Generative Adversarial Networks (GANs): Challenges, Solutions, and Future Directions

Divya Saxena

University Research Facility in Big Data Analytics (UBDA), The Hong Kong Polytechnic University, Hong Kong, divya.saxena.2015@ieee.org

Jiannong Cao

Department of Computing and UBDA, The Hong Kong Polytechnic University, Hong Kong, csjcao@comp.polyu.edu.hk

ABSTRACT

Generative Adversarial Networks (GANs) is a novel class of deep generative models which has recently gained significant attention. GANs learns complex and high-dimensional distributions implicitly over images, audio, and data. However, there exist major challenges in training of GANs, i.e., mode collapse, non-convergence and instability, due to inappropriate design of network architecture, use of objective function and selection of optimization algorithm. Recently, to address these challenges, several solutions for better design and optimization of GANs have been investigated based on techniques of re-engineered network architectures, new objective functions and alternative optimization algorithms. To the best of our knowledge, there is no existing survey that has particularly focused on the broad and systematic developments of these solutions. In this study, we perform a comprehensive survey of the advancements in GANs design and optimization solutions proposed to handle GANs challenges. We first identify key research issues within each design and optimization technique and then propose a new taxonomy to structure solutions by key research issues. In accordance with the taxonomy, we provide a detailed discussion on different GANs variants proposed within each solution and their relationships. Finally, based on the insights gained, we present promising research directions in this rapidly growing field.

Index Terms—Generative Adversarial Networks, GANs Survey, Deep learning, GANs, Deep Generative models, GANs challenges, GANs applications, Image generation, GANs variants

1. INTRODUCTION

Deep generative models (DGMs), such as Restricted Boltzmann Machines (RBMs), Deep Belief Networks (DBNs), Deep Boltzmann Machines (DBMs), Denoising Autoencoder (DAE), and Generative Stochastic Network (GSN), have recently drawn significant attention for capturing rich underlying distributions of the data, such as audio, images or video and synthesize new samples. These deep generative models are modelled by Markov chain Monte Carlo (MCMC) based algorithms [1][2]. MCMC-based approaches calculate the gradient of log-likelihood where gradients vanish during training advances. This is the major reason that sample generation from the Markov Chain is slow as it could not mix between modes fast enough. Another generative model, variational autoencoder (VAE), uses deep learning with statistical inference for representing a data point in a latent space [3] and experiences the complexity in the approximation of intractable probabilistic computations. In addition, these generative models are trained by maximizing training data likelihood where likelihood-based methods go through the curse of dimensionality in many datasets, such as image, video. Moreover, sampling from the Markov Chain in high-dimensional spaces is blurry, computationally slow and inaccurate.

To handle the abovementioned issues, Goodfellow, et al. [4] proposed Generative Adversarial Nets (GANs), an alternative training methodology to generative models. GANs is a novel class of deep generative models in which backpropagation is used for training to evade the issues associated with MCMC training. GANs training is a minimax zero-sum game between a generative model and a discriminative model. GANs has

gained a lot of attention recently for generating realistic images as it avoids the difficulty related to maximum likelihood learning [5]. Figure 1 shows an example of progress in GANs capabilities from year 2014 to 2018.

Figure 1. Progress in the GANs capabilities for image generation from year 2014 to 2018. Figure from [4][6][7][8][9]

GANs are a structured probabilistic model which comprises of two adversarial models: *a generative model*, called Generator (G) for capturing the data distribution and *a discriminative model*, called Discriminator (D) for estimating the probability to find whether a data generated is from the real data distribution or generated by G's distribution. A two-player minimax game is played by D and G until Nash equilibrium using a gradient-based optimization technique (Simultaneous Gradient Descent), i.e., G can generate images like sampled from the true distribution, and D cannot differentiate between the two sets of images. To update G and D, gradient signals are received from the loss induced by calculating divergences between two distributions by D. We can say that the three main GANs design and optimization components are as follows: (i) network architecture, (ii) objective (loss) function, and (iii) optimization algorithm.

GANs has worked well on several realistic tasks, such as image generation [8][9], video generation [11], domain adaptation [12], and image super-resolution [10], etc. Despite its success in many applications, traditional GANs is highly unstable in training because of the unbalanced D and G training. D utilizes a logistic loss which saturates quickly. In addition, if D can easily differentiate between real and fake images, D's gradient vanishes and when D cannot provide gradient, G stops updating. In recent times, many improvements have been introduced for handling the mode collapse problem as G produces samples based on few modes rather than the whole data space. On the other hand, several objective (loss) functions have been introduced to minimize a divergence different from the traditional GANs formulation. Further, several solutions have been proposed to stabilize the training.

1.1. Motivation and Contributions

In recent times, GANs has achieved outstanding performance in producing natural images. However, there exist major challenges in training of GANs, i.e., mode collapse, non-convergence and instability, due to inappropriate design of network architecture, use of objective function and selection of optimization algorithm. Recently, to address these challenges, several solutions for better design and optimization of GANs have been investigated based on techniques of re-engineered network architectures, new objective functions and alternative optimization algorithms. To study GANs design and optimization solutions proposed to handle GANs challenges in contiguous and coherent way, this survey proposes a novel taxonomy of different GANs solutions. We define taxonomic classes and sub-classes addressing to structure the current works in the most promising GANs research areas. By classifying proposed GANs design and optimization solutions into different categories, we analyze and discuss them in a systematic way. We also outline major open issues that can be pursued by researchers further.

There are a limited number of existing reviews on the topic of GANs. [13] and [14] provided an limited overview of the GANs taxonomy and introduced some of the architecture-variants and loss-variants of GANs. [15]–[17] provided a brief introduction of some of the GANs models, while [17] also presented development trends of GANs, and relation of GANs with parallel intelligence. [18] reviewed various GANs methods from the perspectives of algorithms, theory, and applications. [19] categorized GANs models into six fronts, such as architecture, loss, evaluation metric, etc., and discussed them in brief. [20] presented a brief summary of

GANs models addressing the GANs challenges. [21][22] and [23] provided a brief overview of some GANs models and their applications. [24] and [25] discussed on specific methods proposed for GANs stability. On the other hand, several researchers reviewed specific topics related to GANs in detail. [26] reviewed GANs based image synthesis and editing approaches. [27] surveyed threat of adversarial attacks on deep learning. [28] discussed various types of adversarial attacks and defenses in detail.

Despite reviewing the state-of-the-art GANs, none of these surveys, to the best of our knowledge, have particularly focused on broad and systematic view of the GANs developments introduced to address the GANs challenges. In this study, our main aim is to comprehensively structure and summarize different GANs design and optimization solutions proposed to alleviate GANs challenges for the researchers that are new to this field.

Our Contributions. Our paper makes notable contributions summarized as follows:

New taxonomy. In this study, we identify key research issues within each design and optimization technique and present a novel taxonomy to structure solutions by key research issues. Our proposed taxonomy will facilitate researchers to enhance the understanding of the current developments handling GANs challenges and future research directions.

Comprehensive survey. In accordance with the taxonomy, we provide a comprehensive review of different solutions proposed to handle the major GANs challenges. For each type of solution, we provide detailed descriptions and systematic analysis of the GANs variants and their relationships. But still, due to the wide range of GANs applications, different GANs variants are formulated, trained, and evaluated in heterogenous ways and direct comparison among these GANs is complicated. Therefore, we make a necessary comparison and summarize the corresponding approaches w.r.to their novel solutions to address GANs challenges. We provide a detailed investigation into the numerous research domains where GANs has been broadly explored. This survey can be used as a guide for understanding, using, and developing different GANs approaches for various real-life applications.

Future directions. This survey also highlights the most promising future research directions.

1.2. Organization

In this paper, we first discuss three main components for designing and training GANs framework, analyze challenges with GANs framework, and present a detailed understanding of the current developments handling GANs challenges from the GANs design and optimization perspective.

Section 2 explains the GANs framework from the designing and training perspective. In Section 3, we present the challenges in the training of GANs. In Section 4, we identify key issues related to the design and training of GANs and present a novel taxonomy of GANs solutions handling these key issues. In accordance with the taxonomy, Section 5, 6 and 7 summarize GANs design and optimization solutions, their pros and cons, and relationships. Section 8 discusses the future directions and Section 9 summarizes the paper.

2. GENERATIVE ADVERSARIAL NETWORKS

Before discussing in detail about solutions for better design and optimization of GANs in the proposed taxonomy, in this section, we will provide an overview of GANs framework and main GANs design and optimization components.

2.1. Overview

In recent years, generative models are continuously growing and have been applied well for a broad range of real applications. Generative models' compute the density estimation where model distribution p_{model} is learned to approximate the true and new data distribution p_{data}. Methods to compute the density estimation have two major concerns: selection of suitable objective (loss) function and appropriate selection of formulation for the density function of p_{model}. The selection of objective functions for generative model's

training plays an important role for better learning behaviors and performance [29][30]. The de-facto standard of the most widely used objective is based on the maximum likelihood estimation theory in which model parameters maximize the training data likelihood.

Researchers have shown that maximum likelihood is not a good option as training objective because a model trained using maximum likelihood mostly overgeneralise and generate unplausible samples [29]. In addition, marginal likelihood is intractable which requires a solution to overcome this for learning the model parameters. One possible solution to handle the marginal likelihood intractability issue is not to compute it ever and learn model parameters via a tool indirectly [31].

GANs achieves this by having a powerful D which have a capability to discriminate samples from p_{data} and p_{model}. When D is unable to discriminate samples from p_{data} and p_{model}, then model has learned to generate samples similar to the samples from the real data. A possible solution for formulating density function of p_{model} is to use an *explicit density function* in which maximum likelihood framework is followed for estimating the parameters. Another possible solution is to use an implicit density function for estimating the data distribution excluding analytical forms of p_{model}, i.e., train a G where if real and generated data are mapped to the feature space, they are enclosed in the same sphere [23][24]. However, GANs is the most notably pioneered class of this possible solution.

GANs is an expressive class of generative models as it supports exact sampling and approximate estimation. GANs learns high-dimensional distributions implicitly over images, audio, and data which are challenging to model with an explicit likelihood. Basic GANs are algorithmic architectures of two neural networks competing with each other to capture the real data distribution. Both neural nets try to optimize different and opposing objective (loss) function in the zero-sum game to find (global) the Nash equilibrium. The three main components for design and optimization of GANs are: (i) network architecture, (ii) objective (loss) function, and (iii) optimization algorithm. There has been a large amount of works towards improving GANs by re-engineering architecture [5][6][25], better objective functions [35]–[37], and alternative optimization algorithms [29][30].

In the following sections, we shall discuss three main components for the GANs design and optimization, namely network architecture, loss function and the optimization algorithm followed by the minimax optimization for Nash equilibrium in detail.

2.2. Network Architecture

GANs learns to map the simple latent distribution to the more complex data distribution. To capture the complex data distribution p_{data}, GANs architecture should have enough capacity. GANs is based on the concept of a non-cooperative game of two networks, a generator G and a discriminator D, in which G and D play against each other. GANs can be part of deep generative models or generative neural models where G and D are parameterized via neural networks and updates are made in parameter space.

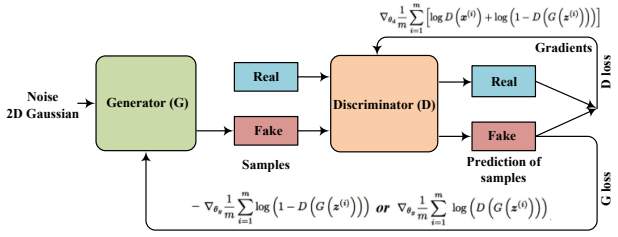


Figure 2. Basic GANs Architecture

Both G and D play a minimax game where G's main aim is to produce samples similar to the samples produced from real data distribution and D's main goal is to discriminate the samples generated by G and samples generated from the real data distribution by assigning higher and lower probabilities to samples from real data and generated by G, respectively. On the other hand, the main target of GANs training is to keep

moving the generated samples in the direction of the real data manifolds through the use of the gradient information from D.

In GANs, x is data extracted from the real data distribution, p_{data}, noise vector z is taken from a Gaussian prior distribution with zero-mean and unit variance p_z, while p_g refers the G's distribution over data x. Latent vector z is passed to G as an input and then G outputs an image G(z) with the aim that D cannot differentiate between G(z) and D(x) data samples, i.e., G(z) resembles with D(x) as close as possible. In addition, D simultaneously tries to restrain itself from getting fooled by G. D is a classifier where D(x) = 1 if $x \sim p_{data}$ and D(x) = 0 if $x \sim p_g$, i.e., x is from p_{data} or from p_g.

The basic GANs architecture for the above discussion is given in Figure 2. For the given basic GANs architecture, the game setup between D and G during training is discussed below.

2.3. Objective (Loss) Function

Objective function tries to match real data distribution p_{data} with p_g. Basic GANs use two objective functions: (1) D minimizes the negative log-likelihood for binary classification; (2) G maximizes the probability of generated samples for being real. D parameters are denoted by θ_D, which are trained to maximize the loss to distinguish between the real and fake samples. G parameters are denoted by θ_G which are optimized such that the D is not able to distinguish between real and fake samples generated by G. θ_G is trained to minimize the same loss that θ_D is maximizing. Hence, it is a zero-sum game where players compete with each other. The following minimax objective applied for training G and D models jointly via solving:

$$\min_{\theta_G} \max_{\theta_D} V(G,D) = \min_{G} \max_{D} \mathbb{E}_{x \sim p_{data}}[\log D(x)] + \mathbb{E}_{z \sim p_z}\left[\log\left(1 - D\big(G(\mathrm{z})\big)\right)\right] \quad (1)$$

$V(G,D)$ is a binary cross entropy function, commonly used in binary classification problems [40]. In Eq. 1, for updating the model parameters, training of G and D are performed by backpropagating the loss via their respective models. In practice, Eq. 1 is solved by alternating the following two gradient updates:

$$\theta_D^{t+1} = \theta_D^t + \lambda^t \nabla_{\theta_D} \mathrm{V}(D^t, G^t) \text{ and } \theta_G^{t+1} = \theta_G^t + \lambda^t \nabla_{\theta_G} \mathrm{V}(D^{t+1}, G^t)$$

where θ_G is the parameter of G, θ_D is the parameter D, λ is the learning rate, and t is the iteration number.

In practice, second term in Eq. 1, $\log\left(1 - D\big(G(z)\big)\right)$ saturates and makes insufficient gradient flow through G, i.e., gradients value gets smaller and stop learning. To overcome the vanishing gradient problem, the objective function in Eq. 1 is reframed into two separate objectives:

$$\max_{\theta_D} \mathbb{E}_{x \sim p_{data}}[\log D(x)] + \mathbb{E}_{z \sim p_z}\left[\log\left(1 - D\big(G(z)\big)\right)\right] \text{ and } \max_{\theta_G} \mathbb{E}_{z \sim p_z}\left[\log\left(D\big(G(z)\big)\right)\right] \quad (2)$$

Moreover, G's gradient for these two separate objectives have the same fixed points and are always trained in the same direction. After cost computation in Eq. 2, Backpropagation can be used for updating the model parameters. Because of these two different objectives, the update rule is given as:

$$\{\theta_D^{t+1}, \theta_G^{t+1}\} \leftarrow \begin{cases} \text{Update} & \text{if } D(x) \text{ predicts wrong} \\ \text{Update} & \text{if } D\big(G(z)\big) \text{ predicts wrong} \\ \text{Update} & \text{if } D\big(G(z)\big) \text{ predicts correct} \end{cases}$$

If D and G are given enough capability with sufficient training iterations, G can convert a simple latent distribution p_g to more complex distributions, i.e., p_g converges to p_{data}, such as $p_g = p_{data}$.

2.4. Optimization Algorithm

In GANs, optimization is to find (global) equilibrium of the minmax game, i.e., saddle point of min-max objective. Gradient-based optimization methods are widely used to find the local optima for classical minimization and saddle point problems. Any traditional gradient-based optimization technique can be used for minimizing each player's cost simultaneously which leads to Nash equilibrium. Basic GANs uses the *Simultaneous Gradient Descent* for finding Nash-equilibria and update D's and G's parameters by

simultaneously using gradient descent on D's and G's utility functions. Each player D and G tries to minimize its own cost/objective function for finding a Nash equilibrium (θ_D, θ_G), $J_D(\theta_D, \theta_G)$ for the D and $J_G(\theta_D, \theta_G)$ for the G, i.e., J_D is at a minimum w.r.to θ_D and J_G is at a minimum w.r.to θ_G.

Minimax optimization for Nash equilibrium. The total cost of all players in a zero-sum game is always zero. In addition, a zero-sum game is also known as a minimax game because solution includes minimization and maximization in an outer and inner loop, respectively. Therefore,

$$J_G + J_D = 0$$

i.e.,

$$J_G = -J_D$$

All the GANs game designed earlier, apply the same cost function to the D, J_D, but they vary in G's cost, J_G. In these cases, D uses the same optimal strategy. D's cost function is as follows w.r.to θ_D:

$$J_D(\theta_D, \theta_G) = -\frac{1}{2}\mathbb{E}_{x\sim p_{data}}[\log D(x)] - \frac{1}{2}\mathbb{E}_z\left[\log\left(1 - D(G(z))\right)\right] \quad (3)$$

Eq. 3 represents the standard cross-entropy cost which is minimized during the training of a standard binary classifier with a sigmoid output. In the minimax game, G attempts to minimize and maximize the log-probability of D being correct and being mistaken, respectively. In GANs, training of a classifier is performed on two minibatches of data: a real data minibatch having examples' label 1 and another minibatch from G having examples' label 0, i.e., the density ratio between true and generated data distribution is represented as follows:

$$D(x) = \frac{p_{data}(x)}{p_g(x)} = \frac{p(x|y=1)}{p(x|y=0)} = \frac{p(y=1|x)}{p(y=0|x)} = \frac{D^*(x)}{1 - D^*(x)}$$

Here, y=0 means generated data and y=1 means real data. $p(y = 1) = p(y = 0)$ is assumed.

