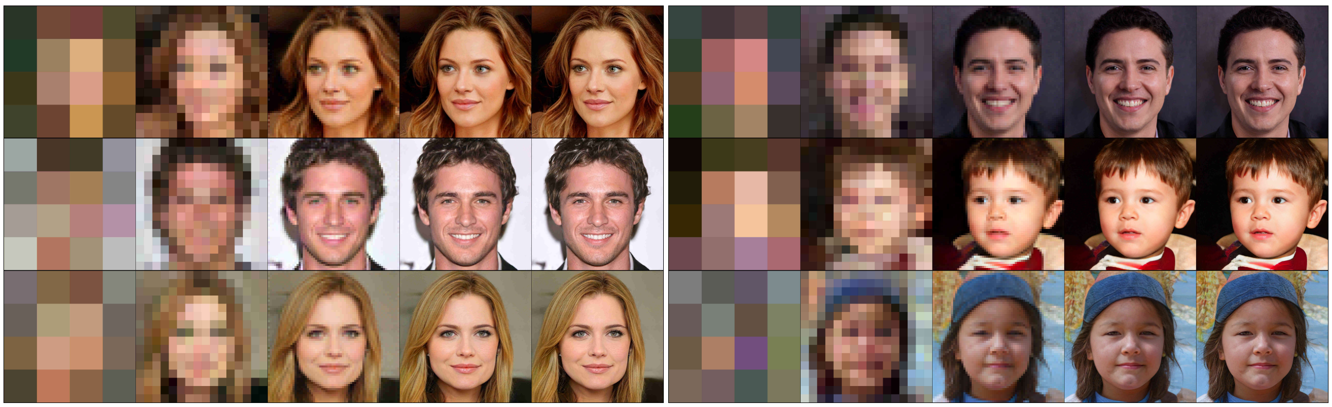
MSG-GAN: Multi-Scale Gradients for Generative Adversarial Networks

Animesh Karnewar
TomTom
animesh.karnewar@tomtom.com

Oliver Wang
Adobe Research
owang@adobe.com



CelebA-HQ FFHQ

Figure 1: Results of our proposed MSG-GAN technique where the generator synthesizes images at all resolutions simultaneously and gradients flow directly to all levels from a single discriminator. The first column has a resolution of `4x4` which increases towards the right reaching the final output resolution of `1024x1024`. Best viewed zoomed in on screen.

Abstract

While Generative Adversarial Networks (GANs) have seen huge successes in image synthesis tasks, they are notoriously difficult to adapt to different datasets, in part due to instability during training and sensitivity to hyperparameters. One commonly accepted reason for this instability is that gradients passing from the discriminator to the generator become uninformative when there isn't enough overlap in the supports of the real and fake distributions. In this work, we propose the Multi-Scale Gradient Generative Adversarial Network (MSG-GAN), a simple but effective technique for addressing this by allowing the flow of gradients from the discriminator to the generator at multiple scales. This technique provides a stable approach for high resolution image synthesis, and serves as an alternative to the commonly used progressive growing technique. We show that MSG-GAN converges stably on a variety of image datasets of different sizes, resolutions and domains, as well as different types of loss functions and architectures, all with the same set of fixed *hyperparameters. When compared to state-of-the-art GANs, our approach matches or exceeds the performance in most of the cases we tried.*

1. Introduction

Since their introduction by Goodfellow *et al.* [10], Generative Adversarial Networks (GANs) have become the de facto standard for high quality image synthesis. The success of GANs comes from the fact that they do not require manually designed loss functions for optimization, and can therefore learn to generate complex data distributions without the need to be able to explicitly define them. While flow-based models such as [6, 7, 27, 18] and autoregressive models such as [32, 31, 29] allow training generative models directly using Maximum Likelihood Estimation (explicitly and implicitly respectively), the fidelity of the generated images has not yet been able to match that of the state-of-the-art GAN models [15, 16, 17, 3]. However, GAN training suffers from two prominent problems: (1) mode collapse and (2) training instability.

