# Generative Adversarial Networks: A Survey and Taxonomy

Zhengwei Wang, Qi She, Tomás E. Ward



**Abstract**—

Generative adversarial networks (GANs) have been extensively studied in the past few years. Arguably the revolutionary techniques are in the area of computer vision such as plausible image generation, image to image translation, facial attribute manipulation and similar domains. Despite the significant success achieved in the computer vision field, applying GANs to real-world problems still poses significant challenges, three of which we focus on here: (1) High quality image generation; (2) Diverse image generation; and (3) Stable training. Through an in-depth review of GAN-related research in the literature, we provide an account of the architecture-variants and loss-variants, which have been proposed to handle these three challenges from two perspectives. We propose loss-variants and architecture-variants for classifying the most popular GANs, and discuss the potential improvements with focusing on these two aspects. While several reviews for GANs have been presented to date, none have focused on the review of GAN-variants based on their handling the challenges mentioned above. In this paper, we review and critically discuss 7 architecture-variant GANs and 9 loss-variant GANs for remedying those three challenges. The objective of this review is to provide an insight on the footprint that current GANs research focuses on the performance improvement. Code related to GAN-variants studied in this work is summarized on *https://github.com/sheqi/GAN_Review*.

**Index Terms**—Generative Adversarial Networks, Computer Vision, Architecture-variants, Loss-variants, Stable Training.

## 1 Introduction

Generative adversarial networks (GANs) are attracting growing interests in the deep learning community [1]–[6]. GANs have been applied to various domains such as computer vision [7]–[14], natural language processing [15]–[18], time series synthesis [19]–[23], semantic segmentation [24]–[28] etc. GANs belong to the family of generative models. Comparing to other generative models e.g., variational autoencoders, GANs offer advantages such as an ability to handle sharp estimated density functions, efficiently generating desired samples, eliminating deterministic bias and good compatibility with the internal neural architecture. These properties have allowed GANs to enjoy success especially in the computer vision field e.g., plausible image generation [29]–[33], image to image translation [2], [34]–[40], image super-resolution [26], [41]–[44] and image completion [45]–[49].

However, GANs suffer challenges from two aspects: (1) Hard to train — It is non-trivial for discriminator and generator to achieve Nash equilibrium during the training and the generator cannot learn the distribution of the full datasets well, which is known as mode collapse. Lots of work has been carried out in this area [50]–[53]; and (2) Hard to evaluate — the evaluation of GANs can be considered as an effort to measure the dissimilarity between the real distribution $p_r$ and the generated distribution $p_g$. Unfortunately, the accurate estimation of $p_r$ is not possible. Thus, it is challenging to produce good estimations of the correspondence between $p_r$ and $p_g$. Previous work has introduced evaluation metrics for GANs [54]–[62]. The first aspect concerns the performance for GANs directly e.g., image quality, image diversity and stable training. In this work. we are going to study existing GAN-variants that handle this aspect in the area of computer vision while those readers interested in the second aspect can consult [54], [62].

Current GANs research focuses on two directions: (1) Improving the training for GANs; and (2) Deployment of GANs to real-world applications. The former seeks to improve GANs performance and is therefore a foundation for the latter aspect. Considering numerous research work in the literature, we give a brief review on the GAN-variants that focus on improving training in this paper. The improvement of the training process provides benefits in terms of GANs performance as follows: (1) Improvements in generated image diversity (also known as mode diversity); (2) Increases in generated image quality; and (3) More stable training such as remedying the vanishing gradient for the generator. In order to improve the performance as mentioned above, modification for GANs can be done from either the architectural side or the loss perspective. We will study the GAN-variants coming from both sides that improve the performance for GANs. The rest of the paper is organized as follows: (1) We introduce the search strategy and part of the results (complete results are illustrated in Supplementary material) for the existing GANs papers in the area of computer vision; (2) We introduce related review work for GANs and illustrate the difference between those reviews and this work; (3) We give a brief introduction to GANs; (4) We review the architecture-variant GANs in the literature; (5) We review the loss-variant GANs in the literature; (6) We summarize the GAN-variants in this study

*Zhengwei Wang and Tomás E. Ward are with Insight Centre for Data Analytics, Dublin City University, Dublin, Ireland. e-mail: zhengwei.wang22@mail.dcu.ie, tomas.ward@dcu.ie*
*Qi She is with Intel Labs, Beijing, China. e-mail: qi.she@intel.com*

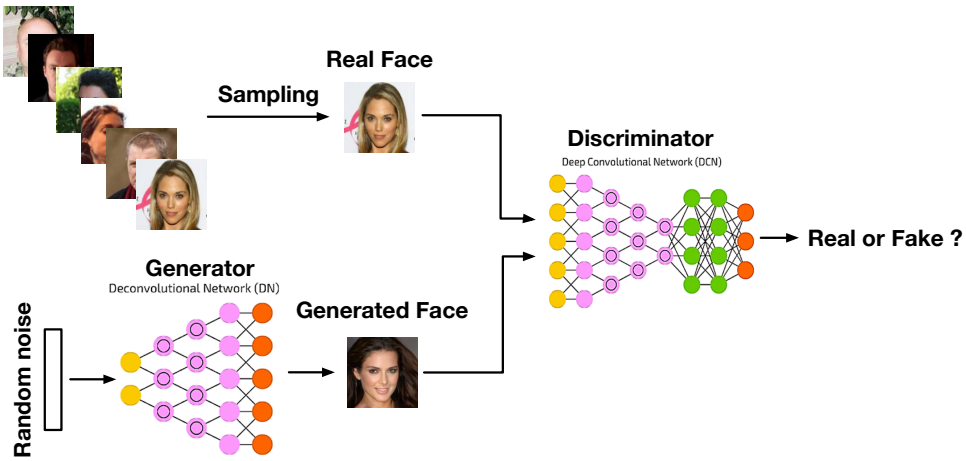



Fig. 1. Architecture of a GAN. Two deep neural networks (discriminator ($D$) and generator $G$)) are synchronously trained during the learning stage. The discriminator is optimized in order to distinguish between real images and generated images while the generator is trained for the sake of fooling the discriminator from discerning between real images and generated images.

and illustrate their difference and relationships; and (7) We conclude this review and preview likely future research work in the area of GANs.

Many GAN-variants have been proposed in the literature to improve performance. These can be divided into two types: (1) Architecture-variants. The first proposed GAN used fully-connected neural networks [1] so specific types of architecture may be beneficial for specific applications e.g., convolutional neural networks (CNNs) for images and recurrent neural networks (RNNs) for time series data; and (2) Loss-variants. Here different variations of the loss function are explored (1) to enable more stable learning of $G$.

## 2 Search Strategy and Results

A review of the literature was performed to identify research work describing GANs. Papers were first identified through manual search of the online datasets (Google Scholar and IEEE Xplore) through use of the keyword "generative adversarial networks". Secondly papers related to computer vision were manually selected. This search concluded 17th May 2019.

A total of 322 papers describing GANs related to computer vision were identified. The earliest paper was in 2014 [1]. Papers were classified into three categories regarding their repository, namely conference, arXiv and journal. More than half the papers were presented at conferences (204, 63.3%). The rest were journal articles (58, 18.0%) and arXiv pre-prints (60, 18.6%). Details of statistics and paper list are described in Supplementary material.

## 3 Related Work

There has been previous GANs review papers for example in terms of reviewing GANs performance [63]. That work focuses on the experimental validation across different types of GANs benchmarking on LSUN-BEDROOM [64], CELEBA-HQ-128 [65] and the CIFAR10 [66] image datasets. The results suggest that the original GAN [1] with spectral normalization [67] is a good starting choice when applying GANs to a new dataset. A limitation of that review is that the benchmark datasets do not consider diversity in a significant way. Thus the benchmark results tend to focus more on evaluation of the image quality, which may ignore GANs efficacy in producing diverse images. Work [68] surveys different GANs architectures and their evaluation metrics. A further comparison on different architecture-variants' performance, applications, complexity and so on needs to be explored. Papers [69]–[71] focus on the investigation of the newest development treads and the applications of GANs. They compare GAN-variants through different applications. Comparing this review to the current review literature, we emphasize an introduction to GAN-variants based on their performance including their ability to produce high quality and diverse images, stable training, ability for handling the vanishing gradient problem, etc. This is all done through the taking of a perspective based on architecture and loss function considerations. This work also provides the comparison and analysis in terms of pros and cons across GAN-variants presented in this paper.

## 4 Generative Adversarial Networks

Figure. 1 demonstrates the architecture of a typical GAN. The architecture comprises two components, one of which is a discriminator ($D$) distinguishing between real images and generated images while the other one is a generator ($G$) creating images to fool the discriminator. Given a distribution $\mathbf{z} \sim p_{\mathbf{z}}$, $G$ defines a probability distribution $p_g$ as the distribution of the samples $G(\mathbf{z})$. The objective of a GAN is to learn the generator's distribution $p_g$ that approximates the real data distribution $p_r$. Optimization of a GAN is performed with respect to a joint loss function for $D$ and $G$

$$\min_G \max_D \mathbb{E}_{\mathbf{x}\sim p_r} \log[D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}} \log\left[1 - D(G(\mathbf{z}))\right]. \quad (1)$$

GANs, as a member of the deep generative model (DGM) family, has attracted exponentially growing interest in the deep learning community because of some advantages comparing to the tradition DGMs: (1) GANs are able to produce

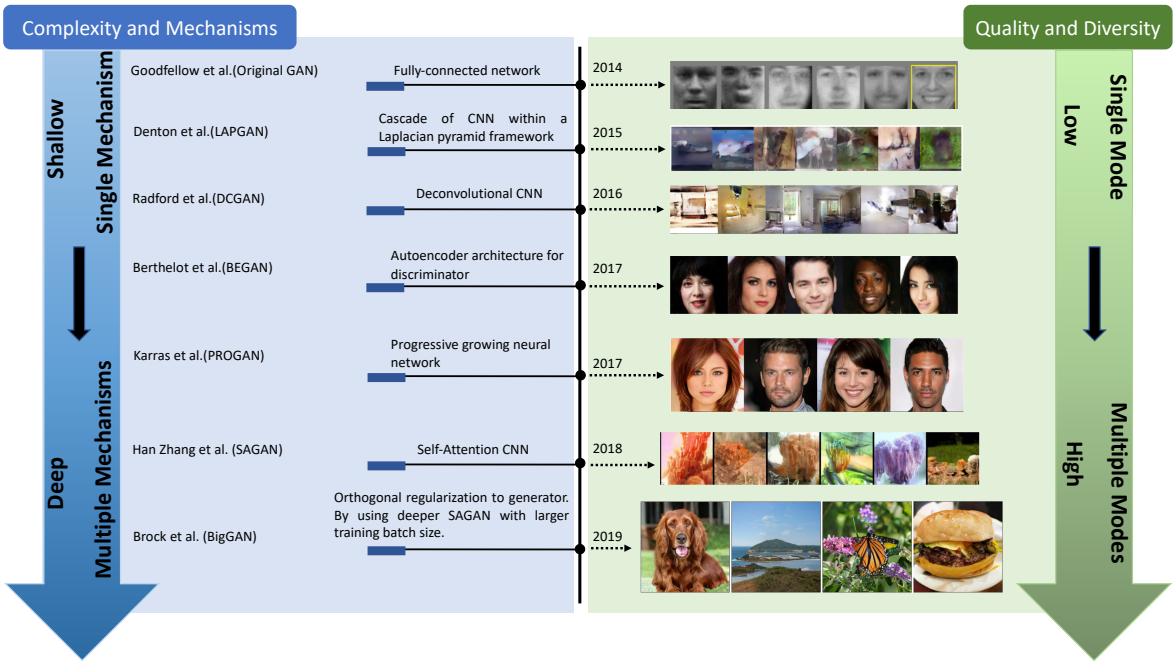



Fig. 2. Timeline of architecture-variant GANs. Complexity in blue stream refers to size of the architecture and computational cost such as batch size. Mechanisms refer to the number of types of models used in the architecture (e.g., BEGAN uses an autoencoder architecture for its discriminator while a deconvolutional neural network is used for the generator. In this case, two mechanisms are used).

better output than other DGMs. Comparing to the most well-known DGMs—variational autoencoder (VAE), GANs are able to produce any type of probability density while VAE is not able to generate sharp images; (2) The GAN framework can train any type of generator network. Other DGMs may have pre-requirements for the generator e.g., the output layer of generator is Gaussian; (3) There is no restriction on the size of the latent variable. These advantages have led GANs to achieve the state of art performance on producing synthetic data especially for image data.

