Image Inpainting For Irregular Holes Using Partial Convolutions

Group Members

- **Name:** Anvit Gupta
  Email: anvit_g@cs.iitr.ac.in
  Phone: +91-9462011044
  Contribution: Explored Diffusion based Image Inpainting (Latent Image generation), Text guided Image Inpainting, Datasets and implementation with Diffusion Models explored.

- **Name:** Sarvasva Gupta
  Email: sarvasva_g@cs.iitr.ac.in
  Phone: +91-7989904811
  Contribution: Explored "Patch based Image Inpainting with GANs" paper, Datasets Explored, Implemented the training loop for model training.

- **Name:** Souvik Karmakar
  Email: souvik_k@cs.iitr.ac.in
  Phone: +91-8642924659
  Contribution: Explored Partial Convolution based Approaches and explored evaluation metrics.

1. What is Partial Convolution?

Traditional deep learning-based image inpainting methods use standard convolutional networks that operate over the entire image, including both valid (uncorrupted) pixels and invalid (masked) regions. These methods typically replace missing pixel values with a constant like the mean of the dataset, and then apply convolutions over the whole image. However, this approach often results in artifacts such as color mismatches, blurriness, and unnatural textures.

To overcome these issues, Guilin Liu et al. proposed **Partial Convolutions**, where the convolution operation is masked and renormalized so that it only depends on valid pixels. This avoids contaminating the convolution with placeholder values.

Formally, a partial convolution operation at a location is defined as:

$$x = \begin{cases} \frac{\mathbf{W}^T(\mathbf{X} \odot \mathbf{M})}{\text{sum}(\mathbf{M})} + b, & \text{if sum}(\mathbf{M}) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where:

- $\mathbf{W}$ is the convolution filter weight.
- b is the bias.
- $\mathbf{X}$ is the input feature matrix.
- $\mathbf{M}$ is a binary mask indicating valid pixels.
- $\odot$ is the element-wise multiplication operator.

This operation ensures that the output at each step only depends on valid pixels, and it is scaled appropriately based on the number of valid pixels involved.

After the convolution, the mask is automatically updated for the next layer using the rule:

$$m = \begin{cases} 1, & \text{if sum}(\mathbf{M}) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

This ensures that the network can progressively recover more of the missing image as more regions are reconstructed.

Key contributions of this work include:

- Introduction of the partial convolutional layer with an automatic mask update.
- Demonstration of successful use of U-Net style architectures with partial convolutions for image inpainting.
- Creation of a new dataset with large irregular holes for training and evaluation.

2. Loss Functions Used

The loss functions in the partial convolution paper are carefully crafted to ensure both pixel-level reconstruction accuracy and perceptual consistency. These include standard pixel-wise L1 losses as well as high-level feature losses.

a) Per-pixel L1 Losses

- **Hole Loss** (L_{hole}): Applies L1 loss only over the pixels inside the masked region:

$$L_{hole} = \frac{1}{N_{I_{gt}}} \sum (1 - M) \cdot |I_{out} - I_{gt}|$$

- **Valid Loss** (L_{valid}): Applies L1 loss only over the known valid pixels:

$$L_{valid} = \frac{1}{N_{I_{gt}}} \sum M \cdot |I_{out} - I_{gt}|$$

b) Perceptual Loss ($L_{perceptual}$)

This compares deep features from a pretrained VGG-16 model. It includes both raw output (I_{out}) and a composed image (I_{comp}) where valid pixels are copied from the ground truth:

$$L_{perceptual} = \sum_{p} \frac{1}{N_{I_{gt}^p}} \left(\|I_{out}^p - I_{gt}^p\|_1 + \|I_{comp}^p - I_{gt}^p\|_1 \right)$$

c) Style Loss

Style loss is based on Gram matrices computed from deep features. It is computed for both I_{out} and I_{comp}:

$$L_{style}^{out} = \sum_{p} \frac{1}{C_p^2} \|G(I_{out}^p) - G(I_{gt}^p)\|_1$$

$$L_{style}^{comp} = \sum_{p} \frac{1}{C_p^2} \|G(I_{comp}^p) - G(I_{gt}^p)\|_1$$

