Evaluating Performance of Pluralistic-Inpainting Proposed in Pluralistic Image Completion

Yunting Chiu
American University
4400 Massachusetts Ave NW, Washington, DC 20016
yc6705a@american.edu

Huong Doan
American University
4400 Massachusetts Ave NW, Washington, DC 20016
hd4275a@american.edu

Abstract

Basically, image inpainting is to recover the damaged image or fill out a missing part of an image. There are a lot of existing image inpainting methods which are published with or without codes. Those methods will have their own advantages and disadvantages. In addition, those methods usually work well on the datasets which were used for the experiments on the paper, but it does not guarantee that those methods can work well for other datasets as well. Several related readings are Texture Memory-Augmented Deep Patch-Based Image Inpainting [1], Image Fine-grained Inpainting [2] and Generative Adversarial Networks [3].

In our study, we choose an existing method which we are interested in and apply it to the interesting datasets we choose to observe its performance as well as its limitation if there is any. We choose the method proposed in the ***Pluralistic Image Completion*** *[4] paper to investigate the two datasets taken from Flickr-Faces-HQ Dataset (FFHQ) [5] and Kaggle Art Images [6]. The main challenging parts are to figure out how to run the GitHub code using our datasets and how to evaluate the performance of our experiments. For the experiments, we are planning to change the number of images in the train file to observe whether the results are different. However, our training images (65536) are greater than the one the authors used for their experiments (24183 for training and 2824 for testing). With the limited computer environment and resources, we cannot comprehensively finish the training process. Despite the fact that the training process is only at epoch 41(epoch is the number of passes of the entire training dataset), the output is satisfactory so far. As a result, one image will generate 50 different types of faces so we believe that the result can fool people. The limitation, we think it could be, is the running time and that is the more images in the training or testing set, the longer the running time of the code will be. The solutions should either decrease the number of input images or increase the training time. We will visualize the recovered images and the original images without missing parts. The expected results are recovered images, but the methodology of the paper does not guarantee that those images exactly look like the original images without missing parts. In order to evaluate the performance of the model proposed in the paper, we will use Structural Similarity Index (SSIM - the higher the better) and Peak Signal-to-Noise Ratio (PSNR - the higher the better. The SSIM measures the similarity between two given images, while PSNR is to compare the output images (the inpainting images) to the input images (the images with holes or masks) with the maximum possible power.*

1. Introduction

1.1. Language

The contents of this paper are written in English.

1.2. GANs application

GANs (Generative Adversarial Networks) [3] are widely used for various image inpainting tasks. Pluralistic Image Completion is one of the applications that is mostly used in GANs. Pluralistic Image Completion is a type of image inpainting that generates a diverse outcome to recover images by using machine learning. The issue with image inpainting is that most image completion methods only produce one result for each input. Zheng et al. (2019) propose Pluralistic model will produce a wide range of possible results with different structure, color, and texture from only one image. To be precise, a single image can generate different types of images.

1.3. PICNet

The paper's model main idea is to create two parallel paths from an input image. The first is a regenerative part, which

