Loss Functions of Generative Adversarial Networks (GANs): Opportunities and Challenges

Zhaoqing Pan, *Senior Member, IEEE*, Weijie Yu, *Student Member, IEEE*, Bosi Wang, *Student Member, IEEE*, Haoran Xie, *Senior Member, IEEE*, Victor S. Sheng, *Senior Member, IEEE*, Jianjun Lei, *Senior Member, IEEE*, and Sam Kwong, *Fellow, IEEE*

(Survey Paper)

***Abstract*—Recently, the Generative Adversarial Networks (GANs) are fast becoming a key promising research direction in computational intelligence. To improve the modeling ability of GANs, loss functions are used to measure the differences between samples generated by the model and real samples, and make the model learn towards the goal. In this paper, we perform a survey for the loss functions used in GANs, and analyze the pros and cons of these loss functions. Firstly, the basic theory of GANs, and its training mechanism are introduced. Then, the loss functions used in GANs are summarized, including not only the objective functions of GANs, but also the application-oriented GANs' loss functions. Thirdly, the experiments and analyses of representative loss functions are discussed. Finally, several suggestions on how to choose appropriate loss functions in a specific task are given.**

***Index Terms*—Loss functions, generative adversarial networks (GANs), deep learning, machine learning, computational intelligence.**

I. Introduction

RECENTLY, Artificial Intelligence (AI) [1], [2] earns a broad prospect not only in scientific research but also in practical applications. As a kind of machine learning technologies, deep learning realizes AI in computing systems by learning data representations with multiple levels of abstraction. The traditional machine learning models such as [3] require people design method to extract features manually, while deep learning is one kind of representation learning [4] methods that extract high-dimensional information from raw data through a series of non-linear models.

Unsupervised learning, as a form of machine learning, is a direction that people have been working on. For unsupervised learning tasks, how to build a generative model is the main topic of research. Early contributions (e.g., Restricted Boltzmann Machines [5], Deep Belief Networks (DBNs) [6], Deep Boltzmann Machines [7]) are usually based on maximum likelihood estimation, Markov chains, and approximate inference. However, these studies have significant limitations due to their high computational complexity and low generalization ability. In recent years, as one of the novel generative models, Generative Adversarial Networks (GANs) [8] have received a widespread attention. The network of GANs consists of a generative model and a discriminative model. During the adversarial training process, the generative model learns the probability distribution of real data, and generates fake samples that can deceive the discriminative model. At the same time, the discriminative model needs to distinguish fake samples from real ones. It does not require Markov chain or approximate inference, and has a good generative ability in computer vision, natural language processing and many other research areas.

In order to reduce the training error of models, and accelerate the converging process of networks, the optimization of loss functions is a key point to be studied at present. The purpose of this survey is to analyze and summarize the loss functions of GANs, including not only the adversarial loss functions of GANs, but also other loss functions used by GANs in different applications.

At present, there have been many surveys on GANs [9]–[12], and these works have introduced and discussed GANs in details, including algorithms, model structures, applications and deficiencies. In this survey, the pros and cons of GANs are analyzed through a loss function point of view. Unlike previous surveys, we not only provide these loss functions, but also test the performance of these loss functions in different tasks through quantitative assessments. By using different loss functions in the same task, readers can intuitively understand the advantages and problems of them, and select suitable loss for their tasks.

The rest of this survey is organized as follows. Section II introduces the principle of GANs, including its basic theory, and

Manuscript received December 10, 2019; revised April 3, 2020; accepted April 24, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61971232, in part by the Six Talent Peaks Project of Jiangsu Province under Grant XYDXXJS-041, in part by the Natural Science Foundation of Tianjin Under Grant 18ZXZNGX00110 and 18JCJQJC45800. Paper no. TETCI-2019-0268. *(Weijie Yu and Bosi Wang contribute equally to this work) (Corresponding author: Jianjun Lei.)*

Zhaoqing Pan is with the School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China, and also with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China (e-mail: zqpan3-c@my.cityu.edu.hk).

Weijie Yu and Bosi Wang are with the School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China (e-mail: weijieyu@nuist.edu.cn; bosi@nuist.edu.cn).

Haoran Xie is with the Department of Computing and Decision Sciences, Lingnan University, Hong Kong, China (e-mail: hxie@eduhk.hk).

Victor S. Sheng is with the Department of Computer Science, Texas Tech University, Lubbock, TX 79409, USA (e-mail: victor.sheng@ttu.edu).

Jianjun Lei is with the School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China (e-mail: jjlei@tju.edu.cn).

Sam Kwong is with the Department of Computer Science, City University of Hong Kong, Hong Kong, China (e-mail: cssamk@cityu.edu.hk).

Digital Object Identifier 10.1109/TETCI.2020.2991774

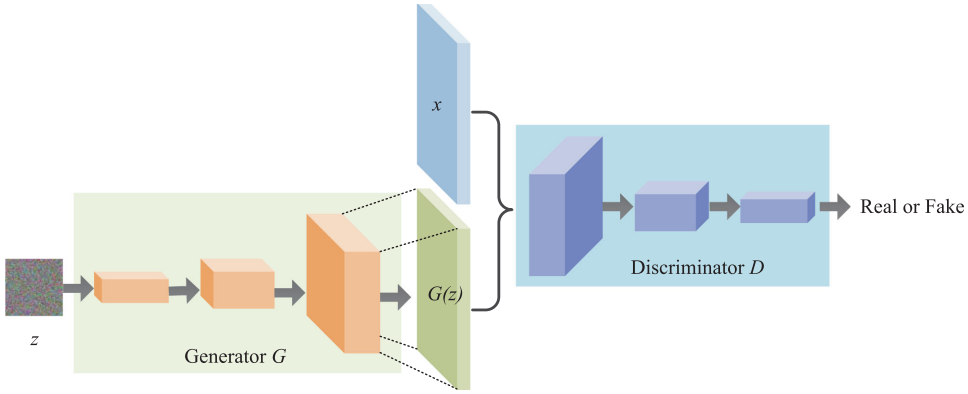


Fig. 1. The Illustration of Generative Adversarial Networks.

training mechanisms. Section III presents the loss functions used in GANs, which is made up of two parts. One part introduces the adversarial loss function optimization based GANs, and the other part introduces the loss functions added in GANs for different tasks. In Section IV, the representative loss functions are selected for comparisons and analyses. Section V discusses how to choose appropriate loss functions for different tasks. Finally, Section VI draws conclusions for this survey.

II. Generative Adversarial Networks

As a deep generative model, GANs are widely used in various applications. This section presents the basic theory of GANs, and its training mechanism.

A. Basic Theory

Generative adversarial networks (GANs) were proposed by Goodfellow *et al.* [8] in 2014. The architecture of GANs is shown in Fig. 1. It consists of two parts: a generative network G and a discriminative network D. The generator G takes a random noise vector z which obeys Gaussian or uniform distribution as input, and maps z to a new probability distribution to obtain fake samples $G(z)$. At the same time, the inputs of the discriminator have two parts: the fake samples $G(z)$ generated by the generator G, and the real samples x obtained from the real datasets. As a binary classifier, the discriminator D gives the probability that the input sample is from the real datasets rather than the generated datasets. Based on the game theory, the goal of the generator is to generate samples as realistic as possible to fraud the discriminator D, and the goal of the discriminator is to distinguish fake samples from the real samples. These two networks compete with each other, when the discriminator D cannot distinguish the authenticity of the sample, the generator training model finally learns the distribution of the real samples.

B. Training Mechanism

During the training process, the generator and the discriminator alternately train their networks. The objective function of GANs can be transformed into a minimax game, as shown in Eq. (1),

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

where x denotes the real data distribution from $p_{data}(x)$, $\mathbb{E}$ denotes the expectation, z denotes the vector from the random noise distribution $p_z(z)$, $G(z)$ and $D(x)$ denote the samples generated by the generator, and the probability that D discriminates x as real data, respectively. $D(G(z))$ denotes the probability that D determines the data generated by G. For the generator G, in order to fraud D, the discriminator probability $D(G(z))$ needs to be maximized, so $\log(1-D(G(z)))$ will be minimized. For the discriminator D, a cross entropy function is used to distinguish between $G(z)$ and x, and D wants $V(D,G)$ to be maximized.

In practice, G will be fixed firstly, and the parameters of discriminator D are updated to maximize the accuracy of D. And then, D is fixed to optimize G. In [8], the optimal condition of the discriminator is $D^*(x) = \frac{p_{data}(x)}{p_{data}(x)+p_g(x)}$. After the discriminator is optimized, the objective function of generator is equivalent to optimizing the Jenson-Shannon Divergence (JSD) between distributions $p_{data}(x)$ and $p_g(x)$. When G and D have a sufficient capacity, the model will converge, and these two parts will reach the Nash equilibrium. At this time, $p_{data}(x) = p_g(x)$, and the discriminator cannot determine the differences between these two distributions.

III. Loss Functions of Generative Adversarial Networks

In the previous section, we mentioned that the objective function of regular GANs is to minimize the JS divergence between $p_{data}(x)$ and $p_g(x)$. In fact, for different tasks, various kinds of loss functions have been proposed to improve the generative quality of the model. In this section, we will show the problems of the regular GANs, and the optimization methods for loss functions in details.

TABLE I
CLASSIFICATION OF ADVERSARIAL LOSS FUNCTION OPTIMIZATION-BASED GANS

f-divergence-based methods	f-GAN [14]; Least-square GAN [15]	
Integral Probability Metrics (IPMs)-based methods	Wasserstein distance	WGAN [13]; WGAN-GP [16]; Loss-sensitive GAN [17]
	Maximum Mean Discrepancy (MMD)	GMMN [18]; MMDGAN [19]
Other loss function methods	EBGAN [20]; BEGAN [21]	

A. Adversarial Loss Function Optimization-Based GANs

The main problem of these regular GANs is that the training is unstable, and easily falls into mode collapse. The work [13] points out that when the overlap between $p_{data}(x)$ and $p_g(x)$ is negligible, the JS divergence is a constant. At this point, the generator will have a vanishing gradient problem, which causes the GANs cannot converge to the Nash equilibrium. To tackle this problem, various solutions have been proposed, and these solutions can be classified into three categories: f-divergence-based methods, integral probability metrics (IPMs)-based methods, and other methods. The classification details are tabulated in Table I.

1) f-Divergence-Based Methods: The f-divergences are general term for all kinds of divergence in the probability theory [22]. This function $D_f(P\|Q)$ is used to measure the differences between two distributions P and Q with a specific convex function f. If the distributions P and Q are absolutely continuous with respect to the measure dx in the domain χ, they can be represented by a convex function which satisfies $f(1) = 0$. The function f is defined as,

$$D_f(P\|Q) = \int_{\chi} Q(x) f\left(\frac{P(x)}{Q(x)}\right) dx. \tag{2}$$

Suppose that for all x, if $P(x) = Q(x)$, the divergence will go to 0. At this time, there is no difference between these two distributions. When these two distributions are different, D_f will get a positive value. It should be pointed out that the JS divergence used in the regular GANs is also a kind of f-divergence. Next, two f-divergence-based works are introduced, including f-GAN [14] and least-square GAN [15].

f-GAN. By optimizing the divergence, the distance between the probability distributions can be measured in different ways. Inspired by f-divergence, Nowozin *et al.* [14] proposed the f-GAN, which uses different divergences as the objective function of GANs. And they have proved that any f-divergence can be used in the GANs frameworks. According to the Fenchel conjugate [23], each convex function f has a corresponding conjugate function f^*. In order to facilitate the estimation of Eq. (2), they converted it into a new form by a convex conjugate function $f^*(t) = \sup_{x \in dom_f}\{xt - f(x)\}$. The converted Eq. (2) is formulated as,

$$\begin{aligned} D_f(P\|Q) &= \int_{\chi} Q(x) \sup_{t \in \mathrm{dom}\, f^*} \left(t\frac{P(x)}{Q(x)} - f^*(t)\right) dx \\ &\geq \sup_{T \in \Gamma} \left\{ \int_x P(x)T(x)dx - \int_x Q(x) f^*\left(T(x)\right) dx \right\} \\ &= \sup_{T \in \Gamma} \left\{ \mathbb{E}_{x \sim P}\left[T(x)\right] - \mathbb{E}_{x \sim Q}\left[f^*\left(T(x)\right) \right] \right\}, \end{aligned} \tag{3}$$

TABLE II
DIVERGENCE COMMONLY USED IN THE OBJECTIVE FUNCTION OF GANS

Divergence	Generator $f(x)$
Jenson-Shannon Divergence	$x\log x - (x+1)\log(x+1)$
Kullback-Leibler Divergence	$x\log x$
Reverse Kullback-Leibler Divergence	$-\log x$
Person χ^2	$(x-1)^2$
Neyman χ^2	$\frac{(1-x)^2}{x}$
Total variation	$\frac{1}{2}\lvert x-1\rvert$
Squared Hellinger	$(\sqrt{x}-1)^2$
Jeffrey	$(x-1)\log x$

where Γ represents an arbitrary function class which satisfies $\chi \to \mathbb{R}$, and $\mathbb{E}$ represents the expectation. They assumed a function T whose input is x and the output is t. Then t can be replaced by $T(x)$. For a generative-adversarial model, if f-divergence is used to represent the differences between two distributions $p_{data}(x)$ and $p_g(x)$, $D_f(p_{\text{data}}\|p_g)$ is defined as,

$$D_f(p_{data}\|p_g) = \sup_{T \in \Gamma} \left\{ \mathbb{E}_{x \sim p_{data}}\left[T(x)\right] - \mathbb{E}_{x \sim p_g}\left[f^*\left(T(x)\right) \right] \right\}. \tag{4}$$

We hope to minimize the divergence between these two distributions, which means to minimize $D_f(p_{\text{data}}\|p_g)$. At this time, we need to minimize the generator function f, and maximize the lower bound of Eq. (4). Therefore, f-divergence is a generalization of GAN models. For any generator function f that satisfies the conditions, we can create a corresponding GAN framework. Based on the [14], the name and generator functions of f-divergence commonly used in GAN frameworks are shown in Table II.

f-GAN has proven that any f-divergence can be used in the GAN frameworks. Among these divergences, the least-square GAN which uses the Person χ^2 divergence, has been widely used. It tries to create a more stable, faster convergence, and higher quality producing network by minimizing the least square loss function. More details of the least-square GAN will be introduced in the next paragraph.