Where $G(\cdot)$ denotes the Gram matrix of feature maps.

d) Total Variation Loss (L_{tv})

This loss enforces spatial smoothness around the hole boundaries:

$$L_{tv} = \sum_{(i,j) \in R} |I_{comp}^{i+1,j} - I_{comp}^{i,j}| + |I_{comp}^{i,j+1} - I_{comp}^{i,j}|$$

e) Total Loss

The total loss used to train the model is a weighted sum:

$$L_{total} = L_{valid} + 6L_{hole} + 0.05L_{perceptual} + 120(L_{style}^{out} + L_{style}^{comp}) + 0.1L_{tv}$$

These combined loss functions help the network generate inpainted images that are both structurally accurate and visually plausible.

In summary, the partial convolution technique allows the model to focus on real data during training and inference, thereby producing higher quality inpainting results.

3. Architecture Explanation

- The core of our model is a **U-Net-like architecture** with **Partial Convolutions**, specifically designed for image inpainting tasks.
- Each encoder and decoder block is built using the `PConvActiv` class, which incorporates partial convolution followed by optional BatchNorm and activation layers.

- The encoder path progressively downsamples the input and extracts features, while the decoder upsamples and fuses these features via skip connections.
- The `PartialConv2d` layer performs masked convolutions, where only valid (non-hole) pixels contribute to the convolution and are renormalized.
- The mask is updated at every layer to track which regions have been filled or modified.
- `UpsampleConcat` is used in the decoder to upsample decoder features and concatenate them with corresponding encoder features and masks.
- The model supports freezing BatchNorm layers in the encoder during fine-tuning to stabilize learning.
- The entire model is trained end-to-end using masked images and their corresponding ground truth.

4. Dataset and Preprocessing

- We used the **Aerial Landscape Dataset** from Kaggle, containing a large collection of 256x256 images from various geographical scenarios.
- All images were combined into a single folder.
- The dataset was split into training and testing sets such that each class was equally represented in both splits.
- We generated **random irregular masks** for each image using a custom script.
  - Unlike previous works that typically use regular rectangular masks, we generated masks with diverse shapes and sizes.
  - This approach ensures that the model learns to handle a wide variety of real-world occlusion scenarios.
  - Irregular masks more closely simulate natural corruptions or obstructions in images compared to simple rectangular blocks.
- Each image-mask pair was used as input for model training.
- The model was trained on the masked images along with their respective masks.

5. Evaluation Metrics

1. PSNR – Peak Signal-to-Noise Ratio

What is it?
PSNR measures the reconstruction quality of an image by comparing the similarity between

the original (ground truth) and the reconstructed (inpainted) image at the pixel level. It is derived from the Mean Squared Error (MSE) between the two images:

$$\text{MSE} = \frac{1}{WH} \sum_{i=1}^{W} \sum_{j=1}^{H} [I(i,j) - K(i,j)]^2 \tag{3}$$

Where:

- I: Original image
- K: Reconstructed/inpainted image
- W, H: Width and height of the image

Then, PSNR is calculated as:

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{\text{MSE}} \right) \tag{4}$$

Where MAX_I is the maximum possible pixel value of the image (usually 255 for 8-bit images).

Range:
Min: 0 dB
Max: theoretically infinite (capped by floating-point precision)

Desired Value:

- >25 dB: Good quality
- >40 dB: Excellent, near-indistinguishable
- <20 dB: Poor quality (visible noise/artifacts)

2. SSIM – Structural Similarity Index Measure

What is it?
SSIM compares two images based on luminance, contrast, and structure—factors important to human visual perception.