Through training D, we get an estimate of the ratio $p_{data}(x)/p_g(x)$ at every point x which allows the computation of divergences and their gradients and sets approximation technique of GANs differ from VAEs and DBMs as they generate lower bounds or Markov Chains based approximations. D learns to distinguish samples from data for any given G. The optimal D for a fixed G is given by $D_G^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_g(x)}$. After sufficient training steps, G and D with enough capacity will converge to $p_g = p_{data}$, i.e., D cannot discriminate between two distributions. For optimal D, the minimax game in Eq. 1 can now be reformulated as follows.

$$J_D = -2\log 2 + 2\left(D_{JS}(p_{data} \parallel p_g)\right) \quad (4)$$

where D_{JS} denotes Jensen-Shannon divergence. When D is optimal, G minimizes Jensen-Shannon divergence (JSD) mentioned in Eq. 4 which is an alternative similarity measure between two probability distributions, bounded by [0,1]. In the basic GANs, it is feasible to estimate neural samplers through approximate minimization of the symmetric JSD.

$$D_{JS}(p_{data} \parallel p_g) = \frac{1}{2}D_{KL}\left(p_{data} \parallel \frac{1}{2}(p_{data} + p_g)\right) + \frac{1}{2}D_{KL}\left(p_g \parallel \frac{1}{2}(p_{data} + p_g)\right),$$

where D_{KL} denotes the KL divergence.

JSD is based on KL divergence, but it is symmetric, and it always has a finite value. Because $D_{JS}(p_{data} \parallel p_g)$ is an appropriate divergence measure between distributions, i.e., real data distribution p_{data} can be approximated properly when sufficient training samples exist and model class p_g can represent p_{data}. The JSD between two distributions is always non-negative and zero when two distributions are equivalent, $J_D^* = -2\log 2$ is the global minimum of J_D which shows $p_g = p_{data}$.

Given the minimax objective, $p_g = p_{data}$ occurs at a saddle point θ_G, θ_D. Saddle point of a loss function occurs at a point which is minimal w.r.to one set of weights and maximal w.r.to another. GANs training

interchanges between minimization and maximization steps. If one step is powerful than another, then solution path "slides off" the loss surface as shown in Figure 3. Due to this, training becomes unstable and network collapses [41].

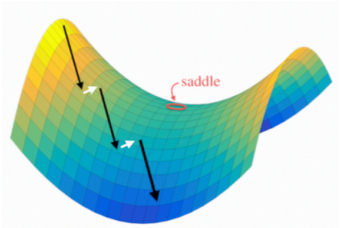


Figure 3. An illustration of gradient method of Adversarial Net [41]

Finding Nash equilibrium in GANs is a very challenging problem as objective functions are non-convex, parameters are continuous, and parameter space is high-dimensional [42], e.g., an update to θ_D that decreases J_D can increase J_G, and an update to θ_G that decreases J_G can increase J_D, i.e., training fails to converge. GANs are still hard to train and training suffers from the following problems, such as difficulty converging and instability, and mode collapse. In the next section, we shall discuss major reasons of these problems in the GANs training in detail.

3. CHALLENGES IN TRAINING GANs

GANs suffer from the limitation of generating samples with little diversity, even trained on multi-model data. E.g., when GANs is trained on hand-written digits' data with ten modes, G may unable to generate some digits [43]. This condition is identified as mode collapse problem and several recent advances in GANs has focused to resolve this problem.

It may also possible that G and D oscillate during training, instead of a fixed-point convergence. When a player gets more powerful than another player, then it may possible that system does not learn and suffers from the vanishing gradients, i.e., instability. When the generated samples are initially very poor, D learns to differentiate easily between real and fake samples. This causes D(G(z)), probability of the generated samples being real, will be close to zero, i.e., gradient of log(1 − D(G(z)) will be very small [44]. This shows that when D fails to provide gradients, G will stop updating. Also, hyperparameters selection, such as batch size, momentum, weight decay, and learning rate is an utmost important factor for GANs training to converge [6].

In this section, we shall discuss about the main challenges in the GANs training in detail.

3.1. Mode Collapse

Mode collapse problem can occur as the max-min solution to the GANs works in a different way from the min-max solution. Therefore, in $G^* = \min_{\theta_G} \max_{\theta_D} V(G,D)$, G^* generates samples from the data distribution. In case of $G^* = \max_{\theta_D} \min_{\theta_G} V(G,D)$, G maps every z value to the single x coordinate that D believes them real instead of fake. Simultaneous gradient descent does not clearly benefit min max over max min or vice versa.

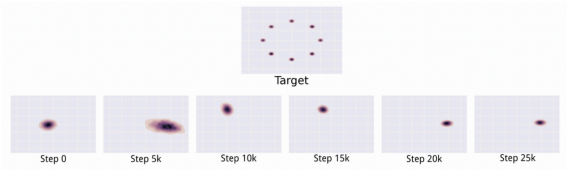


Figure 4. An example for mode collapse problem on a 2D toy dataset. Target distribution is a mixture of Gaussians in 2D space. During GANs training, G generates only single mode at each time step and keep moving among different modes as D is trained to discard separate modes. Figure from [45]

Figure 4 shows GANs training on a toy dataset in which G produces only single mode instead of having multi-model training dataset and keeps cycling between different modes as D continuously rejects samples generated by G. This shows GANs oscillates and have difficulty in achieving Nash equilibrium. Mode collapse is one of the key reasons for the cause of unstable GANs training – another big challenge in GANs.

The main drawback of GANs is that they could not focus on the whole data distribution as GANs' objective function has some similarity with the JSD. Experiments have shown that even for the bi-model distribution, minimizing JSD only produces a good fit to the principal mode and could not generate quality images [30].

Figure 5 illustrates the missing mode problem where G's gradient pushes G towards major mode M_1 for most z while G's gradient pushes G towards M_2 only when G(z) is very close to mode M_2. However, it is likely that in the prior distribution, such z is of low or zero probability. As Figure 5 shows one of the main reasons of mode missing is that the G visits missing modes area rarely, i.e., provides very few examples for improving G around those areas.

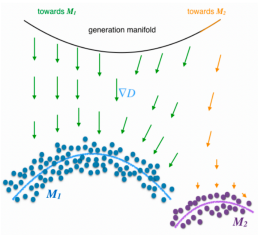


Figure 5. Illustration of missing mode problem. Figure from [46]

Generally, mode collapse is a consequence of poor generalization. There can be two types of mode collapse: (1) most of the modes from the input data are absent from the generated data, (2) only a subset of particular modes is learned by G. An ill-suited objective function can be a major reason for the mode collapse problem where several GANs variants, including modifying D's objective [27][30], modifying G's objective [47] have been proposed. In these variants, G is shown, at equilibrium, and able to learn the whole data distribution, but convergence is elusive in practice. To handle this issue, several recent studies have introduced new network architectures with new objective functions or alternative training schemes.

In the next-subsection, we shall discuss another major challenge in the GANs training.

3.2. Non-convergence and Instability

In the traditional GANs, G uses two loss function as already discussed: $\mathbb{E}_Z[\log \mathrm{D}(\mathrm{G}(z))]$ and $\mathbb{E}_Z\left[\log\left(1-\mathrm{D}(\mathrm{G}(z))\right)\right]$. But, unfortunately, G's loss can lead to potential issues in GANs training.

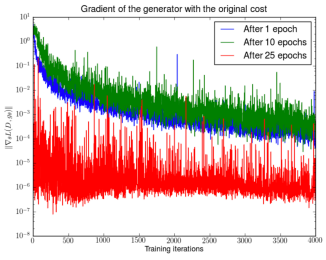


Figure 6. DCGANs for 1, 10 and 25 epochs. G is fixed while D is trained from scratch and using original cost function to compute the gradients [48]

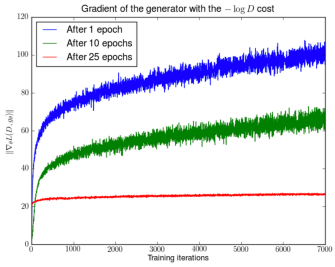


Figure 7. DCGANs for 1, 10 and 25 epochs. G is fixed, D is trained from scratch and using − log D cost function to compute the gradients [48]

The former loss function $\mathbb{E}_Z[\log \mathrm{D}(\mathrm{G}(z))]$ can be the cause of gradient vanishing problem when D can easily differentiate between real and fake samples. For an optimal D, G loss minimization is similar to the minimization of the JSD between real image distribution and generated image distribution. As already discussed, in this case, the JSD will be 2log2. This allows optimal D to assign probability 0 to fake samples,

and 1 to real ones and causes gradient of G loss towards 0 which is called vanishing gradients on G. Figure 6 shows that as D gets better and the gradient of G vanishes.

To minimize the cross-entropy between a target class and a classifier's predicted distribution, classifier requires to choose the correct class. In GANs, D tries to minimize a cross-entropy while G tries to maximize the same cross-entropy. When D confidence is quite high, D rejects the samples generated by G, and then G's gradient vanishes. One possible solution to alleviate this issue is to reverse the target employed for constructing the cross-entropy cost. Thus, G's cost is as follows:

The second one is considered as the −log D trick [4][48][35]. The minimization of the G's loss function $\mathbb{E}_Z\left[\log\left(1-\mathrm{D}(\mathrm{G}(z))\right)\right]$ is equal to minimize $D_{KL}(p_g \parallel p_{data}) - 2\left(D_{JS}(p_{data} \parallel p_g)\right)$ which causes the unstable gradients as it minimizes the KL divergence and maximizes JSD simultaneously. This situation is called instability of G's gradient updates. Figure 7 shows that gradients of the G are growing rapidly. This figure also shows that variance of the gradients growing, i.e., these gradients updates will lead generation of low sample quality.

$$\mathrm{J}_G = -\frac{1}{2}\mathbb{E}_Z\left[\log D(G(z))\right], \text{ or}$$

$$\mathrm{J}_G = -\frac{1}{2}\mathbb{E}_Z\left[\log\left(1-D(G(z))\right)\right],$$

Several GANs design and optimization solutions have been proposed to cope up with the non-convergence and instability problems. We shall discuss key solutions in the subsequent sections.

3.3. Evaluation Metrics

GANs model has been used for the wide applications of the unsupervised representation learning, supervised and semi-supervised learning, inpainting, denoising and many more. For these extensive applications, loads of heterogeneity exists in models' formulation, training and evaluation. Despite the availability of lots of GANs model, the evaluation is still qualitative, (i.e., visual examination of samples by human) even though several approaches and measures have been introduced to evaluate GANs performance. Visual inspection is time-consuming, subjective and cannot capture distributional characteristics, which is an utmost important factor for the unsupervised learning. As the selection of appropriate model is important for getting good performance for an application, the selection of appropriate evaluation metric is important for drawing right conclusion. For designing better GANs model, it is required to overcome the limitations of the qualitative measure by developing or using proper quantitative metrics. Recently, multiple GANs evaluation metrics have been introduced with the emergence of new models.

In this paper, we restrict our focus on GANs design and optimization solutions proposed for handling first two GANs challenges, mode collapse, non-convergence and instability. In the next section, we shall discuss the proposed taxonomy of GANs design and optimization solutions proposed to mitigate these challenges and improve the GANs performance.

4. A TAXONOMY

Table 1 illustrates our proposed taxonomy of GANs designing and optimization solutions proposed to handle two major GANs training challenges as discussed earlier. In recent times, several solutions (S) have been proposed for better design and optimization of basic GANs based on three main techniques, namely re-engineering network architecture (S_1), new loss function (S_2), and alternative optimization algorithm (S_3). *Re-engineered network architecture* (S_1) focuses on the re-engineering GANs network architecture [6][49][8][50][51], *new loss function* (S_2) covers modified or new loss functions for GANs [36][38][46][52], while *alternative optimization algorithm* (S_3) includes modified or regularized optimization algorithms for GANs [45].

Table 1 Proposed taxonomy and different variants proposed within the GANs design and optimization techniques

Techniques	Solutions (S)		Variants
Re-engineered Network Architecture (S_1)	Conditional generation (S_{11})		cGANs [34], FCGAN [53], IRGAN [53], GRANs [54], LAPGAN [5], SGAN [49], IcGAN [55], BiCoGAN [56], MatAN [57], Self-conditioned GANs [58], AC-GANs [59], TripleGAN [60], KDGAN [61], ControlGAN [62]
	Generative-discriminative network pair (S_{12})	Training of Single G (S_{12}(i))	DCGAN [6], ProgressGAN [8], PacGAN [43], BayesianGAN [63], CapsNets [64], QuGANs [65], SAGAN [66]
		Training of Multiple Gs (S_{12}(ii))	cGAN [50], AdaGAN [51], MAD-GAN [67], MGAN [68], MPM GAN [69], FictitiousGAN [70], MIX+GAN [71]
		Training of Multiple Ds (S_{12}(iii))	D2GAN [72], GMAN [73], StabGAN [74], Dropout GAN [75], MicroBatchGANs [76], SGAN [77]
	Joint Architecture (S_{13})	Data space autoencoder (S_{13}(i))	VAE-GAN [78], AAE [79], AVB [80], ASVAE [81], MDGAN [46], Dist-GAN [82], α-GAN [31]
		Latent space autoencoder (S_{13}(ii))	ALI [83], BiGAN [84], DALI [85], CV-BiGAN [86], MV-BiGAN [86], HALI [87], AGE [88], VEEGAN [89], MGGAN [90]
	Improved D (S_{14})		EBGAN [91], BEGAN [92], MAGAN [93], [94], Max-Boost-GAN [95]
	Memory Networks (S_{15})		MemoryGAN [96]
	Latent space engineering (S_{16})		DeLiGAN [97], NEMGAN [98], MultiplicativeNoise [99], DE-GAN [100], InfoGAN [101]
New Loss Function (S_2)	New probability distance and divergence (S_{21})		WGAN [35], LS-GAN [102],RWGAN [37], f-GAN [103], [104], χ2-GAN [105], OT-GAN [106], LSGAN [40], SoftmaxGAN [107], GAN-RL [108], GoGAN [109], IGAN [42], McGAN [110], MMD GAN [52], MMGAN [111], CramerGAN [112]
	Regularization (S_{22})		WGAN-GP [113], BWGAN [36], [114] , CT-GAN [115], SN-GAN [116], [117], FisherGAN [118], [38], Unrolled GANs [45], [119], [120], DRAGAN [121]
Alternative Optimization Algorithm (S_3)			[38], [122], [123], [39], [41]

For each technique, we point out number of key research issues and the corresponding solutions (S) to address the GANs challenges.

1. **Re-engineered network architecture (S_1).** GANs are hard to train as G could not learn the complex data distribution and generates low variety of samples. Therefore, GANs requires better designs of model architectures. In recent times, several architectural solutions have been proposed to handle GANs challenges in different ways, such as conditional generation, generative-discriminative network pair, using strong discriminator, memory network, and encoder-decoder architecture, engineering noise. We classify existing architectural solutions into six categories where each category represents an issue with the existing architecture and its possible solution to improve the GANs performance. The key research issues and their solutions are as follows:

    a. **Conditional generation (S_{11}).** An unconditional generative model cannot control the modes generation. To control the generation process, a generative model can be conditioned on additional information [34]. In this way, conditional GANs learn conditional probability distribution where a condition can be any auxiliary information about the data.

    b. **Generative-discriminative network pair (S_{12}).** Intermediate representation of GANs can only handle the generation of smaller images [6]. It requires architectural changes in the layers and networks for improving the generating ability of GANs. But still, a single pair of G and D in minimax game fluctuates and do not converge as discriminatively trained networks do [50]. To handle this issue, multiple Gs and Ds can be trained to increase the generation capacity of Gs and to get more constructive gradient signals for G, respectively.

    c. **Joint architecture (S_{13}).** Some research works have proposed to use most common approach to address mode collapse, encoder-decoder architecture in GANs, in which features are learned from the latent space or image space.

d. **Improved Discriminator (S_{14}).** Weak discriminators can also be the reason of mode collapse, because of either low capacity or a poorly-chosen architecture. Therefore, researchers proposed to use autoencoder to define the D's objectives.

e. **Memory networks (S_{15}).** Basic GANs encounters instability and divergence in the unsupervised GANs training due to two key issues, namely the structural discontinuity in a latent space and the forgetting problem of GANs. Basic GANs use unimodal continuous latent distribution to embed multiple classes; hence, structural discontinuity between classes is not clear in the generated samples. In addition, Ds forget about the previously generated samples. To handle these issues, [96] proposed a solution to incorporate a memorization module with unsupervised GAN models.

f. **Latent-space engineering (S_{16}).** Basic GANs cannot generate diverse samples in case of limited data availability. Researchers introduce to reparametrize latent space *z* like a mixture model and then learn parameters of mixture model with GANs [97].

2. **New loss function (S_2).** The model parameters oscillate, destabilize and never converge. A use of good loss function improves GANs learning to reach better optima. Several researchers proposed to tackle the training instability problem by finding better distance measures [35] or regularizers [28][45]. The issues and their solutions are as follows:

   a. **New probability distance and divergence (S_{21}).** Basic GANs training is unstable as it uses JS divergence (JSD) which is neither continuous nor provide a usable gradient [35]. There is need of new probability distance and divergence for getting usable gradients everywhere. New probability distance and divergence can solve the mode collapse problem by stabilizing GAN training.

   b. **Regularization (S_{22}).** GANs training is unstable when model distribution and data distribution manifolds do not overlap in the high-dimensional space, i.e., dimensionality misspecification [117]. Regularizing D gives an efficient direction to convolve the distributions.

3. **Alternative optimization algorithm (S_3).** Basic GANs uses *Simultaneous gradient Descent* for finding Nash-equilibria which often fails to find local Nash-equilibria [38]. Some works have suggested to use another gradient descent optimization technique, while some have suggested modifications to optimization or training technique.