Least-square GAN. Since the sigmoid cross entropy is used as the loss function of the discriminator in regular GANs, the quality of generated samples are not very high. The sigmoid cross entropy loss can correctly distinguish the authenticity of samples. When the fake sample successfully deceives the discriminator, it will be classified as a real one at the decision boundary. At this time, they may still be far from the real distribution, but the generator will no longer iterate.

In the least-square GAN, the objective function is a square error, and the objective function for least-square GAN is described in Eqs. (5) and (6),

$$\min_D V(D) = \frac{1}{2}\mathbb{E}_{x\sim p_{data}}\left[(D(x)-b)^2\right] + \frac{1}{2}\mathbb{E}_{z\sim p_z(z)}\left[\left(D\left(G(z)\right)-a\right)^2\right], \tag{5}$$

$$\min_G V(G) = \frac{1}{2}\mathbb{E}_{z\sim p_z(z)}\left[\left(D\left(G(z)\right)-c\right)^2\right], \tag{6}$$

where the labels of the generated sample and the real sample are represented by a and b, respectively. The goal of discriminator D is to minimize the GAN objective function. Meanwhile, the value c represents that D believes the sample generated by G is a real one.

For those samples that are far from the decision boundary but are judged to be true, the least-square loss function will penalize them, and push them to the decision boundary, this approach makes the generated samples more realistic. In Eqs. (5) and (6), when $b-c=1$, $b-a=2$, the objective function is equivalent to the Person χ^2 divergence. During the training process, a, b and c are set to -1, 1 and 0, respectively.

2) Integral Probability Metrics (IPMs)-based methods: In a specific function class F, the integral probability metric (IPM) [24] defines a critic function f. Defining a measurable space $\chi \subset R^d$, $P(\chi)$ represents the probability measures on this space. A set of bounded, measurable and real-valued functions are represented in the function class F, and there are two arbitrary distributions P and Q in $P(\chi)$. The IPM is used to define the maximal measures between them, as defined in Eq. (7),

$$d_F(P,Q) = \sup_{f\in F}\left\{\mathbb{E}_{x\sim P}f(x) - \mathbb{E}_{x\sim Q}f(x)\right\}, \tag{7}$$

and the Wasserstein distance and the maximum mean discrepancy (MMD) belong to the class of IPMs, which will be discussed in the next paragraph.

(a) Wasserstein distance

In the regular GANs, if the discriminating ability of the discriminator is better, the vanishing gradient problem of the generator will be more serious. When the overlap between the generated and real samples is negligible, the divergence will be a constant. At this time, the model cannot continue to be trained. To address this problem, Arjovsky *et al.* proposed the Wasserstein GAN (WGAN) [13], using Wasserstein distance, which is also known as Earth Mover (EM) distance, instead of JS divergence to measure the distance between the real distribution $p_{\text{data}}(x)$ and generated distribution $p_g(x)$. The definition of $W(p_{data}, p_g)$ is shown in Eq. (8),

$$W(p_{data}, p_g) = \inf_{\gamma\in\Pi(p_{data},p_g)}\mathbb{E}_{(x,y)\sim\gamma}\left[\left\|x-y\right\|\right], \tag{8}$$

where $\Pi(p_{\text{data}}, p_g)$ represents the set of all possible joint distributions which is composed by $p_{\text{data}}(x)$ and $p_g(x)$. For each possible joint distribution γ, a real sample x and a generated sample y can be obtained from the $\gamma(x,y)$. And the distance of the pair of samples $\|x-y\|$ represents the expectation of the sample under the joint distribution. The lower bound that can be taken from this expectation in all possible joint distributions is defined as the Wasserstein distance.

The superiority of the Wasserstein distance, compared with the JS divergence, is that even if there is no overlap between two distributions, the Wasserstein distance still reflects the distance between them. This derivability and continuity can provide meaningful gradient information to continue updating the generator. Since it is very difficult to directly deal with the Wasserstein distance, the approach assumes that the sample subjects to the Lipschitz condition, and its probability density distribution $f(x)$ satisfies the following equation,

$$\|f(x_1) - f(x_2)\| \le K\|x_1 - x_2\|, \tag{9}$$

where the minimum value of K is called Lipschitz constant. According to the Kantorovich-Rubinstein duality, we can simplify Eq. (8) to

$$W(p_{data}, p_g) = \sup_{|f|_L\le 1}\mathbb{E}_{x\sim p_{data}}f(x) - \mathbb{E}_{x\sim p_g}f(x), \tag{10}$$

where the function f satisfies Lipschitz continuity. We use neural network to fit function f, so the objective function of WGAN is

$$\min_G\max_C \mathbb{E}_{x\sim p_{data}}C(x) - \mathbb{E}_{\tilde{x}\sim p_g}C(\tilde{x}), \tag{11}$$

where C represents the critic which needs to satisfy the Lipschitz continuity. The WGAN provides a weight clipping method to ensure that C is conducted to a certain range after each update. Unlike the discriminator D, the critic C removes the sigmoid function at the last layer. So the output of C is not a probability, but a score of its input.

The WGAN has been widely used because it is important for solving the problems of training instability and mode collapse. However, the follow-up works based on this improvement found that the weight clipping used by WGAN in the critic could still not converge, which may reduce the quality of the generated samples. Therefore, Gulrajani *et al.* [16] proposed the WGAN-GP, which uses the gradient penalty to enforce the Lipschitz constraints. The Lipschitz conditions limit the gradient of critic to no more than K. The WGAN-GP adds a penalty term that correlates the gradient with K to improve the convergence speed and the stability of the network. The discriminator loss is modified as,

$$L_D = \mathbb{E}_{\tilde{x}\sim p_g}D(\tilde{x}) - \mathbb{E}_{x\sim p_{data}}D(x) + \lambda\mathbb{E}_{\hat{x}\sim p_{\hat{x}}}\left(\left\|\nabla_{\hat{x}}D(\hat{x})\right\|_2 - 1\right)^2, \tag{12}$$

where $x_{data}\sim p_{data}$, $x_g\sim p_g$, $\varepsilon\sim\ Uniformorm[0,1]$, and $\hat{x} = \varepsilon x_{data} + (1-\varepsilon)x_g$ represents random interpolation sampling on x_{data} and x_g. In addition, the gradient penalty coefficient λ is set to 10. With almost no tuning of hyper-parameters, the WGAN-GP has a more stable performance than the original WGAN in various GAN frameworks.

The infinite modeling ability of GANs will lead to unsatisfactory results, to limit this ability, Qi *et al.* proposed the loss-sensitive GAN [17]. The loss-sensitive GAN and WGAN are both based on the Lipschitz density, but they have different purposes. The regular GANs assumed that the model has

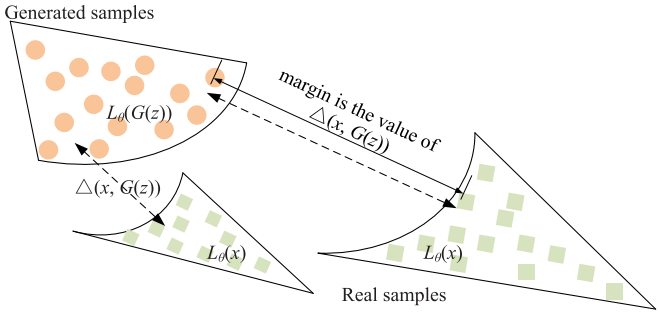


Fig. 2. The schematic diagram of the loss function in the loss-sensitive GAN. $\Delta(x, G(z))$ represents the distance between x and $G(z)$, which can be a p-Norm. With the iteration of the generator G, the similarity between the generated samples and the real samples will increase. can be more reasonably used to address the vanishing gradient problem.

a infinite modeling ability, so there is no restriction to the distribution of real samples, which may lead to the vanishing gradient problem. The loss-sensitive GAN learns a loss function $L_\theta(x)$ parameterized with θ, which can restrict the modeling ability of discriminator D. The loss function is learned by a data-dependent margin because they assumed that the loss of real data distribution should be smaller than the generated one. In this way, the loss-sensitive GAN has proved that the Lipschitz densities of real samples are similar to the densities of generated samples. The objective function of loss-sensitive GAN is defined in Eqs. (13) and (14),

$$\begin{aligned}\min_D V(D) &= \mathbb{E}_{x\sim p_{data}(x)} L_\theta(x) \\ &+ \lambda \mathbb{E}_{\substack{x\sim p_{data}(x) \\ z\sim p_z(z)}} \Big(\Delta\left(x, G(z)\right) + L_\theta(x) - L_\theta\left(G(z)\right) \Big),\end{aligned} \tag{13}$$

$$\min_G V(G) = \mathbb{E}_{z\sim p_z(z)} L_\theta\left(G(z)\right), \tag{14}$$

where $\Delta(x, G(z))$ represents the margin measuring the differences between x and $G(z)$ in terms of their losses, and λ is a balancing parameter. Fig. 2 illustrates this idea in details. It can be seen from Fig. 2, if the generated data distribution is close to the real one, it is no longer treated as a negative sample, and more efforts can be concentrated to improve the samples that are far away from the real samples. It should be noted that the margin is not a fixed constant. The margin is a similarity function defined in specific experiments such as p-Norm. When the generated sample is very close to the real one in a metric space, the margin will be vanished.

(b) Maximum mean discrepancy (MMD)

Assuming that χ is a non-empty metric space, and a class of function $f \in \mathcal{F} : \chi \to \mathbb{R}$, $X \sim p$, $Y \sim q$, the maximum mean discrepancy (MMD) [25] between p and q is defined in Eq. (15),

$$MMD[\mathcal{F}, p, q] = \sup_{f\in\mathcal{F}} \mathbb{E} f(X) - \mathbb{E} f(Y). \tag{15}$$

The reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ is an infinite dimensional space. For each $f \in \mathcal{H}$, there exists a kernel $k \in \mathcal{H}$, then f is formulated as,

$$f(x) = \Big\langle f, k\left(\cdot, x\right) \Big\rangle_{\mathcal{H}} = \sum \alpha_i k(x, x_i). \tag{16}$$

If $\mathcal{F}$ chooses RKHS space $\mathcal{H}$, μ_p represents mean embedding of p, which is calculated as follows,

$$\mu_p = \int_\chi k(x, \cdot) p(dx) \in \mathcal{H}. \tag{17}$$

For each $f \in \mathcal{H}, \mathbb{E}[f(X)] = \langle f, \mu_p \rangle_{\mathcal{H}}$, the MMD can be formulated as another mean feature matching, as shown in Eq. (18),

$$\begin{aligned} MMD[\mathcal{F}, p, q] &= \sup_{\|f\|\mathcal{H}\leq 1} \mathbb{E}_p f(x) - \mathbb{E}_q f(y) \\ &= \sup_{\|f\|\mathcal{H}\leq 1} \mathbb{E}_p \Big\langle \phi(x), f \Big\rangle_{sf} - \mathbb{E}_q \Big\langle \phi(y), f \Big\rangle_{\mathcal{H}} \\ &= \sup_{\|f\|\mathcal{H}\leq 1} \mathbb{E}_p \langle \mu_p - \mu_q, f \rangle_{\mathcal{H}} \\ &= \|\mu_p - \mu_q\|_{\mathcal{H}}, \end{aligned} \tag{18}$$

where $\phi(\cdot)$ represents $x \in \mathcal{H}$, μ_p represents $\mathbb{E}_p[\phi(x)]$ and μ_q represents $\mathbb{E}_q[\phi(y)]$. The MMD was firstly proposed for the problem of two-sample test to determine the differences between two distributions p and q. In practical applications, the square of the MMD is generally used, and it is defined as,

$$\begin{aligned} MMD[\mathcal{F}, p, q] &= \langle \mu_p - \mu_q, \mu_p - \mu_q \rangle_{\mathcal{H}} \\ &= \langle \mu_p, \mu_p \rangle + \langle \mu_q, \mu_q \rangle - 2\langle \mu_p, \mu_q \rangle \\ &= \mathbb{E}_p \Big\langle \phi(x), \phi\left(x'\right) \Big\rangle_{\mathcal{H}} - 2\mathbb{E}_{p,q} \Big\langle \phi(x), \phi(y) \Big\rangle_{\mathcal{H}} \\ &\quad + \mathbb{E}_q \Big\langle \phi(y), \phi(y') \Big\rangle_{\mathcal{H}}. \end{aligned} \tag{19}$$

We can use kernel tricks to measure a kernel function $k(x, y)$. The choice of the kernel function is various, such as linear kernel, Gaussian kernel, Laplacian kernel, etc.

Based on the fixed Gaussian kernel $k(x, y) = \exp(-\|x - y\|^2)$, Li *et al.* [18] proposed the generative moment matching networks (GMMN) to measure the discrepancy of two distributions in GANs by minimizing the MMD distance. Unlike regular GANs, the GMMN used an autoencoder instead of a discriminator to estimate the discrepancy between two distributions. During the training process, although the stability of the generated samples is improved, the training efficiency of GMMN is not satisfactory. To achieve improvements in the generalization ability and computational efficiency of GMMN, the MMDGAN [19] replaced the static fixed Gaussian kernels with the adversarial learned kernels. The adversarial learned kernel consists of a Gaussian kernel and an injective function f_ϕ, where $k(x, y) = \exp(-\|f_\phi(x) - f_\phi(y)\|^2)$. In addition, in order to enforce f_ϕ to be an injective function, they used an autoencoder in the discriminator.