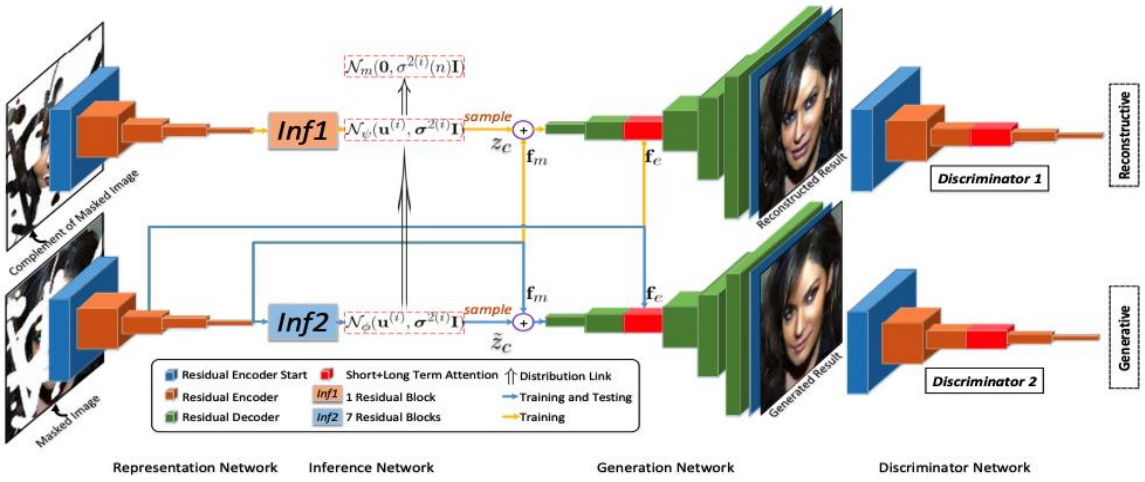


Figure 1: The original framework from Pluralistic Image Completion Paper

This neural network consists of two parallel pipelines. The yellow line (**reconstructive**) merges data from Im and Ic, which are only used for training purposes. The blue (**generative**) pipeline estimates the conditional distribution of hidden regions, which can then be sampled during testing, that is, Ig = {Ic, Im}.

- Ig is the original image, and Im is the masked image. The method is mapping Ig to Im
- Define Ic as the converse of Im, which is constructed from the masked image
- This paper final goal is to take sample from p(Ic|Im) to recover images

uses the only available ground truth to obtain a prior distribution of missing parts and reconstruct the original image from the distribution. The second is a generative part in which the conditional prior is linked to the reconstructive path's distribution. **PICNet** (Pluralistic Image Completion) is one of the applications that is mostly used in GANs. Pluralistic Image Completion is a type of image inpainting that generates a diverse outcome to recover images by using machine learning. The issue with image inpainting is that most image completion methods only produce one result for each input. The authors' proposed Pluralistic model will produce a wide range of possible results with different structure, color, and texture from only one image. To be precise, a single image can generate 49 different types of images. In this case, we are wondering how the result of our training model compares to the original one if we run another dataset and then run their model.

1.4. Our Purpose

In our study, we choose an existing method which we are interested in and apply it to the interesting datasets we choose to observe its performance as well as its limitation if there is any interesting output. To investigate the output of the paper's model with different datasets, we chose the method proposed in the Pluralistic Image Completion paper. Firstly, because this is our first-time using machine learning to apply image inpainting, we are very interested in comparing an inpainting image and a ground truth image, which is why we chose this topic for our computer vision final project. In the second place, because we are both Data Science major students, we are familiar with the basic statistical theories and regression backgrounds used to derive machine learning, but we do not fully understand the relationship between computer vision and machine learning. Based on this, we both believe this is an excellent opportunity to expand our knowledge of machine learning.

2. Related Work

2.1. T-MAD

There are mainly two approaches when it comes to image inpainting. One is a patch-based method, which uses information from within the input image, and the other is employing deep networks, which uses a large image dataset to recover a global environment. A novel approach utilizing both methods gives us new perspectives regarding image completion. Xu et. al (2020) [7] propose a new deep inpainting framework by unifying patch-based methods and deep networks to take advantage of both approaches. Global structure can be recovered using deep networks whereas details can be restored using the notion of patch matching. The proposed method is called Texture Memory-Augmented Deep Patch-Based Image Inpainting (T-MAD).

The method contains three modules. Firstly, a coarse and a roughly global estimation is generated using a

coarse network. The coarse network is trained with reconstruction loss producing a reasonable global structural prior. Then, they split the coarse result into non-overlapping patches of equal size. These patches will be used to guide the patch synthesis process, which distinguishes T-MAD from existing deep learning methodologies. Lastly, high-quality patches guided by coarse result patch and the selected texture prior are generated through synthesizing textures on each small patch. Their evaluation results are superior compared to popular approaches such as PatchMatch, GL, Edge or DeepFill given a smaller loss function.

2.2. SN-PatchGAN

Another simple, fast and stable approach called gated convolution proposed by Yu et al. (2019) gives feasibility in that they use this method, trained with pixel-wise loss and SN-PatchGAN [8] for free-form image inpainting. This framework is easy to implement. They simplify the training process by achieving only two goals, a pixel-wise reconstruction loss and an adversarial loss. The model also produces satisfactory results when the masks have arbitrary shapes, and the inputs are no longer simply RGB channels. As a result, this is the central concept of SN-PatchGAN.

The motivation for the gated convolutional network is that the authors find that vanilla convolutions are not appropriate for free-form image completion. They show in detail that the bias in convolution is ignored and conclude that vanilla convolutions are suitable for image classification and object detection instead of image inpainting. They also discuss how partial convolution improves the quality of image reconstruction on free-form masks but issues such as flaws in pixel validation and incompatibility with additional user inputs. With the intention to solve the above issues, gated convolution is employed to learn a dynamic feature selection mechanism for each channel and each spatial location. Their results show superiority over other methods such as PatchMatch, Global & Local and Context Attention.

