VirtualGAN: Reducing Mode Collapse in Generative Adversarial Networks Using Virtual Mapping

Adel Abusitta
School of Information Studies
McGill University
Montreal, Canada
adel.abusitta@mcgill.ca

Omar Abdel Wahab
Department of Computer Science and Engineering
Université du Québec en Outaouais
Gatineau, Canada
omar.abdulwahab@uqo.ca

Benjamin C. M. Fung
School of Information Studies
McGill University
Montreal, Canada
ben.fung@mcgill.ca

***Abstract*—This paper introduces a new framework for reducing mode collapse in Generative adversarial networks (GANs). The problem occurs when the generator learns to map several various input values (z) to the same output value, which makes the generator fail to capture all modes of the true data distribution. As a result, the diversity of synthetically produced data is lower than that of the real data. To address this problem, we propose a new and simple framework for training GANs based on the concept of virtual mapping. Our framework integrates two processes into GANs: *merge* and *split*. The *merge* process merges multiple data points (samples) into one before training the discriminator. In this way, the generator would be trained to capture the merged-data distribution rather than the (unmerged) data distribution. After the training, the *split* process is applied to the generator's output in order to split its contents and produce diverse modes. The proposed framework increases the chance of capturing diverse modes through enabling an indirect or virtual mapping between an input z value and multiple data points. This, in turn, enhances the chance of generating more diverse modes. Our results show the effectiveness of our framework compared to the existing approaches in terms of reducing the mode collapse problem.**

I. Introduction

Generative adversarial networks (or GANs) [Goodfellow *et al.*2014] is a powerful generative model used for learning any type of data distribution and producing synthetic data. GANs have shown an unprecedented ability to generate new synthetic high-quality data compared to the traditional generative models (e.g., VAE [Doersch2016], DBN [Hinton *et al.*2006]). The real-world applications of GANs can be seen in many domains, including those of computer vision [Wang *et al.*2018], cybersecurity [Arora and Shantanu2020], and mitigating biases in AI systems [Abusitta *et al.*2020].

Although interesting results (in terms of quality) have been achieved using GANs, training GANs for many applications is challenging since it frequently leads to the mode collapse issue. This occurs when the generator learns to map several noise z values to the same output point. As a result, the diversity of synthetically produced samples becomes smaller than that of the real data. For example, with the database of handwritten digits (MNIST), which consists of 10 modes e.g., {digit '1' ... digit '9'}, the generated samples may only produce very few of these modes [Khorramshahi *et al.*2020], as shown in Figure 1a.

In most GAN-related applications, we want our GAN to generate a wide variety of outputs [Zhang *et al.*2018]. As an instance, we need different images for every random input (z) to our face generator [Tripathy *et al.*2020] [Olszewski *et al.*2017]. Also, In GAN-related cybersecurity applications, we need our GAN to produce different synthetic malicious activities in order to enhance AI-powered Intrusion Detection Systems (IDSs) [Arora and Shantanu2020] [Abusitta *et al.*2019]. In real-world applications of GANs, if a generator generates an accepted output, the generator tends to learn how to generate only that output. In other words, the generator is looking to find the most acceptable one to the discriminator. This makes GANs to produce limited outputs.

Several attempts have been proposed to mitigate the mode-collapse issue (e.g., [Lin *et al.*2020], [Che *et al.*2016], [Metz *et al.*2016], [Dieng *et al.*2019], [Tolstikhin *et al.*2017]). However, most of these approaches suffer from mode collapse when applying them on high dimensional datasets that contain large number of modes. Moreover, some approaches require further modifications to the original GANs structure, which makes it not useful in practice.