3) Other objective function methods: In addition to using Lipschitz density to constrain the sample distribution, non-probability forms can also be used to measure GANs. Energy-based GAN (EBGAN) [20] is a typical one in this form. Unlike the discriminator used in the regular GANs, the discriminator

is regarded as an energy function in the EBGAN. By using this energy function, the lower energy will be attributed to the real samples, and the higher energy will be attributed to the generated samples. In the EBGAN, an autoencoder is used to estimate the reconstruction loss $D(x) = \|Decoder(Encoder(x) - x)\|$. The loss function of EBGAN is defined in Eqs. (20) and (21),

$$f_D(x, z) = D(x) + \Big[m - D\left(G(z)\right)\Big]^+, \tag{20}$$

$$f_D(z) = D\left(G(z)\right), \tag{21}$$

where $[\cdot]^+ = max(0, \cdot)$, and m represents a predefined margin that limits the ability of the discriminator to enable the generator to continually converge. In order to generate samples with better diversity, the EBGAN proposed a constraint method named Pulling-away Term (PT) in the generator loss, which is defined as,

$$f_{PT}(S) = \frac{1}{N(N-1)} \sum_i \sum_{j \neq 1} \left(\frac{S_i^T S_j}{\|S_i\| \|S_j\|} \right)^2, \tag{22}$$

where S represents the output of the coding layer in the discriminator. By increasing the value of PT, the EBGAN effectively increases the diversity of generated samples.

Combining the advantages of EBGAN and WGAN, Berthelot *et al.* [21] proposed the boundary equilibrium GAN (BEGAN) to evaluate the generative quality of generator. This approach allows GANs to achieve great training results using a simple network architecture. In addition, the BEGAN achieves a balance between image diversity and generative quality. In their model, an autoencoder architecture is used in the discriminator to compute the reconstruction loss D. The reconstruction loss aims to measure the differences of distributions between real samples and generated samples. Different from the original loss function, inspired by the WGAN, they used Wasserstein distance to optimize the Wasserstein lower bound between two distributions. The loss function of BEGAN is defined as,

$$f_D(x, z) = D(x) - k_t D\Big(\left(G(z)\right) \Big), \tag{23}$$

$$f_G = D\left(G(z)\right), \tag{24}$$

$$k_{t+1} = k_t + \lambda_k \Big(\gamma D(x) - D\left(G(z)\right) \Big), \tag{25}$$

where $\gamma = \frac{\mathbb{E}[D(G(z))]}{\mathbb{E}[D(x)]}$, and $\gamma \in [0, 1]$ represents the ratio of the expected difference between the real samples and the generated ones (the lower the γ, the lower the diversity of the generated samples). k is initialized to 0 at the beginning, so it gives the time to firstly develop the autoencoder. λ_k is the learning rate.

In summary, the generator of GANs implicitly defines a probability distribution, and uses this probability distribution to generate samples. Similarly, the training samples are obtained by continuous independent sampling on a certain probability distribution. Therefore, the goal of GANs is to make the implicit probability distribution defined by the generator close to the probability distribution of datasets. GANs reduce the difference between these two probability distributions through different distance measures to make the distribution between the generated sample and the real sample more consistent. The f-divergence-based objective functions are more dependent on the data distribution. In practice, [36] has proved that when the support of these two probability distributions is a low dimensional manifold in a high dimensional space, there is no overlap between these two distributions, or the overlap is too small. At this time, the mode will fall into the vanishing gradient problem, which will increase the difficulty of training. Therefore, this problem is inevitable for all f-divergence-based objective functions.

For IPMs-based objective functions, [37] has shown that IPMs and f-divergence are different. The total variation distance is the only non-trivial intersection between these two classes of distance measures. Irrespective of the data distributions, IPMs have better convergence rate, and stronger consistency. Based on this proof, these IPMs-based objective functions can avoid the problems caused by the f-divergence-based objective functions. This class of distance measures is continuous almost everywhere, and is derivable when these two distributions have no supports.

In addition, for the other loss function methods, the EBGAN and BEGAN are both energy-based networks. They replaced discriminator with an auto encoder. For EBGAN, it used the total variation distance. Essentially, it does not address the problem that the model convergence depends on data distribution. BEGAN used Wasserstein distance to match loss distribution instead of data distribution, so it is more stable during training, and more effective for mode collapse. More discussions on these loss functions will be given in Section IV.

B. Application-Oriented GANs' Loss Functions

Changing the adversarial loss is one way to improve the modeling ability of GANs. In addition, the application-oriented loss functions have been proposed to enhance the performance of GANs in different tasks. We select representative GANs-based applications, and introduce their contributions to the loss functions. These loss functions are tabulated in Table III.

Pixel-wise loss. Pixel-wise loss is used to compare the differences between two images on the pixel level. As a well-known group of pixel-wise loss functions, it contains l_1 (mean absolute error) loss and l_2 (mean square error) loss. These two loss functions are defined as,

$$l_1 = \frac{1}{WHC} \sum_{ijk} |I_{ijk} - I'_{ijk}|, \tag{26}$$

$$l_2 = \frac{1}{WHC} \sum_{ijk} (I_{ijk} - I'_{ijk})^2, \tag{27}$$

where W, H, C represents the width, height and channels of images, respectively. I' and I represent the generated sample and the ground truth, respectively.

GANs with pixel-wise loss have been used in multiple tasks, such as image-to-image translation [38], [39], image super-resolution [28], [40], image inpainting [41], [42], image synthesis [43], [44], and so on. Comparing two pixel-wise loss

TABLE III
SELECTED LOSS FUNCTIONS USED IN APPLICATIONS OF GANS

Loss function	Description
Pixel-wise loss	Measure image difference on pixel level, including l_1 (MAE) and l_2 (MSE). $l_1 = \frac{1}{WHC}\sum_{ijk}\lvert I_{ijk} - I'_{ijk}\rvert$, $l_2 = \frac{1}{WHC}\sum_{ijk}(I_{ijk} - I'_{ijk})^2$.
Content loss [26]	Extract high frequency information from perceptual similarity between images. $l_{content} = \frac{1}{W_{i,j}H_{i,j}}\sum_{x=1}^{W_{i,j}}\sum_{y=1}^{H_{i,j}}\left(\phi_{i,j}(I)_{x,y} - \phi_{i,j}(I')_{x,y}\right)^2$.
Texture loss [27]	Make the generated images have better texture features. $l_{texture} = \left\lVert G\left(\phi(I)\right) - G\left(\phi(I')\right)\right\rVert_2^2$.
Contextual loss [28]	For non-aligned images, considering the context of the entire image and the region with similar semantics. $l_{contextual} = -log(\frac{1}{N}\sum_j max_i CX_{ij})$.
Cycle consistency loss [29]	Allow mapping between unpaired data learning spaces. $l_{cyc} = \sum_{x\in X}\left\lVert F\left(G(x)\right) - x\right\rVert_1 + \sum_{y\in Y}\left\lVert G\left(F(x)\right) - y\right\rVert_1$.
Total variation loss [30]	Used as regularization term in conjunction with other loss functions to remove noise in the image. $l_{TV} = \frac{1}{WHC}\sum_{i,j,k}\left(\lVert\nabla_x G(x_{i,j,k})\rVert_2 + \lVert\nabla_y G(x_{i,j,k})\rVert_2\right)$.
Color loss [31]	Maintain similar brightness, major color and contrast between images. $l_{color} = \lVert I_b - I_{b'}\rVert_2^2$.
Symmetry loss [32]	Usually used in face synthesis task, to alleviate the self-occlusion problem of the synthesized images. $l_{sym} = \frac{1}{W/2\times H}\sum_{x=1}^{W/2}\sum_{y=1}^{H}\left\lvert I'_{x,y} - I'_{W-(x-1),y}\right\rvert$.
Identity preserving loss [32]	Preserve identity information between images. There are small differences in various identity preserving loss [33]–[35], one of them is, $l_{identity} = \frac{1}{WHC}\sum_{i,j,k}\left\lvert\phi(I_{i,j,k}) - \phi(I'_{i,j,k})\right\rvert$.

functions, due to the use of squared term, the l_2 loss will increase the distance between larger error and smaller error. In order words, the l_2 loss penalizes larger errors more severely, and tolerates smaller ones [45]. However, images generated by pixel-wise loss lack high-frequency details, and have overly smooth texture information [26], [46]. The pixel-wise loss takes no account of the Human Visual System (HVS) [47], and the artifacts still exist in the images.

Content loss. To consider the perceptual similarity and content quality between generated images and target images, inspired by Johnson *et al.* [46], Ledig *et al.* [26] proposed the SRGAN to introduce the content loss into GANs. The content loss is defined based on the RELU activation layer of the pre-trained VGG-19 network, and it is defined as,

$$l_{content} = \frac{1}{W_{i,j}H_{i,j}}\sum_{x=1}^{W_{i,j}}\sum_{y=1}^{H_{i,j}}\left(\phi_{i,j}(I)_{x,y} - \phi_{i,j}\left(I'\right)_{x,y}\right)^2, \tag{28}$$

where $\phi_{i,j}$ represents the feature map obtained by the j-th convolution before the i-th max-pooling layer within the 19-layer VGG network; $W_{i,j}$ and $H_{i,j}$ represent the dimension of each feature map in VGG network. l_{content} allows the low dimensional feature $\phi_{i,j}(I')_{x,y}$ to be consistent with high dimensional feature $\phi_{i,j}(I)_{x,y}$. The content loss has been widely used in many tasks [48]–[54] because it pays more attention to the perceptual quality, and conforms to the HVS.

Texture loss. Gateys *et al.* used the style reconstruction loss in texture synthesis and style transfer tasks [55], [56]. The texture loss achieves an improvement in the texture quality of generated images [27], and the texture loss is defined as,

$$l_{texture} = \left\lVert G\left(\phi(I)\right) - G\left(\phi\left(I'\right)\right)\right\rVert_2^2, \tag{29}$$

where $G(F) \in \mathbb{R}^{n\times n}$ represents the Gram matrix, which executes inner product operations on the two vectorized feature maps. $\phi(\cdot)$ represents the feature map obtained from the pre-trained 19-layer VGG network. The texture loss measures the local texture similarity between the generated and target images, so that the generated images have richer texture information. During the training process, how to choose the patch size is very important. If the texture loss is calculated on the entire image, the texture statistics will be averaged due to the diversity of the texture information. In [27], the patch size is set to 16×16 pixels.

Contextual loss. Mechrez *et al.* [28] proposed the contextual loss to measure the feature similarity between the generated images and the target images. To define the feature similarity in two images, the Cosine distance is introduced, which is shown

in Eq. (30),

$$d_{ij} = 1 - \frac{(x_j - \mu_y)(y_j - \mu_y)}{\|x_i - \mu_y\|_2 \|y_i - \mu_y\|_2}, \tag{30}$$

where $\mu_y = \frac{1}{N}\sum_j y_j$, and x_i and y_i are features obtained from the pre-trained 19-layer VGG network in the image X and target image Y, respectively. When $d_{ij} \ll d_{it}, \forall t \neq j$, x_i and y_i are considered similar to each other. Then Eq. (30) is normalized as,

$$\tilde{d}_{ij} = \frac{d_{ij}}{\min_t d_{it} + \epsilon}, \tag{31}$$

where $\epsilon = 1e-5$. The contextual similarity between these features is defined as,

$$CX_{ij} = \frac{\exp\left(\frac{1-\tilde{d}_{ij}}{h}\right)}{\sum_t \exp\left(\frac{1-\tilde{d}_{it}}{h}\right)}, \tag{32}$$

where $exp(\cdot)$ is used to shift $\tilde{d}_{ij}$ from distance to similarities, and h represents a band-width parameter which must be greater than 0. Finally, the contextual loss is defined as,

$$l_{contextual} = -log\left(\frac{1}{N}\sum_j max_i CX_{ij}\right). \tag{33}$$

According to [28] and [57], the loss function considers both context and semantics information, it can be used not only in super-resolution tasks but also in image-to-image translation tasks.

Cycle consistency loss. This loss function was firstly used in the image-to-image translation task. Zhu *et al.* [29] proposed the CycleGAN, which uses unpaired data to train space mappings. The architecture of the network is a pair of mirroring symmetric GANs to form a ring network. Since both mappings are learned at the same time, when images from one domain are transferred to the other, they should be able to transfer back. In order to prevent the model from converting the image in one domain to the same image in another domain, the CycleGAN used l_1 norm to measure the distortion between the generated image and its reconstructed image. The cycle consistency loss is defined as,

$$l_{cyc} = \sum_{x\in X}\Big\|F\left(G(x)\right) - x\Big\|_1 + \sum_{y\in Y}\Big\|G\left(F(x)\right) - y\Big\|_1, \tag{34}$$

where F is a mapping that converts x from domain X to $F(x)$ in domain Y, and G is another mapping that converts y from domain Y to $G(y)$ in domain X. In addition to the use of the cycle consistency loss in image-to-image translation tasks [29], [58], Yuan *et al.* [33] used the cycle-in-cycle mechanism to super-resolution tasks. Engin *et al.* [54] applied it to generate haze-free images. In [59] and [60], the CycleGAN is used to synthesize computed tomography (CT) images for magnetic resonance (MR). In face generation tasks, the cycle consistency loss is used as well [61].

Total variation loss. The total variation loss encourages spatial smoothness, and reduces the noise of the generated images [30]. It is defined to reduce the differences between adjacent pixels in an image, and this function is defined as,

$$l_{TV} = \frac{1}{WHC}\sum_{i,j,k}\left(\Big\|\nabla_x G\left(x_{i,j,k}\right)\Big\|_2 + \Big\|\nabla_y G\left(x_{i,j,k}\right)\Big\|_2\right), \tag{35}$$

where W, H, and C represent the width, the height and the channels of a generated image $G(x_i)$, respectively. ∇_x and ∇_y represent functions to calculate the horizontal and vertical gradient of $G(x_i)$. In actual tasks, the total variation loss is used with other loss functions together to improve the quality of generated images [26], [31]–[33], [44], [51].

Color loss. In order to have a similar color distribution between the generated images and the target images, Ignatov *et al.* [31] proposed the color loss, which applied the Gaussian blur to an image to extract high-frequency information (i.e., color, brightness and contrast), so that the colors of the image can be easily compared. Then the Euclidean distance between the representations is calculated. The color loss is defined as,

$$l_{color} = \|I_b - I'_b\|_2^2, \tag{36}$$

where I_b and I'_b are blurred images extracted from I and I' by Gaussian blur operation, respectively, and it is defined as

$$I_b(i,j) = A\sum_{k,l} I(i+k, j+l)\exp\left(-\frac{(k-\mu_x)^2}{2\sigma_x} - \frac{(l-\mu_y)^2}{2\sigma_y}\right), \tag{37}$$

where σ is a constant that constrains Gaussian blur to extract the texture and content information of the image.

