Efficient Galaxy Image Generation

Team Milky Boyz, ETH Zürich
Laurin Brandner, Jakub Kotal, Jodok Vieli, Samuel Waeny
{laurinb,jkotal,jvieli,waenys}@ethz.ch

***Abstract*—In this paper we shall explore various machine learning approaches for both the generation and similarity ranking of cosmology images.**
For the generation, we propose a novel ensemble model which we compare to a naive statistical baseline as well as a state of the art GAN architecture. By combining different GAN architectures together with a statistical model, our layered model is capable of greatly reducing the complexity of the generation allowing us to efficiently generate large images that are, to the human eye, indistinguishable to actual cosmology photographs.
For the approximation of the similarity function, we compare a CNN based baseline together with a random forest regression using manually chosen features.

I. Introduction

The field of cosmology relies on high resolution images of galaxies to advance their research. These images are usually captured by satellites or high resolution telescopes and generate an enormous amount of costs. Recent advances in machine learning could address this issue. Generative models like GANs can be used to produce artificial images realistic enough to be used in research. However, models like these, run on specialized hardware, which again consumes a lot of financial resources. In addition, the computational power used for research is usually centralized in a few supercomputers around the globe. Just in times of crisis, such as cyber attacks or global pandemics, these supercomputers can be a bottleneck and might slow down research dramatically. In this report we provide an approach to generate artificial cosmology images with low computational resources. Specifically, we develop a model that makes use of the low information density of cosmology images. Our model, which we will call layered statistical model for future reference, breaks down the image into different, predefined layers and generates each layer individually. This approach allows us to train the model on hardware that comes in off the shelf laptops. The hope is that our solution helps the research community to gain independence from highly specialized and centralized hardware and therefore will make the research community much more resilient to global crises.

II. Related Work

Generative adversarial networks (GANs), introduced by Ian J. Goodfellow et al. [Goo+14], are currently used extensively to generate any kind of artificial data. GANs consist of two neural networks, the generator and the discriminator, optimizing adversarial objectives. The generator tries to trick the discriminator by generating fake data, that is as close as possible to the real data, whereas the discriminator tries to differentiate between fake and real data. This adversarial approach has the advantage that no heuristic cost function is needed. Instead the cost function is replaced by the discriminator. The idea of GANs is to extend Deep Convolutional GANs (DCGANs) that use convolving templates to reduce the number of parameters when working with images. The first working DCGAN was introduced by Radford et al. (2015) [RMC15] based on the idea of convolutional networks proposed by LeCun et al. [LeC+90]. DCGANs are used in many variations, which are extending DCGAN for specific objectives. CycleGAN [Zhu+17] for example is a specialized model to learn transformation of images of different styles, whereas styleGAN [KLA18] produces very high resolution images based on a stack of DCGAN like networks. While GANs generally produce good results, they are sensitive to the choice of parameters. Especially with images carrying low information density, such as cosmology images, GANs struggle to capture the essential features and suffer from mode collapse.

III. Models and Methods

A. Generating cosmology images

Whilst the images we want to generate are very large i.e. 1000×1000 px, the information they contain is vastly simpler. This leads to much less relevant data per image, but also makes it difficult to train an accurate model that is capable of coping with said sparsity. The structure of the images can be decomposed into three independent parts, namely the background, the stars and the distribution of the stars. The background is very dark and is hardly distinguishable from being all black with maximal pixel intensities that rarely surpass 30. The stars are quite small with a diameter of around 10-20 pixels. They are mainly circular, or diamond shaped and have varying brightnesses. Lastly, there is a varying number of stars in each image, somewhere between 5-25 and each star has a location on the image following some distribution.

1) Baseline 1: styleGAN: StyleGAN1 [KLA18] is a state of the art GAN that makes use of an intermediate latent space that gives fine-grained control over the output. Furthermore, it makes use of progressive growing [Kar+17] to stabilize the training procedure. Though it was originally used for the generation of artificial faces, its new architecture as well as its capability to generate high definition images make it an interesting baseline to compare the layered statistical model to.

