AcGAN: Acg avatar Generation network

Well-Promising [1], Vvvvvv [2], Gaozzzz [3], Frrrrl [4], Tracy [5]

[1] dengyang-31520211154032
[2] fengwenjun-31520211154039
[3] gaojianzhe-31520211154040
[4] hesaichao-31520211154047
[5] wuyaoyuan-31520211154093
31520211154032@stu.xmu.edu.cn, 31520211154039@stu.xmu.edu.cn, 31520211154040@stu.xmu.edu.cn,
31520211154047@stu.xmu.edu.cn, 31520211154093@stu.xmu.edu.cn

Abstract

Recently, the technological breakthrough in Generative Adversarial Networks (GAN) are driving advances in AI-based content generation applications such as painting, style conversion, and music composition. However, few teams are targeting the ACGN group. The preference for animated and comic character designs has always accurately reflected each person's ACGN concentration. In this study, we present AcGAN — Acg avatar Generation network, a simple yet powerful generative adversarial architecture that can generate the preferred Acg wife directly. In this work, the picture generation function of GAN network is applied to the task of ACG character generation, which uses a variety of GAN networks and compares the advantages and disadvantages of various networks horizontally. At the end of the paper, we point out the areas and prospects for the future development of the GAN.

Introduction

Generative modeling is an unsupervised learning task in machine learning that involves automatically discovering and learning the regularities or patterns in input data so that models can be used to generate or output new examples that might be extracted from the original dataset. Generative adversarial network (GAN) (Goodfellow et al. 2014) is an approach to generative modeling using deep learning methods.

GAN is committed to generate data that did not exist in the real world, similar to making AI creative or imaginative and it has shown great potential to generate AI-based samples since its first introduction in 2014. And GAN's core ideas derive from the Nash equilibrium in game theory, with Generator (the generator G responsible for generating the image) and Discriminator (the judge D responsible for judging the authenticity of the picture). D's aim is to get a low score on a "fake" image that G "forges" and G's aim is to "spoof" D into getting a high score by imitating the real image to the maximum extent possible. Participating parties in this game continue to complete their own optimization in the competition, thus achieving their respective identification and generation capabilities, until the two sides reach a dynamic balance.

Many people in the world like ACGN, and many heavy two dimensions want to have a Acg wife. Based on this we propose AcGAN — Acg avatar Generation network. The network is able to generate the user's preferred Acg wife based on the desired hair length, hair color, and pupil color entered by the user.

We can think that all the painters in the painting of the Acg girls are subject to a fixed regulation, such as the position of the eyes, hair direction, according to some people feel comfortable proportions to draw a beautiful image. From the perspective of neural networks, however, the picture is achieved by means of a pixel matrix, that is, all the beauty of the girl is subject to a complex distribution. We take samples from this distribution to get the positive samples that we hope them with positive distribution of the five officials. And outside of this distribution, we can get a variety of negative images. Therefore, the generator G has to do is find out the distribution and simulate it so that the image generated by random noise after passing through the generator can satisfy this distribution. On the other side, the discriminator D is responsible for determining whether the distribution generated is positive.

During training, the goal of generating Network G is to generate as many real pictures as possible to spoof Network D, and the goal of D is to separate the pictures generated by G from the real pictures. Thus, G and D constitute a dynamic "game process". The result of the final game is that, ideally, G can generate enough picture G(z) to "fake the truth". For D, it is difficult to determine whether the picture generated by G is real or not, and in this state D(G(z)) is 0.5.

The whole formula consists of two. In the case of generating a picture, x represents a real picture, z represents the noise of the input G network, and G(z) represents a picture generated by the G network. D(x) represents the probability that the D network will judge whether the real picture is true (because x is real, the closer this number is to 1, the better for D). D(G(z)) is the probability that the D network will determine whether the picture generated by G is real. The purpose of G is to make D (G(z) as large as possible, at which point V(D, G) becomes smaller. So we want to get the minimum value of G, which is minG. The stronger D's ability, the larger the D(x) and the smaller the D(G(x). V(D, G) becomes larger, so the formula is the largest (i.e. maxD) for D.