The problem of mode collapse occurs when the generator network is only able to capture a subset of the variance present in the data distribution. Although numerous works [28, 41, 15, 21] have been proposed to address this problem, it remains an open area of study. In this work, however, we address the problem of *training instability*. This is a fundamental issue with GANs, and has been widely reported by

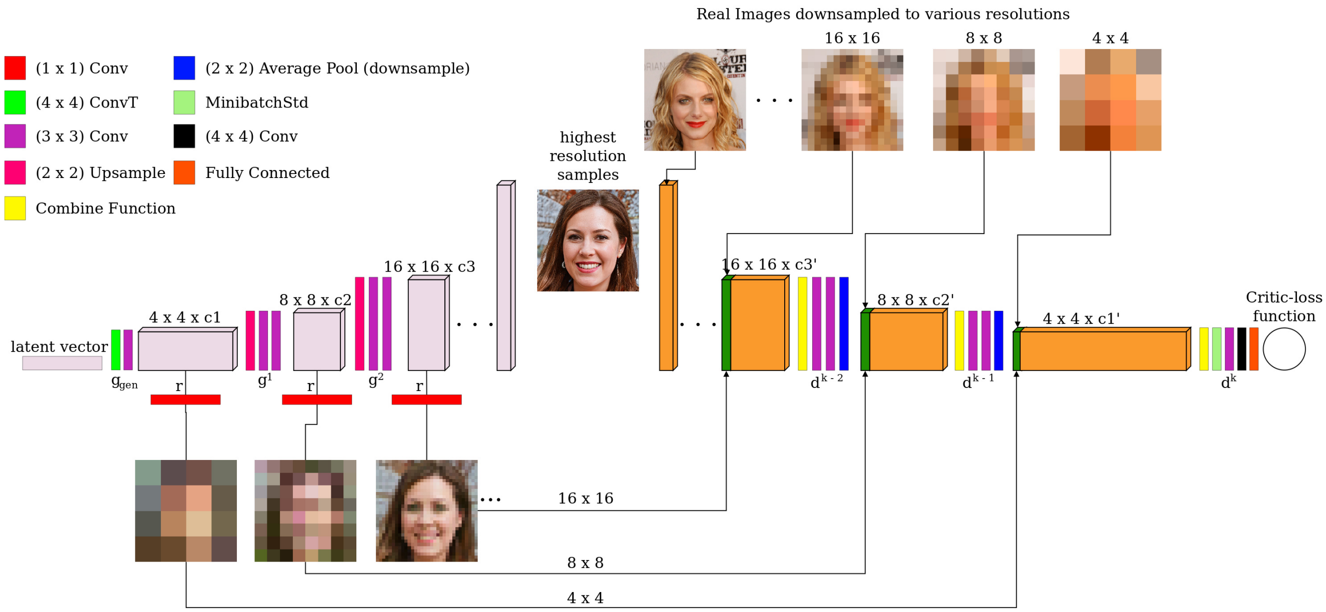


Figure 2: Architecture of MSG-GAN, shown here on the base model proposed in ProGANs [15]. Our architecture includes connections from the intermediate layers of the generator to the intermediate layers of the discriminator. Multi-scale images sent to the discriminator are concatenated with the corresponding activation volumes obtained from the main path of convolutional layers followed by a combine function (shown in yellow).

previous works [28, 22, 2, 11, 19, 33, 14, 15, 37, 25]. We propose a method to address training instability for the task of image generation by investigating how gradients at multiple scales can be used to generate high resolution images (typically more challenging due to the data dimensionality) without relying on previous greedy approaches, such as the progressive growing technique [15, 16]. MSG-GAN allows the discriminator to look at not only the final output (highest resolution) of the generator, but also at the outputs of the intermediate layers (Fig. 2). As a result, the discriminator becomes a function of multiple scale outputs of the generator and importantly, *passes gradients* to all the scales simultaneously (more details in section 1.1 and section 2).

Furthermore, our method is robust to different loss functions (we show results on WGAN-GP and Non-saturating GAN loss with 1-sided gradient penalty), datasets (we demonstrate results on a wide range of commonly used datasets and a newly created Indian Celebs dataset), and architectures (we integrate the MSG approach with both ProGANs and StyleGAN base architectures). Much like progressive growing [15], we note that multi-scale gradients account for a considerable improvement in FID score over the vanilla DCGAN architecture. However, our method achieves better performance with comparable training time to state-of-the-art methods on most existing datasets without requiring the extra hyperparameters that progressive growing introduces, such as training schedules and learning rates for different generation stages (resolutions). This robustness allows the MSG-GAN approach to be easily used "out-of-the-box" on new datasets. We also show the importance of the multi-scale connections on multiple generation stages (coarse, medium, and fine), through ablation experiments on the high resolution FFHQ dataset.

In summary, we present the following contributions. First, we introduce a multiscale gradient technique for image synthesis that improves the stability of training as defined in prior work. Second, we show that we can robustly generate high quality samples on a number of commonly used datasets, including CIFAR10, Oxford102 flowers, CelebA-HQ, LSUN Churches, Flickr Faces HQ and our new Indian Celebs all with the same *fixed* hyperparameters. This makes our method easy to use in practice.

