UNIVERSIDAD DE CONCEPCIÓN
FACULTAD DE INGENIERÍA

Generative Adversarial Neural Networks for the generation of High Resolution Galaxy Images

Alejandra Fernández Molina

Profesor Guía: Guillermo Cabrera
Comisión: Pierluigi Cerulo, Fernando Rannou

Enero 2023

Concepción, Chile

Acknowledgement

First, I would like to thank my advising teacher, Guillermo Cabrera, for his support and advice. I also like to thank the DS-League team for their willingness to answer my questions, especially Monserrat Martínez, for her outstanding participation in the collection of galaxy image data and in the process of the evaluation of the physical properties, and Cristóbal Donoso for helping me when I had trouble using the server to run my experiments.

Secondly, I want to thank my family, friends and teachers for encouraging me and for their unconditional support. Special thanks to Valentina, for accompanying me in the process, helping me gather the strength to work, and being there when things got tough, and Paulo, for the moral support.

Finally, I would like to thank FONDECYT for partially financing this project. (FONDECYT Initiation grant No. 11191130)

Abstract

High-resolution images of nearby galaxies are difficult to obtain since the number of close (low redshift) observations is limited and galaxies with large angular size require several observations or high-demand telescopes. We present a new method to generate synthetic galaxy images in FITS format, utilizing a Style-Based Generator Architecture for Generative Adversarial Networks (StyleGAN) with a fine-tuning technique with JPG and FITS format images, using simulated galaxies from the Illustris project and real galaxies from the Sloan Digital Sky Survey (SDSS). This approach allows more control over the generated samples, and it's able to modify the galaxies using different styles. We test the fine-tuning approach on JPG data to obtain images that resemble the training data, then training of the network with FITS images. We utilized the Fréchet Inception Distance (FID) metric for evaluation, where the lower the score, the better. We obtained generated samples with an FID value of 148.5, a high value in comparison to the one reached on the original StyleGAN paper due to blurriness and noisy elements on the images. Nevertheless, the samples resemble both the morphological and physical properties of the real data.

Keywords – StyleGAN, Image Generation, Astronomy, Galaxies

Resumen

Las imágenes de alta resolución de galaxias cercanas son difíciles de obtener ya que su gran tamaño angular requiere varias observaciones o telescopios de alta demanda. En este trabajo presentamos un nuevo método para generar imágenes de galaxias en formato FITS, utilizando una arquitectura basada en estilos para redes generativas adversarias (StyleGAN) con la técnica de fine-tuning con imagenes en formato JPG y FITS, utilizando galaxias simuladas del proyecto Illustris y galaxias reales del Sloan Digital Sky Survey (SDSS). Este enfoque permite un mayor control sobre las muestras generadas, y es capaz de modificar las galaxias utilizando diferentes estilos. Probamos utilizando fine-tuning en datos JPG para obtener imágenes que se asemejan a los datos de entrenamiento, luego entrenamiento de la red con imágenes FITS. Utilizamos la métrica Fréchet Inception Distance (FID) para la evaluación, donde cuan mas bajo es el valor, mejor. Obtuvimos muestras generadas con un valor FID de 148.5, un valor alto en comparación con el alcanzado en el paper original de la StyleGAN debido a la borrosidad y ruido presente en las imágenes. Sin embargo, las muestras se asemejan tanto a las propiedades morfológicas como físicas de los datos reales.

Palabras clave – StyleGAN, Generación de Imágenes, Astronomía, Galaxias

Contents

List of Tables

List of Figures

1 Introduction

Galaxies are the basic building blocks of the Universe, and in the last decades, there has been a build-up of large datasets of galaxy images. However, real high-resolution images remain scarce due to the need for highly demanded telescopes or the requirement to merge several observations to be produced. High-resolution images can be used for tasks like algorithm calibration in the weak lensing field or deblend overlapping galaxies in compound images. Galaxy images can also be used to calibrate machine-learning tools for classification or event detection. The amount of data these tasks need will continue to grow as the scope of astronomy increases. Machine learning techniques like Generative Adversarial Networks (GANs) are particularly utilized for image generation tasks [1]–[3]. These are deep generative models where two networks, a generator and a discriminator, compete with each other in the form of a zero-sum game to generate new data that resemble the training data. GANs have been used in astronomy in tasks such as denoising and recovering features from degraded images [4], increasing the quality of blurred solar images [5], and for simulating synthetic galaxy images at different pixel resolutions [6]–[9]. We propose to approach the generation of synthetic high-resolution galaxy images utilizing a StyleGAN [10], that is a GAN that modifies the architecture of the generator using the idea of style transfer, where one can add (or transfer) a visual style to a content image without changing the information of the content, which improves the quality of the generated images and reduces feature entanglement. This architecture is able to produce realistic high-resolution images (1024×1024 pixels) for the human eye and, in 2019, became state of the art in human face generation, obtaining the lowest Fréchet Inception Distance (FID) score using the dataset CelebA-HQ. This approach allows for the possibility to modify the generated outputs due to its focus on adding "styles" to the images. This feature has the potential to alter the angle, distance, or luminosity of the image without changing the content by injecting different styles. The StyleGAN also allows the resulting model to be a starting point to generate galaxy images at higher resolution without needing a larger amount of data and extensive training time, making it possible to generate synthetic data of new telescopes that have not been in operation for very long. Therefore, we propose to generate realistic high-resolution galaxy images directly from the data using a style-driven approach utilizing the StyleGAN architecture. Astronomy images are commonly stored in a format named Flexible Image Transport System (FITS), which allows one to store multi-dimensional arrays for image data without compressing them. FITS files can also store extra information about the content in the same file, so we want our final model to produce images in a FITS format. Because we want to generate realistic images, real galaxy images were collected. Nevertheless, due to the low quantity obtained (less than 200 images), a simulated dataset with a larger amount of images was obtained. However, the difference between the images of both datasets was evident. Due to this, we decided to use a fine-tuning technique, which would allow us training the model with all the data, initially with simulated galaxies for an extended time, and then with real images, in a shorter time, obtaining, in theory, results that would be similar to the last dataset used in training. We start by training a model first with JPG images using simulated galaxies, then we fine-tune the model. Afterwards we retrain the model with FITS images of simulated galaxies and then with FITS images of real galaxies. This thesis describes the StyleGAN model and the data collection and training process. We then present a comparison of a randomized set of data created by the generator and the real data using a color-magnitude diagram to verify that the generated images follow the physical properties of real-life galaxies.

1.1 Hypothesis

Using the StyleGAN neural network with a fine tuning approach with simulated and real data will allow one to produce galaxy images similar to the training dataset that are distinct, reaching an FID lower than 50, and realistic, where from $1,000$ synthesised images, at least 70% follow the physical properties of real-life galaxies.

1.2 Objectives

1.2.1 General objective

- Produce high resolution synthetic galaxy images that follow the physical properties of real-life galaxies using a style-driven approach

1.2.2 Specific objectives

- Analyze and understand the structure of the StyleGAN

- Evaluate the performance of the StyleGAN on its original domain using fine-tuning
- Train a StyleGAN neural network to produce galaxy images in a JPG format
- Modify the model to be able to produce FITS images
- Use the technique of fine-tuning to train the model with galaxy images in FITS format
- Evaluate the results of the generated images.

2 Literature Review

2.1 Image Generation

In terms of image generation, the quality and resolution of the generated images have been improving since 2014. The method of Progressive GANs [3] had proved to reduce training time and produce better results by allowing the network to learn first the basic structure of the data using lower resolutions images and then learn more specific details by increasing the resolution. This method uses multiple versions of the training data at different resolutions and focuses on the training process: each time the resolution increases, the training dataset changes to the one with that specific resolution, and new layers are added to both networks (generator and discriminator). Because the only feedback the generator gets is the result of the discriminator, spectral normalization to improve image generation is used in [11]. The main idea is to improve the generator by improving the discriminator's behavior, making it more stable and precise over time. They focus on maintaining control over the discriminator function by constraining each layer's spectral norm. In [12] conditional GANs are occupied, using the image's label as extra information and adding it through a new network to different layers of the generator, while the latent vector is only passed through the input layer. Also, in [13] it is shown that by using the truncation trick, where the values of the input z which are greater than a certain threshold are resampled, there is an improvement in the Fréchet inception distance (FID) and in the quality of the images. In [14] a new StyleGAN version is used for image to image translation to produce cartoon avatars, using the fine-tuning approach to reduce training time and also freezing generator layers to improve the quality of the results. Regarding style transfer in images, [15] demonstrated that by using deep neural networks, content and style can not only be extracted but also separated. In [16], the general idea was optimized using adaptive instance normalization, where an instance is normalized to match its mean and variance with the style input.

2.2 Astronomical Image Generation

GANs have been used in astronomy as an alternative computationally expensive method for simulations. For example, in [17], they produce high-fidelity weak lensing convergence maps, and in [18] a DCGAN is trained using a dataset of N-body simulations to produce realistic Cosmic Web images. GANs have also been used to improve the image quality [5], generate portions of the sky [19], for image translation [20] and as a deblending tool [21]. In [5], a CGAN is trained to reduce blur of solar images, where a blur filter is applied to the solar image, and the generated image is compared to the original image using the Peak Signal Noise Ratio (PSNR) and structural similarity index measurement (SSIM). In [19], 21 GANs are combined to generate large images of the sky that resemble images from the Hubble Space Telescope eXtreme Deep Field (XDF), comparing the galaxies of real and fake images for evaluation. In [20], GANs are used to translate galaxy images from one survey to another in a two-way manner while preserving noise characteristics. They use global shape reconstruction and compare reconstructed Fourier maps of the noise for evaluation. And in [21], an SRGAN is used to separate overlapping galaxies, blending images from Galaxy Zoo Dataset and comparing the generated result with the ground truth. They use the Peak Signal Noise Ratio (PSNR) and the structural similarity index for evaluation.

Examples of works with the same objective (generate galaxy images) are [4], [7]–[9]. In [7] a DCGAN and a chained model with DCGAN and StackGAN are used to generate images at a resolution of 64 × 64 and 128 × 128 pixels. They analyze four properties (ellipticity, the angle of elevation relative to the horizontal of the galaxy's semi-major axis , total flux, and the size measurement of the semimajor axis on real and fake data to evaluate. In [9] a ProgressiveGAN is used to generate images using the Galaxy-Zoo dataset without utilizing a metric to evaluate results. [4] focuses on the deconvolution of galaxies' images using a CGAN whose input is a pair of images, the artificially degraded image, and the original image. For evaluation, Peak Signal Noise Ratio (PSNR) is measured. The previous works did not calculate the Fréchet inception distance (FID). [8] uses generative models to produce galaxy images in a 256 × 256 pixel resolution, also using a fine-tuning approach combining two datasets, producing realistic images of galaxies similar to the final dataset. They measure the half-light radius, magnitude, and color in flux space to compare the real and the generated images, obtaining images that follow the physical properties. This work also measures the Frechet Inception Distance (FID), obtaining the value of 19, however, it does not use the same approach as ours, and it can be used to compare the final results.

3 Generative modeling

The objective of this work is to generate astronomical images automatically. In machine learning, there is a type of model that can learn to generate new data that follow the real data distribution; these are called generative models. A generative model is a type of model that can generate new data instances that resemble the true data distribution. The objective is, given a training set X that follows an unknown distribution, learn and infer a model that estimates the true probability distribution $p(X)$.

3.1 GANs

Generative Adversarial Networks are deep generative models originally proposed to be used in an unsupervised fashion. In unsupervised learning, only features are given, and the goal is to extract patterns about the hidden or underlying structure that exists in the data.

