MSG-GAN: Multi-Scale Gradients GAN for more stable and synchronized multi-scale image synthesis

Animesh Karnewar
Mobiliya
Pune, India
animeshsk3@gmail.com

Raghu Sesha Iyengar
Mobiliya
Bangalore, India
raghu.iyengar@mobiliya.com

Abstract

Generative Adversarial Network (GAN) which is widely used for Image synthesis via generative modelling suffers peculiarly from training instability. One of the known reasons for this instability is the passage of uninformative gradients from the Discriminator to the Generator due to learning imbalance between them during training. In this work, we propose Multi-Scale Gradients Generative Adversarial Network (MSG-GAN), a simplistic but effective technique for addressing this problem; by allowing the flow of gradients from the Discriminator to the Generator at multiple scales. This results in the Generator acquiring the ability to synthesize synchronized images at multiple resolutions simultaneously. We also highlight a suite of techniques that together buttress the stability of training without excessive hyperparameter tuning. Our MSG-GAN technique is a generic mathematical framework which has multiple instantiations. We present an intuitive form of this technique which uses the concatenation operation in the Discriminator computations and empirically validate it through experiments on the CelebA-HQ, CIFAR10 and Oxford102 flowers datasets and by comparing it with some of the current state-of-the-art techniques.

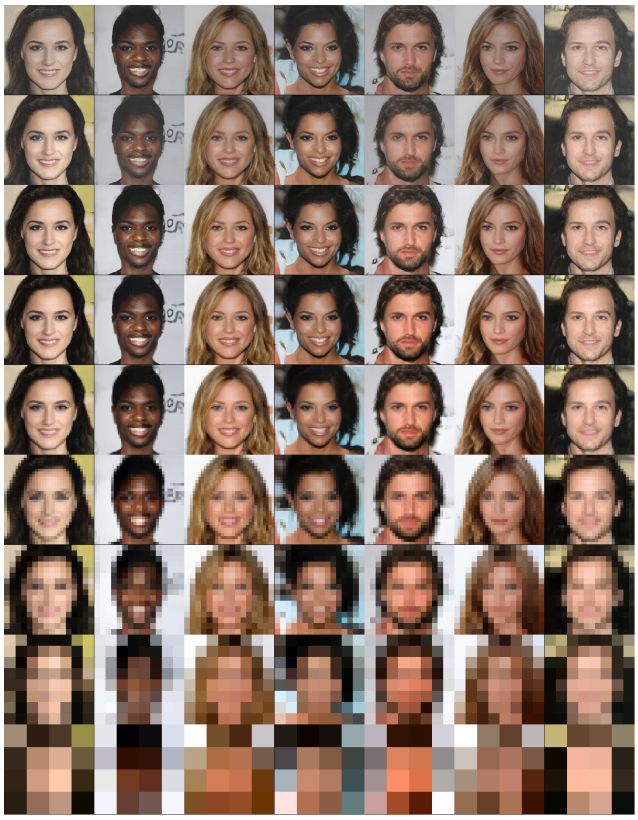

Figure 1: Images synthesized at various resolutions using our proposed MSG-GAN technique. The images are fully synchronized across the resolutions. The bottom row has a resolution 4×4 and doubles towards the top till 1024×1024.

1. Introduction and Related Work

There has been an enourmous amount of work done in the field of Generative Modelling since the advent of the Generative Adversarial Network (GAN) technique by Goodfellow *et al*. [8]. This type of Generative Modelling is copiously used for image synthesis due to the prevalence of continuous and differentiable settings where the GAN definition is very intuitive. Although GANs for coherent text synthesis have been attempted in the past such as [31, 10, 6], they have not been as successful as their image counterparts due to the discrete settings for text-synthesis. Because of this popularity and success of GANs in image-synthesis, the need for a formal objective evaluation metric for the quality of generated images arose; which resulted in the creation of the most used metrics *viz.* **IS** (Inception Score) [23] and **FID** (Fréchet Inception Distance) [11]. We use the IS for the CIFAR10 and Oxford102 flowers experiments since the previous works have used it for their evaluation while the

FID for the CelebA-HQ experiment.

The success of the GAN class of models comes from the technique of cleverly bypassing the requirement of manually created distance metrics for optimization. Although flow based models such as [3, 4, 22, 17] and autoregressive models such as [26, 25, 24] allow training generative models directly using Maximum Liklihood Estimation (explicitly and implicitly respectively), the fidelity of the images generated by them cannot match the current state of the art GAN models [14, 15]. However, the GAN suffers from two prominent problems of: (1) Mode Collapse and (2) Training Instability. The problem of mode collapse occurs when the Generator Network only captures a subset of the variance present in the data distribution. Numerous works [23, 14, 19] have previously addressed this problem.