1.1. Motivation

Arjovsky and Bottou [1] pointed out that one of the reasons for the training instability of GANs is due to the passage of random (uninformative) gradients from the discriminator to the generator when there is insubstantial overlap between the supports of the real and fake distributions. Since the inception of GANs, numerous solutions have been proposed to this problem. One early example proposes adding instance noise to the real and the fake images so that the supports minimally overlap [1, 30]. More recently, Peng *et al*. [25] proposed a mutual information bottleneck between input images and the discriminator's deepest representation of those input images called the variational discriminator bottleneck (VDB) [25], and Karras *et al.* [15] proposed a progressive growing technique to add continually increasing resolution layers. The VDB solution forces the discriminator to focus only on the most discerning features of the images for classification, which can be viewed as an adaptive variant of instance noise. Our work is orthogonal to the VDB technique, and we leave an investigation into a combination of MSG-GAN and VDB to future work.

The progressive growing technique tackles the instability problem by training the GAN layer-by-layer by gradually doubling the operating resolution of the generated images. Whenever a new layer is added to the training it is slowly faded in such that the learning of the previous layers are retained. Intuitively, this technique helps with the support overlap problem because it first achieves a good distribution match on lower resolutions, where the data dimensionality is lower, and then *partially-initializes* (with substantial support overlap between real and fake distributions) higher resolution training with these previously trained weights, focusing on learning finer details.

While this approach is able to generate state-of-the-art results, it can be hard to train, due to the addition of hyper-parameters to be tuned per resolution, including different iteration counts, learning rates (which can be different for the Generator and Discriminator [12]) and the fade-in iterations. In addition, a concurrent submission [17] discovered that it leads to phase artifacts where certain generated features are attached to specific spatial locations. Hence our main motivation lies in addressing these problems by providing a simpler alternative that leads to high quality results and stable training.

Although the current state-of-the-art in class conditional image generation on the Imagenet dataset, i.e. BigGAN [4], doesn't employ multi-scale image generation, note that the highest resolution they operate on is `512x512`. All high resolution state-of-the-art methods [15, 16, 17, 34, 40] use some or the other form of multi-scale image synthesis. Multi-scale image generation is a well established technique, with methods existing well before deep networks became popular for this task [20, 35]. More recently, a number of GAN-based methods break the process of high resolution image synthesis into smaller subtasks [36, 39, 38, 8, 9, 15, 34, 40]. For example, LR-GAN [36] uses separate generators for synthesizing the background, foreground and compositor masks for the final image. Works such as GMAN and StackGAN employ a single generator and multiple discriminators for variation in teaching and multi-scale generation respectively [8, 39, 38]. MAD-GAN [9], instead uses multiple generators to address mode-collapse by training a multi-agent setup in such a way that different generators capture different modalities in the training dataset. LapGAN [5] models the difference between the generated multi-scale components of a Laplacian pyramid of the images using a single generator and multiple discriminators for different scales. Pix2PixHD [34] uses three architecturally similar discriminators acting upon three different resolutions of the images obtained by down-sampling the real and the generated images.

Our proposed method draws architectural inspiration from all these works and builds upon their teachings and ideologies, but has some key differences. In MSG-GAN, we use a single discriminator and a single generator with multi-scale connections, which allows for the gradients to flow at multiple resolutions simultaneously. There are several advantages (driven largely by the simplicity) of the proposed approach. If multiple discriminators are used at each resolution [39, 38, 5, 40, 34], the total parameters grow exponentially across scales, as repeated downsampling layers are needed, whereas in MSG-GAN the relationship is linear. In addition, multiple discriminators with different effective fields [34, 40] are not able to share information across scales, which could make the task easier. Besides having fewer parameters and design choices required, our approach also avoids the need for an explicit color consistency regularization term across images generated at multiple scales, which was necessary, *e.g.* in StackGAN [38].

2. Multi-Scale Gradient GAN

We conduct experiments with the MSG-GAN framework applied to two base architectures, ProGANs [15] and StyleGAN [16]. We call these two methods MSG-ProGAN and MSG-StyleGAN respectively. Despite the name, there is no progressive growing used in any of the MSG variants, and we note that ProGANs without progressive growing is essentially the DCGAN [26] architecture. Figure 2 shows an overview of our MSG-ProGAN architecture, which we define in more detail in this section, and include the MSG-StyleGAN model details in the supplemental material.

