Generating Images with StyleGAN3

Anonymous author

Abstract

This paper proposes and implements StyleGAN3, for generating images from the FFHQ 128x128 dataset. We discuss how our model improves on previous StyleGAN models and discuss the outcome and the limitations of what our results produce.

1 Methodology

The purpose of StyleGAN (and our model StyleGAN3) is to generate images that are not only realistic but also high quality, diverse and controllable, using generative adversarial networks (GANs). The work is based on Progressive GANS, Adaptive Instance Normalization(AdaIN), and style transfer. [1] GAN is a neural network that consists of a generator and a discriminator, where the generator creates images from random noise, and the discriminator tries to distinguish between those images, and the real images. When training, the generator learns to create increasingly realistic images to make it harder for the discriminator. [2] Unlike other GAN models, StyleGAN uses a technique called style mixing that allows it to combine different features from different images, resulting in more diverse outputs, and uses a technique called "adaptive instance normalization" to control the style of the generated images, allowing for fine-grained control over features such as color and texture. As a result, StyleGAN can generate images that are not only realistic but also high quality, diverse and controllable. [1]

This architecture differs from the traditional network as the network latent vectors z from StyleGAN pass through a mapping network f after normalisation, are then transformed, are passed to blocks and are added with noise to model stochastic details. We then have the synthesis network where we apply Adaptive instance normalisation (AdaIN) as outlined below [3]:

$$\text{AdaIN}(x, y) = \sigma(y) \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \mu(y)$$

AdaIN is able to model any arbitrary style by just giving our desired feature image mean as β and variance as γ, it receives an input x(content) and a style input y, and simply aligns the channel-wise mean and variance of x to match those of y. This way the generator can learn to control the style of the generated images by manipulating the values of the style vector, allowing for a greater degree of control and diversity in the generated images[3].

Another change with StyleGAN from the traditional GAN is that it feeds a constant tensor into the synthesis network instead of the latent code as the original way was not found to have any benefits.

Our next step is style mixing where we employ mixing regularisation to be able to mix the styles of different latent codes. For example, if we have two latent codes z_1, z_2 and pass it through f, we can have w_1 control the styles before the crossover point and have a w_2 for after[4]. As a more practical example, if we have two sources A and B, we can take face shape, hairstyle and accessories from A, whilst colors of the persons, hair and eyes can be from B.

StyleGAN is also known for using spherical interpolation (slerp) to interpolate between two latent vectors for latent space W. This allows for smooth transitions between different styles of the generated images. We can define the slerp interpolation with the equation[5]: $l_Z = E\left[\frac{1}{\epsilon^2} d\left(G\left(\text{slerp}\left(\mathbf{z}_1, \mathbf{z}_2; t\right)\right), G\left(\text{slerp}\left(\mathbf{z}_1, \mathbf{z}_2; t + \epsilon\right)\right), \right)\right]$

StyleGAN has several developments over the years. After StyleGAN, StyleGAN2 was created to remove water-droplet artifacts in StyleGAN. StyleGAN2-ada was then created to train StyleGAN2 with limited data. Finally, we have the model that we will be implementing: StyleGAN3 whos main aim was to make transition animation more natural. Below we will be briefly demonstrating each StyleGAN and how each one built on each other to then reach StyleGAN3[3].

1.0.1 StyleGAN2

StyleGAN2 was originally based on this hypothesis "By creating a strong, localized spike that dominates the statistics, the generator can effectively scale the signal as it likes elsewhere." [3] It was theorised that the droplet artifact is due to the normalisation step of AdaIN. To solve this, AdaIN was replaced with Modulation:

$$w'_{ijk} = s_i \cdot w_{ijk}$$

and Demodulation/Normalisation:

$$w''_{ijk} = w'_{ijk} / \quad \sigma_j$$

where:

$$\sigma_j = \sqrt{\sum_{i,k} w'_{ijk}}$$

so finally:

$$w''_{ijk} = w'_{ijk} / \sqrt{\sum_{i,k} w'_{ijk} + \epsilon},$$

StyleGAN2 also introduces Perceptual Path length (PPL) PPL measures the mean squared distance between the feature representations of pairs generated images at different points in the W space. By minimizing this distance, the generator is encouraged to generate smooth and continuous images.

$$\mathbb{E}_{\mathbf{w},\mathbf{y}\sim\mathcal{N}(0,\mathbf{I})} \left(\left\| \mathbf{J}_{\mathbf{w}}^T \mathbf{y} \right\|_2 - a \right)^2$$

where:

$$\mathbf{J}_{\mathbf{w}} = \partial g(\mathbf{w}) / \partial \mathbf{w}$$

StyleGAN2 also introduces, progressive growing, where the generator is trained on images of increasing resolution which allows better convergence and stability. Skip Connections between the encoder and decoder is also an additional change which help preserve information at different scales[5].