## 5 Architecture-variant GANs

There are many types of architecture-variants proposed in the literature (see Fig.2) [33], [34], [72]–[74]. Architecture-variant GANs are mainly proposed for the purpose of different applications e.g., image to image transfer [34], image super resolution [41], image completion [75], and text-to-image generation [76]. In this section, we provide a review on architecture-variants that helps improve the performance for GANs from three aspects mentioned before, namely improving image diversity, improving image quality and more stable training. Review for those architecture-variants for different applications can be referred to work [68], [70].

### 5.1 Fully-connected GAN (FCGAN)

The original GAN paper [1] uses fully-connected neural networks for both generator and discriminator. This architecture-variant is applied for some simple image datasets i.e., MNIST [77], CIFAR-10 [66] and Toronto Face Dataset. It does not demonstrate good generalization performance for more complex image types.

### 5.2 Laplacian Pyramid of Adversarial Networks (LAPGAN)

LAPGAN is proposed for the production of higher resolution images from lower resolution input GAN [78]. Figure. 3 demonstrates the up-sampling process of generator in LAPGAN from right to left. LAPGAN utilizes a cascade of CNNs within a Laplacian pyramid framework [80] to generate high quality images.

### 5.3 Deep Convolutional GAN (DCGAN)

DCGAN is the first work that applied a deconvolutional neural networks architecture for $G$ [72]. Figure. 4 illustrates the proposed architecture for $G$. Deconvolution is proposed to visualize the features for a CNN and has shown good performance for CNNs visualization [81]. DCGAN deploys the spatial up-sampling ability of the deconvolution operation for $G$, which enables the generation of higher resolution images using GANs.

### 5.4 Boundary Equilibrium GAN (BEGAN)

BEGAN uses an autoencoder architecture for the discriminator which was first proposed in EBGAN [82] (see Fig. 5). Compared to traditional optimization, the BEGAN matches the autoencoder loss distributions using a loss derived from the Wasserstein distance instead of matching data distributions directly. This modification helps $G$ to generate easy-to-reconstruct data for the autoencoder at the beginning because the generated data is close to 0 and the real data distribution has not been learned accurately yet, which prevents $D$ easily winning $G$ at the early training stage.

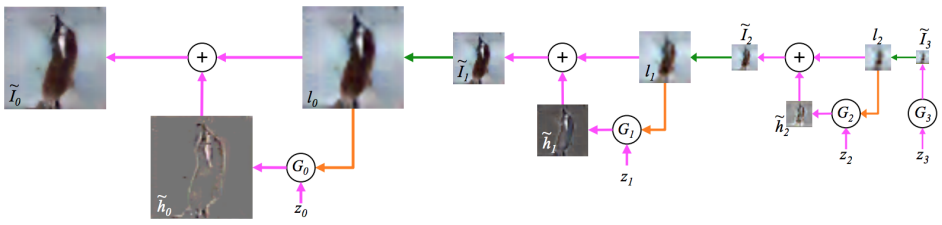


Fig. 3. Up-sampling process of generator in LAPGAN (from right to left). The up-sampling process is marked using green arrow and a conditioning process via a conditional GAN (CGAN) [79] is marked using the orange arrow. The process initially uses $G_3$ to generate image $\widetilde{I}_3$ and then up-samples the image $\widetilde{I}_3$ to $l_2$. Together with another noise $z_2$, $G_2$ generates a difference image $\widetilde{h}_2$ and adds $\widetilde{h}_2$ to $l_2$ which produces the generated image $\widetilde{I}_2$. The rest can be done in the same manner. LAPGAN contains 3 generators in this work in order to up-sample the image. Figure from [78].

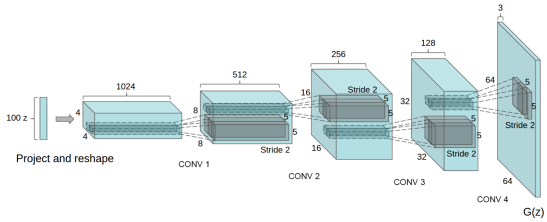



Fig. 4. Detail of DCGAN architecture for generator. This generator successfully generates $64 \times 64$ pixel image for LSUN scene dataset, which is more complex than the datasets used in the original work. Figure from [72].

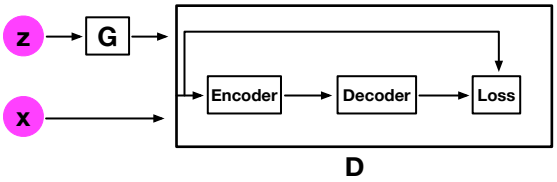



Fig. 5. Illustration of BEGAN architecture. $\mathbf{z}$ is the latent variable for $G$ and $\mathbf{x}$ is input image. BEGAN deploys an antoencoder architecture for the discriminator. Loss is calculated using $L_1$ or $L_2$ norm at pixel level.

### 5.5 Progressive GAN (PROGAN)

PROGAN involves progressive steps toward the expansion of the network architecture [74]. This architecture uses the idea of progressive neural networks first proposed in [83]. This technology does not suffer from forgetting and can leverage prior knowledge via lateral connections to previously learned features. Consequently it is widely applied for learning complex task sequences. Figure. 6 demonstrates the training process for PROGAN. Training starts with low resolution $4 \times 4$ pixels image. Both $G$ and $D$ start to grow with the training progressing. Importantly, all variables remain trainable throughout this growing process. This progressive training strategy enables substantially more stable learning for both networks. By increasing the resolution little by little, the networks are continuously asked a much simpler question comparing to the end goal of discovering a mapping from latent vectors. All current state-of-the-art GANs employ this type of training strategy and it has resulted in impressive, plausible images [29], [74], [84].

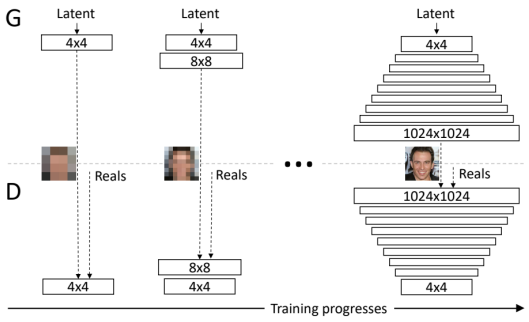



Fig. 6. Progressive growing step for PROGAN during the training process. Training starts with $4 \times 4$ pixels image resolution. With the training step growing, layers are incrementally added to $G$ and $D$ which increases the resolution for the generated images. All existing layers are trainable throughout the training stage. Figure from [74].

### 5.6 Self-attention GAN (SAGAN)

Traditional CNNs can only capture local spatial information and the receptive field may not cover enough structure, which causes CNN-based GANs to have difficulty in learning multi-class image datasets (e.g., ImageNet) and the key components in generated images may shift e.g., the nose in a face-generated image may not appear in right position. Self-attention mechanism have been proposed to ensure large receptive field and without sacrificing computational efficiency for CNNs [85]. SAGAN deploys a self-attention mechanism in the design of the discriminator and generator architectures for GANs [86] (see Fig. 7). Benefiting

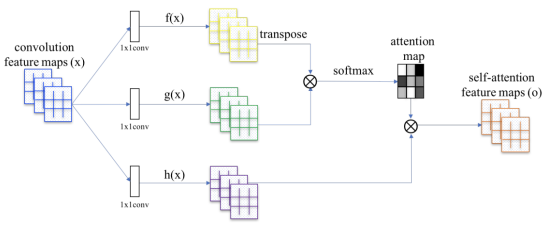



Fig. 7. Self-attention mechanism architecture proposed in the paper. $f$, $g$ and $h$ are correspondence with query, key and value in the self-attention mechanism. The attention map indicates the long-range spatial dependencies. The $\otimes$ is matrix multiplication. Figure from [86].

from the self-attention mechanism, SAGAN is able to learn global, long-range dependencies for generating images. It

has achieved great performance on multi-class image generation based on the ImageNet datasets.

### 5.7 BigGAN

BigGAN [84] has also achieved state-of-the-art performance on the ImageNet datasets. Its design is based on SAGAN and it has been demonstrated that the increase in batch size and the model complexity can dramatically improve GANs performance with respect to complex image datasets.

### 5.8 Summary

We have provided an overview of architecture-variant GANs which aim to improve performance based on the three key challenges: (1) Image quality; (2) Mode diversity; and (3) Vanishing gradient. An illustration of relative performance can be found in Fig. 8. All proposed architecture-variants are able to improve image quality. SAGAN is proposed for improving the capacity of multi-class learning in GANs, the goal of which is to produce more diverse images. Benefiting from the SAGAN architecture, BigGAN is designed for improving both image quality and image diversity. It should be noted that both PROGAN and BigGAN are able to produce high resolution images. BigGAN realizes this higher resolution by increasing the batch size and the authors mention that a progressive growing [74] operation is unnecessary when the batch size is large enough (2048 used in the original paper [84]). However, a progressive growing operation is still needed when GPU memory is limited (a large batch size is hungry for GPU memory). Benefiting from spectrum normalization (SN), which will be discussed in loss-variant GANs part, both SAGAN and BigGAN is effective for the vanishing gradient challenge. These milestone architecture-variants indicate a strong advantage of GANs — compatibility, where a GAN is open to any type of neural architecture. This property enables GANs to be applied to many different applications.