3. Approach

3.1. PICNet's approach and datasets

PICNet (Pluralistic model) split an image into three parts: Ig, Ic, and Im. Ig is an original image, Ic is a complement partial image that contains the missing pixels from the original image, and Im is a masked partial image. In other words, if we combine the clear parts (not missing) of Im and Ic, they will return to their original status. Typically, the standard image completion method is mapping Ig to Im as model training, resulting in a single image. However, the authors' idea is taking a sample with Ic with a given Im for training model, this result will produce a variety of images. During the training, the authors use the visdom and dominate to visualize the image and loss plot. The authors refer to SA-GAN [9] as their inspiration to build the generator and discriminator networks. In this paper, the ablation study is the authors use their Pluralistic model (PICNet) to compare with different methods: C^{VAE} [10, 11, 12, 13], "Instance Blind" and BicycleGAN [14]. About CVAE, their output is similar to CAVE's output since the learned conditional prior is based on the highest latent likelihood solution. Compared to "instance blind", if visible pixels were used for reconstruction loss, the training could become poor. Also, the authors indicate there is a lack of output from BicycleGAN, causing a bad performance. As for the qualitative testing, the authors point out that quantitative analysis is difficult to test the output since one image generates several images, and the original image is no longer a benchmark against which to compare. However, they rank the top 10 samples from the discriminator, which is close to the ground truth. In this case, they have a quantitative comparison standard to do the L1 loss, peak signal-to-noise ratio (PSNR), total variation (TV), and Inception Score tests (IS) [15]. On their official website, the authors display four experiments: (1) For the face dataset, they train 24183 images from CelebA [16] and test 2824 images using the Growing GANs [17] algorithm to obtain the high-resolution CelebA-HQ dataset. (2) For the build dataset, they train 14900 images and test 100 images from Paris [18]. (3) For the natural scenery, they mention original training images from Places2 [19]. (4) They train and test images from ImageNet [20] for the object pictures. All pretrained models can be downloaded from their official website with images of resolution 256*256 and center holes 128x128, which have a high degree of diversity for the large amount of missing information.

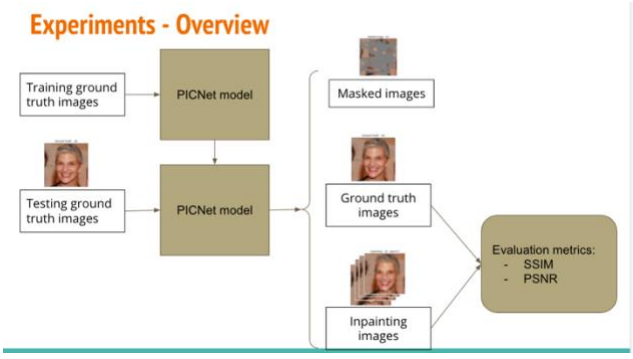


Figure 2: Our framework

3.2. Dataset for Yunting Chiu's Training

In this project, Huong and I apply an established approach that we are interested into the datasets we have chosen to

see how it performs, then we use statistical methods to compare the model outputs and the original images, as shown in figure 2. We choose Flickr-Faces-HQ Dataset [5] and Art Image from Kaggle [6]. Because when we refer to the Pluralistic Image Completion article [4], we detect the image inpainting is good especially for the face recovering. In this case, we choose face images from FFHQ for training and testing. FFHQ is a high quantity image dataset of people's faces that was originally created as a standard for GAN. This dataset contains 70,000 images with resolutions ranging from 128 x 128 to 1024 x 1024. We only took 128 x 128 resolution images for our project dataset from FFHQ due to a lack of GPU resources and a poor computer environment. According to the FFHQ official website, there are 70,000 images, the file size is 1.95GB, the format is PNG, and the path name is thumbnails 128x128. However, we recount the total image number from thumbnails 128x128 is 65,536. I will be responsible for testing and training thumbnails 128 x 128 with the PICNet algorithm.

3.3. Dataset for Huong Doan's Training

At the beginning, we selected Cifar-10 for our training dataset. Because the CIFAR-10 dataset is a well-known image dataset that is frequently used to train machine learning algorithms. However, the problem is that the dimensions of Cifar-10 images are too small, making them unsuitable for image inpainting, so we abandon Cifar-10. Then, we start to find an alternative dataset. Huong's idea is to select art images. Because in the Pluralistic Image Completion paper, the authors used most of the popular image's datasets (CelebA, Paris, Places2, and ImageNet) and they basically just did not use the art images so the art images will be interesting for us to do image inpainting. Houng picks 8 art images for her training dataset. Before running the model, she should downsample the 8 original images at a ratio of 4, because compressing the image sizes can reduce code running time.

