Qualitative Assessment of Generative Adversarial Network Architectures for Image Generation

1st Samed Dogan
Munich University of Applied Sciences
Munich, Germany
samed.dogan@hm.edu

***Abstract*—To this day, image generation is one of the most challenging tasks within the deep learning domain. Albeit many advances, GANs remain notoriously hard to tune. The following paper highlights the recent architectures with promising results and depicts the motivation behind these proposed algorithms. Performance, especially in the field of conditional image synthesis, is further tested in an extensive hyperparameter search. We come to the conclusion that conditional batch normalization in conjunction with projection offers an impressive performance boost, regardless of model size.**

I. Introduction

Image synthesis rose in popularity with the emergence of Generative Adversarial Networks [Goodfellow et al., 2014], framing generative modeling as a min-max game between two competing neural networks. In short, GAN's consist of a generator, producing a model distribution that closely resembles the given target distribution, and a discriminator, distinguishing real from fake data. [Radford et al., 2016] demonstrates the effectiveness of GAN's with deep convolutional neural networks, producing visually appealing samples on datasets with low variability and resolution. Albeit their promising results, the underlying dual optimization problem proves to be unstable [Salimans et al., 2016], rendering GAN training a highly dynamic and sensitive task with respect to the setup at hand [Brock et al., 2018]. This gave rise to a multitude of algorithms, each tackling a different subset of problems within the image generation domain.

[Arjovsky and Bottou, 2017] investigates the convergence properties of the value function being optimized, providing insights into the stability of GAN training from a theoretical standpoint. In particular, unbounded loss functions and sparse gradients of D with respect to G, are assumed to be the main cause of GAN instability. As a result, several publications focus on finding objective functions with superior convergence properties, such as Hinge-[Lim and Ye, 2017] Least Squares-[Mao et al., 2016] or Wassersteinloss [Arjovsky et al., 2017]. On the contrary, other lines of work enforce K-Lipschitz continuity by constraining D with normalization [Miyato et al., 2018] or gradient penalties [Gulrajani et al., 2017], [Mescheder et al., 2018], [Kodali et al., 2017].

[Mirza and Osindero, 2014] proposes control of the data generation process by conditioning the model on additional information. Subsequent work further extends on the idea, incorporating class labels by directly changing the objective function (with auxiliary classifiers), [Odena et al., 2017], concatenating the conditional vector to middle layers (hidden concats) [Reed et al., 2016], or as inner product between the condition and feature vector (projection) [Miyato and Koyama, 2018]. Inspired by recent work in style transfer [Dumoulin et al., 2016], [de Vries et al., 2017] introduces conditional batch normalization, modulating convolutional features thru linguistic embeddings. Using class embeddings instead, conditional batch normalization is seen throughout various state of the art GANs [Brock et al., 2018], [Miyato et al., 2018], [Miyato and Koyama, 2018], [Zhang et al., 2019], [Brock et al., 2018], making it a staple in modern generator architectures.

[Zhang et al., 2019] improves structural coherency in generated images by introducing self-attention blocks from [Wang et al., 2017], within both D and G.

II. Background

This section briefly outlines the theory and motivation behind the chosen algorithms and architectures.

A. Generative Adversarial Networks

Generative Adversarial Networks [Goodfellow et al., 2014] are a framework for estimating generative models by training two competing neural networks . The Generator $G(z)$ transforms an input noise vector z sampled from the distribution $p_z(z)$, commonly $\mathcal{N}(0, I)$ or $\mathcal{U}[-1, 1]$, to the input space. The Discriminator $D(x)$ distinguishes between samples x, drawn from the real $p_{data}(x)$ or fake distribution $p_{model}(x)$. The Generator is trained to fool the constantly improving Discriminator, effectively establishing a two-player minimax game with the objective function:

$$\begin{aligned} &\min_G \max_D \mathbb{E}_{x\sim p_{data}(\mathbf{x})}[\log D(\mathbf{x})] \\ &+ \mathbb{E}_{z\sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))]. \end{aligned} \tag{1}$$

Assuming an optimal Discriminator, minimizing 1 equals to minimizing the Jensen-Shannon divergence between p_{data} and p_{model}, leading to vanishing gradients as $\log(1 - D(G(\mathbf{z})))$ saturates [Arjovsky et al., 2017]. In practice, [Goodfellow et al., 2014] suggests training G to maximize $\log(D(G(\mathbf{z})))$ instead, providing much stronger gradients in early training stages.