The problem of training instability is a long standing one which roots in the very definition of a GAN. The view that the Generator minimizes the Jenson-Shannon divergence between the generated data distribution and the original data distribution as suggested by Goodfellow *et al*. [8] has been impugned by [12]. The main reason being that an optimal Discriminator cannot exist in practice. This has a rather paradoxical contradiction which in simplified language can be described as: 'Discriminator cannot become optimal unless it is trained against all possible Generators[1], while such a set of all possible Generators cannot exist unless there is an optimal Discriminator'. Thus, the Nash equilibrium for a GAN is not the same as the one for divergence minimization. This instability problem has been the cause for a myriad of previous works, some of which include [2, 9, 20, 27, 13, 30, 14, 21]. Our main contributions towards this line of research are:

1. We highlight some of the caveats of using the baseline Progressively Grown GAN [14] and how MSG-GAN addresses them (in section 1.1).

2. We propose our technique of Multi-Scale Gradients GAN (MSG-GAN) for synchronized multi-scale image synthesis which allows gradient flow from the Discriminator to the Generator at multiple scales (in section 2).

3. We empirically validate the approach and show that less hyperparameter tuning is required with MSG-GAN for various datasets through experiments on CIFAR10, CelebA-HQ and Oxford102 flowers datasets (in section 4).

1.1. Motivation

Arjovsky and Bottou [1] pointed out that one of the reasons for the training instabililty 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. There are two recent works which focus on alleviating this problem *viz.* 'Variational Discriminator Bottleneck' (VDB) [21] and the 'Progressive Growing of GANs' (ProGAN) [14]. The VDB's solution is to apply a Mutual Information Bottleneck between the input images to the Discriminator and it's deepest representation for them. This forces the discriminator to focus only on the most discerning features of the images for classification into real and fake images. Our work is orthogonal to the VDB technique and we point out that combination of MSG-GAN and VDB is possible and leave out for subsequent research.

The ProGAN technique tackles the aforementioned problem by training the GAN layer by layer progressively doubling the synthesized image resolution. At every point in training all the previously trained layers are still trainable and whenever a new layer is added to the training it is slowly faded in such that the learnings of the previous layers are not destroyed. Intuitively, this technique addresses the problem because, for a small resolution there would already be a significant overlap[2] in the supports of the distributions while at a higher resolution, the overlap would still be sufficient as the preceeding lower resolution is already trained. However, we encountered certain caveats in our analysis of the ProGAN described as follows:

- ProGAN proposes to use equal number of iterations for all the resolutions from the smallest one to the largest one. However, in our[3] experiments with the ProGAN, we found that a substantial amount of the final iterations are unnecessary for smaller resolutions whereas the same number of iterations are not enough for the higher resolutions. This is conveyed through the plots of figure 2 as well as our training video at https://www.youtube.com/watch?v=lzTm6Lq76Mo. Thus, there exists an optimal training schedule for the ProGAN which we found to be task specific and requiring empirical analysis as well as constant supervision for derivation.

- The model trained on lower resolution doesn't produce the same images after subsequent higher resolution layers are trained. Unfortunately, the parameters used for generating images at those resolutions during their training get wasted in the end. Please refer figure 3 for deeper insight into this highlight. This is not a disadvantage per se, but we just point it out as a caveat because it was obverse to our intuition.

[1] This infinite set includes the Generators ranging from worst performing to the best performing ones.

[2] For a resolution as small as 4×4, there is less scope for divergence, due to the presence of only 4^2 dimensions.

[3] We use our own PyTorch implementation of the ProGAN for this analysis which is made open source at https://github.com/akanimax/pro_gan_pytorch.