2) Baseline 2: Naive statistical model: In a drastic abstraction, a cosmology image consists of a black background, and a specific amount of white circles with different sizes. This observation motivates a very simple statistical model, which we will call *naive statistical model* for future reference. This model is trained by computing a simple probability distribution of occurrences of stars for each position on the image. To reduce the number of positions, a position is modeled as a tile of 10×10 px on the image. The probability distribution is then computed per tile to model a likelihood of a tile containing a star. A new cosmology image is generated by sampling a random data point from the distribution containing a set of stars each modeled by position and radius. The stars are then just drawn as white circles on a black background of size 1000×1000 px.

3) Layered statistical model: The *naive statistical model* is a drastic oversimplification and does not capture any details of the cosmology images such as the particular shape of the stars nor the texture of the background. This lack of detail motivates a more complex statistical model we call *layered statistical model* for future reference. Instead of just making assumptions about the shape of the stars and brightness of the background, we examine the image for each layer individually and then generate a new image as a composition of three independent parts. That is to say, we can first generate a background, a set of coordinates and then for each coordinate we generate a corresponding star.

a) Data preparation: The key difficulty of this approach lies in extracting the three parts from our training data, that is isolating the stars, separating the background and extracting the coordinates of the stars.
In order to isolate the stars. We first choose a patch size of 30×30 px, which we will use to store our stars. We want to choose the smallest patch size possible whilst remaining sure that all stars will be able to fit inside. Afterwards, we compute the values of each patch inside of an image using a filter that tries to center the stars inside of the patches. We than remove all the patches that have a value under a certain threshold and make sure that only pick patches that are at a certain distance from one another so that we don't have multiple samples of the same star. Once the stars have been isolated, we can simply save their coordinates to get their distribution. We can also remove the stars from the images in order to get the background noise distribution.

b) Training: Once the training data has been generated, we can begin training our models. To generate the 30×30 px stars, we first tried a regular DCGAN architecture [RMC15]. However, we ran into the issue of mode collapse, that is to say that our generator would always output the same image, no matter the random noise we would feed it. To remedy this problem, we changed our loss function and adapted our model to the Wasserstein GAN [ACB17].
In order to generate the coordinates of our stars, we started by preprocessing our data. We note that in our training data, no image contains more than 25 stars. We then scale all our coordinates between $[0, 1]$ and for the images that have less than 25 stars, we fill the rest with -1. This time we use a regular DCGAN architecture, but with a fully linear generator and discriminator. The last layer has a tangent activation function and outputs 50 values, or 25 $(x, y) \in [-1, 1]$. Unlike the other two modules, the background module does not require training. Due to the simple nature of the background, a random variable reproducing the brightness density function is sufficient to produce realistic background noise. This procedure requires the single pixels to be independent of one another, an assumption we could empirically verify. Furthermore, our experiments have shown that learning this random variable from just one real image yields the best results.

c) Generation: Once the three modules have been trained, generating galaxy images is quite straight forward. Firstly, we generate the background according to the previously learnt brightness density function. We then generate 25 pairs of x,y coordinates and for each case where both $x, y \in [0, 1]$ we generate a star and insert it at that position scaled between $[0, 1000]$, in this way we can have a variable amount of stars between generated images.

B. Scoring Cosmology Images

Our first attempt at approximating the similarity function uses a deep convolutional backbone like ResNet [He+15]. However, this solution quickly proved to be incapable of approximating the function accurately enough, yielding a mean absolute deviation of around 0.63. We identified three problems that make it difficult to produce a better score: the sparsity of the cosmology images make it difficult for any image detection model to select the relevant features, the heterogeneity of the images' subjects would require multiple backbones to find a decent approximation and lastly, the imbalanced nature of the label distribution give the model a bias towards lower ranges in the regression. The natural answer to these problems is a random forest regression [LW+02]. This model allows us to select the features manually and learns to partition them. A histogram of pixel intensities and a power spectrum were used as main features. This model scored a mean absolute error of 0.116.