To address the above-mentioned problems, we propose a novel framework for reducing mode collapse in GANs. The proposed framework is based on the concept of virtual mapping, which enables an input noise (z) to be virtually mapped to multiple data points. To this end, we integrate two processes into GANs: *merge* and *split*. The *merge* process combines two or more data points (samples) into one single point before training the discriminator. Thus, the generator is trained to produce merged data points rather than separate ones. Thereafter, the *split* process is used to split the generator's outputs and create separate points. The proposed framework, named VirtualGAN, can be seen as a trick which can be applied to the original GANs. For every mode produced from the VirtualGAN's generator, there is a chance that this mode could consist of a set of diverse modes after applying the *split*

(a) Random Samples generated using GAN

(b) Random Samples generated using VirtualGAN

Figure 1: Mode collapse problem in generated samples (MNIST dataset). In (a) generator only produces 0; 1; 4; 7; 9 modes; however, VirtualGAN (the proposed framework) increases the number of modes recovered (b).

process. The *merge* process increases the chance of capturing more modes by combining multiple points and letting the discriminator treat them as one point.

The proposed framework should not be confused with PacGAN [Lin *et al.*2020], which penalizes the generator with mode collapse by letting the discriminator make decisions based on multiple samples. Unlike PacGAN, VirtualGAN merges samples as one unit (merged data) and trains the generator to produce the merged data. The contributions of this paper are as follows:

- Proposing a simple and efficient framework for reducing mode collapse in GANs. The proposed framework can be easily applied and integrated with any version of GANs, with any kind of loss functions and parameters
- Devising a new concept, named virtual mapping in GANs. This concept gives flexibility for training GANs and mitigating mode collapse. Moreover, it provides a new direction for further improvement of GANs and their various versions.
- Evaluating the effectiveness of the proposed framework using several datasets, and comparing our results with those produced with other GAN architectures.

The rest of this paper is organized as follows: Section 2 formulates the proposed framework for mode collapse mitigation in GANs (VirtualGAN). In Section 3, we present our empirical results to show the effectiveness of the proposed framework. Section 4 discuss the related work. Finally, Section 5 concludes the paper and presents the future work.

II. Related Work

Several approaches have been proposed to alleviate mode collapse in GANs, e.g., [Lin *et al.*2020], [Che *et al.*2016], [Metz *et al.*2016], [Dieng *et al.*2019], [Tolstikhin *et al.*2017], [Balaji *et al.*2019] [Hoang *et al.*2018], [Srivastava *et al.*2017]. Martin et al. [Arjovsky *et al.*2017] propose to use Wasserstein loss to mitigate mode collapse. The Wasserstein loss enables the discriminator to learn how to reject these outputs that the generator stabilizes on. This strategy motivates the generator to produce new outputs. Also, Metz et al. [Metz *et al.*2016] propose unrolled GANs, which adopts a generator's loss-function that incorporates the future outputs of the discriminator in addition to the current ones. As a result, the generator becomes unable to over-optimize for a specific and single discriminator.

Recently, Lin et al. [Lin *et al.*2020] propose to use the augmentation of discriminator as a new approach for mitigating mode collapse in GANs. In this approach, the discriminator is adjusted to make decisions based on multiple samples from the same class (real or artificial data). The augmentation strategy mitigates mode collapse by penalizing generators that produce mode collapse [Lin *et al.*2020].

Tolstikhin et al. [Tolstikhin *et al.*2017] propose to train multiple generators rather than a single one. This approach is inspired by boosting algorithms [Welling *et al.*2003], which enable individual predictors to cooperate in order to produce a strong composite.

Another approach used to mitigate mode collapse is the regularization of encoder approach (e.g., [Dumoulin *et al.*2016], [Srivastava *et al.*2017]). For example, Srivastava et al. [Srivastava *et al.*2017] present VEEGAN, which is supported by a reconstructor network (RN) that reverses the generator's action through mapping from data to the latent (noise) space. In particular, they try to train the RN and the generator jointly. As a result, the generator becomes motivated to produce the entirety of the real data distribution.

Recently, Khorramshahi et al. [Khorramshahi *et al.*2020] propose to use GANs with variational entropy regularizers to alleviate mode collapse. Specifically, they propose to maximize a variational lower bound on the generated samples' entropy in order to increase the generator diversity. Similarly, Dieng et al. [Dieng *et al.*2019] propose to maximize the entropy of the generative distribution. This strategy motivates GANs to capture many modes of the data distributions [Dieng *et al.*2019].

III. The Proposed Framework

In this section, we present the details of our framework proposed for reducing mode collapse in GANs. We first give some background on GANs and then present the proposed framework.

