Journal of Innovative Image Processing (ISSN: 2582-4252)
www.irojournals.com/iroiip/

GAN Inversion of High-Resolution Images

Tanmay Deshmukh[1], Mohit Bhat[2]

Department of Computer Engineering, Ramrao Adik Institute of Technology, Navi Mumbai, India

E-mail: [1]tanmay7deshmukh@gmail.com, [2]btmohit2020@gmail.com

Abstract

Image generation is the task of automatically generating an image using an input vector z. In recent years, the quest to understand and manipulate this input vector has gained more and more attention due to potential applications. The previous works have shown promising results in interpreting the latent space of pre-trained Generator G to generate images up to 256 x 256 using supervised and unsupervised techniques. This paper addresses the challenge of interpreting the latent space of pre-trained Generator G to generate high-resolution images, i.e., images with resolution up to 1024x1024. This problem is tackled by proposing a new framework that iterates upon Cyclic Reverse Generator (CRG) by upgrading Encoder E present in CRG to handle high-resolution images. This model can successfully interpret the latent space of the generator in complex generative models like Progressive Growling Generative Adversarial Network (PGGAN) and StyleGAN. The framework then maps input vector zf with image attributes defined in the dataset. Moreover, it gives precise control over the output of generator models. This control over generator output is tremendously helpful in enhancing computer vision applications like photo editing and face manipulation. One downside of this framework is the reliance on a comprehensive dataset, thus limiting the use of it.

Keywords: Latent Space, Generative Adversarial Networks, Encoders, Computer Vision, Supervised Learning

1. Introduction

The rapid rise of the internet and social media sites like Instagram has increased the demand for computer vision processes like image classification, image manipulation, etc. The research to solve these and other problems has exploded in the past decade. This paper will focus on image manipulation. Image manipulation can be described as modifying one or more qualities of the image while preserving the rest of the image attributes. The process of

DOI: https://doi.org/10.36548/jiip.2022.2.005
Received: 26.05.2022, received in revised form: 23.06.2022, accepted: 07.07.2022, published: 18.07.2022

image manipulation can be broadly divided into two categories: Image space editing using generative models is used to implement image-to-image translation between two images of different domains like day tonight. Examples of these types of methods include work done in papers[1][2][3][4]. The second method is Latent space editing via generative models. These methods focus on manipulating input to the generative models to get the desired image.

Papers like VAEGAN[5], pSp[6], e4e[7], MaskFaceGAN[8], HifaFace[9], etc., provide us with methods and models to manipulate the output of generative models. Our work is based on a method called Cyclic Reverse Generator (CRG)[10]. These methods work by manipulating the input of image generation models to produce the desired image. Manipulating the generation of new images is tremendously helpful in face editing[11], texture synthesis[12],[13], and image inpainting[14]. Many of these methods are limited in manipulating images of low resolution, i.e., resolution up to 256 x 256. These methods explore the latent space of the image generation model and map image attributes with the latent space of the model. The latent space represents compressed data in which similar data points are together in the area. The latent space of any image generation model is a set of nearby data points that generate similar images. The process of mapping image attributes to the latent space of the Generator is called GAN inversion.

Exploring latent space becomes increasingly more complex with an increase in image resolution. We plan to tackle this problem in our paper. We plan to manipulate the image generation models to produce higher resolution images up to the resolution of 1024 x 1024. These high-resolution images provide us with more detailed and sharp images compared to the lower resolution of 256 x 256. As everybody now has high-quality displays with resolutions up to 2048 × 1080. The lower resolution of the previous method looks like blurry and jagged images on such high-resolution displays, thus making it unsuitable for real-world applications. The increased detail can be tremendously useful in multiple computer vision applications and successfully enable image editing in real-world applications.

The main contributions of this work can be summarized as iterating upon a Cyclic Reverse Generator (CRG) to enable attribute editing of high-resolution images, i.e., images of resolution 1024x1024. This framework enables us to modify the output of pre-trained generative models like Progressive Growing Generative Adversarial Network (PGGAN)[15] and StyleGAN[16].

