Frechet Inception Distance (FID) for Evaluating GANs

Preprint · September 2021

3 authors, including:

Yu Yu
China University of Mining and Technology-Beijing
3 PUBLICATIONS **0** CITATIONS

SEE PROFILE

Frechet Inception Distance (FID) for Evaluating GANs

Yu Yu, Weibin Zhang, Yun Deng

I. Introduction

The Frechet Inception Distance score, or FID for short, is a metric that calculates the distance between feature vectors calculated for real and generated images.

The score summarizes how similar the two groups are in terms of statistics on computer vision features of the raw images calculated using the inception v3 model used for image classification. Lower scores indicate the two groups of images are more similar, or have more similar statistics, with a perfect score being 0.0 indicating that the two groups of images are identical. The FID score is used to evaluate the quality of images generated by generative adversarial networks, and lower scores have been shown to correlate well with higher quality images. In this tutorial [1], you will discover how to implement the Frechet Inception Distance for evaluating generated images.

The Frechet Inception Distance, or FID for short, is a metric for evaluating the quality of generated images and specifically developed to evaluate the performance of generative adversarial networks. The FID score was proposed and used by Martin Heusel, et al. in their 2017 paper titled GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. The score was proposed as an improvement over the existing Inception Score, or IS. The inception score estimates the quality of a collection of synthetic images based on how well the top-performing image classification model Inception v3 classifies them as one of 1,000 known objects. The scores combine both the confidence of the conditional class predictions for each synthetic image (quality) and the integral of the marginal probability of the predicted classes (diversity). The inception score does not capture how synthetic images compare to real images. The goal in developing the FID score was to evaluate synthetic images based on the statistics of a collection of synthetic images compared to the statistics of a collection of real images from the target domain. Like the inception score, the FID score uses the inception v3 model. Specifically, the coding layer of the model (the last pooling layer prior to the output classification of images) is used to capture computer-vision-specific features of an input image. These activations are calculated for a collection of real and generated images. The activations are summarized as a multivariate Gaussian by calculating the mean and covariance of the images. These statistics are then calculated for the activations across the collection of real and generated images. The distance between these two distributions is then calculated using the Frechet distance, also called the Wasserstein-2 distance. The use of activations from the Inception v3 model to summarize each image gives the score its name of "Frechet Inception Distance." A lower FID indicates better-quality images; conversely, a higher score indicates a lower-quality image and the relationship may be linear. The authors of the score show that lower FID scores correlate with better-quality images when systematic distortions were applied such as the addition of random noise and blur.

[1]https://machinelearningmastery.com/how-to-implement-the-frechet-inception-distance-fid-from-scratch/

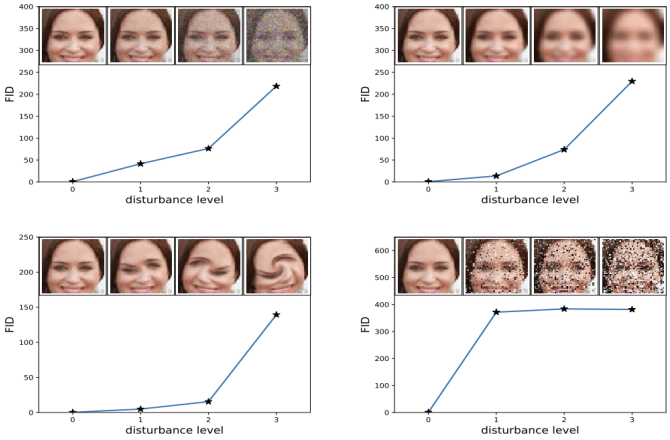

Fig. 1: Taken from: GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium.

II. Related Work

There are many papers related to GANs [1], [2], [3], [4], [5], [6] such as CSGAN [7] and LOGAN [8]. In this subsection, we will introduce GANs' representative variants.

1) InfoGAN: Rather than utilizing a single unstructured noise vector z, InfoGAN [9] proposes to decompose the input noise vector into two parts: z, which is seen as incompressible noise; c, which is called the latent code and will target the significant structured semantic features of the real data distribution. InfoGAN [9] aims to solve

$$\min_G \max_D V_I(D,G) = V(D,G) - \lambda I(c; G(z,c)), \qquad (1)$$

where $V(D,G)$ is the objective function of original GAN, $G(z,c)$ is the generated sample, I is the mutual information, and λ is the tunable regularization parameter. Maximizing $I(c; G(z,c))$ means maximizing the mutual information between c and $G(z,c)$ to make c contain as much important and meaningful features of the real samples as possible. However, $I(c; G(z,c))$ is difficult to optimize directly in practice since

it requires access to the posterior $P(c|x)$. Fortunately, we can have a lower bound of $I(c;G(z,c))$ by defining an auxiliary distribution $Q(c|x)$ to approximate $P(c|x)$. The final objective function of InfoGAN [9] is

$$\min_G \max_D V_I(D,G) = V(D,G) - \lambda L_I(c;Q), \tag{2}$$

where $L_I(c;Q)$ is the lower bound of $I(c;G(z,c))$. InfoGAN has several variants such as causal InfoGAN [10] and semi-supervised InfoGAN (ss-InfoGAN) [11].

