2024

Galaxy Classification

A Deep Learning Model

Team Members :

- Messaoudi Nabehet
- Bouazza Ayat
- Meribai Khouloud
- Khoudja Yousra

05/09/2024

Abstract

Galaxy classification is a fundamental task in astrophysics, enabling astronomers to understand the structure, formation, and evolution of galaxies. In this study, we propose an automated approach to classify galaxies based on their images using deep learning techniques. Our model leverages convolutional neural networks (CNNs) to learn discriminative features directly from raw image data, eliminating the need for manual feature extraction. The dataset comprises images of galaxies spanning different morphological types, including elliptical, spiral, and irregular galaxies.

We designed and trained a deep CNN architecture tailored to galaxy classification tasks, optimizing its performance through extensive experimentation and hyperparameter tuning. Our model achieved an acceptable classification accuracy on the validation set, demonstrating its capability to distinguish between different galaxy types accurately. Additionally, to enhance the robustness and generalization capability of our deep learning model, we employed Generative Adversarial Networks (GANs) for data augmentation. This innovative technique allowed us to generate synthetic galaxy images, effectively expanding the diversity of our training dataset. By introducing variations in the dataset through GAN-based augmentation, our model was better equipped to handle different galaxy morphologies and variations in image quality.

Furthermore, we applied our trained model to classify a separate test set of galaxy images, achieving promising results consistent with those obtained during validation. The robustness and generalization capability of our model indicate its potential for practical applications in large-scale galaxy surveys and astronomical research. Our study contributes to advancing the field of astrophysics by providing a reliable and efficient automated solution for galaxy classification, facilitating the analysis of vast astronomical datasets and accelerating discoveries in galaxy morphology and evolution.

Table of Contents

Introduction

1.1 Background

Galaxy classification stands as a cornerstone in astrophysical research, enabling astronomers to glean insights into the formation, dynamics, and evolution of galaxies. Traditionally, this classification has relied heavily on manual methods, where experts painstakingly analyze images to categorize galaxies based on their morphological features. While effective, manual classification methods suffer from subjectivity, inconsistency, and scalability issues, particularly when dealing with large datasets. As the volume of astronomical data continues to grow exponentially, there arises a pressing need for automated approaches that can handle the deluge of data efficiently.

1.2 Motivation for the Project

The motivation behind this research stems from the potential of deep learning techniques to revolutionize galaxy classification. Deep learning, with its ability to learn hierarchical representations directly from raw data, offers a promising avenue for automating the classification process. Specifically, Convolutional Neural Networks (CNNs) have demonstrated remarkable prowess in image recognition tasks, making them well-suited for analyzing galaxy images. Additionally, Generative Adversarial Networks (GANs) present an innovative means of data augmentation, which can enhance the robustness and generalization capability of deep learning models.

1.3 Objectives of the Study

The primary objective of this study is to develop a deep learning-based approach for galaxy image classification, leveraging CNNs and GANs. Specifically, we aim to:

- Design and implement a CNN architecture tailored for galaxy classification tasks.
- Explore the utility of GANs for data augmentation to enhance the diversity and quality of the training dataset.
- Evaluate the performance of the proposed deep learning model on a diverse range of galaxy images.

1.4 Structure of the Paper

This paper is structured as follows:

- **Introduction:** Provides background, motivation, objectives, and outlines the paper's structure.

- **Chapter 1:** The CNN Model: Covers CNN implementation, including custom model design, popular architectures, and transfer learning.

- **Chapter 2:** Data Augmentation with GANs: Discusses GAN-based data augmentation, methodology, results, challenges, and conclusions.

- **Chapter 3:** Model Deployment: Explores practical deployment considerations for the trained models.

- **Conclusion:** Summarizes findings, implications, and future directions.