Symmetry loss. The symmetry loss is always applied in face synthesis tasks [32], [62]. For producing high quality real face images, Huang *et al.* proposed a two-pathway generative adversarial network (TP-GAN) [32], which uses GANs for face frontalization. The TP-GAN uses a single face image to generate high-resolution frontal face image, and combines multiple loss functions to achieve the best results. Due to the symmetry of the human face, in order to alleviate the self-occlusion problem of the synthesized images and complement the occluded part, the symmetry loss is proposed as,

$$l_{sym} = \frac{1}{W/2 \times H}\sum_{x=1}^{W/2}\sum_{y=1}^{H}\left|I'_{x,y} - I'_{W-(x-1),y}\right|, \tag{38}$$

where I' is the predict face image generated from different poses.

Identity preserving loss. To preserve identity information, [32] proposed the identity preserving loss, which is based on the activations of the last two layers of the Light-CNN [63]. The identity preserving loss is defined as,

$$l_{identity} = \frac{1}{WHC}\sum_{i,j,k}\left|\phi(I_{i,j,k}) - \phi(I'_{i,j,k})\right|. \tag{39}$$

In addition, there are many ways to realize the identity preserving loss. For example, in face synthesis [34][64] and image synthesis tasks [33][35], they used own solutions to preserve identity information.

In conclusion, these loss functions are proposed for specific GAN-based applications. For different tasks, the common

goal is to generate more realistic samples. Among them, the pixel-wise loss is the most widely used, especially in tasks that use Peak Signal-to-Noise Ratio (PSNR) as quantitative assessment standard. Since PSNR focuses on the difference between corresponding pixels, the pixel-wise loss is suitable for these tasks. However, the samples generated by this loss lack texture information, and the samples are too smooth, so it is not friendly on the visual experience. The content loss, texture loss, and contextual loss are all proposed to enrich high frequency information and enhance texture content. They all improve the perceptual quality of generated images, and in different tasks [27], [65]–[67], they can be combined to achieve better results. It is worth noting that in quantitative assessment, especially in PSNR, their performance is not as good as the pixel-wise loss. Therefore, finding more general evaluation metrics is very important. When noise exists in the generated samples, the total variation loss can be added. At present, the weights of various combined losses are set through experiments, and how to set the weights through theory instead of experience is also a problem that needs to be faced. More analyses of these loss functions will be given through experiments in Section IV.

IV. Experiments

In this section, we select representative loss functions, and discuss their performance through experiments. Firstly, the details of datasets and experimental implementations are presented. Furthermore, the quantitative evaluation metrics are used to evaluate the experimental results. The experiments are divided into two parts. One is objective functions used in GANs, and the other is application-oriented GANs' loss functions. Moreover, the performance of loss functions according to the results are compared, and the pros and cons of these loss functions are analyzed.

A. Implementation Details and Evaluation Metrics

1) Adversarial Loss Functions Used in GANs: For the adversarial loss functions used in GANs, in order to test their performance, we run experiments on the image generation task. The datasets we selected are MNIST [68], fashion-MNIST [69] and CelebA [70]. The MNIST dataset, which consists of 28×28 grayscale handwritten digits and corresponding tags, is the most widely used dataset with 60,000 training sets and 10,000 testing sets. The fashion-MNIST is another dataset that has the same image size, data format, and number of training and testing sets as MNIST. It consists of 70,000 fashion products, and is divided into 10 categories, each of which contains 7,000 grayscale images. And the CelebA is a large face attributes dataset with more than 200K celebrity images.

We use WGAN [13] as the basic architecture of our network model, using a generator of DCGAN [71], and only change its loss functions, training 25 epochs on each dataset. It should be noted that since the MMD-based GANs [18][19] and energy-based GANs [20][21] use different generation and discrimination mechanisms, we use their own solutions in our experiments.

To evaluate the performance of various GANs mentioned above, two commonly used quantitative evaluation metrics (Inception Scores (IS) [72] and Fr chet Inception Distance (FID) [73]) are used in our experiments.

Inception Scores (IS). As the most widely used evaluation metric, the inception score was proposed by Salimans *et al.* [72], which is expressed as,

$$IS(P_g) = exp\Big(\mathbb{E}_{x\sim p_g}\left[KL\left(p(y|x)\|p(y)\right)\right]\Big), \tag{40}$$

where $p(y|x)$ represents the conditional label distribution of images containing meaningful objects using a Inception model pretrained on the ImageNet [74], and $p(y)$ represents the marginal distribution of the overall generated samples. This score computes the average Kullback¨CLeibler divergence between $p(y|x)$ and $p(y)$. A higher IS means the diversity and high quality of the generated samples.

Fréchet Inception Distance (FID). Heusel *et al.* [73] uses suitable feature functions to make the generated images follow a multi-dimensional Gaussian distribution, and estimates them using mean and covariance operations. Then, the FID measures the distance between two Gaussians using the Fr chet distance [75], which is also known as Wasserstein-2 distance [76], to evaluate the quality of the generated samples. The FID is defined as,

$$\begin{aligned}\mathrm{FID}(P_r, P_g) = &\|\mu_x - \mu_y\|_2^2 \\ &+ \mathrm{Tr}\left(C_x + C_y - 2\left(C_x C_y\right)^{\frac{1}{2}}\right),\end{aligned} \tag{41}$$

where (μ_x, C_x) and (μ_y, C_y) represent the mean and covariance of the real data samples and the generated samples, respectively. For the distance between generated and real samples, we want the value of FID to be lower.

2) Application-oriented GANs' loss functions: For the application-oriented GANs' loss functions, we select several of them, and implement them into $4\times$ super-resolution [77], image denoising [65] and image inpainting [67] tasks to test their performance. Considering the computational complexity of these task, the most widely used SRGAN [26] is adopted as the basic model in the $4\times$ super-resolution task, and for denoising and inpainting tasks, the upsampling layers of SRGAN are removed. All experiments only changed their loss functions in the network. Specifically, in addition to the adversarial loss, we add other loss functions mentioned above in the experiments, which are divided into the following combinations: $l_{adv} + l_1; l_{adv} + l_2; l_{adv} + l_{color}; l_{adv} + l_{content}; l_{adv} + l_{texture}; l_{adv} + l_{cyc}; l_{adv} + l_{contextual}$. In addition, as a benchmark, we also test the results generated only from the adversarial loss function. It's worth noting that since total variation loss is a regularization term, it is always added to other loss functions. To test its performance, we add the total variation loss to the color loss in experiment: $l_{adv} + l_{color} + l_{TV}$. For the symmetry loss and identity preserving loss, since they are not suitable for these three tasks, their performance is not tested here. In addition, it should be noted that the cycle consistency loss is an unsupervised learning method. The use of unsupervised training methods in image inpainting and image denoising tasks are research focuses.

At present, the cycle consistency loss has been used in the audio and image denoising tasks [78], [79], but the denoising and inpainting tasks of nature scene datasets need to research, and we will continue to pay attention on this area. So, the performances of the cycle consistency loss in the denoising and inpainting tasks are not tested here.

For the super-resolution task, the DIV2K [80] is adopted as our training dataset, which includes 1,000 images of multiple categories. We train it on 800 images for 300 epochs and use the remaining 200 images to test. In addition, to further test the performance of each loss, the Set5 [81], Set14 [82] and B100 [83] are also adopted as our test datasets. For the image denoising task, the Smartphone Image Denoising Dataset (SIDD)[84] medium dataset is used, which contains 320 image pairs collected from 160 scene instances. Since each image is 5380×3000, 5328×3000 or 4032×3024 pixels, to augment data, we use 512×512 pixels sub-image cropped from the upper left, upper right, lower left, lower right and center of each image as the training dataset. 16 image pairs in sRGB space are used for testing, and the number of training epochs is 200. For the image inpainting task, the Places2 [85] dataset is used, which consists of 10 million images collected from 365 scene categories. We randomly select 36,000 images for training and 100 images for testing, training 200 epochs on each loss. Moreover, two most commonly used evaluation metrics, Peak Single-to-Noise Ratio (PSNR) and Structural Similarity (SSIM) [47], are adopted to evaluate the objective performance of these loss functions.

Peak Signal-to-Noise Ratio (PSNR). As one of the most widely used image quality assessment (IQA) methods, the PSNR measures the error between corresponding pixels of the image. A higher PSNR (in dB) indicates that the distortion of the image is smaller. The PSNR is defined as follows,

$$PSNR = 10 \times log_{10}\left(\frac{(2^n-1)^2}{MSE}\right), \tag{42}$$

where $MSE = \frac{1}{N}\sum_{i=1}^{N}(I_i - I_i')^2$, I and I' denote the reconstructed image and target image, respectively. n represents the number of bits of the pixel in each image, usually with 8 bits. However, since the Human Visual System (HVS) [47] is not considered, existing works have found that a higher PSNR does not always have a high subjective perception of human beings [47].

Structural Similarity (SSIM). Considering the shortcomings of PSNR, the SSIM is proposed to measure the structural similarity between two images from three aspects: luminance (L), contrast (C) and structure (S), which are defined as,

$$L = \frac{2\mu_I\mu_{I'} + C_1}{\mu_I^2 + \mu_{I'}^2 + C_1}, \tag{43}$$

$$C = \frac{2\sigma_I\sigma_{I'} + C_2}{\sigma_I^2 + \sigma_{I'}^2 + C_2}, \tag{44}$$

$$S = \frac{\sigma_{II'}, +C_3}{\sigma_I\sigma_{I'} + C_3}, \tag{45}$$

where μ_I and $\mu_{I'}$ denote the mean value of image I and I'; σ_I and $\sigma_{I'}$ represent the standard deviation of image I and I'; $\sigma_{II'}$ represents the covariance of I and I'; σ_I^2 and $\sigma_{I'}^2$ represent variance of two images. $C_1 = (k_1L)^2, C_2 = (k_2L)^2$ and $C_3 = C_2/2$ are constants for avoiding instability. The SSIM is formulated in Eq. (46),

$$SSIM = L(I, I')^\alpha C(I, I')^\beta S(I, I')^\gamma, \tag{46}$$

where α, β, γ are relative coefficients. In our experiments, we set $k_1 = 0.01, k_2 = 0.03, L = 255, \alpha = \beta = \gamma = 1$, and the SSIM is defined as,

$$SSIM = \frac{(2\mu_I\mu_{I\prime} + C_1)(2\sigma_I\sigma_{I\prime} + C_2)}{(\mu_I^2 + \mu_{I\prime}^2 + C_1)(\sigma_I^2 + \sigma_{I\prime}^2 + C_2)}. \tag{47}$$

The value of SSIM is between [0,1], and a higher SSIM indicates a high similarity between images, so the distortion of images is small.

TABLE IV
RESULTS OF DIFFERENT ADVERSARIAL LOSS FUNCTIONS GENERATED ON THE MNIST DATASET. (HIGHER VALUES ARE BETTER FOR IS AND LOWER VALUES ARE BETTER FOR FID)

Loss function	Inception Score (IS)	Fréchet Inception Distance (FID)
f-GAN_KL	5.16	3.11
f-GAN_JS	2.76	9.76
f-GAN_Person χ^2	3.54	7.9
f-GAN_reverse_KL	4.06	5.08
f-GAN_total_variation	4.72	5.17
f-GAN_ hellinger	4.73	6.13
WGAN	3.79	6.32
WGAN-GP	3.16	10.67
Least-square GAN	**5.58**	**2.2**
Loss-sensitive GAN	**6**	**2.75**
GMMN	3.43	12.44
MMDGAN	5.35	3.51
EBGAN	4.74	5.36
BEGAN	3.89	15.93

TABLE V
RESULTS OF DIFFERENT ADVERSARIAL LOSS FUNCTIONS GENERATED ON THE FASHION-MNIST DATASET. (HIGHER VALUES ARE BETTER FOR IS AND LOWER VALUES ARE BETTER FOR FID)

Loss function	Inception Score (IS)	Fréchet Inception Distance (FID)
f-GAN_KL	5.51	5.42
f-GAN_JS	3.16	8.61
f-GAN_Person χ^2	3.12	9.87
f-GAN_reverse_KL	5.23	4.34
f-GAN_total_variation	5.12	4.69
f-GAN_ hellinger	**6.57**	3.68
WGAN	4.41	5.80
WGAN-GP	5.21	**2.08**
Least-square GAN	**6.36**	3.44
Loss-sensitive GAN	5.45	4.79
GMMN	4.63	4.84
MMDGAN	6.24	**2.29**
EBGAN	3.34	10.28
BEGAN	5.22	3.35

B. Comparisons and Analyses

1) Adversarial Loss Functions Used in GANs: Table IV, Table V and Table VI are test scores on three datasets: MNIST, fashion-MNIST, and CelebA, respectively. From Table IV, the performance of least-square GAN and loss-sensitive GAN on IS and FID is the best among these compared loss functions. The loss-sensitive GAN has the highest IS with a value of 6, and the least-square GAN has the lowest FID with a value of

TABLE VI
RESULTS OF DIFFERENT ADVERSARIAL LOSS FUNCTIONS GENERATED ON THE CELEBA DATASET. (HIGHER VALUES ARE BETTER FOR IS AND LOWER VALUES ARE BETTER FOR FID)

Loss function	Inception Score (IS)	Fréchet Inception Distance (FID)
f-GAN_KL	7.12	12.23
f-GAN_JS	4.29	16.64
f-GAN_Person χ^2	5.74	29.33
f-GAN_reverse_KL	7.23	14.18
f-GAN_total_variation	7.82	15.88
f-GAN_ hellinger	5.51	12.39
WGAN	6.67	26.46
WGAN-GP	**8.54**	**9.91**
Least-square GAN	**9.01**	**9.62**
Loss-sensitive GAN	7.44	13.75
GMMN	5.93	14.11
MMDGAN	7.12	12.54
EBGAN	3.85	23.22
BEGAN	5.28	12.1

2.2. In addition, the MMDGAN and f-GAN which use the Kullback-Leibler divergence, also have a good performance. Due to the limited variety of images in the MNIST dataset, the gap between various loss functions is not particularly noticeable. Hence, in Fig. 4 and Fig. 5, we use the fashion-MNIST and CelebA dataset to further test the generation ability of each loss. In the fashion-MNIST, the f-GAN using squared Hellinger divergence achieves a better performance on IS, while the WGAN-GP performs better on FID. In the results of CelebA, the least-square GAN achieves the best performance on IS and FID, which are 9.01 and 9.62, respectively, and the results of WGAN-GP is worthy of attention. From these results, we can find that the least-square GAN, loss-sensitive GAN and MMDGAN are relatively stable on different datasets.