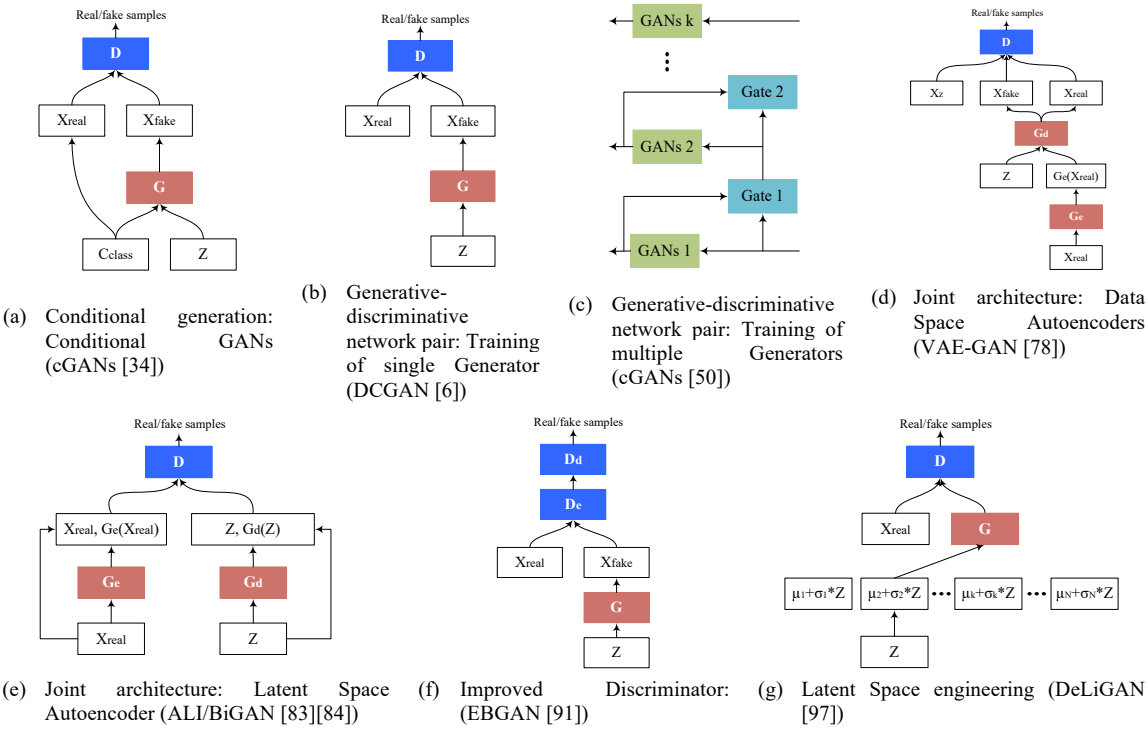
In the following sections (Section 5, 6 and 7), we introduce and compare existing GANs designing and optimization solutions and their variants according to the proposed taxonomy. We review and critically discuss these solutions with their advantages and drawbacks. Furthermore, we provide a supplementary material which contains the loss function(s) used in the re-engineering network architecture taxonomic class and a comparison among new loss functions proposed for GANs. Table 1 lists the different variants proposed within the GANs design and optimization solutions.

5. RE-ENGINEERED NETWORK ARCHITECTURE

As we have already discussed in Section 4, several network architectures have been proposed to scale up the GANs performance, such as hierarchical GANs structure, and engineering on the latent space for generating high quality samples for the limited data, use of autoencoders for D for more stable training, and combination of any of these structures.

Figure 8 shows the schematic illustrations of the most representative solutions within the taxonomic class of re-engineered network architecture. In conditional generation sub-class, conditional GANs use an auxiliary information on G and/or D during the generation to control the mode generation (see Figure 8(a)). In the sub-class of generative-discriminative network pair, single and multiple pair of G and D are trained for better GANs performance. DCGANs uses fully convolutional downsampling/upsampling layers instead of Fully connected layers used in traditional GANs (see Figure 8(b)), while cascade of GANs (cGANs) consists of multiple GANs and gates where each GANs redirects badly modelled part of training data to next GANs for capturing whole distribution of the data (see Figure 8(c)). In addition, within the sub-class of joint

architecture, VAE-GAN (see Figure 8(d)) and ALI/BiGAN (see Figure 8(e)) have introduced an efficient inference mechanism which includes features from the latent space and data space, respectively. In the sub-class of improved discriminator, energy-based GANs replaced encoder architecture for D with an autoencoder architecture for the stable training where D's objective is to match autoencoder loss distribution instead of data distribution (see Figure 8(f)). In the sub-class of memory networks, memory within the network is introduced to handle the mode collapse and instability problem in unsupervised GANs framework. Within the sub-class of latent space engineering, latent space is reparameterized using a mixture of Gaussian model to get samples in the high probability regions in the latent space which supports better GANs performance (see Figure 8(g)).



(a) Conditional generation: Conditional GANs (cGANs [34])

(b) Generative-discriminative network pair: Training of single Generator (DCGAN [6])

(c) Generative-discriminative network pair: Training of multiple Generators (cGANs [50])

(d) Joint architecture: Data Space Autoencoders (VAE-GAN [78])

(e) Joint architecture: Latent Space Autoencoder (ALI/BiGAN [83][84])

(f) Improved Discriminator: (EBGAN [91])

(g) Latent Space engineering (DeLiGAN [97])

D: Discriminator, G: Generator, Z: Latent (noise) space, Xreal: Real data, Xfake: Fake data generated by G, Cclass: an auxiliary information, Ge: Encoder for G, Gd: Decoder for G, De: Encoder for D, Dd: Decoder for D.

Figure 8. Schematic view of most representative GANs variants within the taxonomic class of network architecture GANs

In this section, we shall thoroughly study the different designs of GANs network architectures with their strengths and drawbacks.

5.1. Conditional Generation

Conditional generation (e.g., cGANs [34]) have shown a significant improvement in generating good sample quality. Conditional GANs has shown a potential application for the image synthesis and image editing applications. In addition, a class-conditional model of images is not significant if it produces only one image per class. In conditional GANs (cGANs), a condition c is induced on both G and D. The main aim of cGANs is to generate realistic images instead of making difference between generated samples based on input conditions. On the other hand, in conditional GANs, condition vectors are concatenated into some layers of G and D (see Figure 9). For example, in fully conditional GANs (FCGAN), each layer of D including x is conditioned. This joint representation cannot capture the complex relationships between two distinct modalities [124]. To handle this issue, [53] introduced a Spatial Bilinear Pooling (SBP) approach where each pixel of an image is conditioned, i.e., multiplicative interaction between all elements of two vectors.

Moreover, an Information Retrieving GANs (IRGAN) [53] is also proposed in which latent codes are conditioned explicitly. IRGAN explicitly put condition information in the latent codes.

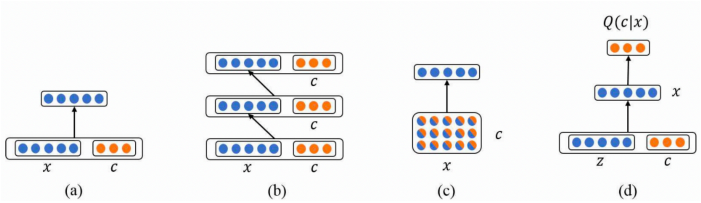


Figure 9. Conditional GANs: (a) cGANs, (b) FCGAN, (c) SBP, and(d) IRGAN. Figure from [53]

Im, et al. [54] introduced a new class of GANs based on the Recurrent Neural Networks (RNNs) instead of CNNs as Gs for GANs, called Generative Recurrent Adversarial Networks (GRANs). GRANs generate an image in a sequence of structurally identical steps without imposing a coarse-to-fine (or any other) structure on the generation process. The main advantage of sequential modelling in which generated output is conditioned on previous states repetitively is that it makes the complex data distributions modelling problem simple by mapping them to a sequence of easy problems. Moreover, authors also proposed new metric, called Generative Adversarial Metric (GAM) for evaluating adversarial networks quantitatively.

As image data have high variations, generation of diverse images with sufficient details is quite challenging for GANs. Recently, some works have focused to decompose GANs into a sequence of GANs for breaking difficult task into tractable sub-tasks. Denton, et al. [5] integrated a conditional model on the cascade of convolutional GANs with the framework of Laplacian pyramid (LAPGAN) with *k* levels. LAPGAN model generates and upsamples images in multiple steps and have shown the generation of the higher quality images. But they suffered from the objects looking wobbly as noise added in chaining multiple models. In particular, to generate an image, LAPGAN model explicitly decomposes task into a sequence of conditional generations of levels of a Laplacian pyramid. [49] proposed another model, called Stacked GANs (SGAN) which is composed of a top-down stack of GANs and each stack is trained to produce low-level representations conditioned on high-level representations. The problem of estimating image distribution is decomposed into small tasks and intermediate supervision is provided by representation D at each training hierarchy. Authors introduced conditional loss to let G employ the high-level conditional information and an entropy loss to encourage each G for generating diverse representations in addition to adversarial loss (i.e., divergence measure). In addition, a representation Ds is proposed to be used at each hierarchy for providing the intermediate supervision to G at that level. Previous approaches introduced several loss terms for regularizing G's output without regularizing its internal representations. SGAN architecture is similar to LAPGAN as both consist of a sequence of GANs but LAPGAN generates *multi-resolution* images from coarse-to-fine (i.e., a sequential adversarial network) while SGAN models *multi-level representations* from abstract-to-specific. SGAN also works similar to InfoGAN [101] w.r.to the variational mutual information maximization technique but InfoGAN is used to forecast simply a small set of latent code, whereas SGAN predicts noise variables in each stack. Furthermore, InfoGAN maximizes mutual information between the output and the latent code, whereas SGAN maximizes the entropy of the output h_i conditioned on h_{i+1}.

On the other hand, A cGANs must be capable to disentangle the intrinsic (latent variables) and extrinsic factors (known as auxiliary information), and also disentangle extrinsic factors' components from each other, in the generation process. Inverse cGANs produces disentangled information-rich representation of data which can be employed for some downstream tasks, such as classification. To achieve such optimal framework, [55] proposed an Invertible cGANs (IcGAN) to learn inverse mappings to intrinsic and extrinsic factors for pretrained cGANs by using two standalone encoders (Es) trained post-hoc, one for each task. However, IcGAN suffers from the following two limitations: IcGAN prevents E from having an effect on factors' disentanglement during the generation process and avoids E from learning the inverse mapping to intrinsic factors effectively.

In addition, existing encoder-based cGANs models suffer from the two major shortcomings: (1) extrinsic factors are not encoded [79], (2) if encoded, then encode them in fixed-length continuous vectors which do

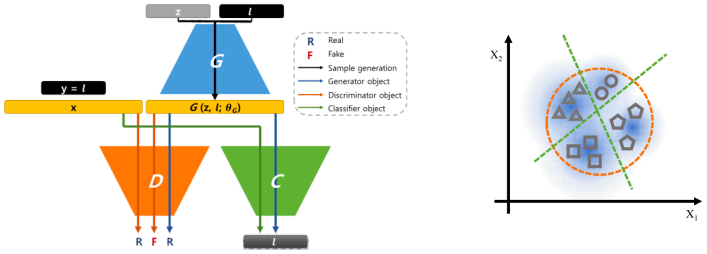
not have an explicit form [125]. This avoids data generation with arbitrary combinations of extrinsic attributes. To alleviate encoder-based cGANs issues, [56] proposed a network architecture, called Bidirectional cGANs (BiCoGAN) which learns inverse mappings of data samples to both intrinsic and extrinsic factors, and is trained simultaneously with G and D. BiCoGAN is similar to Bidirectional GANs (BiGANs) [84] w.r.to implicit regularization, mode coverage and robustness against mode collapse. Unlike conditional extension of ALI model (cALIM) [83], BiCoGAN involves both data samples and extrinsic attributes as inputs to encode the intrinsic features. However, it does not involve extrinsic attributes encoding from data samples. BiCoGAN could not produce better results as E could not perform the inverse mapping to extrinsic attributes and G does not include the extrinsic factors during the data samples generation. To handle these problems, training techniques are also introduced.

Existing researches show that cGANs could not perform well for the supervised tasks, such as semantic segmentation, instance segmentation, line detection, etc. The possible reason can be that G is optimized by minimizing a loss function that does not depend directly on the real data labels. To handle the aforementioned issue, [57] proposed to replace D with a siamese network working on both the real data and the generated samples to allow G's loss function to depend on the targets of the training examples. This approach is called Matching Adversarial Networks (MatAN) which can be utilized as a D network for supervised tasks. In addition, [58] proposed a class conditional GANs which is trained without manually annotated class labels. While, labels are derived automatically by applying clustering in the D's feature space. Clustering step finds diverse modes automatically and requires G to cover them explicitly.

On the other hand, some researchers proposed to use the Classifier (C), reconstructing side information to increase the performance of cGAN. [59] proposed auxiliary classifier GANs (AC-GAN) where a C is used as D of GANs architecture and a condition is categorial class label Cclass to only G instead of additional condition c to both G and D. In AC-GAN, D estimates a probability distribution over both sources and Cclass. To generate realistic images, AC-GAN demonstrated that addition of more structure to latent space with a particular loss function works well. But this variant cannot perform well for the semi-supervised learning as D suffers from two incompatible convergence points: discrimination of real and fake data and prediction of class label, and G cannot produce data in a particular class. Moreover, GANs, a two-player game consumes high time to reach equilibrium due to high-variance gradient updates. Also, G cannot control the semantics of generated samples.

To handle this issues, [60] proposed a three-player adversarial game to drive G to match the conditional distribution $p(x|y)$, called Triple-GAN for both classification and class-conditional image generation in semi-supervised learning where C results is passed to D as input. D identifies data-label pair is from the real dataset or not, while G and C characterize the conditional distributions between images and labels. But, TripleGAN does not have constraints to guarantee that semantics of interest will be captured by y and also does not have a structure for achieving posterior inference for z.

Wang, et al. [61] also introduced a three-player game, called KDGAN, consists of a C, a teacher T, and a D in which C and T are trained from each other using distillation losses and are adversarially trained against D using adversarial losses. C learns the true data distribution at the equilibrium through the simultaneously optimization of distillation and adversarial losses. To achieve the low-variance gradient updates for speed up the training, samples are generated from the concrete distribution. Moreover, KDGAN achieves stable training when C impeccably models the true data distribution. But KDGAN suffers from G collapse when the class count increases. KDGAN idea is inspired by [126] in which D is used to train C for learning the data distribution produced by T, while in KDGAN, D trains C to learn the real data distribution directly. In addition, even though both Triple-GAN and KDGAN introduced three-players game, both models have some differences as follows: (1) In KDGAN, C and T (i.e., G) learn conditional distribution over labels given features, while in Triple-GAN, C and G learn a conditional distribution over labels given features and a conditional distribution over features given labels, respectively, (2) In KDGAN, generated samples from Gs are discrete data, while in Triple-GAN, generated samples include both discrete and continuous data.



(a) (b)

Figure 10. (a) ControlGAN architecture [62]**. (b) ControlGAN concept. Green dashed line represents C and orange line represents D. Grey boxes represents samples labelled with different class. Blue region, i.e., represents G which tries to learn data distribution and classify samples to correct labels, simultaneously.**

Furthermore, AC-GAN, TripleGAN and KDGAN use a classifier C connected to D, i.e., D chooses the samples condition. Therefore, these solutions can hardly deal with cGANs limitation, i.e., cGANs can generate images of different styles but it is not possible to generate images from two different domains, such as color and depth image domains. In addition, cGANs generates specific images, such as, face conditioned on the attribute vector, but cannot model image distribution conditioned on a part of that image or on previous frames [127], i.e., for image generation, major features, such as smile can be conditioned instead of detailed features, such as pointy nose. This occurs because D decides whether the condition/label is correct or not. Due to this, limitation of cGANs cannot be handled by [60][59] as if very few examples are available for a particular condition/label in a dataset and/or condition/label is far from the data distribution's center where the samples densely exist, D distinguishes samples with such conditions fake. To handle this limitation, Lee, et al. [62] proposed to combine GANs with a decoder-encoder structure based architecture for controlling generated samples with detailed feature, called Controllable Generative Adversarial Network (ControlGAN) in which GANs game is formulated as three players game of G/De, D and C/E (see Figure 10). In ControlGAN, G tries to make fool D and be classified correctly by C. In ControlGAN, an independent network is used to map the features into corresponding input labels which dedicate D only for distinguishing real and fake samples and enhances the quality of generated data. This also allows ControlGAN to generate data beyond the training data. In addition, ControlGAN also uses an equilibrium parameter to balance between GANs and a decoder-encoder structure for stable training.

5.2. Generative-discriminative Network Pair

In recent years, GANs has achieved great success due to its ability to generate realistic samples, but traditional single-generator GANs worked well for only small images, such as MNIST but could not model the large images. To handle this issue, some researchers proposed to use the multiple Gs rather than a single one for generating high quality images by increasing the generation capacity of Gs

In addition, some researchers advocated to use the multiple Ds while some formulated the minimax game using multiple Ds and Gs. We classify these approaches into three categories: (1) training of single generator (G), (2) training of multiple generators (Gs), and (3) training of multiple discriminators (Ds). We shall discuss these approaches in this section in detail.

5.2.1. Training of Single Generator

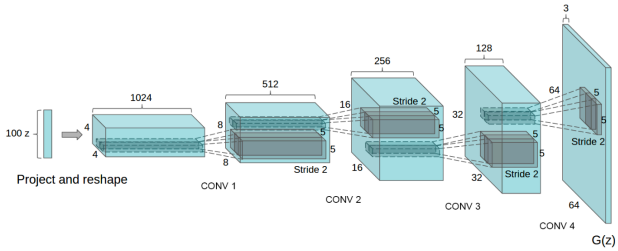


Figure 11. DCGANs Generator. A sequence of four fractionally-strided convolutions for converting 100-dimensional uniform distribution *z* into a 64 × 64 pixel image. Figure from [6]

Basic GANs generated images are noisy and incomprehensible. GANs are unstable to train. Also, generation of high-resolution images is difficult. Therefore, there is need of a set of architectures resulting stable training for a range of datasets and allowing training of higher resolution and deeper generative models.

Radford et al. [6] successfully designed a class of architecturally constrained deep convolutional GANs, called DCGANs, which has shown substantial advancements on unsupervised image representation learning, i.e., more stable training and generates superior quality images. The architectural changes for stable DCGANs are as follows: (1) Use strided convolutions (D) and fractional-strided convolutions (G) instead of pooling layers (see Figure 11), (2) exploit batchnorm in both G and D, (3) do not use fully connected hidden layers for deep architectures, (4) apply ReLU activation in all G's layers excluding the output which uses Tanh, and (5) Use LeakyReLU activation in D for all layers. Even though DCGANs performed well in compared to basic GANs, DCGANs still suffered from some form of model instability – authors observed that as models training time is extended, DCGANs used to collapse a subset of filters to a single oscillating mode at times. It requires further investigation to handle this form of instability.

