A Simple yet Effective Way for Improving the Performance of GANs

Yong-Goo Shin, Yoon-Jae Yeo, Min-Cheol Sagong, Cheol-Hwan Yoo, Sung-jea Ko
Korea university
{ygshin, yjyeo, mcsagong, chyyoo}@dali.korea.ac.kr
sjko@korea.ac.kr

Abstract

This paper presents a simple yet effective way that improves the performance of generative adversarial networks (GANs) without imposing the training overhead or modifying the network architectures of existing methods. The proposed method employs a novel cascading rejection (CR) module for discriminator, which extracts multiple non-overlapped features in an iterative manner. The CR module supports the discriminator to effectively distinguish between real and generated images, which results in a strong penalization to the generator. In order to deceive the robust discriminator containing the CR module, the generator produces the images that are more similar to the real images. Since the proposed CR module requires only a few simple vector operations, it can be readily applied to existing frameworks with marginal training overheads. Quantitative evaluations on various datasets including CIFAR-10, Celeb-HQ, LSUN, and tiny-ImageNet confirm that the proposed method significantly improves the performance of GANs and conditional GANs in terms of Frechet inception distance (FID) indicating the diversity and visual appearance of the generated images.

1. Introduction

Generative adversarial networks (GANs) \cite{gan} based on deep convolutional neural networks (CNNs) have shown considerable success to capture complex and high-dimensional image data, and been utilized to numerous applications including image-to-image translation \cite{4,5,6}, image inpainting \cite{22,26,24}, and text-to-image translation \cite{23,11}. Despite the recent advances, however, the training of the GANs is known to be unstable and sensitive to the choices of hyper-parameters \cite{33}. To address this problem, some researchers proposed the novel generator and discriminator structures \cite{13,33,35}. These methods effectively improves image generation task on challenging datasets such as ImageNet \cite{15} but are difficult to apply to other various applications since they impose training overhead or need to modify the network architectures.

Several works \cite{1,8,19,22,17,3} attempted to stabilize the training of the GANs by using novel loss functions or regularization terms. Arjovsky et al. \cite{1} applied the Wasserstein distance to adversarial loss function, which shows better training stability than the original loss function. Gulrajani et al. \cite{8} extended the method in \cite{1} by adding the gradient regularization term, called gradient penalty, to further stabilize the training procedure. Miyato et al. \cite{19} proposed the weight normalization technique called spectral normalization which limits the spectral norm of weight matrices to stabilize the training of the discriminator. By combining the projection-based discriminator in \cite{20}, this approach significantly improved the performance of the image generation task on ImageNet \cite{15}.

Recently, Chen et al. \cite{3} combined the GANs with self-supervised learning by adding an auxiliary loss function. Although this method improves the performance of the GANs, it needs an additional task-specific network and objective functions for self-supervised learning, which results in extra computational loads on training procedure. Mao et al. \cite{17} proposed a simple regularization term which maximizes the ratio of the distance between images with respect to the distance between latent vectors. This technique imposes no training overheads and does not require the network structure modification, which makes it readily applicable to various applications.

Inspired by the method in \cite{17}, this paper presents a simple yet effective way that greatly improves the performance of the GANs without modifying the original network architectures or imposing the training overhead. In general, the discriminator of GANs extracts features using multiple convolutional layers and predicts whether the input image is real or fake using a fully connected layer which produces a single scalar value, i.e. probability value. Indeed, the operation of fully connected layer predicting the single probability value is equivalent to the inner product operation. In other words, to predict the probability, this layer conducts the inner product between a single embedding vector, i.e. a weight vector, and an image feature vector obtained via
CNNs. In the inner product process, however, the discriminator unintentionally ignores the part of the feature space which is perpendicular to the weight vector. Since the generator is trained through adversarial learning which focuses on deceiving the discriminator, it produces an image without considering the ignored feature space. For instance, if the discriminator first learns the global structure for distinguishing between the real and generated images, the generator will naturally attempt to produce the images having a similar global structure with real images without considering local structure. In other words, the generator fails to fully capture the complex and high-dimensional feature space on the image data.