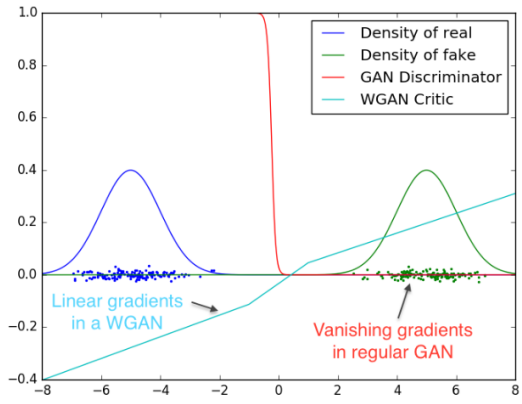

Fig. 1: Gradients of discriminator and critic with their respective loss function. The Discriminator saturates as p_{data} diverges from p_{model}. [Arjovsky et al., 2017]

B. Wasserstein GAN

According to [Arjovsky et al., 2017], proper optimization demands the objective loss function, precisely the divergence function between the estimated and real distribution, to be continuous with respect to the generators parameters. Any divergences potentially violating continuity, could lead to training difficulties. Under mild assumptions, the *Earth-Mover*, or *Wasserstein-1* distance $W(p, q)$, is continuous everywhere and differentiable almost everywhere, making it a more suitable cost function than the Jensen-Shannon divergence (see figure 1).
$W(p, q)$ is defined as the minimum cost of transporting mass in order to transform the distribution p into the distribution q. Using the Kantorovich-Rubinstein duality [Villani, 2008], the new objective function can be defined as

$$\min_G \max_D \mathbb{E}_{x\sim p_{data}}[D(\mathbf{x})] - \mathbb{E}_{\tilde{x}\sim p_{model}}[D(\tilde{\mathbf{x}})] \tag{2}$$

where p_{model} is the model distribution, implicitly defined by $G(z)$ with $z \sim p_z$. D outputs a credibility score, taking on the role of a *critic*. The Kantorovich-Rubinstein duality requires D to be a set of 1-Lipschitz functions, which can be achieved by clipping the weights to lie within a compact space $[-c, c]$. However, weight clipping can lead to exploding or vanishing gradients and biases the critic towards learning simpler functions, ignoring higher moments of the data distribution. Thus, [Gulrajani et al., 2017] proposes penalizing the gradient norm of the critic with respect to its input. The gradient penalty term is added to the original critic loss

$$\begin{aligned} L_{critic} = \mathbb{E}_{\tilde{x}\sim p_{model}}[D(\tilde{\mathbf{x}})] - \mathbb{E}_{x\sim p_{data}}[D(\mathbf{x})] \\ + \lambda \mathbb{E}_{\hat{x}\sim p_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2], \end{aligned} \tag{3}$$

with λ being the *penalty coefficient* and $p_{\hat{x}}$ sampling uniformly along straight lines between pairs of points sampled from p_{data} and p_{model}.

C. Spectral Normalization

Given a matrix A, the spectral norm of A is defined as

$$\sigma(A) \coloneqq \max_{h\neq 0} \frac{\|Ah\|_2}{\|h\|_2} = \max_{h\leq 0} \|Ah\|_2, \tag{4}$$

which equals to the largest singular value of A. For a linear layer $g(h) = Wh$, the Lipschitz norm of g is equivalent to the spectral norm of their respective weights as

$$\|g\|_{Lip} = sup_h \sigma(\nabla g(h)) = sup_h \sigma(W) = \sigma(W). \tag{5}$$