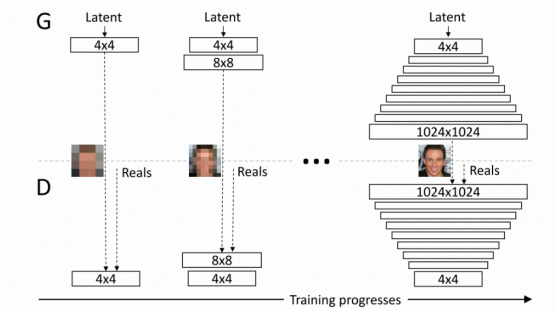


Figure 12. Illustration of training process of ProgressGAN. G and D start with the low spatial resolution of 4×4 pixels and it keeps growing as the training advances. Figure from [8].

Karras, et al. [8] introduced a novel training strategy for GANs, called ProgressGAN, in which low-resolution images are passed as input and then resolution of images is increased gradually by adding layers at each stage. The main idea of the proposed methodology is to grow the network of G and D step-by-step as it allows learning of large-scale structure of the image distribution first and then other finer details. Figure 12 demonstrates the training process of ProgressGAN This speedup training and improves stability. On the other hand, new layers are fade in smoothly to avoid the unexpected shocks to the well-trained, smaller-resolution layers. The proposed GANs training strategy of increasing resolutions performed well on CelebA dataset with size 1024 × 1024 and generated most realistic looking faces. A layer of minibatch standard deviation is also added at the end of D for capturing the diversity in the minibatch. This idea is quite simple to capture minibatch statistics and resolves the issue of sensitivity to hyperparameter tuning of the original minibatch idea of [42]. Lin, et al. [43] presented a new GANs framework, called PacGAN, which can be used by any existing GANs with a slight change to D. The main aim is to pass *m* packed/concatenated samples from the same class to D to be jointly classified as either real or generated. Packing penalizes Gs with mode collapse, therefore, favors distribution of G with less mode collapse during training. PacGAN needs no hyperparameter tuning but induces a little architecture's overhead.

On the other hand, some researchers proposed to integrate the GANs framework with other frameworks to improve the generating ability of the GANs. As a result, a strong generative model with better generation capability is achieved instead of two separate models. [63] introduced a Bayesian framework for unsupervised and semi-supervised learning with GANs to improve the generative ability. In GANs, updates are implicitly conditioned on a set of noise samples *z*, but in Bayesian GANs, *z* is marginalized from posterior updates using simple Monte Carlo. Jaiswal, et al. [64] introduced a change in D's architecture and argued to use a capsule networks (CapsNets) instead of standard CNNs. Results show that the learned images by CapsNets are more robust to changes in pose and spatial relationships of parts of objects in images. Authors also updated GANs objective function to *CapsNets margin loss* for training. CapsNets achieved good performance for the datasets, MNIST and CIFAR-10, while could not perform better for the complex datasets, such as ImageNet.

Lloyd, et al. [65] proposed a quantum adversarial game, called quantum generative adversarial networks (QuGANs) to show that quantum adversarial networks may have a great benefit over classical adversarial networks for high-dimensional data scenarios. In QuGANs, D optimizes strategy with the G's fixed strategy and G optimizes strategy with the D's fixed strategy over a fixed number of trials. QuGANs achieves Nash equilibrium when G finds the correct statistics while D cannot find difference between true and generated data. Most of the GANs models [6][42][105] for the image generation have used convolutional layers where convolution processes the information in a local neighborhood only and could not model long-range dependencies in images efficiently. Zhang, et al. [66] introduced a method for both G and D to model spatial relationship among separated regions in the images, called Self Attention GAN (SAGAN). Authors also presented two techniques, such as use of spectral normalization [116] in the G and D for stable GANs training (spectral normalization does not require extra hyperparameter tuning) and use of two-timescale update rule (TTUR) [39] for addressing slow learning in regularized Ds. In addition, to handle mode collapse and support stable training, [94] introduced explicit manifold learning as prior for GANs. A new target of Minimum Manifold Coding is further enforced for manifold learning to find simple and unfolded manifolds which works even in the case of the sparsely or unevenly distributed data.

5.2.2. Training of Multiple Generators

Ensembles have already shown potential for improving the results of discriminative CNNs. Wang, et al. [50] explored the usage of ensembles of GANs and proposed a framework having the cascades of the GANs, called cGANs, in which G focuses on capturing the whole data distribution instead of the principal mode of the data. cGANs consists of multiple GANs where a G in each GANs tries to capture current data distribution which previous GANs could not captured efficiently. For selecting data, which is passed to the next GANs, it is considered that for badly modeled data x, the D value D(x) should be high, i.e., D is confident that x is real data. Gate function is used to re-direct the data to the next GANs. If D value D(x) is greater than a pre-determined threshold value tr, then x will be used in the next GANs. However, additive procedure of cGANs is not motivated by the theoretical analysis of optimality conditions. Further, Tolstikhin, et al. [51] have shown through the empirical analysis that cGANs heuristic fails to address the mode collapsing problem. They introduced a meta-algorithm, called Adaptive GAN (AdaGAN) similar to AdaBoost [129] in which each iteration corresponds to learning a weak generative model w.r.to a re-weighted data distribution. The assigned weight keeps changing to handle the hard samples. AdaBoost reweighs the training data and trains new Gs incrementally to get a mixture covering the whole data space. AdaGAN can be used on the top of a G architecture, such as a Gaussian mixture model, VAE [3], WGAN [35], UnrolledGAN [45] or mode-regularized GANs [46], which have been developed to handle the mode collapse problem. However, training multiple Gs in an iterative manner is computationally expensive. In addition, AdaBoost assumes that GANs based on single-generator can produce realistic images for some modes, such as dogs or cats but cannot capture other modes, such as giraffe [51]. So, by removing the images of dogs or cats manually from the training data and train a next GANs over this data can create a better mixture. But, in practice, this assumption does not work as single-generator GANs produces images of unrecognizable objects for diverse datasets, such as ImageNet.

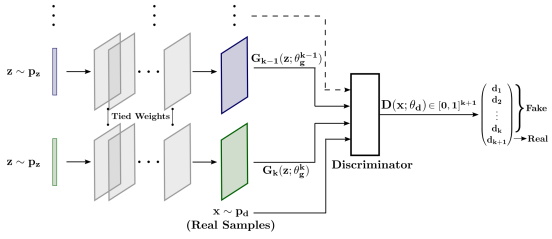


Figure 13. Multi-Agent Diverse GAN (MAD-GAN). The D outputs $k + 1$ softmax scores signifying the probability of its input sample being from either one of the k Gs or the real distribution [67]**.**

Ghosh, et al. [67] introduced a novel adversarial architecture consisting of a C and multiple Gs trained by a multi-class D which differentiate among fake and real samples with the identification of the G generated fake

sample, called MAD-GAN. The idea is to approximate data distribution using a mixture of multiple distributions where each distribution captures a subset of data modes separately from those of others (see Figure 13). The idea of using multiple Gs is same as a mixture model where each G learns one data mode. In addition, objective function penalizes Gs for generating fake samples while does not encourage Gs to focus in generating variety of modes. Experimental results have shown that MAD-GAN performed extremely well for the highly challenging diverse-class dataset. Hoang, et al. [68] extended the concept of multi-generator GANs, called MGAN, where a set of Gs are trained simultaneously (share parameters) to approximate data distribution using a mixture of multiple distributions with the encouragement to specialize in different modes in the distribution. MGAN uses an additional model to classify (rather than modifying D) which G generated fake input where output of the classifier is further used as penalty term to force diversity among Gs. The objective function of MGAN focus to minimize the JSD between the mixture of distributions induced by the Gs and the data distribution and to maximize the JSD among Gs. Ghosh, et al. [69] presented a multi-agent GANs framework of multiple Gs communicating through message passing, called MPM GANs, for better image generation. Objectives are to pass messages among Gs for capturing different data modes and promote competition among the Gs and tries to make other G better than the current G. Ge, et al. [70] proposed a training algorithm, called Fictitious GANs in which D is trained on the mixed outputs from a series of trained Gs. Fictitious GANs is a meta-algorithm which can be used on the top of existing GANs variants.

Another direction for handling mode collapse and support more stable training is to train multiple Gs and Ds in GANs as use of mixture guarantees existence of approximate equilibrium. Arora, et al. [71] proposed to train multiple Gs and Ds with different parameters, called MIX+GAN and optimize the minimax game with weighted average reward function over all pairs of G and D. But, training of MIX+GAN is computationally expensive as solution lacks parameter sharing and enforcing divergence among Gs is missing.

5.2.3. Training of Multiple Discriminators

Nguyen, et al. [72] proposed to use two Ds for yielding constructive gradient signals for G. In single generator dual discriminator architecture (D2GAN), one D works on the KL divergence to reward samples from the true data distribution while another D works on the reverse KL divergence to reward samples generated by G where G tries to fool both two Ds. This helps to avoid mode collapse problem. The combined use of KL divergence and reverse KL divergence into a unified objective function attempts to diversify the estimated density in learning multi-modes effectively. However, they tend to increase instability because the goals of two antithetical Ds conflict. Durugkar, et al. [73] also used several Ds for boosting the learning of G and stabilize GANs, called Generative Multi-Adversarial Network (GMAN). In GMAN, either average loss of all Ds or a D with minimum loss is used to train G. The main idea is to accelerate training of G to a more robust state irrespective of the choice of cost function.

Despite the GANs progress, GANs is unstable in the case of high-dimensional data as real data distribution can have the possibility to be focused in a small fraction of the ambient space. Due to this, D can easily classify almost all generated samples as fake and does not provide meaningful gradients to D. Neyshabur, et al. [74] proposed a different approach to handle the instability in which G is trained against an array of Ds where each D handles different, randomly-chosen, low-dimensional projection of the data, (StabGAN). Existing similar approaches either train ensemble of GANs [50] or ensemble of Ds [73] while in this, low-dimensional projection of the data is provided to D so that D cannot reject generated samples perfectly, i.e., provides meaningful gradients to G during training. Instead of combining each D's output directly, each D's loss is calculated individually and then an average of all D's loss is calculated. Moreover, G learns real data distribution to fool all Ds simultaneously.

In the previous two works, G's output depends on the feedback given by a specific set of Ds which is not a robust solution for handling mode collapsing and compromises the extensibility of framework. As a solution, Mordido, et al. [75] proposed to apply dropout mechanism by dynamically selecting feedback from ensemble of Ds that change at each batch for G's learning and to promote variety in its output, called Dropout-GAN. On the other hand, the drop of a particular D's loss selected with a probability *d* before updating parameters of G, supports variability in the solution. In this case, G fools a dynamic set of Ds at every batch instead of

one or a static set of Ds. Therefore, this solution can be known as regularization as it targets to encourage more generalizability on the fake samples produced by G. Durugkar et al. [73], and Nguyen, et al. [72] approaches limit the extensibility as they condition the D's architecture for promoting variety either by having convolutional architecture for D or by having different architectures for each D, while, the use of dropout approach does not compromise extensibility of the solution. In addition, to handle the issue of mode collapse in GANs, [76] proposed to assign a different portion of each minibatch, called microbatch, to a D. D's task of distinguishing between real and fake data is also gradually changed to discriminating samples from inside or outside its assigned microbatch with the use of a diversity parameter α. G is asked to generate diversity in each minibatch so that it is hard for each D to discriminate microbatch. Existing approaches of multiple discriminators perform single-objective optimization on some simple consolidation of the losses, e.g., an arithmetic average. [130] extended this single-objective optimization problem to multi-objective in which losses provided by different models are minimized simultaneously.

Chavdarova and Fleuret [77] proposed an alternative way for GANs training, called SGAN, in which a global pair of G_0 and D_0 is trained indirectly, i.e., D_0 is trained with G_i, $i = 1, \ldots, N$, and G_0 is trained with D_i, $i = 1, \ldots, N$. The main advantage of such training procedure is as follows: (1) global networks continue to learn with higher probability even if a particular local pair's training worsens; and (2) High computation can be performed in parallel which makes the time overhead less important factor. Compared to [51], SGAN can be used by any existing GANs framework as it runs in parallel and yields a single G.

5.3. Joint Architecture

GANs can generate more visually compelling sample images in compared to (V)AEs but in GANs, more complex loss functions are used than (V)AEs. However, basic GANs lacks an efficient inference mechanism. On the other hand, even though GANs produces more natural-looking images, instabilities in optimization induce mode collapse problem. To alleviate the above-mentioned issues with the GANs, recently, some research works have introduced to use the feature learning which includes features from the latent space and data space for improving GANs. The main idea to employ feature learning is that features from different spaces are complementary for producing realistic images. Combining the strengths and weaknesses of both (V)AE and GANs approach have provided a promising research direction for the unsupervised, supervised and reinforcement learning.

The combination of two architectures (AEs and GANs), i.e., encoder-decoder architecture, supports numerous benefits, such as can be used to reconstruct data (i.e., inpainting [111][112]), can be used for representation learning. AE-GANs can be primarily grouped into two methodologies: (1) combining AEs and GANs as data space AEs to learn a mapping from the data to the latent space and back to the data space; (2) combining AEs and GANs as latent space AEs, i.e., autoencoding the latent/noise space. In this section, we shall discuss both encoder-decoder architectures in detail.

5.3.1. Data space autoencoders

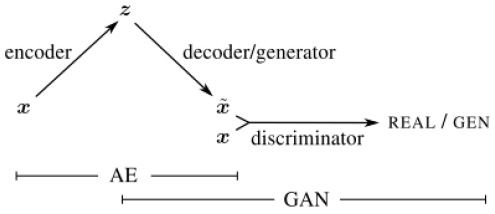


Figure 14. Overview of VAE-GAN. VAE is combined with GANs by collapsing Dec and G. Figure from [78]

Larsen, et al. [78] trained VAE and GANs jointly, called VAE-GAN that uses learned representations to measure similarities in data space as element-wise metrics do not generate high resolution and realistic images (see Figure 14). In VAE-GAN, VAE decoder and the GAN's G are merged and trained jointly. The replacement of element-wise metric with feature-wise metric generates better image samples. VAE-GAN uses the combination of KL divergence and reconstruction loss (i.e., distance measure) for training the

inference model. For this, optimization via backpropagation requires exact form of prior distribution and re-parameterization tricks. As GANs updates G with the reconstruction loss, GANs can successfully handle the mode collapse issue as G will able to reconstruct every input x. Both VAE and GANs cannot be applied for unsupervised conditional generation tasks as both cannot find disentangled mappings.

[79] introduced an adversarial autoencoder (AAE) that uses the GANs for performing variational inference. Matching the aggregated posterior to the prior guarantees that samples generated from prior space will be more realistic. In AAE, encoder (E) is used to map the data distribution to the prior distribution while decoder (Dec) is used to learn a deep generative model for mapping the imposed prior to the data distribution. Both adversarial and autoencoder network in AAE are trained together in two parts: *reconstruction phase* in which AE updates E and Dec for minimizing the reconstruction error of the inputs, and *regularization phase* in which adversarial network updates its discriminative network to discriminate real and fake data. This regularization does not guarantee that G will have capability to approximate data distribution accurately and handle the mode missing problem. Further, Mescheder, et al. [80] proposed a new training technique, called Adversarial Variational Bayes (AVB), to train VAEs adversarially for having the flexible inference model. Another line of research is the new adversarial learning for VAE: joint distribution of data and codes [81]. One possible solution is to feed observed data through E to produce codes while another solution is to draw latent codes from a prior and propagate through Dec to manifest data.

[46] introduced a mode regularized GANs (MDGAN) related to VAE-GAN [78] in terms of training an VAE jointly with the GANs model. However, VAE in VAE-GAN is used to generate samples whereas MDGAN's autoencoder based losses used as a regularizer to penalize missing modes which improves GAN's training stability and sample qualities. For getting more stable gradients, MDGAN used distance between real data and reconstructed data as a regularizer. MDGAN uses two Ds, one to discriminate between data and reconstructions (i.e., manifold) for learning a good AE and one to distinguish between two distributions, i.e., diffusion. Authors also proposed a set of evaluation metrics for evaluating diversity of modes and distribution fairness of the probability mass. Training of MDGAN is composed of two steps, a mode regularization step to reduce the model's variance and a diffusion step to reduce instability. However, MDGAN uses an additional autoencoder as well as a two-step training procedure which can be computationally expensive.

Tran, et al. [82] proposed two novel distance constraints for improving the G's training, called Dist-GAN, by an AE. First, latent-data distance constraint to impose compatibility between latent sample distances and the corresponding data sample distances so that G does not generate samples close to each other. Second, D-score distance constraint to align the distribution of fake data with the real data so that G can generate data similar to real ones. Dist-GAN is applicable to any prior distribution as it limits AE by the data and latent sample distances. Dist-GAN using only a D network while MDGAN [46] requires two Ds. To alleviate the mode collapse, VAE-GAN [78] considers the reconstructed data as fake, MDGAN also uses this similarly in the manifold step, while Dist-GAN employs them as real data, which is crucial to control D for avoiding gradient vanishing. Moreover, both VAE-GAN and MDGAN use reconstruction loss regularization for G, while Dist-GAN uses additional regularization for AE.

Furthermore, in GANs, the effective likelihood is not known and intractable, and GANs are based on density ratio estimation [47], [104], [133], [134] which provides a tool to overcome intractable distributions. One possible solution to handle the intractability of the marginal likelihood issue is not to compute it ever, and learn about the model parameters via a tool indirectly [31]. To implement this solution, Rosca, et al. [31] introduced α-GANs using the variational inference for training in which the synthetic likelihood is used instead of intractable likelihood function of basic GANs and an implicit function used instead of the unknown posterior distribution. This study combines variational lower bound on the data likelihood with the density ratio trick to understand the VAEs and GANs connection. The use of the density ratio allows GANs for the marginal likelihood intractability through its relative behavior w.r.to the real distribution. This trick is very useful to deal with implicit distributions or likelihood-free models as it only needs data from the two distributions without accessing to their analytical forms. Moreover, α-GANs uses adversarial loss with a data reconstruction loss to handle both the issue of samples blurriness and mode collapse.

