Boosting Generative Networks for Incremental Learning

Habib Slim
UGA - Grenoble INP
Grenoble, France
habib.slim@grenoble-inp.org

Supervised by: Pr. Georges Quénot.

I understand what plagiarism entails and I declare that this report is my own, original work.

Habib Slim - **June 12, 2020**

Abstract

With the recent progress of Generative Adversarial Networks in the conditional image generation task, GANs could potentially become a viable alternative image source to real images in numerous machine learning tasks. In this work, we investigate classifier training using synthetic images generated from trained BigGANs. We first study the effects of various generation and training methods in the single GAN setting, and measure a significant gap in accuracy between classifiers trained on real and synthetic data. We make an attempt to bridge this gap by using multiple GANs conditioned under a sample reweighting scheme, and find that this method consistently underperforms compared to using multiple unconditioned GANs.

1 Introduction

1.1 SoTA of GANs

From their introduction by [Goodfellow *et al.*, 2014], Generative Adversarial Networks (GANs) have made great progress in generating realistic data by learning from complex and highly dimensional datasets. In particular, the state-of-the-art of GAN-based image generation has greatly improved recently, with the introduction of novel neural network architectures that were shown to be able to generate samples of great quality, some of which very hard to distinguish from real images.

The introduction of conditional generators, in which a class label is given as additional input alongside the latent variable, enabled control over the data generation process. After their introduction in [Mirza and Osindero, 2014], conditional GANs have known multiple leaps in performance, among which we can mention DCGAN [Radford *et al.*, 2015], ImGAN [Salimans *et al.*, 2016] and more recently BigGAN from [Brock *et al.*, 2018] that achieved state-of-the-art results on the ImageNet and CIFAR-10 datasets .

1.2 Related work

In light of those recent developments in quality of the generated images, GANs could now potentially be considered as a viable replacement for datasets, for various applications like classifier training. Replacing training data by a GAN could have applications in cases where information cannot be directly shared for privacy or sensitivity concerns, or when the original dataset is too big and a GAN can be used as a compressed representation of the data.

Some work has already been done in that direction. In [Shmelkov *et al.*, 2018], the authors use multiple GAN models achieving state-of-the-art (at the time) on ImageNet and CIFAR-100, and showed that classifiers trained on synthetic images consistently underperformed compared to those trained on real data. They also suggested the use of classifier accuracy as a score to evaluate the quality of generated samples from GANs.

In [Ravuri and Vinyals, 2019], the authors train a ResNet classifier on BigGAN generated data (without any data transformations), and show that accuracy on the ImageNet and CIFAR-10 datasets is greatly reduced (around 20% additional error for CIFAR-10). They also show that the Fréchet Inception Distance, often used to characterize the proximity of the generated distribution to the original one, does not correlate well with the accuracy of classifiers trained on synthetic data.

In [Pham *et al.*, 2019], a ResNet classifier trained on data generated by multiple BigGANs has been shown to give similar accuracy to a ResNet trained on slightly reduced datasets (around 10% error). In [Besnier *et al.*, 2020], the authors use a pre-trained BigGAN on ImageNet and employ a hard sample mining technique on generated images to get "harder" and more informative samples, which they find has a positive impact on accuracy.

So far, for the same number of optimization steps, classifiers trained on synthetic data have been shown to perform worse than their real data trained counterparts. In this work, trained BigGANs are used to try to bridge this gap, using a single GAN and various training techniques (3), and multiple complementary GANs (5).

2 Background

Generative Adversarial Networks (GANs) are composed of two fundamental blocks: the discriminator D, and the generator G (with parameters θ_d and θ_g respectively). Given a true target distribution p_x and a latent distribution p_z such that $\boldsymbol{x} \sim p_x$ and $\boldsymbol{z} \sim p_z$, D and G play the following minimax game:

$$\min_{\theta_g} \max_{\theta_d} \mathbb{E}_{\boldsymbol{x}\sim p_x} [\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_z} [\log (1 - D(G(\boldsymbol{z}))]$$

In the Conditional GAN setting (cGAN), additional class information is given to the generator and the discriminator. With $(\boldsymbol{x}, \boldsymbol{y}) \sim p_d$, we now have:

$$\min_{\theta_g} \max_{\theta_d} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim p_d} [\log D(\boldsymbol{x}, \boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z} [\log (1 - D(G(\boldsymbol{z}, \boldsymbol{y}))]$$

In practice, BigGAN samples the latent $\boldsymbol{z}$ from a normal distribution $\mathcal{N}(0, I)$, and feeds class information through batch normalization layers for the generator, and by projecting image features onto the class vectors for the discriminator.