Let the initial block of the generator function g_{gen} be defined as $g_{gen} : Z \mapsto A_{begin}$, such that sets Z and A_{begin} are respectively defined as $Z = \mathbb{R}^{512}$, where $z \sim N(0, \mathbb{I})$ such that $z \in Z$ and $A_{begin} = \mathbb{R}^{4\times4\times512}$ contains `[4x4x512]` dimensional activations. Let g^i be a generic function which acts as the basic generator block, which in our implementation consists of an upsampling operation followed by two conv layers.

$$g^i : A_{i-1} \mapsto A_i \tag{1}$$

$$\text{where, } A_i = \mathbb{R}^{2^{i+2}\times 2^{i+2}\times c_i} \tag{2}$$

$$\text{and, } i \in \mathbb{N}; A_0 = A_{begin} \tag{3}$$

where c_i is the number of channels in the i^{th} intermediate activations of the generator. We provide the sizes of c_i in all layers in the supplementary material. The full generator

$GEN(z)$ then follows the standard format, and can be defined as a sequence of compositions of k such g functions followed by a final composition with g_{gen}:

$$y' = GEN(z) = g^k \circ g^{k-1} \circ ...g^i \circ ...g^1 \circ g_{gen}(z). \quad (4)$$

We now define the function r which generates the output at different stages of the generator (red blocks in Fig. 2), where the output corresponds to different downsampled versions of the final output image. We model r simply as a (`1x1`) convolution which converts the intermediate convolutional activation volume into images.

$$r^i : A_i \mapsto O_i \quad (5)$$

$$\text{where, } O_i = \mathbb{R}^{2^{i+2} \times 2^{i+2} \times 3}_{[0-1]} \quad (6)$$

$$\text{hence, } r^i(g^i(z)) = r^i(a_i) = o_i \quad (7)$$

$$\text{where, } a_i \in A_i \text{ and } o_i \in O_i \quad (8)$$

In other words, o_i is an image synthesized from the output of the i^{th} intermediate layer of the generator a_i. Similar to the idea behind progressive growing [15], r can be viewed as a regularizer, requiring that the learned feature maps are able to be projected directly into RGB space.

Now we move on to defining the discriminator. Because the discriminator's final critic loss is a function of not only the final output of the generator y', but also the intermediate outputs o_i, gradients can flow from the intermediate layers of the discriminator to the intermediate layers of the generator. We denote all the components of the discriminator function with the letter d. We name the final layer of the discriminator (which provides the critic score) $d_{critic}(z')$, and the function which defines the first layer of the discriminator $d^0(y)$ or $d^0(y')$, taking the real image y (true sample) or the highest resolution synthesized image y' (fake sample) as the input. Similarly, let d^j represent the intermediate layer function of the discriminator. Note that i and j are always related to each other as $j = k - i$. Thus, the output activation volume a'_j of any j^{th} intermediate layer of the discriminator is defined as:

$$a'_j = d^j(\phi(o_{k-j}, a'_{j-1})) \quad (9)$$

$$= d^j(\phi(o_i, a'_{j-1})), \quad (10)$$

where ϕ is a function used to combine the output o_i of the $(i)^{th}$ intermediate layer of the generator (or correspondingly downsampled version of the highest resolution real image y) with the corresponding output of the $(j-1)^{th}$ intermediate layer in the discriminator. In our experiments, we experimented with three different variants of this combine function:

$$\phi_{simple}(x_1, x_2) = [x_1; x_2] \quad (11)$$

$$\phi_{lin\_cat}(x_1, x_2) = [r'(x_1); x_2] \quad (12)$$

$$\phi_{cat\_lin}(x_1, x_2) = r'([x_1; x_2]) \quad (13)$$

where, r' is yet another (`1x1`) convolution operation similar to r and $[;]$ is a simple channelwise concatenation operation. We compare these different combine functions in Sec 4.

The final discriminator function is then defined as:

$$DIS(y', o_0, o_1, ...o_i, ...o_{k-1}) = \quad (14)$$

$$d_{critic} \circ d^k(., o_0) \circ d^{k-1}(., o_1) \circ ...d^j(., o_i) \circ ...d^0(y') \quad (15)$$

We experimented with two different loss functions for the d_{critic} function namely, WGAN-GP [11] which was used by ProGAN [15] and Non-saturating GAN loss with 1-sided GP [10, 23] which was used by StyleGAN [16]. Please note that since the discriminator is now a function of multiple input images generated by the generator, we modified the gradient penalty to be the average of the penalties over each input.

3. Experiments

While evaluating the quality of GAN generated images is not a trivial task, the most commonly used metrics today are the Inception Score (IS, higher is better) [28] and Fréchet Inception Distance (FID, lower is better) [12]. In order to compare our results with the previous works, we use the IS for the CIFAR10 experiments and the FID for the rest of the experiments, and report the "number of real images shown" as done in prior work [15, 16].