An increase in the resolution for facial editing makes it possible for application in law enforcement. This method will allow law enforcement to focus on creating the face based on the user's description to reduce the delay caused by the traditional process, which involves creating a sketch by an artist, which is a relatively slow process. Moreover, the commercial use of this method will be tremendously helpful in improving the quality of facial editing software/apps like photoshop and face tune.

The rest of the paper is organized as follows. A summary of the previous related works is presented in Section 2. The proposed method is discussed in detail in Section 3. The results of the image reconstructions are provided in Section 4. The paper is concluded with future directions in Section 5.

2. Related Work

Generative Adversarial Networks (GAN)[17] is a deep learning framework with a Generator(G) and Discriminator(D). The Generator's (G) job is to generate images from uniform random noise z. The Discriminator's (D) job is classifying whether the images are part of a real dataset or generated by the Generator (G). As stated in the paper[18], the main objective of GAN is to find a Nash equilibrium, i.e., the solution to the min-max problem between Generator (G) and Discriminator (D). The work done by Muhammad Muneeb Saad et.al[19] shows us that GAN struggles to reach this Nash equilibrium in practice. The four major problems are Non-convergence and instability, Mode collapse, Unstable generator gradient, and Highly sensitive hyper-parameter tuning. Progressive Growling Generative Adversarial Network (PGGAN) overcomes these obstacles by progressively training the Generator and Discriminator from low-resolution to high resolution, resulting in faster and more stable training.

A. Jahanian et al., [20] explained Generator (G) models in GANs like PGGAN map latent points into an input space. This input space is referred to as latent space. GANs use these sets of latent points to learn nonlinear data distribution, i.e., our training data. These latent points are also referred to as latent code. The Generator (G) model takes a sampled latent code as the input and outputs an image. To utilize GAN for image manipulation and other computer vision applications, we need to find a way to control the output of the Generator model to get the desired image. The most popular method to control the result of the generator model is to reverse the mapping from the latent space to the image. This method is also known as GAN Inversion.

Xia Weihao et al., [21] defined GAN inversion as the process of inverting a given image back into the latent space of a pre-trained GAN model so that we can faithfully reconstruct the image. As GAN inversion is essential to implement applications like image restoration and image manipulation, thus there have been significant advances in the field.

There are three main techniques of GAN inversion.

- Learning-based GAN Inversion: In these types of methods [22],[23],[24], a different neural network or network is used to map images into the latent space of the Generator. While the computational cost of training is high, the inference time of these types of methods is extremely low.
- Optimization-based GAN Inversion: In these methods[25][26][27], an optimization algorithm slowly minimizes the loss function to get a faithfully reconstructed image. These types of methods can provide high-quality results but are computationally expensive.
- Hybrid of Learning-based and Optimization-based GAN Inversion: In these types of methods[28][29], first, the learning-based techniques are used to find an approximate latent code of the image, and then optimization algorithms are used to refine them further. These methods try to find a balance between computational cost and inference time.

GAN inversion of high-resolution images is limited by the capabilities of the Generator(G) model to generate high-resolution, i.e., 1024 x 1024 images, and the ability of the inversion techniques to handle such large images. Previous research like J.-Y. Zhu et.al[29] focused on interpreting the latent space of simple GAN models with resolutions up to 64x64.

With advancements in generative models like PGGAN and StyleGAN, it was possible to generate higher resolution images up to 1024. There are many new techniques to find GAN inversions for PGGAN and StyleGAN. Many methods like those mentioned by M. Rosca et al., [30] and D. Ulyanov et al., [31] successfully performed GAN inversion on images up to a resolution of 256x256. Some techniques, as mentioned by V. Dumoulin et al., [32] could not work with image resolution greater than 256x256 due to the computational cost of finding latent code of higher resolution images.

There have been multiple methods to find GAN inversion for high-resolution images. One of the first methods was provided by the creators of StyleGAN in their paper on

StyleGAN2[33]. The paper [33] determined that much more accurate latent code can be found for the Generator if latent codes for each separate layer of the Generator is found instead of finding a common latent code for the entire Generator. Much of the further research in this field is based on this observation. Elad Richardson et al., [6] explain the pixel2style2pixel (pSp) framework. The pSp framework proposed a novel encoder network that generates a series of style vectors.