2) Conditional GANs (cGANs): GANs can be extended to a conditional model if both the discriminator and generator are conditioned on some extra information y. The objective function of conditional GANs [12] is:

$$\begin{aligned}\min_G \max_D V(D,G) = &E_{x\sim p_{data}(x)}[\log D(x|y)] \\ &+E_{z\sim p_z(z)}[\log(1-D(G(z|y)))].\end{aligned} \tag{3}$$

By comparing (2) and (3), we can see that the generator of InfoGAN is similar to that of cGANs. However, the latent code c of InfoGAN is not known, and it is discovered by training. Furthermore, InfoGAN has an additional network Q to output the conditional variables $Q(c|x)$.

Based on cGANs, we can generate samples conditioning on class labels [13], [14], text [15], [16], [17], bounding box and keypoints [18]. In [17], [19], text to photo-realistic image synthesis is conducted with stacked generative adversarial networks (SGAN) [20]. cGANs have been used for convolutional face generation [21], face aging [22], image translation [23], synthesizing outdoor images having specific scenery attributes [24], natural image description [25], and 3D-aware scene manipulation [26]. Image/video colorization [27], [28], [29], [30], [31], [32], image inpainting [33], [34], image super-resolution. Besides, pix level image generation, Chrysos et al. [35] proposed robust cGANs. Thekumparampil et al. [36] discussed the robustness of conditional GANs to noisy labels. Conditional CycleGAN [37] uses cGANs with cyclic consistency. Mode seeking GANs (MSGANs) [38] proposes a simple yet effective regularization term to address the mode collapse issue for cGANs. GANs are also utilized to achieve image composition [39], [40], domain adaptation [41], [42], [43], [44].

The discriminator of original GANs [45] is trained to maximize the log-likelihood that it assigns to the correct source [13]:

$$\begin{aligned}L = &E[\log P(S=real|X_{real})] \\ &+E[\log(P(S=fake|X_{fake}))],\end{aligned} \tag{4}$$

The objective function of the auxiliary classifier GAN (AC-GAN) [13] has two parts: the loglikelihood of the correct source, L_S, and the loglikelihood of the correct class label, L_C. L_S is equivalent to L in (4). L_C is defined as

$$\begin{aligned}L_C = &E[\log P(C=c|X_{real})] \\ &+E[\log(P(C=c|X_{fake}))].\end{aligned} \tag{5}$$

The discriminator and generator of AC-GAN is to maximize $L_C + L_S$ and $L_C - L_S$, respectively. AC-GAN was the first variant of GANs that was able to produce recognizable examples of all the ImageNet [46] classes.

Discriminators of most cGANs based methods [47], [48], [49], [50], [51] feed conditional information y into the discriminator by simply concatenating (embedded) y to the input or to the feature vector at some middle layer. cGANs with projection discriminator [52] adopts an inner product between the condition vector y and the feature vector.

Isola et al. [53] used cGANs and sparse regularization for image-to-image translation. The corresponding software is called pix2pix. In GANs, the generator learns a mapping from random noise z to $G(z)$. In contrast, there is no noise input in the generator of pix2pix. A novelty of pix2pix is that the generator of pix2pix learns a mapping from an observed image y to output image $G(y)$, for example, from a grayscale image to a color image. The objective of cGANs in [53] can be expressed as

$$\begin{aligned}L_{cGANs}(D,G) = &E_{x,y}[\log D(x,y)] \\ &+E_y[\log(1-D(y,G(y)))].\end{aligned} \tag{6}$$

Furthermore, l_1 distance is used:

$$L_{l_1}(G) = E_{x,y}[\|x-G(y)\|_1]. \tag{7}$$

The final objective of [53] is

$$L_{cGANs}(D,G) + \lambda L_{l_1}(G), \tag{8}$$

where λ is the free parameter.

As a follow-up to pix2pix, pix2pixHD [54] used cGANs and feature matching loss for high-resolution image synthesis and semantic manipulation. With the discriminators, the learning problem is a multi-task learning problem:

$$\min_G \max_{D_1,D_2,D_3} \sum_{k=1,2,3} L_{GAN}(G,D_k). \tag{9}$$

The training set is given as a set of pairs of corresponding images $\{(s_i,x_i)\}$, where x_i is a natural photo and s_i is a corresponding semantic label map. The ith-layer feature extractor of discriminator D_k is denoted as $D_k^{(i)}$ (from input to the ith layer of D_k). The feature matching loss $L_{FM}(G,D_k)$ is:

$$\begin{aligned}&L_{FM}(G,D_k) = \\ &E_{(s,x)}\sum_{i=1}^{T}\frac{1}{N_i}\left[\left\|D_k^{(i)}(s,x) - D_k^{(i)}(s,G(s))\right\|_1\right],\end{aligned} \tag{10}$$

where N_i is the number of elements in each layer and T denotes the total number of layers. The final objective function of [54] is

$$\min_G \max_{D_1,D_2,D_3} \sum_{k=1,2,3} (L_{GAN}(G,D_k) + \lambda L_{FM}(G,D_k)). \tag{11}$$