1.5 Introduction to the Dataset

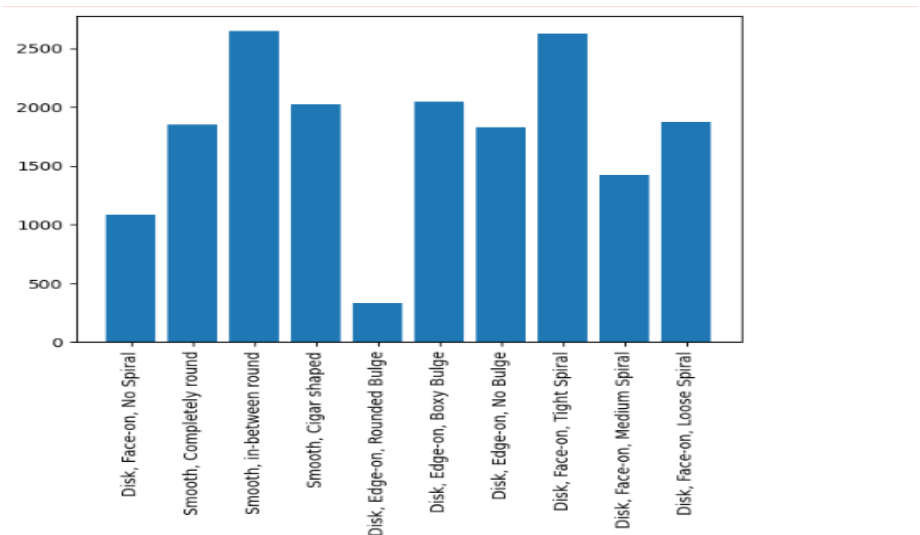
The Galaxy Zoo project, a collaborative effort between astronomers and citizen scientists, provides a rich repository of galactic images, facilitating the classification of these celestial entities. This study focuses on the latest iteration of the Galaxy Zoo dataset, chosen for its superior image quality, to delve into the task of galaxy classification. Additionally, we utilized the Galaxy10 dataset, sourced from astroNN, which offers a curated collection of galaxy images spanning various morphological types. The dataset is meticulously curated to include galaxies of different morphological types, spanning elliptical, spiral, and irregular galaxies. More information about the Galaxy Zoo project and its datasets can be found here.

a- Data Overview:

The Galaxy10 dataset comprises 17736 images, each with dimensions of (256, 256, 3). These images represent unique observations of galaxies, with a standardized size of 256 pixels in width and height, and 3 color channels (RGB) providing detailed visual information. Within the dataset, these images are categorized into 10 distinct classes, representing different morphological types of galaxies. These classes include:

- 'Disk, Face-on, No Spiral'
- 'Smooth, Completely round'
- 'Smooth, in-between round'
- 'Smooth, Cigar shaped'
- 'Disk, Edge-on, Rounded Bulge'
- 'Disk, Edge-on, Boxy Bulge'
- 'Disk, Edge-on, No Bulge'
- 'Disk, Face-on, Tight Spiral'
- 'Disk, Face-on, Medium Spiral'
- 'Disk, Face-on, Loose Spiral'

b - Data Exploration:



c- Dataset Limitations:

The dataset used for galaxy classification in this study presents a notable limitation stemming from class imbalance, where certain types of galaxies are disproportionately represented compared to others. This imbalance introduces a challenge known as the accuracy paradox, wherein classifiers may achieve high accuracy rates by predominantly predicting the majority class while overlooking the minority classes. In the realm of galaxy classification, this means that classifiers may struggle to accurately identify and classify rare or underrepresented types of galaxies, potentially leading to biased or incomplete results. Consequently, reliance solely on accuracy as an evaluation metric may provide a misleading assessment of classifier performance. To address this limitation, it's crucial to employ evaluation metrics that consider class imbalance, such as precision, recall, or F1-score, to ensure a more comprehensive understanding of the classifier's ability to classify galaxies across all types effectively.

Chapter 1: The CNN Model

1.1 Introducing Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a specialized type of artificial neural network designed primarily for processing and analyzing visual data, such as images and videos. CNNs are structured to automatically and adaptively learn spatial hierarchies of features from the input data. Here are some essential points about CNNs:

1. **Convolutional Layers:** The core building blocks of CNNs are convolutional layers. These layers apply convolution operations to the input data using learnable filters or kernels. Each filter detects specific patterns or features within the input, such as edges, textures, or shapes.
2. **Pooling Layers:** After convolutional layers, pooling layers are often used to reduce the spatial dimensions of the feature maps while retaining important information. Common pooling operations include max pooling and average pooling.