Despite its aforementioned limitations, we use the Fréchet Inception Distance (FID) [Heusel *et al.*, 2017] to measure the distance between real images and synthetic images, that computes the distance between two image distributions embedded in the Inception v3 network [Szegedy *et al.*, 2016]:

$$\|\mu_r - \mu_g\|_2^2 + \mathrm{Tr}\left(\Sigma_r + \Sigma_g - 2 \cdot (\Sigma_r \Sigma_g)^{\frac{1}{2}}\right)$$

Where (μ_r, Σ_r) and (μ_g, Σ_g) are the mean vectors and covariance matrices of the real and generated features respectively.

3 Using data from a single GAN

3.1 Approach

In this section, we first experiment making use of various training methods, all based on a single GAN trained on the original data. We compare a ResNet classifier trained on the CIFAR-10 dataset to the same model trained on synthetic data, with the following methods.

On-the-fly generation. Instead of training the classifier on a dataset of fixed size, on-the-fly generation is employed: we train the classifier with different batches of synthetic images at every epoch.

Data transformations. Data transformations are applied to the generated batches. We experiment with a combination of horizontal axis flipping and random cropping.

Sample filtering. In [Pham *et al.*, 2019], the authors propose to use classification accuracy as a filtering metric for the quality of generated samples. To that effect, a classifier C_θ is trained on the original dataset with high accuracy, and used as a filter for synthetic samples. Given a generated sample x and its conditional class label y, we check that the class hypothesized by C_θ is equal to y:

$$\arg\max_{k\in[\![1,K]\!]} C_{\theta,k}(x) = y$$

Samples for which the prediction probability for the correct class is below a given threshold value p_t are also filtered out[1]:

$$C_{\theta,y}(x) \geq p_t$$

Truncated normal. In [Brock *et al.*, 2018], the authors propose an alternative method for sampling the latent variable z, which they call the "truncation trick". When training the GAN, the latent variable is sampled normally with $z \sim \mathcal{N}(0, 1)$, but when evaluating the network the latent is sampled from a truncated normal distribution with interval $[-\psi, \psi]$, where ψ is the truncation threshold. Sampling with lower truncation values is associated with a higher sample quality with a trade-off in diversity. We make tests with $\psi = 1.5$.

3.2 Experiments

Method. We experiment using the CIFAR-10 dataset. We use a BigGAN trained on CIFAR-10, and a pre-trained ResNet classifier used for sample filtering achieving 94.26% accuracy on CIFAR-10, trained using data augmentation.

For our baseline, we consider the same ResNet classifier used for sample filtering, trained on augmented original data with the same number of optimization steps (same number of epochs, number of batches per epoch, batch size).

Results. The results for those first experiments are presented in table 1. We can see that each method improves classification accuracy when used alone and conjointly, except for when sampling from a truncated normal which hints that sample diversity is more important in classification accuracy than sample quality. Overall, applying transformations to the generated data gives the best accuracy improvement. On-the-fly generation seems to perform slightly better than using a dataset of fixed size.

[1] In practice, we experiment with $p_t = 0.9$.

	Fixed Dataset						**On-the-fly Generation**						**Baseline**
Data Transformations	-	✓	-	-	✓	✓	-	✓	-	-	✓	✓	✓
Sample Filtering	-	-	✓	-	✓	✓	-	-	✓	-	✓	✓	
Truncated Normal	-	-	-	✓	-	✓	-	-	-	✓	-	✓	
Accuracy	62.26	79.76	67.98	59.35	**83.84**	78.02	59.48	84.25	66.50	56.98	**88.56**	83.85	**94.26**

Table 1: Accuracy measurements for CIFAR-10, training from images generated using a single BigGAN, and using various methods. All ResNet classifiers are trained for 180 epochs with 400 batches per epoch and a batch size of 125 (to match the size of a 50k images dataset). The best accuracy over the whole training session is given. When using a fixed dataset, we use 50k synthetic images for our training set, and validate using 10k samples. For a complete description of the experimental setup, please refer to appendix A.1.

