Wafer Defect Generation

Part 1: Using Cycle GAN

1. Steps required to run the code:

# Install Dependencies:

Make sure you have the required libraries installed.

# Prepare Dataset:

Make sure you have a dataset in the specified dataroot directory. In this case, it's set to "data/training/Edge-Loc". The dataset should be structured in a way that is compatible with ImageFolder from torchvision.

# Set up Environment:

Ensure you have a working environment with GPU support if available since the code checks for GPU availability (cuda). You can adjust the device variable accordingly.

# Run the Code:

Copy the provided code into a Python script or Jupyter Notebook. Execute the script or run each cell in the Jupyter Notebook sequentially.

# Monitor Training:

The training loop will print information about the training progress, including the generator and discriminator losses. The generated images will be saved every 50 epochs in the "generated\_imgs\_edgeloc" directory.

# Results:

After training, you can check the generated images in the "generated\_imgs\_edgeloc" directory.

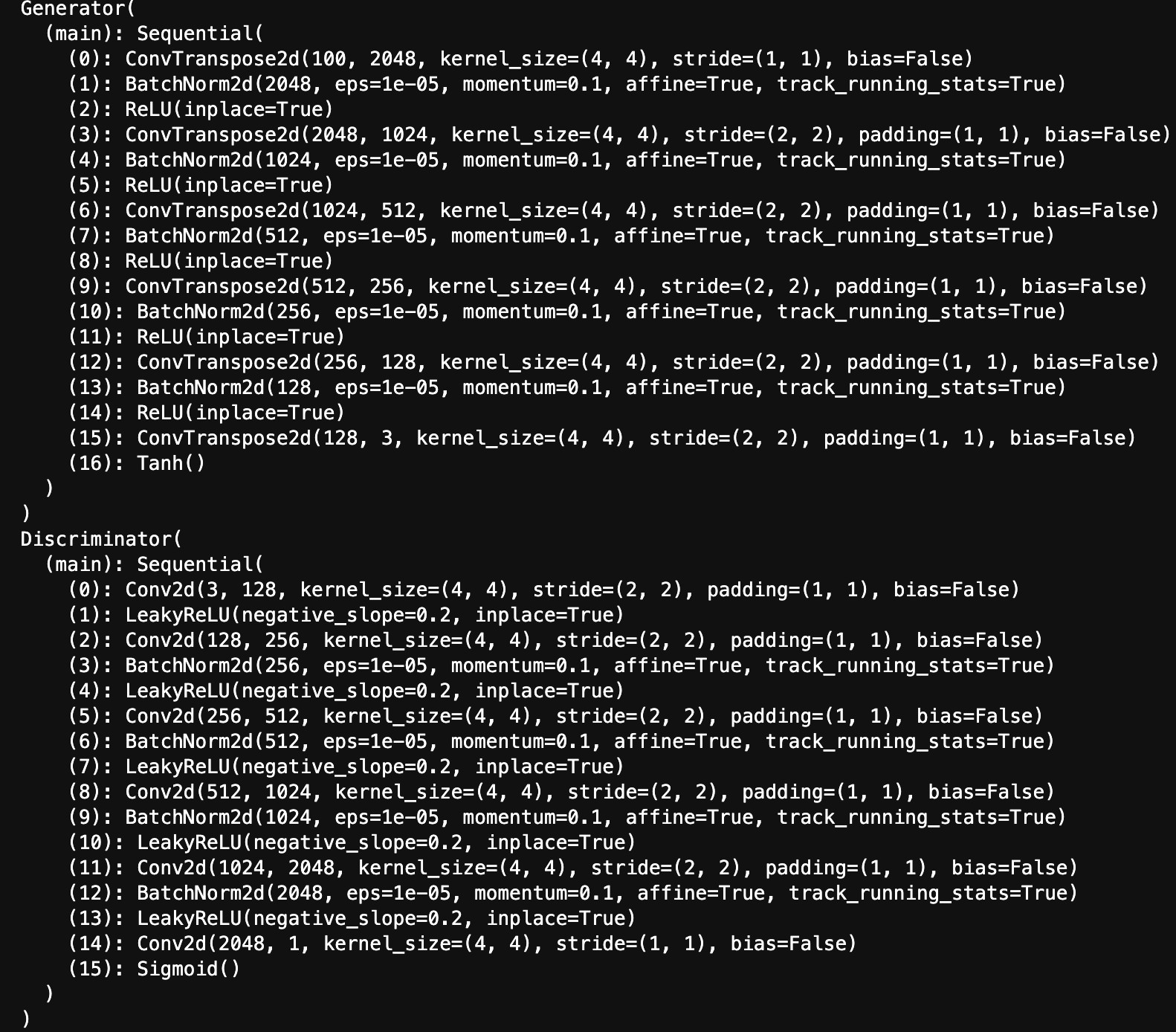
The final generator model will be saved in the current directory with the name "generator\_final.pth".

Additionally, the generator model will be saved at the end of each epoch as "generator\_epoch\_{epoch}.pth".

# Visualization:

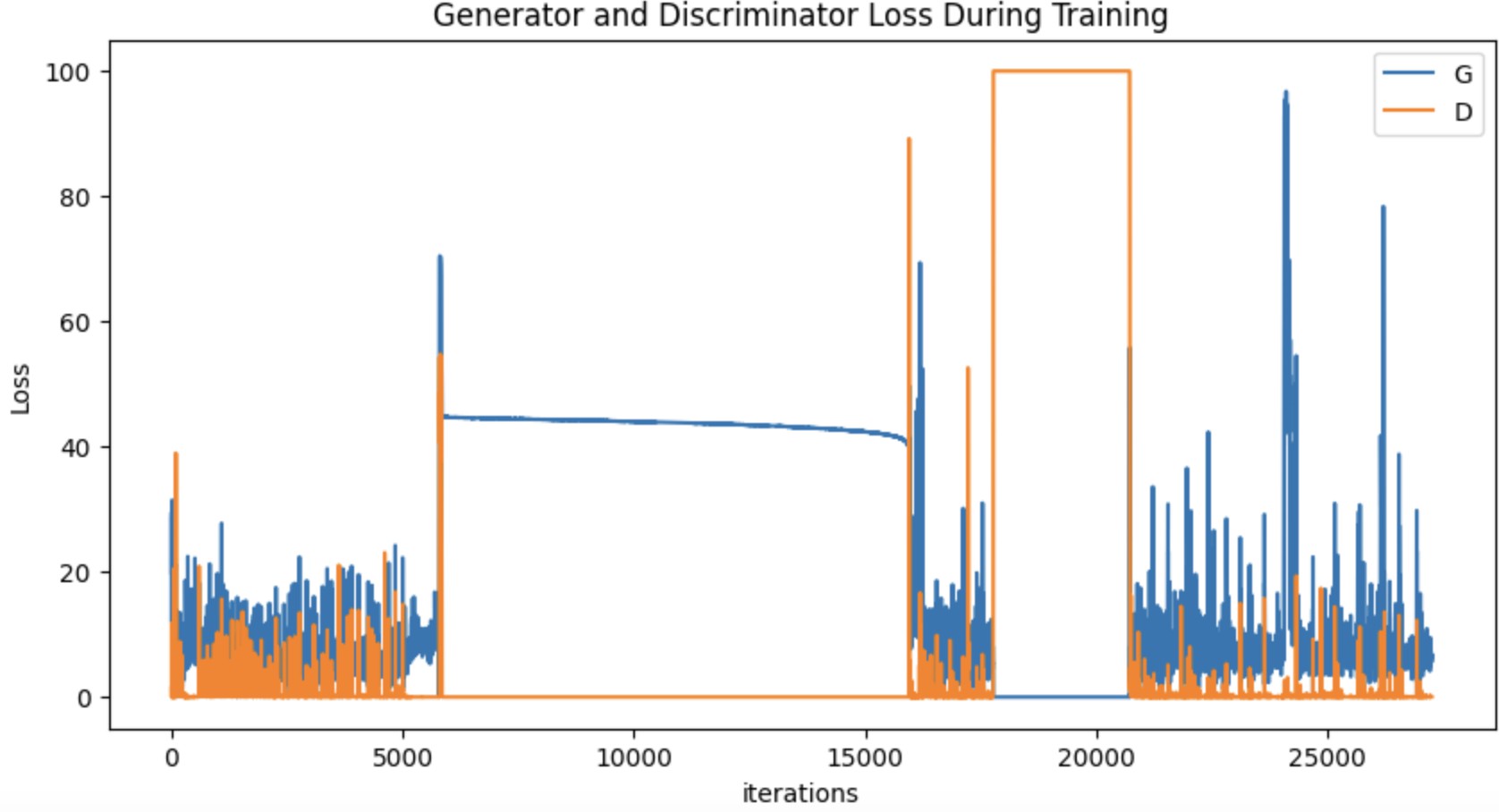
A plot showing the generator and discriminator losses during training will be displayed at the end.