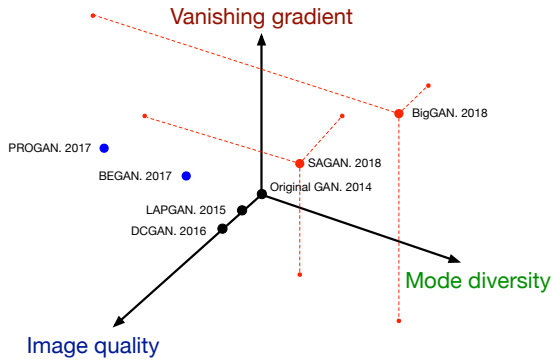



Fig. 8. Summary of recent architecture-variant GANs for solving the three challenges. The challenges are categorized by three orthogonal axes. A larger value for each axis indicates better performance. Red points indicate GAN-variants which cover all three challenges, blue points cover two, and black points cover only one challenge.

Regarding the improvements achieved by different architecture-variant GANs, we next present an analysis on the interconnections and comparisons between the architecture-variants presented here. Starting with the FCGAN described in the original GAN literature, this architecture-variant can only generate simple image datasets. Such a limitation is caused by the network architecture, where the capacity of FC networks is very limited. Research on improving the performance of GANs starts from designing more complex architectures for GANs. A more complex image datasets (e.g., ImageNet) has higher resolution and diversity comparing to simple image datasets (e.g., MNIST) and needs accordingly more sophisticated approaches.

In the context of producing higher resolution images, one obvious approach is to increase the size of generator. LAPGAN and DCGAN up-sample the generator based on such a perspective. Benefiting from the concise deconvolutional up-sampling process and easy generalization of DCGAN, the architecture in DCGAN is more widely used in the GANs literature. It should be noticed that most GANs in the computer vision area use the deconvolutional neural network as the generator, which is first used in DCGAN. Therefore, DCGAN is one of the classical GAN-variants in the literature.

The ability to produce high quality images is an important aspect of GANs clearly. This can be improved through judicious choice of architecture. BEGAN and PROGAN demonstrate approaches from this perspective. With the same architecture used for the generator in DCGAN, BEGAN redesigns the discriminator by including encoder and decoder, where the discriminator tries to distinguish the difference between the generated and autoencoded images in pixel space. Image quality has been improved in this case. Based on DCGAN, PROGAN demonstrates a progressive approach that incrementally trains an architecture similar to DCGAN. This novel approach cannot only improve image quality but also produce higher resolution images.

Producing diverse images is the most challenging task for GANs and it is very difficult for GANs to successfully produce images such as those represented in the ImageNet sets. It is difficult for traditional CNNs to learn global and long-range dependencies from images. Thanks to self-attention mechanism though, approaches such as those in SAGAN integrate self-mechanisms to both discriminator and generator, which helps GANs a lot in terms of learning multi-class images. Moreover, BigGAN, which can be considered an extension of SAGAN, introduces a deeper GAN architecture with a very large batch size, which produces high quality and diverse images as in ImageNet and is the current state-of-the-art.

## 6 Loss-variant GANs

Another design decision in GANs which significantly impacts performance is the choice of loss function in equation (1). While the original GAN work [1] has already proved global optimality and the convergence of GANs training. It still highlights the instability problem which can arise when training a GAN. The problem is caused by the global optimality criterion as stated in [1]. Global optimality is achieved when an optimal $D$ is reached for any $G$. So the optimal $D$ is achieved when the derivative of $D$ for the loss

in equation (1) equals 0. So we have

$$-\frac{p_r(\mathbf{x})}{D(\mathbf{x})}+\frac{p_g(\mathbf{x})}{1-D(\mathbf{x})}=0, \\ D^*(\mathbf{x})=\frac{p_r(\mathbf{x})}{p_r(\mathbf{x})+p_g(\mathbf{x})}, \tag{2}$$

where $\mathbf{x}$ represents the real data and generated data, $D^*(\mathbf{x})$ is the optimal discriminator, $p_r(\mathbf{x})$ is the real data distribution and $p_g(\mathbf{x})$ is the generated data distribution. We have got the optimal discriminator $D$ so far. When we have the optimal $D$, the loss for $G$ can be visualized by substituting $D^*(\mathbf{x})$ into equation (1)

$$\mathcal{L}_G = \mathbb{E}_{\mathbf{x}\sim p_r}\log\frac{p_r(\mathbf{x})}{\frac{1}{2}\left[p_r(\mathbf{x})+p_g(\mathbf{x})\right]}+ \\ \mathbb{E}_{\mathbf{x}\sim p_g}\log\frac{p_g(\mathbf{x})}{\frac{1}{2}\left[p_r(\mathbf{x})+p_g(\mathbf{x})\right]}-2\cdot\log 2. \tag{3}$$

Equation (3) demonstrates the loss function for a GAN when discriminator is optimized and it is related to two important probability measurement metrics. One is Kullback–Leibler (KL) divergence which is defined as

$$KL(p_1\|p_2)=\mathbb{E}_{\mathbf{x}\sim p_1}\log\frac{p_1}{p_2}, \tag{4}$$

and the other is Jensen-Shannon (JS) divergence which is stated as

$$JS(p_1\|p_2)=\frac{1}{2}KL(p_1\|\frac{p_1+p_2}{2})+ \\ \frac{1}{2}KL(p_2\|\frac{p_1+p_2}{2}). \tag{5}$$

Thus the loss for $G$ regarding the optimal $D$ in equation (3) can be reformulated as

$$\mathcal{L}_G=2\cdot JS(p_r\|p_g)-2\cdot\log 2, \tag{6}$$

which indicates that the loss for $G$ now equally becomes the minimization of the JS divergence between $p_r$ and $p_g$. With the training $D$ step by step, the optimization of $G$ will be closer to the minimization of JS divergence between $p_r$ and $p_g$. We now start to explain the unstable training problem, where $D$ often easily wins $G$. This unstable training problem is actually caused by the JS divergence in equation (5). Give an optimal $D$, the objective of optimization for equation (6) is to move $p_g$ toward $p_r$ (see Fig. 9). JS divergence for the three plots from left to right are 0.693, 0.693 and 0.336, which indicates that JS divergence stays constant (log2=0.693) if there is no overlap between $p_r$ and $p_g$. Figure 10 demonstrates the change of JS divergence and its gradient corresponding to the distance between $p_r$ and $p_g$. It can be seen that JS divergence is constant and its gradient is almost 0 when the distance is greater than 5, which indicates that training process does not have any effect on $G$. The gradient of JS divergence for training the $G$ is non-zero only when $p_g$ and $p_r$ have substantial overlap i.e., the vanishing gradient will arise for $G$ when $D$ is close to optimal. In practice, the possibility that $p_r$ and $p_g$ do not overlap or have negligible overlap is very high [87].

The original GANs work [1] also highlights the minimization of $-\mathbb{E}_{\mathbf{x}\sim p_g}\log[D(\mathbf{x})]$ for training $G$ to avoid a vanishing gradient. However, this training strategy will lead to another problem called mode dropping. First, let us examine $KL(p_g\|p_r)=\mathbb{E}_{\mathbf{x}\sim p_g}\log\frac{p_g}{p_r}$. With an optimal discriminator $D^*$, $KL(p_g\|p_r)$ can be reformulated as

$$KL(p_g\|p_r)=\mathbb{E}_{\mathbf{x}\sim p_g}\log\frac{p_g(\mathbf{x})/(p_r(\mathbf{x})+p_g(\mathbf{x}))}{p_r(\mathbf{x})/(p_r(\mathbf{x})+p_g(\mathbf{x}))}, \\ =\mathbb{E}_{\mathbf{x}\sim p_g}\log\frac{1-D^*(\mathbf{x})}{D^*(\mathbf{x})}, \\ =\mathbb{E}_{\mathbf{x}\sim p_g}\log[1-D^*(\mathbf{x})]- \\ \mathbb{E}_{\mathbf{x}\sim p_g}\log[\mathrm{D}^*(\mathbf{x})]. \tag{7}$$

The alternative loss form for $G$ now can be stated by switching the order of the two sides in equation (7)

$$-\mathbb{E}_{\mathbf{x}\sim p_g}\log[D^*(\mathbf{x})] \\ =KL(p_g\|p_r)-\mathbb{E}_{\mathbf{x}\sim p_g}\log[1-D^*(\mathbf{x})], \\ =KL(p_g\|p_r)-2\cdot JS(p_r\|p_g)+ \\ 2\cdot\log 2+\mathbb{E}_{\mathbf{x}\sim p_x}\log[D^*(\mathbf{x})], \tag{8}$$

where the alternative loss for $G$ in equation (8) is only affected by the first two terms (the last two terms are constant). It can be noticed that the optimization in equation (8) is contradictory because the first term aims to push the generated distribution toward the real distribution while the second term aim to push in the opposite direction (the negative sign). This will cause an unstable numerical gradient for training $G$. More importantly, KL divergence is an asymmetrical distribution measurement highlighted below

- *When* $p_g(\mathbf{x})\to 0$, $p_r(\mathbf{x})\to 1$, $KL(p_g\|p_r)\to 0$.
- *When* $p_g(\mathbf{x})\to 1$, $p_r(\mathbf{x})\to 0$, $KL(p_g\|p_r)\to+\infty$.

The penalization for two instances of poor performance made by $G$ are totally different. The first instance of poor performance is that $G$ is not producing a reasonable range of samples and yet incurs a very small penalization. The second instance of poor performance concerns $G$ producing implausible samples but has very large penalization. The first example concerns the fact that the generated samples lack diversity while the second concerns that fact that the generated samples are not accurate. Considering this first case, $G$ generates repeated but "safe" samples instead of taking risk to generate diverse but "unsafe" samples, which leads to the mode collapse problem. In summary, using the original loss in equation (1) will result in the vanishing gradient for training $G$ and using the alternative loss in equation (8) will incur the mode collapse problem. These kind of problems cannot be solved by changing the GANs architectures. Therefore, it could be argued that ultimate GANs problem stems from the design of the loss function and that innovative ideas for this redesign of the loss function may solve the problem.

Loss-variant GANs have been researched extensively to improve the stability of training GANs.

### 6.1 Wasserstein GAN (WGAN)

WGAN [88] has successfully solved the two problems for the original GAN by using the Earth mover (EM) or Wasserstein-1 [89] distance as the loss measure for optimization. The EM distance is defined as

$$W(p_r,p_g)=\inf_{\gamma\in\prod(p_r,p_g)}\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\gamma}\|\mathbf{x}-\mathbf{y}\|, \tag{9}$$

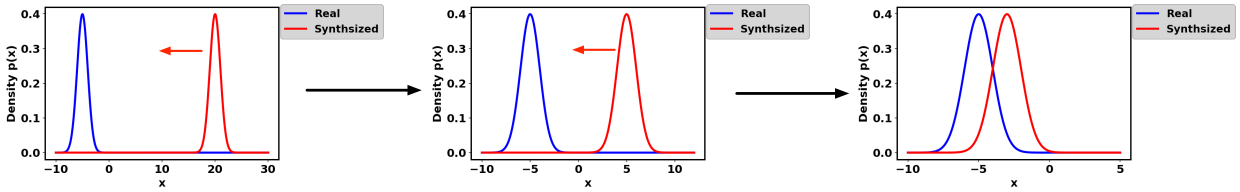
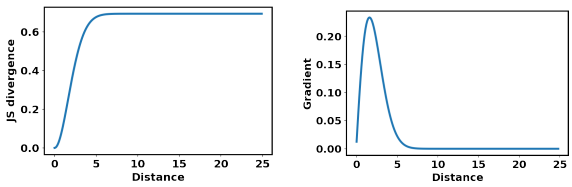



Fig. 9. Illustration of training progress for a GAN. Two normal distributions are used here for visualization. Given an optimal $D$, the objective of GANs is to update $G$ in order to move the generated distribution $p_g$ (red) towards the real distribution $p_r$ (blue) ($G$ is updated from left to right in this figure. Left: initial state, middle: during training, right: training converging). However, JS divergence for the left two figures are both 0.693 and the figure on the right is 0.336, indicating that JS divergence does not provide sufficient gradient at the initial state.