1.0.2 StyleGAN2-Ada

Previously, we have the same augmentation probability for all transformations in the stochastic discriminator augmentation setup. This method aims to automatically tune our augmentation strength by looking at the over-fitting status. We quantify over-fitting with the heuristic[4]:

$$r_v = \frac{\mathbb{E}\left[D_{\text{train}}\right] - \mathbb{E}\left[D_{\text{validation}}\right]}{\mathbb{E}\left[D_{\text{train}}\right] - \mathbb{E}\left[D_{\text{generated}}\right]}$$

which is used for comparison and a second heuristic: $r_t = E[sign(D_{train})]$

1.0.3 StyleGAN3

Finally, we will look into our current model: StyleGAN3 which aims to solve texture sticking which is a result of the morphing transition[7]. Our feature maps from intermediate layers can contain unintentional positional references from image borders, per-pixel noise inputs and positional encoding which the generator might try to falsely replicate to be stuck in certain coordinates. Aliasing is another source of texture sticking, however, it is the biggest problem as it is difficult to identify and fix. This occurs when high-frequency information in the image is lost due to under-sampling or other image processing operations, or point-wise application of linearities[4] resulting in unwanted artifacts. Therefore, the aim of StyleGAN3 is to remove all these causes of texture sticking. These changes are based on Fourier Features[7]. [width=0.5]figures/arc.png ur first step to combating this is to switch from a discrete domain to a continous domain. Previously, the generator produced images by generating discrete pixel values, which are then d to integer values. However, this results in artifacts which can make the generated images look less realistic. Therefore, to change this we need to redesign the convolution, upsampling,downsampling and non-linearity layers. The following equations demonstrate discrete and continuous operations, where s and s' are the input and output sampling rates [4]:

$$\mathbf{f}(z) = \phi_{s'} * \mathbf{F}\left(\amalg_s \odot z\right), \quad \mathbf{F}(Z) = \amalg_{s'} \odot \mathbf{f}\left(\phi_s * Z\right),$$

and our discrete and continuous domain convolution equations are:

$$\mathbf{F}_{\text{conv}}\left(Z\right) = K * Z$$

$$\mathbf{f}_{\text{conv}}\left(z\right) = \phi_s * \left(K * \left(\amalg_s \odot z\right)\right) = K * \left(\phi_s * \left(\amalg_s \odot z\right)\right) = K * z$$

This results in no new frequencies that would require additional computation or special handling to maintain equivariance. For upsampling, we do not want to modify the continuous representation. The discrete domain and the continuous domain are shown respectively as:

$$\mathbf{F}_{\text{up}}\left(Z\right) = \amalg_{s'} \odot \left(\phi_s * Z\right)$$
$$\mathbf{f}_{\text{up}}\left(z\right) = z$$

For down sampling we treat the continuous domain as a low pass filter:

$$\mathbf{F}_{\text{down}}\left(Z\right) = \amalg_{s'} \odot \left(\psi_{s'} * \left(\phi_s * Z\right)\right)$$
$$= 1/s^2 \cdot \amalg_{s'} \odot \left(\psi_{s'} * \psi_s * Z\right) = \left(s'/s\right)^2 \cdot \amalg_{s'} \odot \left(\phi_{s'} * Z\right)$$

$$\mathrm{f}_{\text{down}}\left(z\right) = \psi_{s'} * z$$

where these equations show the discrete and continuous domain respectively.

For Non-Linearity we want to remove high frequency content by using a low pass filter. We have the equations for discrete and continuous as:

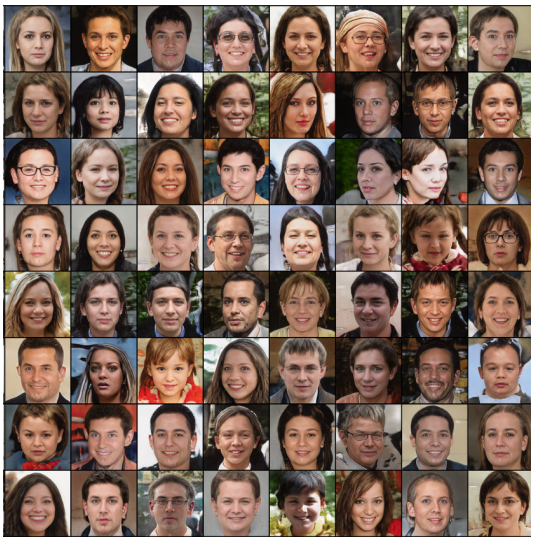
$$\mathbf{F}_{\sigma}(Z) = s^2 \cdot \amalg_s \odot \left(\phi_s * \sigma\left(\phi_s * Z\right)\right)$$
$$\psi_s := s^2 \cdot \phi_s$$
$$f_{\sigma}(z) = \psi_s * \sigma(z)$$
$$= s^2 \cdot \phi_s * \sigma(z)$$

When building upon the original StyleGAN2-Ada code we also add a rotation equivariance where we replace 3x3 convolutions with 1x1 and changing the sinc-based downsampling filter to a radially symmetric jinc-based filter. Furthermore, we add a translation equivariance demonstrated as[6]:

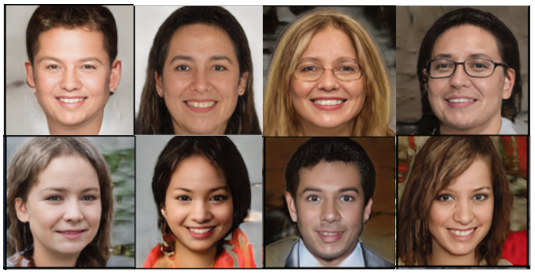
$$\mathrm{EQ-T} = 10 \cdot \log_{10}\left(I_{\max}^2 / \mathbb{E}_{\mathbf{w}\sim\mathcal{W}, x\sim\mathcal{X}^2, p\sim\mathcal{V}, c\sim\mathcal{C}}\left[\left(\mathbf{g}\left(\mathbf{t}_x\left[z_0\right]; \mathbf{w}\right)_c\left(p\right) - \mathbf{t}_x\left[\mathbf{g}\left(z_0; \mathbf{w}\right)\right]_c\left(p\right)\right)^2\right]\right)$$

2 Results

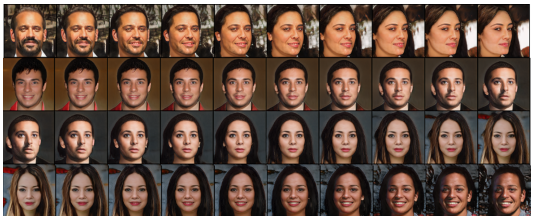
We will first analyse our results with 64 non-cherry picked images.



The results do have some diversity, and whilst some look realistic, others look drastically flawed. However, from 8 cherry-picked images it shows a lot of promise:



The next results show images generated by interpolating between points in the latent space:



3 Limitations

The results shown are not great, however, do show a lot of promise. We were able to train our model on and off for about 15 hours. This issue was due to NCC, lack of time and other computational resources. As we see the progress in such a short amount of time, I do believe that within a few days, the results will be very impressive.

Bonuses

This submission has a total bonus of 0 marks, as it is trained on FFHQ 128x128 but also uses a GAN.

[1] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T. (2020). Analyzing and improving the image quality of stylegan. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 8110-8119).

[2]Understanding the StyleGAN and StyleGAN2 Architecture https://medium.com/analytics-vidhya/understanding-the-stylegan-and-stylegan2-architecture-add9e992747d Accessed: 2023-07-01.

[3]StyleGAN vs StyleGAN2 vs StyleGAN2-ADA vs StyleGAN3 https://medium.com/@steinsfu/stylegan-vs-stylegan2-vs-stylegan2-ada-vs-stylegan3-c5e201329c8a Accessed: 2023-07-01.

[4]StyleGAN3 Clearly Explained! https://medium.com/@steinsfu/stylegan3-clearly-explained-793edbcc8048 Accessed: 2023-07-01.

[5] Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., Aila, T. (2020). Training generative adversarial networks with limited data. Advances in neural information processing systems, 33, 12104-12114.

[6] Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J., Aila, T. (2021). Alias-free generative adversarial networks. Advances in Neural Information Processing Systems, 34, 852-863.

[7] Wang, W. (2022, September). Evolution of StyleGAN3. In 2022 International Conference on Electronics and Devices, Computational Science (ICEDCS) (pp. 5-13). IEEE.