A. Background

GANs is a new generative model, which has been proposed by [Goodfellow *et al.*2014]. GANs have received more attention

in the recent years thanks to their unprecedented ability to generate new synthetic high-quality data compared to the traditional generative models. It consists of two models: a discriminator (D) and a generator (G). The generator is trained to obtain the data distribution through attempting to maximize the probability of the discriminator committing a mistake. The discriminator is trained to increase the probability that a data sample came from a true data (training data) rather than the generator. In practice, the training of D and G is repeated over many iterations until the D model becomes unable to distinguish whether the underlying data point is a sample from the original data point or generated from G. This approach is also known as a minimax two-player game [O'Neill1987] [Goodfellow *et al.*2014] and is described formally as follows [Goodfellow *et al.*2014]:

$$min_G \ max_D V(G,D) = \mathbf{E}_{x\sim p_{data}} log[D(x)] + \mathbf{E}_{z\sim p_z} log[1-D(G(z))] \quad (1)$$

B. VirtualGAN

We address the problem of mode collapse by enabling virtual mapping between the noise space and data space. As can be seen in Figure 2, several data points are sampled in parallel from the data generated distribution. These data points are then merged together as a single data point in order to be used by a discriminator. The *merge* process means concatenating (or combining) data points together either vertically or horizontally. Thus, the generator is trained to produce combined data points rather than separated ones. Post training, there will be chances that each data point produced by the generator would consist of diverse modes rather than single or limited ones. These chances further increase if the *merge* process combines data points from different modes. Then, the *split* process is used to split the generator's output and produce diverse modes.

Equation 2 shows how the minmax two-player game is formulated based on the concept of *merge* process and virtual mapping. Note that the sampling process is done on the merged data ($p_{merged-data}$) rather than separate data (p_{data}) as depicted in Equation 1. Equation 2 represents the theoretical description of the proposed framework. In the implementation (Algorithm 1), we first sample data and then merge these data by taking into consideration the merge level l. For example, if l = 3, this means that we merge three data points together. Therefore, the generator is trained to produce the merged data.

$$min_G \ max_D V(G,D) = \mathbf{E}_{m\sim p_{merged-data}} log[D(m)] + \mathbf{E}_{z\sim p_z} log[1-D(G(z))] \quad (2)$$

$$s1 = G(z)$$
$$s2 = split(s1, l) \quad (3)$$

After training, the generator learns how to map input z values to the output value, which consists of merged or combined data points. The *split* process is then applied on the output of the generator to generate separate data points as shown in Equation (3). The parameter l in Equation (3) represents the merge level used when we merged data at the beginning. For example, if l = 3, $split(s1, l)$ will split $s1$ into 3 data points.

Proposition 1. p_g^m (generator's distribution over merged data m) converges to the merged-data distribution $p_{merged-data}$ if the convergence conditions (presented in [Goodfellow *et al.*2014]) have been achieved with respect to the merged data.

Proof. This is because the prove of convergence with respect to the (unmerged) data (p_g converges to p_{data}), which is presented in [Goodfellow *et al.*2014], works with any type of data used (merged or not). Therefore, if the data used are replaced with the merged data, then p_g^m converges to $p_{merged-data}$.

The proposed framework can be used with any GAN architecture, any loss function and parameters. Algorithm 1 describes the algorithm of the proposed framework integrated with WGAN [Arjovsky *et al.*2017], we call this "VirtualWGAN". As can be seen in Algorithm 1, each sampled data is combined with a set of data of size l. The generator is trained to capture the merged data distribution through trying to maximize the probability of the discriminator committing a mistake. The discriminator is trained to increase the probability that a merged data point sample came from a true merged data distribution. The loop with j in Algorithm 1 is used to create the merged data. Each data point (sampled from the training data) is merged with other data points, which are randomly selected. The loop with l in Algorithm 1 is used to ensure that l data points are merged together. It is worth mentioning here that the merging procedure can be implemented in many ways. The merging procedure can also be optimized to ensure that the merged data consist of different modes.

The merge process in Algorithm 1 adds additional complexity to the WGAN [Arjovsky *et al.*2017]. The additional complexity is equal to $l * n$, where l is the merge level and n is the number of sampled data.