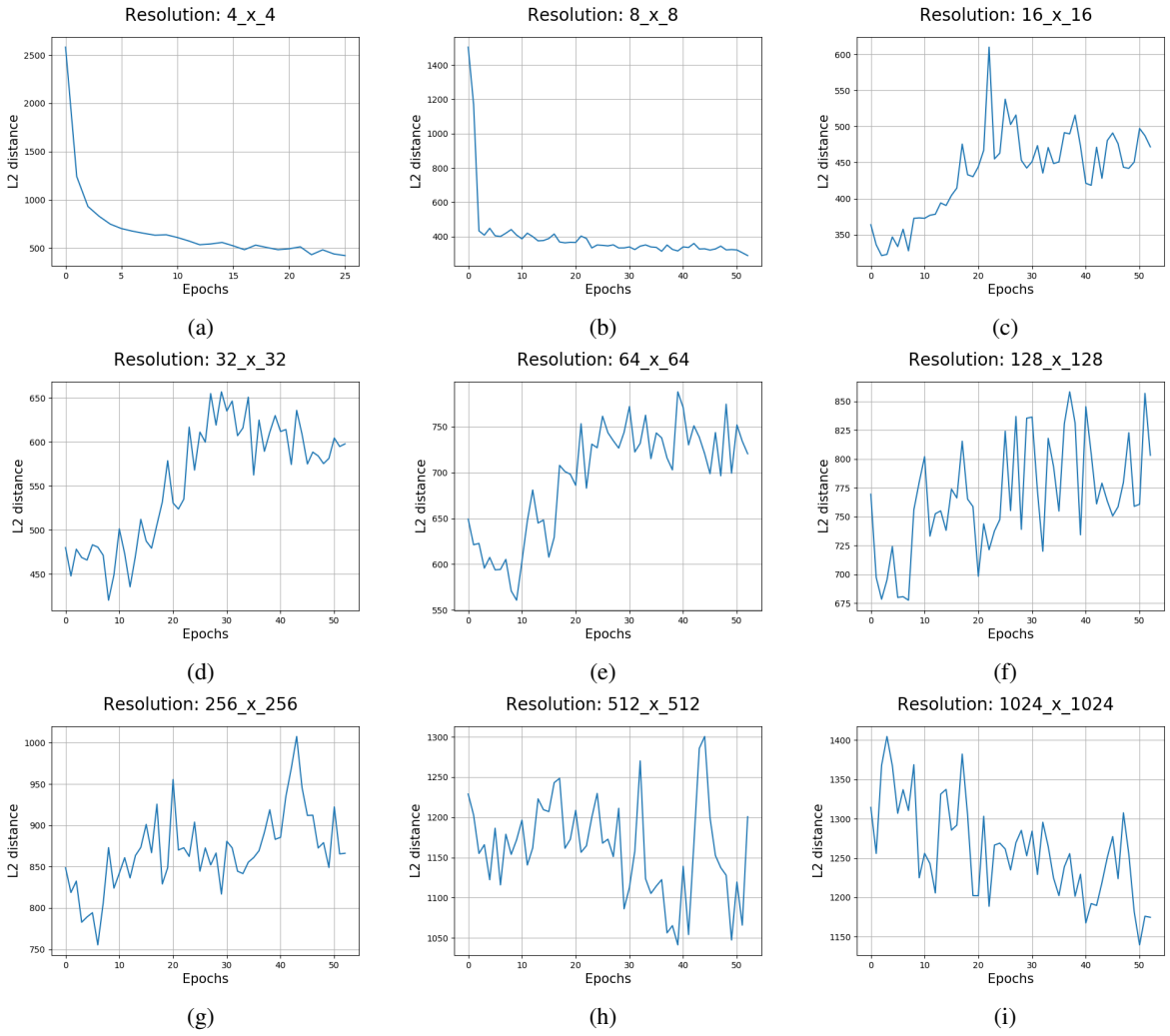

Figure 2: These plots are generated by calculating the L2 distance between two grids of 36 images sampled at the beginning of each epoch and the preceeding epoch with fixed latent points during the training of the progressively grown GAN at all stages (resolutions). Similar plots for the MSG-GAN training on CelebA-HQ are provided in the supplementary material.

- The technique of progressively growing GANs is especially difficult to implement from a resistance to training interruption or failure point of view. If the interruption happens during the stablization iterations, it is still manageable to incorporate resuming in the implementation; but it is very difficult to resume from an interruption during the fade-in iterations without affecting the performance.

These caveats motivated us to find alternative solutions for them which eventually lead to the formalization of the MSG-GAN. During our literature study [29, 33, 32, 5, 7, 14], we found a common trait that breaking the process of high resolution image synthesis into smaller subtasks yields better results. The LR-GAN [29] breaks this process down into high level semantic curriculum by using separate Generators for synthesizing the background, foreground and compositor masks for the final image. We use the simple multi-resolution breaking of the curriculum for image synthesis in our MSG-GAN technique. Our technique is different from StackGAN and GMAN [33, 32, 5] as they use separate multiple Discriminators for a single Generator and also different from MAD-GAN [7] because it uses Multiple Generators and a Single Discriminator. **Our Method MSG-GAN has a single Generator and single Discriminator** which has connections from the Generator to the Discriminator at intermediate layers allowing gradient flow at multiple scales of image synthesis.

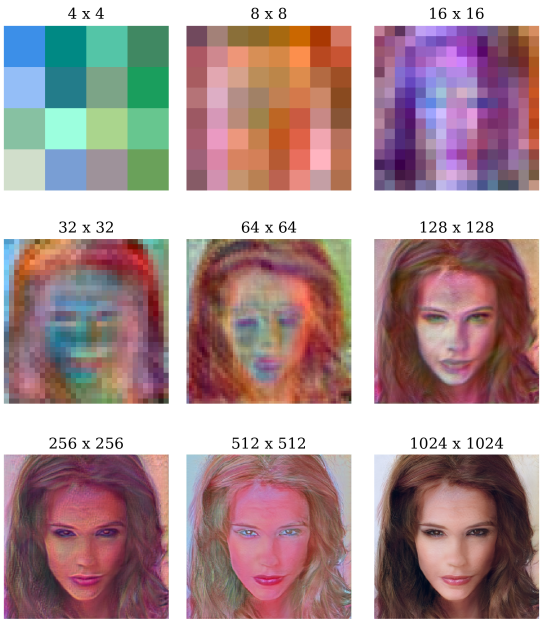

Figure 3: Images synthesized at various resolutions by a fully trained Progressively Grown GAN model. These images are counterintuitively not colour and brighness synchronized across the resolutions, although they preserve certain similarities in structure.

1.2. Fully Recursive View of Single Image Super Resolution

The problem of Single Image super resolution can be mathematically defined as: given a set of n pairs of low resolution $(h \times w)$ and corresponding high resolution $(h^+ \times w^+)$ images,

$$Dataset = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\} \quad (1)$$

learn a mapping from the low resolution images to the high resolution ones.