3.4. Computing Environment

We both use Google Colab free version for training and testing datasets at the beginning. My computer is MacBook Pro (13-inch, 2020, Four Thunderbolt 3 ports), the processor is 2 GHz Quad-Core Intel Core i5 with 16 GB memory 3,733 MHz LPDDR4X. Huong uses Microsoft Windows to run the code. In my experience, when I start running code on Colab, I notice that one epoch requires three hours of training because the image number of my dataset is 65,536, which is greater than the one used by the authors for their experiments (24,183 for tr aining and 2,824 for testing on CelebA). Therefore, we consider upgrading to Colab Pro in order to get faster GPUs, longer runtimes, and more memory.

3.5. 128x128 thumbnails training results (FFHQ)

a. When we run the original code in Colab, we discover an error. The error line is: `self.img = self.img.cuda(self.gpu_ids[0], async=True)`, and the error message is: *`SyntaxError: invalid syntax => async=True`*. Our solution is change from *`async=True`* to *`non_blocking=True`* in line 68, 69 form the *Pluralistic/model/pluralistic_model.py* file
b. The authors provide data visualization of training results and loss plots on their code notes. I send an email to one of the authors, Dr. Zheng, he mentions that they use the visdom and dominate to visualize the image and loss plot. However, because our computing environment is Google Colab, we are unable to run the code that they provided. In the following Experimental Results section, we will compare the outputs and the ground truth using other statistical methods.
c. Our training images (65536) are greater than the one the authors used for their experiments (24183 for training and 2824 for testing). With the limited computer environment and resources, we cannot comprehensively finish the training process. Despite the fact that the maximum value of the training process is only at epoch 41(epoch is the number of passes of the entire training dataset), the output is satisfactory so far. As a result, one image will generate 50 different types of faces so we believe that the result can fool people.
d. The limitation, I think it could be, is the running time and that is the more images in the training or testing set, the longer the running time of the code will be. Colab Pro has a connection time of up to 24 hours (Colab free version has a connection time of up to 12 hours), so it is not a suitable platform for training the PICNet model. The solutions should either reduce the number of input images or prolong the training period.
e. I have tried more than 20 times to run the *train.py*. The maximum epoch number that I experimented is 41. Usually, the epoch number does not surpass 30. The average time of training one epoch normally takes 2-3 hours. I am going to share some training results on figure 3.

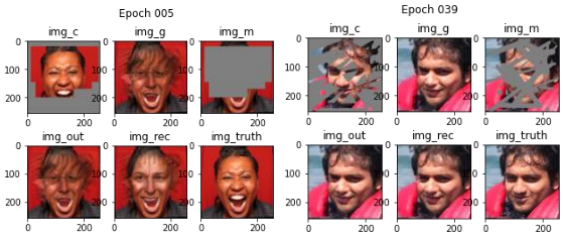


Figure 3: Epoch 5 vs Epoch 39

3.6 Art images training results

Even though we only have 8 images for training, the code is still running. Although we downsample the image sizes to less than 256 x 256, we know the epoch is greater than 15000 now. The dataset's disadvantages are as follows: (1) the number of images is limited. (2) Artwork images are not appropriate for use as a image inpainting training dataset, as I will explain in detail in the conclusion session.

4. Experimental Results

We can find Xu et. al (2020) [7] compare their model T-MAC to PICNet [4] in their paper. In this case, we use SSIM (Structural Similarity Index Measure) and PSRN (Peak Signal-to-Noise Ratio) to assess PICNet's performance. Because there are a lot of images in the "128 x 128 thumbnails" folder (our training dataset), we only choose 25 images from *128 x 128 thumbnails/00000* folder and plus Yunting's selfie to test the model. To be specific, there are 26 input images for our testing. As a single image can generate 50 different types of outputs, we now have a total of 1300 images through PICNet. For the sake of fairness, we calculate the average of each image. That is, we compute the values of 50 different styles and then take an average of these 50 outputs to create a single image value. Now, we have 26 mean values to compare SSIM and PSRN to the ground truths. In the following section, we will begin to compare the outputs and ground truths using quantitative methods.