How to merge and split data. The merging of the data can be done either vertically or horizontally. Knowing the dimensions of data points, the *split* process can easily split the merged points.

Why VirtualGAN can mitigate mode collapse in GANs? When applying the *merge* process in GANs, the generator is trained to generate merged data points rather than separated ones. This, in turn, allows the generator to produce several modes from a single random value z. The *split* process is then used to extract these modes.

C. GANs and the Concept of Virtual Mapping

We should note that the proposed framework does not lead to building a generator that is directly able to produce data that belong to the true data distribution. Instead, it builds a generator that produces data that belong to the true merged data distribution. Although this idea is not popular (perhaps not available) and it is not directly inline with the basic concept of generative models and GANs, we believe that this new approach is not considered as an issue. This is because the

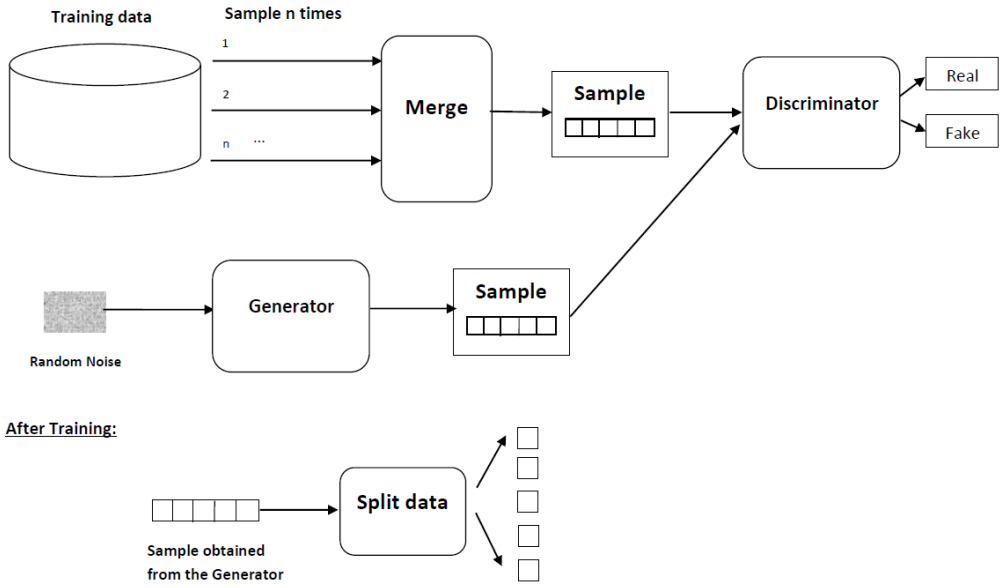

Figure 2: VirtualGAN: The proposed framework.

Algorithm 1: VirtualWGAN

Input: merge level l, learning rate $alpha$, clip value c
Input: number of iterations (per generator) $n-critic$
Input: size of batch n, parameters of critic (initial) w_0
Input: parameters of generator (initial) θ_0
repeat
  for *t=0 to n-critic* **do**
    Sample n data samples $\{x^{(i)},...,x^{(n)}\} from P_{data}$
    for j = *0 to n* **do**
      for l *steps* **do**
        $x^{(j)} \leftarrow concatenate(x^{(j)}, x^{(random(x^{(1)},x^{(n)}))})$
      end
    end
    Sample n noise samples $\{z^{(i)},...z^{(n)}\} from\, p(z)$
    $g_w \leftarrow \nabla_w[\frac{1}{n}\sum_{i=1}^{n} f_w(x^{(i)} - \frac{1}{n}\sum_{i=1}^{n} f_w(g_\theta(z^{(i)}))]$
    $w \leftarrow w + alpha.RMSProp(w, g_w)$
    $w \leftarrow clip(w, -c, c)$
  end
  $g_w \leftarrow -\nabla_w \frac{1}{n}\sum_{i=1}^{n} f_w(g_\theta(z^{(i)}))$
  $\theta \leftarrow \theta + alpha.RMSProp(\theta, g_\theta)$
until θ *has converged*;
function GENERATOR(z)
  $s1 \leftarrow g(z)$
  return $split(s1, l)$
end function

generator is basically considered as a tool for generating new synthetic data [Goodfellow *et al.*2016]. Our generator can do that when applying the *split* process to each generator's output. Our generator also increases the chance of producing diverse modes compared to other approaches. Moreover, it can be integrated with any GAN architecture, including those that are used to mitigate mode collapse and thus it can largely mitigate mode collapse.