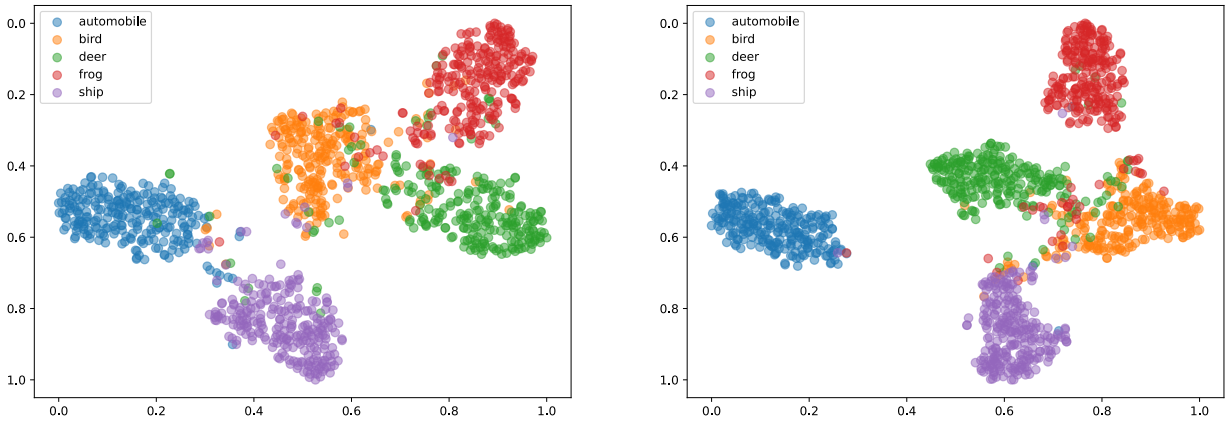


Figure 1: t-SNE embeddings of features extracted from real images (left), and synthetic images (right). Using 256 real and generated images per class, for five selected classes (in the legend). Real images are taken from the CIFAR-10 test set, and generated images are produced from a BigGAN trained with the same parameters as in A.1. Feature vectors are extracted using the same pre-trained ResNet mentioned in 3.2: we average feature maps from the last convolutional layer which gives vectors of dimension 256.

Despite those subsequent improvements, we reach a minimum error rate of 11.5%, compared to the 5.7% error rate achieved when training on real data: the gap is still significant.

In what follows, we will attempt to bridge this gap by using multiple complementary generative networks.

4 Analysis

4.1 Classifying synthetic data

When making the opposite experience as in 3.2, that is evaluating the accuracy of a ResNet trained with real data on generated samples (prior to any transformations/sample filtering), we obtain a validation accuracy of 94.17% which is very close to the value achieved on the original dataset. This suggests that the distribution learned by the BigGAN is within the real data distribution, but not diverse enough to be its fair substitution.

When conducting the same experiment on generated data using the truncation trick ($\psi = 1.5$), we obtain an accuracy of 95.88% (which is even higher than the accuracy of the same ResNet on the CIFAR-10 test set), which further highlights the trade-off in diversity for sample quality.

4.2 Real and synthetic features

In order to compare the distributions of the generated images with real images, we extract features of samples taken from the CIFAR-10 dataset and of generated samples from a BigGAN[2], and compare their distributions using the t-SNE algorithm [Maaten and Hinton, 2008] to reduce dimensionality. Results are shown in figure 1.

We can see that inter-class boundaries are fuzzier with real images compared to generated images, for which boundaries are very clearly defined. Furthermore, intra-class distances are lower with synthetic images, while real images from the same class are more spread out. This effect is further accentuated in synthetic images generated from a truncated latent for which the inter-class distances are even wider, as can be seen in the figure of appendix B.1.

Image Source	MS-SSIM ↓ ($\times 10^2$)	FID ↓
CIFAR-10	40.5 ± 0.8	10.31[4]
BigGAN	42.2 ± 1.6	16.97
BigGAN$_{\psi=1.5}$	44.5 ± 2.6	24.58
BigGAN$_{\psi=0.5}$	61.1 ± 5.8	76.95

Table 2: Intra-class diversity and sample quality.