4.1. Structural Similarity Index Measure (SSIM)

The SSIM value of 26 samples is between 0.52 to 0.92, meaning that there are some images have the high similarity with the ground truths despite the author's ultimate goal is not focusing on reconstructing the original images. Figure 4 depicts the best SSIM outputs as well as the worst SSIM outputs.

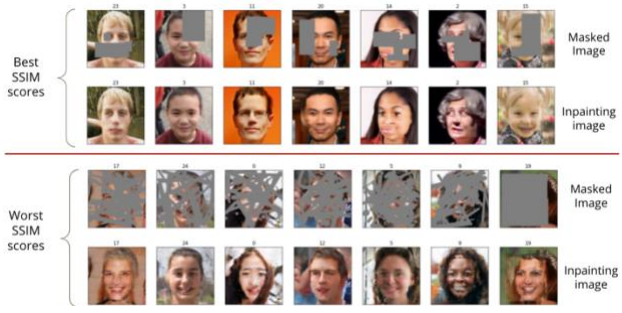


Figure 4: Best and worst SSIM testing outputs

We also compute the SSIM score table in descending order in Table 1. Because one image generates 50 different images so there are 50 different score in one image_id. For each image_id, we extract it average score to compute the SSIM score. If the mean of SSIM is 1, meaning that the output images is the same as the ground truth.

image_id	mean	0	1	...	46	47	48	49
23	0.918419	0.917711	0.915873	...	0.917085	0.920520	0.917907	0.918097
3	0.910127	0.906360	0.908306	...	0.910367	0.909299	0.903683	0.908254
11	0.894535	0.899388	0.892184	...	0.893571	0.896297	0.888303	0.882920
20	0.894138	0.891464	0.897670	...	0.888794	0.890198	0.894168	0.892992
...	...	...	...	...	...	...	...	...
17	0.691193	0.693020	0.689691	...	0.695928	0.695544	0.694331	0.692799
25	0.676537	0.694302	0.679272	...	0.683070	0.678657	0.678499	0.662027
9	0.615126	0.618748	0.618975	...	0.619402	0.618387	0.618162	0.608343
19	0.520405	0.522354	0.530161	...	0.512296	0.540669	0.501832	0.522253

Table 1: SSIM table

4.2. PSRN (Peak Signal-to-Noise Ratio)

We also compare PSRN value of the testing results and the ground truths. PSNR rank is very similar to SSIM rank, meaning that if the PSNR value is not too bad, the SSIM value is good. For instance, the PSRN rank of the top five is image ID: 23, 11, 3, 20, and 14, which has the same rank as the SSIM, such as those shown in table 2.

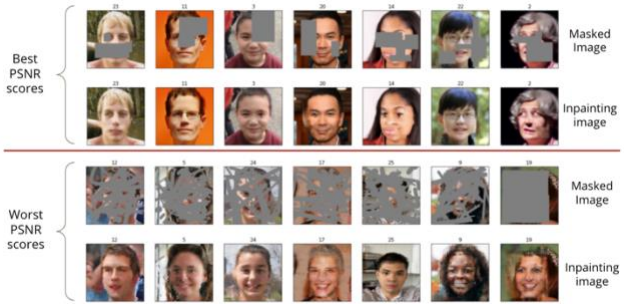
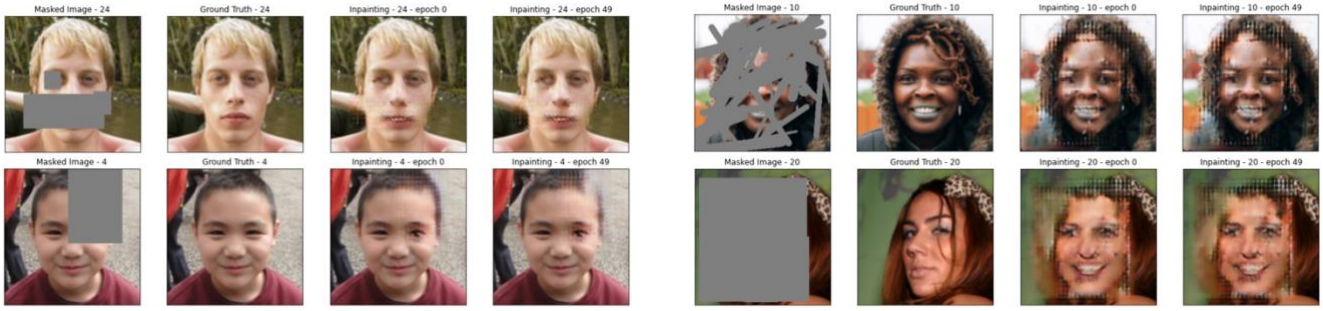