Omer Tov et al., [7] proposed an e4e network architecture that explores the tradeoff between optimizing the encoder for finding latent code for human perception and image editability. Daniel Roich et al., [34] expanded on ideas in Omer Tov et al., [7]to create a new framework called Pivotal Tuning Inversion (PTI), where the initial latent code is kept as a pivot around which the latent code is further finetuned. Similar to Daniel Roich et al., [34], Yuval Alaluf et al., [35] also proposed a new framework called ReStyle where instead of directly predicting the latent code of the image, the Encoder predicts with the latent code using the optimization method, i.e., in a self-correcting manner.

These GAN inversion methods are the essential step in the process of image manipulation by editing the latent space. It provides us with the ability to map a specific image attribute to GAN's latent space. Xia Weihao et al., [21] defined image manipulation as the process of editing a specific region of the image by manipulating its latent code. We can express image manipulation as equations 1 and 2.

$$z\alpha = zb + \alpha * zf \tag{1}$$

$$x' = G(z\alpha) \tag{2}$$

Where zb is the latent code of the image, α is the step for manipulation, za is the feature vector of the encoded for a particular image attribute. We can change the image attribute in the positive or negative direction as desired by changing the value of α. The x ′ signifies the newly generated image created by passing za into our pre-trained Generator G. We can use the image manipulation technique for background removal, as shown by the David Bau et al., [38] by subtracting the latent code corresponding to the background from the latent code of the input image. Another application of image manipulation is facial editing. Jiapeng Zhu et al., [36] and Tero Karras et al., [37] explored how we can use GAN inversion and image manipulation to edit one or more than one facial attribute of the image. Paper [36] proposed a new method for in-domain GAN inversion, ensuring that latent code

found using GAN inversion can be used for image manipulation and image editing. Jiapeng Zhu et al., [36] achieved this by using a Domain Guided Encoder to finetune the latent code produced by another Encoder. We will be iterating upon one such architecture called Cyclic Reverse Generator (CRG).

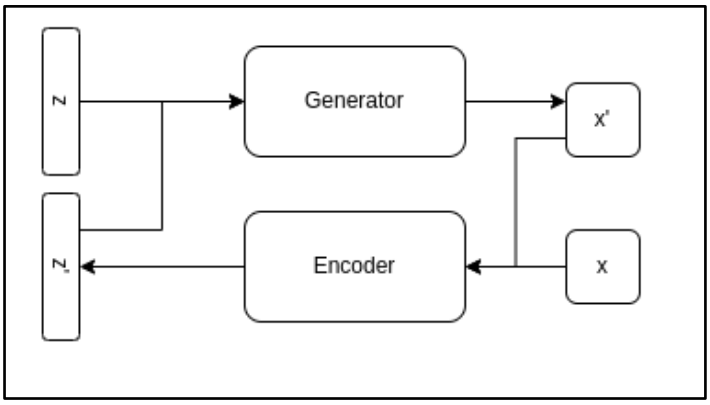

Figure 1. The architecture of the Cyclic Reverse Generator

CRG involves a cyclically connected to a pretrained Generator of PGGAN or StyleGAN and Encoder, as shown in figure 1. In Cyclic Reverse Generator (CRG), an Encoder (E) is cyclically added with the pretrained Generator (G). This Encoder converts images generated by Generator (G) i.e., x‘ into the latent code z‘ using the error function between z and z‘ as shown in figure 1. The Encoder (E) is trained until it can approximate z‘ to initial latent code z.

Cyclic Reverse Generator is limited to finding GAN inversion of images up to 256 x 256 resolution. Our paper proposes an iteration of the Cyclic Reverse Generator(CRG) to modify the Encoder (E) to handle high resolution, 1024 x 1024 images. Our Encoder allows CRG to approximate the face generator’s inverse function by mapping the faces generated by Generator G to the corresponding latent code. Previous methods for finding latent vectors of high-resolution images are limited to specific GAN architecture. Our method is future-proof as it allows us to find the latent representation of any GAN architecture, giving us the capability to find a latent representation of any GAN architecture that may be developed in the future.