1. Screenshot of the summary of the generator and discriminator architecture printed from your codes:

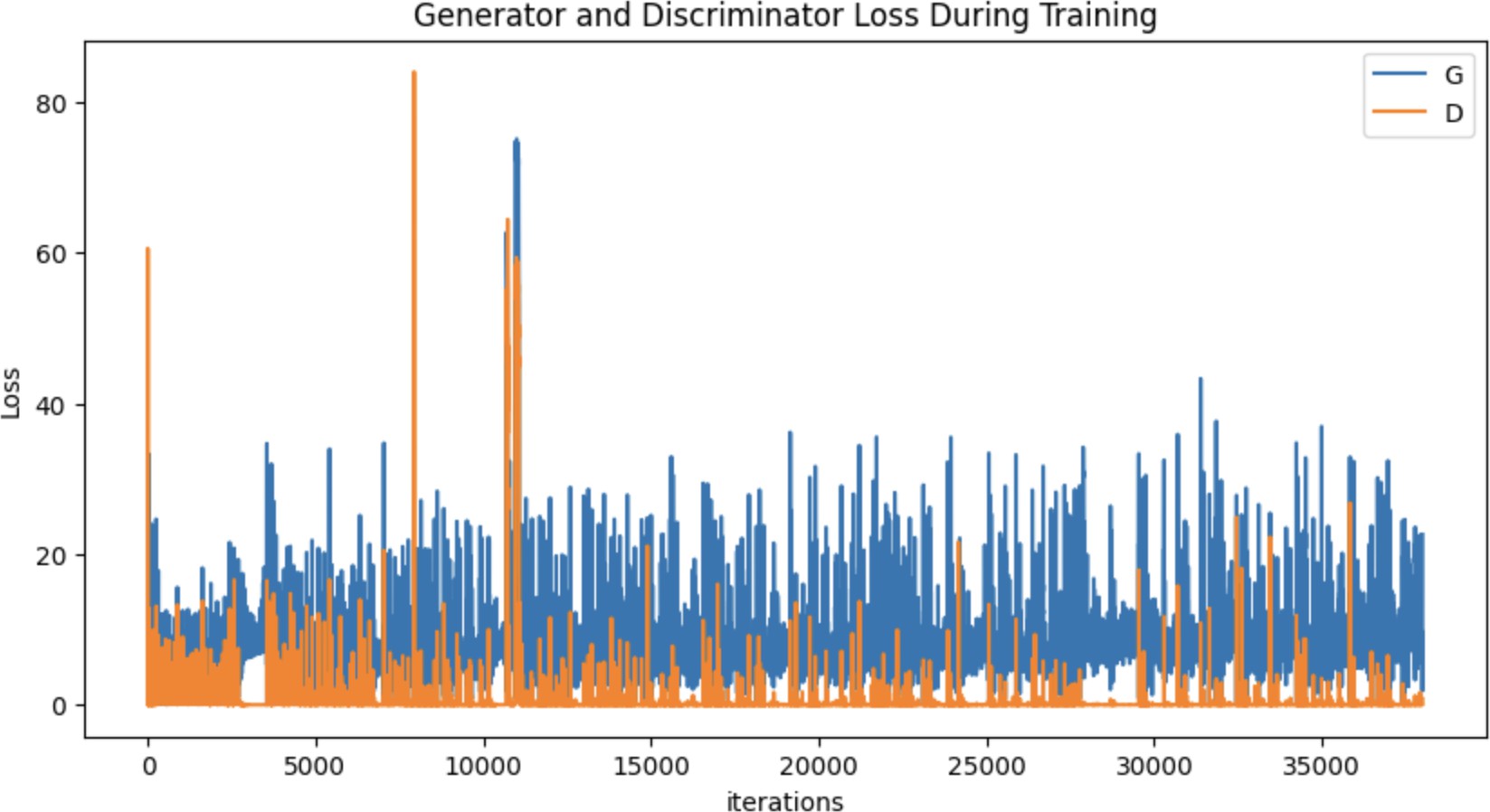


1. Generated discriminator and generator loss graphs:

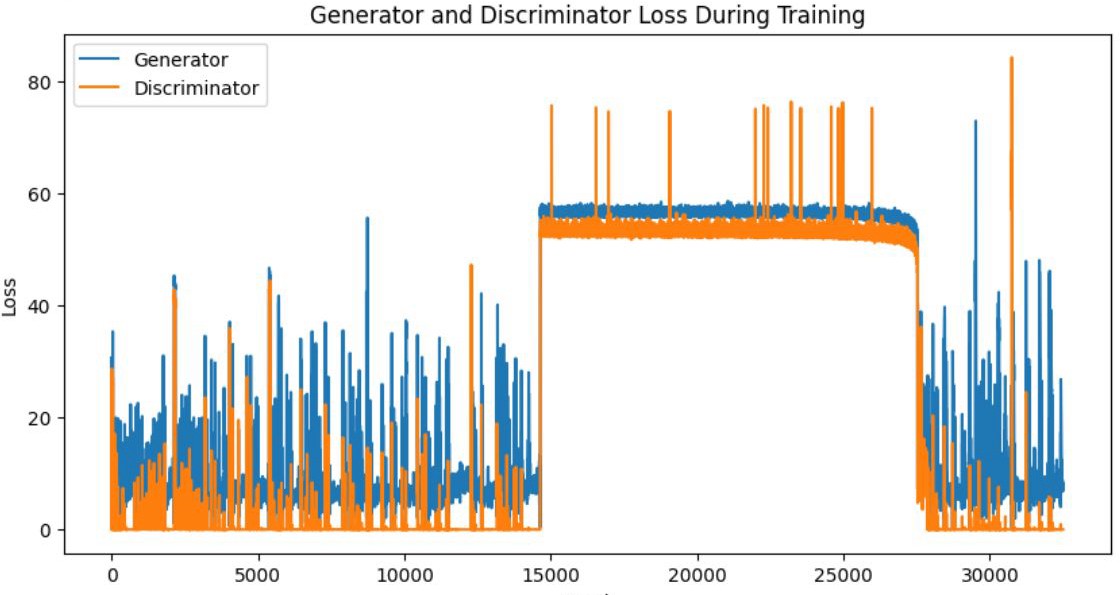
**Center** :