Overall, CNNs have revolutionized the field of computer vision and have become indispensable tools for analyzing and understanding visual data in diverse domains.

1.2 Building a custom CNN model

To craft our CNN model, we embraced a meticulous approach akin to "babysitting." This involved a hands-on process of continual experimentation, wherein we methodically tweaked various parameters to fine-tune our models.

Rather than settling for a single configuration, we actively explored a range of possibilities, adjusting parameters such as learning rates, batch sizes, activation functions, and more. This iterative process allowed us to observe firsthand how each adjustment influenced the model's performance.

We started by a model with 2 conv layers and increased the size of the conv layer in each new model and we found these parameters to work the best :

- 20 epochs .

- padding.

- Max Pooling.

These are the models summaries :

First Model

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 256, 256, 3)       228

 max_pooling2d (MaxPooling2  (None, 128, 128, 3)       0
 D)

 conv2d_1 (Conv2D)           (None, 128, 128, 8)       608

 max_pooling2d_1 (MaxPoolin  (None, 64, 64, 8)         0
 g2D)

 flatten (Flatten)           (None, 32768)             0

 dense (Dense)               (None, 120)               3932280

 dense_1 (Dense)             (None, 84)                10164

 dense_2 (Dense)             (None, 10)                850

=================================================================
Total params: 3944130 (15.05 MB)
Trainable params: 3944130 (15.05 MB)
Non-trainable params: 0 (0.00 Byte)
```

Second Model

```
 conv2d_2 (Conv2D)           (None, 256, 256, 6)       456

 max_pooling2d_2 (MaxPoolin  (None, 128, 128, 6)       0
 g2D)

 conv2d_3 (Conv2D)           (None, 128, 128, 16)      2416

 max_pooling2d_3 (MaxPoolin  (None, 64, 64, 16)        0
 g2D)

 conv2d_4 (Conv2D)           (None, 64, 64, 32)        12832

 max_pooling2d_4 (MaxPoolin  (None, 32, 32, 32)        0
 g2D)

 flatten_1 (Flatten)         (None, 32768)             0

 dense_3 (Dense)             (None, 120)               3932280

 dense_4 (Dense)             (None, 84)                10164

 dense_5 (Dense)             (None, 10)                850

=================================================================
Total params: 3958998 (15.10 MB)
Trainable params: 3958998 (15.10 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Third Model

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_5 (Conv2D)           (None, 256, 256, 3)       228

 max_pooling2d_5 (MaxPoolin  (None, 128, 128, 3)       0
 g2D)

 conv2d_6 (Conv2D)           (None, 128, 128, 8)       224

 max_pooling2d_6 (MaxPoolin  (None, 64, 64, 8)         0
 g2D)

 conv2d_7 (Conv2D)           (None, 64, 64, 16)        3216

 max_pooling2d_7 (MaxPoolin  (None, 32, 32, 16)        0
 g2D)

 conv2d_8 (Conv2D)           (None, 32, 32, 32)        4640

 max_pooling2d_8 (MaxPoolin  (None, 16, 16, 32)        0
 g2D)

 conv2d_9 (Conv2D)           (None, 16, 16, 64)        51264

 max_pooling2d_9 (MaxPoolin  (None, 8, 8, 64)          0
 g2D)
```

Fourth Model

```
conv2d_13 (Conv2D)          (None, 32, 32, 32)        4640

max_pooling2d_13 (MaxPooli  (None, 16, 16, 32)        0
ng2D)

conv2d_14 (Conv2D)          (None, 16, 16, 64)        51264

max_pooling2d_14 (MaxPooli  (None, 8, 8, 64)          0
ng2D)

conv2d_15 (Conv2D)          (None, 8, 8, 128)         73856

max_pooling2d_15 (MaxPooli  (None, 4, 4, 128)         0
ng2D)

flatten_3 (Flatten)         (None, 2048)              0

dense_9 (Dense)             (None, 120)               245880

dense_10 (Dense)            (None, 84)                10164

dense_11 (Dense)            (None, 10)                850

=================================================================
Total params: 390322 (1.49 MB)
Trainable params: 390322 (1.49 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