3) CycleGAN: Image-to-image translation is a class of graphics and vision problems where the goal is to learn the mapping between an output image and an input image using a training set of aligned image pairs. When paired training data is available, reference [53] can be used for these image-to-image translation tasks. However, reference [53] can not be used for unpaired data (no input/output pairs), which was well solved by Cycle-consistent GANs (CycleGAN) [55]. CycleGAN is an important progress for unpaired data. It is

proved that cycle-consistency is an upper bound of the conditional entropy [56]. CycleGAN can be derived as a special case within the proposed variational inference (VI) framework [57], naturally establishing its relationship with approximate Bayesian inference methods. Specifically, GAN models has been broadly applied in image synthesis [58], [59], [60], [61], [62], The basic idea of DiscoGAN [63] and CycleGAN [55] is nearly the same. Both of them were proposed separately nearly at the same time. The only difference between CycleGAN [55] and DualGAN [64] is that DualGAN uses the loss format advocated by Wasserstein GAN (WGAN) rather than the sigmoid cross-entropy loss used in CycleGAN. Some other works utilize GAN to achieve style transfer [55], [63], [64], [65], [66], [67].

4) f-GAN: As we know, Kullback-Leibler (KL) divergence measures the difference between two given probability distributions. A large class of assorted divergences are the so called Ali-Silvey distances, also known as the f-divergences [68]. Given two probability distributions P and Q which have, respectively, an absolutely continuous density function p and q with regard to a base measure dx defined on the domain X, the f-divergence is defined,

$$D_f\left(P\left\|Q\right.\right)=\int_X q\left(x\right)f\left(\frac{p\left(x\right)}{q\left(x\right)}\right)dx. \tag{12}$$

Different choices of f recover popular divergences as special cases of f-divergence. For example, if $f\left(a\right)=a\log a$, f-divergence becomes KL divergence. The original GANs [45] is a special case of f-GAN [69] which is based on f-divergence. The reference [69] shows that any f-divergence can be used for training GAN. Furthermore, the reference [69] discusses the advantages of different choices of divergence functions on both the quality of the produced generative models and training complexity. Im et al. [70] quantitatively evaluated GANs with divergences proposed for training. Uehara et al. [71] extend the f-GAN further, where the f-divergence is directly minimized in the generator step and the ratio of the distributions of real and generated data are predicted in the discriminator step.

5) Integral Probability Metrics (IPMs): Denoting $\mathcal{P}$ the set of all Borel probability measures on a topological space $(M,\mathcal{A})$. The integral probability metric (IPM) [72] between two probability distributions $P\in\mathcal{P}$ and $Q\in\mathcal{P}$ is defined as

$$\gamma_{\mathcal{F}}(P,Q)=\sup_{f\in\mathcal{F}}\left|\int_M fdP-\int_M fdQ\right|, \tag{13}$$

where $\mathcal{F}$ is a class of real-valued bounded measurable functions on M. Nonparametric density estimation and convergence rates for GANs under Besov IPM Losses is discussed in [73]. IPMs include such as RKHS-induced maximum mean discrepancy (MMD) as well as the Wasserstein distance used in Wasserstein GANs (WGAN).

a) Maximum Mean Discrepancy (MMD):
The maximum mean discrepancy (MMD) [74] is a measure of the difference between two distributions P and Q given by the supremum over a function space $\mathcal{F}$ of differences between the expectations with regard to two distributions. The MMD is defined by:

$$\begin{aligned}&MMD(\mathcal{F},P,Q)=\\&\sup_{f\in\mathcal{F}}\left(E_{X\sim P}\left[f\left(X\right)\right]-E_{Y\sim Q}\left[f\left(Y\right)\right]\right).\end{aligned} \tag{14}$$

MMD has been used for deep generative models [75], [76], [77] and model criticism [78].

b) Wasserstein GAN (WGAN):
WGAN [79] conducted a comprehensive theoretical analysis of how the Earth Mover (EM) distance behaves in comparison with popular probability distances and divergences such as the total variation (TV) distance, the Kullback-Leibler (KL) divergence, and the Jensen-Shannon (JS) divergence utilized in the context of learning distributions. The definition of the EM distance is

$$W(p_{data},p_g)=\inf_{\gamma\in\Pi(p_{data},p_g)}E_{(x,y)\in\gamma}\left[\|x-y\|\right], \tag{15}$$

where $\Pi\left(p_{data},p_g\right)$ denotes the set of all joint distributions $\gamma\left(x,y\right)$ whose marginals are p_{data} and p_g, respectively. However, the infimum in (15) is highly intractable. The reference [79] uses the following equation to approximate the EM distance