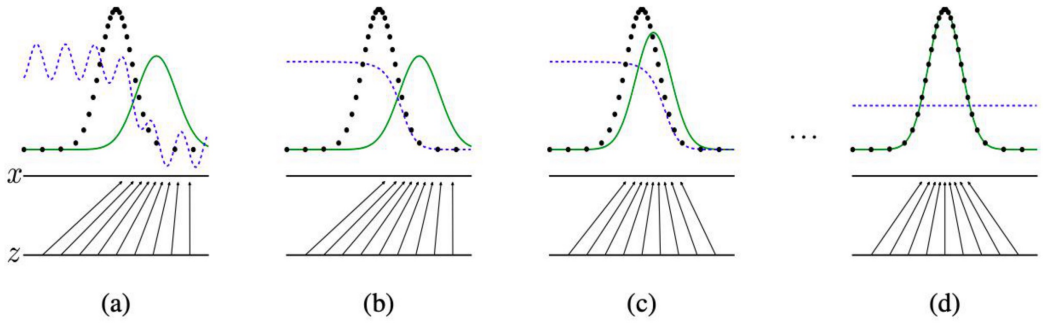

Figure 1: The GAN training process. The black dashed line in the figure represents the distribution of the real sample, the blue dashed line represents the distribution of the probability of the differential discrimination, and the solid green line represents the distribution of the resulting sample. Z represents noise, and Z to x represents the mapping of the distribution after passing through the generator.

From the figure 2, our goal is to use the generated sample distribution (green solid line) to fit the real sample distribution (black dashed line) to produce a false sample. You can see that when (a) is in its initial state, the distribution generated by the generator differs greatly from the real distribution, and the probability of the dissiferr distinguishing the sample is not very stable, so the dissenter is trained to better distinguish the sample. (b) sample status is achieved by training the discriminator several times, at which point the distinguishing sample is very significant and well distinguished. The generator is then trained. Train the generator to reach the (c) sample state, where the generator distribution is approximated compared to the previous real sample distribution. After many iterations of repeated training, it is hoped that the (d) state can be reached, that the resulting sample distribution fits the real sample distribution, and that the D cannot tell whether the sample is generated or true (the probability of identification is 0.5). This means that a very real sample can be generated at this point.

Related work

Generative adversarial networks is only able to synthesize low-resolution images, the follow-ups improved it with multiple discriminators (Durugkar, Gemp, and Mahadevan 2016; Zhao et al. 2020; Theagarajan and Bhanu 2019), self-attention mechanism (Brock, Donahue, and Simonyan 2018; Zhang et al. 2019; Emami et al. 2020) and progressive training strategy (Karras et al. 2017).So, now GANs have enabled photorealistic synthesis for many vision and graphics applications (Ledig et al. 2017; Isola et al. 2017; Pathak et al. 2016; Zhao et al. 2020). As image quality and compute capacity have increased, we have computational costs and inference time.

Image generation via GANs. There are two major ways of using GANs for image generation: (1) The DCGAN (Radford, Metz, and Chintala 2015) was trained so that it was able to generate high-quality secondary images. A large number of samples are then produced, which we manually label to turn the problem into a classification problem. Put these labeled samples into a CNN-based classification network for classification; (2) conditional GANs (Isola et al. 2017; Liu, Breuel, and Kautz 2017; Lee et al. 2018; Zhu et al. 2017), which learn to directly translate an input image into a target domain. In this work, we aim to learn efficient generators for getting high-quality generated pictures, which can control the progress of the project accurately. In this work, we tried three models as the primary network, namely SAGAN, BigGAN and StyleGAN.

DCGAN. Image recognition has always been a hot research topic in the scientific community and industry(Fang et al. 2018). The advent of convolutional neural networks (CNN) has made this technology a focus of research in the field of computer vision, especially in the field of image recognition. But it makes the identification results largely dependent on the quantity and quality of the training samples. Recently, DCGAN has become a cutting-edge method for generating images, sounds, and videos. DCGAN's evaluators and generators use convolutional neural networks (CNN) to replace multi-layer sensors in GAN, while removing the pooled layer from CNN in order to make the entire network micro, and replacing the full-connected layer with a global pooled layer to reduce computation.