The basic GAN consists of two neural networks, a discriminator and a generator. The generator is responsible for generating data and the discriminator receives the data and must discern where it comes from. The discriminator's output is the probability p of the data being real (coming from the training data), if p is greater than or equal to 0.5 the data is considered real and if p is less than 0.5 the data is considered fake. Meaning the discriminator wants its output to be closer or equal to 1 for training data and closer or equal to 0 for generated data. This probability serves as feedback for the generator, whose objective is to generate data that resembles the training data to confuse the discriminator. The discriminator's feedback is direct since it is known in advance where the data it receives comes from. Both networks improve during training due to their corresponding feedback, but since they have opposite objectives, they compete with each other in the form of a zero-sum game.

To rewrite both networks' objectives into a mathematical expression, we need expressions to define where the data comes from and to define the output from the discriminator D and the generator G. We define the training data as $\boldsymbol{x}$, $p_{data}(\boldsymbol{x})$ as the distribution of the training data, $p_z(\boldsymbol{z})$ as a prior of the distribution of the generator input data where $\boldsymbol{z}$ is random noise, $G(\boldsymbol{z})$ as a generated sample, $D(G(\boldsymbol{z}))$ as the output of the discriminator for a generated sample and $D(\boldsymbol{x})$ as the output of the discriminator for a sample coming from the training data.

If we take the perspective of the generator G, we want a generated image to be perceived as real by the discriminator D. We want the output of the discriminator D for a generated sample to be closer or equal to 1, meaning that we want to maximize the probability given by D (maximize $D\big(G(\boldsymbol{z})\big)$). Rewriting the idea as a minimization, we want $1 - D(G(\boldsymbol{z}))$ to be closer or equal to 0; however, we want this for every generated sample, meaning that each z that is an input for the generator (each z in $p_z(\boldsymbol{z})$) needs to follow this requirement. Using the expected value to generalize, the expression transforms into a minimization of $\mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[(1 - D(G(\boldsymbol{z})))]$.

If we take the perspective of the discriminator D, we want every real sample to be categorized as real and every generated sample to be categorized as fake. In other words, we want the output for every sample from the training data to be closer or equal to 1 (maximize $D(\boldsymbol{x})$) and the output of a sample from the generator to be closer or equal to 0 (minimize $D\big(G(\boldsymbol{z})\big)$). Expressing the second like a maximization problem, we can write the objective like a maximization of the sum of both expressions (maximize $D(\boldsymbol{x}) + \big(1 - D(G(\boldsymbol{z})\big)$). To generalize the expression to every sample, we use the expected values and the formula transforms to maximize $\mathbb{E}_{\boldsymbol{x} \sim p_{data}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$.

Having the formulas that describe the objectives of the generator and the discriminator, the GAN loss function is simply the combination of both, adding the logarithm of the calculations for simplicity.

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x} \sim p_{data}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]. \tag{3.1}$$

3.2 StyleGAN

In the StyleGAN model, the architecture of the generator is modified using the idea of style transfer to improve the quality of the generated images and reduce feature entanglement, a problem that consists in a disorganized latent space that causes the attributes of the latent vector to affect one feature of the output or more in an incomprehensible manner.

The StyleGAN architecture inherits its base from ProgressiveGAN, a GAN which uses progressive training, a method that increases the size of the generated images progressively during training by adding layers to the generator and discriminator ([3]), altering the generator and dividing it into two networks: a non-linear mapping network and a synthesis network. The mapping network receives the input and transforms it, and the synthesis network generates the image using the transformed input. The general architecture can be seen in Figure 3.1.

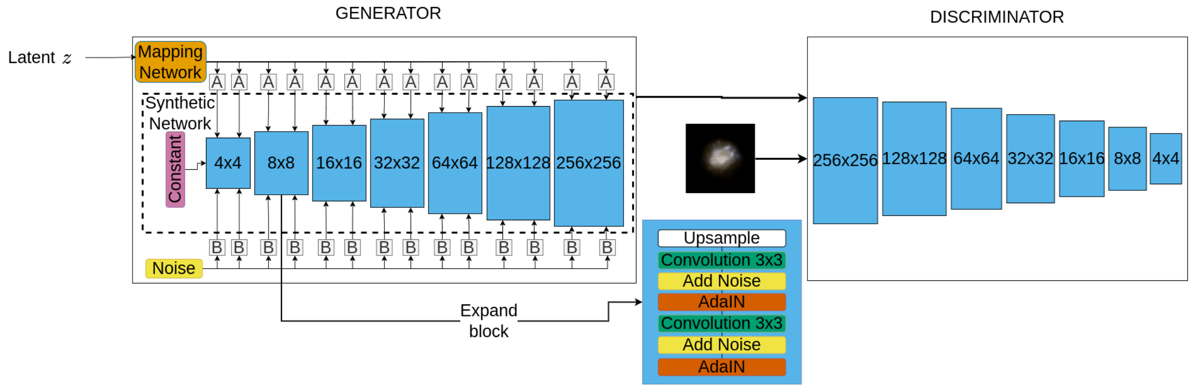

Figure 3.1: StyleGAN Architecture. The style-based generator consists of a mapping network in orange and a synthesis network in dashed lines. The mapping network is a perceptron of 8 layers. The synthesis network structure consists of "blocks" and each block receives two types of inputs: inputs from "A" and inputs from "B". "A" transforms the the output vector of the mapping network to a style vector and feeds it into the synthesis network, and "B" takes random Gaussian noise inputs and applies scaling factors per channel to it. The discriminator follows a mirrored structure of the generator, where each block has two convolutions and a downscale operation.

The mapping network transforms the input vector z from latent space Z to a new vector w in an intermediate latent space W where the features are less entangled. This network is an 8-layer fully connected perceptron. The dimensionality of input z and output w is 512 , and leaky ReLU is used.

Instead of having an input layer, the synthesis network starts with a learned constant, following the idea of having a base of the object because the network always generates images of the same object, and its structure consists of "blocks". Each block of the synthesis network consists of an upsampling layer and then two convolutional layers, and after each convolution, two operations are performed: noise addition and Adaptive Instance Normalization (AdaIN). The added noise consists of single-channel images of uncorrelated Gaussian noise scaled with learned per-channel scaling factors. The noise affects all feature maps and then is added to the output of the convolution layer. This is delivered in order for the generator to be able to generate stochastic details. The first and last blocks are different from the others; the first block removes the upsampling layer and replaces the first convolutional layer with the learned constant. The last block takes the previous block's output and converts it to RGB using a separate 1 × 1 convolution. The blocks can be seen in the synthesis network (light blue boxes) in Figure 3.1. The number of blocks depends on the output resolution, and these are progressively added through training. The discriminator mirrors the synthesis network's architecture only considering the upsampling operations and the convolutional layers, and both grow in synchrony.

The mapping network communicates with the synthesis network two times per block. For each of these connections, the latent code $\boldsymbol{w}$ is transformed through a learned affine transformation to a vector $\boldsymbol{y} = (\boldsymbol{y}_s, \boldsymbol{y}_b)$ that defines a particular style of the output image. AdaIN consists of two parts: Instance Normalization (IN) and an "Adaptive" operation. Instance Normalization is a normalization process by channel considering only one instance $\boldsymbol{x}$. The "Adaptive" operation uses the vector $\boldsymbol{y}$ to scale and shift the normalized values; this operation is the one that injects the styles of the input vector on the image.

$$AdaIN(\boldsymbol{x_i}, \boldsymbol{y}) = \boldsymbol{y}_{s,i} \frac{\boldsymbol{x}_i - \mu(\boldsymbol{x}_i)}{\sigma(\boldsymbol{x}_i)} + \boldsymbol{y}_{b,i} \tag{3.2}$$

Where $\boldsymbol{x_i}$ is the i-th feature map of an instance $\boldsymbol{x}$, $\mu(\boldsymbol{x}_i)$ and $\sigma(\boldsymbol{x}_i)$ are its mean and standard deviation respectively, and $y_{s,i}$ and $y_{b,i}$ are the corresponding scaling factors from y_s and y_b in the $i - th$ position.

3.2.1 Training

The structure of "blocks" is necessary due to the progressive training method used, presented in [3]. Progressive training starts by generating low-resolution images and increasing the resolution during training. This method allows learning the basic structure of the distribution of images and then, with higher resolutions, learning to generate more specific details, not affecting the knowledge obtained in lower resolutions. In order for the input (discriminator) and output (generator) resolutions to be modified during training, their structure is based on blocks. When a block is added to each network, the output resolution of the generator is doubled due to upsampling operations and the input resolution of the discriminator is doubled due to convolution operations. The starting resolution for the training is 8×8 pixels.

3.2.2 Loss function

The loss function is used to measure and minimize the distance between the distributions from both fake and real data. Previously we analyzed the traditional GAN loss; however, in this work, we utilize another function: the Wasserstein Gan with Gradient Penalty loss. We decide to use this loss function because it produces better results for the dataset we use in the experiment on the original domain (this is stated in the original StyleGAN paper).

3.2.2.1 Wasserstein Gan with Gradient Penalty

Wasserstein Gan with Gradient Penalty (WGAN-GP) loss is used in the network. This loss is a modification of the Wasserstein loss (WGAN). The Wasserstein loss is based on the Wasserstein distance, a measure of the distance between two probability distributions, and is equivalent to the minimum cost to transform a probability distribution $P(X)$ to a probability distribution $Q(X)$. The WGAN changes the discriminator's behavior: the sigmoid activation of the last layer is removed, no longer constraining the value of predictions between 0 and 1, so instead of a probability, for each instance the discriminator outputs a real number. Due to these modifications, the discriminator is called a "critic" whose objective is to maximize the difference between real images and generated images. This "critic" must be a 1-Lipschitz function to ensure the function is continuous and differentiable, this is achieved by restraining its weights within a certain range.

In this loss the critic D aims at distinguishing between real and fake samples, maximizing the difference between the outputs for real and fake instances, and the generator G aims at the opposite, minimizing that difference.

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})}[D(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[D(G(\boldsymbol{z}))]. \tag{3.3}$$

The WGAN-GP loss replaces the restriction of the critic's weights with a gradient penalty to enforce the Lipschitz constraint: the norm of given gradients must be at most 1. This penalty is applied to random samples $\hat{\boldsymbol{x}} \sim \mathbb{P}_{\hat{\boldsymbol{x}}}$, where $\mathbb{P}_{\hat{\boldsymbol{x}}}$ is the distribution obtained by uniformly sampling along a straight line between the real $(p_{data}(\boldsymbol{x}))$ and generated distributions $(p_z(\boldsymbol{z}))$. Basically, the objective of both networks remain the same as Wasserstein loss and the penalty is applied to a randomly weighted average between a real and generated sample. The loss equation of the critic is :

$$\min_D \max_G V(D,G) = \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[D(G(\boldsymbol{z})))] - \mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})}[D(\boldsymbol{x})] + \lambda\,\mathbb{E}_{\hat{\boldsymbol{x}}\sim\mathbb{P}_{\hat{\boldsymbol{x}}}}[(\|\nabla_{\hat{\boldsymbol{x}}}\boldsymbol{D}(\hat{\boldsymbol{x}})\|_2 - 1)^2] \tag{3.4}$$

The loss function gives information about the state of the zero-sum game of the generator and the discriminator, however, the Wassertein loss with gradient penalty correlates with the sample quality [22], giving an insight of how well the model is learning.