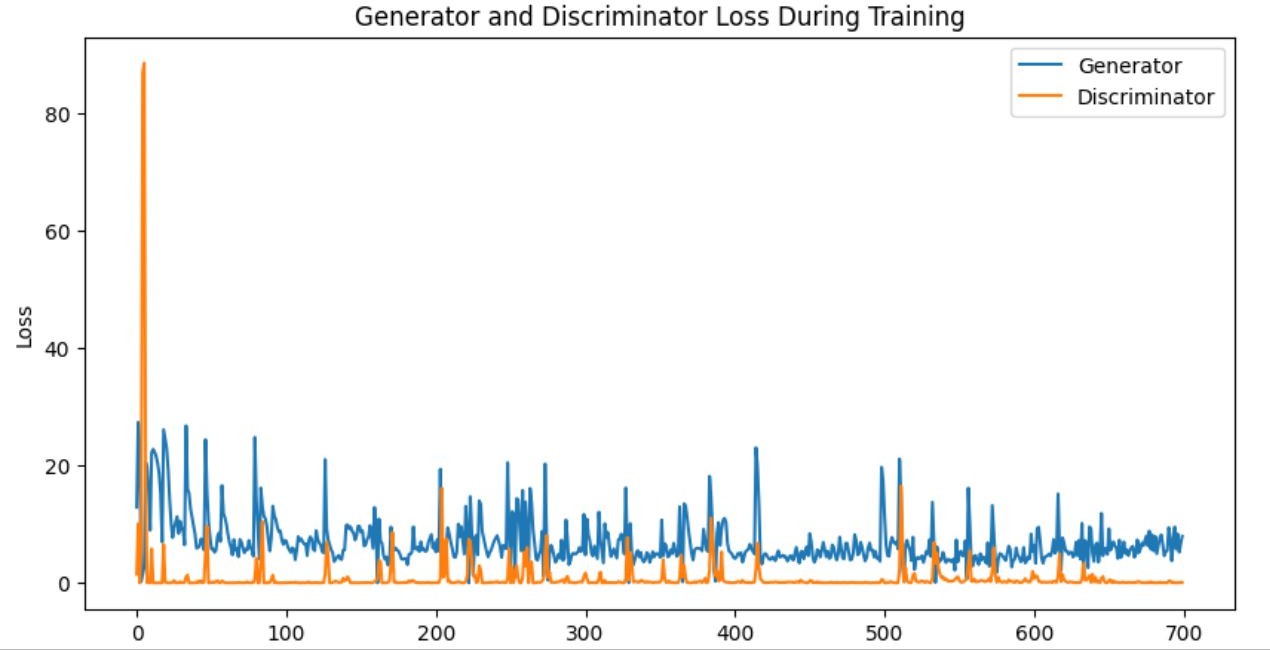
**Edge-Loc:**



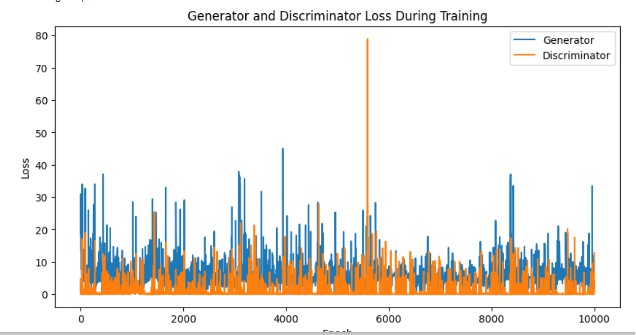
**None:**



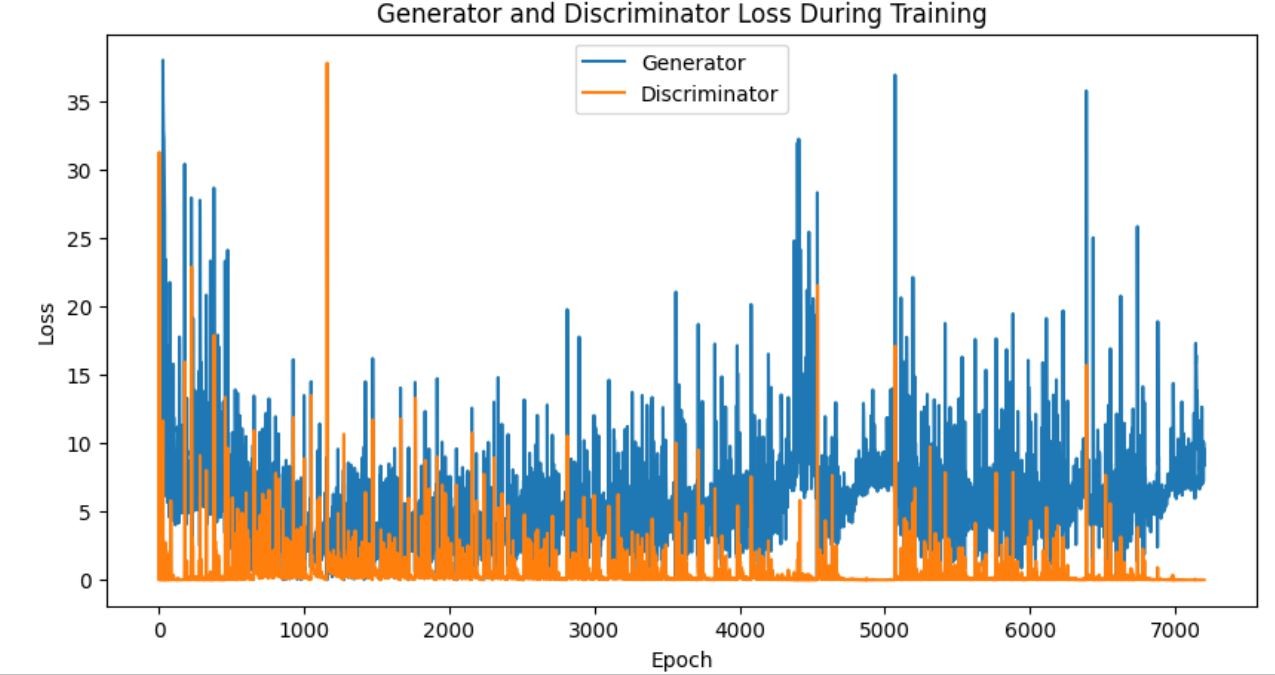
**Near Full:**



**Random:**



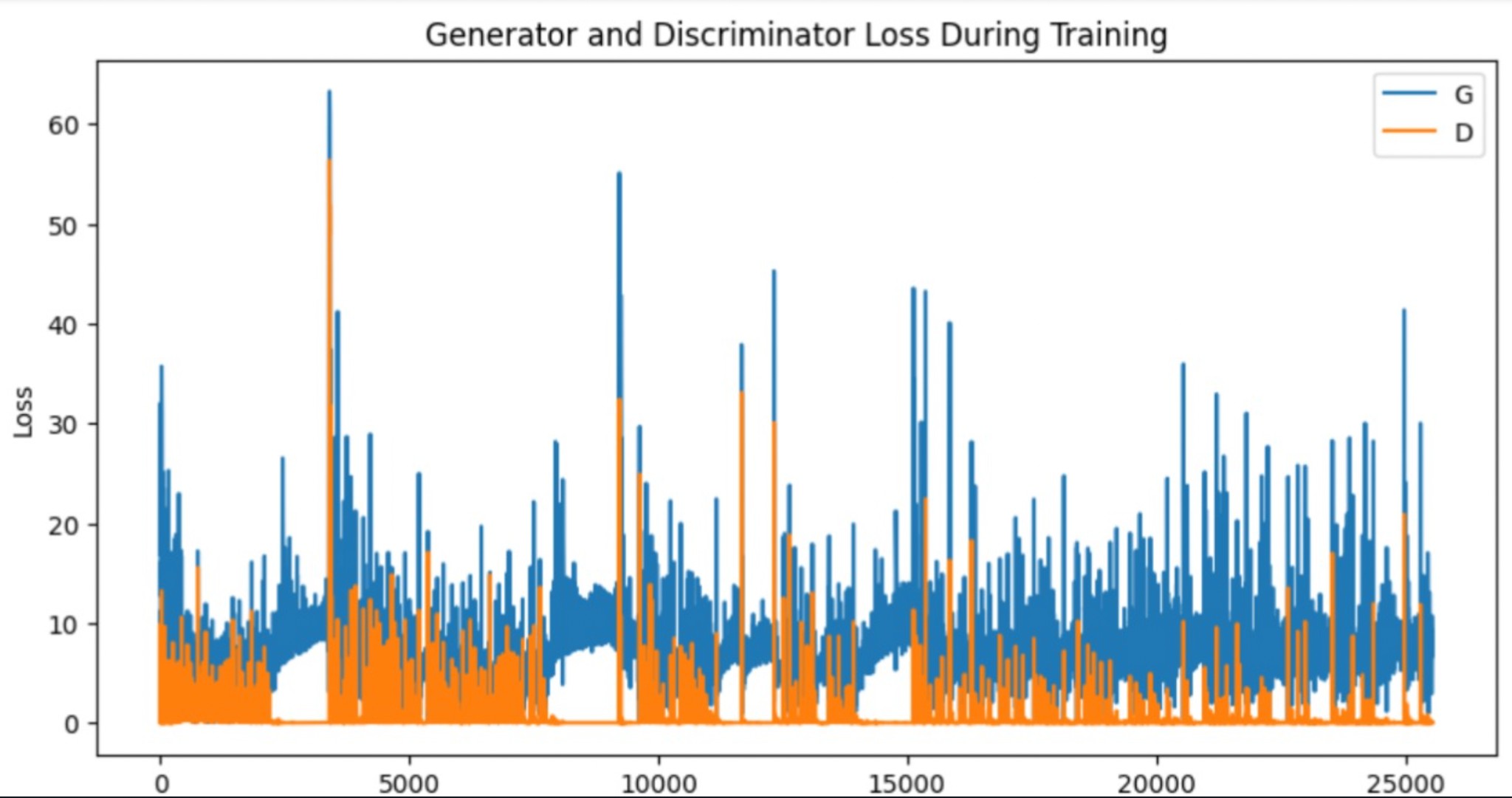
**Scratch:**



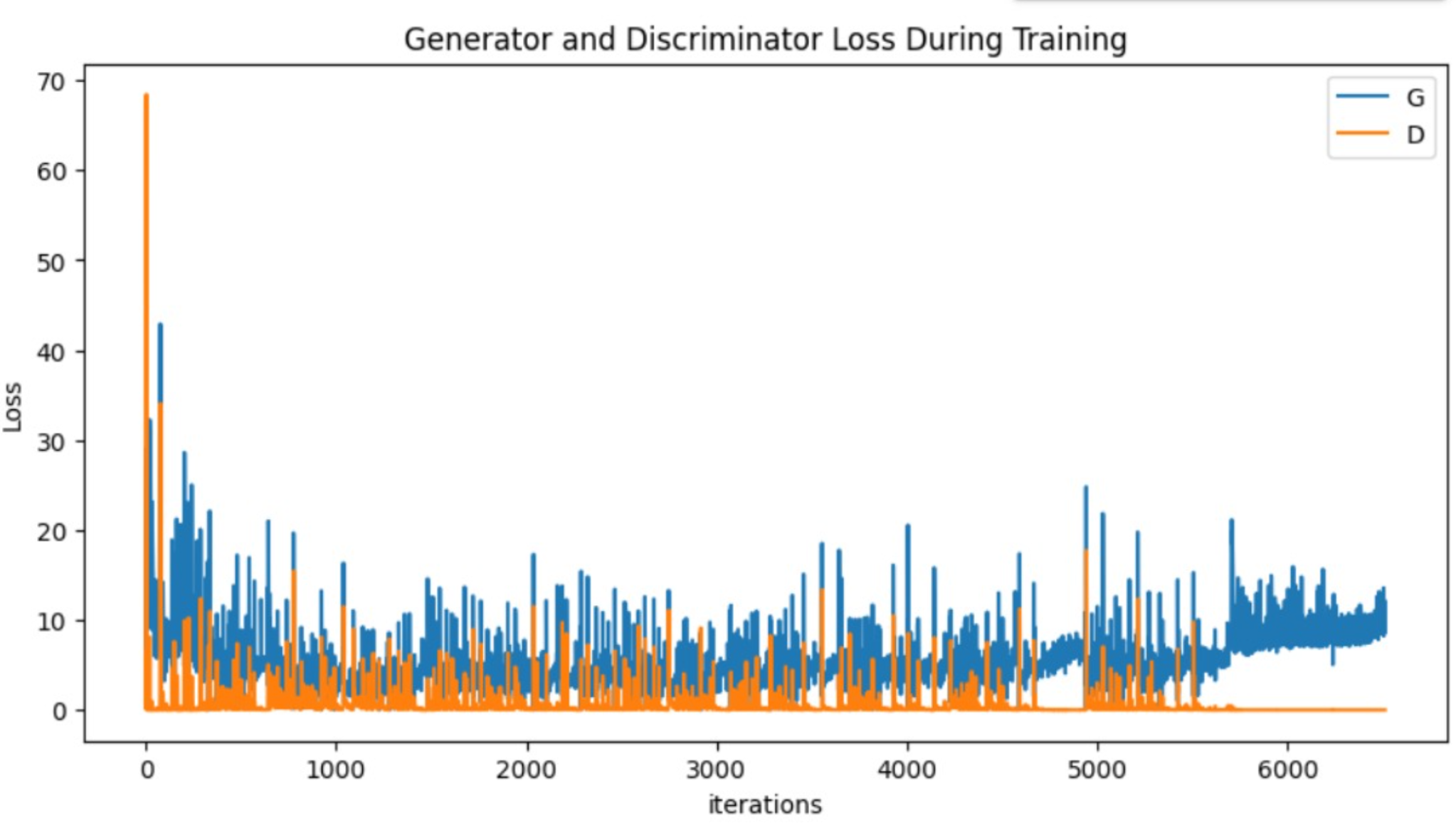
**Edge Ring:**



**Loc:**

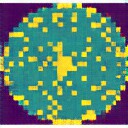
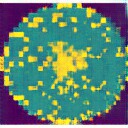
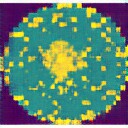


**Donut:**

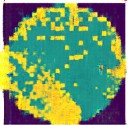
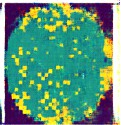
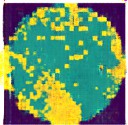


In a Generative Adversarial Network (GAN), the optimization process involves training the generator and discriminator simultaneously in a competitive manner. The goal is to find a balance where the generator creates realistic data, and the discriminator becomes proficient at distinguishing between real and generated samples. The optimization process continues through multiple epochs, allowing the generator and discriminator to improve iteratively. The key challenge is finding the right balance between the generator and discriminator, as improving one may adversely affect the other. Hyperparameters such as learning rates, network architectures, and training strategies play a crucial role in the optimization process.