The results:

	Accuracy	Weighted-avg Precision	Weighted-avg Recall	Weighted-avg F1-score
2 conv layers	0.41	0.43	0.41	0.41
3 conv layers	0.45	0.45	0.43	0.44
5 conv layers	0.55	0.54	0.55	0.54
6 conv layers	0.65	0.64	0.65	0.64

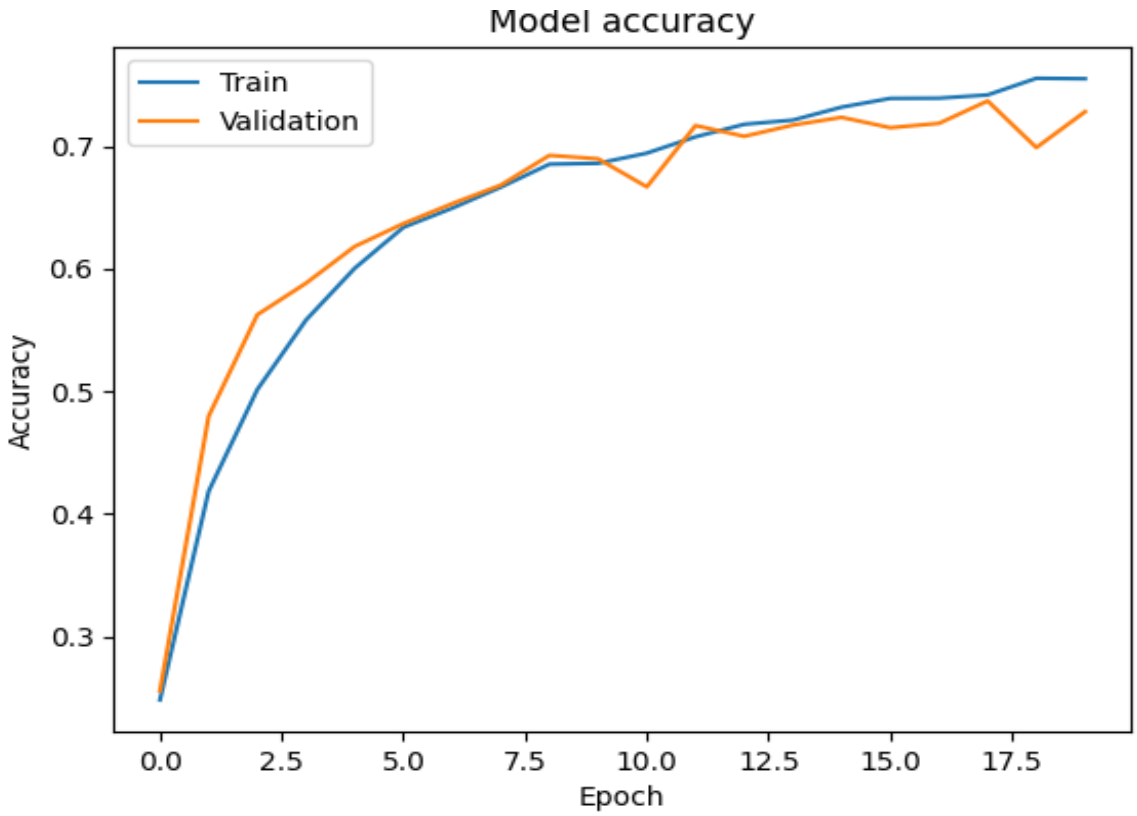
We chose weighted average because it provides a more representative measure of overall performance across all classes by considering class imbalances, whereas macro average treats all classes equally regardless of their prevalence in the dataset.