In addition, in Figs. 3, 4 and 5, the samples generated by different loss functions are chosen for visual comparison. As shown in Fig. 3, for the MNIST dataset, the samples generated by each loss have great performance, but the samples generated by the GMMN are less realistic in subjective perception. At the same time, in Fig. 5, the performance of this loss function on the CelebA dataset is also unsatisfactory. For the WGAN, the results generated by the WGAN on these three datasets are relatively realistic. The EBGAN does not perform satisfactorily results on fashion-MNIST dataset from quantitative assessment and subjective evaluation. By contrast, the BEGAN scores are better on the fashion-MNIST, but it performs poorly on the MNIST and CelebA datasets. This means that the training process of these two loss functions is not stable. Moreover, we can see that the samples generated by the least-square GAN have a realistic performance on these three datasets in terms of quantitative evaluation and subjective evaluation.

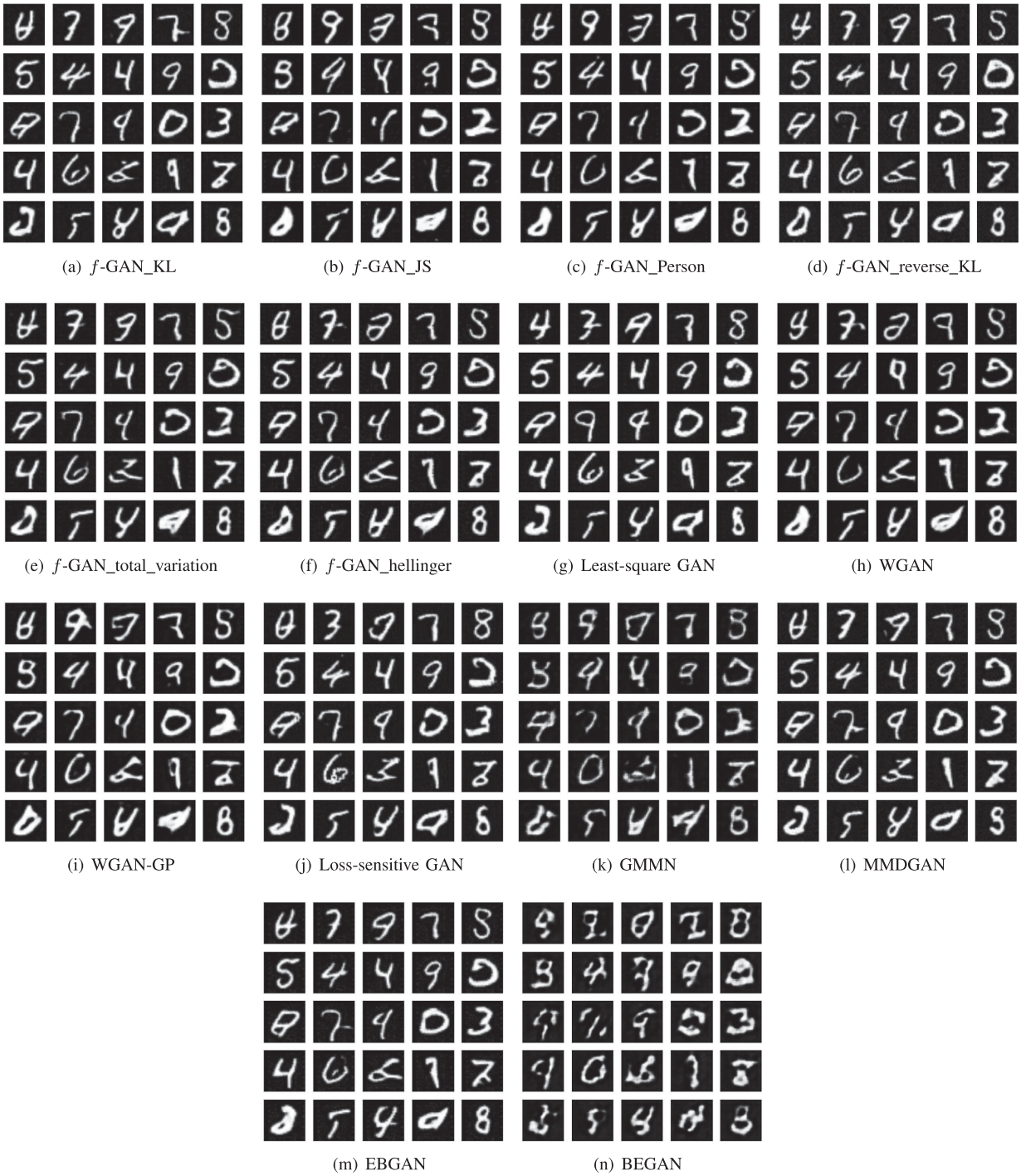
2) Application-Oriented GANs' Loss Functions: From the results in Tables VII, we can find that in the super-resolution task, the adversarial loss has the smallest average PSNR and SSIM, while when the adversarial loss is jointly used with the color loss, the average PSNR and average SSIM can be increased. When adding the total variation loss into the adversarial loss and color loss, the performance can be further improved. Comparing $l_{adv} + l_1$ and $l_{adv} + l_2$, they perform better than others in these four test datasets. Moreover, jointly using the adversarial loss and the l_1 loss achieves the best performance in these loss functions. The average PSNR and SSIM both have the highest values. The performance of jointly using the adversarial loss and the content loss is close to $l_{adv} + l_2$. In addition, the average PSNR of the contextual loss is higher than the cycle consistency loss, but the average SSIM is a little lower than it.

For the image inpainting task, in Table VIII, we can find that only using color loss or texture loss is not suitable for this task. The performance of jointly using the adversarial loss and the l_1 loss is better than that using $l_{adv} + l_2$. The average PSNR and SSIM are 21.862 dB and 0.867, respectively. The content loss also performs well in this task, the average PSNR and SSIM of which are 20.176 dB and 0.811, respectively.

The results of the image denoising task are shown in Table IX. We can see that the average PSNR and SSIM of noisy image in sRGB space are 20.522 dB and 0.940, respectively. The l_1, l_2, content, and contextual losses are suitable for this task, and the content loss achieves the best performance. The average PSNR and SSIM of $l_{adv} + l_{content}$ are 20.829 dB and 0.851, respectively.

Furthermore, in order to intuitively show the performance of each loss function, we choose one of these images, and discuss the effects of these loss functions in different tasks. The subjective quality comparisons for different loss functions in these three tasks are given in Figs. 6–11. In all tasks, when we only use adversarial loss, the quality of generated image is far from the ground truth, the artifacts are very serious, and the color excursion has been appeared, so its PSNR and SSIM are the lowest. Comparing between the l_1 loss and l_2 loss, the image using l_1 loss has a higher PSNR/SSIM than the image using l_2 loss. Although the PSNR and SSIM of these two loss functions are higher than other comparisons, the lack of high frequency information makes the generated images too smooth, and unsatisfactory in visual perception, especially in the super-resolution task. For the content loss, although it does not have the same PSNR/SSIM performance as pixel-wise loss, it provides a better perceptual quality due to the consideration of the HVS. However, at the same time, we can also find that there still exist artifacts because it does not consider the matching of local texture information. For the cycle consistency loss and contextual loss in the super-resolution task, the texture information of the contextual loss is richer, and the image looks more natural. Because the contextual loss takes into account the semantic information and the context relationships of the entire image.

In conclusion, from the evaluation results of different tasks, we can find that only using adversarial loss cannot achieve satisfactory results. The pixel-wise loss, as the most commonly used loss, performs well in each task, and the results of l_1 loss are better than l_2 loss. This is because that the l_1 does not overly penalize large errors, but l_2 is more sensitive to them due to the use of square item. When only use the texture loss and the adversarial loss, the image will have more texture features than the other loss functions. In [50], the texture loss and content loss are jointly used to improve both texture and perceptual quality. When the generated image needs to constrain noise,



(a) f-GAN_KL (b) f-GAN_JS (c) f-GAN_Person (d) f-GAN_reverse_KL

(e) f-GAN_total_variation (f) f-GAN_hellinger (g) Least-square GAN (h) WGAN

(i) WGAN-GP (j) Loss-sensitive GAN (k) GMMN (l) MMDGAN

(m) EBGAN (n) BEGAN

Fig. 3. MNIST results generated from different objective functions.

the total variation loss can be added. Furthermore, to improve the subjective quality, the contextual loss is also a good choice.

V. Discussion and Future of GANs

At present, the main problem of GANs need to face is that it always gets into mode collapse. Optimizing the objective function is one of the effective pathes to address this problem. The main optimization approaches are concentrated on two branches: f-divergence and integral probability metrics. By experiments, we found that the least-square GAN [15], which is the representative objective function of f-divergence, has satisfactory effects on different datasets in terms of both evaluation metrics and subjective analyzes. In addition, in the IPMs-based approaches, the methods of using Wasserstein distance [13], [16], [17] have also achieved realistic effect. It should be noted that although the least-square GAN works better in our experiments, it has been proved in [13] that when learning the distribution supported

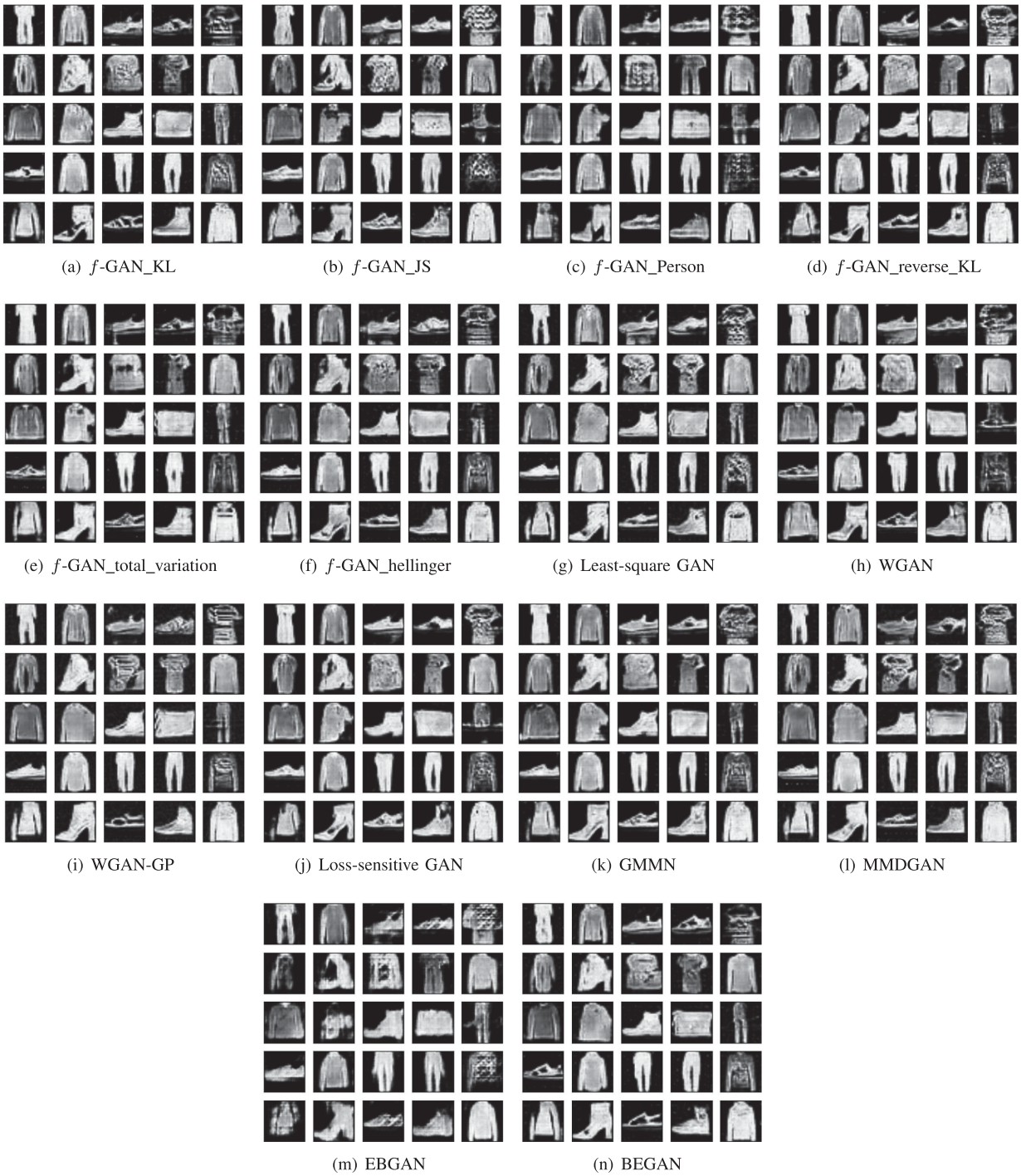


Fig. 4. Fashion-MNIST results generated from different objective functions.

by low dimensional manifolds, the distribution of real data and generated data can hardly overlap, so the gradient of the generator is easily to disappear, which makes the model difficult to converge and the training process will become harder than before.