$$\max_{w\in\mathcal{W}}E_{x\sim p_{data}(x)}\left[f_w\left(x\right)\right]-E_{z\sim p_z(z)}\left[f_w\left(G\left(z\right)\right)\right], \tag{16}$$

where there is a parameterized family of functions $\{f_w\}_{w\in\mathcal{W}}$ that are all K-Lipschitz for some K and f_w can be realized by the discriminator D. When D is optimized, (16) denotes the approximated EM distance. Then the aim of G is to minimize (16) to make the generated distribution as close to the real distribution as possible. Therefore, the overall objective function of WGAN is

$$\begin{aligned}&\min_G\max_{w\in\mathcal{W}}E_{x\sim p_{data}(x)}\left[f_w\left(x\right)\right]-E_{z\sim p_z(z)}\left[f_w\left(G\left(z\right)\right)\right]\\&=\min_G\max_D E_{x\sim p_{data}(x)}\left[D\left(x\right)\right]-E_{z\sim p_z(z)}\left[D\left(G\left(z\right)\right)\right].\end{aligned} \tag{17}$$

we can see three differences between the objective function of original GANs and that of WGAN:

- First, there is no log in the objective function of WGAN.
- Second, the D in original GANs is utilized as a binary classifier while D utilized in WGAN is to approximate the Wasserstein distance, which is a regression task. Therefore, the sigmoid in the last layer of D is not used in the WGAN. The output of the discriminator of the original GANs is between zero and one while there is no constraint for that of WGAN.
- Third, the D in WGAN is required to be K-Lipschitz for some K and therefore WGAN uses weight clipping.

Compared with traditional GANs training, WGAN can improve the stability of learning and provide meaningful learning curves useful for hyperparameter searches and debugging. However, it is a challenging task to approximate the K-Lipschitz constraint which is required by the Wasserstein-1 metric. WGAN-GP [80] is proposed by utilizing gradient penalty for restricting K-Lipschitz constraint and the objective function is

$$\begin{aligned}L=&-E_{x\sim p_{data}}\left[D\left(x\right)\right]+E_{\tilde{x}\sim p_g}\left[D\left(\tilde{x}\right)\right]\\&+\lambda E_{\hat{x}\sim p_{\hat{x}}}\left[\left(\left\|\nabla_{\hat{x}}D\left(\hat{x}\right)\right\|_2-1\right)^2\right]\end{aligned} \tag{18}$$

where the first two terms are the objective function of WGAN and $\hat{x}$ is sampled from the distribution $p_{\hat{x}}$ which samples uniformly along straight lines between pairs of points sampled from the real data distribution p_{data} and the generated distribution p_g. There are some other methods closely related to WGAN-GP such as DRAGAN [81]. Wu et al. [82] propose a novel and relaxed version of Wasserstein-1 metric: Wasserstein divergence (W-div), which does not require the K-Lipschitz constraint. Based on W-div, Wu et al. [82] introduce a Wasserstein divergence objective for GANs (WGAN-div), which can faithfully approximate W-div by optimization. CramerGAN [83] argues that the Wasserstein distance leads to biased gradients, suggesting the Cramér distance between two distributions. Other papers related to WGAN can be found in [84], [85], [86], [87], [88], [89].

6) Loss Sensitive GAN (LS-GAN): Similar to WGAN, LS-GAN [90] also has a Lipschitz constraint. It is assumed in LS-GAN that p_{data} lies in a set of Lipschitz densities with a compact support. In LS-GAN , the loss function $L_\theta(x)$ is parameterized with θ and LS-GAN assumes that a generated sample should have larger loss than a real one. The loss function can be trained to satisfy the following constraint:

$$L_\theta(x) \leq L_\theta(G(z)) - \Delta(x, G(z)) \tag{19}$$

where $\Delta(x, G(z))$ is the margin measuring the difference between generated sample $G(z)$ and real sample x. The objective function of LS-GAN is

$$\begin{aligned} &\min_D \mathcal{L}_D = E_{x\sim p_{data}(x)}\left[L_\theta(x)\right] \\ &+\lambda E_{\substack{x\sim p_{data}(x),\\ z\sim p_z(z)}}\left[\Delta(x, G(z)) + L_\theta(x) - L_\theta(G(z))\right]_+, \end{aligned} \tag{20}$$

$$\min_G \mathcal{L}_G = E_{z\sim p_z(z)}\left[L_\theta(G(z))\right], \tag{21}$$

where $[y]^+ = \max(0, y)$, λ is the free tuning-parameter, and θ is the paramter of the discriminator D.

7) Summary: There is a website called "The GAN Zoo" (https://github.com/hindupuravinash/the-gan-zoo) which lists many GANs' variants. Please refer to this website for more details.

A. GANs Training

Despite the theoretical existence of unique solutions, GANs training is hard and often unstable for several reasons [91], [92], [93]. One difficulty is from the fact that optimal weights for GANs correspond to saddle points, and not minimizers, of the loss function.