Spectral Normalization [Miyato et al., 2018] enforces the Lipschitz constraint by normalizing the weight matrix W of each layer with its spectral norm $\sigma(W)$

$$\bar{W}_{SN}(W) \coloneqq W/\sigma(W), \tag{6}$$

so that $\sigma(\bar{W_{SN}}(W) = 1$. The motivation behind this approach is to bound the Lipschitz-Norm of the discriminative function $\|f\|_{Lip}$ from above by 1, by ensuring $\|g_l\|_{Lip} = 1 \, \forall \, l \in [1, L+1]$ given

$$\|f\|_{Lip} \leq \prod_{l=1}^{L+1} \|g_l\|_{Lip} = \prod_{l=1}^{L+1} \sigma(W^l). \tag{7}$$

Note that the inequality only holds true if the activation functions a_l are *1-Lipschitz* continous i.e $\|a_l\|_{Lip} = 1$. [Zhang et al., 2019] further showed that spectral normalization in the generator may help in stabilizing training, which allows for fewer discriminator steps per generator steps or use of the Two Time-Scale Update Rule (TTUR) [Heusel et al., 2017].

D. Conditional GAN

Conditional GANs facilitate control over the output by providing both D and G with additional information, thus effectively conditioning the model on arbitrary data y i.e $D(x|y)$ and $G(z|y)$. The provided information can be of different modality and depends on the given task, such as conditional image generation [Mirza and Osindero, 2014], text-to-image synthesis [Reed et al., 2016] or image-to-image translation [Zhu et al., 2017]. Previous lines of work have incorporated conditional information into the discriminator by concatenating the embedded conditioning variable $\mathbf{y}$ to the input vector or hidden feature vector $\mathbf{x}$, as seen in figure 2. Pro-

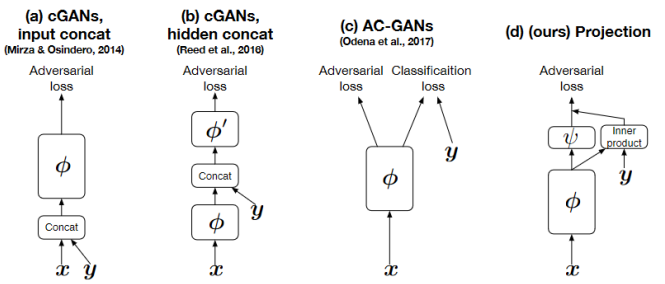

Fig. 2: Different conditional discriminator models [Miyato and Koyama, 2018].

jection [Miyato and Koyama, 2018] incorporates conditional information into the discriminator by measuring the similarity of the embedding with a set of features, providing additional evidence for distinguishing real and fake samples. Ultimately, Projection is adding the inner product of the globally pooled feature vectors and the embedded class labels to the scalar output of the discriminator.

One efficient way of conditioning G is through conditional batch normalization [de Vries et al., 2017], short *CBN*, which was slightly adjusted for GANs. Given an input batch x and conditioning vector y, *CBN* can be written as

$$CBN(x|y) = \gamma_c * \frac{x - \mathbb{E}[x]}{\sqrt{Var(x) + \epsilon}} + \beta_c \tag{8}$$

where ϵ is a small constant for numerical stability, and γ_c and β_c are learned class dependent vectors, capable of manipulating entire feature maps by scaling them up or down or negating them. These parameters can be obtained by embedding [Miyato et al., 2018], [Miyato and Koyama, 2018] or linear layers [Brock et al., 2018], however, *1-centered gains* $\tilde{\gamma_c} = \gamma_c + 1$ are advised for the latter.

E. Self Attention

As quality and resolution of synthetic images in multi-class datasets constantly improves, the difficulty in modeling global structure becomes apparent. Current state-of-the-art GANs [Miyato and Koyama, 2018] have high difficulty in modeling classes with consistently occurring structural and geometric patterns, such as distinct body parts. On the contrary, classes which are rather distinguished by texture than structure, provide deceivingly real samples, as depicted in figure 3. Considering the local receptive field of convolutions, several