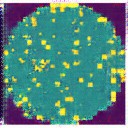
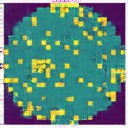
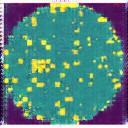
4 .Samples of best generated images for each pattern: Center:



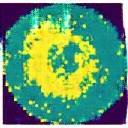
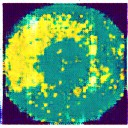
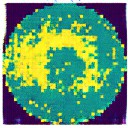
Edge Loc:



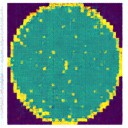
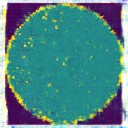
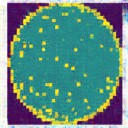
None:



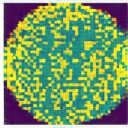
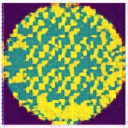
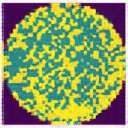
Donut:



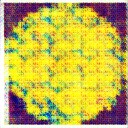
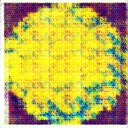
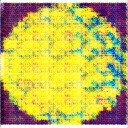
Edge Ring:



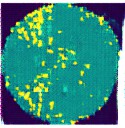
Random:

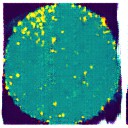
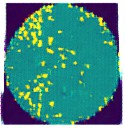


Near Full:

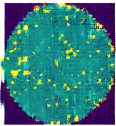
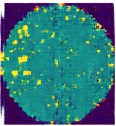
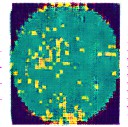


Scratch:





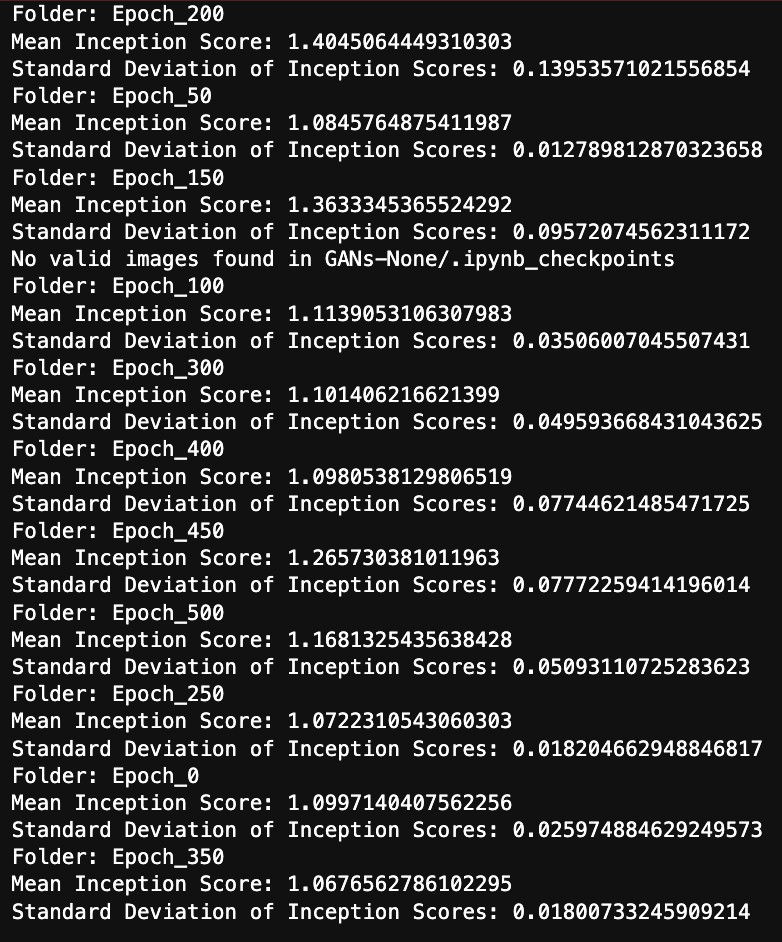
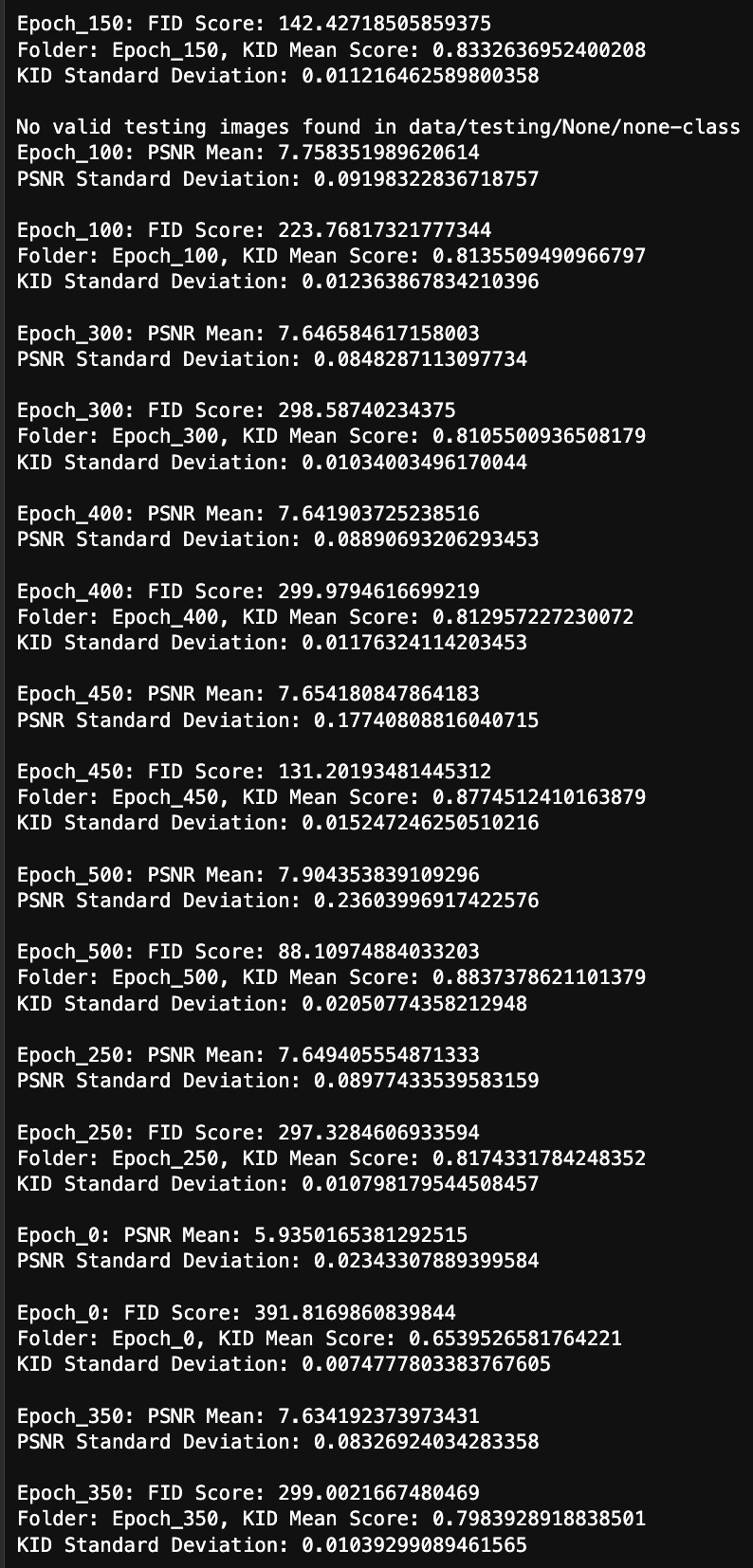
Loc:

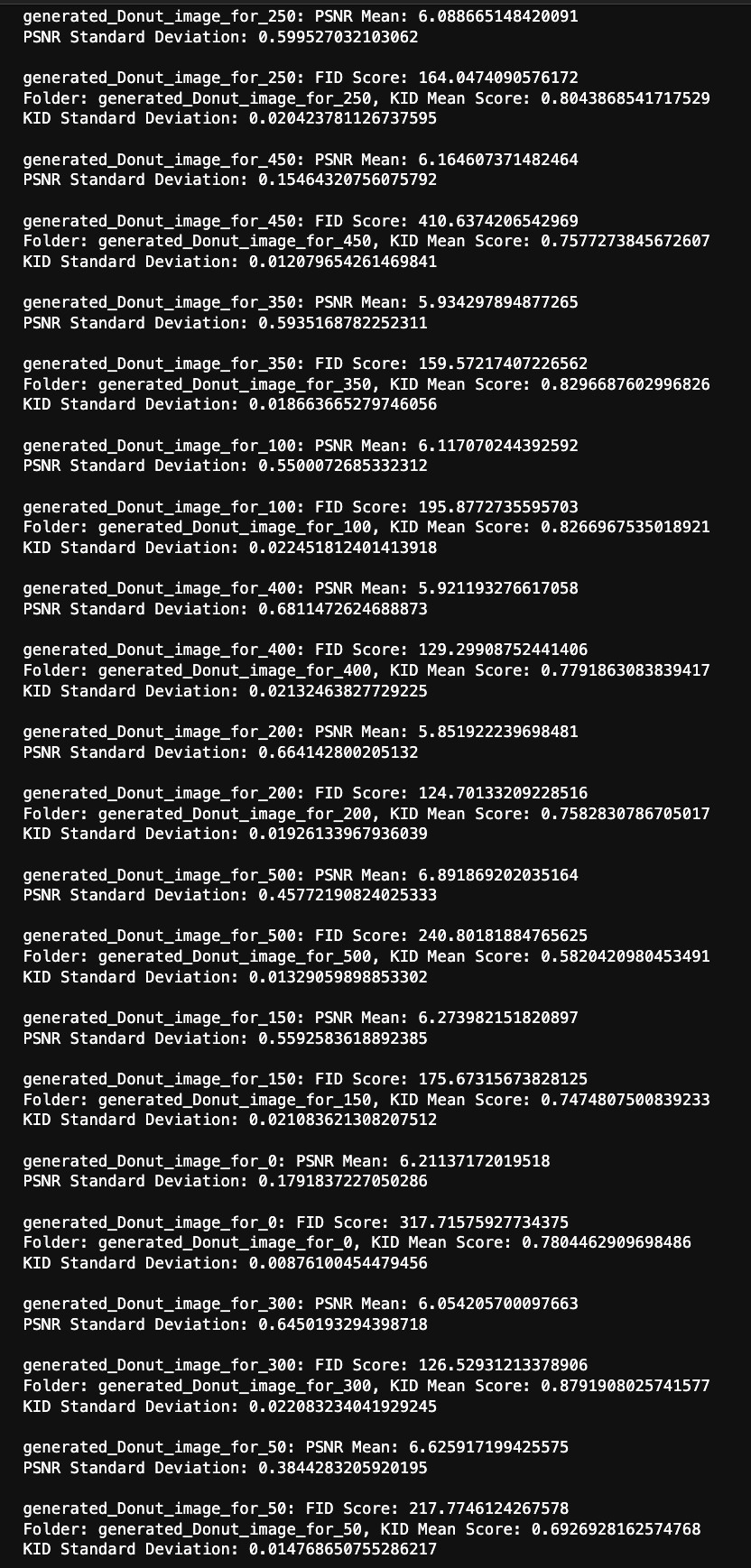
1. Evaluation metrics Results

PSNR, IS, KID and FID scores:

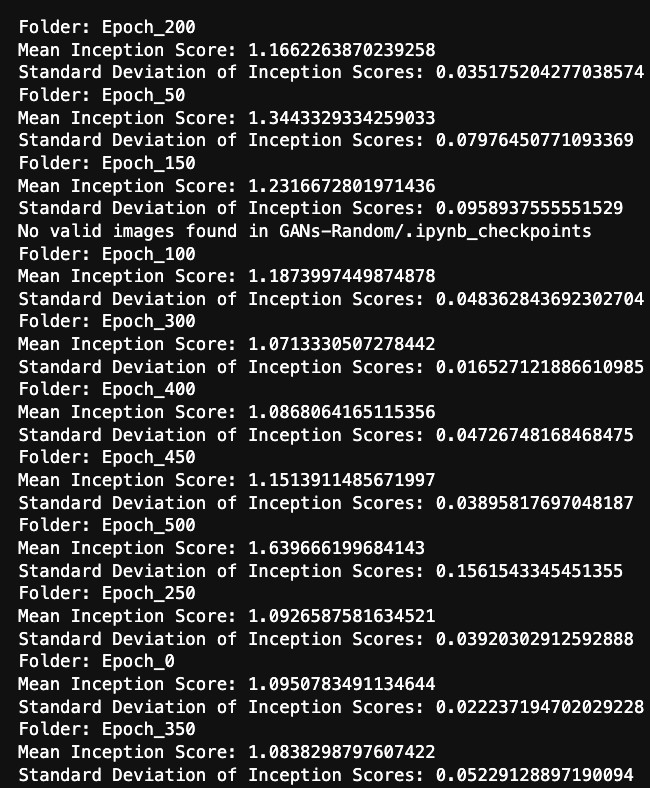
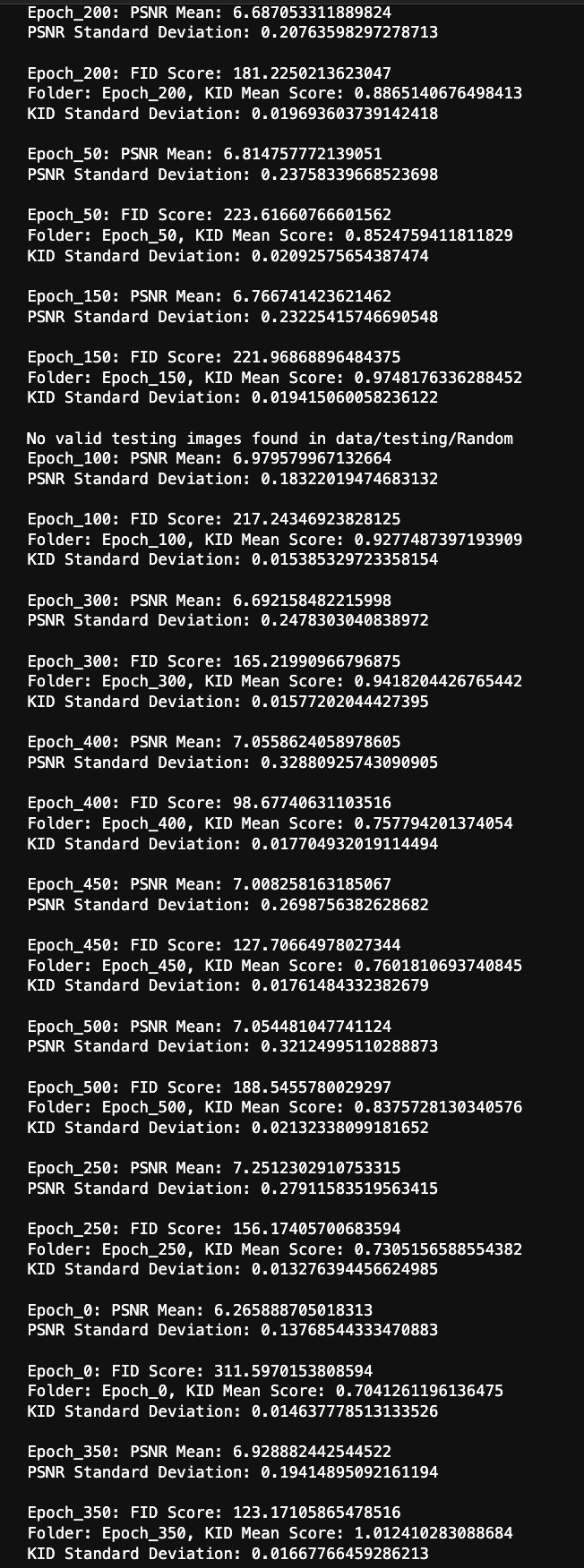
**None:**