Proposed Solution

SAGAN

Self-Attention Generative Adversarial Networks. Convolutional GAN can effectively handle textures which in pictures in general, but underperform when dealing with the structural characteristics of the whole (e.g. the texture of the dogs' hair and the structure of the limbs), often resulting the part of eyes asymmetric when creating a human face. The reason for it should be that the local perception domain of the convolution kernel is small, which cannot see the effect of the right eye on the left eye when convolution is done in the area of the left eye, thus resulting in a picture that lacks the integrity of the structural features of the face. Although high-dimensional abstraction and global grip can be achieved through multi-layer convolution, how to train

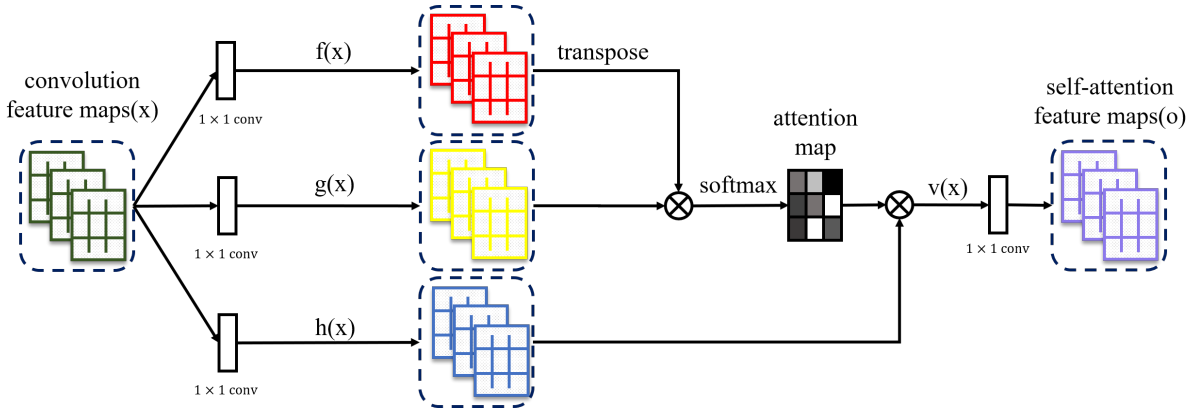

Figure 2: SAGAN network structure

such a multi-layer network becomes a challenge, and if a large convolution kernel is used, it will reduce the computational and statistical advantage of convolution kernel relative to fully connected neurons. As a result, SAGAN introduces self-attention mechanism into convolutional GAN to balance the overall structural information and computational and statistical efficiency better.

Self-attention (Mi et al. 2020; Parmar et al. 2018). In order to achieve self-attention mechanism, the attention layer is added to the network. Suppose the characteristics of the image are output from the previous hidden layer, where C is the number of channels, N is the number of features, and the image size equals wide $\times$ high.

$$f(x) = W_f X, g(x) = W_g X, s_{ij} = f\left(x_i\right)^T g\left(x_i\right) \quad (1)$$

$$\beta_{j,i} = \frac{e^{s_{ij}}}{\sum_{i=1}^{N} e^{s_{ij}}} \quad (2)$$

Indicates how much attention the model pays to position i when it generates position j, at which point a connection is established between different locations. In order to increase the descriptive power of the attention layer, the output of the attention layer introduces more parameters:

$$o_j = v\left(\sum_{i=1}^{N} \beta_{j,i} h\left(x_i\right)\right), h\left(x_i\right) = W_h x_i, v\left(x_i\right) = W_v x_i \quad (3)$$

Among them, $W_g \in R^{\bar{C} \times C}$, $W_f \in R^{\bar{C} \times C}$, $W_h \in R^{\bar{C} \times C}$ and $W_v \in R^{C \times \bar{C}}$, are achieved by a 1×1 convolutional kernel. $\bar{C}$ can be self-made, you can specify a number less than C to reduce the amount of calculation. The final output o is added to the image feature by a scale factor γ to achieve attention-to-image adjustment:

$$y_i = \gamma o_i + x_i \quad (4)$$

SAGAN uses two techniques to make training more stable. First, SAGAN uses spectral normalization to implement Lippschitz constraints at every layer of generators and differentiators compared to previous work. The advantage of spectral normalization is that it does not rely on additional hypersorginant fine-tuning (spectrum normalization works well when all layers are set to 1) and the computational overhead is small. And using spectral normalization on both the generator and the evaluator also makes it possible to have a fewer updates to the evaluator after the update of each generator, reducing computational overhead.

Because regularization sometimes slows down the learning curve of the GAN, in practice, the method of using regularization judgment requires multiple update steps (for example, five) per generator during training typically. Independently, Heusel advocates the use of separating learning rates (TTRs) for generators and raters. We recommend that TTUR be used specifically to compensate for slow learning problems in regularized disparaters, making it possible to use a fewer generator steps for each disparate step. With this approach, we can produce better results in the same unit time.

The above is the self-attention mechanism for solving the problem of getting global information proposed by SAGAN, which not only considers the global information at each layer, but also does not introduce too many parameters, finding a good balance between improving the perception of wild and reducing the amount of parameters.