IV. Results

We use the random forest regression model to compute an average score over 500 randomly sampled cosmology images generated by our layered model as well as the two baselines. This allows us to compare the quality of generated images across models. Figure 1 shows the averaged score for each model. With a score of 2.00, the layered statistical model achieved the highest score. Furthermore, we note that the layered model experiences the largest score discrepancy, ranging from 0.35 up to 4.13. We conclude that our solution is capable of producing a larger variety of images than the two baselines.

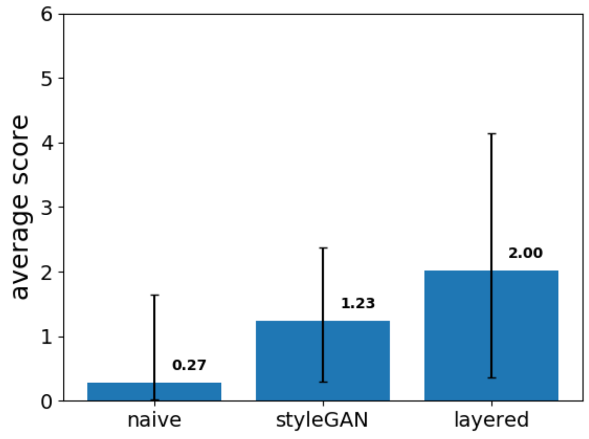
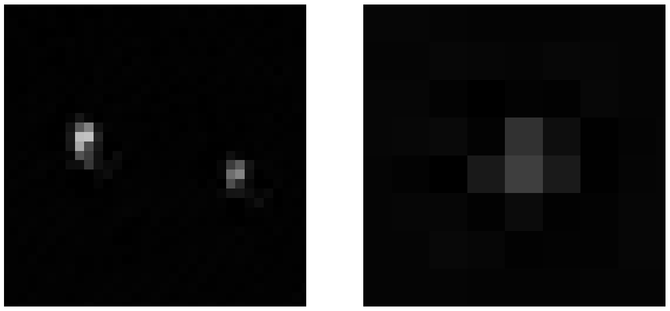


Fig. 1: Averaged score over 500 images generated by the naive statistical model, styleGAN, and the layered statistical model. With 2.00 the layered statistical model achieved the highest score.

The cosmology images produced by our styleGAN baseline look realistic and achieve a decent score of 1.23. However, the training regularly results in a mode collapse, where it only produces all-black images. This could potentially be fixed using a larger batch size, however this would have required multiple GPUs which was out of scope for our project. Another way of circumventing this problem is to slice the high-resolution images into smaller tiles of size 256 × 256px. As can be seen in figure 2, the model produces better results for the smaller images, where the stars are much more pronounced and experience a less boxy appearance. The histograms in figure 4 show the distribution of pixel intensities of images generated by each model. The brightness distribution generated by styleGAN is much more evenly spread out compared to real cosmology images which explains the low similarity score.

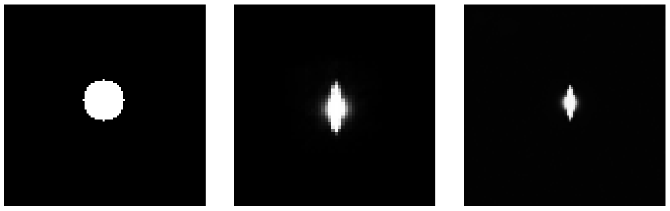
The cosmology images generated by the naive statistical model achieve a score of 0.27. The images look similar to the real cosmology images but lack details such as variable shapes of the individual stars and texture of the background. The layered statistical model improves on this and captures



(a) 256 × 256 px (b) 1024 × 1024 px

Fig. 2: Close-up images (64 × 64 px) produced by styleGAN

details such as the different shapes of the individual stars. Figure 3 shows the improvement of the layered statistical model compared to the circle shaped stars generated by the naive statistical model. In addition to a higher level of detail of the individual stars, the layered statistical model also improves the quality of the background and the amount of detail around the individual stars. As can be seen in figure **??**, the real image has a significant amount of pixels with different shades of gray while the images generated by the naive statistical model are mostly black. This is a direct result of the oversimplification to model the background all black and the stars as all white circles. The histogram created on the images generated by the layered statistical model closely resembles the one of the real images.