(a) JS divergence changes with distance. (b) Gradient JS divergence changes with distance.

Fig. 10. JS divergence and gradient change with the distance between $p_r$ and $p_g$. The distance is the difference between two distribution means.

where $\prod(p_r, p_g)$ denotes the set of all joint distributions and $\gamma(\mathbf{x}, \mathbf{y})$ whose marginals are $p_r$ and $p_g$. Compared with KL and JS divergence, EM is able to reflect distance even when $p_r$ and $p_g$ do not overlap and it is also continuous and thus able to provide meaningful gradient for training the generator. Figure 18 illustrates the gradient of WGAN comparing to the original GAN. It is noticeable that WGAN has a smooth gradient for training the generator spanning the complete space. However, the infimum in equation (9) is intractable but the creators demonstrate that instead the Wasserstein distance can be estimated as

$$\max_{w \sim \mathcal{W}} \mathbb{E}_{\mathbf{x}_{p_r}}[f_w(\mathbf{x})] - \mathbb{E}_{\mathbf{z} \sim p_z}[f_w(G(\mathbf{z}))], \tag{10}$$

where $f_w$ can be realized by $D$ but has some constraints (for details the interested reader can refer to the original work [88]) and $\mathbf{z}$ is the input noise for $G$. So $w$ here is the parameters in $D$ and $D$ aims to maximize equation (10) in order to make the optimization distance equivalent to Wasserstein distance. When $D$ is optimized, equation (9) will become the Wasserstein distance and $G$ aims to minimize it. So the loss for $G$ is

$$-\min_G \mathbb{E}_{\mathbf{z} \sim p_z}[f_w(G(\mathbf{z}))] \tag{11}$$

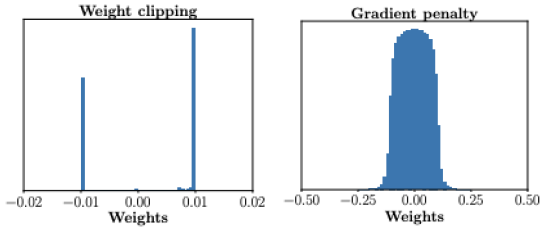
An important difference between WGAN and the original GAN is the function of $D$. The $D$ in the original work is used as a binary classifier but $D$ used in WGAN is to fit the Wasserstein distance, which is a regression task. Thus, the sigmoid in the last layer of $D$ is removed in the WGAN.

### 6.2 WGAN-GP

Even though WGAN has been shown to be successful in improving the stability of GAN training, it is not well generalized for a deeper model. Experimentally it has been determined that most WGAN parameters are localized at -0.01 and 0.01 because of parameter clipping. This will dramatically reduce the modeling capacity of $D$. WGAN-GP has been proposed using gradient penalty for restricting $\|f\|_L \leq K$ for the discriminator [56] and the modified loss for discriminator now becomes

$$\begin{aligned}\mathcal{L}_D =& \mathbb{E}_{\mathbf{x}_g \sim p_g}[D(\mathbf{x}_g)] - \mathbb{E}_{\mathbf{x}_r \sim p_r}[D(\mathbf{x}_r)] + \\ & \lambda \mathbb{E}_{\hat{\mathbf{x}} \sim p_{\hat{x}}}[(\|\nabla_{\hat{\mathbf{x}}} D(\hat{\mathbf{x}})\|_2 - 1)^2],\end{aligned} \tag{12}$$

where $\mathbf{x}_r$ is sample data drawn from the real data distribution $p_r$, $\mathbf{x}_g$ is sample data drawn from the generated data distribution $p_g$ and $p_{\hat{\mathbf{x}}}$ sampling uniformly along the straight lines between pairs of points sampled from the real data distribution $p_r$ and the generated data distribution $p_g$. The first two terms are original loss in WGAN and the last term is the gradient penalty. WGAN-GP demonstrates a better distribution of trained parameters compared to WGAN (Fig. 11) and better stability performance during training of GANs.




(a) Weights of WGAN (b) Weights of WGAN-GP

Fig. 11. Comparison of parameter distribution between WGAN and WGAN-GP. Top is WGAN and bottom is WGAN-GP. Figure from [56].

### 6.3 Least Square GAN (LSGAN)

The LSGAN is a new approach proposed in [90] to remedy the vanishing gradient problem for $G$ from the perspective of the decision boundary determined by the discriminator. This work argues that the decision boundary for $D$ of the original GAN penalizes very small error to update $G$ for those generated samples that are far away from the decision boundary. The author proposes using a least square loss for $D$ instead of sigmoid cross entropy loss stated in the original paper [1]. The proposed loss function is defined as

$$\begin{aligned}\min_D \mathcal{L}_D =& \frac{1}{2}\mathbb{E}_{\mathbf{x} \sim p_r}[(D(\mathbf{x}) - b)^2] + \\ & \frac{1}{2}\mathbb{E}_{\mathbf{z} \sim p_z}[(D(G(\mathbf{z})) - a)^2], \\ \min_G \mathcal{L}_G =& \frac{1}{2}\mathbb{E}_{\mathbf{z} \sim p_z}[(D(G(\mathbf{z})) - c)^2],\end{aligned} \tag{13}$$

where $a$ is the label for the generated samples, $b$ is the label for the real samples and $c$ is the value that $G$ wants $D$ to believe for the generated samples. This modified change has two benefits: (1) The new decision boundary made by $D$ penalizes large error arising from those generated samples that are far away from the decision boundary, which pushes those "bad" generated samples towards the decision boundary. This is beneficial in terms of generating improved image quality; (2) Penalizing the generated samples far away from the decision boundary can provide more gradient when updating the $G$, which remedies the vanishing gradient problems for training $G$. Figure 12 demonstrates the comparison of decision boundaries for LSGAN and the original GAN. The decision boundaries for $D$ that have been trained by the original sigmoid cross entropy loss and the proposed least square loss are different. The work [90] has proven that the optimization of LSGAN is equivalent to minimizing the Pearson $\chi^2$ divergence between $p_r + p_g$ and $2p_g$ when $a$,$b$ and $c$ satisfy the condition of $b-c=1$ and $b-a=2$. Similar to WGAN, $D$ here behaves as regression and the sigmoid is also removed.

### 6.4 f-GAN

f-GAN summarizes that GANs can be trained by using an f-divergence [91]. f-divergence is a function $D_f(P\|Q)$ that measures the difference between probability distributions $P$ and $Q$. e.g., KL divergence, JS divergence and Pearson $\chi^2$ as mentioned before. This work discusses the efficacy of various divergence function in terms of training complexity and the quality of the generative models.

### 6.5 Unrolled GAN (UGAN)

UGAN is a design proposed to solve the problem of mode collapse for GANs during training [92]. The core design innovation of UGAN is the addition of a gradient term for updating $G$ that captures how the discriminator would react to a change in the generator. The optimal parameter for $D$ can be expressed as the fixed point of an iterative optimization procedure

$$\begin{aligned} \boldsymbol{\theta}_D^0 =& \boldsymbol{\theta}_D, \\ \boldsymbol{\theta}_D^{k+1} =& \boldsymbol{\theta}_D^k + \eta^k \frac{\mathrm{d}f(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D^k)}{\mathrm{d}\boldsymbol{\theta}_D^k}, \\ \boldsymbol{\theta}_D^*(\boldsymbol{\theta}_G) =& \lim_{k\to\infty} \boldsymbol{\theta}_D^k, \end{aligned} \tag{14}$$

where $\eta^k$ is the learning rate, $\boldsymbol{\theta}_D$ represents parameters for $D$ and $\boldsymbol{\theta}_G$ represents parameters for $G$. The surrogate loss by unrolling for $K$ steps can be expressed as

$$f_K(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D) = f(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D^K(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D)). \tag{15}$$

This surrogate loss is then used for updating parameters for $D$ and $G$

$$\begin{aligned} \boldsymbol{\theta}_G &\leftarrow \boldsymbol{\theta}_G - \eta \frac{\mathrm{d}f_K(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D)}{\mathrm{d}\boldsymbol{\theta}_G}, \\ \boldsymbol{\theta}_D &\leftarrow \boldsymbol{\theta}_D + \eta \frac{\mathrm{d}f(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D)}{\mathrm{d}\boldsymbol{\theta}_D} \end{aligned} \tag{16}$$

Figure 13 illustrates the computation graph for an unrolled GAN with 3 unrolling steps. Equation (17) illustrates the gradient for updating $G$.

$$\begin{aligned} \frac{\mathrm{d}f_K(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D)}{\mathrm{d}\boldsymbol{\theta}_G} =& \frac{\partial f(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D^K(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D))}{\boldsymbol{\theta}_G} + \\ & \frac{\partial f(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D^K(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D))}{\partial \boldsymbol{\theta}_D^K(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D)} \frac{\mathrm{d}\boldsymbol{\theta}_D^K(\boldsymbol{\theta}_G, \boldsymbol{\theta}_D)}{\mathrm{d}\boldsymbol{\theta}_G} \end{aligned} \tag{17}$$

It should be noted that the first term in equation (17) is the gradient for the original GAN. The second term here reflects how $D$ reacts to changes in $G$. If $G$ tends to collapse to one mode, $D$ will increase the loss for $G$. Thus, this unrolled approach is able to prevent the mode collapse problem for GANs.