$g_{sr} : I_{low} \mapsto I_{high}$
where,

$$I_{low} = (x_i | x_i \in \mathbb{W}_{255}^{h \times w \times 3}) \quad (2)$$

$$I_{high} = (y_i | y_i \in \mathbb{W}_{255}^{h^+ \times w^+ \times 3}) \quad (3)$$

$$Dataset = zip(I_{low}, I_{high}) = \{(x_1, y_1), ..., (x_n, y_n)\}$$

$$i \in \{1, 2, ...n\}$$

The upscaling factor u defined as: $u = h^+/h = w^+/w$ could be any real value. Typically it is taken as a power of 2, for example `2x`, `4x`, `8x` upsampling etc. The function is modelled using a form of Convolutional Neural Network which was earlier optimized using the pixelwise Mean Squared Error. Recent works [18, 28] however use an adversarial GAN loss for optimization.

We motivate our MSG-GAN technique as a fully recursive view of the Single Image Super Resolution. Firstly, we restrict the super resolution function g_{sr2x} to always provide a `2x` upsampling. Next, we define the main Generator function g_{gen} as $g_{gen} : Z \mapsto I_{begin}$, such that $Z = \mathbb{R}^{\texttt{latent\_size}}$ & $Z \sim N(0, I)$ and I_{begin} contains only the `4 x 4` images. Thus, Our total Generator function GEN_l which synthesizes 2^l x 2^l images can be defined as a composition of $k = l - 2$ [4] such g_{sr2x} upsampling functions with the main Generator function g_{gen}.

$$GEN_l(z) = g_{sr2x}^k \circ g_{sr2x}^{k-1} \circ ... g_{sr2x}^1 \circ g_{gen}(z) \quad (4)$$

Thus, our MSG-GAN network is architecturally similar to the ProGANSR [28] in essence, but the overall technique is vastly different because, there are no optimal super resolution images available for our unconditional image generation and we train our entire MSG-GAN network instead of progressively growing the layers.

2. Multi-Scale Gradients GAN

Using a similar definition for the Generator function, as instituted in equation (4) in the prior section, we generalize the concept to a broader family of functions, not necessarily the `2x` super resolution function g_{sr2x}. Let the function g be a generic function which acts as the most basic constituent of the Generator computations for a GAN. The Generator of our GAN can be defined as:

$$GEN(z) = g^k \circ g^{k-1} \circ ... g^1 \circ g_{gen}(z) \quad (5)$$

where, the g_{gen} function still has the same definition as in section (1.2).

We now define the r function which is able to generate the output at different stages of the curriculum of the generator computations. In case of image synthesis, the most natural form of the curriculum is downsampled resolutions of the ultimate synthesis resolution. The r function can be very easily modelled using the `1 x 1` convolution which converts the intermediate convolutional activation volume into images.

$r^i : g^i(z) \mapsto O^i$ where, $O^i = \mathbb{R}^{h^i \times w^i \times 3}$
Thus, $r^i(g^i(z)) = o^i$

Basically, the o^i is an image synthesized from the output of the i^{th} intermediate layer in the Generator computations.