There are many papers on GANs training. Yadav et al. [94] stabilized GANs with prediction methods. By using independent learning rates, [95] proposed a two time-scale update rule (TTUR) for both discriminator and generator to ensure that the model can converge to a stable local Nash equilibrium. Arjovsky [93] made theoretical steps towards fully understanding the training dynamics of GANs; analyzed why GANs was hard to train; studied and proved rigorously the problems including saturation and instability that occurred when training GANs; examined a practical and theoretically grounded direction to mitigate these problems; and introduced new tools to study them. Liang et al. [96] think that GANs training is a continual learning problem [97].

One method to improve GANs training is to assess the empirical "symptoms" that might occur in training. These symptoms include: the generative model collapsing to produce very similar samples for diverse inputs [92]; the discriminator loss converging quickly to zero [93], providing no gradient updates to the generator; difficulties in making the pair of models converge [91].

We will introduce GANs training from three perspectives: objective function, skills, and structure.

III. Methods

The FID score is calculated by first loading a pre-trained Inception v3 model. The output layer of the model is removed and the output is taken as the activations from the last pooling layer, a global spatial pooling layer. This output layer has 2,048 activations, therefore, each image is predicted as 2,048 activation features. This is called the coding vector or feature vector for the image. A 2,048 feature vector is then predicted for a collection of real images from the problem domain to provide a reference for how real images are represented. Feature vectors can then be calculated for synthetic images. The result will be two collections of 2,048 feature vectors for real and generated images. Implementing the calculation of the FID score in Python with NumPy arrays is straightforward. First, let's define a function that will take a collection of activations for real and generated images and return the FID score. The calculate fid function listed below implements the procedure. Here, we implement the FID calculation almost directly. It is worth noting that the official implementation in TensorFlow implements elements of the calculation in a slightly different order, likely for efficiency, and introduces additional checks around the matrix square root to handle possible numerical instabilities. I recommend reviewing the official implementation and extending the implementation below to add these checks if you experience problems calculating the FID on your own datasets. We can then test out this function to calculate the inception score for some contrived feature vectors.

Feature vectors will probably contain small positive values and will have a length of 2,048 elements. We can construct two lots of 10 images worth of feature vectors with small random numbers as follows: One test would be to calculate the FID between a set of activations and itself, which we would expect to have a score of 0.0. We can then calculate the distance between the two sets of random activations, which we would expect to be a large number. Now that we know how to calculate the FID score and to implement it in NumPy, we can develop an implementation in Keras. This involves the preparation of the image data and using a pretrained Inception v3 model to calculate the activations or feature vectors for each image. First, we can load the Inception v3 model in Keras directly. This will prepare a version of the inception model for classifying images as one of 1,000 known classes. We can remove the output (the top) of the model via the

include top=False argument. Painfully, this also removes the global average pooling layer that we require, but we can add it back via specifying the pooling='avg' argument. When the output layer of the model is removed, we must specify the shape of the input images, which is 299x299x3 pixels, e.g. the input shape=(299,299,3) argument. Therefore, the inception model can be loaded. This model can then be used to predict the feature vector for one or more images. Our images are likely to not have the required shape. We will use the scikit-image library to resize the NumPy array of pixel values to the required size. The scale images() function below implements this. Once resized, the image pixel values will also need to be scaled to meet the expectations for inputs to the inception model. This can be achieved by calling the preprocess input() function. We can update our calculate fid() function defined in the previous section to take the loaded inception model and two NumPy arrays of image data as arguments, instead of activations. The function will then calculate the activations before calculating the FID score as before. The updated version of the calculate fid() function is listed below. It may be useful to calculate the FID score between two collections of real images. The Keras library provides a number of computer vision datasets, including the CIFAR-10 dataset. These are color photos with the small size of 32×32 pixels and is split into train and test elements and can be loaded The training dataset has 50,000 images, whereas the test dataset has only 10,000 images. It may be interesting to calculate the FID score between these two datasets to get an idea of how representative the test dataset is of the training dataset. Scaling and scoring 50K images takes a long time, therefore, we can reduce the "training set" to a 10K random sample.

In this tutorial, you discovered how to implement the Frechet Inception Distance for evaluating generated images. Specifically, you learned: The Frechet Inception Distance summarizes the distance between the Inception feature vectors for real and generated images in the same domain. How to calculate the FID score and implement the calculation from scratch in NumPy. How to implement the FID score using the Keras deep learning library and calculate it with real images.

REFERENCES

[1] M. Kocaoglu, C. Snyder, A. G. Dimakis, and S. Vishwanath, "Causalgan: Learning causal implicit generative models with adversarial training," *arXiv preprint arXiv:1709.02023*, 2017.

[2] S. Feizi, F. Farnia, T. Ginart, and D. Tse, "Understanding gans: the lqg setting," *arXiv preprint arXiv:1710.10793*, 2017.