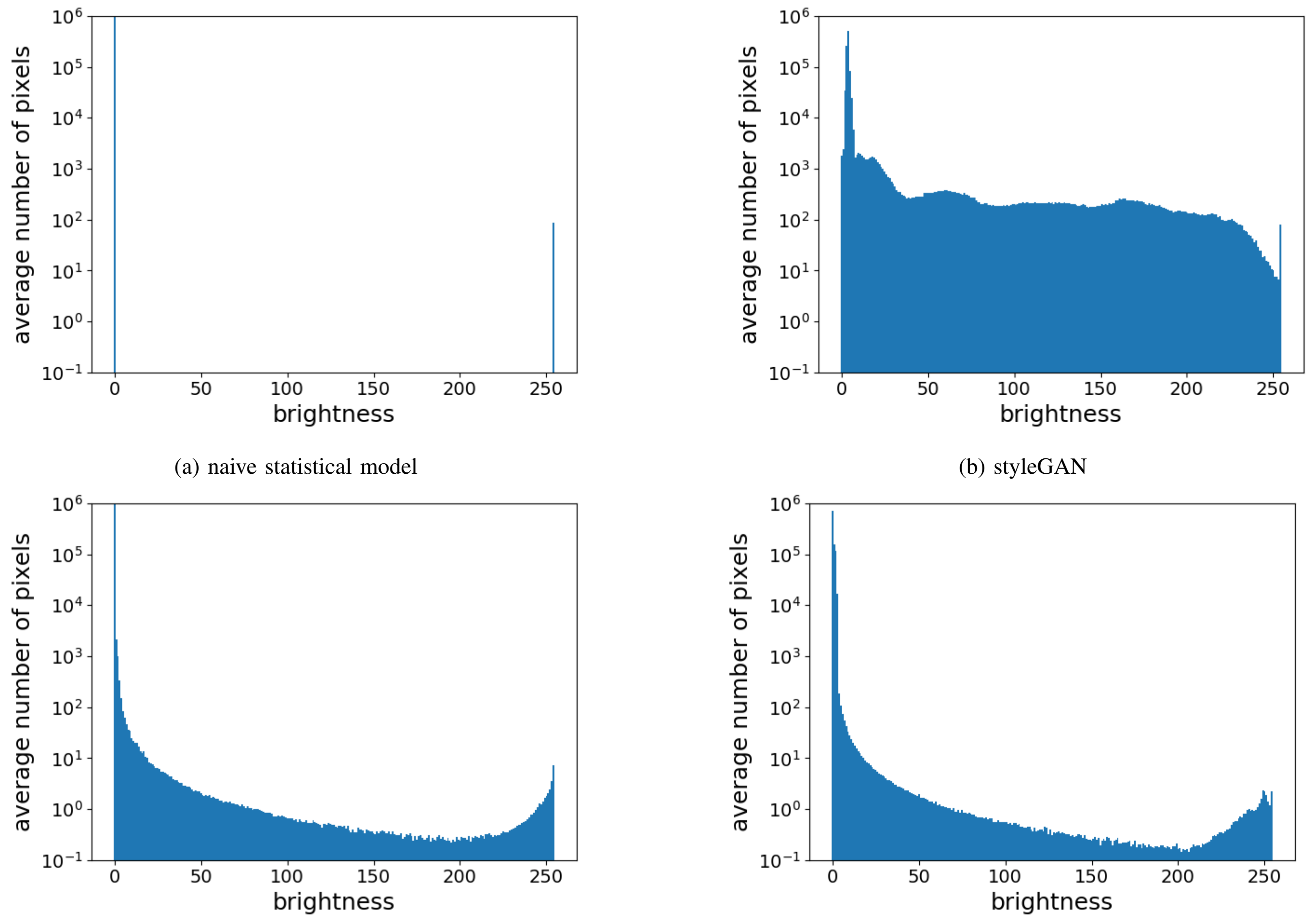
(a) naive (b) layered (c) real

Fig. 3: Comparison of the level of detail in generated and real stars

	styleGAN	**naive model**	**layered model**
hardware	p3.8xlarge	t3.xlarge	
cost [USD/h]	12.24	0.17	
runtime [h]	1.93	0.006	0.228
total cost [USD]	23.623	0.001	0.038