To alleviate this problem, we propose a novel cascading rejection (CR) module which extracts different features in an iterative procedure. The CR module leads the discriminator to effectively distinguish between real and generated images, which results in a strong penalization to the generator. In order to deceive the robust discriminator having the CR module, the generator generates the images that are more similar to the real images. Since the proposed CR module needs only a few simple vector operations, it can be readily applied to existing frameworks with marginal training overheads. We conducted extensive experiments on various datasets including CIFAR-10[28], Celeb-HQ[13][16], LSUN[31], and tiny-ImageNet[5][30]. Experimental results show that the proposed method significantly improves the performance of the GANs and conditional GANs in terms of the FID.

2. Preliminaries

2.1. Generative adversarial networks

Typically, the GANs consist of the generator $G$ and the discriminator $D$. In the GANs, both networks are simultaneously trained: $G$ is trained to create a new image which is indistinguishable from real images, whereas $D$ is optimized to differentiate between real and generated images. This relation can be considered as a two-player min-max game where $G$ and $D$ compete with each other. Formally, the $G (D)$ is trained to minimize (maximize) the loss function, called adversarial loss, as follows:

$$\min_G \max_D \mathbb{E}_{x \sim P_{\text{data}}(x)} [\log D(x)]$$
$$+ \mathbb{E}_{z \sim P_z(z)} [\log (1 - D(G(z)))],$$

(1)

where $z$ and $x$ denote a random noise vector and a real image sampled from the noise $P_z(z)$ and real data distribution $P_{\text{data}}(x)$, respectively. It is worth noting that $D(x)$ and $D(G(z))$ are scalar values indicating the probabilities that $x$ and $G(z)$ came from the data distribution.

The conditional GANs (cGANs) which aim at producing the class conditional images have been actively researched[18][22][19][34]. The cGANs techniques usually add conditional information $c$, such as class labels or text condition, to both generator and discriminator in order to control the data generation process in a supervised manner. This can be formally expressed as follows:

$$\min_G \max_D \mathbb{E}_{(x,c) \sim P_{\text{data}}(x)} [\log D(x)]$$
$$+ \mathbb{E}_{z \sim P_z(z), c \sim P_{\text{data}}(z)} [\log (1 - D(G(z,c)))].$$

(2)
By training the networks based on the above equation, the generator can select an image category to be generated, which is not possible when employing the standard GANs framework.

2.2. Revisit the Fully Connected Layer

To train the discriminator using Eqs. [1] and [2] the discriminator should produce the single scalar value as an output. To this end, in the last layer, the discriminator usually employs a fully-connected layer with a single output channel which acts like the inner product between an embedding vector \( w \), i.e. weight vector, and an image feature vector \( v \) obtained through the multiple convolutional layers. Even if the last layer consists of several pixels such as PatchGAN [12], the discriminator conducts the inner product for each pixel and averages all values for the adversarial loss function.

The inner product of \( v \) onto \( w \) is illustrated in Fig.1. As shown in Fig. 1, the inner product produces the scalar value \( V_P \), but it ignores the part of the feature space which is perpendicular to the \( w \). In other words, the discriminator only considers the feature space which is parallel to the \( w \) when predicting the probability value for the adversarial loss. This problem often makes the discriminator difficult to effectively penalize the generator. For instance, as shown in Fig. 2, even if the generator produces low-quality images having the ignored feature space, the discriminator cannot distinguish between real and generated images. In other words, the generator can minimize the adversarial loss by producing the low-quality images having the ignored feature space, which results in the performance degradation of the generator. To alleviate this problem, this paper proposes the CR module which encourages the discriminator to consider ignored feature space in the last layer.

3. Proposed Method

3.1. Cascading rejection module

In Euclidean space, the fully-connected layer producing a single scalar value as an output is equivalent to the inner product \( P(v, w) \) as follows:

\[
P(v, w) = v^T w
\]  

(3)

where \( v \) and \( w \) indicate the input feature vector and the embedding vector, i.e. weight vector, in the last fully connected layer, respectively. From this formulation, we observe that the ignored feature caused by the inner product, \( \hat{v} \), can be obtained by the vector rejection of \( v \) from \( w \), which is defined as follows:

\[
\hat{v} = v - \frac{v^T w}{w^T w} w.
\]  

(4)

In other words, by minimizing the adversarial loss using the additional probability value obtained through the inner product of \( \hat{v} \) and another weight vector \( \hat{w} \), the discriminator is able to consider the ignored feature space.