[4] $k = l - 2$ because the starting resolution is 4 x 4, i.e. 2^2 x 2^2

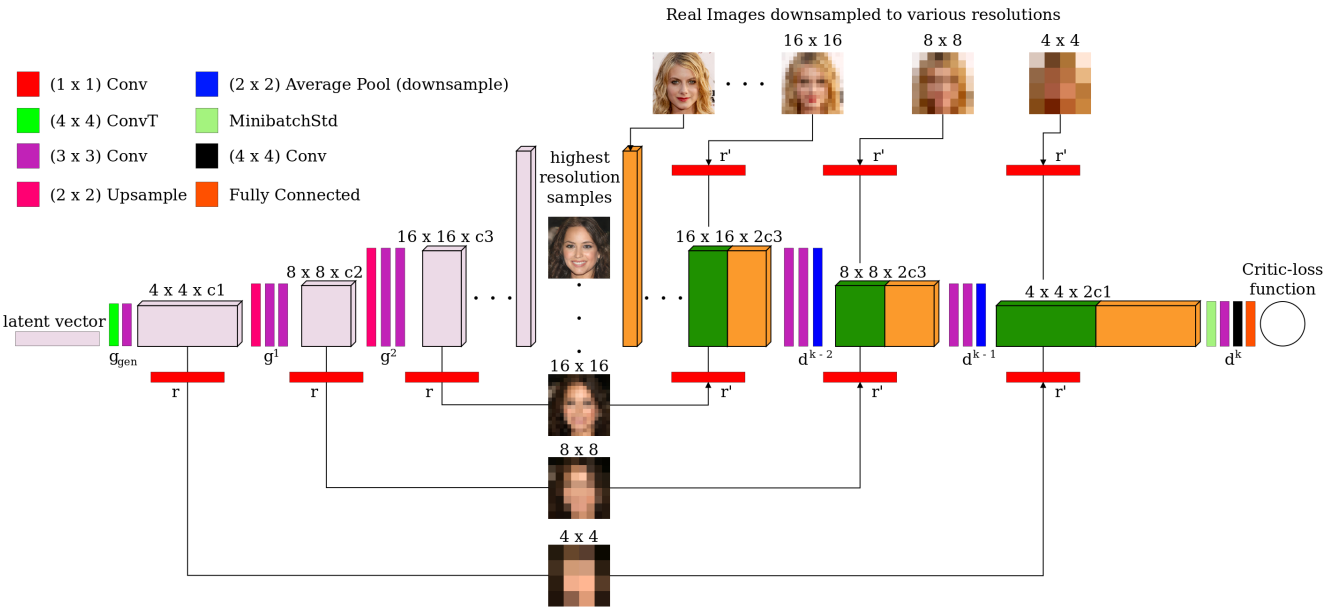

Figure 4: Architecture of the concatenation instantiation of MSG-GAN for generating synchronized multi-scale images.

The Discriminator definition encompasses not only the final output of the Generator y' (or o^k), but also the intermediate outputs of the generator o^i. Because the Discriminator's final critic loss is a function of the final output as well as intermediate outputs of the Generator, gradients 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. Let $d_{critic}(z')$ be the final layer of the Discriminator which provides the critic score. Let $d^0(y)$ or $d^0(y')$ be the function which defines the first layer of the Discriminator which takes the real image y or the highest resolution synthesized image y' as the input. Let d^i be the function which represents the intermediate layers of the Discriminator. Thus, the output activation volume o'^i of any i^{th} intermediate layer of the Discriminator can be defined as:

$$o'^i = layer^i(o^{k-i}, o'^{i-1}) \tag{6}$$

$$= d^i(combine(r'^i(o^{k-i}), o'^{i-1})) \tag{7}$$

Where, the r' function does the converse of the r function, i.e. it converts the intermediate input image to an activation volume. The *combine* function combines the output of r' with the corresponding intermediate activation volume in the Discriminator computations. The Complete Discriminator function can be defined as:

$$DIS(y) = d_{critic} \circ layer^k \circ layer^{k-1} ... \circ d^0(y) \tag{8}$$

Note that we have skipped the intermediate outputs of the generator which are also arguments of the *DIS* function for simplicity in the above equation. But it is disambiguated from the context of the $layer^i$ function.

The *combine* function is a very important component of the Discriminator computations. There are numerous ways in which it can be modelled using either parametric or non-parametric families of functions. We do highlight that further study of the subtleties of using different versions of the *combine* function will result in a different architecture. But, for this study, we simply provide a basic form of the combine function as follows:

$$combine(x_1, x_2) = [x_1; x_2] \tag{9}$$

Where, the $[;]$ denotes channelwise concatenation. Figure (4) gives a complete view of the MSG-GAN architecture which uses the concatenation version of the *combine* function. In the section (4) we go into details about the performance of the architecture as evaluted on various image-synthesis datasets.

3. Suite of techniques used for better stability

We also employ a suite of techniques in order to bolster the performance of the MSG-GAN model. The performance improvements are observed in terms of additional training stability, generated sample quality and sample variance. We have used all the techniques instituted in the ProGAN [14] apart from the progressive growing partially for apples-to-apples comparison and also because they improved the stability. Additionally, we also used the Exponential Moving Averaging of the generator weights [30] and

the Relativistic Average Hinge loss [13] which resulted in highly stable training. Following subsections describe the motivation for and empirical experience of using these techniques.

3.1. PixNorm in Generator

The magnitudes of the intermediate feature vectors obtained in the Generator during the training of the GAN have a tendancy of escalating to very high values. This results in a much harder optimization problem to solve than a normalized one. In case of MSG-GAN, since the intermediate feature volumes are transformed into RGB images via `1 x 1` convolutions, the Generator architecture highly benefits from a feature normalization methodology. We used the pixNorm technique as described in the proGAN [14] for restraining the feature vector magnitudes. This improved the training speed and allowed the network to start generating relevant sample images fairly early in the training.

3.2. Equalized Learning rate for all Parameters

We also employed the equalized learning rate technique from proGAN [14]. Though not extensively analysed through ablation study, we found that using it allowed us to use the same learning rate across all three datasets that we trained.

3.3. MinibatchStdDev in Discriminator

The MinibatchStdDev technique which is introduced in the proGAN [14] is very simple to implement and incorporate into the GAN architecture. As per this technique, the last layer of the Discriminator calculates the standard deviation of all the samples in the batch averaged across the spatial positions and concatenates it as a constant feature map for the input to the subsequent convolutional operations. We use it for improving the generated sample variance. This technique as opposed to other similar ones such as Batch Discrimination [23], doesn't use any additional parameters. Thus our final MSG-GAN model still has equal number of parameters as that of the proGAN [14].

3.4. Exponential Moving Averaging of Generator Parameters

For additional training stability, we use the Exponential Moving Averaging (EMA) of the Generator as described in [30]. We keep a second copy of the Generator Model which is initialized to have same values as that of the actual training Generator and updated to accumulate the EMA of the training Generator's weights. This drastically reduces the deviations in the generated samples and also leads to a better equilibrium during training.