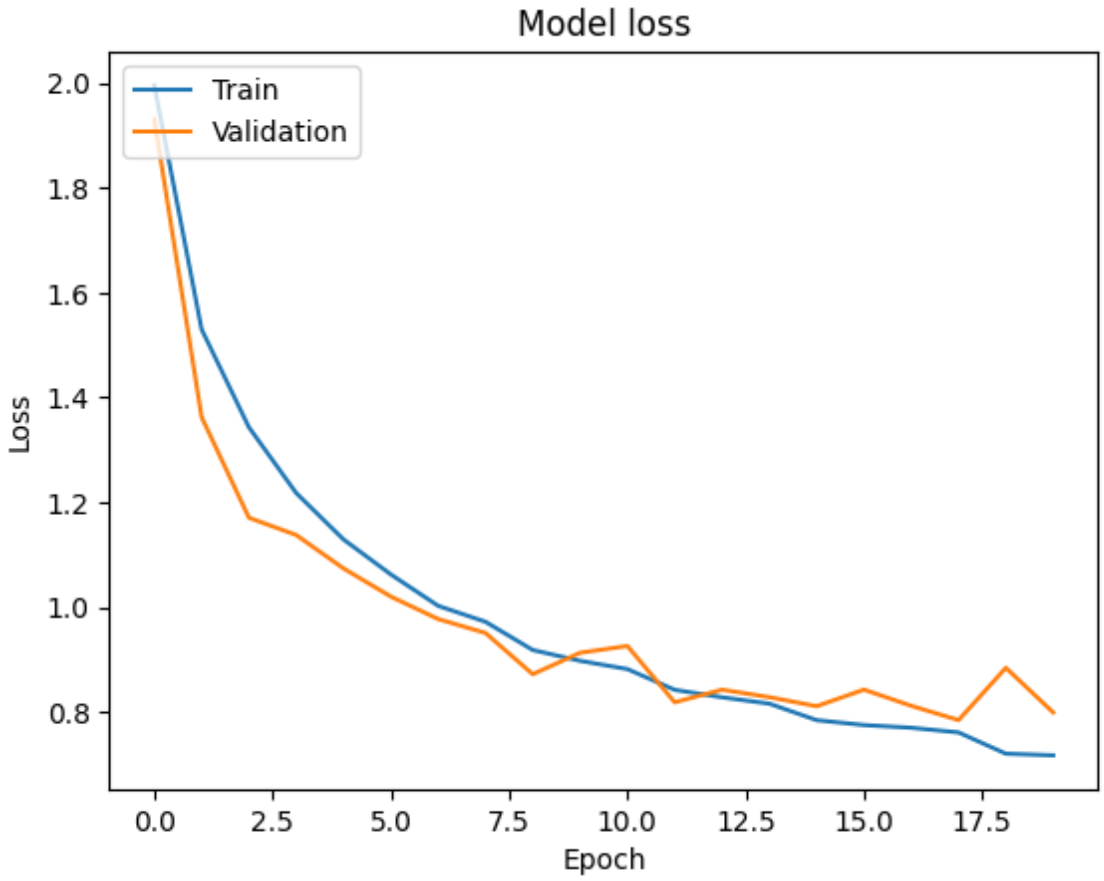
We then settled on a model with 4 conv layers but with more kernels

max_pooling2d_4 (MaxPooling2D)	(None, 32, 32, 32)	0
conv2d_5 (Conv2D)	(None, 32, 32, 64)	18496
max_pooling2d_5 (MaxPooling2D)	(None, 16, 16, 64)	0
conv2d_6 (Conv2D)	(None, 16, 16, 128)	73856
max_pooling2d_6 (MaxPooling2D)	(None, 8, 8, 128)	0
conv2d_7 (Conv2D)	(None, 8, 8, 256)	295168
max_pooling2d_7 (MaxPooling2D)	(None, 4, 4, 256)	0
flatten_1 (Flatten)	(None, 4096)	0
dense_3 (Dense)	(None, 512)	2097664
dropout_2 (Dropout)	(None, 512)	0
dense_4 (Dense)	(None, 256)	131328
dropout_3 (Dropout)	(None, 256)	0

Although this model had less accuracy than the model with 6 conv layers , when we tried it on resized images with data augmentation it performed better and reached an accuracy of 73%.

	precision	recall	f1-score	support
0	0.78	0.08	0.15	216
1	0.70	0.81	0.75	371
2	0.88	0.86	0.87	529
3	0.83	0.84	0.84	405
4	0.61	0.49	0.55	67
5	0.70	0.73	0.71	409
6	0.60	0.66	0.63	366
7	0.61	0.61	0.61	526
8	0.77	0.89	0.83	284
9	0.76	0.87	0.81	375
accuracy			0.73	3548
macro avg	0.73	0.69	0.68	3548
weighted avg	0.73	0.73	0.71	3548