3. Proposed Work

A combination of the CelebA dataset and CelebAHQ dataset has been used. CelebA has a lot of data and annotations, with 202,599 face images, five iconic sites, and 40 binary

attribute annotations per picture. The CelebA dataset contains an image of resolution 256x256.CelebAHQ dataset contains thirty thousand images with images of resolution 1024x1024. We will be using those 30,000 images for training with an 0.8 train and test split i.e, 24,000 training images and 6 thousand testing images.

A pre-trained Progressively Growing Generative Adversarial Network (PGGAN) is used as Generator. Progressive Growing GAN uses a generator and discriminator model with the same general structure and starts with tiny images, such as 4×4 pixels. During training, new blocks of convolutional layers are systematically added to both the generator and discriminator models. The incremental addition of the layers allows the models to learn broader detail effectively and then later learn finer or minute detail, both on the Generator and discriminator. This approach enables the generation of large-high-quality images, such as 1024×1024 photorealistic faces that do not exist.

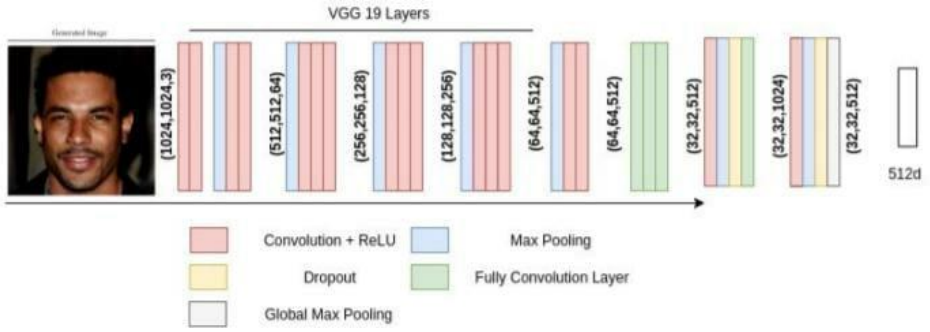

Figure 2. Design of Encoder

Figure 2 shows the modified design to create a CRG encoder that can generate latent space attributes for images up to size 1024x1024 by changing the backbone of the system from VGG16 to VGG19. Our model is based on VGG19, with convolution layers followed by a max-pooling layer which reduces the height and width of a volume. The model has 64 filters, and we will double it to 128 and then to 256 and use 512 layers in the last layers. On each step or doubling through each stack of convolution layers, the sum of the filter to use is approximately doubled. That is another fundamental concept used to build this network's architecture. Train the Encoder using CRG to find corresponding features in latent space. MSE loss is used as the loss to calculate between difference between latent vectors. The Encoder parameters are optimized at each iteration using gradient descent algorithms. In our case, it is the Adam optimization algorithm. All weights of the Encoder's layers, including the VGG Network's pre-trained layers and weights, are trained at the same learning rate.

Hyperparameters of our Encoder are the batch size of 16, epsilon of 1e08. A 1e4 learning rate was used to optimize all of the parameters in the encoder architecture and to adapt the current VGG layer weights appropriately. When the minimum validity loss improves for ten epochs, the learning rate is decreased by two. Overfitting can be avoided when different regularizations are used in training, and 0.5 spatial dropouts are used. The model checkpointing is used to save the best possible Generator and to finish the training process using early stopping if the validity loss does not change for 20 epochs. After the training, it is presumed that the images generated by the Generator and the reconstruction created by using the feature vector generated by our Encoder will look the same. The actual picture reconstructions are of sufficient quality to enable attribute embedding to find precise latent vector directions. With the precise interpretation of latent space, we can successfully enhance image generation for higher resolution images and applications requiring fine-grained image editing while preserving the image features like face editing and photo makeover.

Pytorch is used to create the model, and the model is trained on the computer server with 2 RTX 6000 with 24GB VRAM each. The higher VRAM of RTX 6000 has been a tremendous help in training our model for higher resolution images from CelebA HQ dataset.

4. Results and Discussion

We will be looking at how the encoder performs by comparing the image generated by z‘, the feature vector generated by the encoder, and the image generated by the actual feature vector.