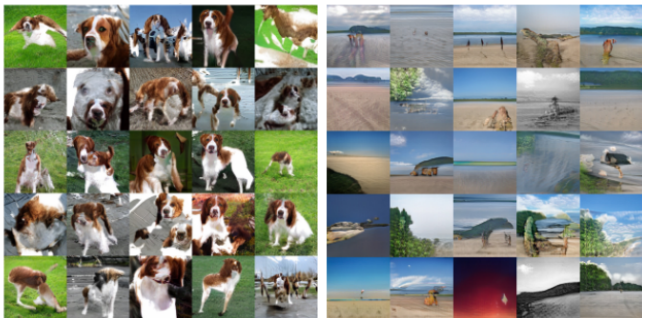

Fig. 3: Generated images for the welsh springer spaniel (left) and sandbar class (right). Note the structural inconsistency of the dogs body across several examples [Miyato et al., 2018]

convolutional layers are needed in order to process long range dependencies. This is in turn may explain the lack of global structure. [Zhang et al., 2019] adds self-attention blocks from [Wang et al., 2017] to both D and G, providing means in modeling long range, multi-level dependencies across image regions.

Attention is widely used in many domains such as natural language processing [Bahdanau et al., 2014], [Vaswani et al., 2017] or image captioning [Xu et al., 2015]. It is a vector of importance weights, depicting how strongly one input element like a pixel or word, attends to other elements. [Vaswani et al., 2017] describes the attention function as mapping a query and a set of key-value pairs to an output, where the output is a weighted sum of the input values. These weights are obtained by multiplying the query with the corresponding key and transforming the result to a probability distribution over all input positions, known as dot product attention. In self attention layers, the

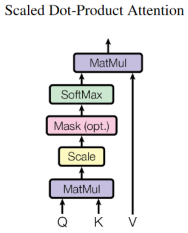

Fig. 4: Dot product attention with an additional scaling factor. The scaling can be used to lower the magnitude of the dot product, resulting in better gradients of the softmax function [Vaswani et al., 2017]

query, key and value pairs are not distinct and solely the output of the previous layer. As depicted in figure 5, the previous features from the hidden layer $x \in \mathbb{R}^{C \times N}$ with $N = H \times W$ are first transformed into three feature spaces f, g, h where $f(x) = W_f x$ and $g(x) = W_g x$ and $h(x) = W_h x$ with $W_{f,g,h} \in \mathbb{R}^{\bar{C} \times C}$ being the learned weights matrices and $\bar{C} = C/8$. The attention weights β are then calculated

$$\beta_{j,i} = softmax(f(x_i)^T g(x_j)) \tag{9}$$

and multiplied with the values $h(x)$ before transforming them to the output feature space

$$o_j = v\left(\sum_{i=1}^{N} \beta_{j,i} h(x_i)\right) with\, v(x) = W_v x, W_v \in \mathbb{R}^{C \times \bar{C}}. \tag{10}$$

Afterwards, the input feature map is added to the output of the scaled attention layer

$$y = \gamma o + x, \tag{11}$$

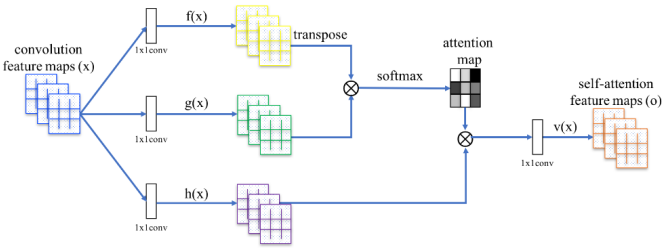

Fig. 5: Architecture of the self attention layer. Softmax is performed on each row and $\otimes$ denotes matrix multiplication [Zhang et al., 2019].

where γ is a learnable scalar initialized with zero. This enables the networks to first rely on local operations and gradually incorporate non-local information.

III. Results

A. Failure Cases

Common papers start by pointing out a problem within the GAN domain and propose their respective solution. Their claims are further backed up by extensive research and comparisons with equally performing algorithms. However, GAN training can be fairly unstable, where small changes can lead to non converging loss, mode collapse or bad generated samples. Thus, solely incorporating newly introduced architectures do not necessarily ensure good results. Figure 6 depicts one GAN failure case trained on the Cifar10 [Krizhevsky, 2009] dataset. The model was trained in

Fig. 6: Badly generated images of a GAN trained similarly to [Miyato et al., 2018]. The images depict common Cifar10 classes.