TABLE I: Approximated training cost

To compare the results of the styleGAN, the naive statistical model and the layered statistical model considering the cost of the needed hardware we trained each model on *Amazon AWS EC2* [Ama20], using a GPU instance (*p3.8xlarge*, 32C@2.5GHz, 244GB, 4xTesla V100) for styleGAN and a CPU only instance (*t3.xlarge*, 4C@2.5GHz, 16GB) that is comparable to a common laptop for the other baseline and our approach. The results can be found in table I. We can



(a) naive statistical model

(b) styleGAN

(c) layered statistical model

(d) real cosmology images

Fig. 4: Average histogram of 500 images produced by the respective models.

easly conclude that our final model, which achieves the best average similarity score, is not only much cheaper to train but can also be trained locally on a personal computer.

V. Discussion

Using our layered statistical model, we are able to generate large high resolution images of galaxies that are, to the human eye, indistinguishable from actual images. Our method, profiting of the sparse nature of the data contained in the training set, was able to greatly simplify the generation process without having to reduce the resolution of the output. This allowed us to have a model that is fast to train, does not require any specialised hardware and there is no need for tedious hyper parameter tuning to ensure convergence. Furthermore, the layered approach in the design of the model, gives us a greater control over the output. For example, if we wanted to dictate the exact amount of stars contained in the generated image, this would be straightforward to implement.

However, this model does suffer some shortcomings. Most notably, it is impossible to model a dependency among stars. Let's assume that in the actual galaxy images, the distribution would dictate that all stars close to each other share the same shape or brightness. This behaviour would be impossible to represent in our current model as each star is generated independently from random data.

Due to the amount of problem specific knowledge we used to design our model, it also lacks the capability of generalizing well for other problems. Indeed, even if we were to increase the complexity of our galaxy images to include e.g. clusters of stars in particular shapes it would be complicated to adapt our model to fit such requirements.

Though the regression model used to score cosmology images achieves an adequate mean average error. It is highly likely we could further improve upon this score by defining more specific ranges for the power spectrum, or computing these values for specific zones of the images. However the improvements would certainly be very minor so we did not find it worthwhile to run these experiments. Furthermore, our method does not take the actual shapes of the stars into account, which leaves it vulnerable to adversarial examples that reproduce the power spectrum and exhibit a similar brightness histogram.

References

[LeC+90] Yann LeCun et al. "Handwritten Digit Recognition with a Back-Propagation Network". In: *Advances in Neural Information Processing Systems 2*. Ed. by D. S. Touretzky. Morgan-Kaufmann, 1990, pp. 396–404. URL: http://papers.nips.cc/paper/293-handwritten-digit-recognition-with-a-back-propagation-network.pdf.

[LW+02] Andy Liaw, Matthew Wiener, et al. "Classification and regression by randomForest". In: *R news* 2.3 (2002), pp. 18–22.

[Goo+14] Ian J. Goodfellow et al. *Generative Adversarial Networks*. 2014. arXiv: 1406.2661 `[stat.ML]`.

[He+15] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 `[cs.CV]`.

[RMC15] Alec Radford, Luke Metz, and Soumith Chintala. *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*. 2015. arXiv: 1511.06434 `[cs.LG]`.

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. *Wasserstein GAN*. 2017. arXiv: 1701.07875 `[stat.ML]`.

[Kar+17] Tero Karras et al. *Progressive Growing of GANs for Improved Quality, Stability, and Variation*. 2017. arXiv: 1710.10196 `[cs.NE]`.

[Zhu+17] Jun-Yan Zhu et al. *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*. 2017. arXiv: 1703.10593 `[cs.CV]`.

[KLA18] Tero Karras, Samuli Laine, and Timo Aila. *A Style-Based Generator Architecture for Generative Adversarial Networks*. 2018. arXiv: 1812.04948 `[cs.NE]`.

[Ama20] Amazon. *Amazon Web Services EC2*. [Online; accessed 05.07.2020]. 2020. URL: https://aws.amazon.com/ec2/.