Figure 5: Best and worst PSNR testing output

image_id	ssim_mean	psnr_mean	ssim_rank	psnr_rank
23	0.918419	36.643850	1.0	1.0
3	0.910127	35.976883	2.0	3.0
11	0.894535	36.148645	3.0	2.0
20	0.894138	35.894791	4.0	4.0
14	0.888750	35.352410	5.0	5.0
...	...	...	...	...
24	0.692855	31.427512	22.0	21.0
17	0.691193	31.462293	23.0	20.0
25	0.676537	31.547474	24.0	18.0
9	0.615126	30.901778	25.0	25.0
19	0.520405	29.737357	26.0	26.0

Table 2: SSIM and PSNR comparison table



Best SSIM and PSNR scores

Worst SSIM and PSNR scores

Figure 6: SSIM and PSNR outputs overview

5. Conclusion

Despite the fact that the training process is incomplete, we can conclude from our experiments that smaller the mask region, the better the result of the image inpainting. For most of the cases, the better result images have higher SSIM and PSNR values. In general, given a masked input image, the model can produce natural, realistic-looking images. However, those painted images are not look plausible for our cognition. Our weakness, I believe, is that we did not properly train the model. We are able to get higher epoch for our training if we have opportunities to upgrade our computing environment or have better GPUs solution, such as AWS. I also think art images are useless for training image inpainting, because sometimes art images are too abstract. We should focus more on realistic images for the inpainting training dataset, such as animal, face, and landscape images. We are still interested in a completely trained FFHQ dataset, and we would like to see the image results and quantitative comparison of this model if we have access to more computing resources.

Acknowledgements Pluralistic Image Completion [4] paper provides support for our project, and their work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License. Our purpose is only to conduct academic research for Dr. Bei Xiao's Spring 2021 CSC-676 Computer Vision class assignment at American University. The experimental results are licensed under a MIT license from Yunting Chiu and Huong Doan, and Yunting Chiu write this research paper on his own. Please see my GitHub for more information on this research project.

6. References

[1] Xu, R., Guo, M., Wang, J., Li, X., Zhou, B., & Loy, C. C. (2020). Texture Memory-Augmented Deep Patch-Based Image Inpainting. *arXiv preprint arXiv:2009.13240.*

[2] Hui, Z., Li, J., Wang, X., & Gao, X. (2020). Image fine-grained inpainting. *arXiv preprint arXiv:2002.02609.*

[3] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial networks. *arXiv preprint arXiv:1406.2661.*

[4] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. Pluralistic image completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1438– 1447, 2019

[5] Flickr-Faces-HQ Dataset (FFHQ) dataset (https://github.com/NVlabs/ffhq-dataset)

[6] Art Image: https://www.kaggle.com/ikarus777/best-artworks-of-all-time

[7] Xu, R., Guo, M., Wang, J., Li, X., Zhou, B., & Loy, C. (2020). *Texture Memory-Augmented Deep Patch-Based Image Inpainting.*

[8] Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., & Huang, T. S. (2019). Free-form image inpainting with gated convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 4471-4480).

[9] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318, 2018.*

[10] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, pages 3483–3491, 2015.

[11] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *European Conference on Computer Vision (ECCV)*, 2016.

[12] Jianmin Bao, Dong Chen, Fang Wen, Houqiang Li, and Gang Hua. Cvae-gan: Fine-grained image generation through asymmetric training. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2764–2773. IEEE, 2017.

[13] S. M. Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S. Morcos, Marta Garnelo, Avraham Ruderman, Andrei A. Rusu, Ivo Danihelka, Karol Gregor, David P. Reichert, Lars Buesing, Theophane Weber, Oriol Vinyals, Dan Rosenbaum, Neil Rabinowitz, Helen King, Chloe Hillier, Matt Botvinick, Daan Wierstra, Koray Kavukcuoglu, and Demis Hassabis. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[14] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *European conference on computer vision*, pages 286–301. Springer, 2016.

[15] 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.

[16] Liu, Z., Luo, P., Wang, X., & Tang, X. (2018). Large-scale celebfaces attributes (celeba) dataset. *Retrieved August, 15*(2018), 11.

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

[18] Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., & Efros, A. A. (2016). Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2536-2544).

[19] Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., & Torralba, A. (2017). Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, *40*(6), 1452-1464.

[20] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*, 1097-1105.