The other loss functions added in different specific tasks, according to the results of our experiments, we can see that when we added the pixel-wise loss (MAE and MSE) to the adversarial loss, the quality of generated samples can be greatly improved. Since the PSNR is used as the evaluation metric in image quality assessment, the pixel-by-pixel loss function is very helpful for increasing the value of samples. In addition, the content loss [26] is another significant loss function which can promote the perceptual quality of samples. But we must point out that because of more attention on the HVS [47], it cannot achieve a better PSNR/SSIM than the pixel-wise loss. For style transfer and texture synthesis tasks, we argue that the texture loss [27] is much effective because it can enrich the texture information

Fig. 5. CelebA results generated from different objective functions.

of samples. In our experiments, we also found that when the texture loss is only added to the adversarial loss, the generated sample looks more abstract, so it needs to be combined with other loss functions. When noise exists in the generated samples, the total variation loss [30] can be added. It is also found from the experiments that the value of TV loss during the training process is small, so an appropriate coefficient should be chosen on a case by case basis. The emergence of the cycle consistency loss [29] is also very exciting. Since unpaired data is required as input, as an unsupervised learning method, the model can also be trained when a training dataset is insufficient. It has broad prospects not only in image translation but also in super resolution tasks and other tasks. When there are brightness and chrominance errors between the generated images and the target images, we recommend that the color loss [31] be used for compensating the difference between them to ensure consistency. When the task needs to measure the feature similarity between the generated images and the target images, we need to use the contextual loss

TABLE VII
AVERAGE PSNR/SSIM RESULTS ON DIFFERENT LOSS FUNCTIONS IN THE SUPER-RESOLUTION TASK. FOR BOTH PSNR AND SSIM, A HIGHER VALUES IS BETTER

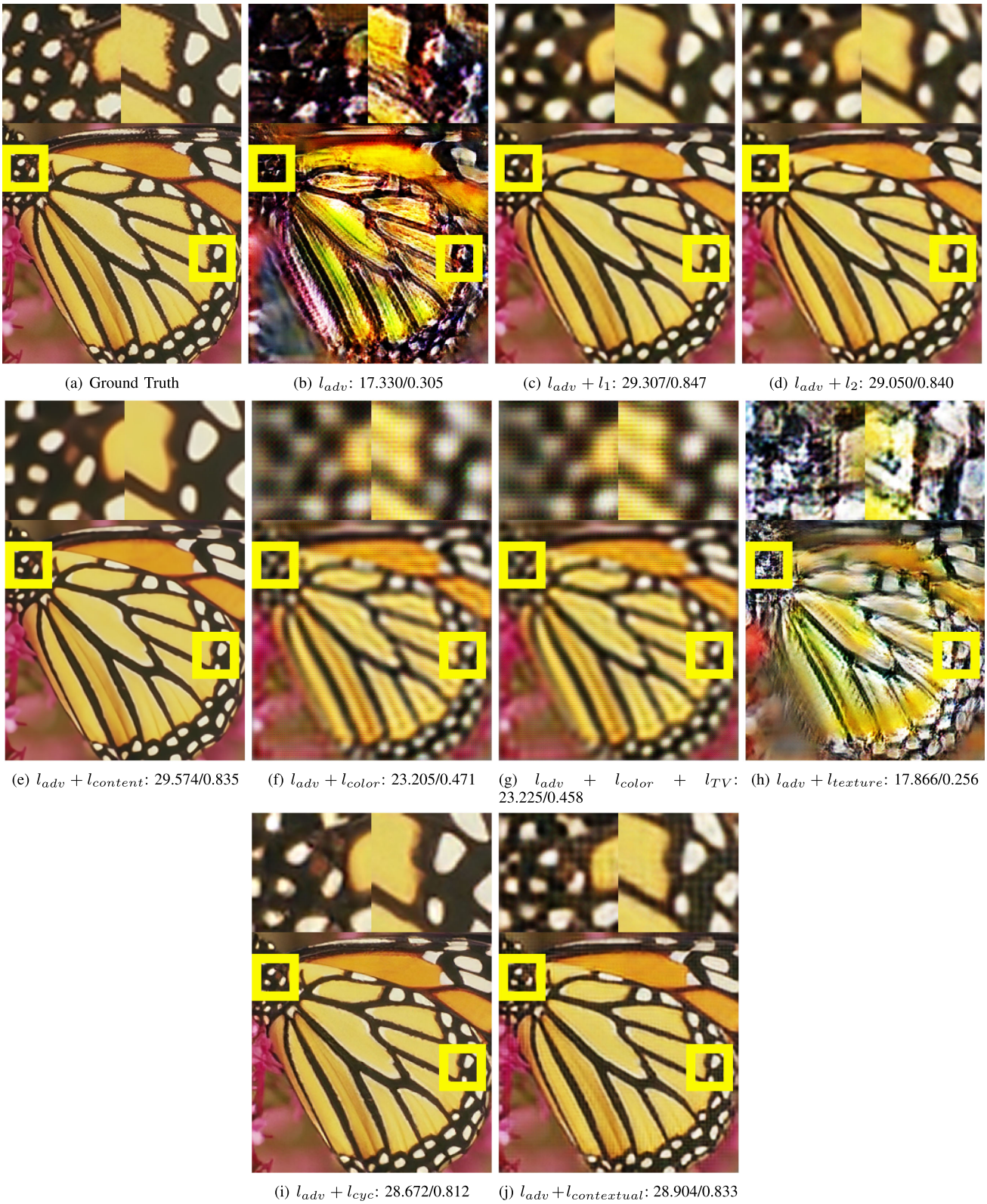
Loss function	DIV2K		Set5		Set14		B100	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
l_{adv}	21.855	0.439	15.252	0.465	14.989	0.446	14.249	0.338
$l_{adv} + l_{color}$	28.071	0.532	21.476	0.650	20.921	0.674	20.956	0.616
$l_{adv} + l_{color} + l_{TV}$	28.356	0.544	22.042	0.652	21.171	0.673	20.978	0.614
$l_{adv} + l_1$	**32.739**	**0.796**	**28.068**	**0.897**	23.979	**0.856**	**22.777**	**0.807**
$l_{adv} + l_2$	32.267	**0.792**	27.812	**0.894**	**24.029**	**0.855**	22.670	**0.806**
$l_{adv} + l_{content}$	**32.271**	0.785	**27.892**	0.891	**24.141**	0.853	**22.758**	0.801
$l_{adv} + l_{texture}$	23.606	0.473	16.805	0.473	16.580	0.488	16.761	0.405
$l_{adv} + l_{cyc}$	31.185	0.788	27.633	0.895	23.679	0.823	21.975	0.802
$l_{adv} + l_{contextual}$	31.735	0.745	27.821	0.835	23.879	0.857	22.020	0.768

(a) Ground Truth (b) l_{adv}: 29.605/0.284 (c) $l_{adv} + l_1$: 34.229/0.841 (d) $l_{adv} + l_2$: 34.123/0.840

(e) $l_{adv} + l_{content}$: 33.629/0.820 (f) $l_{adv} + l_{color}$: 29.605/0.603 (g) $l_{adv} + l_{color} + l_{TV}$: 30.027/0.618 (h) $l_{adv} + l_{texture}$: 23.309/0.349

(i) $l_{adv} + l_{cyc}$: 33.424/0.833 (j) $l_{adv} + l_{contextual}$: 31.652/0.763

Fig. 6. Images generated on different loss functions in 4× super-resolution tasks (PSNR/SSIM). [From DIV2K].

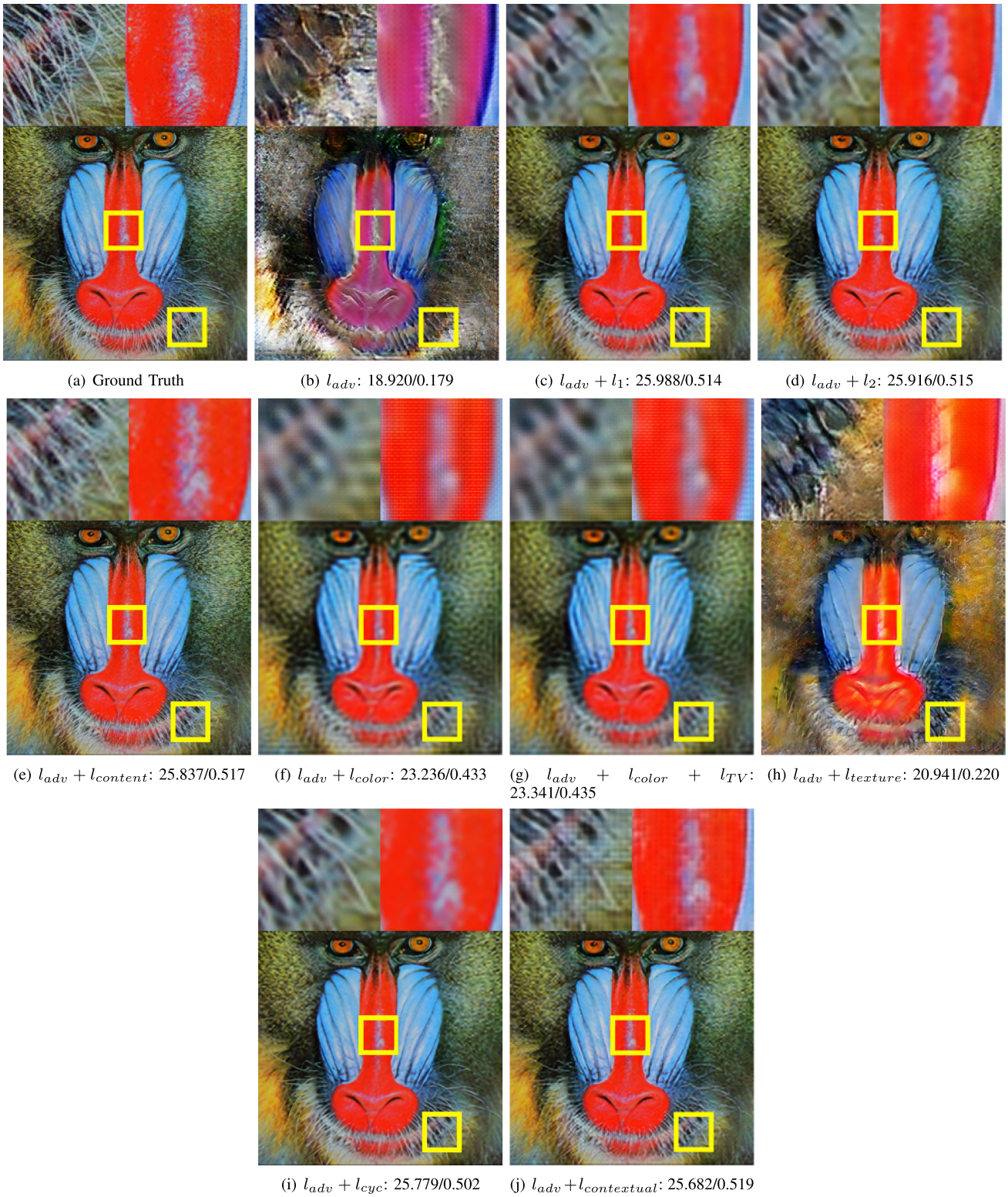


(a) Ground Truth (b) l_{adv}: 17.330/0.305 (c) $l_{adv} + l_1$: 29.307/0.847 (d) $l_{adv} + l_2$: 29.050/0.840

(e) $l_{adv} + l_{content}$: 29.574/0.835 (f) $l_{adv} + l_{color}$: 23.205/0.471 (g) $l_{adv} + l_{color} + l_{TV}$: 23.225/0.458 (h) $l_{adv} + l_{texture}$: 17.866/0.256

(i) $l_{adv} + l_{cyc}$: 28.672/0.812 (j) $l_{adv} + l_{contextual}$: 28.904/0.833

Fig. 7. Images generated on different loss functions in 4× super-resolution tasks (PSNR/SSIM). [From Set5].



(a) Ground Truth (b) l_{adv}: 18.920/0.179 (c) $l_{adv} + l_1$: 25.988/0.514 (d) $l_{adv} + l_2$: 25.916/0.515

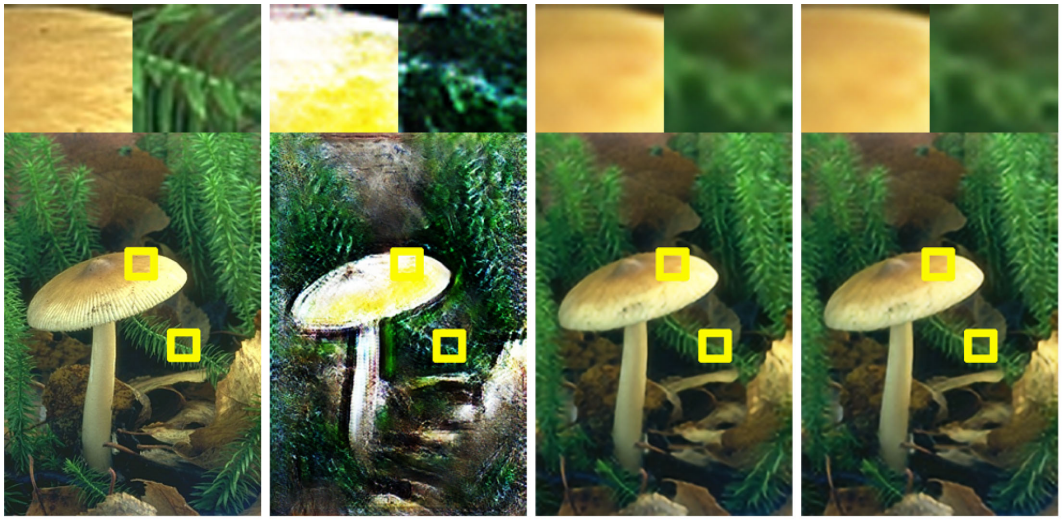
(e) $l_{adv} + l_{content}$: 25.837/0.517 (f) $l_{adv} + l_{color}$: 23.236/0.433 (g) $l_{adv} + l_{color} + l_{TV}$: 23.341/0.435 (h) $l_{adv} + l_{texture}$: 20.941/0.220

(i) $l_{adv} + l_{cyc}$: 25.779/0.502 (j) $l_{adv} + l_{contextual}$: 25.682/0.519

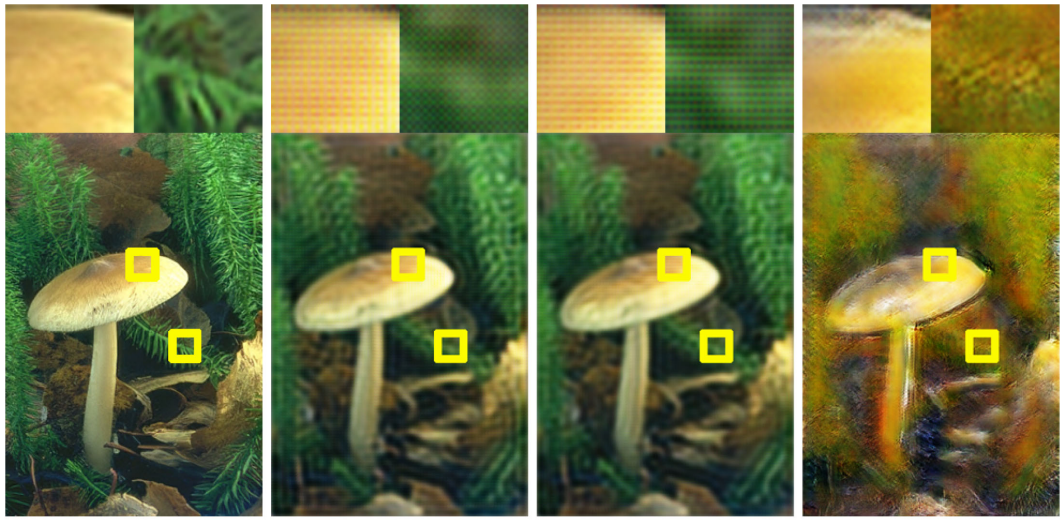
Fig. 8. Images generated on different loss functions in 4× super-resolution tasks (PSNR/SSIM). [From Set14].