a similar fashion to [Miyato et al., 2018]. Surprisingly, the model did not convergence, whereas it does in Fashion-Mnist [Xiao et al., 2017] and regular Mnist [LeCun et al., 2010]. This raises the question weather training success is not only determined by the components used, but the careful tuning of the interacting components, see chapter III-B. Thus, as model complexity rises, metrics become all the more important in revealing failure cases at early training stages. Model performance is monitored by the commonly known Frechet-Inception Distance [Heusel et al., 2017] at runtime. Fake images are randomly sampled and compared to a constant set of images of the same class every 10 epochs. The FID-score is then calculated by averaging across all class scores.
The structural similarity index can provide a good measure in perceived image quality. Here, MS-SSIM [Wang et al., 2003] is used as a means to monitor the diversity within one generated image batch, indicating ongoing mode collapse, which can be seen in figure 7. The L2-Norm of both generator and discriminator gradients can provide valuable insights into training mechanics. Ideally, well behaved gradients are steady in nature. They neither explode, nor vanish. Models with spiking gradients, as seen in figure 7, are more prone to failure. We suspect that occasional batches of the generator strongly perturb the discriminator, which initiates a small episode of "mode collapse" from which the generator recovers. This however, slows down training. Further analysis in terms of gradient clipping of the discriminator is advised, but won't be treated in this paper.

B. Hyperparameter search

New algorithmic insights should be able to clearly leverage baseline model performance, without the need to precisely

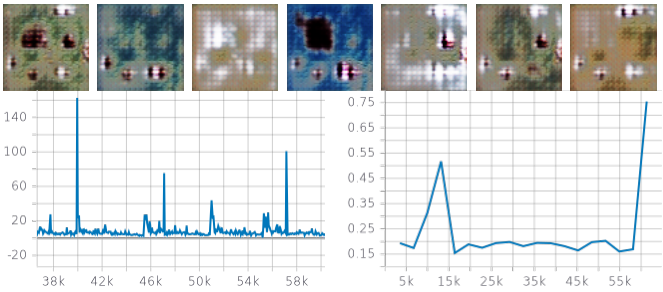

Fig. 7: Top: Mode collapse across several different image classes with depicted MS-SSIM score (bottom right). Spiking gradient norms (bottom left) could potentially lead to training instability.

copy every hyperparameter. Many breakthroughs in recent image synthesis tasks are build upon previous modules and not tested in an isolated environment, such that their benefit is not always discernible. In order to evaluate the impact and interaction between those modules, a hyperparameter search is performed. The motivation behind this approach is that any randomly chosen module that frequently appears within the hyperparameter list, should be of significance to model performance.
The baseline discriminator and generator consist of three fixed convolutional layers each. Downsampling in D is performed with strided convolutions and upsampling in G with transposed convolutions. The exact baseline model can be seen in table I.

$z \in \mathbb{R}^{50-300} \sim \mathcal{N}(0, I)$
Dense $\rightarrow 4 \times 4 \times (10 - 512)$
kernel=$(3-5)$, stride=2, Deconv. 512, lReLu α=$(0.0-0.3)$
kernel=$(3-5)$, stride=2, Deconv. 256, lReLu α=$(0.0-0.3)$
kernel=$(3-5)$, stride=2, Deconv. 128, lReLu α=$(0.0-0.3)$
kernel=3, stride=1, Conv. 3, Tanh

Table I: Baseline hypermodel of the generator. Brackets indicate variable parameters.

RGB image $x \in \mathbb{R}^{32\times32\times3}$
kernel=$(3-5)$, stride=2, Conv. 128, lReLu α=$(0.0-0.3)$
kernel=$(3-5)$, stride=2, Deconv. 256, lReLu α=$(0.0-0.3)$
kernel=$(3-5)$, stride=2, Deconv. 512, lReLu α=$(0.0-0.3)$
kernel=3, stride=1, Conv. 3, Tanh

Table II: Baseline hypermodel of the discriminator. Brackets indicate variable parameters. Final prediction layer is variable and depends on the chosen model type.