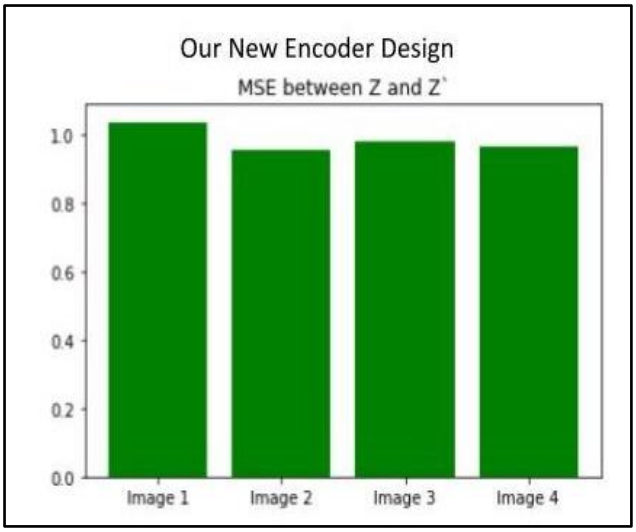

Figure 3. Comparison of MSE between two encoders

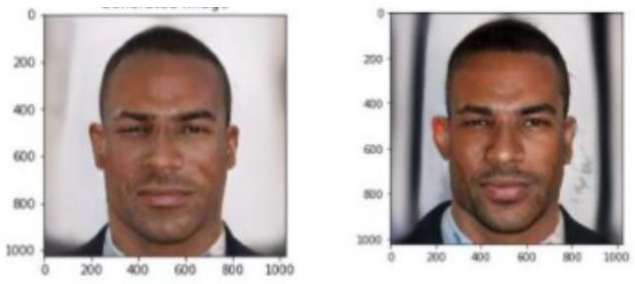

Figure 4. Comparison between generated image(left) and original image(right)

Figure 3 shows how the model calculates MSE between the original image and image reconstruction of 4 high-resolution images, with an average MSE of 0.92 across all photos. Figure 4 shows us a comparison between images generated by our method and the original image. MSE, also referred to as Mean Squared Error determines the similarity between two images. MSE closer to 1 suggests that our model successfully approximates the images. Our paper successfully demonstrated the ability to perform GAN inversion of high-resolution images using a modified cyclic reverse architecture.

5. Conclusion

This proposed model and framework can successfully interpret the latent space and create a semantic mapping of complex generative models like Progressively Growing Generative Adversarial Network. The previous method using CRG was limited to generating and manipulating images up to 256x256 our mappings allow us to generate higher resolution images 1024x1024 where and manipulate images on the granular level while preserving the overall characteristics of the image. However, being a supervised learning process, the major drawback of this framework is the acquisition of labeled data. In future work, the dependency on labeled data is aimed to be reduced by implementing unsupervised methods to enable mapping latent space.

References

[1] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, et al. Image-to-image translation with conditional adversarial networks. In CVPR, 2017.

[2] Binod Bhattarai and Tae-Kyun Kim. Inducing optimal attribute representations for conditional gans. ECCV, 2020.

[3] Yunjey Choi, Minje Choi, Munyoung Kim, et al. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In CVPR, 2018.

[4] Po-Wei Wu, Yu-Jing Lin, Che-Han Chang, et al. Relgan: Multi-domain image-to-image translation via relative attributes. In ICCV, 2019.

[5] Larsen, Anders Boesen Lindbo et al. "Autoencoding beyond pixels using a learned similarity metric." ArXiv abs/1512.09300 (2016): n. pag.

[6] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In CVPR, 2021.

[7] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel CohenOr. Designing an encoder for stylegan image manipulation. TOG, 2021.

[8] Pernuvs, Martin et al. "High-Resolution Face Editing with Masked GAN Latent Code Optimization." (2021).

[9] Gao, Yue et al. "High-Fidelity and Arbitrary Face Editing." 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2021): 16110-16119.

[10] Dogan, Yahya and Hacer Yalim Keles. "Iterative Facial Image Inpainting using Cyclic Reverse Generator." ArXiv abs/2101.07036 (2021): n. pag.