Model accuracy
Train
Validation
Accuracy
0.7
0.6
0.5
0.4
0.3
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epoch


Model loss
Train
Validation
Loss
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epoch

1.3 Models with Popular CNN architectures :

We opted to utilize popular CNN architectures due to their well-established track records and widespread acceptance within the deep learning community. These architectures, such as VGG, ResNet, and Inception, have been rigorously tested and fine-tuned on large-scale datasets, demonstrating superior performance across a variety of computer vision tasks. By leveraging these established frameworks, We aimed to capitalize on their proven effectiveness in feature extraction and hierarchical representation learning, ultimately streamlining the development process and increasing the likelihood of achieving optimal results for my specific application.

The results were the following :

	Accuracy	**Weighted-avg F1-score**	**Training Time**	**Epochs**
VGG16	0.15	0.04	1.02 h	8

ResNet50	0.50	0.46	21.37 mn	5
MobileNET	0.59	0.59	17.82 mn	16
MObileNetV2	0.10	0.02	8.3 mn	4

From the table we can see that some models like MObileNetV2 and VGG16 were'nt able to learn at all , even when dropping the early stopping , while MobileNET was able to achieva an acceptable performance .This shows us that some architecture are appropriate to some images while others are not and that we should find the architecture that works with our model the best.

1.4 Transfert Learning

As a third approach, we harnessed the power of transfer learning trained on ImageNet to enhance the performance of our CNN model. By leveraging pre-trained models trained on ImageNet, a vast dataset with millions of labeled images spanning thousands of categories, we capitalized on the wealth of knowledge and feature representations learned by these models during their training process. This strategy allowed us to transfer the learned features from the pre-trained model to our specific task, saving significant time and computational resources while still achieving competitive performance. By fine-tuning the pre-trained model on our target dataset, we effectively tailored its knowledge to suit the nuances of our particular problem domain, enabling the model to learn from a smaller dataset and potentially achieve higher accuracy and generalization ability.

The results were the following :

	Accuracy	**Weighted-avg F1-score**	**Training Time**	**Epochs**
VGG16	0.56	0.55	31 mn	8

ResNet50	0.46	0.42	11.20 mn	7
MobileNET	0.47	0.45	2 mn	5
MObileNetV2	0.36	0.36	7.74 mn	7

From the table we can see variance in the performance of the models , where VGG16 performed the best , consider that only the last convolutional layer and the 3 dense layers were trained.

We can say that implementing popular architectures and transfer learning didn't give us a better performing model , which takes to study other methods to improve the model's performance

Chapter 2 : Data augmentation with GANs.

2.1 Introduction

To mitigate the issue of our imbalanced dataset, we've implemented a Generative Adversarial Network (GAN) from scratch to generate synthetic images, thereby bridging the gap in the representation of minor classes.

2.2 Methodology

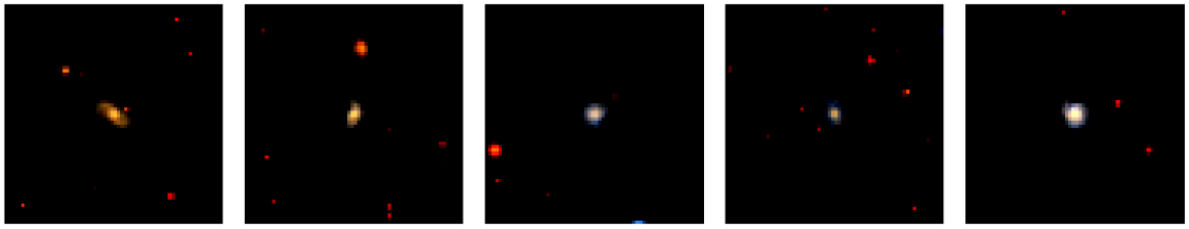
Architecture Design: We designed the architecture for both the generator and discriminator networks utilizing Convolutional Neural Networks (CNNs). For the generator, we aimed at creating realistic images that closely resemble the distribution of the minority classes in the dataset. Meanwhile, the discriminator was trained to distinguish between real and synthetic images.