IV. EXPERIMENTS

In this section, we compare the proposed framework with a set of generative models. These models are GAN [Goodfellow *et al.*2014], Deep Convolutional GAN (DCGAN) [Radford *et al.*2015], and WGAN [Arjovsky *et al.*2017]. The aim of the experiments is to show how the idea of virtual mapping can work any GAN architecture and improve their performance. Table I shows the parameters used for the experiments.

Table I: Experimentation parameters.

parameter	considered values
Number of epochs	200
Size of the batches	64
Learning rate	0.00005
Clip value	0.01
n-critic	5

The datasets used in the experiments are MNIST [LeCun *et al.*1998], 2D-grid [Srivastava *et al.*2017], 2D-ring [Srivastava *et al.*2017] and Stacked MNIST dataset [Lin *et al.*2020].

For the evaluation, we measure the number of modes recovered and the quality of generated samples. To measure the quality of samples, we used the state-of-the-art methods [Lala *et al.*2018] [Lin *et al.*2020] [Khorramshahi *et al.*2020]. For each dataset, we used the most suitable method to it as the following [Lala *et al.*2018] [Lin *et al.*2020] [Khorramshahi *et al.*2020].

2D-grid and 2D-ring datasets. In these two datasets, we measured the quality of generated samples by computing the

number of high quality samples, which is the proportion of the samples that are within 3 standard deviation to their nearest mode [Lin *et al.*2020].

MNIST dataset. For MNIST dataset, the proportion of high quality samples were calculated based on a predefined classifier. High quality samples are those samples that the classifier is highly confident on i.e., 99.8% confidence score [Lala *et al.*2018].

Stacked MNIST. In this dataset, the quality of generated sampled were calculated based on Kullback-Leibler (KL) divergence, which measures the difference between the generated distribution and the true data distribution. the lower KL value, the higher overall samples quality.

A. Results

Below, we compare GAN with VirtualGAN to show the advantage of using virtual mapping in GAN. As can be seen in Tables II, III and IV, our framework (VirtualGAN) allowed us to recover the whole modes when applying it on MNIST, 2D-grid and 2D-ring datasets. Specifically, the proposed framework recovered 10 modes from the MNIST, 25 modes from the 2D-grid, and 8 modes from the 2D-ring dataset. The results also show that GAN was not able to recover the whole modes when applying it in the same datesets. The superiority of the proposed framework compared to GAN indicates that the *merge* process used by our framework enables us to capture more modes compared to GAN.

Our results in Tables II, III and IV also show the proportion of high quality samples generated by each model. The proposed framework produced a higher proportion of high quality samples compared to GAN. The proportion value increased when the merge level has been increased from 2 (VirtualGAN-2) to 4 (VirtualGAN-4). These results can be interpreted as the impact of discriminator's input size used by each model [Curtó *et al.*2017]. The larger input size is, the larger number hyperparameters (number of layers and hidden units) are required to train. This makes the large-sized discriminator's input size to produce better performance [Curtó *et al.*2017].

Table II: The GAN and VirtualGAN results on MNIST dataset (max 10 modes). The results are averaged over 10 trials.

Model	No. of modes	% High quality samples
GAN	5	12.3
VirtualGAN-2 (ours)	**10**	19.2
VirtualGAN-4 (ours)	**10**	**23.4**

Table III: The GAN and VirtualGAN results on 2D-grid dataset (max 25 modes). The results are averaged over 10 trials.

Model	No. of modes	% High quality samples
GAN	17.2	93.6
VirtualGAN-2 (ours)	**25**	95.2
VirtualGAN-4 (ours)	**25**	**96.7**

Table IV: The GAN and VirtualGAN results on 2D-ring dataset (max 8 modes). The results are averaged over 10 trials.