4.3 Evaluating intra-class diversity

The MS-SSIM score [Wang *et al.*, 2003] has been used before to numerically assess image diversity ([Hu *et al.*, 2020], [Li *et al.*, 2019])[3]. We use this metric here to evaluate intra-class diversity of synthetic samples by summing the similarity scores of 200 image pairs from each class and averaging the result (4000 images per image source, 200 pairs per class). We also compute the FID scores for each image source.

Overall, the similarity score highlights well the slight reduction in diversity when switching from the real dataset to a BigGAN trained on it. Synthetic distributions also suffer from a significant distance from the original image distribution, as shown by the computed FID scores.

[2]This is done prior to using sample filtering or any other method.

[3]The LPIPS [Zhang *et al.*, 2018] score has also been suggested for this specific task, but is not well suited for CIFAR-10 images which are too low in resolution.

[4]FID computed between the training and test sets.

5 Using multiple GANs

5.1 Approach

To produce synthetic image distributions approaching the real distributions, we use multiple GANs trained under different conditions in order to cover more of the real image space. To that end, three main approaches are tested.

Baseline. In [Pham *et al.*, 2019], the authors train multiple GANs with different starting random seeds, and use synthetic images generated from all GANs. We use a similar straightforward method here, and train a set G of GANs. When training a classifier, we randomly select a generator from G for each batch, with equiprobability.

Conditioned GANs. In this method, multiple complementary BigGANs are trained sequentially under different conditions: we use sample reweighting on the training set of the generators, so that real samples that are "far" from the previously generated samples are weighted more. The idea behind this method is simple: we want the secondary GANs to incrementally complete the generated distributions of the previous GANs, by learning to generate the border cases of the previous distribution. To that effect, a simple algorithm is used to compute sample weights.

In what follows, $\mathcal{D}_R$ is the set of real images and $\mathcal{D}_G$ the set of synthetic images, and $\mathcal{D}_{N,k}$ denotes the subset of $\mathcal{D}_N$ of samples belonging to class k. To evaluate a distance between two images, we use their feature vectors: we note $\phi : \mathbb{R}^{32\times32\times3} \to \mathbb{R}^{256}$ the function that associates an image to its feature vector[5]. Finally, w is a mapping associating each real sample to its weight. The complete description is given in algo.1.

Distances are normalized within each class and not globally to ensure class balancing. For the weight transformation function f, we experiment with multiple solutions:

Method A. We put all weights within class k with a value less than a threshold θ to zero. We experiment with $\theta = \mu_k$ and $\theta = \mu_k - \sigma_k$ (where μ_k and σ_k are the average and standard deviation values within weights of class k). We refer to those two variants as A_μ and A_σ respectively.

Method B. To distribute weights more evenly across samples, we apply a sigmoid function to the input weights, defined as:

$$f\left(w;\mu_k,\sigma_k\right) = \left[1+\exp\left(-2\cdot\frac{w-\mu_k}{\sigma_k}\right)\right]^{-1}$$

Method C. We directly use the normalized weights. ($f = \mathrm{Id}$).

In figure 2, a t-SNE visualization of weight distributions within a class of CIFAR-10 is given (with and without thresholding). We can see that this method attributes higher weights to samples at the edge of the image distribution, while images close to the mode are given lower weights.[6]

[5]For details about the output dimension, please refer to the caption of figure 1.

[6]Visualizations of weight distributions for method A_σ can also be found in appendix B.2.

$d_k \colon \mathcal{D}_{R,k} \to \mathbb{R}_{\geq 0}$

$d_k(s) = 0,\ \forall k \in [\![1, K]\!],\ \forall s \in \mathcal{D}_{R,k}$

$w \colon \mathcal{D}_R \to [0, 1]$

for k ***in*** $1...K$ **do**

  for $g \in \mathcal{D}_{G,k} \mid r \in \mathcal{D}_{R,k}$ **do**

    $d_k(r) \leftarrow d_k(r) + \|\phi(r) - \phi(g)\|_2$

  end

  $d_{min} = \min_{s\in\mathcal{D}_{G,k}} d_k(s),\ d_{max} = \max_{s\in\mathcal{D}_{G,k}} d_k(s)$