3.5. Relativistic Average HingeGAN critic loss

Our Experiments on the CIFAR-10 with the WGAN-GP, LSGAN and HingeGAN loss functions [9, 20, 27] required extensive tuning of the different learning rates for the Discriminator and Generator (as per TTUR [11]) to obtain a stable training which didn't diverge. Using Relativistic versions [13] of these loss functions however almost always worked with a reasonably small learning rate (even without TTUR). We found the Relativistic Average HingeGAN loss to work most stably. Most importantly, using the same learning rate of **0.001** with the RMSprop optimization algorithm resulted in stable GAN training across all the different datasets. The loss functions for Generator and Discriminator are mathematically defined as:

$$rf = D(x_{real}) - \mathbb{E}[D(x_{fake})] \tag{10}$$
$$fr = D(x_{fake}) - \mathbb{E}[D(x_{real})] \tag{11}$$
$$l_{dis}(x_{real}, x_{fake}) = \mathbb{E}[R(1 - rf] + \mathbb{E}[R(1 + fr)] \tag{12}$$
$$l_{gen}(x_{real}, x_{fake}) = \mathbb{E}[R(1 + rf)] + \mathbb{E}[R(1 - fr)] \tag{13}$$
$$\text{where, } R(x) = max(x, 0) \tag{14}$$

Although this requires the calculation of the Discriminator critic function values over the real samples even for the Generator update pass, these additional computations are worth performing due to the benefits of additional stability and avoidance of learning rate tuning.

4. Experiments

We used the proposed MSG-GAN technique to train models to generate unconditional Image samples on the CIFAR10, CelebA-HQ and the Oxford102 flowers datasets. These datasets were selected specifically to show that the MSG-GAN technique can be used for varied image settings. The CIFAR10 dataset contains low resolution images (`32 x 32`) for 10 different classes of objects ranging from animals to automotive vehicles, whereas the CelebA-HQ dataset as procured by [14] contains very high resolution images (`1024 x 1024`) of people (celebs) cropped to show only the face region. For the Oxford102 flowers dataset, we resized the flower images to (`256 x 256`) resolution in order to have a mid-range experiment.

The three models used for these experiments have the same initial latent dimensionality of **512**. The latent distribution used is the standard normal distribution $N(0, \mathbb{I})$. The architecture of the model used for the CelebA-HQ experiment has the exact same structure as the model described in proGAN [14]. The other two models are ablated versions of the biggest model. The details about all the model architectures are provided in the supplementary material.

All the models were trained by using a learning rate of **0.001** for both Generator and Discriminator. We used the

(a) (b) (c)

Figure 5: Samples generated by MSG-GAN on different datasets. (a), (b) and (c) are uncurated highest resolution samples produced for the CIFAR10, Oxford102 flowers and CelebA-HQ datasets respectively.

Adam Optimizer [16] with the values of (**0**, **0.99**) for the β_1 and β_2 for the training. Note that we do not use any specialized initialization technique and initialize all the paramters of both Generator and Discriminator according to the standard normal $N(0, \mathbb{I})$ distribution. But, the parameters are scaled according to the equalized learning rate setting of proGAN [14] at runtime during training and inference. For the EMA technique [30] detailed in section (3.4), we use the standard decay value equal to **0.999**. All the code for reproducing our work is available on our GitHub repository at `https://github.com/akanimax/BMSG-GAN`.

As described in Table (1), our results are better than most of the previous works that have reported scores on these datasets. Our scores are slightly worse than proGAN [14] because, we have refrained from doing extensive hyperparameter search to obtain the best performing models which would surpass them. For CIFAR10, we trained a few different settings and obtained the Inception Score of **7.63**. We then used the same settings for the other two experiments which supports our claim of requiring less hyperparameter tuning. Our Inception score of **3.56** on the Oxford102 flowers dataset is better than StackGAN++ and StackGAN [32, 33] and thus becomes current state of the art. Also, our experiment for the flower generation is unconditional as opposed to the conditional generation of StackGAN. We report the score on CelebA-HQ dataset only for completeness, because our main purpose for this experiment was to empirically validate our technique for as high resolution as `1024 x 1024`. In our experiments, we did not fish for the best results by overfitting the metrics because, firstly for all our experiments, we were restricted to only two V100 GPUs of a DGX-1 machine and secondly, our main motivation for these experiments was to provide an empirical proof for our mathematical MSG-GAN technique.

Dataset	Method	Value	Metric
CIFAR10	proGAN [14]	8.80	Inception Score (higher better)
	MSG-GAN (ours)	**7.63**	
	LR-GAN [29]	7.17	
	ImprovedGAN [23]	6.86	
	GMAN [5]	6.00	
Oxford102 flowers	**MSG-GAN (ours)**	**3.56**	
	StackGAN++ [32]	3.26	
	StackGAN [33]	3.20	
CelebA-HQ	StyleGAN [15]	5.06	FID (lower better)
	proGAN [14]	7.30	
	MSG-GAN (ours)	**XXX**	

Table 1: Results of our experiments.