Every model iteration is conditioned on label y, though the precise conditioning mechanism is chosen from the optimizer. Every parameter dependent on a variable module is variable as well. For instance, the optimizer may choose projection, concat or hidden concat for the discriminator. If hidden concat is chosen, the layer at which the concat is performed, will be an hyperparameter, too. The generator is conditioned with either

concat/hidden concat + batch normalization or conditional batch normalization. For the latter, different implementations are provided as well. The model is trained with the hinge version [Lim and Ye, 2017] of the adversarial loss,

$$\begin{aligned}\mathcal{L}_D =& \mathbb{E}_{x,y\sim p_{data}}[max(0, 1 - D(x, y))] \\ &+ \mathbb{E}_{z\sim p_z, y\sim p_{data}}[max(0, 1 + D(G(z, y), y))] \\ \mathcal{L}_G =& - \mathbb{E}_{z\sim p_z, y\sim p_{data}}[D(G(z, y), y)]\end{aligned} \tag{12}$$

as it reported consistent results across several papers. Bayesian optimization was used as search algorithm and FID was chosen as objective to be minimized.
Figure 8 depicts the results after 5 iterations. The best configuration (blue) with the lowest fid score used conditional batch normalization in the generator and projection in the discriminator. We can not confirm the effectiveness of spectral normalization yet, as it was sporadically used across all iterations. We can also confirm that low embedding dimensions performed better than high dimensions. Figure 9 shows

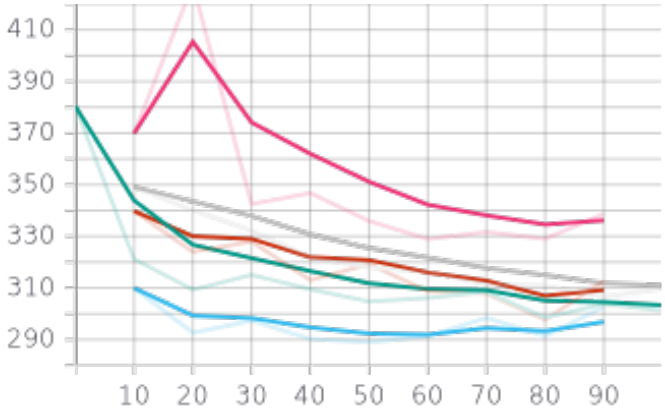

Fig. 8: Different model configurations trained on cifar10. Lowest fid score (blue) was achieved by conditional batch normalization with linear projection, consistend with [Brock et al., 2018] and projection [Miyato and Koyama, 2018]

generated examples of the best iteration.

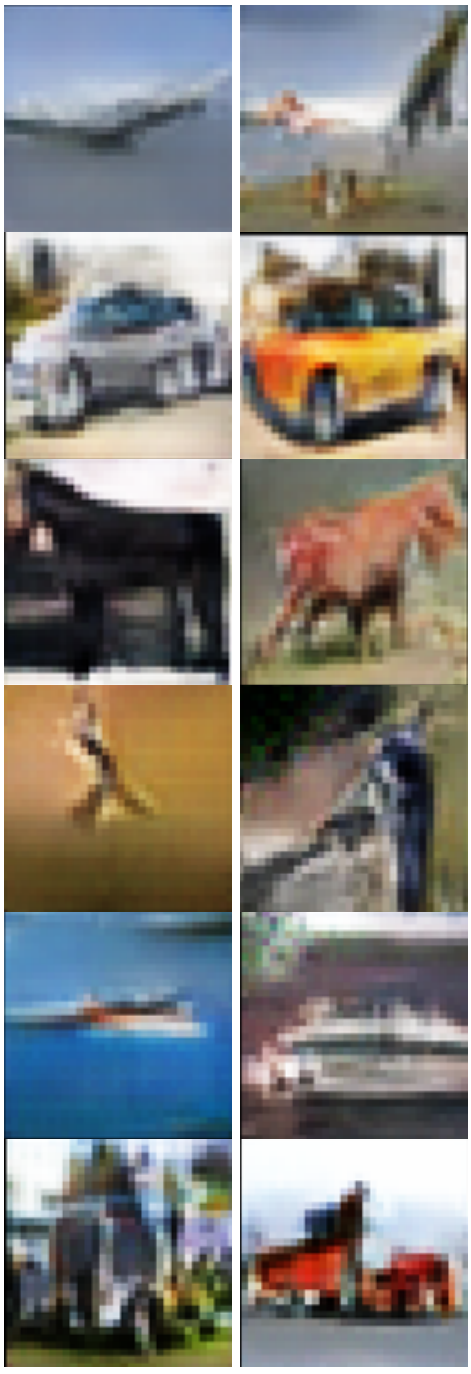

Fig. 9: Generated images with the labels plane, car, horse, bird, ship, truck.