3.2.3 Mixing Regularization

Mixing regularization is a method used during the training that prevents the network from assuming a correlation between adjacent styles and enhances style localization. It consists of having two inputs to generate one image instead of only one and switching between them at a random moment. Consider inputs z_1 and z_2, with their corresponding mapped latent vectors w_1 and w_2. During training, the network will generate an image using the first vector, but from a random point onwards, the network will start to use the second vector as input to generate the image. If we have two inputs that are very different from each other, if the network assumes correlation between styles, when we use the second input vector, the styles will try to modify parts of the image that already should have been filled by the previous style vector, this could lead to an unrealistic output, resulting in a bad performance for the generator and therefore, will try to just affect one specific feature of an image given a style.

3.3 Evaluation Metrics

3.3.1 Fréchet Inception Distance

The Fréchet Inception Distance (FID) is a metric that models the distribution of the features from the real and fake data separately as multivariate Gaussian distributions and then calculates the distance between the two distributions. The features are extracted from an intermediate layer of the Inception V3 classifier [23]. Considering $\mu_r, \sum_r$ and $\mu_g, \sum_g$ as the means and covariances of the feature distribution of the real data and the fake data, respectively, the FID is calculated by:

$$FID(r, g) = \|\mu_r - \mu_g\|_2^2 + Tr(\Sigma_x + \Sigma_g - 2(\Sigma_r \Sigma_g)^{\frac{1}{2}}), \tag{3.5}$$

where $Tr(M)$ defines the sum of all the elements in the diagonal of the matrix M.

3.3.2 Magnitude

The flux of an object its the total detected energy in a specific area in a given time. Using the flux of two sources we can calculate the magnitude of an object. The magnitude is a relative measure of brightness, where higher values indicate fainter objects and lower values indicate brighter objects. Due to the value depending on the flux of two sources, the magnitude of an object is variable (for example, one object A can have two different magnitude values if one measures the magnitude of the object A with respect to a source B and then respect to another source C), so for the values to be comparable, a fix object is used to calculate the magnitudes, the object used depends of the magnitude scale. By filtering the light, we can calculate the flux in a specific filter, and because the flux is used to calculate the magnitude, we can calculate the magnitude in a specific filter. Examples of filters are: ultraviolet (u), green (g), red (r), near infrared (i) and infrared (z).

3.3.3 Color-magnitude diagram

This diagram relates two physical properties of a galaxy: flux(magnitude) and color. The absolute magnitude gives a measure of the luminosity of a galaxy, while the color is correlated to its star-formation rate, metallicity, stellar age and structure [24]. Galaxies have a bimodal color distribution which can be approximated with a double Gaussian. Higher values of the color mean that a galaxy is red, while low values of the color mean that a galaxy is blue. The bimodality translates itself into a red sequence and a blue cloud when one looks at the color-magnitude diagram. Dense structures such as clusters of galaxies have a tight and prominent red sequence and a sparsely populated blue cloud. As sparser cosmic structures are considered, the blue cloud becomes more prominent at the expense of the red sequence. This will be used to corroborate whether the data generated at the end is realistic or not, comparing these two properties between the real data and the generated samples. To obtain the brightness of a galaxy, we need to calculate the magnitude of any of the bands (u, g, r, i, or z); for this work, we will focus on the r-band for brightness. The color is defined as the difference between the magnitudes measured in two filters. This work will be focused on the $g - r$ color. Sets of galaxies that follow the same properties will be plotted in the same area on the diagram.

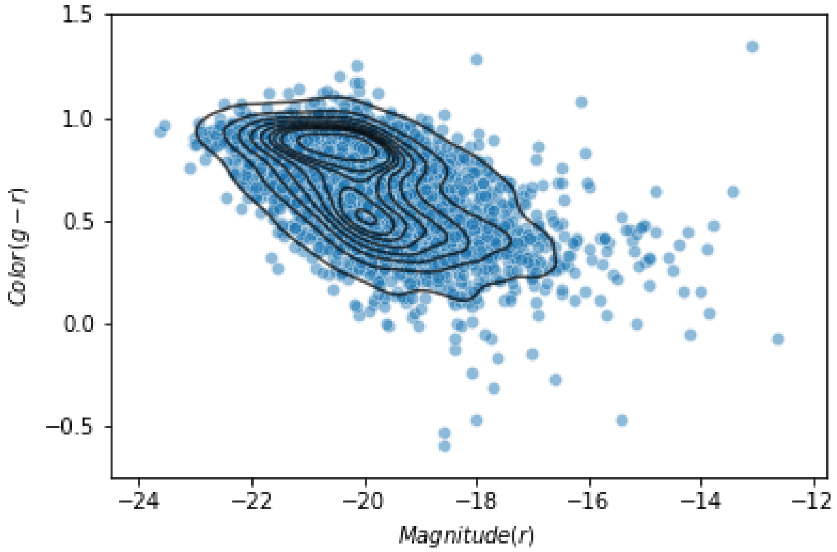

Figure 3.2: Example of a color-magnitude diagram. This diagram was made with 5000 galaxies extracted from the Sloan Digital Sky Survey. The r-band magnitude is reported in the x axis, while the $g - r$ color is in the y axis. The blue circles represent the galaxies, while the black contour highlight the red sequence and the blue cloud.

4 Experiments and Results

As the work's objective is to generate astronomical images of galaxies, galaxy datasets had to be found so that the network could learn the characteristics from the images. Because the learning is done directly from the images, a dataset with poor variability or few data means our model cannot learn correctly, leading to poor results. Two galaxy image datasets were used to perform the experiments. Something essential to indicate is that, as we want to create realistic and detailed galaxies, using real images is ideal. However, the real images collected that meet the necessary size are scarce, we collected the bright nearest galaxies on the SDSS resulting in less than 200 real images. Working with only the real images dataset could lead to not promising results: less data makes it more likely that the network will memorize and replicate the training data or generate identical samples or samples that follow the same properties with little variations. To avoid this, it was decided to use the fine-tuning technique that would allow training first with a large dataset with less detailed images (less realistic) and then fine-tune the network using the dataset with the real images. We added a new dataset consisting of simulated images to use this technique.

As the fine-tuning technique had not been used with the original paper's network, it was necessary to carry out a previous experiment with the original domain's images to evaluate their behavior using this technique. Thus, the work presents two experiments: fine-tuning with the network in the original domain, human faces, and fine-tuning with the network with galaxy images in JPG and FITS format, respectively. The first experiment was executed on 1 NVIDIA Tesla K80, and the second was executed utilizing NVIDIA A100 (we specify the number of GPUs used in the description of each experiment.) Both experiments utilize the same configuration as the original StyleGAN except when we specify it otherwise.

4.1 Data Description

We needed two different type of images: human faces and galaxies. For the human faces, we collected two datasets containing images in JPG format, and for the galaxy dataset, we collected two datasets containng images in JPG and FITS format. Each dataset is briefly described below:

- Celeba-HQ: High-quality version of the CelebA dataset, that is a large-scale face attributes dataset with celebrity images, each with 40 attribute annotations. It contains 30000 PNG images at 1024x1024 resolution. Figure 4.1 contains sample images.
- SCUT-FBP5500: A dataset that contains 5500 frontal faces of Asian and Caucasian people (both female and male) at a 256×256 resolution and labels of face landmarks and beauty scores. Figure 4.2 contains sample images.
- Illustris Galaxy Dataset: Dataset containing 6897 images of simulated galaxies. The images were extracted from the Galaxy Image Dataset of the Illustris Galaxy Observatory [25]. For each simulated galaxy we collected the FITS file at a 256×256 resolution and the generated JPG image by the algorithm friends-of-friends (FoF) at a resolution of 254×254 and scaled it to 256×256. All images were collected through the official web page's API, obtaining 6897 samples that were rotated in degrees of (45, 90, 135, 180, 225, 270, 315), obtaining a total of 55176 images of each format. Figure 4.3 contains sample images.
- SDSS Galaxy Dataset: Dataset containing 149 images of real galaxies. We use galaxy images from the Sloan Digital Sky Survey (SDSS) ([26] [27] [28]) Data Release 14 (DR14) ([29]). The dataset have a sample of galaxies from the nearby universe, with large angular size (6-12 arc minutes). The JPG images had a resolution of 0.396127 arcsec/pix, and a size of 512×512 pixels that were scaled to a size of 256×256 pixels, the FITS files were centered on the galaxy and cropped to a size of 256×256 pixels. Every sample was rotated in degrees of (45 , 90, 135, 180, 225, 270, 315), obtaining a total of 1192 images of each format. Figure 4.4 contains sample images.

Figure 4.1: Images from Celeba-HQ dataset

Figure 4.2: Images from SCUT-FBP5500 dataset

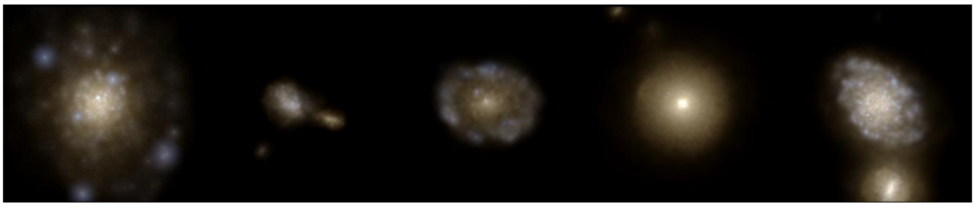

Figure 4.3: Images from Illustris Galaxy Dataset

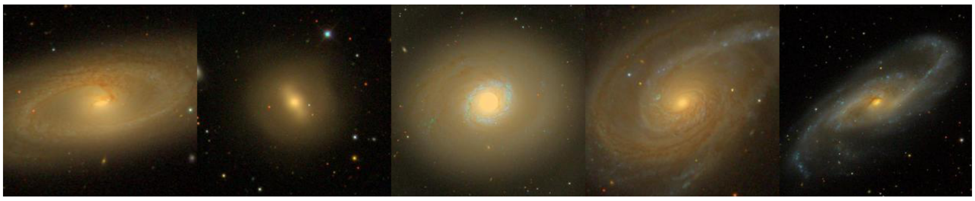

Figure 4.4: Images from SDSS Galaxy Dataset

4.2 Experiment 1: Face Image Generation

This experiment consisted of training the StyleGAN using the Celeba-HQ dataset and then fine-tune the network using only the images from Asian Females of the SCUT-FBP5500 dataset to evaluate the performance of the StyleGAN and to evaluate its behavior using fine-tuning. The number of processed images was arbitrarily decided and it was set to 12.2M for the experiment: the first training was set to process 11M images and the second training was set to process 1.2M images.

The first part of the experiment was performed using the Celeba-HQ dataset. Setting the network to generate images in a resolution of 256 × 256 pixels (with the configuration specified for Celeba-HQ mentioned in [10]) while monitoring the changes in the FID. This experiment lasted 33 days and 20 hours and the metric on the training is not registered in the original StyleGAN paper, which only contains the final FID score (5.06) of the StyleGAN trained with the dataset at 1024 × 1024 resolution.

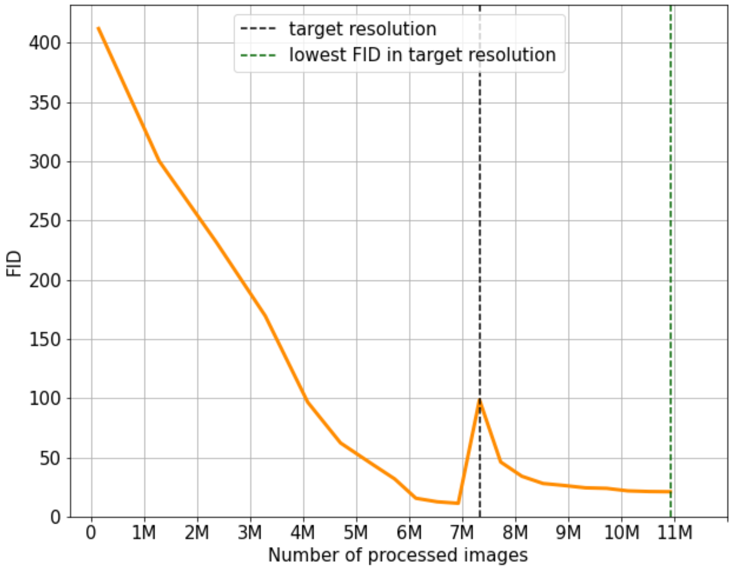

Figure 4.5: FID during the training with Celeba-HQ. FID scores registered in the face image generation experiment during training with Celeba-HQ dataset, where the x axis represents denotes the number of training images processed by the discriminator.