5. Discussion

The training of the MSG-GAN progresses in a highly symbiotic way, being very close to our human understanding. Figure (6) describes how the synchronization across the layers takes place ultimately resulting in a coherent set of images synthesized at multiple resolutions. All this is possible without the need of additional Discriminators or multiple Generators. Due to the ability to generate images at multiple scales, a very informative feedback regarding the training of large models starts to come very early. Since the upsampling layers use `(2 x 2)` upsampling followed by convolutional operations, the Generator first learns an identity function with the parametric convolutions and then builds upon that to provide more and more details for the particular resolutions. This identity function learning phenomenon gets achieved without the use of residual additive connections in the Generator layers. Also, since the con-

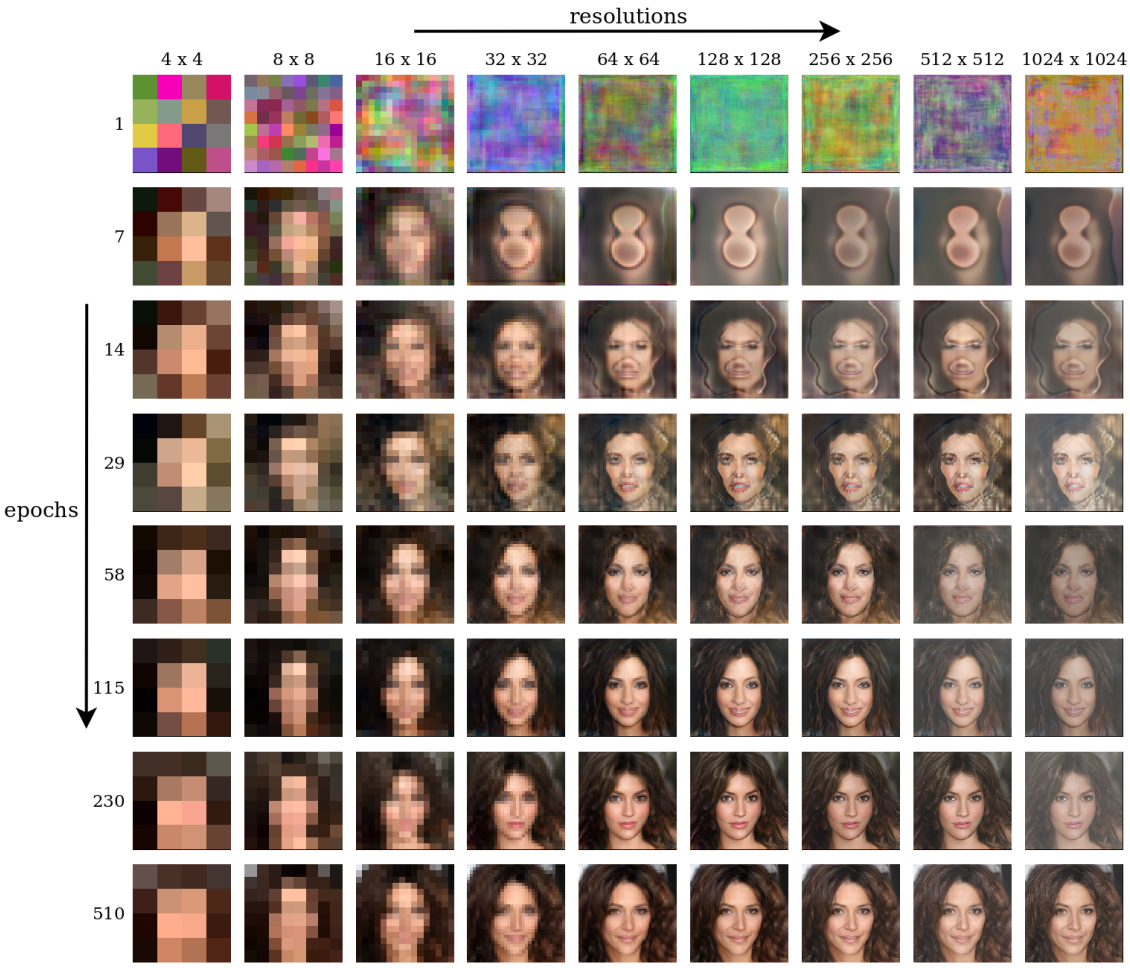

Figure 6: During training all the layers in the MSG-GAN first synchronize colourwise and subsequently improve the generated images at various scales. The brightness of the images across all layers (scales) synchronizes eventually.

nections from the Generator to the Discriminator at intermediate layers are maintained throughout the training, the output of the intermediate layers never changes to arbitrary structures (as in case of proGAN, figure 3). Finally, one of the added benefits of the MSG-GAN technique is that the images generated at higher resolution, maintain symmetry of certain features such as same colour for both eyes, or earrings in both ears using the normal versions of the convolution operation. This is still not possible with proGAN [14] because of the local nature of the convolution operation. However, in case of MSG-GAN, since the consistency is required between various resolutions of the synthesized images, the symmetry restriction gets applied automatically.