  for $s \in \mathcal{D}_{R,k}$ **do**

    $w(s) \leftarrow \frac{d_k(s)-d_{min}}{d_{max}}$

  end

  compute μ_k, σ_k **from** w_k

  for $s \in \mathcal{D}_{R,k}$ **do**

    $w(s) \leftarrow f\left(w(s); \mu_k, \sigma_k\right)$

  end

end

Algorithm 1: Sample reweighting algorithm

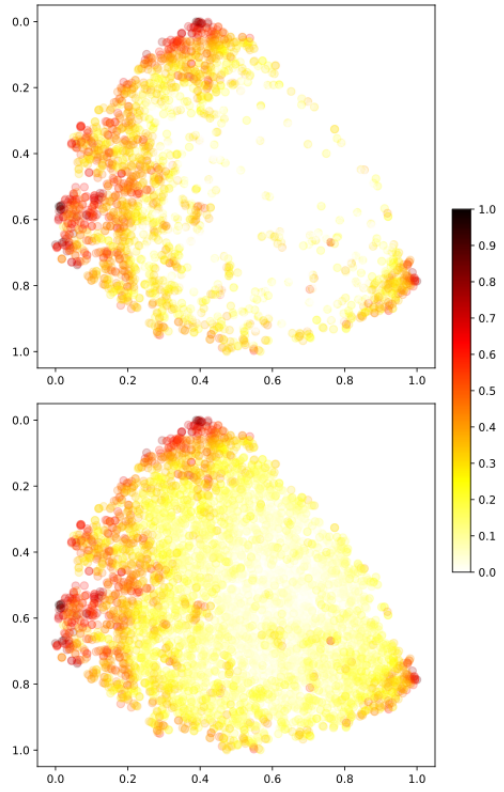


Figure 2: t-SNE visualization of the distribution of weights within class "ship" of CIFAR-10. Top: with thresholding ($\theta = \mu$), bottom: without thresholding.

$\|G\|$	Conditioned GANs				Baseline
	A_μ	A_σ	B	C	
1	-	-	-	-	88.56[7]
2	88.12	89.06	89.27	89.33	**90.24**
2$*$	88.81	89.43	89.56	89.69	
3	88.02	88.57	-	90.25	**90.61**

Table 3: Accuracy results for CIFAR-10.

Image Source	MS-SSIM $\downarrow$ ($\times 10^2$)	FID $\downarrow$
CIFAR-10	40.5 ± 0.8	10.31
MultiGAN	42.4 ± 1.7	17.02
MultiGAN Cond. B	42.4 ± 1.7	17.83
MultiGAN Cond. C	42.4 ± 1.7	18.68
MultiGAN Cond. A_σ	43.1 ± 2.3	21.15
MultiGAN Cond. A_μ	48.1 ± 4.0	88.52

Table 4: MS-SIM and FID scores for $|G| = 2$.

Weighted conditioned GANs. We use the same weighting method as previously, but instead of sampling from each generator with equiprobability, we attribute a weight to each generator in the ensemble. We experiment with this method using two GANs, and weights $w = (\frac{3}{4}, \frac{1}{4})$ to favor the impact of the first generator that should represent a larger portion of the total distribution. In table 3 above, the star sign $*$ beside the number of generators indicates the use of weighted GANs.

5.2 Experiments

Method. We experiment with a maximum of 3 GANs, both for the baseline and the sample reweighting method. We compute sample weights using 50k synthetic and real samples, and apply them as sampling probabilities rather than loss weights. To train the classifiers, we use on-the-fly generation with data transformations and sample filtering, which was the best performing method with a single GAN. FID scores of all of the models trained can be found in appendix A.2.

Results. The results for these experiments can be found in table 3. In addition to accuracy measurements, FID and MS-SSIM scores for two combined GANs in various settings are presented in table 4.

In these experiments, GANs trained on subsets of the support of the real distribution seemed to exhibit undesired properties. The networks collapsed earlier during training compared to their unconditioned counterparts. For the method A_μ, this effect was the most prevalent with a collapse happening around 70% earlier in the training process, which can be imputed to the increased imbalance in the distribution of weights. Methods A_σ, B and C were less subject to early collapse.

Conditioned GANs exhibit higher FID values on average (as seen in appendix A.2) which is expected, however this is not linked with better combined image diversity and higher training accuracies: the conditioned networks are not able to reliably learn the edge cases of the previous distribution to complement the prior network.

[7]Using reported results in table 1 for single-GAN training.