Figure 4.5 shows the FID on the training process in terms of number of iterations, and it shows that the value decreases during the training, meaning that the network's output images improve at each resolution. The network producing images at the desired resolution that obtains the lowest FID must be used to obtain the best images. The target resolution was reached after 7M of processed images, and the lowest FID after that was of 21.0986, obtained after processing 10.9M images. This FID corresponded to the network state at the end of the training and this network was the one used for fine-tuning. Figure 4.6 contains examples of generated images with the full-trained network (last network) obtained with the experiment.

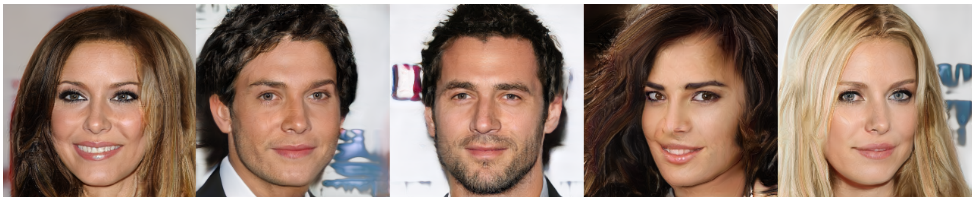

Figure 4.6: Generated images of best network after training with Celeba-HQ dataset

To demonstrate that the generated images were not exact copies of the dataset's images, 5 random images generated by the network with the lowest FID were compared to all the Celeba-HQ dataset images. In Figure 4.7 the images generated are shown together with the 4 most similar images of the dataset using Mean Standard Error (MSE). To only take into account the generated faces (without the background), the comparison was made cropping the images, extracting a 38-pixel border, assuming for simplicity that the face was at the center of the image.

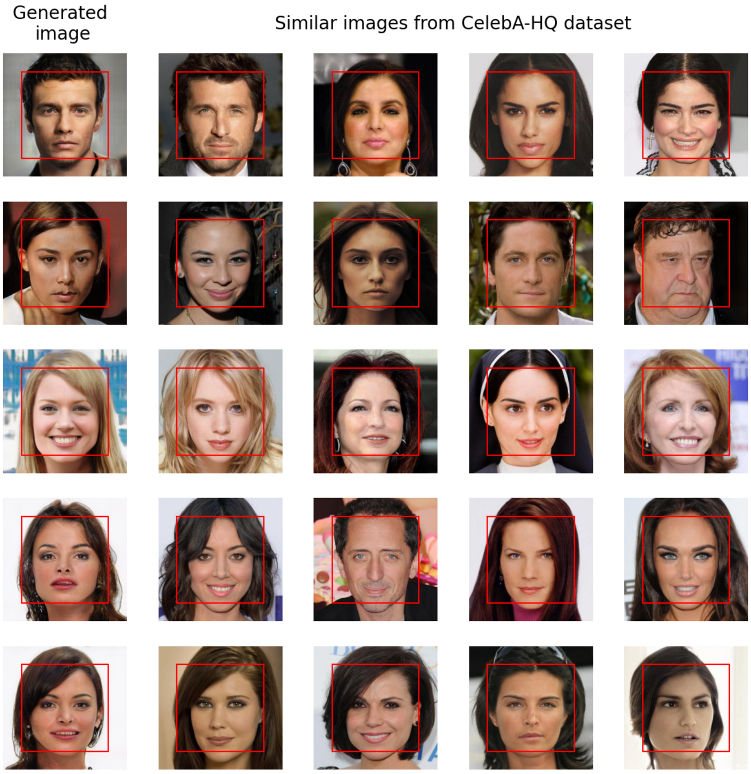

Figure 4.7: Most similar images in Celeba-HQ using MSE. The red square indicates the pixels considered for the MSE comparison.

The second part of the experiment was to fine-tune the pre-trained network with the images of Asian Females from the SCUT-FBP5500 dataset at a resolution of 256 × 256. In this part of the experiment, the network does not add new layers to its generator (due to having reached the final resolution) and only modifies the existing layers' weights to produce better images. This experiment lasted approximately 6 days and 1 hour. It is observed in Figure 4.8 that the generated images are far from perfect, some faces look distorted or seem very unrealistic, but it is seen that the GAN adapted to the characteristics of the new dataset. The generated images present white borders, monochromatic backgrounds, and specific features from Asian Females (the shape of the eyes, skin color, and hair), making them more similar to the new dataset than the images from Celeba-HQ. The GAN adapting to a different dataset quickly without the FID increasing exponentially could be explained through the use of *styles.* Using this architecture, each image feature should be associated with a *style*, this allows to change features, like the background or hair color, without changing/affecting other output features.

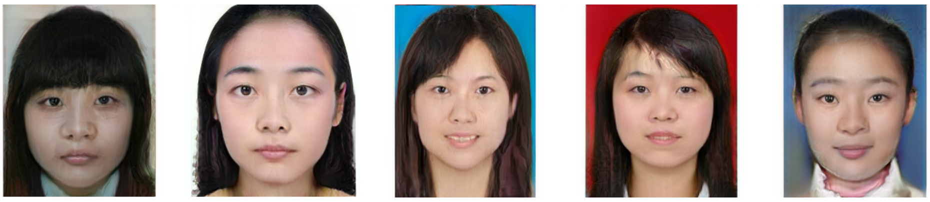

Figure 4.8: Generated images of best network after training with SCUT-FBP5500 dataset

Figure 4.9 contains the FID scores during the training with the SCUT-FBP5500 data. It is observed that the new network did not score a lower FID in the experiment, reaching 30.2571 at the end, but observing the loss curve, it could be possible that the FID could reach a lower value by increasing the training time or by decreasing the learning rate.

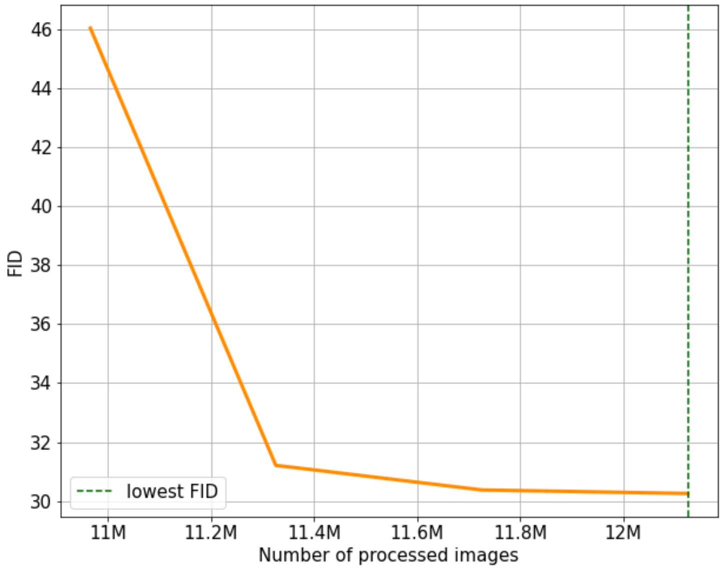

Figure 4.9: FID during the training with SCUT-FBP5500. FID registered in the face image generation experiment during the training with the SCUT-FBP5500 dataset, where the x axis represents the number of training images processed by the discriminator. The first FID score is calculated with the network resulted from the training with Celeba-HQ (no extra training or modifications). However, even if we use the same network, the FID score is not the same as the one previously obtained because the dataset is different. It can be seen that the value decreases during the experiment, implying that there is an improvement in the generator and in the resulting images.

4.3 Experiment 2: Astronomical Image Generation

The SDSS Galaxy Dataset contains less than 200 images, so in order to generate realistic images and not end in mode collapse, a problem where the model produces a single kind or a limited variety of samples, or generating the exact same images of the dataset, we start by training the model with the Illustris Galaxy Dataset (46 times larger than SDSS) for the network to learn the basic structure of galaxies and reach the desired resolution (256×256 pixels), and then we fine-tune the network with the real data.

Our final objective is being able to generate FITS images that look realistic and resembles both the morphological and physical properties (flux and color) of the real data (SDSS images), obtaining networks that are able to produce realistic JPG galaxy images in the process. We start training with JPG data as a way of evaluating the behavior of the network in the new domain, and then continue training with FITS data. Because of this, we segment the experiments by the training data:

- Training with JPG data:
  - Training with Illustris JPG: We train the network with the larger dataset (Illustris) in JPG format arbitrarily until 12M of processed images, reaching a one-digit FID value.
  - Training with SDSS JPG: We fine-tune the snapshot that reached the best FID on the previous experiment with the SDSS data in JPG format, freezing between 0 and 5 layers of the generator.
- Training with FITS data
  - Training with Illustris FITS: We fine-tune the best network of each previous experiments using Illustris FITS data.
  - Training with SDSS FITS: We fine-tune the snapshot that reached the best FID on the previous experiment with the SDSS data in FITS format without freezing.

Each experiment is explained in detail in the next paragraphs.

4.3.1 Training with JPG data

The Illustris JPG data was rotated for data augmentation (angles detailed in section 4.1), filling all values beyond the edge with the same constant value (a 0 value). This method does not interfere with the majority of images due to it containing a black background (approximately a 7% of the images contains some object or light in the corners); however, it does not work well with images with galaxies with high surface brightness, this can be seen in Figure 4.10. Because of this, the generator will produce in occasions, bright surface galaxies with abnormal black spaces on the corners. The SDSS images always contain some object in the background, so we take a different approach: the empty spaces of the rotation were filled by reflecting about the edge of the last pixel of the image. This method is not perfect as can be see in Figure 4.11 where in the corners a light without a source is visible, but it does not produce a such strong difference between the original images and the added space as the method by filling with a constant.

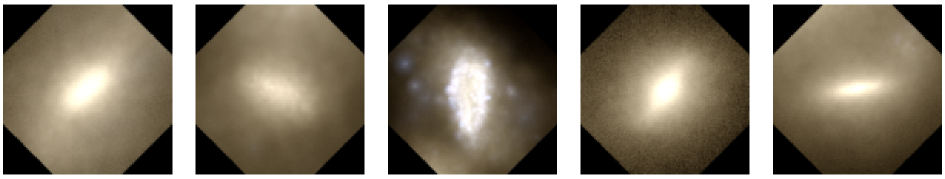

Figure 4.10: Problem with Illustris rotated images. This is produced by filling the spaces created by the rotation with a constant value.

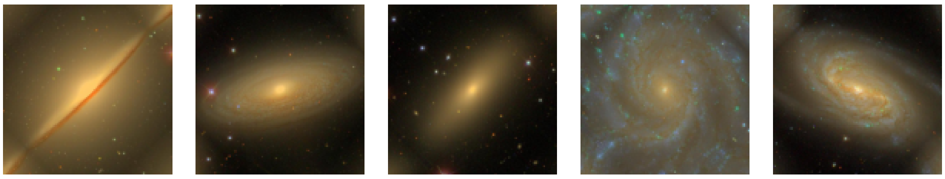

Figure 4.11: Problem with SDSS rotated images. This is produced by filling the spaces created by the rotation by reflecting about the edge of the last pixel of the image. A light without an apparent source can be seen in the images.