BigGAN

BigGAN(Chang and Lu 2020)is implemented on the infrastructure of SAGAN, focusing primarily on training large-scale GNs, with considerable improvements that preserves the ability of recognition even for poorly trained images, but with up to 1.6 billion network parameters, it also brings significantly increased computing overhead. The improvements of BigGAN are mainly reflected in three aspects: model structure, timely truncation and processing of the prior distribution z, and the control of model stability.

The model structure. Simply increasing the batch-size can improve the performance. The reason may be that the more patterns are overwritten per batch the better gradient will be provided for generating and identifying the two networks by increasing the batch-size. However, it also leads to a decrease in the stability of the model in training. At the same time, based on the results of the experiment, the

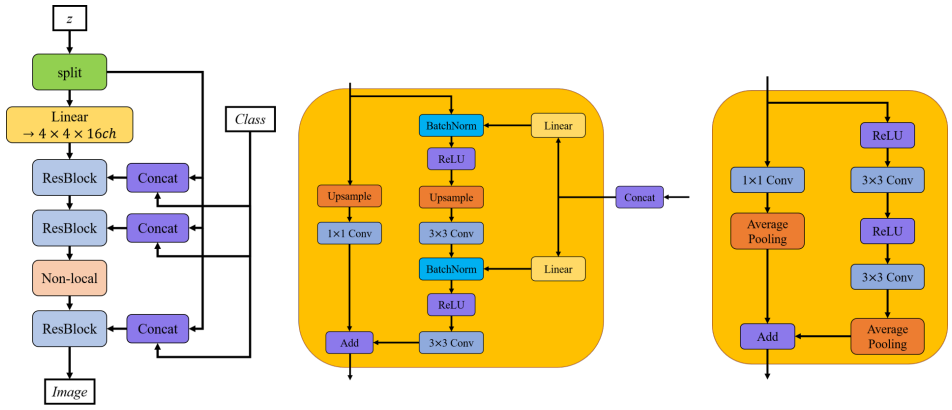

Figure 3: BigGAN network structure

number of channels per layer should also be increased. The increase of parameters with the increase of channels makes the capacity of the model more complex than the dataset. In general, GAN embeds z as input into the built network(Bao et al. 2017; Lu et al. 2018), while BigGAN connects the noise vector z and bar label C to the various layers of the generator, affecting the characteristics of different resolution and hierarchy levels by affecting the weights and offsets of each layer.

Timely truncation and treatment of a priori distribution z. GAN is known to assume a prior distribution z as the excitation input to the network. Depending on the quality metrics IS and FID, you can set the appropriate threshold to truncate the sample of z, and the out-of-range values can be resampled(Qian et al. 2018). As the threshold decreases, the image quality gets better and better, but as the sampling range narrows, the resulting images become monolithic, resulting in insufficient diversity of the resulting images. IS generally reflects the quality, and FID reflects the diversity, therefore there are trade-offs in threshold decisions. In addition, some large models produce saturated artifacts when the truncation noise z is embedded, so orthogonalization is introduced, enforcing orthogonal conditions to smooth the generator:

$$R_\beta(W) = \beta \left\| W^T W - I \right\|_F^2 \tag{5}$$

W is the weight which requires proper relaxation, because the rule is often too strict. The best constraints by experimentation are:

$$R_\beta(W) = \beta \left\| W^\top W \odot (1 - I) \right\|_{\mathrm{F}}^2 \tag{6}$$

The control of model stability(Hou and Xiong 2019; Berberich et al. 2020). For the generator, the experiment attempted to avoid crashes by adjusting the first three singular values of each weight matrix, but the results were not ideal. Therefore, the steering is to control the dissector. Experiments have found that the spectra of the disspirator contain noise and assume that it is related to training crashes. And by using R1 zero center gradient penalty:

$$R_1 : \frac{\gamma}{2} E_{p_D(x)} \left[\|\nabla D(x)\|_F^2 \right] \tag{7}$$

And we found that the higher penalty for the dissitortor, the training will be more stable, but the performance will be worse.

StyleGAN

Style-Based Generator Architecture for Generative Adversarial Network(Karras et al. 2020; Nie et al. 2020). StyleGAN can be said to draw on some of the ideas of VEA(Variational Autoencoder)(Pu et al. 2016), whose main contribution is to provide a framework that can be involved in the image generation process of the generator, and to add details to the picture in a more efficient way, making it more efficient for the generator to generate pictures with specific styles with more details.