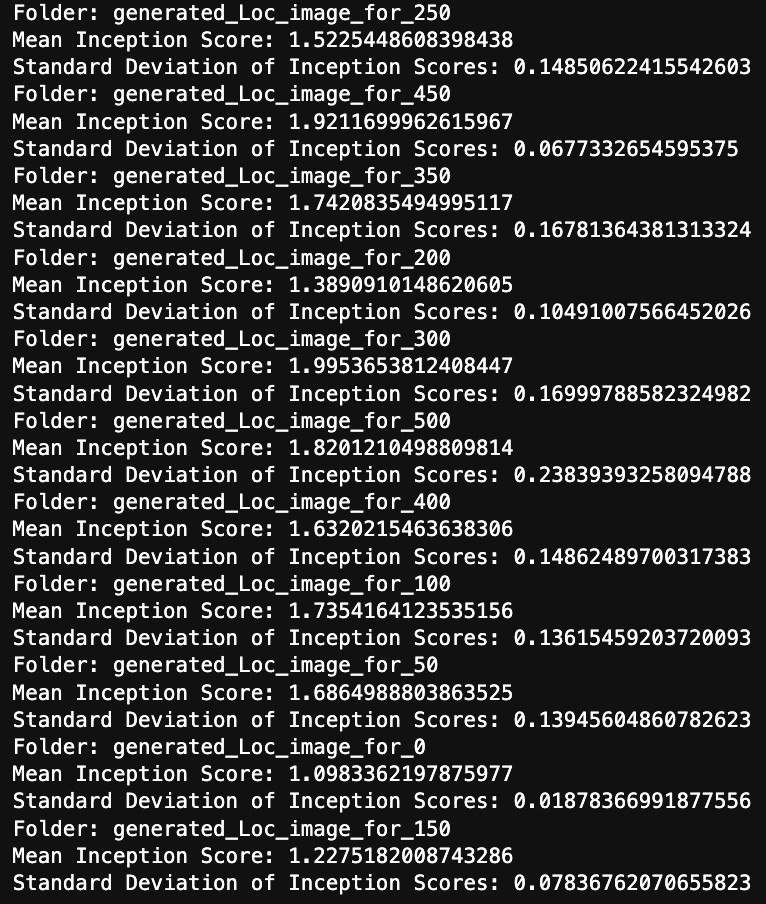
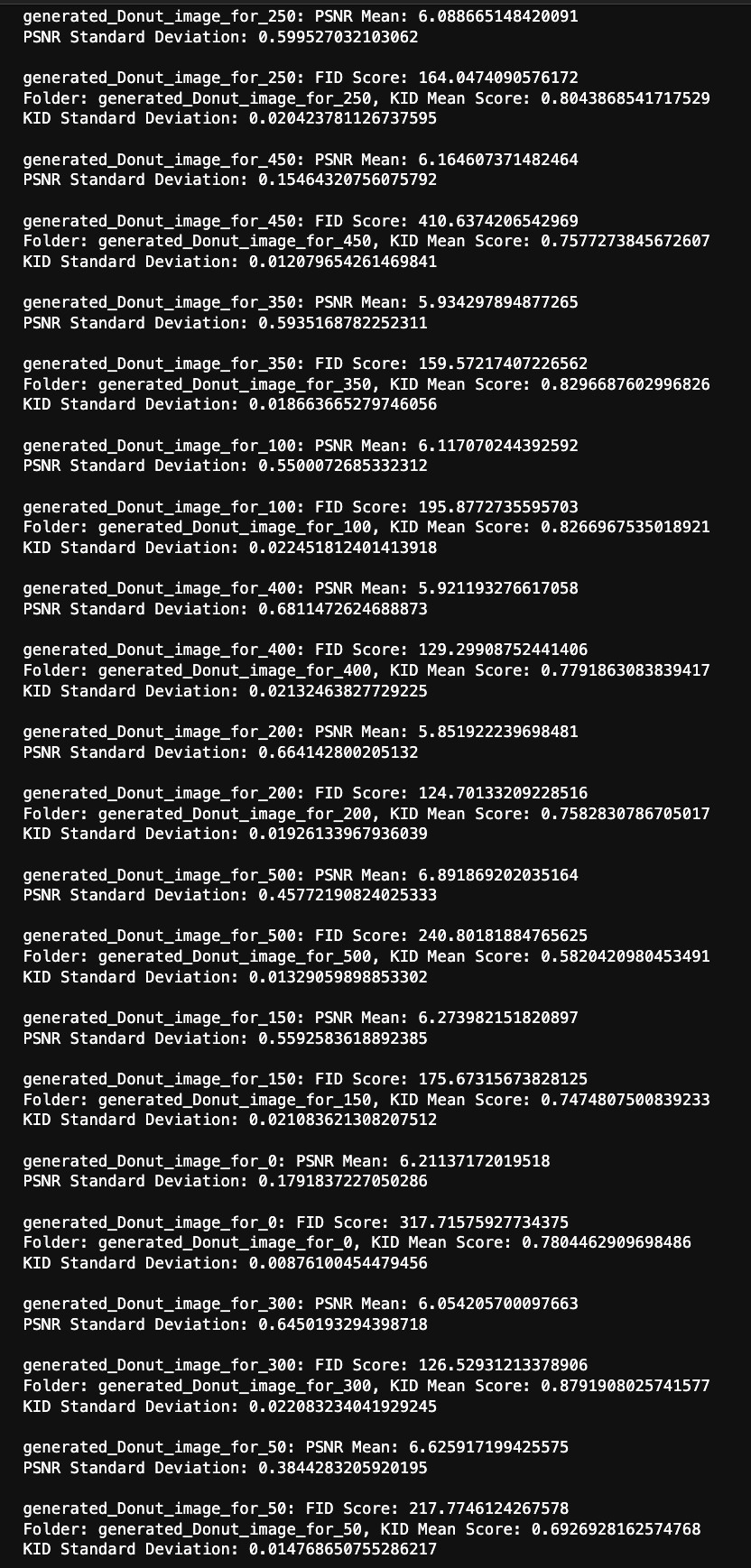
**Donut:**



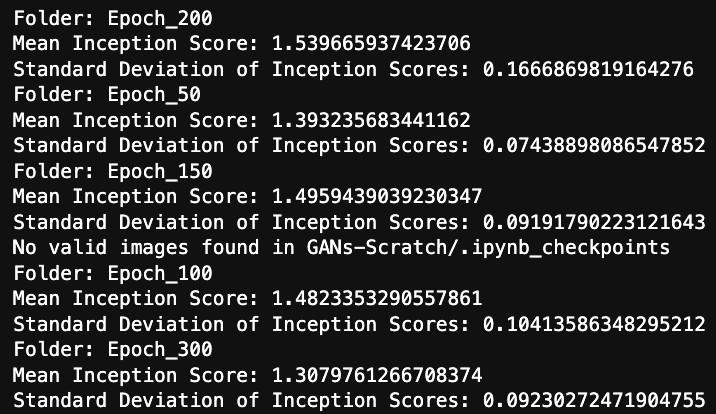
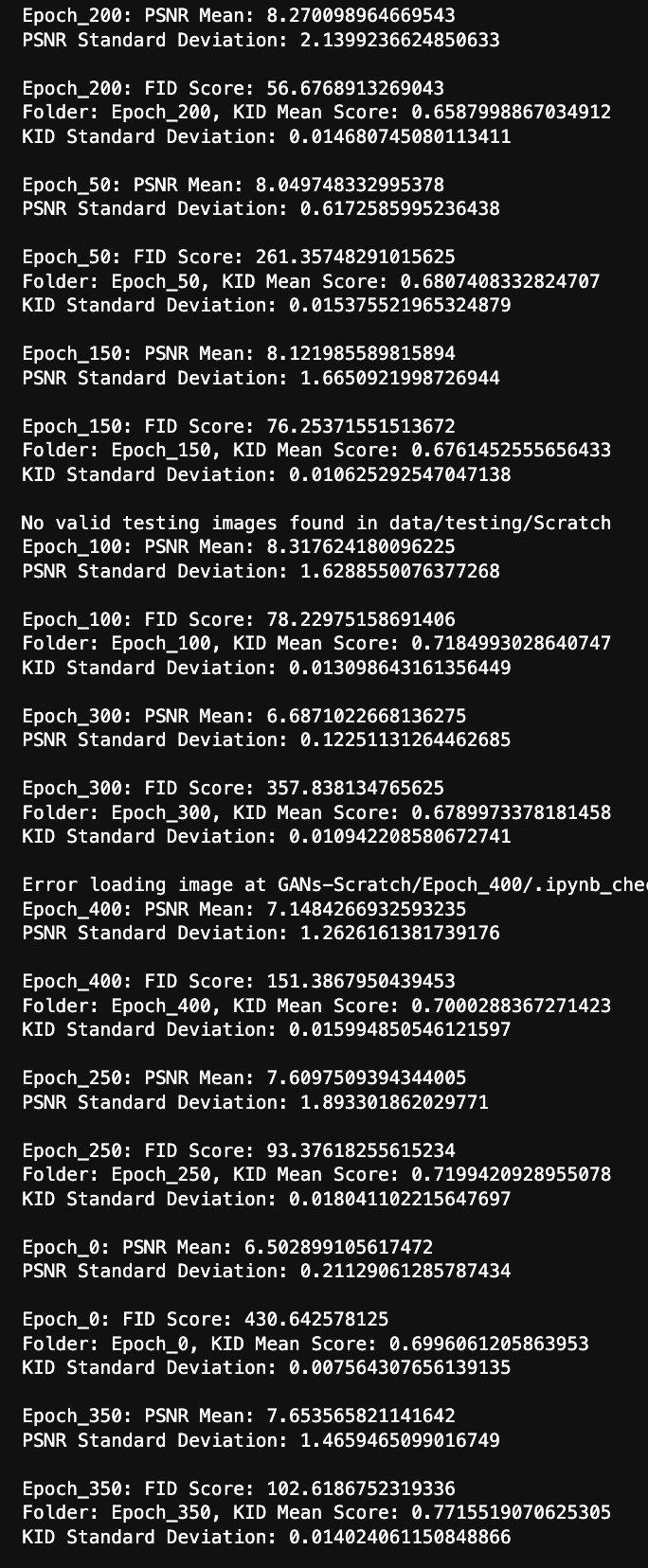
**Random:**