[28]. In the face synthesis task, in order to obtain a symmetrical face and more consistent information, both symmetry loss [32] and identity preserving loss [32] can be used.

Currently, GANs has been widely used in various applications, not only in the computer vision [86]–[89] and nature language processing [90]–[92] tasks, but also in many other domains. In the information security domain, Shin *et al.* [93] proposed a GANs to prevent the Android mode locking system from being attacked. Zheng *et al.* [94] proposed a key secret-sharing scheme based on GANs in blockchain. In medicine,



(a) Ground Truth (b) l_{adv}: 20.058/0.186 (c) $l_{adv} + l_1$: 29.841/0.709 (d) $l_{adv} + l_2$: 30.026/0.712



(e) $l_{adv} + l_{content}$: 30.083/0.698 (f) $l_{adv} + l_{color}$: 27.787/0.413 (g) $l_{adv} + l_{color} + l_{TV}$: 27.487/0.412 (h) $l_{adv} + l_{texture}$: 20.938/0.295

(i) $l_{adv} + l_{cyc}$: 29.846/0.627 (j) $l_{adv} + l_{contextual}$: 29.871/0.635

Fig. 9. Images generated on different loss functions in 4× super-resolution tasks (PSNR/SSIM). [From B100].

(a) Ground Truth (b) Noisy: 27.554/0.952 (c) l_{adv}: 7.685/0.419 (d) $l_{adv} + l_1$: 27.595/0.957

(e) $l_{adv} + l_2$: 27.520/0.942 (f) $l_{adv} + l_{content}$: 27.966/0.958 (g) $l_{adv} + l_{color}$: 13.130/0.325 (h) $l_{adv}+l_{color}+l_{TV}$:15.558/0.335

(i) $l_{adv} + l_{texture}$: 9.059 /0.401 (j) $l_{adv}+l_{contextual}$: 25.125/0.913

Fig. 10. Images generated on different loss functions in the denoising tasks (PSNR/SSIM). [From SIDD].

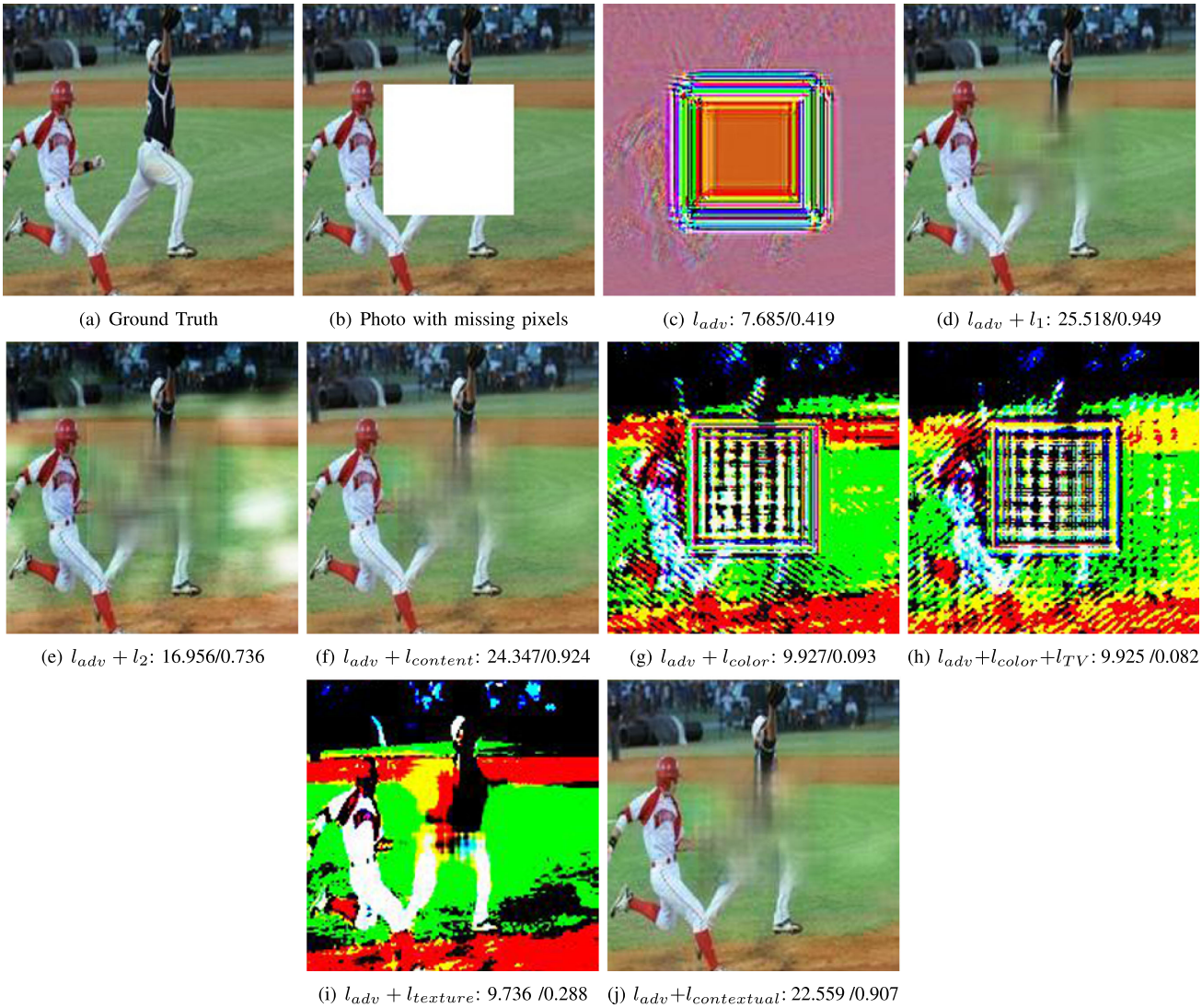
TABLE VIII
AVERAGE PSNR/SSIM RESULTS ON DIFFERENT LOSS FUNCTIONS IN THE IMAGE INPAINTING TASK. FOR BOTH PSNR AND SSIM, A HIGHER VALUES IS BETTER

Loss function	PSNR	SSIM
l_{adv}	9.351	0.160
$l_{adv} + l_{color}$	10.441	0.131
$l_{adv} + l_{color} + l_{TV}$	10.704	0.153
$l_{adv} + l_1$	**21.862**	**0.867**
$l_{adv} + l_2$	14.817	0.706
$l_{adv} + l_{content}(l_{perceptual})$	**20.176**	**0.811**
$l_{adv} + l_{texture}$	11.354	0.341
$l_{adv} + l_{contextual}$	20.021	0.799

TABLE IX
AVERAGE PSNR/SSIM RESULTS ON DIFFERENT LOSS FUNCTIONS IN THE IMAGE DENOISING TASK (SRGB SPACE). FOR BOTH PSNR AND SSIM, A HIGHER VALUES IS BETTER

Loss function	PSNR	SSIM
Noisy(benchmark)	20.522	0.940
l_{adv}	9.058	0.433
$l_{adv} + l_{color}$	14.954	0.355
$l_{adv} + l_{color} + l_{TV}$	15.600	0.331
$l_{adv} + l_1$	**20.739**	0.828
$l_{adv} + l_2$	20.590	**0.829**
$l_{adv} + l_{content}(l_{perceptual})$	**20.829**	**0.851**
$l_{adv} + l_{texture}$	9.393	0.174
$l_{adv} + l_{contextual}$	19.032	0.801

Bhattacharya *et al.* [95] utilized GANs to augment medical image datasets, thus improving the classification accuracy of diseases. Zhang *et al.* [96] proposed a GANs for X-ray image segmentation. Gupta *et al.* [97] proposed an FBGAN to generate synthetic DNA sequences encoding. All of the optimizations can be roughly classified into two groups, one is to optimize the architecture of GANs, and the other is to optimize the loss function. For the loss functions, the existing loss has partly addressed the problem of mode collapse, but this is still a direction that needs people to research. In addition, training



(a) Ground Truth (b) Photo with missing pixels (c) l_{adv}: 7.685/0.419 (d) $l_{adv} + l_1$: 25.518/0.949

(e) $l_{adv} + l_2$: 16.956/0.736 (f) $l_{adv} + l_{content}$: 24.347/0.924 (g) $l_{adv} + l_{color}$: 9.927/0.093 (h) $l_{adv}+l_{color}+l_{TV}$: 9.925 /0.082

(i) $l_{adv} + l_{texture}$: 9.736 /0.288 (j) $l_{adv}+l_{contextual}$: 22.559 /0.907

Fig. 11. Images generated on different loss functions in the inpainting tasks (PSNR/SSIM). [From Place2].

GANs is usually unstable, and not easily to converge. At present, many approaches to optimize the GANs performance by adding multiple losses, and how to choose the weight for each loss depends on researchers' experiences. Hence, it is essential to develop theories for weight determination.

VI. Conclusion

In this paper, we performed a survey for the loss functions of GANs, including objective functions which can increase the generation ability, and many other loss functions which can be added to GANs for different real-world applications. The definition of these loss functions, and the concrete usage of these loss functions are explained in details. In addition, the performance of these loss functions is quantitatively evaluated, and the pros and cons of these loss functions are further analyzed. Finally, we provide suggestions on how to choose proper loss functions for specific tasks.

References

[1] Q. Li *et al.*, "AF-DCGAN: Amplitude feature deep convolutional gan for fingerprint construction in indoor localization systems," *IEEE Trans. Emerg. Topics Comput. Intell.*, pp. 1–13, 2019.

[2] K. Zheng, W. Q. Yan, and P. Nand, "Video dynamics detection using deep neural networks," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 3, pp. 224–234, Jun. 2018.

[3] V. Kuppili, M. Biswas, D. R. Edla, K. J. R. Prasad, and J. S. Suri, "A mechanics-based similarity measure for text classification in machine learning paradigm," *IEEE Trans. Emerg. Topics Comput. Intell.*, pp. 1–21, 2018.

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[5] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," Colorado Univ Boulder Dept Comput. Sci., Tech. Rep., 1986.

[6] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[7] R. Salakhutdinov and G. Hinton, "Deep boltzmann machines," in *Artif. Intell. and Stat.*, 2009, pp. 448–455.

[8] I. Goodfellow *et al.*, "Generative adversarial nets," in *Adv. Neural. Inf. Process. Syst.*, 2014, pp. 2672–2680.

[9] Y. Hong, U. Hwang, J. Yoo, and S. Yoon, "How generative adversarial networks and their variants work: An overview," *ACM Comput. Surveys*, vol. 52, no. 1, pp. 1–43, 2019.

[10] 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. 36 322–36 333, Mar. 2019.

[11] Z. Wang, Q. She, and T. E. Ward, "Generative adversarial networks: A survey and taxonomy," 2019, *arXiv:1906.01529*.

[12] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," 2020, *arXiv:2001.06937*.

[13] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 214–223.

[14] S. Nowozin, B. Cseke, and R. Tomioka, "F–GAN: Training generative neural samplers using variational divergence minimization," in *Adv. Neural. Inf. Process. Syst.*, 2016, pp. 271–279.

[15] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "On the effectiveness of least squares generative adversarial networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 12, pp. 2947–2960, Dec. 2019.

[16] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein gans," in *Adv. Neural. Inf. Process. Syst.*, 2017, pp. 5767–5777.

[17] G.-J. Qi, "Loss-sensitive generative adversarial networks on lipschitz densities," *Int. J. Comput. Vision.*, pp. 1–23, 2019.

[18] Y. Li, K. Swersky, and R. Zemel, "Generative moment matching networks," in *Int. Conf. Mach. Learn.*, 2015, pp. 1718–1727.

[19] C.-L. Li, W.-C. Chang, Y. Cheng, Y. Yang, and B. Póczos, "MMD GAN: Towards deeper understanding of moment matching network," in *Adv. Neural. Inf. Process. Syst.*, 2017, pp. 2203–2213.

[20] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," in *Int. Conf. Mach. Learn.*, 2016, pp. 1–17.

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

[22] X. Nguyen, M. J. Wainwright, and M. I. Jordan, "Estimating divergence functionals and the likelihood ratio by convex risk minimization," *IEEE Trans. Inf. Theory*, vol. 56, no. 11, pp. 5847–5861, Nov. 2010.

[23] J.-B. Hiriart-Urruty and C. Lemaréchal, *Fundamentals of convex analysis*, 2012.

[24] A. Müller, "Integral probability metrics and their generating classes of functions," *Adv. Appl. Probability*, vol. 29, no. 2, pp. 429–443, 1997.

[25] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, no. Mar, pp. 723–773, 2012.

[26] C. Ledig *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2017, pp. 4681–4690.

[27] M. S. Sajjadi, B. Scholkopf, and M. Hirsch, "Enhancenet: Single image super-resolution through automated texture synthesis," in *IEEE Int. Conf. Comput. Vision*, 2017, pp. 4491–4500.

[28] R. Mechrez, I. Talmi, F. Shama, and L. Zelnik-Manor, "Maintaining natural image statistics with the contextual loss," in *Asian Conf. Comput. Vision*, 2018, pp. 427–443.