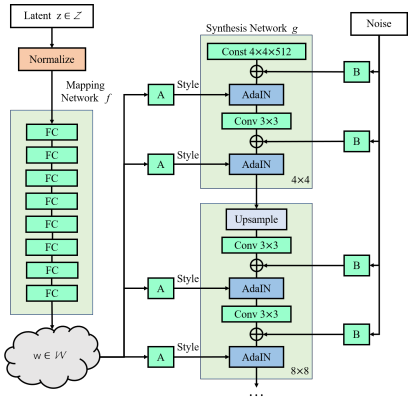

Figure 4: Style-based generator contrasts with traditional structures

Style-based generator(Karras, Laine, and Aila 2019).. StyleGAN replaces the initial input layer with a constant layer in the generator and adds a fully connected mapping

network to generate inputs with a set of style control vectors w:

$$f : z \rightarrow w \tag{8}$$

The generation of certain features is controlled separately by each element of w to reduce feature entanglement. The resulting w transforms A into style with a learnable affine after truncation:

$$y = (y_s, y_b) \tag{9}$$

Thus, at each level of AdaIN (Adaptive Instance Normalization):

$$AdaIN\left(x_i, y\right) = y_{s,i}\frac{x_i - \mu\left(x_i\right)}{\sigma\left(x_i\right)} + y_{b,i} \tag{10}$$

The result generated at each layer is machined so that the resulting picture is globally controlled by style y. In addition, layer-by-layer normalization allows the operation to affect only a portion of the final result.

Style mixing(Karras et al. 2020; Mikołajczyk and Grochowski 2018; Shorten and Khoshgoftaar 2019). To localize the style further, mixing regularization controls different high-level properties by randomly switching different styles of w at some layers during the build process after two ws are calculated nonlinearly. This approach can transition from one style to another in a coherent way.

Stochastic variation. Traditional generator inputs have only the input layer, so generator itself must look for a way to generate pseudo-random numbers to produce random transformations, but it consumes the ability of network and makes hiding the periodicity of the generated signals difficult. Therefore, the image can control local characteristics randomly and directly by introducing Gaussian noise B. The method makes it possible to enrich the details of the image, thus making it more realistic and diverse.

Experiment

Experimental data and loss. We used 34,093 original secondary images with a resolution of 128×128 as our dataset. Due to length issues, we only select the training results of SAGAN, which has relatively good training results, for presentation.

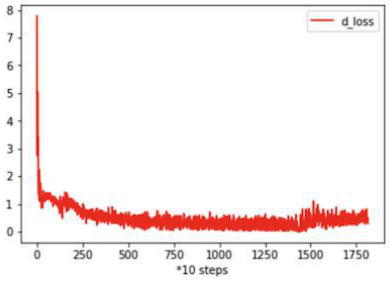

Figure 5: The test error of the discriminator.

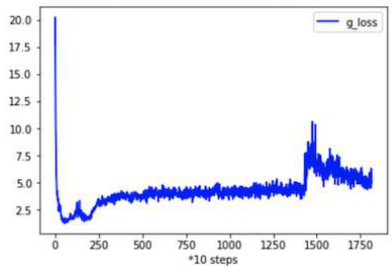

Figure 6: The test error of the generator.

Conclusion

In order to generate ACG images, we tried to build AcGAN using three GAN structures in this experiment. Through this experiment, our team members learned about the development of GAN and their variants, forming their own knowledge network in the field of image generation, but also have their own views. Although everyone's research direction is different, the idea of confrontation and rivalry can be transferred to various fields, which opens up scientific research and innovative thinking for us. We are also happy to apply what we have learned to the ACG areas of interest to us. At the same time, it boldly predicted that most of the work in the field of comics (two-dimensional) in the future will be replaced by artificial intelligence. We believe that the innovative applications of artificial intelligence in the field of comics include: (1) Intelligent coloring; (2) Generating images frame by frame according to the text description of the script to form animation; (3) Generating animation and movies directly from novels in the film and television industry.

Figure 7: Some of the Acg images generated.

References

Bao, J.; Chen, D.; Wen, F.; Li, H.; and Hua, G. 2017. CVAE-GAN: fine-grained image generation through asymmetric training. In *Proceedings of the IEEE international conference on computer vision*, 2745–2754.