Based on these observations, we propose the CR module which iteratively conducts the inner product and vector rejection processes. Fig.2 illustrates the proposed CR module, where \( v_1 \) indicates the input feature vector of the CR module, which is obtained through the multiple convolutional layers in the discriminator. The iterative vector rejection process generates \( N - 1 \) vectors, i.e. \( v_i, \{i = 2, \ldots, N\} \), which represent the ignored feature in the previous inner product operation, whereas the iterative inner product produces \( N \) scalar values, i.e. \( p_i, \{i = 1, \ldots, N\} \), which indicate the probabilities that the \( v_i \) came from the real data distribution. Note that \( v_i \)'s are non-overlapped with each other since they obtained through the vector rejection operation. By using the probabilities obtained via the CR module, the adversarial loss of the discriminator \( L_D \) and that of the generator \( L_G \) can be rewritten as

\[
L_D = -E\left[\frac{1}{N} \sum_{i=1}^{N} \lambda_i \log(p_{x,i})\right] - E\left[\frac{1}{N} \sum_{i=1}^{N} \lambda_i (1 - \log(p_{z,i}))\right],
\]  

(5)

\[
L_G = E\left[\frac{1}{N} \sum_{i=1}^{N} \lambda_i (1 - \log(p_{z,i}))\right],
\]  

(6)
where \( p_{x,i} \) (\( p_{z,i} \)) indicates the \( i \)-th probability value when the input of discriminator is real (generated) image, and \( \lambda_i \) is a hyper parameter which controls the relative importance of each loss term. In order to predict the probability using the more essential features in the earlier stage of the CR module, we set \( \lambda_i \) as follows:

\[
\lambda_i = \frac{0.5^{i-1}}{N \sum_{i=1}^{N} (0.5^{i-1})}.
\]

When \( N \) is one, the loss functions in Eqs. 5 and 6 are equivalent to the original one in Eq. 1. In contrast, when \( N \) is larger than one, the discriminator and generator should consider the ignored feature space to minimize the \( L_D \) and \( L_G \), respectively. It is worth noting that since the iterative inner product process and vector rejection process are simple vector operations, the proposed CR module does not impose the training overhead. In addition, since the proposed CR module is appended after the last fully connected layer in the discriminator, there is no modification to the existing architecture of the discriminator.

3.2. Conditional Cascading Rejection module

In this subsection, we introduce the CR module for the cGANs, called the conditional cascading rejection (cCR) module. Among the various cGANs frameworks [18, 23, 25, 21, 22, 20], we design the cCR module based on the conditional projection discriminator [20] which shows superior performance than other existing discriminators for the cGANs. As depicted in Fig. 4, the conditional projection discriminator takes an inner product between the embedded condition vector \( w_c \), which is a different vector depending on the given condition, and the feature vector of the discriminator, i.e., \( v \), so as to impose a regularity condition. Based on the regularity condition, the standard discriminator, the conditional projection discriminator predicts the conditional probability \( P_c(v, w, w_c) \) as follows:

\[
P_c(v, w, w_c) = v^T w + v^T w_c = v^T (w + w_c).
\]

Based on the above equation, we design the cCR module by replacing the \( w_i \) in the CR module with \( (w_i + w_{c,i}) \), as shown in Fig. 5. More specifically, the \( i \)-th vector rejection process of the cCR module can be expressed as follows:

\[
v_{i+1} = v_i - \frac{v_i^T (w_i + w_{c,i})}{(w_i + w_{c,i})^T (w_i + w_{c,i})} (w_i + w_{c,i}).
\]

Unlike the CR module for the GANs, the cCR module conducts the vector rejection by considering the conditional information. After predicting \( N \) probabilities via the cCR module, the adversarial loss can be easily computed by using Eqs. 5 and 6.