4.3.1.1 Training with Illustris JPG

We start by training the network using the Illustris dataset with an arbitrary number of 12M of processed images using the WGAN-GP loss. Figure 4.12 shows the FID score in terms of the number of training iterations, the first experiment (blue curve) was conducted and stopped after 6M of images due to the FID increasing drastically its value. It was suspected that this behaviour could be produced by the change in the resolution of the output; however, checking the resulting images (Figure 4.13) we concluded the network was affected by mode collapse. A second experiment (orange curve) was done following the same conditions to check if it was a phenomenon that occurred randomly and if it could be avoided by redoing the training. This attempt followed the same pattern; however, this time the experiment was not stopped and the FID started to decrease after 7.3M iterations, reaching its lowest value, 6.06, after processing 11.5M images approximately. The mode collapse phase of the network can be outcome due to the loss function: if the generator produces repeated outputs, the discriminator would label them as "fake", forcing the generator to produce new samples to obtain a better score.. Examples of generated images by the best network can be seen in Figure 4.14.

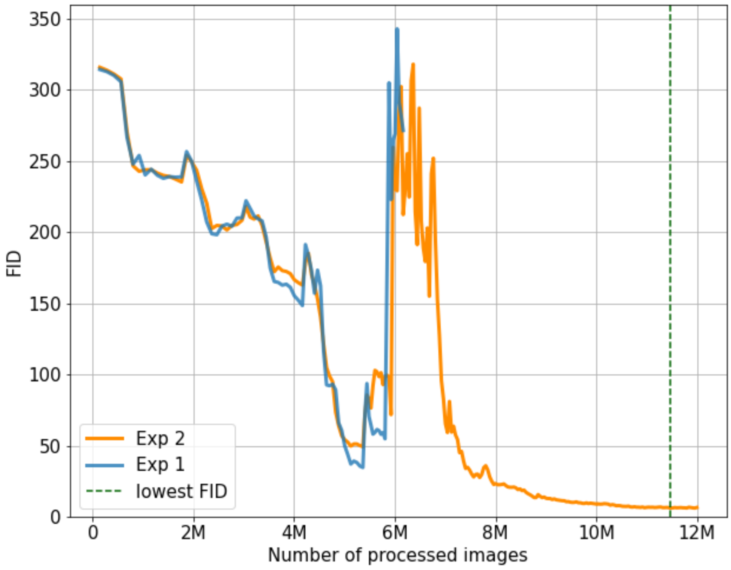

Figure 4.12: FID during JPG Illustris Training. We start by training the model with the larger dataset in JPG format. We decide to do the experiment again due to the first attempt falling into mode collapse, resulting in the same pattern, however after some training time, the metric decreases and reaches a one digit value. The lowest value obtained was 6.0674

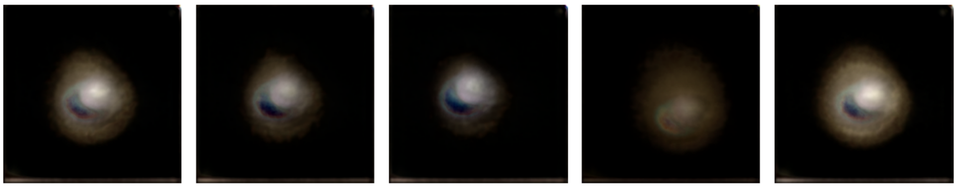

Figure 4.13: Mode collapse example. Images generated by the first experiment before it was stopped.

To verify that the generated images were not exact copies of the original dataset, we generated five random images and searched for the four most similar images of the original dataset. We use the mean square error metric (MSE) to calculate the difference between two images. This experiment can be seen in Figure 4.15. Visually one can compare the images and conclude the generated data is not an exact copy. This is corroborated by

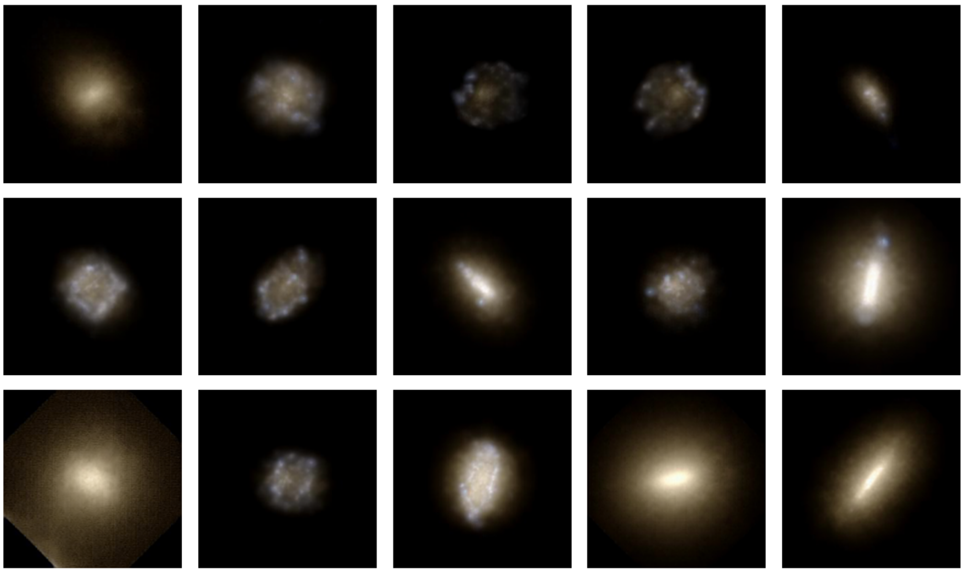

Figure 4.14: Generated images with model trained with Illustris JPG. Randomly-selected examples of generated galaxies using the best performing model

observing the calculated MSE, since it never reached a value of zero. The training time was: 14 hours and 31 hours for the first and second experiment, respectively, and both were executed on 2 NVIDIA A100 GPUs.

4.3.1.2 Training with SDSS JPG

Reaching a low value on the FID score (one-digit value) on the previous experiment and observing the generated images, we can state the model learned the basic structure of a galaxy image (black background with an illuminated shape in the center), and we continue by fine-tuning the network using the SDSS data in JPG format. With real data, we expect the model to add variation in the shape of the galaxies, the light intensity, the colors, and add background elements. Due to the data being smaller in quantity, as an attempt to avoid mode collapse and replicating the original data, we only desire to train for a maximum of half the processed images of the previous experiment (this means reaching 17M iterations in total). In addition, we try freezing layers of the generator as an attempt to improve results without increasing the training time.

Figure 4.16 shows the FID score in terms of the number of training iterations, where each curve is a different training fine-tuned using the best network from the previous experiment

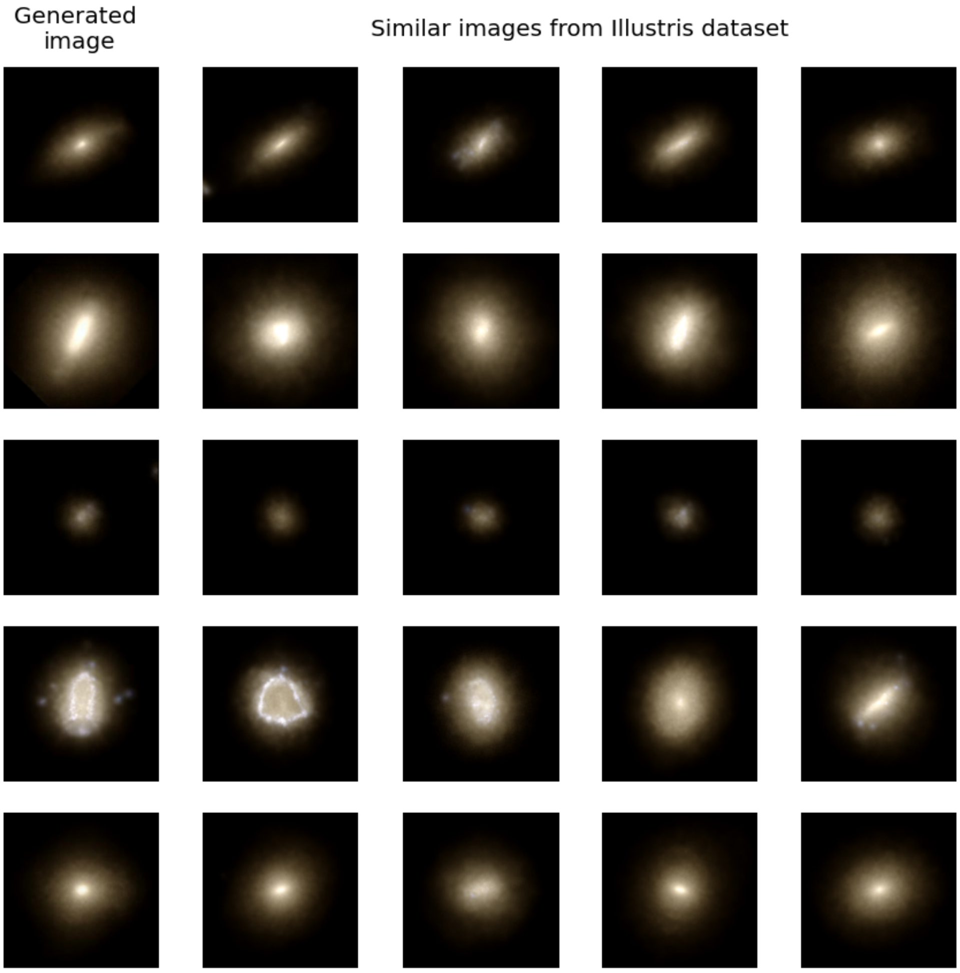

Figure 4.15: Most similar images in Illustris using MSE. The first image of each row is a generated image from a random input vector. The generated image is compared with all the images in the Illustris dataset. The difference between the images is calculated using MSE (We add the square differences pixel by pixel and divide the total by the area of the image). The next 4 images of every row correspond to the 4 lowest values obtained.

with a fixed amount of frozen layers. All the curves show a similar decreasing behaviour, meaning the network is adapting to the new data; however, even if the network adapts, the FID value does not reach a one-digit value like the previous experiment. An interesting behavior can be seen in the FID curve: it was expected that increasing the freezing of layers would only worsen the metrics from a certain point onward but that is not the case. Freezing three layers (red curve) produces the most distant (and worse) results compared to experiments where we froze fewer layers, so it was expected that freezing four layers would perform similarly or worse. Still, freezing four layers produces better results than freezing two or three. This could be a result of freezing in a layer level instead of freezing in a block level (a block consists of 2 convolution layers). The lowest FID (70.8283) was reached by freezing one layer of the generator after 16.6M processed images and Figure 4.17 contains examples of generated images by this network.