Model	No. of modes	% High quality samples
GAN	6.4	96.7
VirtualGAN-2 (ours)	**8**	97.6
VirtualGAN-4 (ours)	**8**	**98.4**

Table V shows the results of comparing DCGAN with VirtualDCGAN (our model) on stacked MNIST dataset. VirtualDCGAN represents DCGAN integrated with the proposed framework (virtual mapping). As shown in Table 4, our model captured a higher number of modes compared to DCGAN. These results indicate that the proposed framework is able to produce outputs of more diverse modes compared DCGAN. The results also show that more modes can be captured when using a higher merge level as shown with VirtualDCGAN-4, which used a merge level = 4.

Table V also shows the value of KL divergence for each model. Our model reported a lower value of KL divergence compared to DCGAN. This suggests that the virtual mapping allow us to capture more modes of higher quality compared to DCGAN.

Table V: The DCGAN and VirtualDCGAN results on the Stacked MNIST dataset (max 1000 modes). The results are averaged over 10 trials.

Model	No. of modes	KL
DCGAN	79.2	3.2
VirtualDCGAN-2 (ours)	550.0	1.06
VirtualDCGAN-4 (ours)	**720.3**	**1.04**

B. WGAN Experiment

To verify that the virtual mapping idea can also work on Wasserstein loss, we compare WGAN with VirtualWGAN (Algorithm 1) on stacked MNIST dataset. Table 5 shows that VirtualWGAN captured a higher number of modes compared to WGAN. These results suggest that the concept of virtual mapping allowed us to capture more diverse modes compared to WGAN. The results also show that more number of modes have been captured when using a higher merge level. Table VI also shows that our model yields a lower value of KL divergence compared to WGAN. This means that VirtualWGAN generated more higher quality samples than WGAN.

Table VI: The WGAN and VirtualWGAN results on the Stacked MNIST dataset (max 1000 modes). The results are averaged over 10 trials.

Model	No. of Modes	KL
WGAN	312.4	2.63
VirtualWGAN-2 (ours)	513.7	1.56
VirtualWGAN-4 (ours)	**820.3**	**1.33**

C. Discussion

We should note that VirtualGAN could take longer time to train compared to the traditional GANs. The additional time comes from the size (dimension) of inputs and outputs used by the discriminator and the generator, respectively. These extra dimensions increase the size of the network and the number

of parameters while training. While this can be considered as the main disadvantage (limitation) of our new schema, we do not consider it an issue since the extra time leads at the end to reduce mode collapse in GANs. Moreover, our results show that the large size of inputs and output lead to improve the quality of generated samples.

V. Conclusion and Future Work

This paper presents a new framework for mitigating mode collapse in GANs. The proposed framework maps (virtually) each latent value z to several different outputs (data). This has been achieved by adding two processes in GANs: *merge* and *split*. The *merge* process combines data points into a single one in order to increase the chance of capturing more modes. Therefore, the discriminator and generator are trained based on combined data points. After training, the generator becomes able to produce combined points rather than separated ones. The *split* process splits the generator's output to produce diverse modes. Our framework can be integrated with any architecture of GANs, with any setup and any loss function. Experimental results show the effectiveness of the proposed framework in generating more modes compared to the existing approaches.

In the future, we would like to integrate and test the proposed framework with many GANs architectures. This is useful to understand to which extent the concept of virtual mapping could enhance the existing and different GAN architectures. Also, we plan to derive an efficient algorithm for training VirtualGAN. In particular, we will investigate the use of lossless data compression with the *merge* process in order to reduce the size of inputs and outputs of the discriminator and generator, respectively. We suggest that the concept of virtual mapping introduced in this paper could prove useful and we encourage researchers to work in this new direction.

References

[Abusitta *et al.*2019] Adel Abusitta, Martine Bellaiche, Michel Dagenais, and Talal Halabi. A deep learning approach for proactive multi-cloud cooperative intrusion detection system. *Future Generation Computer Systems*, 98:308–318, 2019.