4. Experiments

4.1. Implementation details

In order to evaluate the effectiveness of the CR module, we conducted extensive experiments using the CIFAR-10 [28], LSUN [31], Celeb-HQ [13, 16], and tiny-ImageNet [5, 30] datasets. The CIFAR-10 [28] and LSUN [31] datasets consist of 10 classes, whereas the tiny-ImageNet [5, 30], which is a subset of the ImageNet [5], is composed of 200 classes. Among a large number of images in the LSUN dataset, we randomly selected 30,000 images for each class. In addition, we compressed the images from the Celeb-HQ, LSUN, and tiny-ImageNet datasets as 64 \( \times \) 64 pixels. For the objective function, we adopted the hinge version of adversarial loss. The hinge version loss with the CR module is defined as follow:

\[
L_D = E\left[ \frac{1}{N} \sum_{i=1}^{N} \lambda_i \max(0, 1 - p_{x,i}) \right] + E\left[ \frac{1}{N} \sum_{i=1}^{N} \lambda_i \max(0, 1 + p_{z,i}) \right],
\]

\[
L_G = -E\left[ \frac{1}{N} \sum_{i=1}^{N} \lambda_i p_{z,i} \right].
\]
Table 1. Detailed architectures of the generator and discriminator used for the $32 \times 32$ CIFAR-10 dataset. In our experiments, we follow the implementation of the Miyato et al. [20] (a) Architecture of the generator, (b) Architecture of the discriminator.

Table 2. Detailed architectures of the generator and discriminator used for the $64 \times 64$ Celeb-HQ, LSUN, and tiny-ImageNet datasets. In our experiments, we follow the implementation of the Miyato et al. [20] (a) Architecture of the generator, (b) Architecture of the discriminator.

Since all parameters in the generator and the discriminator including the CR module can be differentiated, we performed an optimization using the Adam optimizer [14], which is a stochastic optimization method with adaptive estimation of moments. We set the parameters of Adam optimizer, i.e., $\beta_1$ and $\beta_2$, to 0 and 0.9, respectively, and set the learning rate to 0.0002. During training procedure, we updated the discriminator five times per each update of the generator. For the CIFAR-10 dataset, we used a batch size of 64 and trained the generator for 100k iterations, whereas the generators for the Celeb-HQ and LSUN datasets were trained for 100k iterations with a batch size of 32. For the tiny-ImageNet, we set a batch size as 64 and trained the generator for 400k iterations (about 100 epochs). Our experiments were conducted on CPU Intel(R) Xeon(R) CPU E3-1245 v5 and GPU RTX 2080 Ti, and implemented in TensorFlow.

4.2. Baseline models

In this work, we employed the generator and discriminator architectures of the leading cGANs scheme [20][19], as our baseline models. The detailed architectures of the two different models for $32 \times 32$ and $64 \times 64$ resolutions are presented in Tables 1 and 2, respectively. Both of cases employ the multiple residual block [9] (ResBlocks) as depicted in Fig. 6. In the ResBlock of the discriminator, we employed the spectral normalization [19] for all layers including the proposed CR module. For the discriminator, the down-sampling (average-pooling) is performed after the second convolutional layer, whereas the generator up-sampled the feature maps using a nearest neighbor interpolation prior to the first convolution layer.

4.3. Comparison of Sample Quality

Evaluation metrics To evaluate the performance of the generator, in this paper, we employed the principled and comprehensive metric, called frechet inception distance (FID) [10], which measures the visual appearance and diversity of the generated images. The FID can be obtained by calculating the Wasserstein-2 distance between the distribution of the real images, $P_R$, and that of the generated ones, $P_G$, in the feature space obtained via the Inception model [27], which is defined as follows:

$$FID(p, q) = \|\mu_p - \mu_q\|^2 \quad + \text{trace}(C_p + C_q - 2(C_pC_q)^{1/2}),$$

(12)

where $\{\mu_p, C_p\}, \{\mu_q, C_q\}$ are the mean and covariance of the samples with distribution $P_R$ and $P_G$, respectively. Lower FID scores indicate better quality of the generated images. Indeed, there is an alternative approximate measure of image quality, called inception score (IS). However,
since the IS has some flaws as mentioned in \cite{2, 3}, we employed the FID as the main metric in this paper.