New Indian Celebs Dataset In addition to existing datasets, we also collect a new dataset consisting of Indian celebrities. To this end, we collected the images using a process similar to CelebA-HQ. First, we downloaded images for Indian celebrities by scraping the web for related search queries. Then, we detected faces using an off the shelf face-detector and cropped and resized all the images to `256x256`. Finally, we manually cleaned the images by filtering out low-quality, erroneous, and low-light images. In the end, the dataset contained only 3K samples, an order of magnitude less than CelebA-HQ.

3.1. Implementation Details

We evaluate our method on a variety of datasets of different resolutions and sizes (number of images); **CIFAR10** (60K images at `32x32` resolution); **Oxford flowers** (8K images at `256x256`), **LSUN** churches (126K images at `256x256`), **Indian Celebs** (3K images at `256x256` resolution), **CelebA-HQ** (30K images at `1024x1024`) and **FFHQ** (70K images at `1024x1024` resolution).

For each dataset, we use the same initial latent dimensionality of **512**, drawn from a standard normal distribution $N(0, \mathbb{I})$ followed by hypersphere normalization [15]. For all experiments, we use the same hyperparameter settings for MSG-ProGAN and MSG-StyleGAN (lr=0.003), with

(a) LSUN churches (b) Indian Celebs (c) Oxford Flowers

Figure 3: Random, uncurated samples generated by MSG-StyleGAN on different mid-level resolution (`256x256`) datasets. Our approach generates high quality results across all datasets with the same hyperparameters. Best viewed zoomed in on screen.

Dataset	Size	Method	# Real Images	GPUs used	Training Time	FID (↓)
Oxford Flowers (`256x256`)	8K	ProGANs*	10M	1 V100-32GB	104 hrs	60.40
		MSG-ProGAN	1.7M	1 V100-32GB	44 hrs	28.27
		StyleGAN*	7.2M	2 V100-32GB	33 hrs	64.70
		MSG-StyleGAN	1.6M	2 V100-32GB	16 hrs	**19.60**
Indian Celebs (`256x256`)	3K	ProGANs*	9M	2 V100-32GB	37 hrs	67.49
		MSG-ProGAN	2M	2 V100-32GB	34 hrs	36.72
		StyleGAN*	6M	4 V100-32GB	18 hrs	61.22
		MSG-StyleGAN	1M	4 V100-32GB	7 hrs	**28.44**
LSUN Churches (`256x256`)	126K	StyleGAN*	25M	8 V100-16GB	47 hrs	6.58
		MSG-StyleGAN	24M	8 V100-16GB	50 hrs	**5.2**

Table 1: Experiments on mid-level resolution (*i.e.* `256x256`) datasets. We use author provided scores where possible, and otherwise train models with the official code and recommended hyperparameters (denoted "*")

the only differences being the number of upsampling layers (fewer for lower resolution datasets).

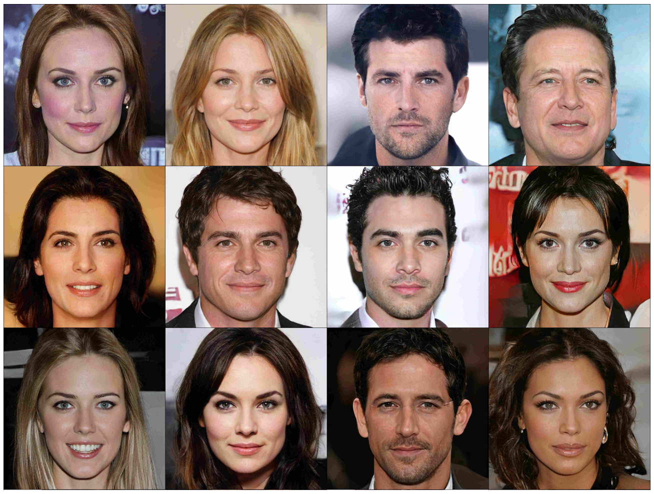
All models were trained with **RMSprop** (lr=0.003) for generator and discriminator. We initialize parameters according to the standard normal $N(0, \mathbb{I})$ distribution. To match previously published work, StyleGAN and MSG-StyleGAN models were trained with Non-saturating GAN loss with 1-sided GP while ProGANs and MSG-ProGAN models were trained with the WGAN-GP loss function.

We also extend the MinBatchStdDev technique [15, 16], where the average standard deviation of a batch of activations is fed to the discriminator to improve sample diversity, to our multiscale setup. To do this, we add a separate MinBatchStdDev layer at the beginning of each block in the discriminator. This way, the discriminator obtains batch-statistics of the generated samples along with the straight-path activations at each scale, and can detect some degree of mode collapse by the generator.