The formula is:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{5}$$

Where:

- μ_x, μ_y: Mean intensities
- σ_x^2, σ_y^2: Variances
- σ_{xy}: Covariance between the two images

- C_1, C_2: Small constants to stabilize the division

Range:
Min: -1 (theoretically), but in practice: 0
Max: 1
Desired Value:

- >0.9: Excellent quality
- 0.75-0.9: Good quality
- <0.6: Significant quality degradation

6. Results

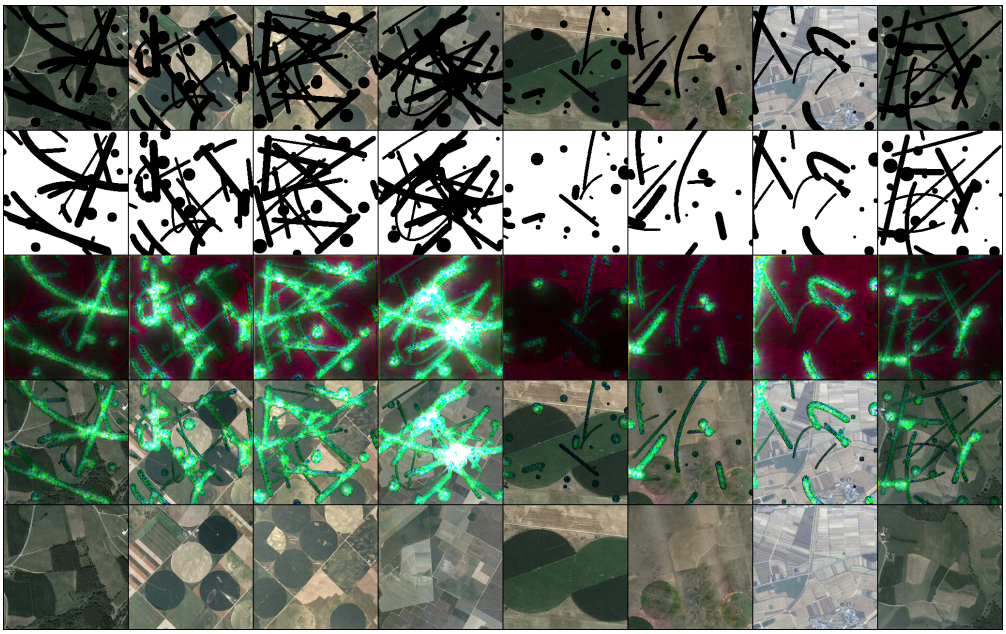


Figure 1: Predictions after 0 steps

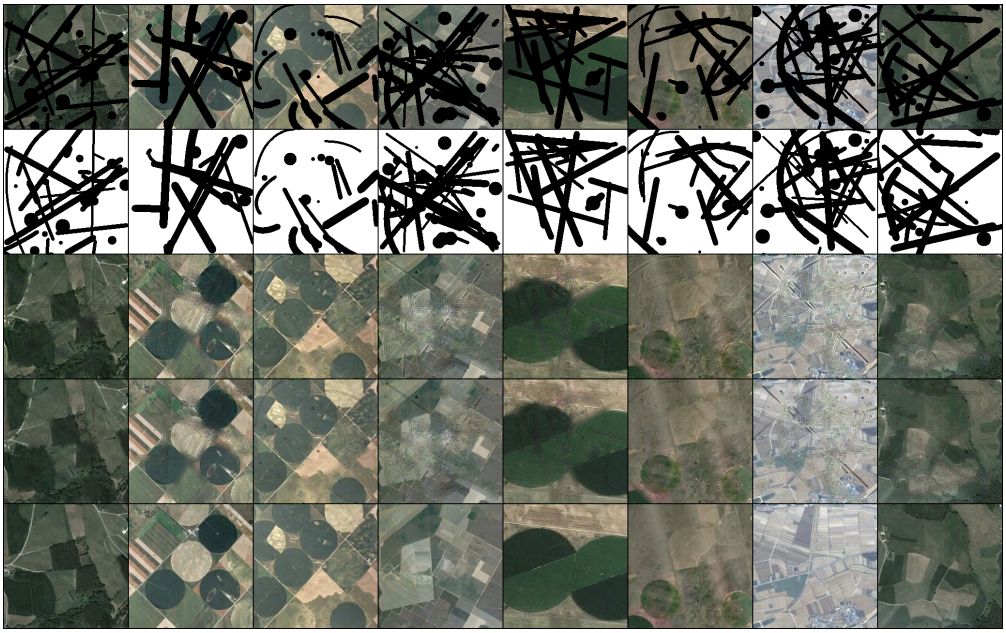


Figure 2: Predictions after 799 steps

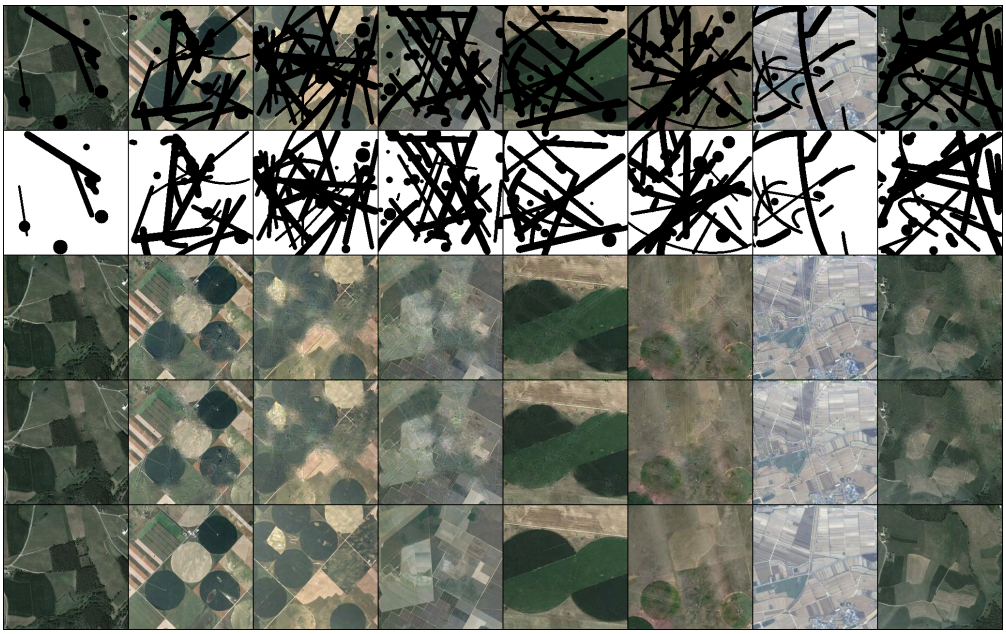


Figure 3: Predictions after 1253 steps

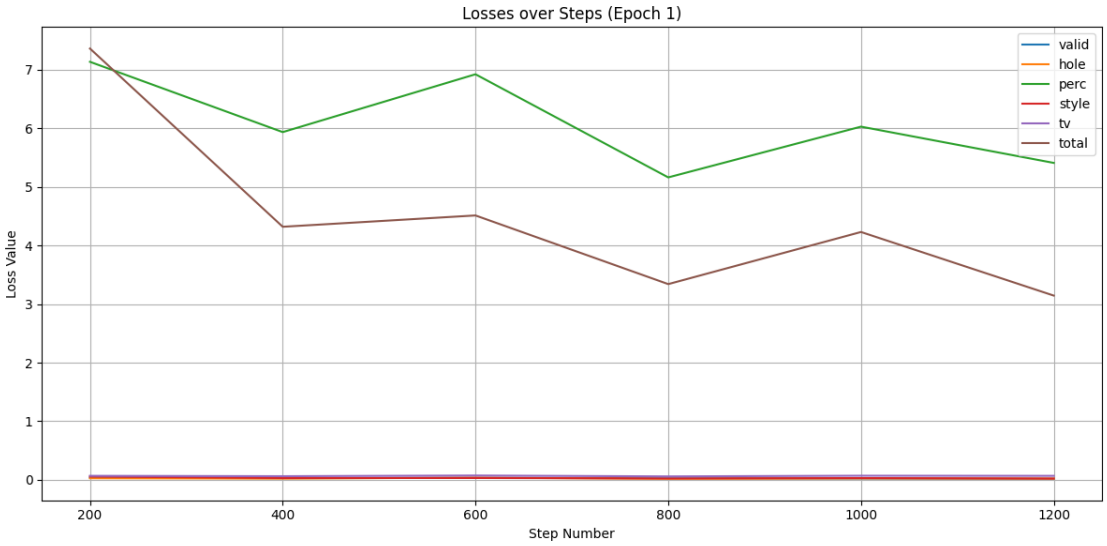


Figure 4: Loss curve during training showing convergence over time.