Unlike α-GANs, VAE-GAN applies the analytical KL loss for minimizing the distance between prior and posterior of the latent and does not elaborate its connection to density ratio estimation. In addition, α-GANs is end-to-end unsupervised model which maximizes a lower bound on the real data likelihood and do not use any pre-trained classifier or a feature matching loss.

5.3.2. Latent space autoencoders

Data space autoencoders autoencode data points in which a reconstruction loss is calculated on input and encoded images. But selecting appropriate loss function is a challenging task. Otherwise, combining autoencoder with adversarial learning could be enough for better image generation. While selecting a loss function for autoencoder on noise vector *z* is an easy task as *z* are drawn from a standard normal distribution. In addition, the biggest question in GANs is "can GANs be used for unsupervised learning of rich feature representations for random data distributions?". The possible reason is that G supports the mapping of latent space to generated data, i.e., an inverse mapping from data to latent representation is not supported by GANs. Inverse mapping is extremely important as it presents an information-rich representation of *x*, that can be employed as input for downstream tasks (such as, classification) [83][84]. To handle this issue, a new autoencoder-type adversarial architecture is required which supports unsupervised learning with both generation and inference.

Unlike prior AEs and adversarial networks hybrids, it is required to setup the adversarial game directly between the E and G with no trained external mappings in the process of learning. To handle the above-mentioned issues, another class of methods proposed in which methods learn to map latent space to the data space and backward. The main advantage of these methods over data space autoencoders is that the noise vectors can be reconstructed easily as the distribution from which they are chosen is known.

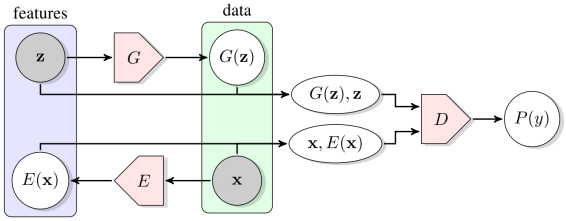


Figure 15. Bidirectional Generative Adversarial Networks (BiGAN) [84]**.**

Dumoulin, et al. [83] introduced an approach similar to adversarial autoencoder, called Adversarially Learned Inference (ALI), in which inference is performed by starting an adversarial game between E and Dec/G through a D that works on *x* and *z* space. D, Dec/G, and E are parametric deep (multi-layer) network. In ALI, both G and E are trained together, and D is trained to discriminate between two joint distributions over image and latent spaces produced either by E on the real data or by G to the latent prior. Donahue, et al. [84] proposed the same model as ALI, called Bidirectional GANs (BiGAN), and explored the benefits of learned features for supervised and semi-supervised tasks. In addition, unlike adversarial autoencoder, ALI does not optimize explicit reconstruction loss and D receives joint pairs of samples (*x*, *z*) rather than *z* samples (see Figure 15). ALI and BiGAN learn bidirectional mapping between the data and latent space while basic GANs learns a unidirectional mapping from latent to data space. In this case, gradients should propagate from D to E and Dec which can be achieved through reparameterization trick [3][135][136]. For the sampling, a random variable is computed as a deterministic transformation of some *z* such that its distribution is the desired distribution. As gradient propagation into the E and Dec depend on the reparametrization trick, ALI and BiGAN are not directly applicable to either applications with discrete data or to models with discrete latent variables. ALI and BiGAN can match joint distributions of Dec and E and performs inference by sampling from E's conditional that also matches the Dec's posterior. However, achieving equilibrium of the jointly adversarial game is difficult as the dependency structure between data and codes is not explicitly specified [85]. Therefore, inference in ALI is not always effective. Unlike ALI, decomposed adversarial learned inference (DALI) [85] breaks the problem of matching the joint distributions into two sub-problems -

matching priors on the latent codes and conditionals on the data explicitly. Due to this constraint, DALI gets better generation capability.

On the other hand, most of the applications require to handle multiple source of information, i.e., multiple views. For example, in traffic prediction, multiple views, such as traffic data, weather data, Point-of-Interest data, etc., are needed for efficient prediction. Multi-view machine learning is a long-studied problem where most of the existing studies have focused on the classification viewpoint and assume that all the views are present all the time. To handle this issue, Chen, et al. [86] proposed an extended architecture of BiGAN, called conditional-views BiGAN (CV-BiGAN) which supports to model a conditional distribution P(y|.). Authors proposed another model on the top of the CV-BiGAN architecture, called multi-view BiGANs (MV-BiGAN) which can predict in case of one or few views availability and updates its prediction if new views are available. For the stable MV-BiGAN model, authors have also proposed a regularization term to add new views to existing views for controlling the uncertainty over the outputs. Belghazi, et al. [87] attempts to extend ALI [83], named HALI, by achieving better perceptual matching in the reconstructions, and by being capable to reduce the observables using a sequence of composed features maps. Unlike VAE-GAN, HALI offers more meaningful reconstructions with different levels of fidelity and minimizes perceptual reconstruction error implicitly during adversarial training. [88] proposed a novel autoencoder-type architecture, called Adversarial Generator-Encoder (AGE) Network which comprises two parametric mappings: E and G. E maps data x to latent space z while G maps z to x. E and G game is useful in matching distributions. In AGE, E encodes data into codes taken from the prior. During the time of encoding, Dec tries to generate samples where encoded samples will match to the prior distribution. As E and Dec do not depend on D, it reduces the computational complexity and the converge time in comparison to the previous models utilizing a bidirectional mapping.

[89] has shown that latent space-based encoding, which is learned independently of the generated data, does not support good quality reconstruction and enhances the chance of mode collapse in the generated data space. To address this, authors proposed a Variational Encoder Enhancement to GANs, called VEEGAN, to map both the real and generated data space to a fixed distribution in a variational framework. VEEGAN proposed to use an additional reconstructor network (Rn) where G and Rn are trained jointly using an implicit variational principle to encourage Rn for mapping the data distribution to a Gaussian and for approximately inverting G's act. As a result, this forces G to map from the noise distribution to the true data distribution. Like VEEGAN, [137] also tried to enforce diversity in latent space. [137] used the last layer of D as a feature map for studying real and fake data distribution. Authors used the Bures distance between covariance matrices in feature space for matching real and fake batch diversity. The results show that matching the diversity handles the issue of mode collapse and also generate good sample quality.

Unlike MDGAN [46], VEEGAN uses the reconstruction loss in the latent domain instead of the data domain and generates good quality images. Also, MDGAN and VEEGAN improve inference mapping through this reconstruction loss in compared to ALI [83] and BiGAN. Even though, these methods perform well in compared to data space autoencoders, still they do not support the learning of disentangled relation between latent space and data as data is inverted to a semantically useless fixed noise distribution. To handle the issue of bidirectional mapping, Bang, et al. [90] introduced a naïve weakly bidirectional mapping approach, called manifold guided generative adversarial network (MGGAN), in which a guidance network is induced with the basic GANs framework for learning all modes of data distribution. In MGGAN, E and G are not connected, i.e., they are trained separately. This separation is assumed effective w.r.to performance as both can focus on their own objectives. In MGGAN, bidirectional mapping is used to regularize G training so that G cannot fool D by producing the same or similar samples corresponding to a single major mode of true data distribution. [83][84][46][89] also proposed encoder based architecture to support the bidirectional mapping and mapped p_{data} into low dimensional manifold space. Even though network architecture is similar to MGGAN, existing bidirectional mapping is designed to map p_{data} into p_z (i.e., inference mapping), while MGGAN maps p_{data} onto meaningful manifold space; called manifold mapping. Moreover, authors claimed

that MGGAN learns the meaningful landscape in latent space; therefore, MGGAN does not overfit the training data.

5.4. Improved Discriminator

To handle the bad gradients problem during the GANs training, D is replaced with autoencoder where D assigns low energy to training data while high energy to samples generated by G. G and D objectives are generalized to consider real-valued "energies" as input instead of probabilities [91].

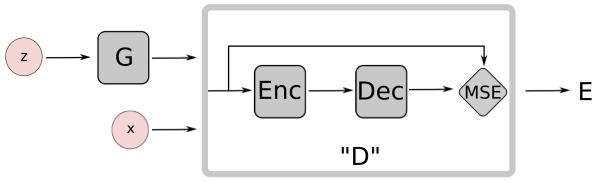


Figure 16. EBGAN architecture which replaces D with an autoencoder. Figure from [91]

Some researchers proposed to project GANs as energy-based model in which a function is built to map [138] each point of an input space to a single scalar, called energy. In energy-based models, energy surface is modelled like preferred patterns get assigned low energies and undesirable patterns are getting assigned high energies. Zhao, et al. [91] presented an Energy-based GANs (EBGAN) in which an autoencoder is used for D and implemented an energy function on D to assign low energies to regions near the data manifold and higher energies to other regions (see Figure 16). Moreover, to stabilize the EBGAN training, a hinge loss is used as objective function so that D can remove synthetic samples with energy over a margin *m*. Obtaining an appropriate value of *m* is a vital factor for effective training and it highly depends on both architecture selection and data complexity. On the other hand, EBGAN has proposed to reduce total variation (TV) distance between the real and generated data distributions. TV distance has same regularity as JSD. Therefore, EBGAN can also have the same problems as basic GANs, not able to train D till optimality which causes bad gradients. On a contrary, to restrict D from degenerating to uniform prediction, [139] proposed a regularized EBGAN with entropy loss to generate many possible outputs from the same conditional input. Berthelot, et al. [92] extended EBGAN, called BEGAN by introducing a loss function matching the fake data's energy to a fraction of the true data's energy.

Then, Wang, et al. [93] proposed a new robust training procedure, called Margin Adaptation for Generative Adversarial Networks (MAGANs), to maintain the equilibrium between D and G through hinge loss margin using the expected energy of the target distribution. EBGANs, MAGANs and BEGANs take care of the expected energy of real and fake data for generating realistic samples and for controlling training. But, BEGANs is more stable, easy to train and robust to hyperparameter variations. Unlike EBGANs, MAGANs do not use any new hyperparameter and eliminate the dependence on the margin hyperparameter. Convergence of MAGAN to global optima is easier than both EBGANs and BEGANs. Moreover, MAGAN converges to its global optimum when distribution of real and generated data match exactly, while BEGANs provide no such guarantee. In addition, to handle mode collapse and support stable training, [94] introduced explicit manifold learning as prior for GANs in which a manifold preserving reconstruction loss is used during G's training. Reconstruction loss is one of the efficient ways to guarantee not missing information in GANs. Even EBGAN replaced D loss with reconstruction loss for showing other energy functions performance. A new target of Minimum Manifold Coding is further enforced for manifold learning to find simple and unfolded manifolds which works even in the case of the sparsely or unevenly distributed data.

Another line of research is to improve generative power of G when a huge size of training data is unavailable. In this case, existing energy-based models cannot perform as GANs requires to learn enough features for unsupervised learning. To handle above-mentioned issue, Deepearthgo, et al. [95] proposed to train a G by learning two mapping functions simultaneously, called Max-Boost-GAN. This improves the power of G without expanding network's size. Besides this, cost of GANs does not increase. In addition, to get better gradients when G is distant from convergence, authors have introduced a margin loss in GANs objective.

5.5. Memory Networks

One of the architecture solutions introduced memory within the network to handle the mode collapse and instability problem in unsupervised GANs framework.

[96] discussed two main causes of instability in the unsupervised GANs framework, namely structural discontinuity problem and forgetting problem. First, basic GANs use a unimodal continuous latent space (e.g. Gaussian distribution). Due to this, it cannot handle structural discontinuity between different classes and suffers from the mode collapse problem. E.g., in basic GANs, both building and cats are embedded into a common continuous latent distribution while both do not share any intermediate structure. Second, D forgets the past generated samples by G during the training as D often focuses only on latest input images. This forgetting behavior causes instability as loss functions of G and D computed on the basis of each other's performance.

To handle the aforementioned problems, [96] introduced a memory network for the unsupervised GANs training, called MemoryGAN. To handle the structure discontinuity problem, memory network learns representation of training samples and helps G to better understand the underlying class. The forgetting problem can be handled by learning to memorize distribution of the data generated by G, including rare ones. MemoryGAN introduces a life-long memory into D to increase the model's memorization capacity explicitly instead of adding regularization terms [128] or modifying GANs algorithms [89]. In addition, MemoryGAN is quite similar to one of the unsupervised GANs frameworks, InfoGAN [101], but InfoGAN learns latent cluster information of data into small-sized model parameters implicitly.

The objective of MemoryGAN depends on InfoGAN, where MemoryGAN also adds a mutual information loss between K_i and $G(z, K_i)$. This loss computes structural similarity between the sampled memory information and generated sample. $\hat{I}$ is the expectation of negative cosine similarity.

$$L_D = -\mathbb{E}_{x\sim p_{data}}[\log D(x)] + \mathbb{E}_{(z,c)\sim p_{(z,c)}}\left[\log\left(1 - D\left(G(z, K_i)\right)\right)\right] + \lambda\hat{I}$$

$$L_G = \mathbb{E}_{(z,c)\sim p_{(z,c)}}\left[\log\left(1 - D\left(G(z, K_i)\right)\right)\right] + \lambda\hat{I}$$

5.6. Latent Space Engineering

Existing works to handle the issues of GANs focused on the use of a fixed latent distribution, either a unimodal Normal [4][6][35][83] or a factored multimodal distribution [101][140] with uniform mode priors. But still, these distributional selections for the latent space leads a dissimilarity between the modes of generated and real data distributions (has a non-uniform mode prior). This case is possible when every data mode denotes a class and class distribution is imbalanced [98]. Therefore, for better performance, some researchers have proposed approaches for learning a better distribution of noise.

Gurumurthy, et al. [97] proposed an architecture to handle the mode collapse problem in the case of diverse and limited training data, called DeLiGAN. DeLiGAN reparametrizes the latent generative space as a mixture model and learns the parameters of mixture model along with GANs. The primary aim of this work is to increase modeling power of the prior distribution rather than increasing the model depth. Mishra, et al. [98] introduced to construct the latent space (additive noise) in such a way that it matches to its target distribution w.r.to number of modes and modal mass or mode priors, called NEMGAN. Authors provided a solution for "*How to learn the latent space distribution in a GANs framework such that modal properties of the true and the generated space are matched?*". Furthermore, [99] introduced to use the multiplicative noise instead of additive noise for better visual quality and diversity of generated features, and explored for the unsupervised learning of features.

In DCGANs, z is sampled from a Gaussian distribution and then G maps whole normal distribution to the images. Due to this, DCGANs cannot reflect the inherent structure of the training data. To handle this issue, Zhong, et al. [100] proposed a Decoder-Encoder structure by transforming the original Gaussian noise z to an informative one z'. Authors have shown the analysis that proposed solution accelerate the training process and enhance the quality of generated images. Authors also proposed hidden-space loss function to make

model robust. On the other hand, some researchers proposed that the noise vector z used in GANs formulation can be used by G in any manner. Due to this, it is feasible that z can be utilized in highly entangled way which can cause individual dimension of z to not correspond to semantic features of the data. Therefore, z can be divided into two parts instead of the use of a single unstructured noise vector [101]. First, z as a source of incompressible noise and second, c is a latent code to target the salient structured semantic features of the data distribution. This is a simple modification to GANs objective to learn interpretable and disentangled representations. Authors have also claimed that InfoGAN is able to disentangle both discrete and continuous latent factors without any extra training time. Further, [101] proposed an information-theoretic regularization to maximize mutual information between a small subset of the z and the G's distribution, called Information Maximizing Generative Adversarial Networks (InfoGAN). The unsupervised disentangled representation learned by InfoGAN outperforms existing supervised label information works [142][143]–[145][79]. In addition, a latent-code reconstruction based penalty is used in the cost function. Like VEEGAN [89], InfoGAN uses an AE over the latent codes, while in InfoGAN, reconstructor network are not trained on the true data distribution. Unlike VEEGAN, in InfoGAN, a latent code part is reconstructed. Unlike VEEGAN, InfoGAN requires some stabilization tricks as basic GANs.

6. NEW LOSS FUNCTION

Learning in implicit generative models, or likelihood-free models comprises of two steps: *comparison* and *estimation*. The comparison step uses the density ratio or difference estimators, while in estimation step, parameters of generative models are learned. For learning, two sets of samples drawn from the true data distribution and the model distribution are compared, this process is known as *density estimation-by-comparison*. To compare the true data distribution $p(x)$ with model distribution $q(x)$, two methods are used density difference $p(x)$-$q(x)$) and the density ratio $p(x)/q(x)$. The density ratio estimation can be further categorized into three general approaches: *class-probability estimation*, *divergence minimisation* and *ratio matching*, while density difference estimation involves moment matching. The density ratio trick is widely used [4][133][79], [80], [146], [147][148] where *class probability estimation* is the most popular approach.

Basic GANs is based on the class probability estimation where density ratio can be computed by building a classifier to distinguish observed data from that generated by the model. Density ratio includes samples only from two distributions, therefore, makes it suitable for handling with implicit distributions or likelihood-free models. Divergence minimization minimizes the divergence between the $p(x)$ and $q(x)$ and uses it as an objective to drive learning in the generative model. Ratio matching directly minimises the error between the true density ratio and an estimate of it. Moment matching compares the moments of the two distributions by minimising their distance. Several works have been introduced based on the divergence minimisation [103], ratio matching [104], and moment matching [52].

Statistical divergences, such as KL divergence, Reverse-KL divergence and JSD, measure the difference between two given probability distributions and belong to f-divergence class of probability distance metrics (used in basic GANs [4], f-GAN [103], b-GAN [104]). Integral Probability Metrics (IPMs) is another class which involves Wasserstein distance (used in WGAN, WGAN-GP [35][113]) and Maximum Mean Discrepancy (MMD) (used in MMDGAN [52]). The difference between f-divergences and IPMs is that f-divergences computes distance using $p(x)/q(x)$, i.e., density ratio, while IPMs uses the difference, $p(x) - q(x)$, i.e., density difference.