When we trained the models ourselves, we report training time and GPUs used. We use the same machines for corresponding set of experiments so that direct training time comparisons can be made. Please note that the variation in numbers of real images shown and training time is because, as is common practice, we report the best FID score obtained in a fixed number of iterations, and the time that it took achieve that score. All the code and the trained models required for reproducing our work are made available for research purposes at `https://github.com/akanimax/msg-stylegan-tf`.

3.2. Results

Quality Table 1 shows quantitative results of our method on various mid-level resolutions datasets. Both our MSG-ProGAN and MSG-StyleGAN models achieve better FID



(a) CelebA-HQ

(b) FFHQ

Figure 4: Random, uncurated samples generated by MSG-StyleGAN on high resolution (`1024x1024`) datasets. Best viewed zoomed in on screen.

Dataset	Size	Method	# Real Images	GPU Used	Training Time	FID (↓)
CelebA-HQ (`1024x1024`)	30K	ProGANs [16]	12M	-	-	**7.79**
		MSG-ProGAN	3.2M	8 V100-16GB	1.5 days	8.02
		StyleGAN [16]	25M	-	-	**5.17**
		MSG-StyleGAN	11M	8 V100-16GB	4 days	6.37
FFHQ (`1024x1024`)	70K	ProGANs*	12M	4 V100-32GB	5.5 days	9.49
		ProGANs [15]	12M	-	-	**8.04**
		MSG-ProGAN	6M	4 V100-32GB	6 days	8.36
		StyleGAN*	25M	4 V100-32GB	6 days	4.47
		StyleGAN [16]	25M	-	-	**4.40**
		MSG-StyleGAN	9.6M	4 V100-32GB	6 days	5.8

Table 2: Experiments on high resolution (`1024x1024`) datasets. We use author provided scores where possible, and otherwise train models with the official code and recommended hyperparameters (denoted "*").

scores than the respective baselines of ProGANs and StyleGAN on the (`256x256`) resolution datasets of Oxford Flowers, LSUN Churches and Indian Celebs. While each iteration of MSG-GAN is slower than the initial lower resolution iterations of progressive growing, due to all layers being trained together, MSG-GAN tends to converge in fewer iterations, requiring fewer total hours of GPU training time to achieve these scores. Figure 3 shows random samples generated on these datasets for qualitative evaluation.

For high-resolution experiments (Table 2), the MSG-ProGAN model trains in comparable amount of time and gets similar scores on the CelebA-HQ and the FFHQ datasets (**8.02** vs **7.79**) and (**8.36** vs **8.04**) respectively. We note a small difference in the author reported scores and what we were able to achieve with the author provided code. This could be due to subtle hardware differences or variance between runs. Our MSG-StyleGAN model was unable to beat the FID score of StyleGAN on the CelebA-HQ dataset (**6.37** vs **5.17**) and the FFHQ dataset (**5.8** vs **4.40**). We discuss some hypotheses for why this might be in Sec 4, but note that our method does have other advantages, namely that it seems to be easier to generalize to different datasets as shown in our other experiments. Also, our generated images do not show any traces of the phase artifacts [17] which are prominently visible in progressively grown GANs.

Stability during training To compare the stability of MSG-ProGAN with ProGANs during training, we measure the changes in the generated samples for the same fixed latent points as iterations progress (on CelebA-HQ dataset). This method was introduced by [37] as a way to measure stability during training, which we quantify by calculat-

Figure 5: During training, all the layers in the MSG-GAN synchronize across the generated resolutions fairly early in the training and subsequently improve the quality of the generated images at all scales simultaneously. Throughout the training the generator makes only minimal incremental improvements to the images generated from fixed latent points.