### 6.6 Loss Sensitive GAN (LS-GAN)

LS-GAN was introduced to train the generator to produce realistic samples by minimizing the designated margins between real and generated samples [93]. This work argues that the problems such as the vanishing gradient and mode collapse as appearing in the original GAN is caused by a non-parametric hypothesis that the discriminator is able to distinguish any type of probability distribution between real samples and generated samples. As mentioned before, it is very normal for the overlap between the real samples distribution and the generated samples distribution to be negligible. Moreover, $D$ is also able to separate real samples and generated samples. The JS divergence will become a constant under this situation, where the vanishing gradient arises for $G$. In LS-GAN, the classification ability of $D$ is restricted and is learned by a loss function $L_\theta(\mathbf{x})$ parameterized with $\theta$, which assumed that a real sample ought to have smaller loss than a generated sample. The loss function can be trained as the following constraint

$$L_\theta(\mathbf{x}) \leq L_\theta(G(\mathbf{z})) - \Delta(\mathbf{x}, G(\mathbf{z})), \tag{18}$$

where $\Delta(\mathbf{x}, G(\mathbf{z}))$ is the margin measuring the difference between real samples and generated samples. This constraint indicates that a real sample is separated from a generated sample by at least a margin of $\Delta(\mathbf{x}, G(\mathbf{z}))$. The optimization for the LS-GAN is then stated as

$$\begin{aligned} \min_D \mathcal{L}_D =& \mathbb{E}_{\mathbf{x}\sim p_r} L_\theta(\mathbf{x}) + \\ & \lambda \mathbb{E}_{\substack{\mathbf{x}\sim p_r \\ \mathbf{z}\sim p_z}} (\Delta(\mathbf{x}, G(\mathbf{z})) + \\ & L_\theta(\mathbf{x}) - L_\theta(G(\mathbf{z})))_+, \\ \min_G \mathcal{L}_G =& \mathbb{E}_{\mathbf{z}\sim p_z} L_\theta(G(\mathbf{z})), \end{aligned} \tag{19}$$

where $\lambda$ is a positive balancing parameter, $(a)_+ = \max(a, 0)$ and $\boldsymbol{\theta}$ are the parameters in $D$. From the second term in $\mathcal{L}_D$ in the equation (19), $\Delta(\mathbf{x}, G(\mathbf{z}))$ is added as a regularization term for optimizing $D$ in order to prevent $D$ from overfitting the real samples and the generated samples. Figure 14 demonstrates the efficacy of equation (19). The loss for $D$ puts a restriction on the ability of $D$ i.e., It challenges the ability of $D$ for to separate well generated samples from real samples, which is the original cause for the vanishing gradient. More formally, LS-GAN assumes that $p_r$ lies in a set of Lipschitz densities with a compact support.

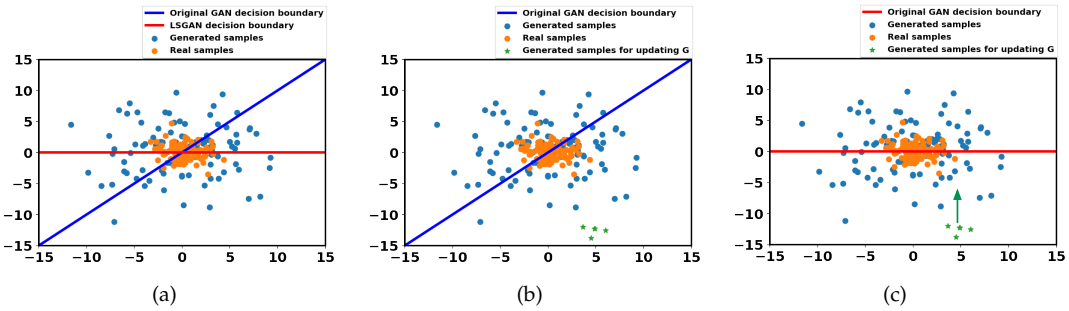



Fig. 12. Decision boundary illustration of original GAN and LSGAN. (a). Decision boundaries for $D$ of original GAN and LSGAN. (b). Decision boundary of $D$ for the original GAN. It has small errors for the generated samples, which is far away from the decision boundary (in green), for updating $G$. (c). Decision boundary for $D$ of LSGAN. It penalizes the large error for any generated sample that is far away from the boundary (in green). Thus it pushes generated samples (in green) toward the boundary [90].

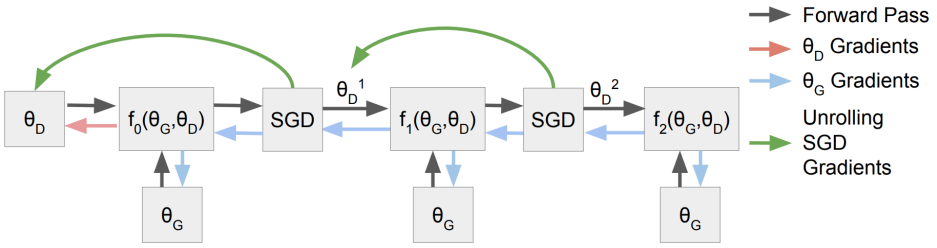



Fig. 13. An example of computation for an unrolled GAN with 3 unrolling steps. $G$ and $D$ update using equation (16). Each step $k$ uses the gradients of $f_k$ regarding $\boldsymbol{\theta}_D^k$ stated in the equation (14). Figure from [92].

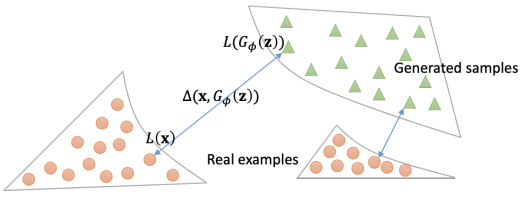



Fig. 14. Demonstration of the loss in equation (19). $\Delta(\mathbf{x}, G(\mathbf{z}))$ is used to separate real samples and generated samples. If some generated samples are close enough to the real samples, LS-GAN will focus on other generated samples that are far away from the real samples. This optimization loss puts restriction on $D$ to prevent it from separating generated and real samples too well. Thus, it solves the vanishing gradient problem which arises in original GANs. ($G_\phi(\mathbf{z})$ here is equivalent to $G(\mathbf{z})$ where $\phi$ represents the parameters for generator). Figure from [93].

### 6.7 Mode Regularized GAN (MRGAN)

MRGAN proposes a metric regularization to penalize missing modes [94], which is then used to solve the mode collapse problem. The key idea behind this work is the use of an encoder $E(\mathbf{x})$: $\mathbf{x} \rightarrow \mathbf{z}$ to produce the latent variable $\mathbf{z}$ for $G$ instead of using noise. This procedure has two benefits: (1) The encoder reconstruction can add more information to $G$ so that is not that easy for $D$ to distinguish between generated samples and real samples; and (2) the encoder ensures correspondence between $\mathbf{x}$ and $\mathbf{z}$ ($E(\mathbf{x})$), which means $G$ can cover different modes in the $\mathbf{x}$ space. So it prevents the mode collapse problem. The loss function for this mode regularized GAN is

$$\begin{aligned}\mathcal{L}_G &= -\mathbb{E}_{\mathbf{z}}[\log[D(G(\mathbf{z}))]] + \\ &\quad \mathbb{E}_{\mathbf{x}\sim p_r}[\lambda_1 d(\mathbf{x}, G\circ E(\mathbf{x})) + \\ &\quad \lambda_2 \log[D(G(\mathbf{x}))]], \\ \mathcal{L}_E &= \mathbb{E}_{\mathbf{x}\sim p_r}[\lambda_1 d(\mathbf{x}, G\circ E(\mathbf{x})) + \\ &\quad \lambda_2 \log[D(G(\mathbf{x}))]],\end{aligned} \tag{20}$$

where $d$ is a geometric measurement which can be chosen from many options e.g., pixel-wise $L^2$ and distance of extracted features.

### 6.8 Geometric GAN

Geometric GAN [95] is proposed using SVM separating hyperplane, which has the maximal margins between the two classes. Figure 15 demonstrates the update rule for the discriminator and generator based on the SVM hyperplane. Geometric GAN has been successfully demonstrated to be more stable for training and less prone to mode collapse.

### 6.9 Relativistic GAN (RGAN)

RGAN [96] is proposed as a general approach to devising new cost functions from the existing one i.e., it can be generalized for all integral probability metric (IPM) [97], [98] GANs. The discriminator in the original GAN measures *the probability for a given real sample or a generated sample*. The author argues that key relative discriminant information between real data and generated data is missing in original GAN. The discriminator in RGAN takes into account that

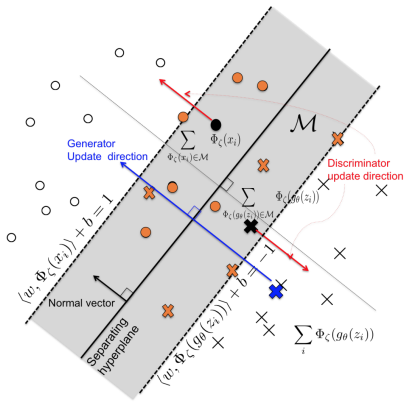



Fig. 15. SVM hyperplane used in Geometric GAN. Discriminator updates by pushing real data samples and generated data samples away from the hyperplane while generator updates by pushing generated data samples towards the hyperplane. Figure from [95].

*how a given real sample is more realistic than a randomly sampled generated sample*. Loss function of RGAN applied to original GAN is stated as

$$\begin{aligned} &\min_D \mathbb{E}_{\substack{\mathbf{x}_r\sim p_r \\ \mathbf{x}_g\sim p_g}} \\ &[\log(\text{sigmoid}(C(\mathbf{x}_r) - C(\mathbf{x}_g)))], \\ &\min_G \mathbb{E}_{\substack{\mathbf{x}_r\sim p_r \\ \mathbf{x}_g\sim p_g}} \\ &[\log(\text{sigmoid}(C(\mathbf{x}_g) - C(\mathbf{x}_r)))], \end{aligned} \tag{21}$$

where $C(\mathbf{x})$ is the non-transformed layer. Figure 16 demon-

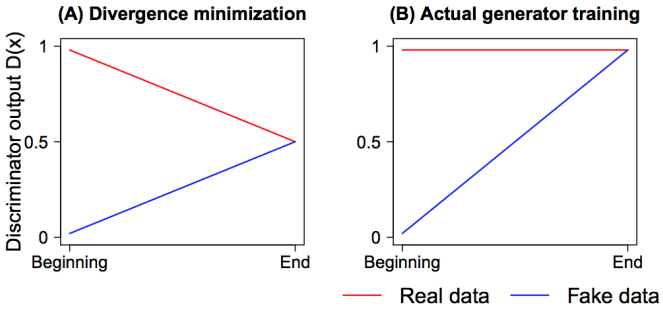



Fig. 16. $D$ output comparison between RGAN and original GAN. (a) $D$ output in RGAN; (b) $D$ output in original GAN when training the $G$. Figure from [96].

strates the effect on $D$ of using the RGAN approach compared to the original GAN. In terms of the original GAN, the optimization aims to push the $D(\mathbf{x})$ to 1 (right one). For RGAN, the optimization aims to push $D(\mathbf{x})$ to 0.5 (left one), which is more stable compared to the original GAN. The author also claims that RGAN can be generalized to other types of loss-variant GANs if those loss functions belong to IPMs. The generalization loss is stated as

$$\begin{aligned} \mathcal{L}_D = &\mathbb{E}_{\substack{\mathbf{x}_r\sim p_r \\ \mathbf{x}_g\sim p_g}}[f_1(C(x_r) - C(x_g))] + \\ &\mathbb{E}_{\substack{\mathbf{x}_r\sim p_r \\ \mathbf{x}_g\sim p_g}}[f_2(C(x_g) - C(x_r))], \\ \mathcal{L}_G = &\mathbb{E}_{\substack{\mathbf{x}_r\sim p_r \\ \mathbf{x}_g\sim p_g}}[g_1(C(x_r) - C(x_g))] + \\ &\mathbb{E}_{\substack{\mathbf{x}_r\sim p_r \\ \mathbf{x}_g\sim p_g}}[g_2(C(x_g) - C(x_r))], \end{aligned} \tag{22}$$

where $f_1(y) = g_2(y) = -y$ and $f_2(y) = g_1(y) = y$. Details of loss generalization for other GANs refers to the original paper [96].