REFERENCES

[Arjovsky and Bottou, 2017] Arjovsky, M. and Bottou, L. (2017). Towards principled methods for training generative adversarial networks.

[Arjovsky et al., 2017] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR.

[Bahdanau et al., 2014] Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

[Brock et al., 2018] Brock, A., Donahue, J., and Simonyan, K. (2018). Large scale GAN training for high fidelity natural image synthesis. *CoRR*, abs/1809.11096.

[de Vries et al., 2017] de Vries, H., Strub, F., Mary, J., Larochelle, H., Pietquin, O., and Courville, A. C. (2017). Modulating early visual processing by language. *CoRR*, abs/1707.00683.

[Dumoulin et al., 2016] Dumoulin, V., Shlens, J., and Kudlur, M. (2016). A learned representation for artistic style. *CoRR*, abs/1610.07629.

[Goodfellow et al., 2014] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks.

[Gulrajani et al., 2017] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). Improved training of wasserstein gans. *CoRR*, abs/1704.00028.

[Heusel et al., 2017] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Klambauer, G., and Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500.

[Kodali et al., 2017] Kodali, N., Abernethy, J., Hays, J., and Kira, Z. (2017). On Convergence and Stability of GANs. *arXiv e-prints*, page arXiv:1705.07215.

[Krizhevsky, 2009] Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. Technical report.

[LeCun et al., 2010] LeCun, Y., Cortes, C., and Burges, C. (2010). Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2.

[Lim and Ye, 2017] Lim, J. H. and Ye, J. C. (2017). Geometric gan.

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

[Mescheder et al., 2018] Mescheder, L., Geiger, A., and Nowozin, S. (2018). Which training methods for gans do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR.

[Mirza and Osindero, 2014] Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *CoRR*, abs/1411.1784.

[Miyato et al., 2018] Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. (2018). Spectral normalization for generative adversarial networks. *CoRR*, abs/1802.05957.

[Miyato and Koyama, 2018] Miyato, T. and Koyama, M. (2018). cgans with projection discriminator. *CoRR*, abs/1802.05637.

[Odena et al., 2017] Odena, A., Olah, C., and Shlens, J. (2017). Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning*, pages 2642–2651. PMLR.

[Radford et al., 2016] Radford, A., Metz, L., and Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks.

[Reed et al., 2016] Reed, S. E., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. (2016). Generative adversarial text to image synthesis. *CoRR*, abs/1605.05396.

[Salimans et al., 2016] Salimans, T., Goodfellow, I. J., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. *CoRR*, abs/1606.03498.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *CoRR*, abs/1706.03762.

[Villani, 2008] Villani, C. (2008). *Optimal Transport: Old and New*. Grundlehren der mathematischen Wissenschaften. Springer Berlin Heidelberg.

[Wang et al., 2017] Wang, X., Girshick, R. B., Gupta, A., and He, K. (2017). Non-local neural networks. *CoRR*, abs/1711.07971.

[Wang et al., 2003] Wang, Z., Simoncelli, E., and Bovik, A. (2003). Multiscale structural similarity for image quality assessment. volume 2, pages 1398 – 1402 Vol.2.

[Xiao et al., 2017] Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747.

[Xu et al., 2015] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A. C., Salakhutdinov, R., Zemel, R. S., and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. *CoRR*, abs/1502.03044.

[Zhang et al., 2019] Zhang, H., Goodfellow, I., Metaxas, D., and Odena, A. (2019). Self-attention generative adversarial networks. In *International conference on machine learning*, pages 7354–7363. PMLR.

[Zhu et al., 2017] Zhu, J., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593.