**Results** To demonstrate the advantage of the CR module, we conducted extensive experiments by adjusting the \( N \) value. In our experiments, we randomly generated 50,000 images for CIFAR-10, LSUN, and tiny-ImageNet datasets and 30,000 images for Celeb-HQ dataset. Table 3 shows the comprehensive performances of the proposed method. The bold numbers in Table 3 represent the best performance among the results. As shown in Table 3, the CR module significantly improves the performance of GANs. In addition, Fig. 7 shows the FID results on tiny-ImageNet dataset during the training procedure. As shown in Fig. 8, the proposed method shows better performance than the standard GANs during the training procedure. These results reveal that the proposed method is able to effectively improve the GANs performance by considering the ignored features in the discriminator. Thus, we confirmed that, by strongly penalizing the generator using the CR module, the proposed method leads the generator to produce the images that are more similar to the real images and results in the low FID scores.

Moreover, to demonstrate the validity of the cCR module, we conducted additional cGAN experiments on CIFAR-10, LSUN, and tiny-ImageNet datasets, except for the Celeb-HQ dataset which does not contain the conditional information, i.e., class information. We employed the same baseline models with the experiments of GANs, but replaced the BN in the generator with the conditional batch normalization layer \cite{6}. Also, the CR module in the discriminator is replaced with cCR module. As shown in Table 3, the proposed method shows better performance than cGANs \cite{20}, which reveals the performance of cGANs can be improved by applying the cCR module to the cGANs framework. We notice that the cCR module with \( N = 4 \) shows worse performance than that with \( N = 2 \) on datasets having a small number of classes such as CIFAR-10 and LSUN, whereas shows better performance on tiny-ImageNet dataset consisting of 200 classes. These results indicate that \( N = 2 \) is enough to penalize the generator for training the datasets having a small number of classes. In

| Dataset       | GANs \((N = 1)\) | Proposed method \((N = 2)\) | Proposed method \((N = 4)\) | cGANs \((N = 1)\) | Proposed method \((N = 2)\) | Proposed method \((N = 4)\) |
|---------------|-----------------|-----------------------------|-----------------------------|-----------------|-----------------------------|-----------------------------|
| CIFAR-10 \cite{28} | 18.26           | 16.90                       | **16.46**                   | 11.34           | **10.87**                   | 10.92                       |
| Celeb-HQ \cite{13, 16} | 11.81           | 11.11                       | **8.49**                    | -               | -                           | -                           |
| LSUN \cite{31}     | 26.86           | 24.37                       | **21.48**                   | 22.69           | **19.09**                   | 19.96                       |
| tiny-ImageNet \cite{5, 30} | 39.12           | 38.37                       | **36.57**                   | 32.15           | 31.75                       | **29.69**                   |

Table 3. Comparison of the proposed method with the standard GANs and cGANs on the CIFAR-10, Celeb-HQ, LSUN, and tiny-ImageNet datasets in terms of the FID.
other words, to achieve the fine performance, the $N$ of the cCR module should be adjusted depending on the number of the given condition.

Fig. 9 shows the examples images on Celeb-HQ, LSUN and tiny-ImageNet datasets. As depicted in Fig. 9, the proposed method allows the generator to produce visually pleasing images. In addition, the proposed method additionally requires the $(N - 1) \times C_L$ network parameters, where $C_L$ indicates the dimension of the last layer in the discriminator, which are very small numbers compared to the overall discriminator parameters. Thus, the proposed CR module can be added to the discriminator with a marginal training overhead. It is worth noting that this work does not intend to design an optimal generator and discriminator architectures for the CR and cCR modules; there could be another structure that leads to better performance and generates more high quality images. In contrast, we care more about whether it is possible to improve the performance of the GANs and cGANs frameworks by simply adding the CR and cCR module to the discriminator, respectively.

4.4. Image-to-image translation with CR module

To demonstrate the generalization ability of the proposed method, we applied the CR module to the image-to-image translation scheme. In this paper, we selected the CycleGAN [36], one of the state-of-the-art frameworks to con-
duct image-to-image translation with the unpaired training data, as the baseline framework. The detailed architectures are described in Table 4. Note that we did not conduct global pooling on the last layer of the discriminator. Instead, in the CR module, we predicted $N$ probabilities for each pixel using the $1 \times 1$ convolutional layer which is equivalent to the fully connected layer. In addition, in the ResBlock of the generator, we employed the instance normalization [29] (IN) instead of the BN.