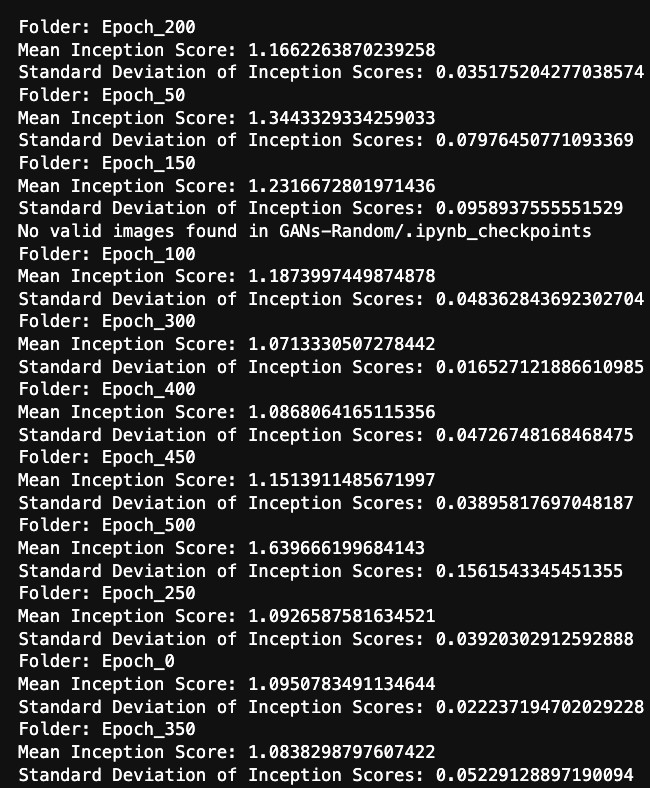
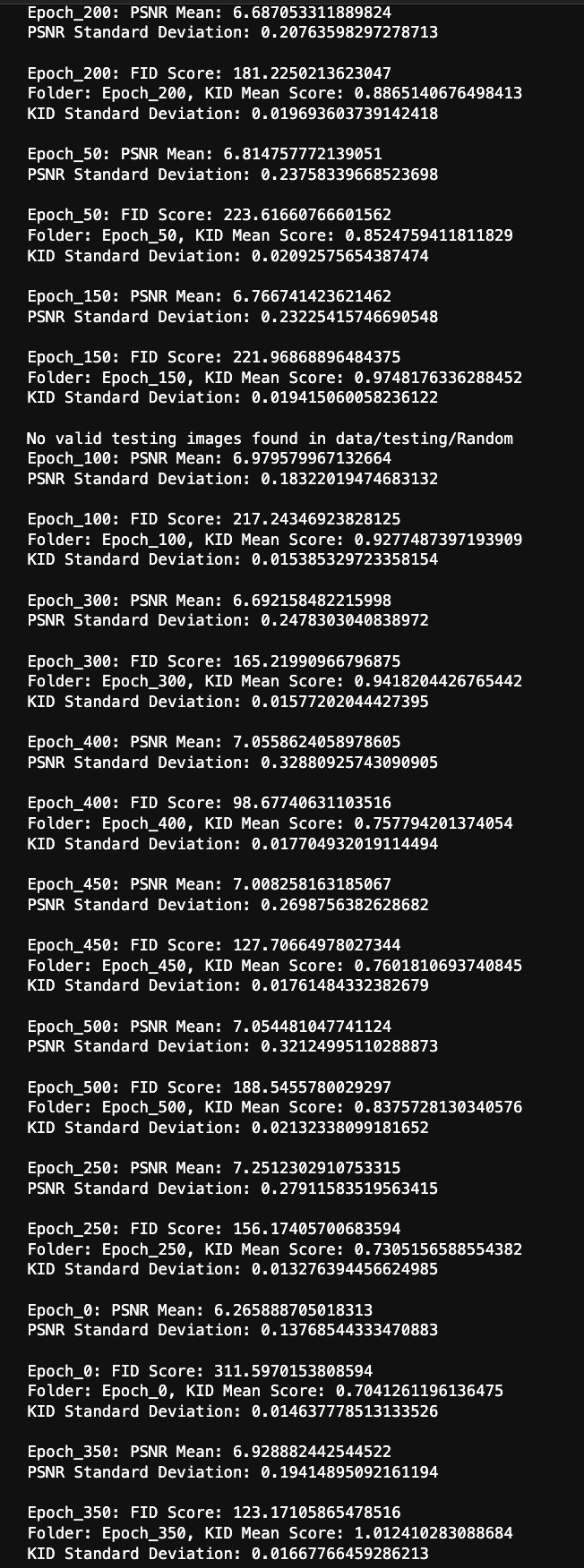
**Loc:**



**Scratch:**



**EdgeLoc:**



1. Discussion of the results based on both evaluation metrics and loss graphs:

The evaluation of Generative Adversarial Networks (GANs) involves assessing both quantitative metrics and qualitative aspects. In your provided code, you are using the following evaluation metrics:

# Frechet Inception Distance (FID):

* + - **Purpose:** Measures the similarity between the distribution of real and generated images in feature space.
    - **Interpretation:** Lower FID scores indicate better similarity between real and generated distributions.
    - **Discussion:** A low FID score suggests that your GAN is generating images that are close to real images in terms of both quality and diversity.

# Kernel Inception Distance (KID):

* + - **Purpose:** Similar to FID, but using kernelized embeddings.
    - **Interpretation:** Similar to FID, lower KID scores indicate better performance.
    - **Discussion:** Consistency between FID and KID scores reinforces the reliability of your GAN's performance.

# Peak Signal-to-Noise Ratio (PSNR):

* + - **Purpose:** Measures the quality of reconstructed images by comparing them to the ground truth.
    - **Interpretation:** Higher PSNR values indicate better image quality.
    - **Discussion:** Monitoring PSNR helps assess the visual fidelity of generated images. A high PSNR suggests that the generated images are close to the ground truth in terms of pixel-wise similarity.

# Inception Score:

* + - **Purpose:** Measures the quality and diversity of generated images.
    - **Interpretation:** Higher Inception Scores indicate better quality and diversity.
    - **Discussion:** A high Inception Score implies that the generated images are both realistic and diverse.
  + **Consistency between Metrics:** If FID, KID, PSNR, and Inception Score show consistent trends across different folders, it indicates a robust and reliable evaluation. Consistency suggests that the model is performing consistently across different aspects of image quality and diversity.
  + **FID and KID Scores:** Low FID and KID scores suggest that the GAN is successful in generating images that are close to the distribution of real images. This is a positive sign of the GAN's capability to capture the underlying data distribution.
  + **PSNR Values:** High PSNR values imply that the generated images closely resemble the ground truth images. This is indicative of pixel-wise similarity between generated and real images.
  + **Inception Score:** High Inception Scores suggest that the generated images are both of high quality and exhibit diversity. A balance between quality and diversity is crucial for a successful GAN.