[3] F. Farnia and D. Tse, "A convex duality framework for gans," in *Neural Information Processing Systems*, 2018, pp. 5248–5258.

[4] J. Zhao, J. Li, Y. Cheng, T. Sim, S. Yan, and J. Feng, "Understanding humans in crowded scenes: Deep nested adversarial learning and a new benchmark for multi-human parsing," in *ACM Multimedia Conference on Multimedia Conference*. ACM, 2018, pp. 792–800.

[5] A. Jahanian, L. Chai, and P. Isola, "On the"steerability" of generative adversarial networks," *arXiv preprint arXiv:1907.07171*, 2019.

[6] B. Zhu, J. Jiao, and D. Tse, "Deconstructing generative adversarial networks," *arXiv preprint arXiv:1901.09465*, 2019.

[7] K. B. Kancharagunta and S. R. Dubey, "Csgan: cyclic-synthesized generative adversarial networks for image-to-image transformation," *arXiv preprint arXiv:1901.03554*, 2019.

[8] Y. Wu, J. Donahue, D. Balduzzi, K. Simonyan, and T. Lillicrap, "Logan: Latent optimisation for generative adversarial networks," *arXiv preprint arXiv:1912.00953*, 2019.

[9] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Neural Information Processing Systems*, 2016, pp. 2172–2180.

[10] T. Kurutach, A. Tamar, G. Yang, S. J. Russell, and P. Abbeel, "Learning plannable representations with causal infogan," in *Neural Information Processing Systems*, 2018, pp. 8733–8744.

[11] A. Spurr, E. Aksan, and O. Hilliges, "Guiding infogan with semi-supervision," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2017, pp. 119–134.

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

[13] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in *International Conference on Machine Learning*, 2017, pp. 2642–2651.

[14] A. Nguyen, J. Clune, Y. Bengio, A. Dosovitskiy, and J. Yosinski, "Plug & play generative networks: Conditional iterative generation of images in latent space," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4467–4477.

[15] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *International Conference on Machine Learning*, 2016, pp. 1–10.

[16] S. Hong, D. Yang, J. Choi, and H. Lee, "Inferring semantic layout for hierarchical text-to-image synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7986–7994.

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

[18] S. E. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, and H. Lee, "Learning what and where to draw," in *Neural Information Processing Systems*, 2016, pp. 217–225.

[19] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. Metaxas, "Stackgan++: Realistic image synthesis with stacked generative adversarial networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1947–1962, 2019.

[20] X. Huang, Y. Li, O. Poursaeed, J. Hopcroft, and S. Belongie, "Stacked generative adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5077–5086.

[21] J. Gauthier, "Conditional generative adversarial nets for convolutional face generation," *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester*, vol. 2014, no. 5, p. 2, 2014.

[22] G. Antipov, M. Baccouche, and J.-L. Dugelay, "Face aging with conditional generative adversarial networks," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 2089–2093.

[23] H. Tang, D. Xu, N. Sebe, Y. Wang, J. J. Corso, and Y. Yan, "Multi-channel attention selection gan with cascaded semantic guidance for cross-view image translation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2417–2426.

[24] L. Karacan, Z. Akata, A. Erdem, and E. Erdem, "Learning to generate images of outdoor scenes from attributes and semantic layouts," *arXiv preprint arXiv:1612.00215*, 2016.

[25] B. Dai, S. Fidler, R. Urtasun, and D. Lin, "Towards diverse and natural image descriptions via a conditional gan," in *IEEE International Conference on Computer Vision*, 2017, pp. 2970–2979.

[26] S. Yao, T. M. Hsu, J.-Y. Zhu, J. Wu, A. Torralba, B. Freeman, and J. Tenenbaum, "3d-aware scene manipulation via inverse graphics," in *Neural Information Processing Systems*, 2018, pp. 1887–1898.

[27] R. Zhang, J.-Y. Zhu, P. Isola, X. Geng, A. S. Lin, T. Yu, and A. A. Efros, "Real-time user-guided image colorization with learned deep priors," *arXiv preprint arXiv:1705.02999*, 2017.

[28] P. L. Suárez, A. D. Sappa, and B. X. Vintimilla, "Infrared image colorization based on a triplet dcgan architecture," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 18–23.

[29] M. He, D. Chen, J. Liao, P. V. Sander, and L. Yuan, "Deep exemplar-based colorization," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–16, 2018.

[30] B. Zhang, M. He, J. Liao, P. V. Sander, L. Yuan, A. Bermak, and D. Chen, "Deep exemplar-based video colorization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8052–8061.

[31] Z. Xu, T. Wang, F. Fang, Y. Sheng, and G. Zhang, "Stylization-based architecture for fast deep exemplar colorization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9363–9372.