Several methods have been proposed for improving the quality of gradients and providing additional supervision to G and D. New probability distance and divergence are introduced to handle the issue of *vanishing gradients* and non-convergence in training. In addition, some techniques have been proposed to introduce different kinds of regularization on weights or gradients for stable training, including gradient regularization, spectral-norm regularization [49], etc.

In this section, we shall discuss more stable alternatives of the objective functions for G and D and objective function with the regularization in detail.

6.1. New Probability Distance and Divergence

In this sub-section, we shall discuss popular probability distances and divergences used in the context of learning distributions to improve the GANs training stability and mode collapse problem.

(a) (b)

Figure 17. ρ(Pθ, P0) as a function of θ, (a) ρ is the EM distance (b) ρ is the JSD. The EM plot is continuous and gives a usable gradient everywhere, whereas JS plot is not continuous and does not give a usable gradient. Figure from [35]**.**

Arjovsky, et al. [35] proposed a loss function which also acts as a measure of convergence, called Wasserstein Generative Adversarial Networks (WGAN). WGAN has non-zero gradients everywhere and the implementation includes removing the sigmoid function in the objective and adding weight clipping to the D's network. To accelerate the GANs training and make the training process stable, WGAN introduced to reduce the JSD with an efficient approximation of Earth-Mover (EM) distance [149] as shown in Figure 17. EM distance is continuous and differentiable. This new metric is effective in solving mode collapse by stabilizing GANs training. WGAN allows to train the critic till optimality which supports the equilibrium between G and D. A well-trained critic provides high quality gradients which are used to train G. However, WGAN can be unstable when gradients of the loss function are large. Thus, too large weights are clipped after each SGD update. Even though WGAN supports stability and better mode coverage, it suffers from slow training. Moreover, tuning weight clipping and hyperparameters is a tedious task.

[102] discussed that assuming model should have infinite capacity in basic GANs causes training issues as it constrains a model to lie in Lipschitz continuous function space. In the case of WGAN, Lipschitz condition is from the Kantorovich-Rubinstein duality and only the critic is constrained. Loss sensitive GAN (LS-GAN) [102] used a weight-decay regularization technique so that weights of a model to lie in a bounded area to guarantee the Lipschitz function condition. Like WGAN, LS-GAN also uses a Lipschitz constraint in which it assumes that the density of real samples is Lipschitz continuous, i.e., nearby data do not suddenly change.

Guo, et al. [37] proposed a new class of statistical divergence, Relaxed Wasserstein (RW) divergence for the large-scale computations. RW divergence is the Wasserstein divergence parametrized by the class of strictly convex and differentiable functions containing different curvature information. RWGANs are robust and converge faster than WGAN. [103] introduced f-GAN to minimize the variational estimate of f-divergence where it formulates training D as a density ratio estimation (like GANs [4]). The main objective of f-GAN is to minimize the f-divergence between the true data distribution p(x) and the model distribution q(x). In addition, [104] proposed to use density ratio estimation based on the Bregman divergence, called b-GAN. The objective of b-GAN is the direct estimation of the true density ratio without estimating p(x) and q(x) independently. Basically, GANs are trained using a distribution discrepancy measure, such as information-theoretic divergences, integral probability metrics (IPM), and Hilbert space discrepancy metrics. Tao, et al. [105] studied these training metrics in connection and introduced a novel metric, called $\chi 2$- GAN, for stable training and reduced mode collapse.

In Original GANs, D uses a sigmoid cross entropy loss for differentiating real samples from the fake samples. If D classifies a generated sample as real, then G will stop updating even though generated sample is far from the real data distribution. In this case, sigmoid cross entropy loss will not push generated samples towards real data distribution as classification role is done. To handle this issue, Mao, et al. [40] proposed the Least Squares Generative Adversarial Networks (LSGANs) that uses least squares loss function (l2 loss) instead of sigmoid cross entropy loss to stabilize the learning process and to reduce the possibility of mode collapse. For handling vanishing gradients problem, samples are penalized on the basis of their distances to decision boundary to generate more gradients to update G. Authors have also shown that minimizing loss function of

LSGANs minimizes the Pearson $\chi 2$ divergence. However, LSGANs could not achieve good performance for generating diverse images with real datasets. [107] introduced to use Softmax cross-entropy loss, called Softmax GANs, instead of the classification loss in the basic GANs. In basic GANs, D uses a logistic loss which saturates quickly, and its gradient vanishes if it is simple to differentiate between real samples and generated samples. As gradient vanishes, G stops updating. In Softmax GANs, gradient is always non-zero unless the Softmax distribution matches the target distribution. In addition, Saliman, et al. [106] introduced a GANs variant, called Optimal Transport GANs (OT-GAN), for minimizing a newly proposed metric, mini-batch energy distance to compute the distance between the G's generated and real data distribution. However, OT-GAN suffers from large amounts of computation and memory.

Traditional GANs is related to Noise Contrastive Estimation (NCE) in which a binary classification task is defined between true and noise samples with a logistic loss, while Softmax GANs is related to Importance Sampling version of GANs in which Importance Sampling replaces the logistic loss with a multi-class Softmax and cross-entropy loss. Furthermore, [108] introduced to exploit the features learned by D to handle the issue of mode collapse and instability. A reconstruction loss is added with GANs objective function in which real data features generated by D is fed into G for real data generation. Authors claimed that in order to enhance the generated sample quality, proposed reconstruction loss can be applied with other regularization loss function, such as gradient penalty. [109] proposed a GANs training approach in which D is trained using a maximum margin formulation to improve the D's capability, i.e., a better G.

On the other hand, previous GANs training approaches have employed gradient descent on the cost of each player simultaneously in which gradient descent enters a stable orbit instead of converging to the desired equilibrium point [150]. Feature matching can handle GANs training stability by having G's new objective as it avoids GANs from overtraining on current D [42]. The new objective allows G to produce data equivalent to the statistics of the real data instead of directly maximizing the output of D. D is used only to state the statistics that are worth matching. Feature matching is similar to techniques that use *maximum mean discrepancy* [151]–[153] for G's network training [128][129]. Several feature matching based GANs have been proposed with an aim to improve convergence.

Salimans, et al. [42] proposed three techniques for fast convergence of GANs game, called IGAN: *feature matching*, *minibatch features* and *virtual batch normalization* (VBN). A new objective function in feature matching does not work on directly boosting D's output, while it involves G to generate data that matching statistics of the real data and D is used only to identify the statistics worth matching. Feature matching's objective performs well for classification but could not generate indistinguishable samples. In addition, the idea of minibatch features is equivalent to batch normalization [156] while VBN is a direct extension of batch normalization. Minibatch discrimination allows D to process the correlations between training data points in one batch, rather than handling them separately. Minibatch discrimination works well for generating realistic images, but it could not predict labels accurately. Moreover, minibatch discrimination is computationally complex and highly sensitive to the selection of hyperparameters [43]. VBN can handle this issue, but it has high computational complexity as it involves running forward propagation on two minibatches of data which can be used only in G network. In addition, IGAN uses the same network architecture as DCGAN, but training of IGAN is more advanced.

IGAN works only for semi-supervised learning, so for unsupervised learning, [47] proposed to attach G's training criterion with a second training objective which directs G in the direction of data by explicitly modeling data density in addition to D. The second training objective is calculated in the space of features learned by D to make it computationally inexpensive by not having separate convolutional network for it. Then, in that space, denoising autoencoder is trained to estimate energy gradient of data on which it is trained.

The parametrization used in WGAN suffers from *l1* mean feature matching problem. To handle this issue, [110] extended the concept of WGAN by matching second order moment feature through the singular value decomposition concept. [110] used the mean feature matching and covariance matching GANs (McGAN) for stable training and reduced mode collapse and used the Integral Probability Metrics (IPM) loss as it

correlates with the generated samples' quality. McGAN also maximizes an embedding covariance discrepancy between real samples and generated samples. McGAN matches second order moment from the primal-dual norm perspective which requires matrix (tensor) decompositions because of exact moment matching [157] and hard to scale to higher order moment matching. In contrast, Maximum Mean Discrepancy GANs (MMDGAN) [52] can match high-order moments with kernel tricks by giving up exact moment matching. MMD distance is derived from the Generative moment matching networks (GMMN) [154] and is an alternative to Wasserstein distance. MMDGAN uses the adversarial kernel learning techniques instead of Gaussian kernels which can better represent the feature space. But still, the computational complexity of MMDGAN increases as number of sample increases. [111] introduced a new manifold-matching GANs (MMGAN) to stabilize GANs training by discovering two manifolds for the vector representations of real and fake images in which real and fake images are statistically identical, if they are equal. To match the two manifolds, authors have proposed a loss function to train G. In addition, for enhancing diversity in generated samples, authors have proposed kernel tricks for getting better manifold structures, moving-averaged manifolds across mini-batches, and a correlation matrix based regularizer.

Bellemare, et al. [112] has pointed that WGAN suffers from the biased gradients. As a solution, they proposed an energy function without the biased gradients, called CramerGAN, where Cramer distance is related to the kernel embedded space distance of MMDGAN.

6.2. Regularization

In this sub-section, we shall discuss regularization strategies proposed to stabilize GAN training.

[113] proved that the usage of weight clipping in WGAN for imposing a Lipschitz constraint on the critic generates lower-quality samples or fails to converge. Authors proposed another solution to WGAN's weight clipping which is to penalize the norm of gradient of the critic w.r.to its input, called Wasserstein Generative Adversarial Networks – gradient penalty (WGAN-GP). Further, [36] extended WGAN-GP concept to any separable complete normed space, called Banach Wasserstein GANs (BWGAN). The main difference is that the l2 norm is replaced with a dual norm. In addition, authors generalized the WGAN-GP theory to Banach spaces to allow features selection for G.

On the other hand, [114] have present a theoretical claim that BWGAN is not good for GANs training and then propose a less restrictive regularization. While, [115] have shown that gradient penalty in WGAN-GP cannot support stable training as GP often does not check the continuity of region near the real data. To handle this issue, authors proposed training in WGAN-GP which explicitly checks the continuity condition using two perturbed version of x, near any observed real data point x.

[116] have shown that in WGAN-GP, weight clipping reduces the rank of the weight matrix which reduces the features used by D to distinguish the distributions. To handle this issue, authors proposed spectral normalization technique which does not affect the rank of the weight matrix and stabilize the training of D networks. Authors have an analysis that spectral normalization for GANs outperforms other regularization techniques, such as weight normalization [158], weight clipping [35], and gradient penalty [113].

A geometric mismatch is not beneficial for f-GAN as the resulting f-divergence is not finite [48]. As a solution [143][144], it is suggested to use an alternative family of distance functions, IPMs which includes Wasserstein distance and RKHS-induced maximum mean discrepancies. On the other hand, [117] has proposed noise-induced regularization scheme to make f-GAN models robust against dimensional misspecifications. [118] proposed a GANs variant for stable training which builds on the IPM framework and normalizes the critic with its second moment, like $\chi 2$-GAN. Fisher GANs relies on a more sophisticated augmented Lagrangian to optimize the same objective for both the critic and G, while $\chi 2$-GAN decouples the critic and G objectives, requiring simpler (unconstrained) SGD type updates. Fisher GANs reduce the distance between two distributions as well as in-class variance. Moreover, it does not add any weight clipping or gradient penalty.

On the other hand, some researchers introduced to add regularization term to objective function for easily converging the objective function to the global optima.

[38] revisited the basic GANs algorithm for finding the Nash-equilibrium and proposed to add a regularization term to the loss function for handling the non-convergence of simultaneous gradient ascent based on the Jacobian of the gradients for both D and G. Authors have identified theoretically, key elements responsible for the failure of the SGD to achieve Nash-equilibria. Then, authors introduced a new design for the GANs training on the basis of insights driven from the SGD failure analysis. Further, to address the convergence issue in the training of GANs, [45] introduced a surrogate objective for G's update which unrolls D to regularize its updates. But, computational cost of unrolling step is high, which makes it unsuitable for the large-scale datasets. Authors also present two techniques, inference via optimization, and pairwise distance distributions for generating diverse samples.

[119] and [120] introduced a regularized penalty for G and D, respectively, to achieve better convergence in the GANs algorithm. [121] analyzed the convergence of GANs training for finding the main reason(s) of the mode collapse. Then proposed gradient penalty approach, called DRAGAN to avoid the local equilibria which causes sharp D's gradients nearby some real data points.

Further discussion. The selection of appropriate distance measure plays a vital role in the training of generative networks. The major difference between these distances is the impact on the convergence of sequences of probability distributions. As discussed earlier, several GANs variants have been proposed for minimizing the probability distances/divergences between real and generated data distribution, such as JSD, f-divergence, maximum mean discrepancy (MMD) and Wasserstein distance. Some approaches also used probability measure for distance calculation. Cramer distance, a family of IPMs, calculates the distance between probability distributions of generated and real data in D's activations. Divergences based strategies try to ensure the existence of Nash-equilibria.

In most of the works, objective functions are derived from the Wasserstein distance instead of f-divergences for improving the stability in the GANs training. Wasserstein distance does not support the theoretical guarantee, but it is computationally efficient. In addition, moment matching based approaches have handled the issue of vanishing gradients, but these approaches are computationally expensive than training a classifier. While, the use of gradient penalty [113] and spectral normalization [116] are beneficial in the high-capacity architectures. Several solutions have been proposed to regularize the GANs for handling the issue of vanishing gradients and non-convergence where most of the works are theoretically formulated. Gradient-norm based regularization enhance local stability. However, regardless of the progress, stable GANs training is an open challenge which requires a careful balance during the adversarial optimization.

7. ALTERNATIVE OPTIMIZATION ALGORITHM

Previous approaches in GANs used gradient descent techniques which mainly focus on finding low value of cost function instead of Nash equilibrium of a game. These algorithms may fail to converge when find the Nash equilibrium [133]. Introducing a stable algorithm for GANs training is a long-awaited problem and numerous solutions have been introduced. In this section, we shall discuss optimization approaches introduced to address the issues in GANs using Simultaneous Gradient Descent.

[38] analyzed the main reasons associated with simultaneous gradient ascent algorithm failure in finding local Nash-equilibria of smooth games. Authors have shown through theoretical analysis that the main factors are the presence of eigenvalues of the Jacobian of the associated gradient vector field with zero real-part and eigenvalues with a large imaginary part. The second case makes saddle-point problems more challenging than local optimization problems. To handle these problems, [38] proposed a robust algorithm using consensus optimization to find Nash-equilibria of smooth two-player games. The proposed algorithm is orthogonal to other strategies making the GANs framework better, such as using new distances, or regularization. These solutions assume the existence of Nash-equilibria while proposed algorithm works with their computation and the numerical difficulties occurring in practice. The proposed solution is an

approximation to the implicit Euler method for integrating the gradient vector field where implicit Euler method has strong stability properties [161] which can be translated into convergence theorems for local Nash-equilibria. However, in implicit Euler method, solution of a nonlinear equation in each iteration is required. Moreover, it can be extended by having better approximations to the implicit Euler method. Alternatively, the proposed work can be seen as second order method where it can be extended by revisiting second order optimization methods [162] in the context of saddle point problems. Authors claimed that it is a first step to understand main factors of GANs training and different objective functions. The main aim of the work is to stabilize GANs training on different architectures and divergence functions. The results show that the proposed algorithm achieves stability in the training and handles the issue of mode collapse successfully. However, stability suffers in the case of deeper architectures as gradients can have different scales in such architectures. Also, if regularization parameter set to high, proposed method can make unstable stationary points of the gradient vector field stable and may lead to poor solutions.

Simultaneous Gradient Descent is similar to use no-regret dynamics for each player where in game theory, it is assumed that this limits the oscillatory behavior in zero sum games. In the case of convex-concave zero-sum games, average of the weights of the two players establishes an equilibrium and not the last-iterate. But, average the neural nets weights is not allowed as zero-sum game defined by training two deep networks against each other is not a convex-concave zero-sum game. Therefore, it is important to get training algorithms making the last iterate of the training be very close to the equilibrium, instead of only the average. [122] proposed to use a variant of gradient descent, called Optimistic Mirror Descent (OMD) for WGAN training which achieves faster convergence rate to equilibrium. Gradient Descent (GD) dynamics are bound to cycle while the last iterate of OMD dynamics converges to an equilibrium. OMD considers that in zero-sum game, another player is also training through similar algorithm; OMD predicts the strategy of another player to achieve faster regret rates. Results show that OMD gets smaller KL divergence w.r.to the true underlying distribution compared to GD variants. Experiments have also shown that GD suffers from limit cycles even tested in a simple distribution learning setting (learning the mean of a multi-variate distribution). In addition, OMD converges pointwise. an optimistic variant of Adam is also introduced which shows better performance than Adam.

In recent times, actor-critic learning has been applied for stochastic approximation. [123] used the two time-scale update rule (TTUR) which guarantees that training achieves a local Nash equilibrium when the critic learns faster than the actor. [39] used the same approach and train the GANs by TTUR to reach the local Nash equilibrium where D and G both have different learning rates. The main idea is that D converges to a local minimum when G is fixed. If changes in G are quite slow, still D still converges, since G perturbations are small. In addition, performance also enhances as D must first learn new patterns before transferring to G. On the other hand, Adam stochastic approximation is used to handle the issue of mode collapse where Adam stochastic optimization is introduced as a heavy ball with friction (HBF) dynamics in which Adam try to get flat minima and avoids small local minima. TTUR enhances learning of DCGANs and WGAN-GP and outperforms basic GANs training for various image datasets.

GANs is difficult to train as optimal weights of the loss functions in GANs relate to saddle points, and not minimizers. Alternative SGD methods used for zero-sum games either do not reliably converge to saddle points or if converges, they are highly sensitive to learning rates. Standard alternating SGD methods generally moves between minimization and maximization steps. It is possible that minimization step overpower maximization step where iterates will “slide off” the edge of saddle and it will lead to instability. To handle this issue, [41] proposed modified stochastic gradient descent, called *prediction step* to stabilize adversarial networks in which maximization step can exploit information about minimization step. Authors have also shown that proposed algorithm converges to saddle points and is stable with a wider range of training parameters than plain SGD methods. This supports faster training with larger learning rates. A theoretical proof is also provided to show that prediction step is asymptotically stable for solving saddle point problems.