6. Conclusion

Although huge strides have been made for solving the problem of photo-realistic high resolution image synthesis, the true photo-realism which is indistinguishable from real images is yet to be achieved. We present our MSG-GAN technique which contributes to these strides with a simplistic technique also adding to the stability of GAN training. As mentioned in section (3), the concatenation model is a basic instantiation of the MSG-GAN technique and finding other variants is a promising research direction to explore. Another direction worth exploring is the augmentation of the MSG-GAN technique with some of the recent orthagonal works such as VDB [21], pacGAN [19], etc.

7. Acknowledgements

We would like to thank Alexia Jolicoeur-Martineau (Ph.D. student at MILA) for her guidance and expertise over Relativism in GANs and Dr. Oliver Wang (Research Scientist at Adobe) for his detailed advice to convert the project idea into a research. We also acknowledge Mobiliya for

sponsoring our work and providing 2 GPUS of the DGX-1 machine for experimentation.

References

[1] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. *CoRR*, abs/1701.04862, 2017. 2

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. 2

[3] L. Dinh, D. Krueger, and Y. Bengio. NICE: non-linear independent components estimation. *CoRR*, abs/1410.8516, 2014. 2

[4] L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real NVP. *CoRR*, abs/1605.08803, 2016. 2

[5] I. P. Durugkar, I. Gemp, and S. Mahadevan. Generative multi-adversarial networks. *CoRR*, abs/1611.01673, 2016. 3, 7

[6] W. Fedus, I. Goodfellow, and A. M. Dai. MaskGAN: Better text generation via filling in the _____. In *International Conference on Learning Representations*, 2018. 1

[7] A. Ghosh, V. Kulharia, V. P. Namboodiri, P. H. Torr, and P. K. Dokania. Multi-agent diverse generative adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 3

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. 1, 2

[9] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5767–5777. Curran Associates, Inc., 2017. 2, 6

[10] J. Guo, S. Lu, H. Cai, W. Zhang, Y. Yu, and J. Wang. Long text generation via adversarial training with leaked information. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5141–5148, 2018. 1

[11] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6626–6637. Curran Associates, Inc., 2017. 1, 6

[12] A. Jolicoeur-Martineau. Gans beyond divergence minimization. *CoRR*, abs/1809.02145, 2018. 2

[13] A. Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard GAN. *CoRR*, abs/1807.00734, 2018. 2, 6

[14] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. 2, 3, 5, 6, 7, 8

[15] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948, 2018. 2, 7

[16] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 7

[17] D. P. Kingma and P. Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 10215–10224. Curran Associates, Inc., 2018. 2

[18] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 4

[19] Z. Lin, A. Khetan, G. Fanti, and S. Oh. Pacgan: The power of two samples in generative adversarial networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 1498–1507. Curran Associates, Inc., 2018. 2, 8

[20] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, and Z. Wang. Multi-class generative adversarial networks with the L2 loss function. *CoRR*, abs/1611.04076, 2016. 2, 6

[21] X. B. Peng, A. Kanazawa, S. Toyer, P. Abbeel, and S. Levine. Variational discriminator bottleneck: Improving imitation learning, inverse RL, and GANs by constraining information flow. In *International Conference on Learning Representations*, 2019. 2, 8

[22] D. J. Rezende and S. Mohamed. Variational inference with normalizing flows. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 1530–1538. JMLR.org, 2015. 2

[23] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, and X. Chen. Improved techniques for training gans. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc., 2016. 1, 2, 6, 7

[24] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma. Pixelcnn++: A pixelcnn implementation with discretized logistic mixture likelihood and other modifications. In *ICLR*, 2017. 2

[25] A. van den Oord, N. Kalchbrenner, L. Espeholt, k. kavukcuoglu, O. Vinyals, and A. Graves. Conditional image generation with pixelcnn decoders. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett,

editors, *Advances in Neural Information Processing Systems 29*, pages 4790–4798. Curran Associates, Inc., 2016. 2

[26] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. *CoRR*, abs/1601.06759, 2016. 2

[27] R. Wang, A. Cully, H. J. Chang, and Y. Demiris. MAGAN: margin adaptation for generative adversarial networks. *CoRR*, abs/1704.03817, 2017. 2, 6

[28] Y. Wang, F. Perazzi, B. McWilliams, A. Sorkine-Hornung, O. Sorkine-Hornung, and C. Schroers. A fully progressive approach to single-image super-resolution. *CoRR*, abs/1804.02900, 2018. 4

[29] J. Yang, A. Kannan, D. Batra, and D. Parikh. LR-GAN: layered recursive generative adversarial networks for image generation. *CoRR*, abs/1703.01560, 2017. 3, 7

[30] Y. Yazıcı, C.-S. Foo, S. Winkler, K.-H. Yap, G. Piliouras, and V. Chandrasekhar. The unusual effectiveness of averaging in GAN training. In *International Conference on Learning Representations*, 2019. 2, 5, 6, 7

[31] L. Yu, W. Zhang, J. Wang, and Y. Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 2852–2858, 2017. 1

[32] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *CoRR*, abs/1710.10916, 2017. 3, 7

[33] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 3, 7