher, including the standard adversarial loss L_{GAN} further improves the image quality. The ablation study demonstrates that all the proposed objectives in Section 3 are useful for training our TinyGAN.

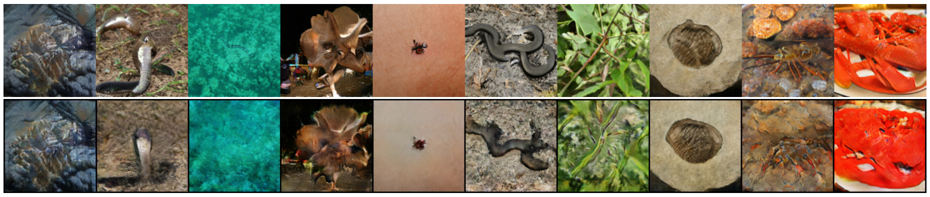

Fig. 11: Samples of the 10 worst classes. The first row is generated by BigGAN and the second row is by TinyGAN-dw.

Interpolation. To understand the generalization ability of our TinyGAN-dw, we perform linear interpolations between random noise vectors z_1, z_2 and class labels y_1, y_2.

We first interpolate between the class embedding $E(y_1)$ and $E(y_2)$ with the noise vectors fixed. In Figure 10, the second and third rows demonstrate that TinyGAN can successfully perform category morphing. We then interpolate between the noise vectors z_1 and z_2 with fixed class labels. The last two rows in Figure 10 show that TinyGAN can also smoothly manipulate some coarse features such as poses and sizes of the animals.

Quality Analysis. Finally, to better understand the weakness of our TinyGAN, we investigate on classes with high intra-FID scores. We first show the positive correlation (Pearson's correlation coefficient $= 0.54, 0.64$) between teacher and student networks (-dw, -std) in Figure 9, which reveals that TinyGAN's failure in a few classes can be attributed to the teacher network. We then focus on the 10 worst classes with the highest FID scores, which are chambered nautilus, Indian cobra, sea snake, triceratops, tick, ringneck snake, walking stick, trilobite, crayfish, and American lobster. As the samples in Figure 11 show, most of them have complicated or delicate appearances and bear little resemblance to most of the others, making them hard to model with others by a small network.

6 Conclusion

Training GANs from scratch has well-known problems, especially for complex datasets such as ImageNet, and the recent work of BigGAN shows that scaling up GANs can mitigate some of the problems and produce high-quality images. However, it requires huge computational resources not only for training but also for testing, which may prevent its use in resource-limited devices. We propose a novel black-box knowledge distillation method for GANs, which allows us to learn a much smaller generator with competitive performance in an efficient and stable way when given a well-trained large generator such as BigGAN.

Acknowledgement We would like to thank NTU MiuLab for their support, and all anonymous reviewers for insightful comments.

References

1. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. (2014) 2672–2680
2. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
3. Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. ICLR (2019)
4. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. arXiv preprint arXiv:1412.6550 (2014)
5. Han, S., Mao, H., Dally, W.J.: Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. ICLR (2016)
6. Gong, Y., Liu, L., Yang, M., Bourdev, L.: Compressing deep convolutional networks using vector quantization. ICLR (2015)
7. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
8. Zagoruyko, S., Komodakis, N.: Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. ICLR (2017)
9. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. CVPR (2017)
10. Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H.: Generative adversarial text to image synthesis. ICML (2016)
11. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2017) 4681–4690
12. Che, T., Li, Y., Jacob, A.P., Bengio, Y., Li, W.: Mode regularized generative adversarial networks. ICLR (2017)
13. Zhao, J., Mathieu, M., LeCun, Y.: Energy-based generative adversarial network. ICLR (2017)
14. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein gan. ICML (2017)
15. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: Advances in neural information processing systems. (2017) 5767–5777
16. Salimans, T., Zhang, H., Radford, A., Metaxas, D.: Improving gans using optimal transport. ICLR (2018)
17. Miyato, T., Koyama, M.: cgans with projection discriminator. ICLR (2018)
18. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. ICLR (2018)
19. Zhang, H., Goodfellow, I., Metaxas, D., Odena, A.: Self-attention generative adversarial networks. arXiv preprint arXiv:1805.08318 (2018)
20. Aguinaldo, A., Chiang, P.Y., Gain, A., Patil, A., Pearson, K., Feizi, S.: Compressing gans using knowledge distillation. arXiv preprint arXiv:1902.00159 (2019)
21. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2018)
22. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)

23. Lim, J.H., Ye, J.C.: Geometric gan. arXiv preprint arXiv:1705.02894 (2017)
24. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: European conference on computer vision, Springer (2016) 630–645
25. Dumoulin, V., Shlens, J., Kudlur, M.: A learned representation for artistic style. ICLR (2017)
26. De Vries, H., Strub, F., Mary, J., Larochelle, H., Pietquin, O., Courville, A.C.: Modulating early visual processing by language. In: Advances in Neural Information Processing Systems. (2017) 6594–6604
27. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. ICLR (2018)
28. Perez, E., Strub, F., De Vries, H., Dumoulin, V., Courville, A.: Film: Visual reasoning with a general conditioning layer. In: Thirty-Second AAAI Conference on Artificial Intelligence. (2018)
29. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
30. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR (2016)
31. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115** (2015) 211–252
32. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. In: Advances in neural information processing systems. (2016) 2234–2242
33. Lucic, M., Kurach, K., Michalski, M., Gelly, S., Bousquet, O.: Are gans created equal? a large-scale study. In: Advances in neural information processing systems. (2018) 700–709
34. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Advances in Neural Information Processing Systems. (2017) 6626–6637
35. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2015) 1–9
36. Parikh, A.P., Täckström, O., Das, D., Uszkoreit, J.: A decomposable attention model for natural language inference. EMNLP (2016)
37. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. (2017) 5998–6008
38. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. ICLR (2015)