The FID scores indicate that using methods B and C leads to a reduced drift from the initial distribution, which is expected - however, the slight drift in distribution is not compensated by a better sample diversity, which globally penalizes the accuracy.

Overall, weighted GANs consistently underperformed compared to the unconditioned baseline. Weighting the conditioned GANs for $|G| = 2$ only gives marginal improvements, as seen in table 3.

On a side note, using 4 and 5 unconditioned GANs yields accuracies of 90.56 % and 91.04 % respectively, which is still significantly far from the real-data baseline despite the high number of generators used.

6 Conclusion

In this work, we explored different ways to train a classifier from synthetic data generated by conditional GANs. For the single-GAN setting, we have conducted a brief survey of previous techniques used in the literature to train classifiers from synthetic data, and studied the effects of their combination in the accuracy of the resulting classifier. We have shown that a significant gap still exists between classifiers trained on synthetic and real data, and given some elements identifying the reduced intra-class diversity in synthetic images as a likely culprit. Finally, we attempted to use multiple GANs trained incrementally with different sample weights in order to generate the previously uncovered parts of the real distribution, but have found that the resulting generators did not exhibit desired properties, resulting in accuracies for the trained classifier worse than the unconditioned baseline.

7 Future work

A lot of work has been done recently in the direction of improving the diversity of synthetic images produced by GANs. In particular, [Li *et al.*, 2019] proposed an original method involving optimizing the class embeddings of pre-trained BigGAN networks to maximize intra-class diversity. They reported state-of-the-art results regarding realism and diversity of generated samples using the MS-SSIM and LPIPS metrics alongside human evaluation - beating the previous iteration of the BigGAN model on the ImageNet dataset - which is promising in the context of our work.

Improvements could also be considered to the sample weighting method, possibly involving other weight transformation functions or more sophisticated GAN weighting schemes.

Acknowledgment

Experiments presented in this paper were carried out using the Grid'5000 experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr).

References

[Balouek *et al.*, 2013] Daniel Balouek, Alexandra Carpen Amarie, Ghislain Charrier, Frédéric Desprez, Emmanuel Jeannot, Emmanuel Jeanvoine, Adrien Lèbre, David Margery, Nicolas Niclausse, Lucas Nussbaum, Olivier Richard, Christian Pérez, Flavien Quesnel, Cyril Rohr, and Luc Sarzyniec. Adding virtualization capabilities to the Grid'5000 testbed. In Ivan I. Ivanov, Marten van Sinderen, Frank Leymann, and Tony Shan, editors, *Cloud Computing and Services Science*, volume 367 of *Communications in Computer and Information Science*, pages 3–20. Springer International Publishing, 2013.

[Besnier *et al.*, 2020] Victor Besnier, Himalaya Jain, Andrei Bursuc, Matthieu Cord, and Patrick Perez. This dataset does not exist: Training models from generated images. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020.

[Brock *et al.*, 2018] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis, 2018.

[Deng *et al.*, 2009] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[Goodfellow *et al.*, 2014] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Heusel *et al.*, 2017] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2017.

[Hu *et al.*, 2020] Mengxiao Hu, Jinlong Li, Maolin Hu, and Tao Hu. Hierarchical modes exploring in generative adversarial networks, 2020.

[Krizhevsky and others, 2009] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. *University of Toronto*, 2009.

[Li *et al.*, 2019] Qi Li, Long Mai, Michael A. Alcorn, and Anh Nguyen. Improving sample diversity of a pre-trained, class-conditional gan by changing its class embeddings, 2019.

[Maaten and Hinton, 2008] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[Mirza and Osindero, 2014] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014.

[Pham *et al.*, 2019] Thanh Dat Pham, Anuvabh Dutt, Denis Pellerin, and Georges Quénot. Classifier Training from a Generative Model. In *CBMI 2019 - 17th International Conference on Content-Based Multimedia Indexing*, Dublin, Ireland, September 2019.

[Radford *et al.*, 2015] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[Ravuri and Vinyals, 2019] Suman Ravuri and Oriol Vinyals. Seeing is not necessarily believing: Limitations of biggans for data augmentation. *ICLR 2019 Workshop*, 2019.

[Salimans *et al.*, 2016] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016.