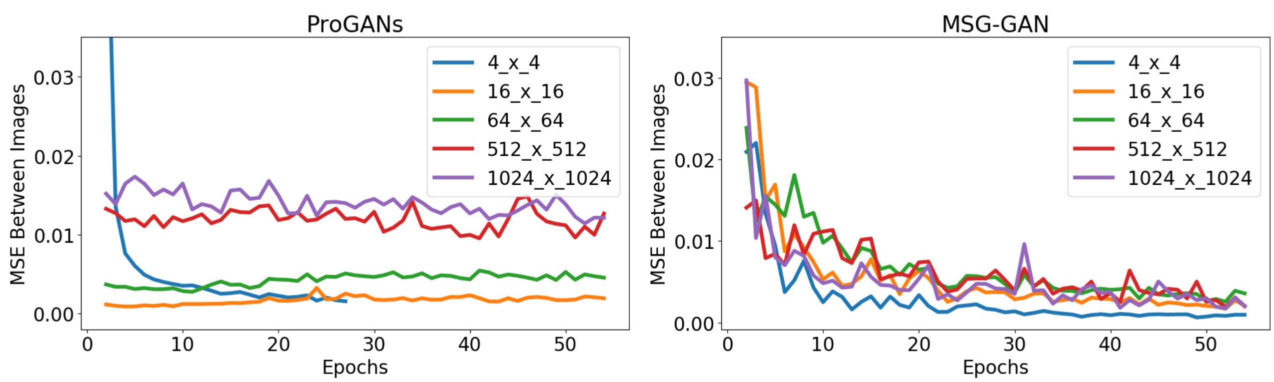


Figure 6: Image stability during training. These plots show the MSE between images generated from the same latent code at the beginning of sequential epochs (averaged over 36 latent samples) on the CelebA-HQ dataset. MSG-ProGAN converges stably over time while ProGANs [15] continues to vary significantly across epochs.

ing the mean squared error between two consecutive samples. Figure 6 shows that while ProGANs tends towards convergence (making less changes) for lower resolutions only, MSG-ProGAN shows the same convergence trait for all the resolutions. The training epochs for the ProGANs take place in sequence over each resolution, whereas for the MSG-ProGAN they are simultaneous (Fig. 5). While not necessary for generating good results, methods with high stability can be advantageous in that it is easier to get a reasonable estimate for how the final result will look by visualizing snapshots during training, which can help when training jobs take on the order of days to weeks.

Robustness to learning rate It has been observed by prior work [28, 14, 24, 23] and also our experience, that convergence of GANs during training is very heavily dependant on the choice of hyperparameters, in particular, learning rate. To validate the robustness of MSG-ProGAN,

Method	# Real Images	Learning rate	IS (↑)
Real Images	-	-	11.34
MSG-ProGAN	12M	0.003	**8.63**
MSG-ProGAN	12M	0.001	8.24
MSG-ProGAN	12M	0.005	8.33
MSG-ProGAN	12M	0.01	7.92

Table 3: Robustness to learning rate on CIFAR-10. We see that our approach converges to similar IS scores over a range of learning rates.

Level of Multi-scale connections	FID (↓)
No connections (DC-GAN)	14.20
Coarse Only	10.84
Middle Only	9.17
Fine Only	9.74
All (MSG-ProGAN)	**8.36**
ProGAN*	9.49

Table 4: Ablation experiments for varying degrees of multiscale gradient connections on the high resolution (`1024x1024`) FFHQ dataset. Coarse contains connections at (`4x4`) and (`8x8`), middle at (`16x16`) and (`32x32`); and fine at (`64x64`) till (`1024x1024`).

Method	Combine function	FID (↓)
MSG-ProGAN	ϕ_{lin_cat}	11.88
	ϕ_{cat_lin}	9.63
	ϕ_{simple}	**8.36**
MSG-StyleGAN	ϕ_{simple}	6.46
	ϕ_{lin_cat}	6.12
	ϕ_{cat_lin}	**5.80**

Table 5: Experiments with different combine functions on the high resolution (`1024x1024`) FFHQ dataset.

we trained our network with four different learning rates (0.001, 0.003, 0.005 and 0.01) for the CIFAR-10 dataset (Table. 3). We can see that all of our four models converge, producing sensible images and similar inception scores, even with large changes in learning rate. Robust training schemes are significant as they indicate how easily a method can be generalized to unseen datasets.

4. Discussion

Ablation Studies We performed two types of ablations on the MSG-ProGAN architecture. Table 4 summarizes our experiments on applying ablated versions of the Multi-Scale Gradients, where we only add subsets of the connections from the generator to the discriminator at different scales. We can see that adding multi-scale gradients at any level to the ProGANs/DCGAN architecture improves the FID score. Interestingly, adding only mid-level connections performs slightly better than adding only coarse or fine-level connections, however the overall best performance is achieved with the connections at all levels.

Table 5 presents our experiments with the different variants of the combine function ϕ on the MSG-ProGAN and the MSG-StyleGAN architectures. ϕ_{simple} (Eq 11) performed best on the MSG-ProGAN architecture while the ϕ_{cat_lin} (Eq 13) has the best FID score on the MSG-StyleGAN architecture. All results shown in this work employ these respective combine functions. We can see through these experiments that the combine function also plays an important role in the generative performance of the model, and it is possible that a more advanced combine function such as multi-layer densenet or AdaIN [13] could improve the results even further.

Limitations and Future Work Our method is not without limitations. We note that using progressive training, the first set of iterations at lower resolutions take place much faster, whereas each iteration of MSG-GAN takes the same amount of time. However, we observe that MSG-GAN requires fewer total iterations to reach the same FID, and often does so after a similar length of total training time.