8. DISCUSSION AND RESEARCH DIRECTIONS

According to the proposed taxonomy (in Table 1), we can see that the rapidly growing body of research focused on GANs now comprises several novel solutions for handling the challenges of GANs. Here, we look at what could be explored in future research to further improve the existing works.

8.1. GANs Design and Optimization Solutions

GANs has shown its potential for generating natural images, still it goes through the problem of mode collapse as discussed earlier, i.e., G collapses and could capture only limited varieties of modes in the data. GANs commonly fails to learn some of the modes trained on the multi-modal distribution data. Most of the time, GANs could not converge to true equilibrium and settles for sub-optimal local solutions. Due to mode collapse, samples produced by the trained generative model often lack diversity. In addition, we can say that mode collapse is connected to instable GANs training which leads to another main challenge in GANs.

Despite the promising achievements, GANs is still hard to train due to several common problematic unstable training and convergence behaviors, such as vanishing gradients, mode collapse, and diverging or oscillatory behavior. These issues in GANs training often hinders further research and applicability in this area. In recent times, to alleviate the above-mentioned issues, existing studies have proposed several solutions, such as designing more stable network architectures, modifying the learning objectives, regularize objectives, training strategies, tuning hyperparameters, etc. However, in most of the cases, their achievement is often the result of sacrificing the image quality and image diversity where these issues have trade-off relationships.

Most of the existing works have focused either on image quality or on image diversity. One possible research direction can be to work on image quality without suffering from the low image diversity. Furthermore, to handle the issue of GANs training instability, existing methods still depend on heuristics which are very sensitive to amendments. This is one of the main reasons that restricts applicability of these approaches in new domains. On the other hand, we observe that most of the existing works have proposed to solve only one training issue at a time and often do not include theoretical analysis. Another important research direction is to have a theoretical framework/analysis for handling issues in GANs training process with the aim of exploring more tractable formulations and to make training stable and straightforward.

In addition, proposed solutions differ in the training improvement scale. Most of the models can achieve similar results by improving the hyperparameters settings and computational resources. Therefore, we can say that majority of related works mainly emphasized on achieving state-of-the-art accuracy instead of state-of-the-art efficiency. Developing solutions having the algorithmic improvements over existing works can be a future direction.

To handle the GANs challenges, existing approaches are focused on three major directions: re-engineered network architecture, new objective functions and optimization algorithms as discussed in Section 5, 6 and 7. Objective function GANs variants often show more improvement in training than architectural GANs but still they cannot boost mode diversity in the generated visual samples. Several GANs design and optimization solutions have been proposed in these three directions where proper selection of architecture, objective functions and optimization techniques have improved GANs training stability. In addition, objective functions are sensitive to the use of optimization strategies, hyperparameter selection, and number of training steps which can be explored in the future research for the different GANs.

On the other hand, research on using other technologies for improving the GANs training, such as online learning [163], game theory variants, etc., is still at the early stage. A combination of proper architecture, loss function and optimization techniques can prove to yield superior results and can be a future research direction to explore.

8.2. Applications

Generation of realistic images has broad range of practical applications, such as face aging, face editing, pose estimation, entertainment, etc. Recently, GANs has gained momentum for generating naturalistic images

through adversarial training. Apart from it, GANs has also shown its potential for several other applications, such as Steganalysis [164], information retrieval [165], spatio-temporal data prediction, such as transportation [166], autonomous driving [167]–[169], speech enhancement [170], single-cell RNA-sequence imputation [171], etc. Exploring GANs applicability for new application domains can be a future direction. Moreover, the support of GANs for maintaining ethics in AI is new research direction.

8.3. Evaluation Metrics

Generative models were evaluated based on the model likelihood which is intractable. Basic GANs used a proxy for log-likelihood, i.e., Parzen window estimate for evaluation [172]. Theis, et al. [30] presented that likelihood estimation is often incorrect for Parzen window estimate. GANs lacks the meaningful measure for evaluating the GANs output. Due to this issue, it is very challenging to compare different GANs variants and still based on the visual assessment of generated images.

Furthermore, because of the lack of robust and consistent metrics, it is difficult to evaluate which GANs algorithm(s) outperforms other algorithms. A good evaluation is needed as it will allow to choose appropriate algorithm from a very large set. Also, to have better algorithms and their understanding, like which modifications are critical, and which algorithms cannot make a significant difference in practice [172]. To overcome above-mentioned issues, researchers have proposed several evaluation methods for GANs [154][155]. Moreover, different applications choose different evaluation metrics as different applications require different trade-offs between different measures. It is better to know a combination of training and evaluation metrics for the target application.

9. SUMMARY AND CONCLUSION

Recently, GANs has gained significant attention for generating realistic images and has become important in modern world applications, such as image generation, domain adaptation, etc. However, GANs is hard to train and training faces two main challenges, mode collapse, non-convergence and stability. The possible solutions to handle these GANs challenges are to design an efficient model by choosing appropriate network architecture, by using suitable objective function or by selecting proper optimization techniques. Within these solutions, many different GANs variants have already been proposed with diverse characteristics, but still some issues remained unsolved.

Research on GANs is quite broad and several designing and training solutions of GANs handling these challenges lay ahead. In this paper, we recapitulate the basic GANs framework and survey the developments of solutions for better design and optimization of GANs. More concretely, we proposed a novel taxonomy of GANs design and optimization techniques based on re-engineered network architectures, new objective functions and alternative optimization algorithms and discussed how existing works deal with these challenges. We mapped the existing works to the taxonomy for finding the research gaps in GANs. Our work provides a panorama of current progress and an in-depth analysis of the reviewed methods to serve both novices and experienced researchers. In addition, the new taxonomy aims to build a problem-solution structure with a hope to suggest a guideline when readers are selecting their research topics or developing their approaches. Based on insights gained, we proposed promising directions that the researchers can pursue in the future.

ACKNOWLEDGMENT
This work is supported by RGC Collaborative Research Fund (No.: C5026-18G) and RGC Research Impact Fund (No.: R5060-19).

References

[1] Y.-W. Hinton, Geoffrey E and Osindero, Simon and Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527--1554, 2006.
[2] R. Salakhutdinov and G. Hinton, "Deep Boltzmann machines," in *Journal of Machine Learning Research*, 2009, vol. 5, pp.

448–455.
[3] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” in *2nd International Conference on Learning Representations, ICLR*, 2014.
[4] I. J. Goodfellow *et al.*, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems*, 2014, vol. 3, no. January, pp. 2672–2680.
[5] E. Denton, S. Chintala, A. Szlam, and R. Fergus, “Deep generative image models using a laplacian pyramid of adversarial networks,” in *Advances in Neural Information Processing Systems*, 2015, pp. 1486–1494.
[6] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” in *4th International Conference on Learning Representations, ICLR*, 2016.
[7] M. Y. Liu and O. Tuzel, “Coupled generative adversarial networks,” in *Advances in Neural Information Processing Systems*, 2016, pp. 469–477.
[8] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of GANs for improved quality, stability, and variation,” in *6th International Conference on Learning Representations, ICLR*, 2018.
[9] T. Karras, S. Laine, and T. Aila, “A Style-Based Generator Architecture for Generative Adversarial Networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410.
[10] C. Ledig *et al.*, “Photo-realistic single image super-resolution using a generative adversarial network,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2017, pp. 105–114.
[11] C. Spampinato, S. Palazzo, P. D’Oro, D. Giordano, and M. Shah, “Adversarial Framework for Unsupervised Learning of Motion Dynamics in Videos,” *Int. J. Comput. Vis.*, Mar. 2019.
[12] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, “Learning to discover cross-domain relations with generative adversarial networks,” in *34th International Conference on Machine Learning, ICML 2017*, 2017, vol. 4, pp. 2941–2949.
[13] Y. Hong, U. Hwang, J. Yoo, and S. Yoon, “How generative adversarial networks and their variants work: An overview,” *ACM Computing Surveys*, vol. 52, no. 1. 2019.
[14] Z. Wang, Q. She, and T. E. Ward, “Generative Adversarial Networks: A Survey and Taxonomy,” *arXiv Prepr. arXiv1906.01529*, 2019.
[15] Z. Pan, W. Yu, X. Yi, A. Khan, F. Yuan, and Y. Zheng, “Recent Progress on Generative Adversarial Networks (GANs): A Survey,” *IEEE Access*, vol. 7, pp. 36322–36333, 2019.
[16] S. Hitawala, “Comparative Study on Generative Adversarial Networks,” *arXiv Prepr. arXiv1801.04271*, 2018.
[17] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, and F. Y. Wang, “Generative adversarial networks: Introduction and outlook,” *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 4. pp. 588–598, 2017.
[18] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, “A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications,” vol. 14, no. 8, pp. 1–28, 2020.
[19] A. Bissoto, E. Valle, and S. Avila, “The Six Fronts of the Generative Adversarial Networks,” *arXiv Prepr. arXiv1910.13076*, 2019.
[20] T. Motwani and M. Parmar, “A Novel Framework for Selection of GANs for an Application,” *arXiv Prepr. arXiv2002.08641*, 2020.
[21] Y.-J. CAO, “Recent Advances of Generative Adversarial Networks in Computer Vision,” *IEEE Access*, vol. 7, 2019.
[22] A. Jabbar, X. Li, and B. Omar, “A Survey on Generative Adversarial Networks: Variants, Applications, and Training,” *arXiv*, Jun. 2020.
[23] L. Jin, F. Tan, and S. Jiang, “Generative Adversarial Network Technologies and Applications in Computer Vision,” 2020.
[24] M. Wiatrak, S. V. Albrecht, and A. Nystrom, “Stabilizing Generative Adversarial Networks: A Survey,” Sep. 2019.
[25] M. Lee and J. Seok, “Regularization Methods for Generative Adversarial Networks: An Overview of Recent Studies,” *arXiv*, May 2020.
[26] S. N. Esfahani and S. Latifi, “A Survey of State-of-the-Art GAN-based Approaches to Image Synthesis,” in *CS & IT-CSCP*, 2019, pp. 63–76.
[27] N. Akhtar and A. Mian, “Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey,” *IEEE Access*, vol. 6. pp. 14410–14430, 2018.
[28] A. Chakraborty, A. Manaar, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, “Adversarial Attacks and Defences: A Survey,” *arXiv Prepr. arXiv1810.00069*, 2018.
[29] F. Huszár, “How (not) to Train your Generative Model: Scheduled Sampling, Likelihood, Adversary?,” *arXiv Prepr. arXiv1511.05101*, Nov. 2015.
[30] L. Theis, A. Van Den Oord, and M. Bethge, “A note on the evaluation of generative models,” in *4th International Conference on Learning Representations, ICLR*, 2016.
[31] M. Rosca, B. Lakshminarayanan, D. Warde-Farley, and S. Mohamed, “Variational Approaches for Auto-Encoding Generative Adversarial Networks,” *arXiv Prepr. arXiv1706.04987*, 2017.
[32] A. Statnikov, C. F. Aliferis, D. P. Hardin, and I. Guyon, “Support Vector Clustering,” in *A Gentle Introduction to Support Vector Machines in Biomedicine*, vol. 2, 2011, pp. 136–153.
[33] T. Le, H. Vu, T. D. Nguyen, and D. Phung, “Geometric enclosing networks,” in *International Joint Conference on Artificial Intelligence IJCAI*, 2018, pp. 2355–2361.
[34] M. Mirza and S. Osindero, “Conditional Generative Adversarial Nets,” *arXiv Prepr. arXiv1411.1784*, Nov. 2014.
[35] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *International conference on machine learning*, 2017, pp. 214–223.
[36] J. Adler and S. Lunz, “Banach Wasserstein GaN,” in *Advances in Neural Information Processing Systems*, 2018, vol. 2018-Decem, pp. 6754–6763.
[37] X. Guo, J. Hong, T. Lin, and N. Yang, “Relaxed Wasserstein with Applications to GANs,” *arXiv Prepr. arXiv1705.07164*, May 2017.
[38] L. Mescheder, S. Nowozin, and A. Geiger, “The numerics of GANs,” in *Advances in Neural Information Processing Systems*, 2017, vol. 2017-Decem, pp. 1826–1836.
[39] 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 *Advances in Neural Information Processing Systems*, 2017, pp. 6627–6638.

[40] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "Least Squares Generative Adversarial Networks," *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 2813–2821, 2017.
[41] A. Yadav, S. Shah, Z. Xu, D. Jacobs, and T. Goldstein, "Stabilizing Adversarial Nets With Prediction Methods," *arXiv Prepr. arXiv1705.07364*, 2018.
[42] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Advances in Neural Information Processing Systems*, 2016, pp. 2234–2242.
[43] Z. Lin, G. Fanti, A. Khetan, and S. Oh, "PacGan: The power of two samples in generative adversarial networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 1498–1507.
[44] I. Goodfellow, "Tutorial: Generative Adversarial Networks," *arXiv Prepr. arXiv1701.00160*, 2016.
[45] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, "Unrolled Generative Adversarial Networks," in *arXiv Preprint arXiv:1611.02163*, 2016.
[46] T. Che, Y. Li, A. P. Jacob, Y. Bengio, and W. Li, "Mode regularized generative adversarial networks," in *5th International Conference on Learning Representations, ICLR*, 2019.
[47] D. Warde-Farley and Y. Bengio, "Improving generative adversarial networks with denoising feature matching," in *5th International Conference on Learning Representations, ICLR*, 2019.
[48] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," in *5th International Conference on Learning Representations, ICLR*, 2019.
[49] S. Huang, Xun and Li, Yixuan and Poursaeed, Omid and Hopcroft, John and Belongie, "Stacked Generative Adversarial Networks," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 5077–5086, 2017.
[50] Y. Wang, L. Zhang, and J. van de Weijer, "Ensembles of Generative Adversarial Networks," *arXiv Prepr. arXiv1612.00991*, 2016.
[51] I. Tolstikhin, S. Gelly, O. Bousquet, C. J. Simon-Gabriel, and B. Schölkopf, "AdaGAN: Boosting generative models," in *Advances in Neural Information Processing Systems*, 2017, pp. 5425–5434.
[52] C. L. Li, W. C. Chang, Y. Cheng, Y. Yang, and B. Póczos, "MMD GAN: Towards deeper understanding of moment matching network," in *Advances in Neural Information Processing Systems*, 2017, vol. 2017-Decem, pp. 2204–2214.
[53] H. Kwak and B.-T. Zhang, "Ways of Conditioning Generative Adversarial Networks," *arXiv Prepr. arXiv1611.01455*, Nov. 2016.
[54] D. J. Im, C. D. Kim, H. Jiang, and R. Memisevic, "Generating images with recurrent adversarial networks," *arXiv Prepr. arXiv1602.05110*, Feb. 2016.
[55] G. Perarnau, J. van de Weijer, B. Raducanu, and J. M. Álvarez, "Invertible Conditional GANs for image editing," *arXiv Prepr. arXiv1611.06355*, Nov. 2016.
[56] A. Jaiswal, W. AbdAlmageed, Y. Wu, and P. Natarajan, "Bidirectional Conditional Generative Adversarial Networks," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11363 LNCS, pp. 216–232, Nov. 2019.
[57] G. Máttyus and R. Urtasun, "Matching Adversarial Networks," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 8024–8032, 2018.
[58] S. Liu, T. Wang, D. Bau, J.-Y. Zhu, and A. Torralba, "Diverse Image Generation via Self-Conditioned GANs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14274–14283.
[59] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in *34th International Conference on Machine Learning, ICML*, 2017, vol. 6, pp. 4043–4055.
[60] L. Chongxuan, T. Xu, J. Zhu, and B. Zhang, "Triple generative adversarial nets," *Adv. Neural Inf. Process. Syst.*, pp. 4088--4098, 2017.
[61] X. Wang, Y. Sun, R. Zhang, and J. Qi, "KDGAN: Knowledge distillation with generative adversarial networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 775–786.
[62] M. Lee and J. Seok, "Controllable generative adversarial network," *IEEE Access*, vol. 7, pp. 28158–28169, Aug. 2019.
[63] Y. Saatchi and A. G. Wilson, "Bayesian GAN," in *Advances in Neural Information Processing Systems*, 2017, pp. 3623–3632.
[64] A. Jaiswal, W. AbdAlmageed, Y. Wu, and P. Natarajan, "CapsuleGAN: Generative adversarial capsule network," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2019, vol. 11131 LNCS, pp. 526–535.
[65] S. Lloyd and C. Weedbrook, "Quantum Generative Adversarial Learning," *Phys. Rev. Lett.*, vol. 121, no. 4, Apr. 2018.
[66] A. Zhang, Han and Goodfellow, Ian and Metaxas, Dimitris and Odena, "Self-Attention Generative Adversarial Networks," *arXiv Prepr. arXiv1805.08318*, 2018.
[67] A. Ghosh, V. Kulharia, V. Namboodiri, P. H. S. Torr, and P. K. Dokania, "Multi-agent Diverse Generative Adversarial Networks," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8513–8521.
[68] Q. Hoang, T. D. Nguyen, T. Le, and D. Phung, "Multi-Generator Generative Adversarial Nets," *arXiv Prepr. arXiv1708.02556*, Aug. 2017.
[69] A. Ghosh, V. Kulharia, and V. Namboodiri, "Message Passing Multi-Agent GANs," *arXiv Prepr. arXiv1612.01294*, Dec. 2016.
[70] H. Ge, Y. Xia, X. Chen, R. Berry, and Y. Wu, "Fictitious GAN: Training GANs with Historical Models," Mar. 2018.
[71] S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang, "Generalization and equilibrium in generative adversarial nets (GANs)," in *34th International Conference on Machine Learning, ICML*, 2017, vol. 1, pp. 322–349.
[72] T. D. Nguyen, T. Le, H. Vu, and D. Phung, "Dual discriminator generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2017, pp. 2671–2681.
[73] I. Durugkar, I. Gemp, and S. Mahadevan, "Generative multi-adversarial networks," in *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 2019.
[74] B. Neyshabur, S. Bhojanapalli, and A. Chakrabarti, "Stabilizing GAN Training with Multiple Random Projections," *arXiv Prepr. arXiv1705.07831*, May 2017.
[75] G. Mordido, H. Yang, and C. Meinel, "Dropout-GAN: Learning from a Dynamic Ensemble of Discriminator," in *arXiv*