[Shmelkov *et al.*, 2018] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. How good is my gan? *CoRR*, abs/1807.09499, 2018.

[Szegedy *et al.*, 2016] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[Wang *et al.*, 2003] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003.

[Zhang *et al.*, 2018] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

A Experimental setup

A.1 Single GAN training

Generator. We use skip-z connections, shared embeddings and orthogonal normalization with a batch size of 64 to train the network. We use a hierarchical latent z, with $z \in \mathbb{R}^{80}$ and shared embeddings of the same dimension. GANs are trained for 12 hours sessions, which amounts to $\sim 100k$ iterations using two GPUs, and a learning rate of $2 \cdot 10^{-4}$ for both G and D, and keep the best G obtained across the whole training session. We use a fork of the original BigGAN implementation in PyTorch.

Classifier. For the classifier, a ResNet20 is trained with a width of 64, with cross entropy loss and SGD with learning rate of 0.1, momentum of 0.9 and weight decay of $1 \cdot 10^{-4}$. We train classifiers for 180 epochs, with 400 batches per epoch and a batch size of 125. We divide the learning rate by 10 at epochs 100 and 150.

A.2 FID scores

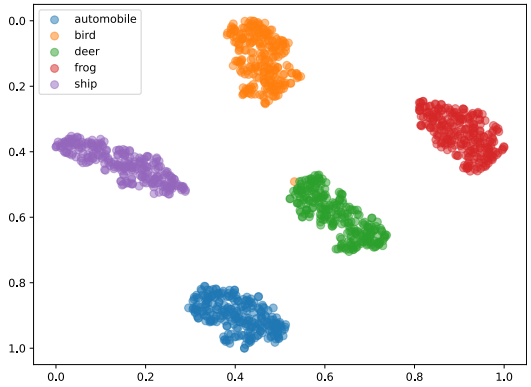
FID scores for individual GANs (noted "BigGAN") trained with sample reweighting schemes A_μ, A_σ, etc., and for GANs ensembles (noted "MultiGAN")[8].

Model	FID	Model	FID
BigGAN	16.97	BigGAN C, it2	17.66
BigGAN$_{\psi=1.5}$	24.58	MultiGAN $A_\mu, n=2$	88.52
BigGAN$_{\psi=0.5}$	76.96	MultiGAN $A_\sigma, n=2$	21.15
BigGAN B, it1	19.62	MultiGAN $B, n=2$	17.83
BigGAN A_μ, it1	239.52	MultiGAN $C, n=2$	18.68
BigGAN A_μ, it2	172.81	MultiGAN $n=2$	17.03
BigGAN A_σ, it1	26.96	MultiGAN $n=3$	17.08
BigGAN A_σ, it2	86.96	MultiGAN $n=4$	17.36
BigGAN C, it1	21.12	MultiGAN $n=5$	17.45

B Additional figures

B.1 t-SNE for truncated latent

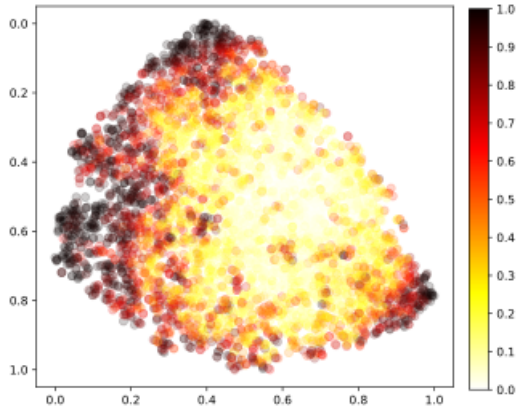
t-SNE embeddings from images generated using a truncated latent, $\psi = 0.5$:



[8]FID scores are computed using the "5k" method, that is by computing the moments for 5000 generated samples and 5000 samples from the test set, and applying the Wasserstein-2 formula.

B.2 Weight distribution

t-SNE visualization of the distribution of weights within class "ship" using a sigmoid transformation on computed distances:



B.3 Generated samples

Effects of truncation. Samples generated without truncation (left), and with truncation (right, $\psi = 0.5$).

Samples generated with truncation exhibit very low diversity. We observe higher similarity in poses, colors and backgrounds, compared to samples generated without truncation.

Mode collapse. Typical mode collapse in a GAN trained with method A_μ, after $\sim 30k$ iterations.