[29] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE Int. Conf. Comput. Vision*, 2017, pp. 2223–2232.

[30] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.

[31] A. Ignatov, N. Kobyshev, R. Timofte, K. Vanhoey, and L. Van Gool, "Dslr-quality photos on mobile devices with deep convolutional networks," in *IEEE Int. Conf. Comput. Vision*, 2017, pp. 3277–3285.

[32] R. Huang, S. Zhang, T. Li, and R. He, "Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis," in *IEEE Int. Conf. Comput.*, 2017, pp. 2439–2448.

[33] Y. Yuan, S. Liu, J. Zhang, Y. Zhang, C. Dong, and L. Lin, "Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks," in *IEEE Conf. Comput. Vision. Pattern. Recognit. Workshops*, 2018, pp. 701–710.

[34] J. Deng, S. Cheng, N. Xue, Y. Zhou, and S. Zafeiriou, "UV-GAN: Adversarial facial UV map completion for pose-invariant face recognition," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2018, pp. 7093–7102.

[35] F. Zhan, H. Zhu, and S. Lu, "Spatial fusion GAN for image synthesis," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2019, pp. 3653–3662.

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

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

[38] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2017, pp. 1125–1134.

[39] L. Wang, Y.-S. Ho, K.-J. Yoon *et al.*, "Event-based high dynamic range image and very high frame rate video generation using conditional generative adversarial networks," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2019, pp. 10 081–10 090.

[40] H. Bin, C. Weihai, W. Xingming, and L. Chun-Liang, "High-quality face image sr using conditional generative adversarial networks," 2017, *arXiv:1707.00737*.

[41] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2016, pp. 2536–2544.

[42] 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 *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2017, pp. 5485–5493.

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

[44] Y. Liu, Z. Qin, T. Wan, and Z. Luo, "Auto-painter: Cartoon image generation from sketch by using conditional Wasserstein generative adversarial networks," *Neurocomputing*, vol. 311, pp. 78–87, 2018.

[45] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Trans. Comput. Imag.*, vol. 3, no. 1, pp. 47–57, Mar. 2017.

[46] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Eur. Conf. Comput.*, 2016, pp. 694–711.

[47] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli *et al.*, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, April 2004.

[48] A. Bulat and G. Tzimiropoulos, "Super-fan: Integrated facial landmark localization and super-resolution of real-world low resolution faces in arbitrary poses with gans," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2018, pp. 109–117.

[49] X. Wang, K. Yu, C. Dong, and C. Change Loy, "Recovering realistic texture in image super-resolution by deep spatial feature transform," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2018, pp. 606–615.

[50] X. Wang *et al.*, "ESRGAN: Enhanced super-resolution generative adversarial networks," in *Eur. Conf. Comput. Vision*, 2018, pp. 1–16.

[51] Q. Yang *et al.*, "Low-dose ct image denoising using a generative adversarial network with Wasserstein distance and perceptual loss," *IEEE Trans. Med. Imag.*, vol. 37, no. 6, pp. 1348–1357, Jun. 2018.

[52] X. Di, V. A. Sindagi, and V. M. Patel, "GP-GAN: Gender preserving gan for synthesizing faces from landmarks," in *The 24th Int. Conf. Pattern. Recognit.*, 2018, pp. 1079–1084.

[53] Z. Wang, M. Ye, F. Yang, X. Bai, and S. Satoh, "Cascaded sr-gan for scale-adaptive low resolution person re-identification," in *Int. Joint. Conf. Artif. Intell.*, 2018, pp. 3891–3897.

[54] D. Engin, A. Genç, and H. Kemal Ekenel, "Cycle-dehaze: Enhanced cyclegan for single image dehazing," in *Conf. Comput. Vision. Pattern. Recognit. Workshops*, 2018, pp. 825–833.

[55] L. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Adv. Neural. Inf. Process. Syst.*, 2015, pp. 262–270.

[56] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2016, pp. 2414–2423.

[57] R. Mechrez, I. Talmi, and L. Zelnik-Manor, "The contextual loss for image transformation with non-aligned data," in *Eur. Conf. Comput. Vision*, 2018, pp. 768–783.

[58] D. Lee, J. Kim, W.-J. Moon, and J. C. Ye, "Collagan: Collaborative GAN for missing image data imputation," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2019, pp. 2487–2496.

[59] J. M. Wolterink, A. M. Dinkla, M. H. Savenije, P. R. Seevinck, C. A. van den Berg, and I. Išgum, "Deep MR to CT synthesis using unpaired data," in *Int. Workshop. Simul. Synthesis. Med. Imag.*, 2017, pp. 14–23.

[60] Y. Hiasa *et al.*, "Cross-modality image synthesis from unpaired data using cyclegan," in *Int. Workshop. Simul. Synthesis. Med. Imag.*, 2018, pp. 31–41.

[61] Y. Lu, Y.-W. Tai, and C.-K. Tang, "Attribute-guided face generation using conditional cyclegan," in *Eur. Conf. Comput. Vision*, 2018, pp. 282–297.

[62] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker, "Towards large-pose face frontalization in the wild," in *IEEE Int. Conf. Comput. Vision*, 2017, pp. 3990–3999.

[63] X. Wu, R. He, Z. Sun, and T. Tan, "A light cnn for deep face representation with noisy labels," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 11, pp. 2884–2896, Nov. 2018.

[64] B. Gecer, S. Ploumpis, I. Kotsia, and S. Zafeiriou, "Ganfit: Generative adversarial network fitting for high fidelity 3d face reconstruction," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2019, pp. 1155–1164.

[65] S. Guo, Z. Yan, K. Zhang, W. Zuo, and L. Zhang, "Toward convolutional blind denoising of real photographs," in *2019 IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2019, pp. 1712–1722.

[66] A. Ignatov, N. Kobyshev, R. Timofte, and K. Vanhoey, "DSLR-quality photos on mobile devices with deep convolutional networks," in *2017 Int. Conf. Comput. Vision*, 2017, pp. 3297–3305.

[67] N. Wang, J. Li, L. Zhang, and B. Du, "Musical: multi-scale image contextual attention learning for inpainting," in *28th Int. Joint Conf. Artif. Intell. (IJCAI)*, 2019, pp. 3748–3754.

[68] Y. LeCun *et al.*, "Gradient-based learning applied to document recognition," *Proc. IEEE Proc. IRE* (through 1962)*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[69] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.

[70] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *IEEE Int. Conf. Comput. Vision*, 2015, pp. 3730–3738.

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

[72] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Adv. Neural Inform. Process. Syst. (NIPS)*, 2016, pp. 2234–2242.

[73] 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 *Adv. Neural Inform. Process. Syst. (NIPS)*, 2017, pp. 6626–6637.

[74] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2009, pp. 248–255.

[75] M. Fréchet, "Sur la distance de deux lois de probabilité," *Paris, Sci, Acad, R, C*, pp. 689–692, 1957.

[76] L. N. Vaserstein, "Markov processes over denumerable products of spaces, describing large systems of automata," *Problemy Peredachi Informatsii*, vol. 5, no. 3, pp. 64–72, 1969.

[77] Y. Zhang *et al.*, "Collaborative representation cascade for single-image super-resolution," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 5, pp. 845–860, May 2019.

[78] K. Tanaka, T. Kaneko, N. Hojo, and H. Kameoka, "Synthetic-to-natural speech waveform conversion using cycle-consistent adversarial networks," in *IEEE Spoken Lang. Technol. Workshop (SLT)*, 2018, pp. 632–639.

[79] J. Song, J.-H. Jeong, D.-S. Park, H.-H. Kim, D.-C. Seo, and J. C. Ye, "Unsupervised denoising for satellite imagery using wavelet subband cyclegan," 2020, *arXiv:2002.09847*.

[80] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *IEEE Conf. Comput. Vision and Pattern Recognit. Workshops (CVPRW)*, 2017, pp. 126–135.

[81] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," 2012.

[82] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *Int. conf. curves and Surfaces*, 2010, pp. 711–730.

[83] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *IEEE Int. Conf. Comput. Vision*, vol. 2, 2001, pp. 416–423.

[84] A. Abdelhamed, S. Lin, and M. S. Brown, "A high-quality denoising dataset for smartphone cameras," in *IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2018, pp. 1692–1700.

[85] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1452–1464, June 2018.

[86] G. Daras, A. Odena, H. Zhang, and A. G. Dimakis, "Your local GAN: Designing two dimensional local attention mechanisms for generative models," 2019, *arXiv:1911.12287*.

[87] A. Rahnama, A. T. Nguyen, and E. Raff, "Robust design of deep neural networks against adversarial attacks based on Lyapunov theory," 2019, *arXiv:1911.04636*.

[88] A. Karnewar and R. S. Iyengar, "Msg-gan: Multi-scale gradients gan for more stable and synchronized multi-scale image synthesis," 2019, *arXiv:1903.06048*.

[89] T. R. Shaham, T. Dekel, and T. Michaeli, "Singan: Learning a generative model from a single natural image," in *2019 IEEE/CVF Internat. Conf. Comput. Vision (ICCV)*, 2019, pp. 4569–4579.

[90] M. A. Haidar and M. Rezagholizadeh, "Textkd-gan: Text generation using knowledge distillation and generative adversarial networks," in *Adv. Artif. Intell.*, M.-J. Meurs and F. Rudzicz, Eds., Cham, 2019, pp. 107–118.

[91] K. Lin, D. Li, X. He, Z. Zhang, and M.-T. Sun, "Adversarial ranking for language generation," in *Adv. Neural Inform. Proc. Syst. (NIPS)*, 2017, pp. 3155–3165.

[92] L. Yu, W. Zhang, J. Wang, and Y. Yu, "Seqgan: Sequence generative adversarial nets with policy gradient," in *The Thirty-First AAAI Conf. Artif. Intell. (AAAI)*, pp. 2852–2858.

[93] S.-Y. Shin, Y.-W. Kang, and Y.-G. Kim, "Android-gan: Defending against android pattern attacks using multi-modal generative network as anomaly detector," *Expert Syst. Appl.*, vol. 141, p. 112964, 2020.

[94] W. Zheng, K. Wang, and F.-Y. Wang, "Gan-based key secret-sharing scheme in blockchain," *IEEE Trans. Cybern.*, 2020.

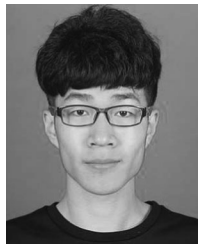
[95] D. Bhattacharya, S. Banerjee, S. Bhattacharya, B. U. Shankar, and S. Mitra, "GAN-based novel approach for data augmentation with improved disease classification," in *Adv. Mach. Intell. Interactive Medical Imag. Anal.*, 2020, pp. 229–239.

[96] Y. Zhang, S. Miao, T. Mansi, and R. Liao, "Unsupervised x-ray image segmentation with task driven generative adversarial networks," *Medical Imag. Anal.*, vol. 62, p. 101664, 2020.

[97] A. Gupta and J. Zou, "Feedback GAN for dna optimizes protein functions," *Nature Mach. Intell.*, vol. 1, no. 2, pp. 105–111, 2019.

Zhaoqing Pan (Senior Member, IEEE) received the Ph.D. degree in computer science from the City University of Hong Kong, Kowloon, Hong Kong, in 2014. In 2013, he was a Visiting Scholar with the Department of Electrical Engineering, University of Washington, Seattle, WA, USA, for six months. He is currently a Professor with the School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China. His research interests focus on video coding, image quality assessment, and machine learning.

Weijie Yu (Setdent Member, IEEE) received the B.S. degree in computer science and technology from the Nanjing University of Information Science and Technology, Nanjing, China, in 2018. He is currently working toward the master of science degree in engineering at the same university. His research interests focus on video coding and machine learning.



Bosi Wang (Setdent Member, IEEE) received the B.S. degree in computer science and technology from the Nanjing University of Information Science and Technology, Nanjing, China, in 2018. He is currently working toward the master of science in engineering degree at the same university. His research interests focus on video coding and machine learning.

Haoran Xie (Senior Member, IEEE) received the Ph.D. degree in Computer Science from the City University of Hong Kong, Hong Kong, China. He is currently an Associate Professor at the Department of Computing and Decision Sciences, Lingnan University, Hong Kong, China. His research interest includes artificial intelligence, big data, and educational technology. He has published 199 research publications, including 85 journal articles such as IEEE TPAMI, IEEE TKDE, IEEE TAFF, IEEE TCVST, and so on. He is the Academic Editor of PLOS One, Education Research International, and Editorial Member of the *Journal of Computers in Education*. He has been Guest Editors of 11 journals, and co-chairs/committee members of about 60 conferences.

Victor S. Sheng (Senior Member, IEEE) received the master¡¯s degree in computer science from the University of New Brunswick, Canada, in 2003, and the Ph.D. degree in computer science from Western University, London, ON, Canada, in 2007. He was an Associate Research Scientist and a NSERC Postdoctoral Fellow of information systems with the Stern Business School, New York University, after he received his Ph.D. He was an Associate Professor of computer science with the University of Central Arkansas, and the Founding Director of the Data Analytics Lab (DAL). Since 2018, he has been a Tenured Associate Professor with the Department of Computer Science, Texas Tech University, Lubbock, TX, USA. His research interests include data mining, machine learning, and related applications.

Jianjun Lei (Senior Member, IEEE) received the Ph.D. degree in signal and information processing from the Beijing University of Posts and Telecommunications, Beijing, China, in 2007. He was a Visiting Researcher with the Department of Electrical Engineering, University of Washington, Seattle, WA, USA, from August 2012 to August 2013. He is currently a Professor with Tianjin University, Tianjin, China. His research interests include 3D video processing, virtual reality, and artificial intelligence.

Sam Kwong (Fellow, IEEE) received the B.S. degree in electrical engineering from the State University of New York at Buffalo in 1983, the M.S. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 1985, and the Ph.D. degree from the University of Hagen, Germany, in 1996. From 1985 to 1987, he was a Diagnostic Engineer with Control Data Canada. He joined Bell Northern Research Canada as a member of Scientific Staff. In 1990, he became a Lecturer with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, where he is currently a Professor with the Department of Computer Science. His research interests include video, image coding, evolutionary algorithms, and machine learning.