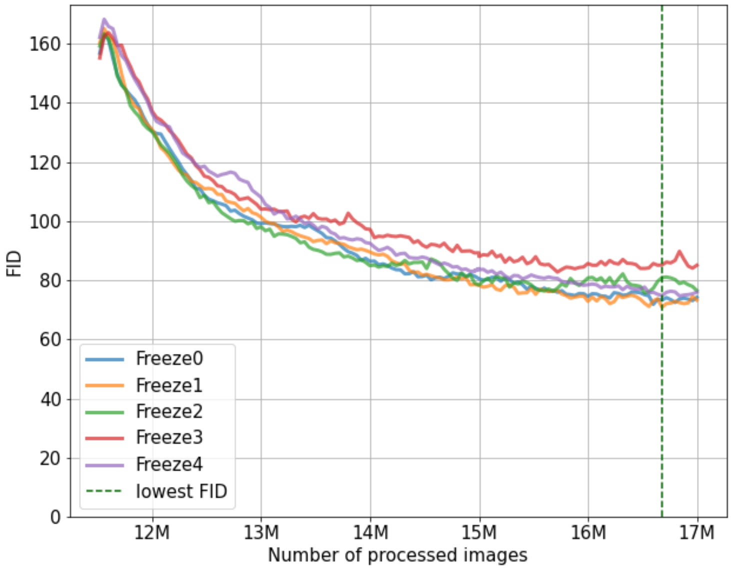

Figure 4.16: FID during JPG SDSS Training. FreezeX indicates the number of frozen layers, freezing the weights and bias. The best network had two frozen layers, reaching an FID of 70.8283

Figure 4.18 shows 5 randomly generated images and the four most similar images of the training data using MSE; it is observed that the generated and the most similar image are identical, indicating that the generator is already starting to imitate the training data. The generated images are not exact copies yet, because the value of the MSE never reaches 0. The decreasing tendency on the FID curve could mean that the model could improve the quality of the images, but observing the generated images, this would be produced by continuing to reproduce the training data. The training time for each experiment was approximately 38 hours, resulting in a total of 190 hours, each executed on 2 NVIDIA A100 GPUs.

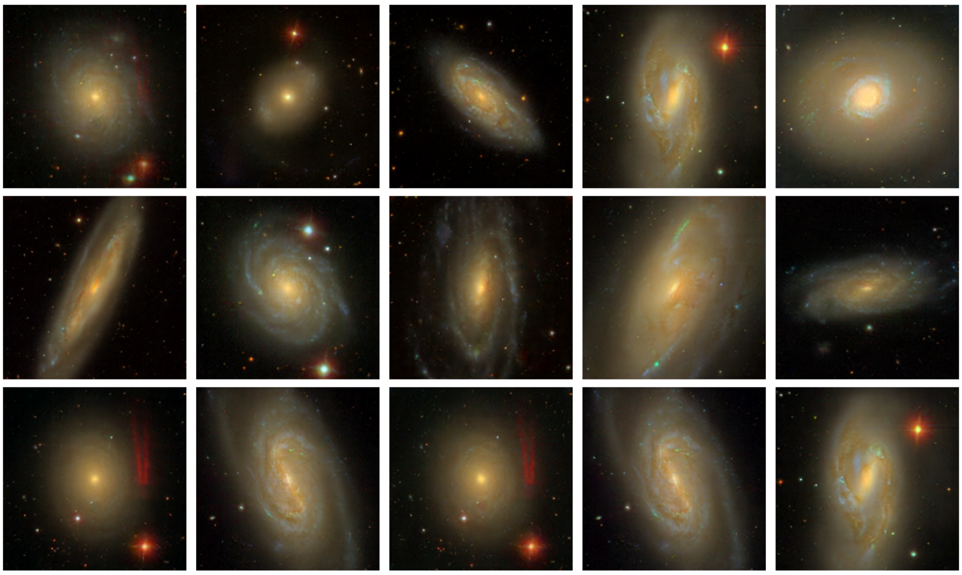

Figure 4.17: Generated images with model trained with SDSS JPG. Randomly-selected examples of generated galaxies using the best performing model

4.3.2 Training with FITS data

The SDSS FITS images are larger than the required format, so they were cropped to generate an image size of 256×256 pixels (containing the galaxy in the center). Afterwards, we apply a min-max normalization per channel to all FITS images, Illustris and SDSS, to balance the difference in brightness. For data augmentation, we use the same method as the one described for the SDSS JPG images for both datasets. After this process, we do not do anymore pre-processing before feeding the data to the network.

The FID metric does not support FITS images, so to evaluate the images, the training data and also the generated images must be transformed to RGB images. Because we train the model with normalized images, the outputs will be also normalized, so first the image is de-normalized and then a transformation is applied using the sunpy library [25]. This same processed is utilized to generate the example images.

4.3.2.1 Training with Illustris FITS

For this part, the training data format changes from JPG to FITS, which we consider a significant change. Keep in mind that the structure of the StyleGAN is in blocks, where the first ones learn to synthesize the base of the image and the last layers incorporate the

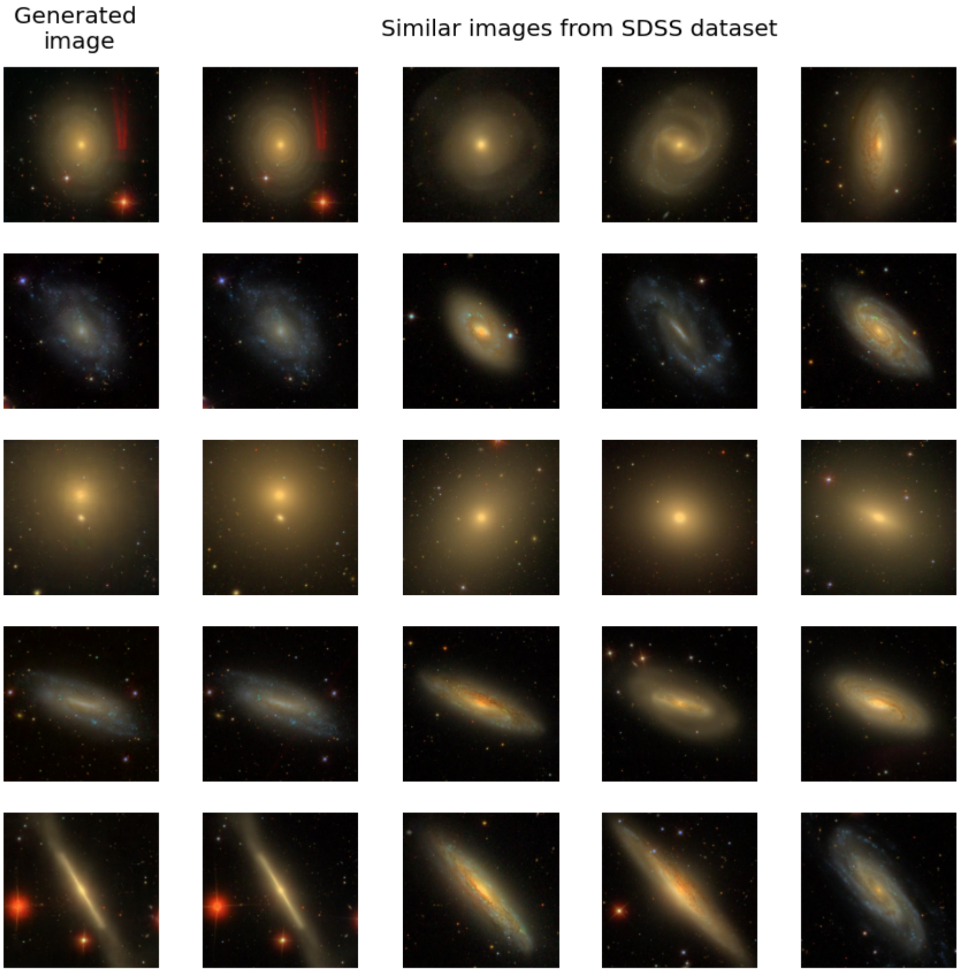

Figure 4.18: Most similar images in SDSS data using MSE. The first image of each row is a generated image from a random input vector. The generated image is compared with all the images in the SDSS dataset. The difference between the images is calculated using MSE (We add the square differences pixel by pixel and divide the total by the area of the image). The next 4 images of every row correspond to the 4 lowest values obtained.

details, and because the format changes, it is assumed that the first layers need to learn and adapt to the new data. Due to this, we do not try to freeze the generator, allowing all the layers to learn. Figure 4.19 shows the FID behaviour during training, we show two experiments, each one starting from the best network of the previous experiments, starting from the Illustris JPG in blue and starting from SDSS JPG in orange. We fine tune both models using Illustris FITS data for the same amount of time (Approximately 9M of processed images). In both networks it is observed that the model is learning because the FID decreases, however, it is not a constant decrease like in the previous experiments. The lowest FID value (122.0133) was reached by the network fine-tuned after Illustris JPG training. Examples of generated images by this network can be seen in Figure 4.20. The experiment fine-tuned from Illustris JPG data decreases the FID constantly until a certain point, and since then the metric is unstable, even if it reaches the lowest value, the difference between this value and the lowest reached until 13.2M iterations (135.0847) is small if we take into consideration the training time. The training time for each experiment with fine-tuning was approximately 46 hours, resulting in a total of 92 hours. Each experiment was executed on 2 NVIDIA A100 GPUs.

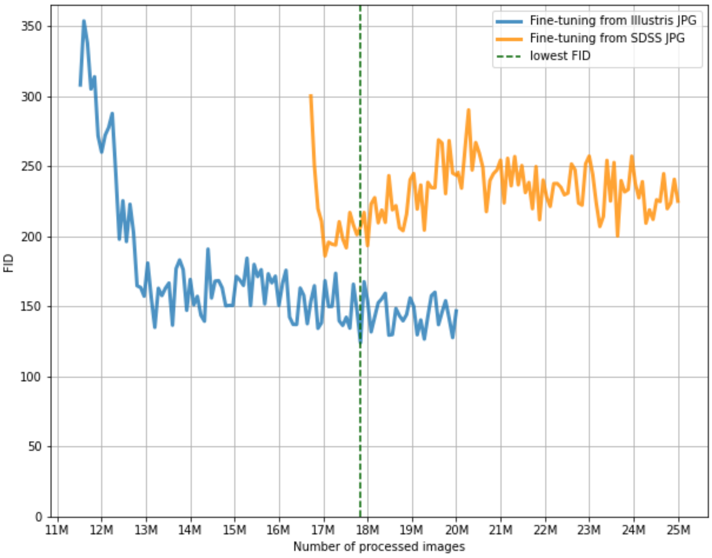

Figure 4.19: FID during FITS Illustris Training. Both networks obtained after the first and second experiments were trained for the same number of iterations (9M). In the x axis are the number of total processed images, and due to the experiments obtaining better results at different points, the starting point for the fine-tuning is different. The lowest FID was obtained using the network from the Illustris JPG training.

4.3.2.2 Training with SDSS FITS

The last experiment consists on fine-tuning the best networks of each previous experiments with SDSS FITS data. Due to the model almost reproducing the training data in the experiment with SDSS JPG, we trained the model with the real images until 23M processed images arbitrarily, freezing layers of the generator as to try to decrease the FID without

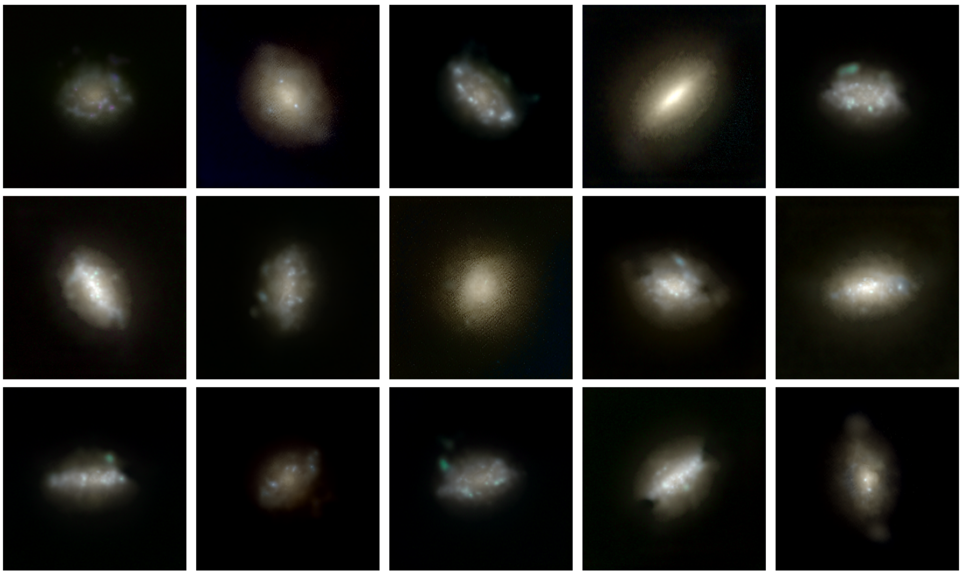

Figure 4.20: Generated images with model trained with Illustris FITS. Randomly-selected examples of generated galaxies using the best performing model

extending the training time, reaching an FID of 148.49110. The FID of all models shows an erratic behaviour, not showing tendencies to decrease, except for the model with two frozen layers; however we tried to continue the training of this model until 25M processed images and the FID did not reach a lower value neither showed a decreasing tendency. Examples of generated images using the final network are in Figure 4.22. It is observed that the network adapts to the new dataset, presenting new colors in the images, background elements and varying galaxies shapes; however, it is not enough to achieve a low FID, due to the generated images presenting noisy and blurry characteristics and also due to producing only noise in some cases.