[11] X. Wu, K. Xu and P. Hall, "A survey of image synthesis and editing with generative adversarial networks," in Tsinghua Science and Technology, vol. 22, no. 6, pp. 660-674, December 2017, doi: 10.23919/TST.2017.8195348.

[12] Wenqi Xian, Patsorn Sangkloy, Varun Agrawal, Amit Raj, Jingwan Lu, Chen Fang, Fisher Yu, James Hays; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 8456- 8465.

[13] Dong J, Liu J, Yao K, Chantler M, Qi L, Yu H, Jian M. Survey of Procedural Methods for Two-Dimensional Texture Generation. Sensors. 2020; 20(4):1135.

[14] Elharrouss, O., Almaadeed, N., Al-Maadeed, S. et al. Image Inpainting: A Review . Neural Process Lett 51, 2007–2028 (2020).

[15] Karras, Tero et al. "Progressive Growing of GANs for Improved Quality, Stability, and Variation." ArXiv abs/1710.10196 (2018): n. pag.

[16] Karras, Tero et al. "Analyzing and Improving the Image Quality of StyleGAN." 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020): 8107-8116.

[17] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. . Generative adversarial nets. In Advances in neural information processing systems (2014)(pp. 2672–2680).

[18] 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 NeurIPS, 2017.

[19] Saad, Muhammad Muneeb et al. "A Survey on Training Challenges in Generative Adversarial Networks for Biomedical Image Analysis." ArXiv abs/2201.07646 (2022): n. pag.

[20] A. Jahanian, L. Chai, and P. Isola, "On the "steerability" of generative adversarial networks," in ICLR, 2020.

[21] Xia, Weihao et al. "GAN Inversion: A Survey." ArXiv abs/2101.05278 (2021): n. pag.

[22] J.-Y. Zhu, P. Krahenb ¨ uhl, E. Shechtman, and A. A. Efros, "Gen- ¨ erative visual manipulation on the natural image manifold," in ECCV, 2016.

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

[24] D. Bau, J.-Y. Zhu, J. Wulff, W. Peebles, H. Strobelt, B. Zhou, and A. Torralba, "Inverting layers of a large generator," in ICLR Workshop, vol. 2, no. 3, 2019.

[25] R. Abdal, Y. Qin, and P. Wonka, "Image2StyleGAN: How to embed images into the StyleGAN latent space?" in ICCV, 2019.

[26] R Abdal, "Image2StyleGAN++: How to edit the embedded images?" in CVPR, 2020.

[27] A. Creswell and A. A. Bharath, "Inverting the generator of a generative adversarial network," TNNLS, 2018.

[28] J. Zhu, Y. Shen, D. Zhao, and B. Zhou, "In-domain gan inversion for real image editing," in ECCV, 2020.

[29] J.-Y. Zhu, P. Krahenb ¨ uhl, E. Shechtman, and A. A. Efros, "Gen- ¨ erative visual manipulation on the natural image manifold," in ECCV, 2016.

[30] M. Rosca, B. Lakshminarayanan, D. Warde-Farley, and S. Mohamed, "Variational approaches for auto-encoding generative adversarial networks," arXiv preprint.

[31] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "It takes (only) two: Adversarial generator-encoder networks," in ThirtySecond AAAI Conference on Artificial Intelligence, 2018.

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

[33] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T. (2020). Analyzing and Improving the Image Quality of StyleGAN. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 8107-8116.

[34] Roich, Daniel et al. “Pivotal Tuning for Latent-based Editing of Real Images.” ArXiv abs/2106.05744 (2021): n. pag.

[35] Alaluf, Yuval et al. “ReStyle: A Residual-Based StyleGAN Encoder via Iterative Refinement.” ArXiv abs/2104.02699 (2021): n. pag.

[36] Zhu, Jiapeng et al. “In-Domain GAN Inversion for Real Image Editing.” ArXiv abs/2004.00049 (2020): n. pag.

[37] Karras, Tero et al. “Alias-Free Generative Adversarial Networks.” ArXiv abs/2106.12423 (2021): n. pag.

[38] Bau, David et al. “Semantic photo manipulation with a generative image prior.” ACM Transactions on Graphics (TOG) 38 (2019): 1 - 11.