[Abusitta *et al.*2020] Adel Abusitta, Esma Aimeur, and Omar Abdel Wahab. Generative adversarial networks for mitigating biases in machine learning systems. In *European Conference on Artificial Intelligence*, pages 937–944, 2020.

[Arjovsky *et al.*2017] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[Arora and Shantanu2020] Aayush Arora and Shantanu. A review on application of gans in cybersecurity domain. *IETE Technical Review*, pages 1–9, 2020.

[Balaji *et al.*2019] Yogesh Balaji, Rama Chellappa, and Soheil Feizi. Normalized wasserstein for mixture distributions with applications in adversarial learning and domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6500–6508, 2019.

[Che *et al.*2016] Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*, 2016.

[Curtó *et al.*2017] Joachim D Curtó, HC Zarza, and T Kim. High-resolution deep convolutional generative adversarial networks. *arXiv preprint arXiv:1711.06491*, 2017.

[Dieng *et al.*2019] Adji B Dieng, Francisco JR Ruiz, David M Blei, and Michalis K Titsias. Prescribed generative adversarial networks. *arXiv preprint arXiv:1910.04302*, 2019.

[Doersch2016] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[Dumoulin *et al.*2016] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

[Goodfellow *et al.*2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27:2672–2680, 2014.

[Goodfellow *et al.*2016] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[Hinton *et al.*2006] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[Hoang *et al.*2018] Quan Hoang, Tu Dinh Nguyen, Trung Le, and Dinh Phung. Mgan: Training generative adversarial nets with multiple generators. In *International Conference on Learning Representations*, 2018.

[Khorramshahi *et al.*2020] Pirazh Khorramshahi, Hossein Souri, Rama Chellappa, and Soheil Feizi. Gans with variational entropy regularizers: Applications in mitigating the mode-collapse issue. *arXiv preprint arXiv:2009.11921*, 2020.

[Lala *et al.*2018] Sayeri Lala, Maha Shady, Anastasiya Belyaeva, and Molei Liu. Evaluation of mode collapse in generative adversarial networks. *High Performance Extreme Computing*, 2018.

[LeCun *et al.*1998] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[Lin *et al.*2020] Zinan Lin, Ashish Khetan, Giulia Fanti, and Sewoong Oh. Pacgan: The power of two samples in generative adversarial networks. *IEEE Journal on Selected Areas in Information Theory*, 1(1):324–335, 2020.

[Metz *et al.*2016] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.

[Olszewski *et al.*2017] Kyle Olszewski, Zimo Li, Chao Yang, Yi Zhou, Ronald Yu, Zeng Huang, Sitao Xiang, Shunsuke Saito, Pushmeet Kohli, and Hao Li. Realistic dynamic facial textures from a single image using gans. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5429–5438, 2017.

[O'Neill1987] Barry O'Neill. Nonmetric test of the minimax theory of two-person zerosum games. *Proceedings of the national academy of sciences*, 84(7):2106–2109, 1987.

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

[Srivastava *et al.*2017] Akash Srivastava, Lazar Valkov, Chris Russell, Michael U Gutmann, and Charles Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *Advances in neural information processing systems*, pages 3308–3318, 2017.

[Tolstikhin *et al.*2017] Ilya O Tolstikhin, Sylvain Gelly, Olivier Bousquet, Carl-Johann Simon-Gabriel, and Bernhard Schölkopf. Adagan: Boosting generative models. In *Advances in neural information processing systems*, pages 5424–5433, 2017.

[Tripathy *et al.*2020] Soumya Tripathy, Juho Kannala, and Esa Rahtu. Icface: Interpretable and controllable face reenactment using gans. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 3385–3394, 2020.

[Wang *et al.*2018] 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*, pages 8798–8807, 2018.

[Welling *et al.*2003] Max Welling, Richard S Zemel, and Geoffrey E Hinton. Self supervised boosting. In *Advances in neural information processing systems*, pages 681–688, 2003.

[Zhang *et al.*2018] Zhaoyu Zhang, Mengyan Li, and Jun Yu. On the convergence and mode collapse of gan. In *SIGGRAPH Asia 2018 Technical Briefs*, pages 1–4. 2018.