In our experiments, we evaluated the performance on the photo ↔ Monet dataset in [36]. We compressed the images from the photo ↔ Monet dataset as 128 × 128 pixels, and trained the generator for 100k iterations using the hinge version of adversarial loss in Eqs. 10 and 11. Experimental results are described in Fig. 10 and Table 5. As shown in Table 5, the proposed method significantly improves the FID score, which represents that the proposed CR module can be applied to the discriminator having several pixels in the last layer such as PatchGAN [12]. Thus, we confirmed that the proposed CR module can be easily utilized for improving the performance of other GAN-based applications.

5. Conclusion

In this paper, we have introduced a straightforward method for improving the performance of the GANs. By using the non-overlapped features obtained via the proposed CR module, the discriminator effectively penalizes the generator during the training procedure, which results in improving the performance of the generator. One of the main advantages of the CR module is that it can be readily integrated with the existing discriminator architectures. Moreover, our experiments reveal that, without imposing the training overhead, the discriminator with the CR module significantly improves the performance of the baseline models. In addition, the generalization ability of the proposed method is demonstrated by applying the CR module to the cGANs and image-to-image translation frameworks. It is expected that the proposed method will be applicable to various applications based on the GANs.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. arXiv preprint arXiv:1701.07875, 2017.
[2] Shane Barratt and Rishi Sharma. A note on the inception score. arXiv preprint arXiv:1801.01973, 2018.
[3] Ting Chen, Xiaohua Zhai, Marvin Ritter, Mario Lucic, and Neil Houlsby. Self-supervised gans via auxiliary rotation loss. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 12154–12163, 2019.
[4] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8789–8797, 2018.
[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee, 2009.
[6] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. Proc. of ICLR, 2, 2017.
[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.
[8] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In Advances in neural information processing systems, pages 5767–5777, 2017.
[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.
[10] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Advances in Neural Information Processing Systems, pages 6626–6637, 2017.
[11] Seunghoon Hong, Dingdong Yang, Jongwook Choi, and Honglak Lee. Inferring semantic layout for hierarchical text-image synthesis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 7986–7994, 2018.
[12] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1125–1134, 2017.
[13] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196, 2017.
[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012.
[16] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In Proceedings of the IEEE International Conference on Computer Vision, pages 3730–3738, 2015.
[17] Qi Mao, Hsin-Ying Lee, Hung-Yu Tseng, Siwei Ma, and Ming-Hsuan Yang. Mode seeking generative adversarial networks for diverse image synthesis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1429–1437, 2019.
[18] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784, 2014.
[19] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. arXiv preprint arXiv:1802.05957, 2018.
[20] Takeru Miyato and Masanori Koyama. Gans with projection discriminator. arXiv preprint arXiv:1802.05637, 2018.
[21] Augustus Odena. Semi-supervised learning with generative adversarial networks. arXiv preprint arXiv:1606.01583, 2016.
[22] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In Proceedings of the 34th International Conference on Machine Learning-Volume 70, pages 2642–2651. JMLR. org, 2017.
[23] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. arXiv preprint arXiv:1605.05396, 2016.
[24] Min-cheol Sagong, Yong-goo Shin, Seung-wook Kim, Seung Park, and Sung-jea Ko. Pepsi: Fast image inpainting with parallel decoding network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 11360–11368, 2019.
[25] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In Advances in neural information processing systems, pages 2234–2242, 2016.
[26] Yong-Goo Shin, Min-Cheol Sagong, Yoon-Jae Yeo, Seung-Wook Kim, and Sung-Jea Ko. Pepsi++: Fast and lightweight network for image inpainting. arXiv preprint arXiv:1905.09010, 2019.
[27] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2818–2826, 2016.
[28] Antonio Torralba, Rob Fergus, and William T Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE transactions on pattern analysis and machine intelligence, 30(11):1958–1970, 2008.
[29] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022, 2016.
[30] Leon Yao and John Miller. Tiny imagenet classification with convolutional neural networks. CS 231N, 2015.
[31] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. arXiv preprint arXiv:1506.03365, 2015.
[32] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. arXiv preprint arXiv:1806.03589, 2018.
[33] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. arXiv preprint arXiv:1805.08318, 2018.
[34] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. IEEE transactions on pattern analysis and machine intelligence, 41(8):1947–1962, 2018.
[35] Jun-Yan Zhu,taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision, pages 2223–2232, 2017.