Berberich, J.; Köhler, J.; Müller, M. A.; and Allgöwer, F. 2020. Data-driven model predictive control with stability and robustness guarantees. *IEEE Transactions on Automatic Control*, 66(4): 1702–1717.

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

Chang, T.-Y.; and Lu, C.-J. 2020. TinyGAN: Distilling BigGAN for Conditional Image Generation. In *Proceedings of the Asian Conference on Computer Vision*.

Durugkar, I.; Gemp, I.; and Mahadevan, S. 2016. Generative multi-adversarial networks. *arXiv preprint arXiv:1611.01673*.

Emami, H.; Aliabadi, M. M.; Dong, M.; and Chinnam, R. B. 2020. Spa-gan: Spatial attention gan for image-to-image translation. *IEEE Transactions on Multimedia*, 23: 391–401.

Fang, W.; Zhang, F.; Sheng, V. S.; and Ding, Y. 2018. A method for improving CNN-based image recognition using DCGAN. *Computers, Materials and Continua*, 57(1): 167–178.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.

Hou, Z.; and Xiong, S. 2019. On model-free adaptive control and its stability analysis. *IEEE Transactions on Automatic Control*, 64(11): 4555–4569.

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

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

Karras, T.; Laine, S.; and Aila, T. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4401–4410.

Karras, T.; Laine, S.; Aittala, M.; Hellsten, J.; Lehtinen, J.; and Aila, T. 2020. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8110–8119.

Ledig, C.; Theis, L.; Huszár, F.; Caballero, J.; Cunningham, A.; Acosta, A.; Aitken, A.; Tejani, A.; Totz, J.; Wang, Z.; et al. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4681–4690.

Lee, H.-Y.; Tseng, H.-Y.; Huang, J.-B.; Singh, M.; and Yang, M.-H. 2018. Diverse image-to-image translation via disentangled representations. In *Proceedings of the European conference on computer vision (ECCV)*, 35–51.

Liu, M.-Y.; Breuel, T.; and Kautz, J. 2017. Unsupervised image-to-image translation networks. In *Advances in neural information processing systems*, 700–708.

Lu, Y.; Wu, S.; Tai, Y.-W.; and Tang, C.-K. 2018. Image generation from sketch constraint using contextual gan. In *Proceedings of the European conference on computer vision (ECCV)*, 205–220.

Mi, Z.; Jiang, X.; Sun, T.; and Xu, K. 2020. GAN-Generated Image Detection With Self-Attention Mechanism Against GAN Generator Defect. *IEEE Journal of Selected Topics in Signal Processing*, 14(5): 969–981.

Mikołajczyk, A.; and Grochowski, M. 2018. Data augmentation for improving deep learning in image classification problem. In *2018 international interdisciplinary PhD workshop (IIPhDW)*, 117–122. IEEE.

Nie, W.; Karras, T.; Garg, A.; Debnath, S.; Patney, A.; Patel, A.; and Anandkumar, A. 2020. Semi-supervised StyleGAN for disentanglement learning. In *International Conference on Machine Learning*, 7360–7369. PMLR.

Parmar, N.; Vaswani, A.; Uszkoreit, J.; Kaiser, L.; Shazeer, N.; Ku, A.; and Tran, D. 2018. Image transformer. In *International Conference on Machine Learning*, 4055–4064. PMLR.

Pathak, D.; Krahenbuhl, P.; Donahue, J.; Darrell, T.; and Efros, A. A. 2016. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2536–2544.

Pu, Y.; Gan, Z.; Henao, R.; Yuan, X.; Li, C.; Stevens, A.; and Carin, L. 2016. Variational autoencoder for deep learning of images, labels and captions. *Advances in neural information processing systems*, 29: 2352–2360.

Qian, R.; Tan, R. T.; Yang, W.; Su, J.; and Liu, J. 2018. Attentive generative adversarial network for raindrop removal from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2482–2491.

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

Shorten, C.; and Khoshgoftaar, T. M. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1): 1–48.

Theagarajan, R.; and Bhanu, B. 2019. DeephESC 2.0: Deep generative multi adversarial networks for improving the classification of hESC. *PloS one*, 14(3): e0212849.

Zhang, H.; Goodfellow, I.; Metaxas, D.; and Odena, A. 2019. Self-attention generative adversarial networks. In *International conference on machine learning*, 7354–7363. PMLR.

Zhao, Z.; Zhang, Z.; Chen, T.; Singh, S.; and Zhang, H. 2020. Image augmentations for GAN training. *arXiv preprint arXiv:2006.02595*.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, 2223–2232.