Training Procedure: The GAN was trained on our imbalanced dataset with a specific focus on enhancing the representation of underrepresented galaxy classes. We utilized a custom

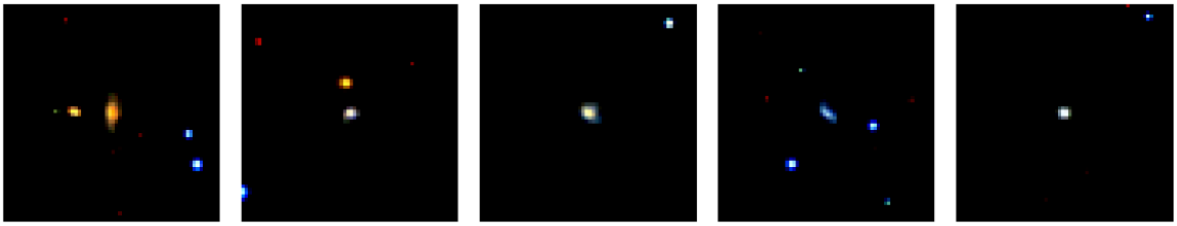
training function that optimizes the parameters of both the generator and discriminator networks in an adversarial manner. During training, the generator learned to produce synthetic images that not only fooled the discriminator *but also captured the characteristics of the minority classes.*

2.3 Results

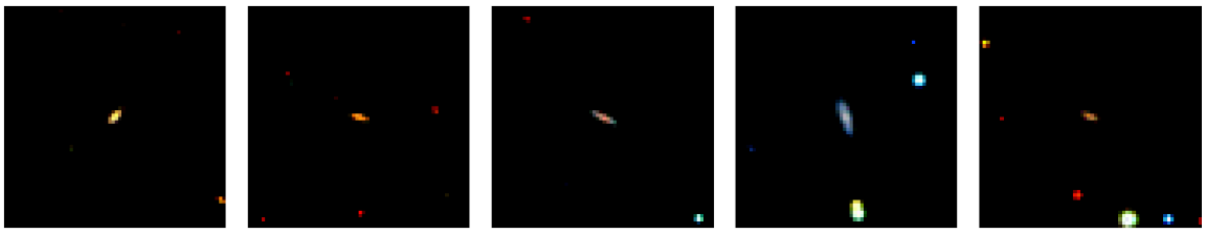
Generated Images: Below are examples of images generated by the GAN alongside real images:



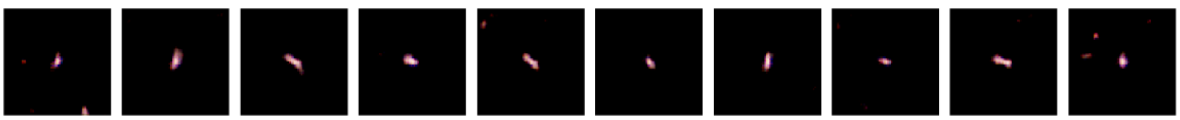
Real images



GAN images

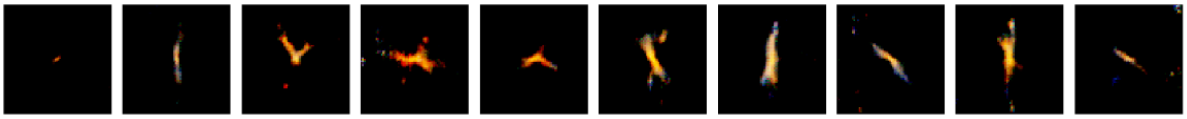


Real Images



GAN Images

Real Images



GAN images

Evaluation Metrics: To assess the effectiveness of our GAN-based approach, we employed the Fréchet Inception Distance (FID) score, a widely used metric for evaluating the quality of generated images.

FID Score Definition: The Fréchet Inception Distance (FID) score measures the similarity between the distributions of real and generated images by comparing their feature representations extracted from a pre-trained Inception network. A lower FID score indicates a higher similarity between the distributions, suggesting that the generated images closely resemble the real ones.