### 6.10 Spectral normalization GAN (SN-GAN)

SN-GAN proposes the use of weight normalization to stabilize the training of the discriminator. This technique is computationally light and easily applied to existing GANs. Previous work for stabilizing the training of GANs [56], [88], [93] emphasizes the importance that $D$ should be from the set of K-Lipshitz continuous functions. Popularly speaking, Lipschitz continuity [99]–[101] is more strict than the continuity, which describes that the function does not change rapidly. This smooth $D$ is of benefit in stabilizing the training of GANs. The work mentioned previously focused on the control of the Lipschitz constant of the discriminator function. This work demonstrates an alternative simpler way to control the Lipschitz constant through spectral normalization of each layer for $D$. Spectral normalization is performed as

$$\bar{\mathbf{W}}_{SN}(\mathbf{W}) = \frac{\mathbf{W}}{\sigma(\mathbf{W})}, \tag{23}$$

where $\mathbf{W}$ represents weights on each layer for $D$ and $\sigma(\mathbf{W})$ is the $L_2$ matrix norm of $\mathbf{W}$. The paper proves this will make $\|f\| \leq 1$. The fast approximation for the $\sigma(\mathbf{W})$ is also demonstrated in the original paper.

### 6.11 Summary

We explain the training problems (mode collapse and vanishing gradient for $G$) in the original GAN and we have introduced loss-variant GANs in the literature, which are mainly proposed for improving the performance of GANs in terms of three key aspects. Figure 17 summarizes the

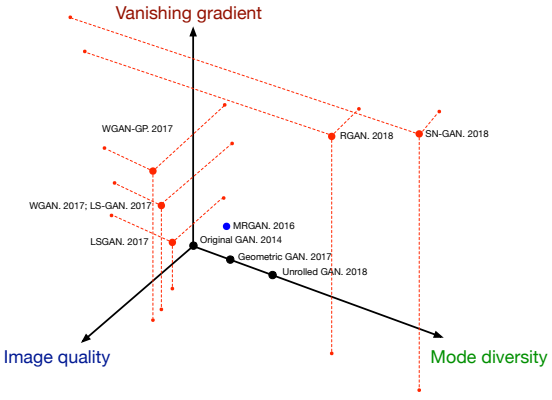
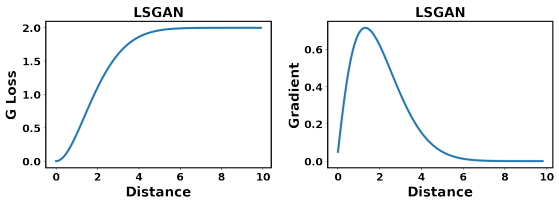
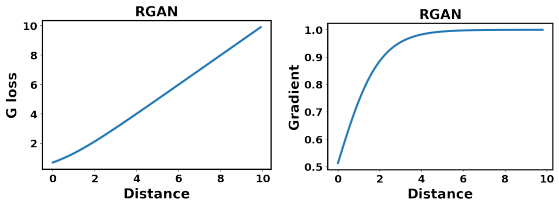



Fig. 17. Current loss-variants for solving the challenges. Challenges are categorized in terms of three independent axes. Red points indicate the GAN-variant covers all three challenges, blue points cover two, and black points cover only one challenge. Larger value for each axis indicates better performance.

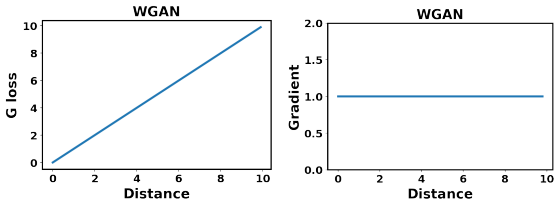
efficacy of loss-variant GANs for the challenges. Losses of LSGAN, RGAN and WGAN are very similar to the original GAN loss. We use a toy example (i.e., two distributions used in Fig. 9) to demonstrate the G loss regarding the distance between real data distribution and generated data distribution in Fig. 18. It can be seen that RGAN and WGAN are able




(a) LSGAN




(b) RGAN




(c) WGAN

Fig. 18. Loss and gradient for generator of different loss-variant GANs.

to inherently solve the vanishing gradient problems for the generator when the discriminator is optimized. LSGAN on the contrary still suffers from a vanishing gradient for the generator, however, it is able to provide a better gradient compared to the original GAN in Fig. 10 when the distance between the real data distribution and the generated data distribution is relatively small. This is demonstrated in the original paper [90] where LSGAN is shown to be better easier to push generated samples to the boundary made by discriminator.

Table 1 gives details of the properties of each loss-variant GAN. WGAN, LSGAN, LS-GAN, RGAN and SN-GAN are proposed to overcome the vanishing gradient for $G$. LSGAN argues that the vanishing gradient is mainly caused by the sigmoid function in the discriminator so it uses a least squares loss to optimize the GAN. LSGAN turns out to be the optimization on Pearson $\chi^2$ divergence and remedies the vanishing gradient problem. WGAN uses Wasserstein (or Earth mover) distance as the loss. Compared to JS divergence, Wasserstein distance is smoother and there is no sudden change with respect to the distance between real samples and generated samples. To be able to use Wasserstein distance as the loss, the discriminator must be Lipschitz continuous, where WGAN deploys the parameter clipping to force discriminator satisfy the Lipschitz continuity. However, it causes problems such as most of parameters in the discriminator locates to the edges of clipping range, which leads to the low capacity of discriminator. WGAN-GP is proposed the use of gradient penalty to make discriminator is Lipschitz continuous, which successfully solves the problems in WGAN. LS-GAN proposes to use a margin that is enforced to separate real samples from generated samples, which restricts the modelling capability of discriminator. It solves the vanishing gradient problem for the generator because this problem arises when the discriminator is optimized. RGAN is a unified framework that is suitable for all IPM-based GANs e.g., WGAN. RGAN adds discriminant information to GANs for better learning. SN-GAN proposes an elegant way for optimizing a GAN. As mentioned before, a Lipschitz continuous discriminator is important for stable learning, vanishing gradient and so on. SN-GAN proposes spectral normalization [67] to constrain the discriminator under the Lipschitz continuous requirement. SN-GAN is the first GAN (we do not consider AC-GANs [11] because an ensemble of 100 AC-GANs is used for ImageNet datasets [102].) that has been successfully applied to ImageNet datasets. In theory, spectral normalization as demonstrated in the SN-GAN could be applied to every GAN type. SAGAN and BigGAN [84], [86] both deploy spectral normalization and achieve good results with ImageNet.

Loss-variant GANs are able to be applied to architecture-variants. However, SN-GAN and RGAN show stronger generalization abilities compared to other loss-variants, where these two loss-variants can be deployed by other types of loss-variants. Spectral normalization can be applied to any GAN-variant while the RGAN concept can be applied to any IPM-based GAN. We strongly recommend the use of spectral normalization for all GANs applications as described here. There are a number of loss-variant GANs mentioned in this paper which is able to solve the mode collapse and unstable training problem. Details are given in Table 1.

## 7 Discussion

We have introduced the most significant problems present in the original GAN design, which are mode collapse and vanishing gradient for updating $G$. We have surveyed significant GAN-variants that remedy these problems through two design considerations: (1) Architecture-variants. This aspect focuses on architectural options for GANs. This approach enables GANs to be successfully applied to different applications, however, it is not able to fully solve the problems mentioned above; (2) Loss-variant. We have provided a detail explanation why these problems arise in the original GAN. These problems are essentially caused by the loss function in the original GAN. Thus, modifying this loss function can solve this problem. It should be noted that the loss function may change for some architecture-variants. However, this loss function is changed according to the architecture thus it is architecture-specific loss. It is not able to generalize to other architectures.

Through a comparison of the different architectural approaches surveyed in this work, it is clear that the modification of the GAN architecture has significant impact on the generated images quality and their diversity. Recent research shows that the capacity and performance of GANs are related to the network size and batch size [84], which indicates that a well designed architecture is critical for good GANs performance. However, modifications to the architecture only is not able to eliminate all the inherent training problems for GANs. Redesign of the loss function including regularization and normalization can help yield

more stable training for GANs. This work introduced various approaches to the design of the loss function for GANs. Based on the comparison for each loss-variant, we find that spectral normalization as first demonstrated in the SN-GAN brings lots of benefits including ease of implementation, relatively light computational requirements and the ability to work well for almost all GANs. We suggest that researchers, who seek to apply GANs to real-world problems, include spectral normalization to the discriminator.

There is no answer to the question of which GAN is the best. The selection of a specific GAN type depends on the application. For instance, if an application requires the production of natural scenes images (this requires generation of images which are very diverse). DCGAN with spectrum normalization applied, SAGAN and BigGAN can be good choices here. BigGAN is able to produce the most realistic images compared to the other two. However, BigGAN is much more computationally intensive. Thus it depends on the actual computational requirements set by the real-world application.

### 7.1 Interconnections Between Architecture and Loss

In this paper, we highlight the problems inherent in the original GAN design. In highlighting how subsequent researchers have remedied those problems, we explored architecture-variants and loss-variants in GAN designs separately. However, it should be noted that there are interconnections between these two types of GAN-variants. As mentioned before, loss functions are easily integrated to different architectures. Benefit from improved convergence and stabilization through a redesigned loss function, architecture-variants are able to achieve better performance and accomplish solutions to more difficult problems. For examples, BEGAN and PROGAN use Wasserstein distance instead of JS divergence. SAGAN and BigGAN deploy spectral normalization, where they achieve good performance based on multi-class image generation. These two types of variants equally contribute to the progress of GANs.

### 7.2 Future Directions

GANs were originally proposed to produce plausible synthetic images and have achieved exciting performance in the computer vision area. GANs have been applied to some other fields, (e.g., time series generation [20], [21], [103] and natural language processing [15], [104]–[106]) with some success. Compared to computer vision, GANs research in other areas is still somewhat limited. The limitation is caused by the different properties inherent in image versus non-image data. For instance, GANs work to produce continuous value data but natural language are based on discrete values like words, characters, bytes, etc., so it is hard to apply GANs for natural language applications. Future research of course is being carried out for applying GANs to other areas.

## 8 Conclusion

In this paper, we review GAN-variants based on performance improvement offered in terms of higher image quality, more diverse images and more stable training. We review the current state of GAN-related research from an architecture and loss basis. Current state-of-art GANs models such as BigGAN and PROGAN are able to produce high quality images and diverse images in the computer vision field. However, research that applies GANs to video is limited. Moreover, GAN-related research in other areas such as time series generation and natural language processing lags that for computer vision in terms of performance and capability. We conclude that there are clearly opportunities for future research and application in these fields in particular.