For the resulting images, we need to corroborate that the outputs are not copies of the training data, an issue when training with SDSS in JPG format. Even if the FID is high, the network could be trying to replicate the training images, but failing at reproducing the exact same images due to the noisy elements and blurriness. In Figure 4.23, which shows 5 randomly generated images and the four most similar images of the training data using MSE, it can be observed that the generated image and the most similar image are distinguishable for the human eye, indicating the network is generating new images that do not exist on the training data.

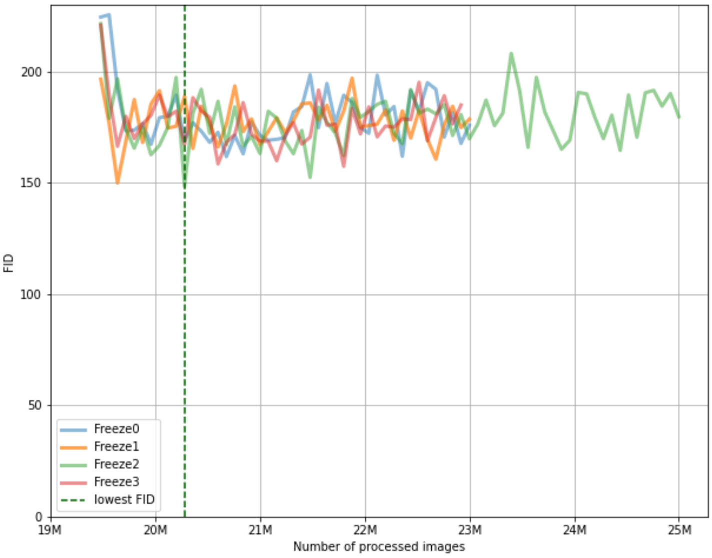

Figure 4.21: FID during FITS Sdss Training. FreezeX indicates the number of frozen layers, freezing the weights and bias. The best network was the one with only one frozen layer, reaching an FID of 148.49110

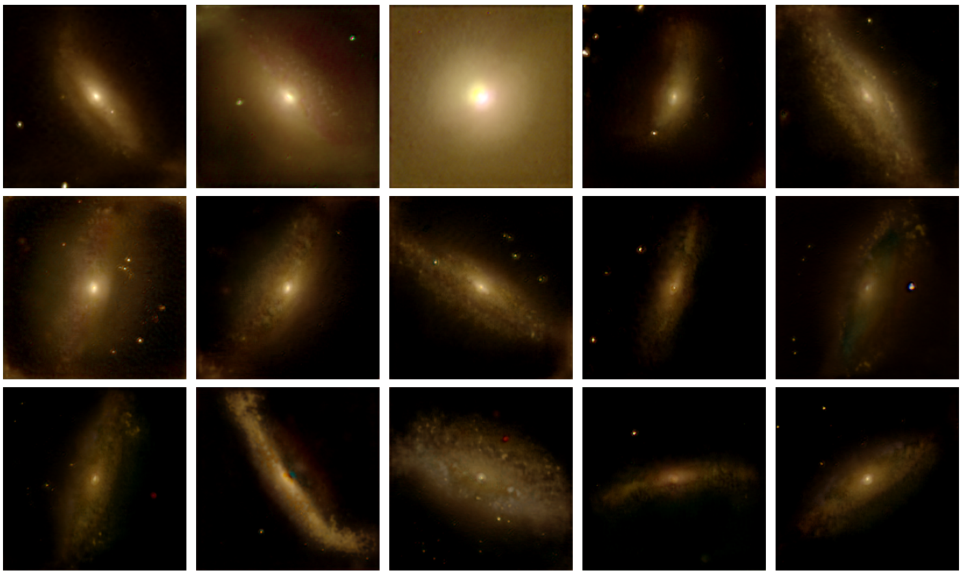

Figure 4.22: Generated images with model trained with SDSS FITS. Randomly-selected examples of generated galaxies using the best performing model

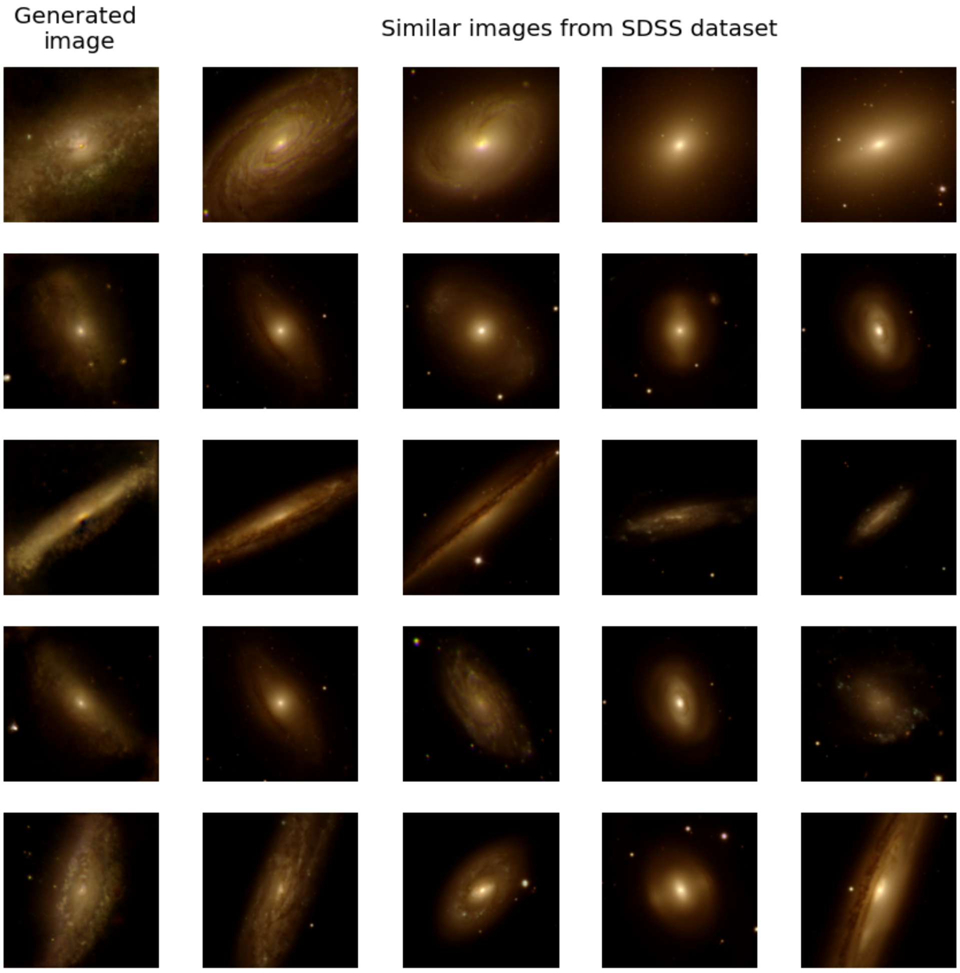

Figure 4.23: Most similar images in SDSS data using MSE. The first image of each row is a generated image from a random input vector. The generated image is compared with all the images in the SDSS dataset. The difference between the images is calculated using MSE (we add the square differences pixel by pixel and divide the total by the area of the image). The next 4 images of every row correspond to the 4 lowest values obtained.

4.4 Analyzing the final results

The main objective of our work is to be able to generate FITS images of galaxies that resemble both morphological and physical properties of galaxies. We calculate the magnitudes on the g-band and the r-band utilizing the source finding and image analysis package ProFound [30], with all parameters by default except for the magnitude zero point with a value of 0 due to the original images being nearby galaxies (having a redshift close to 0). Figure 4.24 shows an histogram of the redshifts of the training data. Using the magnitudes obtained with ProFound and the redshift z of the galaxies, we obtained the absolute magnitudes with the SDSS formula [29]:

$$\text{Absolute magnitude} = \text{Magnitude} - 5 * \log_{10}(4.28 * 10^8 * z) + 5, \tag{4.1}$$

where we use the median of the redshifts of the training data as the redshift for the generated galaxies to calculate the absolute magnitude. In Figure 4.25, we can observe the color-magnitude diagrams, from left to right, of only the training data (SDSS dataset) in blue, of only the $1,000$ generated samples in orange, and the final one contains both. We calculated the median and standard deviation of the absolute magnitude in the r-band and the color (g-r) for the SDSS data. The red dashed lines denote the area of the median value with an error of 1.75 times the standard deviation of the color and the black dashed lines correspond to the area of the median value with an error of 1.75 times the standard deviation of the absolute magnitude. 716 out of the $1,000$ generated examples fall within the ranges considered, meaning that 70.1% of the samples follow the physical properties of the SDSS data. In the example figure of a color-magnitude diagram in section 3.3 we observe a bimodal distribution, a phenomenon that in this diagram is not visible, this could be justified by observing the diagram of the training data, which also do not follow a bimodal distribution. Because this behavior is not in the training data, the generative model can not learn it. The calculations of the magnitude by ProFound are not perfect, this can be seen in the ranges of the absolute magnitude values, comparing the example figure and Figure 4.25. Figure 4.26 shows how our calculated magnitudes and the real magnitudes of the SDSS data are related, showing the values are correlated.

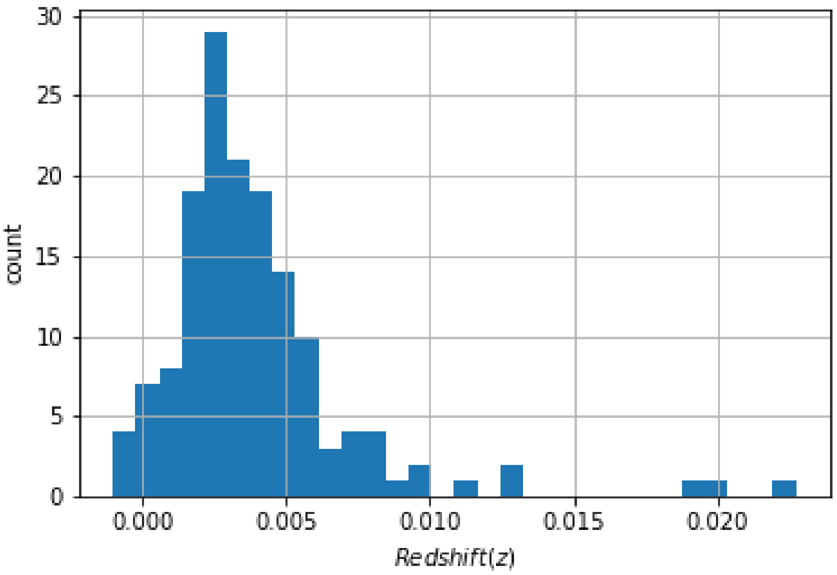

Figure 4.24: Redshift histogram. Histogram of the redshifts values of the SDSS training data.