[32] J. Lee, E. Kim, Y. Lee, D. Kim, J. Chang, and J. Choo, "Reference-based sketch image colorization using augmented-self reference and dense semantic correspondence," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[33] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[34] Y. Song, C. Yang, Z. Lin, X. Liu, Q. Huang, H. Li, and C.-C. J. Kuo, "Contextual-based image inpainting: Infer, match, and translate," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[35] G. G. Chrysos, J. Kossaifi, and S. Zafeiriou, "Robust conditional generative adversarial networks," *arXiv preprint arXiv:1805.08657*, 2018.

[36] K. K. Thekumparampil, A. Khetan, Z. Lin, and S. Oh, "Robustness of conditional gans to noisy labels," in *Neural Information Processing Systems*, 2018, pp. 10 271–10 282.

[37] Y. Lu, Y.-W. Tai, and C.-K. Tang, "Conditional cyclegan for attribute guided face image generation," *arXiv preprint arXiv:1705.09966*, 2017.

[38] Q. Mao, H.-Y. Lee, H.-Y. Tseng, S. Ma, and M.-H. Yang, "Mode seeking generative adversarial networks for diverse image synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1429–1437.

[39] F. Zhan, H. Zhu, and S. Lu, "Spatial fusion gan for image synthesis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 3653–3662.

[40] ——, "Scene text synthesis for efficient and effective deep network training," *arXiv preprint arXiv:1901.09193*, 2019.

[41] Z. Murez, S. Kolouri, D. Kriegman, R. Ramamoorthi, and K. Kim, "Image to image translation for domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[42] J. Cao, O. Katzir, P. Jiang, D. Lischinski, D. Cohen-Or, C. Tu, and Y. Li, "Dida: Disentangled synthesis for domain adaptation," 2018.

[43] A. H. Liu, Y.-C. Liu, Y.-Y. Yeh, and Y.-C. F. Wang, "A unified feature disentangler for multi-domain image translation and manipulation," in *Advances in neural information processing systems*, 2018, pp. 2590–2599.

[44] F. Zhan, C. Xue, and S. Lu, "Ga-dan: Geometry-aware domain adaptation network for scene text detection and recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 9105–9115.

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

[46] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[47] E. L. Denton, S. Chintala, R. Fergus *et al.*, "Deep generative image models using a laplacian pyramid of adversarial networks," in *Neural Information Processing Systems*, 2015, pp. 1486–1494.

[48] G. Perarnau, J. Van De Weijer, B. Raducanu, and J. M. Álvarez, "Invertible conditional gans for image editing," *arXiv preprint arXiv:1611.06355*, 2016.

[49] M. Saito, E. Matsumoto, and S. Saito, "Temporal generative adversarial nets with singular value clipping," in *IEEE International Conference on Computer Vision*, 2017, pp. 2830–2839.

[50] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," *arXiv preprint arXiv:1606.00704*, 2016.

[51] K. Sricharan, R. Bala, M. Shreve, H. Ding, K. Saketh, and J. Sun, "Semi-supervised conditional gans," *arXiv preprint arXiv:1708.05789*, 2017.

[52] T. Miyato and M. Koyama, "cgans with projection discriminator," *arXiv preprint arXiv:1802.05637*, 2018.

[53] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1125–1134.

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

[55] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *International Conference on Computer Vision*, 2017, pp. 2223–2232.

[56] C. Li, H. Liu, C. Chen, Y. Pu, L. Chen, R. Henao, and L. Carin, "Alice: Towards understanding adversarial learning for joint distribution matching," in *Neural Information Processing Systems*, 2017, pp. 5495–5503.

[57] L. C. Tiao, E. V. Bonilla, and F. Ramos, "Cycle-consistent adversarial learning as approximate bayesian inference," *arXiv preprint arXiv:1806.01771*, 2018.

[58] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[59] P. Zhu, R. Abdal, Y. Qin, and P. Wonka, "Sean: Image synthesis with semantic region-adaptive normalization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[60] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "Maskgan: Towards diverse and interactive facial image manipulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5549–5558.

[61] H. Tang, D. Xu, Y. Yan, P. H. Torr, and N. Sebe, "Local class-specific and global image-level generative adversarial networks for semantic-guided scene generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[62] F. Zhan, Y. Yu, K. Cui, G. Zhang, S. Lu, J. Pan, C. Zhang, F. Ma, X. Xie, and C. Miao, "Unbalanced feature transport for exemplar-based image translation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[63] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," in *International Conference on Machine Learning*, 2017, pp. 1857–1865.

[64] Z. Yi, H. Zhang, P. Tan, and M. Gong, "Dualgan: Unsupervised dual learning for image-to-image translation," in *International Conference on Computer Vision*, 2017, pp. 2849–2857.

[65] Y. A. Mejjati, C. Richardt, J. Tompkin, D. Cosker, and K. I. Kim, "Unsupervised attention-guided image-to-image translation," in *Advances in Neural Information Processing Systems*, 2018, pp. 3693–3703.

[66] F. Zhan and C. Zhang, "Spatial-aware gan for unsupervised person re-identification," *Proceedings of the International Conference on Pattern Recognition*, 2020.