## Acknowledgment

This work is funded as part of the Insight Centre for Data Analytics which is supported by Science Foundation Ireland under Grant Number SFI/12/RC/2289. The authors would like to thank the blogs posted on Zhihu, Quora, Medium etc., which have provided informative insight on GAN-related research.

## References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[2] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Advances in neural information processing systems*, 2016, pp. 469–477.

[3] 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.

[4] M. O. Turkoglu, L. Spreeuwers, W. Thong, and B. Kicanaoglu, "A layer-based sequential framework for scene generation with GANs," in *Thirty-Third AAAI Conference on Artificial Intelligence*, Honolulu, Hawaii, United States, 2019.

[5] H. Wu, S. Zheng, J. Zhang, and K. Huang, "GP-GAN: Towards realistic high-resolution image blending," *arXiv preprint arXiv:1703.07195*, 2017.

[6] J. Pan, C. C. Ferrer, K. McGuinness, N. E. O'Connor, J. Torres, E. Sayrol, and X. Giro-i Nieto, "SalGAN: Visual saliency prediction with generative adversarial networks," *arXiv preprint arXiv:1701.01081*, 2017.

[7] G. K. Dziugaite, D. M. Roy, and Z. Ghahramani, "Training generative neural networks via maximum mean discrepancy optimization," *arXiv preprint arXiv:1505.03906*, 2015.

[8] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool, "Pose guided person image generation," in *Advances in Neural Information Processing Systems*, 2017, pp. 406–416.

[9] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," in *Advances In Neural Information Processing Systems*, 2016, pp. 613–621.

[10] C. Yang, X. Lu, Z. Lin, E. Shechtman, O. Wang, and H. Li, "High-resolution image inpainting using multi-scale neural patch synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6721–6729.

[11] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in *Proceedings of the 34th International Conference on Machine Learning*, vol. 70. JMLR, 2017, pp. 2642–2651.

[12] Y. Li, K. Swersky, and R. Zemel, "Generative moment matching networks," in *International Conference on Machine Learning*, 2015, pp. 1718–1727.

[13] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative visual manipulation on the natural image manifold," in *European Conference on Computer Vision*. Springer, 2016, pp. 597–613.

[14] C. Lassner, G. Pons-Moll, and P. V. Gehler, "A generative model of people in clothing," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 853–862.

[15] W. Fedus, I. Goodfellow, and A. M. Dai, "MaskGAN: Better text generation via filling in the _ ." *arXiv preprint arXiv:1801.07736*, 2018.

[16] Z. Yang, J. Hu, R. Salakhutdinov, and W. Cohen, "Semi-supervised QA with generative domain-adaptive nets," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada, 2017, pp. 1040–1050.

[17] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. R. Salakhutdinov, "Good semi-supervised learning that requires a bad GAN," in *Advances in neural information processing systems*, 2017, pp. 6510–6520.

[18] N. Jetchev, U. Bergmann, and R. Vollgraf, "Texture synthesis with spatial generative adversarial networks," *arXiv preprint arXiv:1611.08207*, 2016.

[19] C. Donahue, J. McAuley, and M. Puckette, "Synthesizing audio with generative adversarial networks," *arXiv preprint arXiv:1802.04208*, 2018.

[20] K. G. Hartmann, R. T. Schirrmeister, and T. Ball, "EEG-GAN: Generative adversarial networks for electroencephalographic brain signals," *arXiv preprint arXiv:1806.01875*, 2018.

[21] C. Esteban, S. L. Hyland, and G. Rätsch, "Real-valued (medical) time series generation with recurrent conditional GANs," *arXiv preprint arXiv:1706.02633*, 2017.

[22] D. Li, D. Chen, L. Shi, B. Jin, J. Goh, and S.-K. Ng, "MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks," *arXiv preprint arXiv:1901.04997*, 2019.

[23] E. Brophy, Z. Wang, and T. E. Ward, "Quick and easy time series generation with established image-based GANs," *arXiv preprint arXiv:1902.05624*, 2019.

[24] W. Zhu, X. Xiang, T. D. Tran, and X. Xie, "Adversarial deep structural networks for mammographic mass segmentation," *arXiv preprint arXiv:1612.05970*, 2016.

[25] P. Luc, C. Couprie, S. Chintala, and J. Verbeek, "Semantic segmentation using adversarial networks," *arXiv preprint arXiv:1611.08408*, 2016.

[26] H. Dong, S. Yu, C. Wu, and Y. Guo, "Semantic image synthesis via adversarial learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5706–5714.

[27] Z. Qiu, Y. Pan, T. Yao, and T. Mei, "Deep semantic hashing with generative adversarial networks," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2017, pp. 225–234.

[28] N. Souly, C. Spampinato, and M. Shah, "Semi supervised semantic segmentation using generative adversarial network," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5688–5696.

[29] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," *arXiv preprint arXiv:1812.04948*, 2018.

[30] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional GANs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8798–8807.

[31] B. Poole, A. A. Alemi, J. Sohl-Dickstein, and A. Angelova, "Improved generator objectives for GANs," *arXiv preprint arXiv:1612.02780*, 2016.

[32] J. Choe, S. Park, K. Kim, J. Hyun Park, D. Kim, and H. Shim, "Face generation for low-shot learning using generative adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1940–1948.

[33] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5907–5915.

[34] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *arXiv preprint arXiv:1703.10593v6*, 2017.

[35] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, "Toward multimodal image-to-image translation," in *Advances in Neural Information Processing Systems*, 2017, pp. 465–476.

[36] M. Tomei, M. Cornia, L. Baraldi, and R. Cucchiara, "Art2Real: Unfolding the reality of artworks via semantically-aware image-to-image translation," *arXiv preprint arXiv:1811.10666*, 2018.

[37] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 700–708.

[38] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[39] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8789–8797.

[40] S. Ma, J. Fu, C. Wen Chen, and T. Mei, "DA-GAN: Instance-level image translation by deep attention generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5657–5666.

[41] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2017, pp. 105–114.

[42] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy, "ESRGAN: Enhanced super-resolution generative adversarial networks," in *European Conference on Computer Vision Workshop*, 2018.

[43] D. Mahapatra, B. Bozorgtabar, S. Hewavitharanage, and R. Garnavi, "Image super resolution using generative adversarial networks and local saliency maps for retinal image analysis," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2017, pp. 382–390.

[44] D. Mahapatra, B. Bozorgtabar, and R. Garnavi, "Image super-resolution using progressive generative adversarial networks for medical image analysis," *Computerized Medical Imaging and Graphics*, vol. 71, pp. 30–39, 2019.

[45] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," *arXiv preprint arXiv:1801.07892*, 2018.

[46] 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 of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5485–5493.

[47] B. Dolhansky and C. Canton Ferrer, "Eye in-painting with exemplar generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7902–7911.

[48] Z. Chen, S. Nie, T. Wu, and C. G. Healey, "High resolution face completion with multiple controllable attributes via fully end-to-end progressive generative adversarial networks," *arXiv preprint arXiv:1801.07632*, 2018.

[49] Y. Li, S. Liu, J. Yang, and M.-H. Yang, "Generative face completion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3911–3919.

[50] J. Kossaifi, L. Tran, Y. Panagakis, and M. Pantic, "GANGAN: Geometry-aware generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 878–887.

[51] Q. Dai, Q. Li, J. Tang, and D. Wang, "Adversarial network embedding," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[52] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, "On convergence and stability of gans," *arXiv preprint arXiv:1705.07215*, 2017.

[53] Y. Li, A. Schwing, K.-C. Wang, and R. Zemel, "Dualing GANs," in *Advances in Neural Information Processing Systems*, 2017, pp. 5606–5616.

[54] A. Borji, "Pros and cons of GAN evaluation measures," *Computer Vision and Image Understanding*, vol. 179, pp. 41–65, 2019.

[55] Q. Xu, G. Huang, Y. Yuan, C. Guo, Y. Sun, F. Wu, and K. Weinberger, "An empirical study on evaluation metrics of generative adversarial networks," *arXiv preprint arXiv:1806.07755*, 2018.

[56] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein GANs," in *Advances in Neural Information Processing Systems*, 2017, pp. 5767–5777.

[57] 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. 6626–6637.

[58] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *Journal of Machine Learning Research*, vol. 13, no. Mar, pp. 723–773, 2012.

[59] Z. Wang, G. Healy, A. F. Smeaton, and T. E. Ward, "Use of neural signals to evaluate the quality of generative adversarial network performance in facial image generation," *arXiv preprint arXiv:1811.04172, Submitted to Cognitive Computation*, March, 2019.

[60] Z. Wang, Q. She, A. F. Smeaton, T. E. Ward, and G. Healy, "Neuroscore: A brain-inspired evaluation metric for generative adversarial networks," *arXiv preprint arXiv:1905.04243*, 2019.

[61] S. Barratt and R. Sharma, "A note on the inception score," *arXiv preprint arXiv:1801.01973*, 2018.

[62] L. Theis, A. v. d. Oord, and M. Bethge, "A note on the evaluation of generative models," *arXiv preprint arXiv:1511.01844*, 2015.

[63] K. Kurach, M. Lucic, X. Zhai, M. Michalski, and S. Gelly, "The GAN landscape: Losses, architectures, regularization, and normalization," *arXiv preprint arXiv:1807.04720*, 2018.

[64] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv preprint arXiv:1506.03365*, 2015.

[65] Z. Liu, P. Luo, X. Wang, and X. Tang, "Large-scale celebfaces attributes (celeba) dataset," *Retrieved August*, vol. 15, p. 2018, 2018.

[66] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[67] Y. Yoshida and T. Miyato, "Spectral norm regularization for improving the generalizability of deep learning," *arXiv preprint arXiv:1705.10941*, 2017.

[68] S. Hitawala, "Comparative study on generative adversarial networks," *arXiv preprint arXiv:1801.04271*, 2018.

[69] 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.

[70] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 53–65, 2018.

[71] Y. Hong, U. Hwang, J. Yoo, and S. Yoon, "How generative adversarial networks and their variants work: An overview," *ACM Computing Surveys (CSUR)*, vol. 52, no. 1, p. 10, 2019.

[72] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[73] D. Berthelot, T. Schumm, and L. Metz, "BEGAN: Boundary equilibrium generative adversarial networks," *arXiv preprint arXiv:1703.10717*, 2017.

[74] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[75] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, p. 107, 2017.

[76] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," *arXiv preprint arXiv:1605.05396*, 2016.

[77] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[78] E. L. Denton, S. Chintala, R. Fergus *et al.*, "Deep generative image models using a laplacian pyramid of adversarial networks," in *Advances in neural information processing systems*, 2015, pp. 1486–1494.

[79] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[80] P. Burt and E. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Transactions on communications*, vol. 31, no. 4, pp. 532–540, 1983.

[81] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[82] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," *arXiv preprint arXiv:1609.03126*, 2016.

[83] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[84] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," *arXiv preprint arXiv:1809.11096*, 2018.