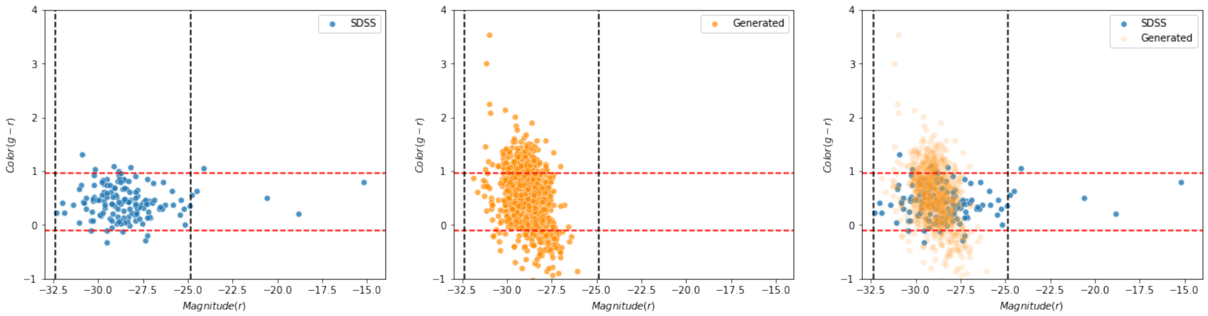

Figure 4.25: Color-magnitude diagrams. We calculate the color-magnitude diagram for all the galaxies from SDSS dataset (only the original images, without considering the rotations) (left) and 1,000 galaxies generated with the network with the lowest FID after the training with SDSS FITS images (center), we also plotted the diagram with the SDSS data and the generated samples (right). The opacity denotes the density of the points. The dashed lines denote the area of the median with an error of 1.75 times the standard deviation of the respective axis variable considering only the SDSS data values.

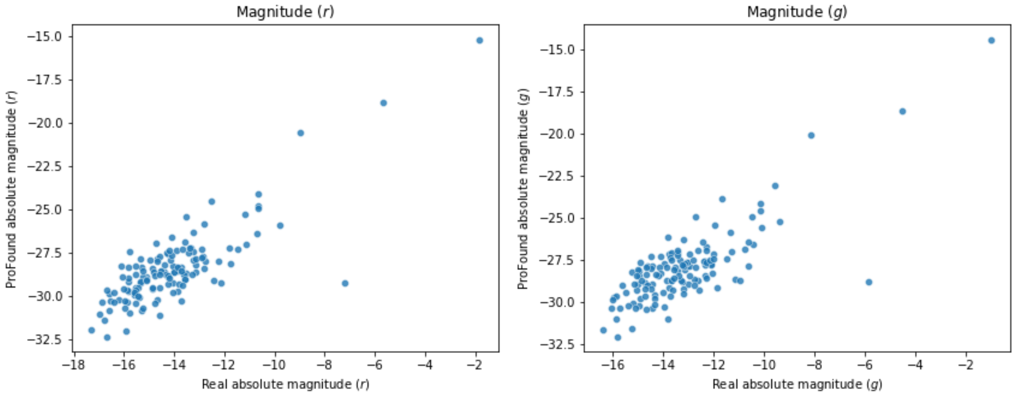

Figure 4.26: Magnitude comparison. Comparison of the absolute magnitudes on the r band (left) and the g band (right). The x axis denotes the real value of the absolute magnitude and the Y-axis denotes the value obtained using ProFound.

5 Conclusions

In this work, we evaluate the performance of the network StyleGAN with the technique of fine-tuning with images of simulated and real galaxies experimentally, and we present an alternative for generating synthetic datasets of realistic galaxy images by using the StyleGAN. We include randomly-selected examples of generated galaxies using our method. The FID in the experiment with Illustris JPG data kept decreasing overall until a certain point due to mode collapse, where the network could overcome this problem due to the loss function, and the value started to decrease constantly again, reaching its lowest value, a one-digit value (6.06). In the training with FITS data, we never reached a point where the metric increased its value significantly; this indicates that our results could be improved by increasing the training time, increasing the number of real galaxy images, or by methods like freezing the generator, trying to freeze blocks instead of layers. The metric during the final experiment, with SDSS FITS data, we never reached a two-digit value, obtaining values not lower than 148, meaning the images still are far from the training distribution and this can be seen in the blurred sections of some generated samples. The final experiments did not reach the desired results of the hypothesis: we obtained at least 70 percent of the samples falling into the training data property ranges, but we did not reach an FID of 50. With an extensive training time and with a lower learning rate, we could decrease the FID value; however, more data is needed to avoid replicating the training data, as we can conclude due to the experiment with SDSS JPG data. Nevertheless, our analysis indicates that the generated images follow the psychical properties of the training images, indicating that by improving the generator, realistic high-resolution galaxy images can be synthesized.

6 Future Work

This work could be improved by changing how the FID is calculated: finding a different network instead of InceptionV3 which specializes in astronomical data. The process to obtain the magnitudes of the generated images can also be improved or the network can be modified to generate, apart from the image, extra information like magnitudes on the bands or the redshift. The evaluation of the model could also be improved by adding the evaluation of the human eye when labeling synthetic images. This technique can potentially be used to increase the size of smaller datasets or to create new datasets with already labeled astronomical objects by combining the proposed work with additional information about the distribution of objects of interest. Another possibility for future work is to create multiple generators of different astronomical objects to simulate a universe. However, one limitation is the training time for the experiments, reaching more than a day in the second experiment only, even using 2 NVIDIA A100.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[2] C. Yang, Y. Shen, Y. Xu, and B. Zhou, “Data-efficient instance generation from instance discrimination,” *arXiv preprint arXiv:2106.04566*, 2021.

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

[4] K. Schawinski, C. Zhang, H. Zhang, L. Fowler, and G. K. Santhanam, “Generative adversarial networks recover features in astrophysical images of galaxies beyond the deconvolution limit,” *Monthly Notices of the Royal Astronomical Society: Letters*, vol. 467, no. 1, pp. L110–L114, 2017.

[5] L. Xu, W. Sun, Y. Yan, and W. Zhang, “Solar image deconvolution by generative adversarial network,” *arXiv preprint arXiv:2001.03850*, 2020.

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

[7] L. Fussell and B. Moews, “Forging new worlds: High-resolution synthetic galaxies with chained generative adversarial networks,” *Monthly Notices of the Royal Astronomical Society*, vol. 485, no. 3, pp. 3203–3214, 2019.

[8] M. J. Smith, J. E. Geach, R. A. Jackson, *et al.*, “Realistic galaxy image simulation via score-based generative models,” *Monthly Notices of the Royal Astronomical Society*, vol. 511, no. 2, pp. 1808–1818, Jan. 2022. DOI: 10.1093/mnras/stac130. [Online]. Available: https://doi.org/10.1093%2Fmnras%2Fstac130.

[9] M. Dia, E. Savary, M. Melchior, and F. Courbin, “Galaxy image simulation using progressive gans,” *arXiv preprint arXiv:1909.12160*, 2019.

[10] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410.

[11] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, “Spectral normalization for generative adversarial networks,” *arXiv preprint arXiv:1802.05957*, 2018.

[12] T. Miyato and M. Koyama, "Cgans with projection discriminator," *arXiv preprint arXiv:1802.05637*, 2018.

[13] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *arXiv preprint arXiv:1809.11096*, 2018.

[14] J. Back, *Fine-tuning stylegan2 for cartoon face generation*, 2021. DOI: 10.48550/ARXIV.2106.12445. [Online]. Available: https://arxiv.org/abs/2106.12445.

[15] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2414–2423.

[16] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1501–1510.

[17] M. Mustafa, D. Bard, W. Bhimji, *et al.*, "Cosmogan: Creating high-fidelity weak lensing convergence maps using generative adversarial networks," *Computational Astrophysics and Cosmology*, vol. 6, no. 1, pp. 1–13, 2019.

[18] A. C. Rodríguez, T. Kacprzak, A. Lucchi, *et al.*, "Fast cosmic web simulations with generative adversarial networks," *Computational Astrophysics and Cosmology*, vol. 5, no. 1, p. 4, 2018.

[19] M. J. Smith and J. E. Geach, "Generative deep fields: Arbitrarily sized, random synthetic astronomical images through deep learning," *Monthly Notices of the Royal Astronomical Society*, vol. 490, no. 4, pp. 4985–4990, 2019.

[20] Q. Lin, D. Fouchez, and J. Pasquet, "Galaxy image translation with semi-supervised noise-reconstructed generative adversarial networks," in *2020 25th International Conference on Pattern Recognition (ICPR)*, IEEE, 2021, pp. 5634–5641.

[21] D. M. Reiman and B. E. Göhre, "Deblending galaxy superpositions with branched generative adversarial networks," *Monthly Notices of the Royal Astronomical Society*, vol. 485, no. 2, pp. 2617–2627, 2019.

[22] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," *Advances in neural information processing systems*, vol. 30, 2017.

[23] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, 2016.

[24] I. Strateva, Ž. Ivezić, G. R. Knapp, *et al.*, "Color separation of galaxy types in the sloan digital sky survey imaging data," *The Astronomical Journal*, vol. 122, no. 4, p. 1861, 2001.

[25] P. Torrey, G. F. Snyder, M. Vogelsberger, *et al.*, "Synthetic galaxy images and spectra from the illustris simulation," *Monthly Notices of the Royal Astronomical Society*, vol. 447, no. 3, pp. 2753–2771, 2015.

[26] D. G. York, J. Adelman, J. Anderson John E., *et al.*, "The Sloan Digital Sky Survey: Technical Summary," vol. 120, no. 3, pp. 1579–1587, Sep. 2000. DOI: 10.1086/301513. arXiv: astro-ph/0006396 [astro-ph].

[27] D. J. Eisenstein, D. H. Weinberg, E. Agol, *et al.*, "SDSS-III: Massive Spectroscopic Surveys of the Distant Universe, the Milky Way, and Extra-Solar Planetary Systems," vol. 142, no. 3, 72, p. 72, Sep. 2011. DOI: 10.1088/0004-6256/142/3/72. arXiv: 1101.1529 [astro-ph.IM].

[28] M. R. Blanton, M. A. Bershady, B. Abolfathi, *et al.*, "Sloan Digital Sky Survey IV: Mapping the Milky Way, Nearby Galaxies, and the Distant Universe," vol. 154, no. 1, 28, p. 28, Jul. 2017. DOI: 10.3847/1538-3881/aa7567. arXiv: 1703.00052 [astro-ph.GA].

[29] B. Abolfathi, D. Aguado, G. Aguilar, *et al.*, "The Fourteenth Data Release of the Sloan Digital Sky Survey: First Spectroscopic Data from the Extended Baryon Oscillation Spectroscopic Survey and from the Second Phase of the Apache Point Observatory Galactic Evolution Experiment," vol. 235, no. 2, 42, p. 42, Apr. 2018. DOI: 10.3847/1538-4365/aa9e8a. arXiv: 1707.09322 [astro-ph.GA].

[30] A. S. G. Robotham, L. J. M. Davies, S. P. Driver, *et al.*, "Profound: Source extraction and application to modern survey data," *Monthly Notices of the Royal Astronomical Society*, vol. 476, no. 3, pp. 3137–3159, Feb. 2018, ISSN: 1365-2966. DOI: 10.1093/mnras/sty440. [Online]. Available: http://dx.doi.org/10.1093/mnras/sty440.