[67] M. Tomei, M. Cornia, L. Baraldi, and R. Cucchiara, "Art2real: Unfolding the reality of artworks via semantically-aware image-to-image translation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5849–5859.

[68] I. Csiszár, P. C. Shields *et al.*, "Information theory and statistics: A tutorial," *Foundations and Trends® in Communications and Information Theory*, vol. 1, no. 4, pp. 417–528, 2004.

[69] S. Nowozin, B. Cseke, and R. Tomioka, "f-gan: Training generative neural samplers using variational divergence minimization," in *Neural Information Processing Systems*, 2016, pp. 271–279.

[70] D. J. Im, H. Ma, G. Taylor, and K. Branson, "Quantitatively evaluating gans with divergences proposed for training," in *International Conference on Learning Representation*, 2018.

[71] M. Uehara, I. Sato, M. Suzuki, K. Nakayama, and Y. Matsuo, "Generative adversarial nets from a density ratio estimation perspective," *arXiv preprint arXiv:1610.02920*, 2016.

[72] B. K. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G. R. Lanckriet, "Hilbert space embeddings and metrics on probability measures," *Journal of Machine Learning Research*, vol. 11, no. Apr, pp. 1517–1561, 2010.

[73] A. Uppal, S. Singh, and B. Poczos, "Nonparametric density estimation & convergence of gans under besov ipm losses," in *Neural Information Processing Systems*, 2019, pp. 9086–9097.

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

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

[76] C.-L. Li, W.-C. Chang, Y. Cheng, Y. Yang, and B. Póczos, "Mmd gan: Towards deeper understanding of moment matching network," in *Neural Information Processing Systems*, 2017, pp. 2203–2213.

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

[78] D. J. Sutherland, H.-Y. Tung, H. Strathmann, S. De, A. Ramdas, A. Smola, and A. Gretton, "Generative models and model crit-

icism via optimized maximum mean discrepancy," *arXiv preprint arXiv:1611.04488*, 2016.
[79] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning*, 2017, pp. 214–223.
[80] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Neural Information Processing Systems*, 2017, pp. 5767–5777.
[81] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, "On convergence and stability of gans," *arXiv preprint arXiv:1705.07215*, 2017.
[82] J. Wu, Z. Huang, J. Thoma, D. Acharya, and L. Van Gool, "Wasserstein divergence for gans," in *European Conference on Computer Vision*, 2018, pp. 653–668.
[83] M. G. Bellemare, I. Danihelka, W. Dabney, S. Mohamed, B. Lakshminarayanan, S. Hoyer, and R. Munos, "The cramer distance as a solution to biased wasserstein gradients," *arXiv preprint arXiv:1705.10743*, 2017.
[84] H. Petzka, A. Fischer, and D. Lukovnicov, "On the regularization of wasserstein gans," in *International Conference on Learning Representations*, 2018, pp. 1–24.
[85] F. Juefei-Xu, V. N. Boddeti, and M. Savvides, "Gang of gans: Generative adversarial networks with maximum margin ranking," *arXiv preprint arXiv:1704.04865*, 2017.
[86] C.-C. Hsu, H.-T. Hwang, Y.-C. Wu, Y. Tsao, and H.-M. Wang, "Voice conversion from unaligned corpora using variational autoencoding wasserstein generative adversarial networks," in *Interspeech*, 2017, pp. 3364–3368.
[87] J. Adler and S. Lunz, "Banach wasserstein gan," in *Neural Information Processing Systems*, 2018, pp. 6754–6763.
[88] Y.-S. Chen, Y.-C. Wang, M.-H. Kao, and Y.-Y. Chuang, "Deep photo enhancer: Unpaired learning for image enhancement from photographs with gans," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6306–6314.
[89] S. Athey, G. Imbens, J. Metzger, and E. Munro, "Using wasserstein generative adversial networks for the design of monte carlo simulations," *arXiv preprint arXiv:1909.02210*, 2019.
[90] G.-J. Qi, "Loss-sensitive generative adversarial networks on lipschitz densities," *International Journal of Computer Vision*, pp. 1–23, 2019.
[91] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.
[92] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Neural Information Processing Systems*, 2016, pp. 2234–2242.
[93] M. Arjovsky and L. Bottou:, "Towards principled methods for training generative adversarial networks," in *International Conference on Learning Representations*, 2017.
[94] A. Yadav, S. Shah, Z. Xu, D. Jacobs, and T. Goldstein, "Stabilizing adversarial nets with prediction methods," *arXiv preprint arXiv:1705.07364*, 2017.
[95] 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 *Neural Information Processing Systems*, 2017, pp. 6626–6637.
[96] K. J. Liang, C. Li, G. Wang, and L. Carin, "Generative adversarial network training is a continual learning problem," *arXiv preprint arXiv:1811.11083*, 2018.
[97] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," in *Advances in Neural Information Processing Systems*, 2017, pp. 2990–2999.