preprint arXiv:1807.11346, 2018.
[76] G. Mordido, H. Yang, and C. Meinel, "MicrobatchGAN: Stimulating diversity with multi-adversarial discrimination," in *Proceedings - 2020 IEEE Winter Conference on Applications of Computer Vision, WACV 2020*, 2020, pp. 3050–3059.
[77] T. Chavdarova and F. Fleuret, "SGAN: An Alternative Training of Generative Adversarial Networks," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9407–9415.
[78] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," *33rd Int. Conf. Mach. Learn. ICML*, vol. 4, pp. 2341–2349, Dec. 2016.
[79] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial Autoencoders," *arXiv Prepr. arXiv1511.05644*, 2015.
[80] L. Mescheder, S. Nowozin, and A. Geiger, "Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks," *34th Int. Conf. Mach. Learn. ICML 2017*, vol. 5, pp. 3694–3707, 2017.
[81] Y. Pu *et al.*, "Adversarial symmetric variational autoencoder," *Adv. Neural Inf. Process. Syst.*, pp. 4331–4340, 2017.
[82] N. T. Tran, T. A. Bui, and N. M. Cheung, "Dist-gan: An improved gan using distance constraints," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11218 LNCS, pp. 387–401, 2018.
[83] V. Dumoulin *et al.*, "Adversarially learned inference," in *5th International Conference on Learning Representations, ICLR*, 2019.
[84] J. Donahue, T. Darrell, and P. Krähenbühl, "Adversarial feature learning," in *arXiv preprint arXiv:1605.09782*, 2016.
[85] A. H. Li, Y. Wang, C. Chen, and J. Gao, "Decomposed Adversarial Learned Inference," *arXiv Prepr. arXiv2004.10267*, 2020.
[86] M. Chen and L. Denoyer, "Multi-view Generative Adversarial Networks," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2017, vol. 10535 LNAI, pp. 175–188.
[87] M. I. Belghazi, S. Rajeswar, O. Mastropietro, N. Rostamzadeh, J. Mitrovic, and A. Courville, "Hierarchical Adversarially Learned Inference," *arXiv Prepr. arXiv1802.01071*, 2018.
[88] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "It takes (only) two: Adversarial generator-encoder networks," *32nd AAAI Conf. Artif. Intell. AAAI*, pp. 1250–1257, 2018.
[89] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton, "VEEGAN: Reducing mode collapse in GANs using implicit variational learning," *Adv. Neural Inf. Process. Syst.*, pp. 3309–3319, May 2017.
[90] D. Bang and H. Shim, "MGGAN: Solving Mode Collapse using Manifold Guided Training," *arXiv Prepr. arXiv1804.04391*, Apr. 2018.
[91] J. Zhao, M. Mathieu, and Y. LeCun, "ENERGY-BASED GENERATIVE ADVERSARIAL NETWORKS," in *arXiv preprint arXiv:1609.03126*, 2016.
[92] D. Berthelot, T. Schumm, and L. Metz, "BEGAN: Boundary Equilibrium Generative Adversarial Networks," *arXiv Prepr. arXiv1703.10717*, 2017.
[93] R. Wang, A. Cully, H. J. Chang, and Y. Demiris, "MAGAN: Margin Adaptation for Generative Adversarial Networks," *arXiv Prepr. arXiv1704.03817*, Apr. 2017.
[94] G. Zheng, J. Sang, and C. Xu, "MMCGAN: Generative Adversarial Network with Explicit Manifold Prior," *arXiv Prepr. arXiv2006.10331*, 2020.
[95] X. Di and P. Yu, "Max-Boost-GAN: Max Operation to Boost Generative Ability of Generative Adversarial Networks," in *Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW*, 2017, pp. 1156–1164.
[96] Y. Kim, M. Kim, and G. Kim, "Memorization Precedes Generation: Learning Unsupervised GANs with Memory Networks," Mar. 2018.
[97] S. Gurumurthy, R. K. Sarvadevabhatla, and R. V. Babu, "DeLiGAN : Generative adversarial networks for diverse and limited data," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, pp. 4941–4949.
[98] D. Mishra, P. A. P., A. Jayendran, V. Srivastava, and S. Chaudhury, "NEMGAN: Noise Engineered Mode-matching GAN," *arXiv Prepr. arXiv1811.03692*, Nov. 2018.
[99] X. Di and P. Yu, "Multiplicative Noise Channel in Generative Adversarial Networks," in *Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW*, 2017, pp. 1165–1172.
[100] G. Zhong, W. Gao, Y. Liu, and Y. Yang, "Generative Adversarial Networks with Decoder-Encoder Output Noise," *arXiv Prepr. arXiv1807.03923*, Jul. 2018.
[101] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets," in *Advances in neural information processing systems*, 2016, pp. 2172–2180.
[102] G.-J. Qi, "Loss-Sensitive Generative Adversarial Networks on Lipschitz Densities," *Int. J. Comput. Vis.*, vol. 128, no. 5, pp. 1118–1140, Jan. 2017.
[103] S. Nowozin, B. Cseke, and R. Tomioka, "f-GAN: Training generative neural samplers using variational divergence minimization," in *Advances in Neural Information Processing Systems*, 2016, pp. 271–279.
[104] M. Uehara, I. Sato, M. Suzuki, K. Nakayama, and Y. Matsuo, "Generative Adversarial Nets from a Density Ratio Estimation Perspective," *arXiv Prepr. arXiv1610.02920*, Oct. 2016.
[105] C. Tao, L. Chen, R. Henao, J. Feng, and L. Carin, "X2 generative adversarial network," in *35th International Conference on Machine Learning, ICML*, 2018, vol. 11, pp. 7787–7796.
[106] T. Salimans, D. Metaxas, H. Zhang, and A. Radford, "Improving GANs using optimal transport," in *6th International Conference on Learning Representations, ICLR*, 2018.
[107] M. Lin, "Softmax GAN," *arXiv Prepr. arXiv1704.06191*, Apr. 2017.
[108] Y. Li, N. Xiao, and W. Ouyang, "Improved generative adversarial networks with reconstruction loss," *Neurocomputing*, vol. 323, pp. 363–372, Jan. 2019.
[109] F. Juefei-Xu, V. N. Boddeti, and M. Savvides, "Gang of GANs: Generative Adversarial Networks with Maximum Margin Ranking," *arXiv Prepr. arXiv1704.04865*, Apr. 2017.
[110] Y. Mroueh, T. Sercu, and V. Goel, "McGan: Mean and covariance feature matching GAN," in *34th International*

Conference on Machine Learning, ICML, 2017, vol. 5, pp. 3885–3899.
[111] N. Park *et al.*, "MMGAN: Manifold-Matching Generative Adversarial Networks," in *Proceedings - International Conference on Pattern Recognition*, 2018, pp. 1343–1348.
[112] M. G. Bellemare *et al.*, "The Cramer Distance as a Solution to Biased Wasserstein Gradients," *arXiv Prepr. arXiv1705.10743*, May 2017.
[113] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein GANs," in *Advances in Neural Information Processing Systems*, 2017, vol. 2017-Decem, pp. 5768–5778.
[114] H. Petzka, A. Fischer, and D. Lukovnikov, "On the regularization of Wasserstein GANs," in *6th International Conference on Learning Representations, ICLR*, 2018.
[115] X. Wei, B. Gong, Z. Liu, W. Lu, and L. Wang, "Improving the Improved Training of Wasserstein GANs: A Consistency Term and Its Dual Effect," *arXiv Prepr. arXiv1803.01541*, Mar. 2018.
[116] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *6th International Conference on Learning Representations, ICLR*, 2018.
[117] K. Roth, A. Lucchi, S. Nowozin, and T. Hofmann, "Stabilizing training of generative adversarial networks through regularization," in *Advances in Neural Information Processing Systems*, 2017, vol. 2017-Decem, pp. 2019–2029.
[118] Y. Mroueh and T. Sercu, "Fisher GAN," in *Advances in Neural Information Processing Systems*, 2017, pp. 2514–2524.
[119] V. Nagarajan and J. Z. Kolter, "Gradient descent GAN optimization is locally stable," in *Advances in Neural Information Processing Systems*, 2017, pp. 5586–5596.
[120] L. Mescheder, A. Geiger, and S. Nowozin, "Which training methods for GANs do actually converge?," in *35th International Conference on Machine Learning, ICML 2018*, 2018, vol. 8, pp. 5589–5626.
[121] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, "On Convergence and Stability of GANs," *arXiv Prepr. arXiv1705.07215*, May 2017.
[122] C. Daskalakis, A. Ilyas, V. Syrgkanis, and H. Zeng, "Training GaNs with optimism," *6th Int. Conf. Learn. Represent. ICLR 2018 - Conf. Track Proc.*, Oct. 2018.
[123] H. Prasad, P. LA, S. B.-P. of the 2015 International, and undefined 2015, "Two-timescale algorithms for learning Nash equilibria in general-sum stochastic games," *ifaamas.org*.
[124] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach, "Multimodal compact bilinear pooling for visual question answering and visual grounding," in *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, 2016, pp. 457–468.
[125] M. Mathieu, J. Zhao, P. Sprechmann, A. Ramesh, and Y. Le Cun, "Disentangling factors of variation in deep representations using adversarial training," in *Advances in Neural Information Processing Systems*, 2016, pp. 5047–5055.
[126] Z. Xu, Y. C. Hsu, and J. Huang, "Training shallow and thin networks for acceleration via knowledge distillation with conditional adversarial networks," in *6th International Conference on Learning Representations, ICLRW 2018*, 2018.
[127] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," *arXiv Prepr. arXiv1411.1784*, 2014.
[128] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved Techniques for Training GANs."
[129] Y. Freund and R. E. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.
[130] I. Albuquerque, J. Monteiro, T. Doan, B. Considine, T. Falk, and I. Mitliagkas, "Multi-objective training of Generative Adversarial Networks with multiple discriminators," *36th Int. Conf. Mach. Learn. ICML 2019*, vol. 2019-June, pp. 292–301, Jan. 2019.
[131] A. Nguyen, J. Clune, Y. Bengio, A. Dosovitskiy, and J. Yosinski, "Plug and play generative networks: Conditional iterative generation of images in latent space," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2017, vol. 2017-Janua, pp. 3510–3520.
[132] R. A. Yeh, C. Chen, T. Yian Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, "Semantic image inpainting with deep generative models," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2017, pp. 6882–6890.
[133] I. J. Goodfellow, "On distinguishability criteria for estimating generative models," in *3rd International Conference on Learning Representations, ICLRW*, 2015.
[134] S. Mohamed and B. Lakshminarayanan, "Learning in Implicit Generative Models," *arXiv Prepr. arXiv1610.03483*, Oct. 2016.
[135] Y. Bengio, N. Léonard, and A. Courville, "Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation," *arXiv Prepr. arXiv1308.3432*, Aug. 2013.
[136] Y. Bengio, É. Thibodeau-Laufer, G. Alain, and J. Yosinski, "Deep generative stochastic networks trainable by backprop," in *31st International Conference on Machine Learning, ICML*, 2014, vol. 2, pp. 1470–1485.
[137] H. De Meulemeester, J. Schreurs, M. Fanuel, B. De Moor, and J. A. K. Suykens, "The Bures Metric for Taming Mode Collapse in Generative Adversarial Networks," *arXiv Prepr. arXiv2006.09096*, 2020.
[138] Y. LeCun, Y. LeCun, S. Chopra, R. Hadsell, F. J. Huang, and E. Al., "A tutorial on energy-based learning," *Predict. Struct. Data*, vol. 1, no. 0, 2006.
[139] Z. Dai, A. Almahairi, P. Bachman, E. Hovy, and A. Courville, "Calibrating Energy-based Generative Adversarial Networks," *arXiv Prepr. arXiv1702.01691*, Feb. 2017.
[140] S. Mukherjee, H. Asnani, E. Lin, and S. Kannan, "ClusterGAN: Latent Space Clustering in Generative Adversarial Networks," *Proc. AAAI Conf. Artif. Intell.*, vol. 33, pp. 4610–4617, Sep. 2019.
[141] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, 2014.
[142] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, "Semi-supervised learning with deep generative models," in *Advances in Neural Information Processing Systems*, 2014, vol. 4, no. January, pp. 3581–3589.
[143] B. Cheung, J. A. Livezey, A. K. Bansal, and B. A. Olshausen, "Discovering hidden factors of variation in deep networks," in *3rd International Conference on Learning Representations, ICLRW*, 2015.
[144] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. B. Tenenbaum, "Deep convolutional inverse graphics network," in *Advances in Neural Information Processing Systems*, 2015, pp. 2539–2547.

[145] W. F. Whitney, M. Chang, T. Kulkarni, and J. B. Tenenbaum, “Understanding Visual Concepts with Continuation Learning,” *arXiv Prepr. arXiv1602.06822*, Feb. 2016.
[146] F. Huszár, “Variational Inference using Implicit Distributions,” *arXiv Prepr. arXiv1702.08235*, Feb. 2017.
[147] T. Karaletsos, “Adversarial Message Passing For Graphical Models,” *arXiv Prepr. arXiv1612.05048*, Dec. 2016.
[148] D. Tran, R. Ranganath, and D. M. Blei, “Hierarchical implicit models and likelihood-free variational inference,” in *Advances in Neural Information Processing Systems*, 2017, pp. 5524–5534.
[149] Y. Rubner, C. Tomasi, and L. J. Guibas, “Earth mover’s distance as a metric for image retrieval,” *Int. J. Comput. Vis.*, vol. 40, no. 2, pp. 99–121, Nov. 2000.
[150] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning: The MIT Press*, vol. 19. 2017.
[151] A. Gretton, O. Bousquet, A. Smola, and B. Sc̈hlkopf, “Measuring statistical dependence with Hilbert-Schmidt norms,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2005, vol. 3734 LNAI, pp. 63–77.
[152] K. Fukumizu, A. Gretton, X. Sun, and B. Schölkopf, “Kernel measures of conditional dependence,” in *Advances in Neural Information Processing Systems*, 2009.
[153] A. Smola, A. Gretton, L. Song, and B. Schölkopf, “A hilbert space embedding for distributions,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2007, vol. 4754 LNAI, pp. 13–31.
[154] Y. Li, K. Swersky, and R. Zemel, “Generative moment matching networks,” in *32nd International Conference on Machine Learning, ICML*, 2015, vol. 3, pp. 1718–1727.
[155] G. K. Dziugaite, D. M. Roy, and Z. Ghahramani, “Training generative neural networks via maximum mean discrepancy optimization,” in *Proceedings of the 31st Conference Uncertainty in Artificial Intelligence - UAI*, 2015, pp. 258–267.
[156] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *32nd International Conference on Machine Learning, ICML*, 2015, vol. 1, pp. 448–456.
[157] W. Zellinger, E. Lughofer, S. Saminger-Platz, T. Grubinger, and T. Natschläger, “Central moment discrepancy (CMD) for domain-invariant representation learning,” in *5th International Conference on Learning Representations, ICLR*, 2019.
[158] T. Salimans and D. P. Kingma, “Weight normalization: A simple reparameterization to accelerate training of deep neural networks,” in *Advances in Neural Information Processing Systems*, 2016, pp. 901–909.
[159] A. Müller, “Integral Probability Metrics and Their Generating Classes of Functions,” *Adv. Appl. Probab.*, vol. 29, no. 2, pp. 429–443, Jun. 1997.
[160] B. K. Sriperumbudur, K. Fukumizu, A. Gretton, B. Schölkopf, and G. R. G. Lanckriet, “On integral probability metrics, \phi-divergences and binary classification,” Jan. 2009.
[161] J. C. Butcher, *Numerical Methods for Ordinary Differential Equations*. Chichester, UK: John Wiley & Sons, Ltd, 2016.
[162] S. I. Amari, “Natural Gradient Works Efficiently in Learning,” *Neural Comput.*, vol. 10, no. 2, pp. 251–276, Feb. 1998.
[163] P. Grnarova, K. Y. Levy, A. Lucchi, T. Hofmann, and A. Krause, “An online learning approach to generative adversarial networks,” in *6th International Conference on Learning Representations, ICLR*, 2018.
[164] D. Volkhonskiy, I. Nazarov, and E. Burnaev, “Steganographic Generative Adversarial Networks,” *arXiv Prepr. arXiv1703.05502*, Mar. 2017.
[165] W. Zhang, “Generative adversarial nets for information retrieval: Fundamentals and advances,” in *41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR*, 2018, vol. 18, pp. 1375–1378.
[166] D. Saxena and J. Cao, “D-GAN: Deep Generative Adversarial Nets for Spatio-Temporal Prediction,” *arXiv Prepr. arXiv1907.08556*, Jul. 2019.
[167] M. Uricar, P. Krizek, D. Hurych, I. Sobh, S. Yogamani, and P. Denny, “Yes, we GAN: Applying adversarial techniques for autonomous driving,” *arXiv Prepr. arXiv1902.03442*, Feb. 2019.
[168] M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid, “DeepRoad: GAN-based Metamorphic Autonomous Driving System Testing,” in *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, 2018, pp. 132–142.
[169] M. Teichmann, M. Weber, M. Zöllner, R. Cipolla, and R. Urtasun, “MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving,” in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2018, pp. 1013–1020.
[170] S. Pascual, A. Bonafonte, and J. Serra, “SEGAN: Speech enhancement generative adversarial network,” in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2017, pp. 3642–3646.
[171] Y. Xu, Z. Zhang, L. You, J. Liu, Z. Fan, and X. Zhou, “scIGANs: single-cell RNA-seq imputation using generative adversarial networks,” *Nucleic Acids Res.*, vol. 48, no. 15, p. 85, Jun. 2020.
[172] M. Lucic, K. Kurach, M. Michalski, O. Bousquet, and S. Gelly, “Are Gans created equal? A large-scale study,” in *Advances in Neural Information Processing Systems*, 2018, pp. 700–709.
[173] Q. Xu *et al.*, “An empirical study on evaluation metrics of generative adversarial networks,” *arXiv Prepr. arXiv1806.07755*, Jun. 2018.
[174] A. Borji, “Pros and cons of GAN evaluation measures,” *Comput. Vis. Image Underst.*, vol. 179, pp. 41–65, 2019.