FID Score Calculation:

1. **Extract Features**: Let $\mathbf{X}$ be the matrix of feature representations extracted from real images, and $\mathbf{Y}$ be the matrix of feature representations extracted from generated images.
   - Real images feature matrix: $\mathbf{X}=[x_1,x_2,\ldots,x_n]$
   - Generated images feature matrix: $\mathbf{Y}=[y_1,y_2,\ldots,y_n]$
2. **Compute Statistics**:
   - Mean of real features: $\mu_{\mathbf{X}}=\frac{1}{n}\sum_{i=1}^{n}x_i$
   - Mean of generated features: $\mu_{\mathbf{Y}}=\frac{1}{n}\sum_{i=1}^{n}y_i$
   - Covariance of real features: $\Sigma_{\mathbf{X}}=\frac{1}{n}\sum_{i=1}^{n}(x_i-\mu_{\mathbf{X}})(x_i-\mu_{\mathbf{X}})^T$
   - Covariance of generated features: $\Sigma_{\mathbf{Y}}=\frac{1}{n}\sum_{i=1}^{n}(y_i-\mu_{\mathbf{Y}})(y_i-\mu_{\mathbf{Y}})^T$
3. **Compute Fréchet Distance**:

- Fréchet distance between the two multivariate Gaussian distributions: FID=‖ □□−□□‖ 2+Tr(Σ□+Σ□−2(Σ□Σ□)1/2)FID=‖ μx−μY ‖ 2+Tr(Σx+ΣY−2(ΣxΣY)1/2)

Where ‖ · ‖ ‖ · ‖ denotes the Euclidean norm, and Tr(·)Tr(·) denotes the trace of a matrix.

REMARK : Our experiments demonstrated that the synthetic images generated by the GAN achieved a commendable FID score, indicating a high degree of similarity between the generated and real images. This suggests that the GAN effectively captured the underlying distribution of the minority classes, thereby reducing the class imbalance in our dataset.

2.4 Challenges with Accuracy Improvement

Despite the successful generation of synthetic images and the reduction in class imbalance, we encountered challenges with improving classification accuracy, particularly with class 0. Despite efforts to augment the dataset with synthetic images, the overall accuracy did not show significant improvement, and class 0 continued to exhibit low accuracy.

2.5 Conclusion

In conclusion, while our implementation of a GAN-based approach successfully addressed the imbalance in our galaxy classification dataset and generated realistic synthetic images, we faced difficulties in improving classification accuracy, especially with class 0.

Chapter 3 : Model Deployment

Following the successful training and optimization of our deep learning model, we deployed our best-performing model using Streamlit, a user-friendly web application framework. Leveraging Streamlit's capabilities, we created an interactive website where users can upload an image of a galaxy, and our model will perform real-time classification to determine its morphological type. This website provides a seamless and intuitive interface for users to engage with our model and gain insights into the classification of galaxies. By democratizing access to advanced classification techniques, we aim to empower both amateur astronomers and researchers alike to explore and understand the vast universe of galactic morphology.

Conclusion

In conclusion, this study has demonstrated the efficacy of deep learning techniques, particularly Convolutional Neural Networks (CNNs) and Generative Adversarial Networks (GANs), in automating the classification of galaxy images. Through extensive experimentation and hyperparameter tuning, we developed a deep CNN architecture tailored to galaxy classification tasks, achieving commendable accuracy in distinguishing between different morphological types of galaxies. By leveraging GANs for data augmentation, we enhanced the robustness and generalization capability of our model, enabling it to effectively handle variations in galaxy images. Additionally, the deployment of our model through Streamlit resulted in the creation of an interactive website, providing users with a user-friendly platform to perform real-time classification of galaxy images. This research contributes to advancing the field of astrophysics by offering a reliable and efficient automated solution for galaxy classification, paving the way for further exploration and understanding of the universe's vast celestial objects. Looking ahead, future research could focus on refining the model architecture, exploring additional data augmentation techniques,

and extending the application to larger and more diverse datasets, ultimately advancing our knowledge of galaxy morphology and evolution.