[85] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[86] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," *arXiv preprint arXiv:1805.08318*, 2018.

[87] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," *arXiv preprint arXiv:1701.04862*, 2017.

[88] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *arXiv preprint arXiv:1701.07875*, 2017.

[89] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *International journal of computer vision*, vol. 40, no. 2, pp. 99–121, 2000.

[90] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *2017 IEEE International Conference on Computer Vision*. IEEE, 2017, pp. 2813–2821.

[91] 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.

[92] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, "Unrolled generative adversarial networks," *arXiv preprint arXiv:1611.02163*, 2016.

[93] G.-J. Qi, "Loss-sensitive generative adversarial networks on lipschitz densities," *arXiv preprint arXiv:1701.06264*, 2017.

[94] T. Che, Y. Li, A. P. Jacob, Y. Bengio, and W. Li, "Mode regularized generative adversarial networks," *arXiv preprint arXiv:1612.02136*, 2016.

[95] J. H. Lim and J. C. Ye, "Geometric GAN," *arXiv preprint arXiv:1705.02894*, 2017.

[96] A. Jolicoeur-Martineau, "The relativistic discriminator: a key element missing from standard GAN," *arXiv preprint arXiv:1807.00734*, 2018.

[97] B. K. Sriperumbudur, K. Fukumizu, A. Gretton, B. Schölkopf, and G. R. Lanckriet, "On integral probability metrics,$\phi$-divergences and binary classification," *arXiv preprint arXiv:0901.2698*, 2009.

[98] A. Müller, "Integral probability metrics and their generating classes of functions," *Advances in Applied Probability*, vol. 29, no. 2, pp. 429–443, 1997.

[99] T. Donchev and E. Farkhi, "Stability and euler approximation of one-sided lipschitz differential inclusions," *SIAM journal on control and optimization*, vol. 36, no. 2, pp. 780–796, 1998.

[100] L. Armijo, "Minimization of functions having lipschitz continuous first partial derivatives," *Pacific Journal of mathematics*, vol. 16, no. 1, pp. 1–3, 1966.

[101] A. Goldstein, "Optimization of lipschitz continuous functions," *Mathematical Programming*, vol. 13, no. 1, pp. 14–22, 1977.

[102] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. IEEE, 2009, pp. 248–255.

[103] Y. Luo, X. Cai, Y. Zhang, J. Xu *et al.*, "Multivariate time series imputation with generative adversarial networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 1596–1607.

[104] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence generative adversarial nets with policy gradient," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[105] D. Bahdanau, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, and Y. Bengio, "An actor-critic algorithm for sequence prediction," *arXiv preprint arXiv:1607.07086*, 2016.

[106] J. Li, W. Monroe, A. Ritter, M. Galley, J. Gao, and D. Jurafsky, "Deep reinforcement learning for dialogue generation," *arXiv preprint arXiv:1606.01541*, 2016.

TABLE 1
Summary of loss-variant for GANs.

| Author & Year | GAN type | Pros | Cons |
|---|---|---|---|
| Goodfellow et al. 2014 | Original GAN | 1. Generates samples very fast.<br>2. Able to deal with a sharp probability distribution. | 1. Vanishing gradient for $G$.<br>2. Mode collapse.<br>3. Resolution of generated images is very low. |
| Chen et al. 2016 | MRGAN | 1. Improves mode diversity.<br>2. Stabilizes GAN training. | 1. Generated image quality is low.<br>2. Does not solve vanishing gradient problem for $G$.<br>3. Only tested on CelebA dataset. Has not been tested on more diverse image datasets e.g., CIFAR and ImageNet. |
| Nowozin et al. 2016 | f-GAN | 1. Provides a unified framework based on f-divergence. | 1. Has not specified stability for different f-divergence functions. |
| Arjovsky et al. 2017 | WGAN | 1. Solves the vanishing gradient problem.<br>2. Improves image quality.<br>3. Solves the mode collapse problem. | 1. Large weight clipping causes longer time of convergence and small weights Clipping causes a vanishing gradient.<br>2. Weight clipping reduces the capacity of the model and limits the capability to model complex functions.<br>3. A very deep WGAN does not converge easily. |
| Gulrajani et al. 2017 | WGAN-GP | 1. Converges much faster than WGAN.<br>2. Model is more stable during training.<br>3. Able to use deeper GAN to model more complex function. | 1. Cannot use batch normalization because gradient penalization is done for each sample in the batch. |
| Mao et al. 2017 | LSGAN | 1. Remedy the vanishing gradient and stabilized GAN training.<br>2. Improves the mode diversity for the model.<br>3. Easy to implement. | 1. Generated samples are pushed to decision boundary instead of real data, which may affect the generated image quality. |
| Qi. 2017 | LS-GAN | 1. Solves the vanishing gradient problem.<br>2. Solves the mode collapse problem. | 1. Difficult to implement. Lots of components have to be carefully designed for the loss function.<br>2. The margin added between real samples and generated samples may affect the quality of the generated images. |
| Lim et al. 2017 | Geometric GAN | 1. Less mode collapse.<br>2. More stable training.<br>3. Converges to the Nash equilibrium between the discriminator and generator. | 1. Has not demonstrated capability in the face of the vanishing gradient problem.<br>2. Experimental tests have to be done on more complex datasets e.g., ImageNet. |
| Metz et al. 2018 | Unrolled GAN | 1. Solves the mode collapse problem.<br>2. Demonstrates that high order gradient information can help in training.<br>3. Improves GAN training stability. | 1. The quality of the generated images are low. |
| Martineau. 2018 | RGAN | 1. Solves the vanishing gradient problem.<br>2. Unified framework for IPM-based GANs.<br>3. Solves the mode collapse problem. | 1. Lack of mathematical implications of adding relativism to GANs.<br>2. Has not done a survey on which IPM-based GAN will achieve the best performance through the addition of this relativism. |
| Miyato et al. 2018 | SN-GAN | 1. Computationally light and easy to implement on existing GANs.<br>2. Improves image quality and solves mode collapse.<br>3. Stabilizes GAN training and solves the vanishing gradient problem. | 1. Requires testing on more complex image datasets. |

## SUPPLEMENTARY MATERIAL

Details of searched papers are included in Fig. 19 and Fig. 20. Figure 19 illustrates the number of papers in each

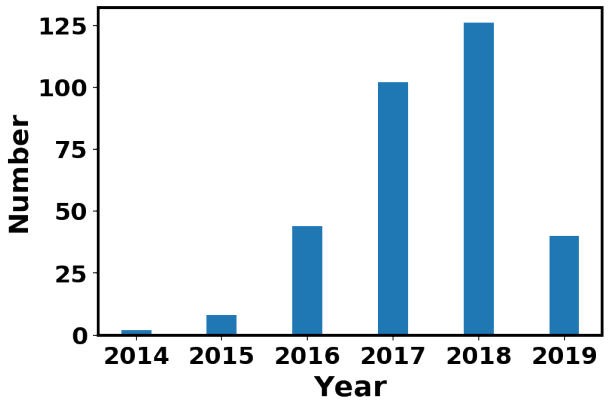



Fig. 19. Number of papers in each year from 2014 to 17th May 2019.

year from 2014 to 2019. It can be seen that the number of papers increases each year from 2014 to 2018. As our search ends up on 17th May 2019, this number can not represent the overall number of papers in 2019. Especially there are several upcoming top-tier conferences e.g., CVPR, ICCV, NeurIPS, and ICML, where much more papers may come out later this year. Even given this situation, the number of papers in 2019 is close to that in 2016. It can be noticed that there is significant rise of papers in 2016 and 2017. Indeed we see lots of exciting research in these two years e.g., CoGAN, f-GAN in 2016 and WGAN, PROGAN in 2017, which pushes the GANs research and exposes GANs to the public. In 2018, GANs still attracts lots of attention and the number of papers is more than that in previous years.

Figure 20 illustrates the number of papers published on

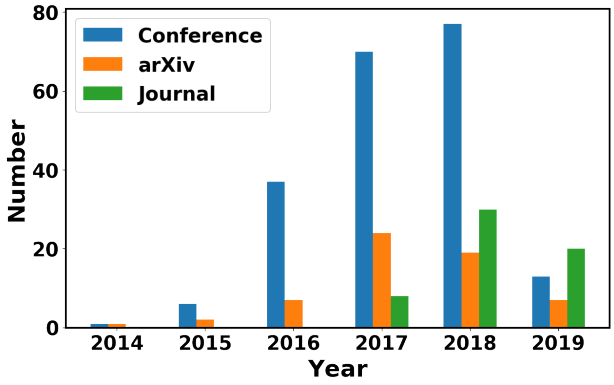



Fig. 20. Categories of papers from 2014 to 17th May 2019. Papers are categorized as conference, arXiv and journal.

three repositories, namely conference, arXiv and journal. Conference takes the largest amount from 2015 to 2018 and dramatic increase appears in 2016 and 2017. As mentioned before, there are several top-tier upcoming conferences later this year, conference supposes to take the lead on the number of papers in 2019. Papers published on Journal starts to increase since from 2017, which may be caused by the reviewing duration for a journal paper is longer than a conference paper and of course much longer than an arXiv paper. As GANs are well-developed and well-known to researchers from different areas today, number of journal papers related to GANs supposes to maintain the increasing tendency in 2019. It is interesting that number of arXiv pre-prints reaches the peak in 2017 and then starts to descend. We guess this is caused by more and more papers are accepted by conference and journal so arXiv pre-prints claim the publication details, which leads to the decreasing number of pre-prints on arXiv. This indicates higher quality of GANs research in recent years from the other side. Figure 21 gives an illustration on the percentage

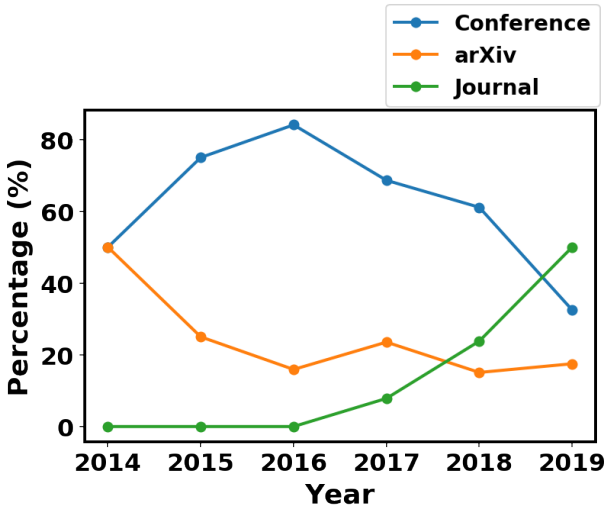



Fig. 21. Percentages of each category take account the total number of papers in each year.

of each category taking account the total number of papers in each year. Supporting results in Fig. 20, tendency of number of journal papers keeps going up. Percentage of number of conference papers reaches peak at 2016 then begins to descend. It should be noted that this does not mean the decrease of number of conference papers. This is due to other categories (i.e., arXiv and journal papers) start to increase.

A detail of searched papers are listed on this link: *https://cutt.ly/GAN-CV-paper.*