In addition, because of our multi-scale modification in MSG-StyleGAN, our approach cannot take advantage of the mixing regularization trick [16], where multiple latent vectors are mixed and the resulting image is forced to be realistic by the discriminator. This is done to allow the mixing of different styles at different levels at test time, but also improves overall quality. Interestingly, even though we do not explicitly enforce mixing regularization, our method is still able to generate plausible mixing results (see supplementary material).

Conclusion Although huge strides have been made towards photo-realistic high resolution image synthesis [3, 16, 17], true photo-realism has yet to be achieved, especially with regards to domains with substantial variance in appearance. In this work, we presented the MSG-GAN technique which contributes to these efforts with a simple approach to enable high resolution multi-scale image generation with GANs.

5. Acknowledgements

We would like to thank Alexia Jolicoeur-Martineau (Ph.D. student at MILA) for her guidance over Relativism in GANs and for proofreading the paper. Finally we extend special thanks to Michael Hoffman (Sr. Mgr. Software Engineering, TomTom) for his support and motivation.

References

[1] Martín Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *CoRR*, 2017.

[2] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, 2017.

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

[5] Emily L. Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, 2015.

[6] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: non-linear independent components estimation. *CoRR*, 2014.

[7] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. *CoRR*, 2016.

[8] Ishan Durugkar, Ian Gemp, and Sridhar Mahadevan. Generative multi-adversarial networks. *arXiv preprint arXiv:1611.01673*, 2016.

[9] Arnab Ghosh, Viveka Kulharia, Vinay P. Namboodiri, Philip H.S. Torr, and Puneet K. Dokania. Multi-agent diverse generative adversarial networks. In *CVPR*, June 2018.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, 2014.

[11] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein GANs. In *Advances in Neural Information Processing Systems*, 2017.

[12] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017.

[13] Xun Huang and Serge J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. *ICCV*, 2017.

[14] Alexia Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard GAN. In *International Conference on Learning Representations*, 2019.

[15] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018.

[16] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, pages 4401–4410, 2019.

[17] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan, 2019.

[18] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, 2018.

[19] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of GANs. *arXiv preprint arXiv:1705.07215*, 2017.

[20] Sylvain Lefebvre and Hugues Hoppe. Parallel controllable texture synthesis. In *ACM Transactions on Graphics (ToG)*, volume 24. ACM, 2005.

[21] Zinan Lin, Ashish Khetan, Giulia Fanti, and Sewoong Oh. PacGAN: The power of two samples in generative adversarial networks. In *Advances in Neural Information Processing Systems*, 2018.

[22] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, and Zhen Wang. Multi-class generative adversarial networks with the l2 loss function. *ArXiv*, abs/1611.04076, 2016.

[23] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Which training methods for GANs do actually converge? In *ICML*, 2018.

[24] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks, 2016.

[25] Xue Bin Peng, Angjoo Kanazawa, Sam Toyer, Pieter Abbeel, and Sergey Levine. Variational discriminator bottleneck: Improving imitation learning, inverse RL, and GANs by constraining information flow. In *ICLR*, 2019.

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

[27] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *ICML*, 2015.

[28] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in neural information processing systems*, 2016.

[29] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. PixelCNN++: A PixelCNN implementation with discretized logistic mixture likelihood and other modifications. In *ICLR*, 2017.

[30] Casper Kaae Sønderby, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár. Amortised MAP inference for image super-resolution. *ArXiv*, abs/1610.04490, 2016.

[31] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in Neural Information Processing Systems*, 2016.

[32] Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *CoRR*, abs/1601.06759, 2016.

[33] Ruohan Wang, Antoine Cully, Hyung Jin Chang, and Yiannis Demiris. Magan: Margin adaptation for generative adversarial networks. *ArXiv*, abs/1704.03817, 2017.

[34] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[35] Yonatan Wexler, Eli Shechtman, and Michal Irani. Space-time completion of video. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (3), 2007.

[36] Jianwei Yang, Anitha Kannan, Dhruv Batra, and Devi Parikh. Lr-gan: Layered recursive generative adversarial networks for image generation. *ICLR*, 2017.

[37] Yasin Yazıcı, Chuan-Sheng Foo, Stefan Winkler, Kim-Hui Yap, Georgios Piliouras, and Vijay Chandrasekhar. The unusual effectiveness of averaging in GAN training. In *International Conference on Learning Representations*, 2019.

[38] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *arXiv: 1710.10916*, 2017.

[39] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, Oct 2017.

[40] Zizhao Zhang, Yuanpu Xie, and Lin Yang. Photographic text-to-image synthesis with a hierarchically-nested adversarial network. In *CVPR*, 2018.

[41] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems*, 2017.