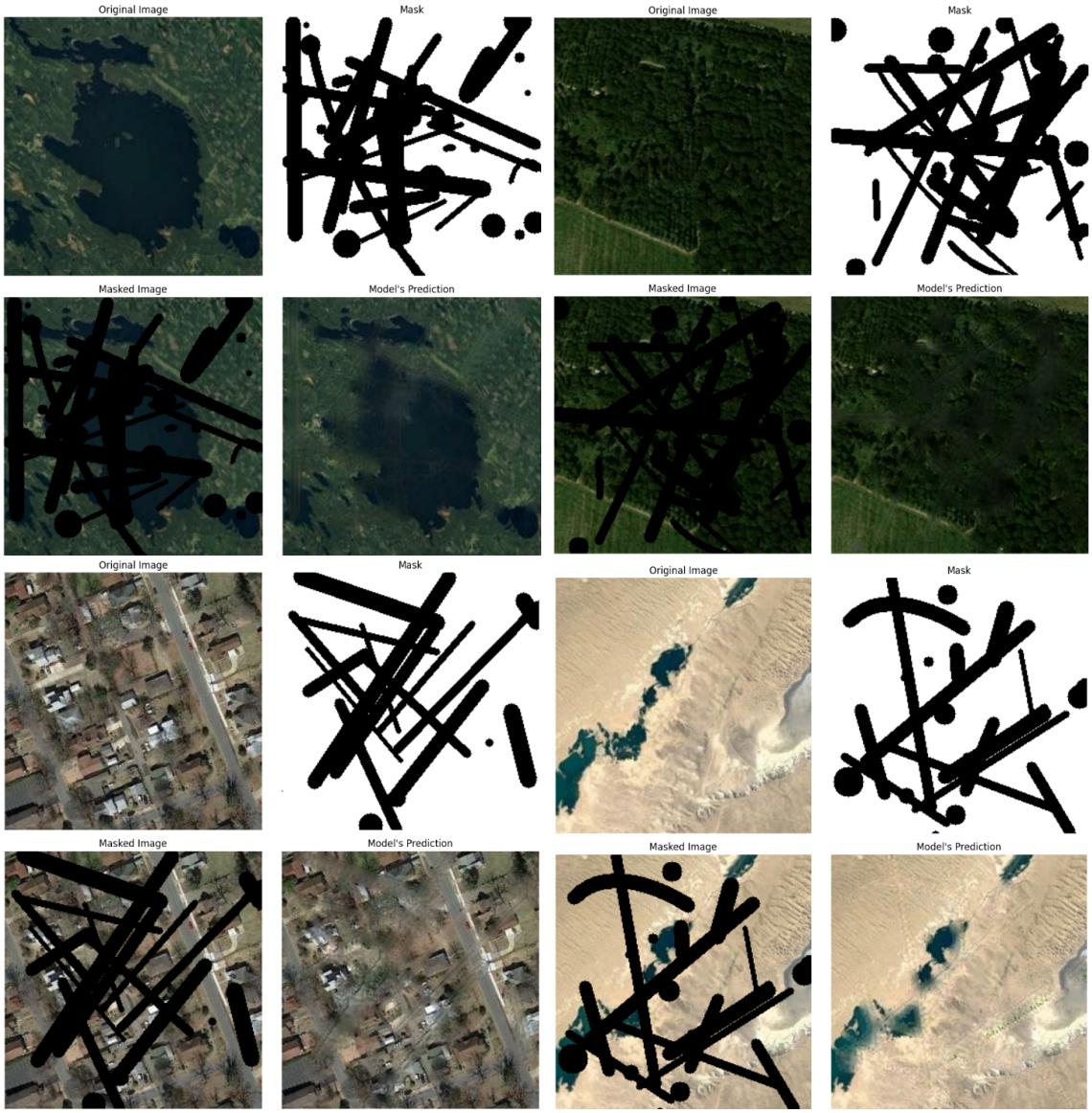
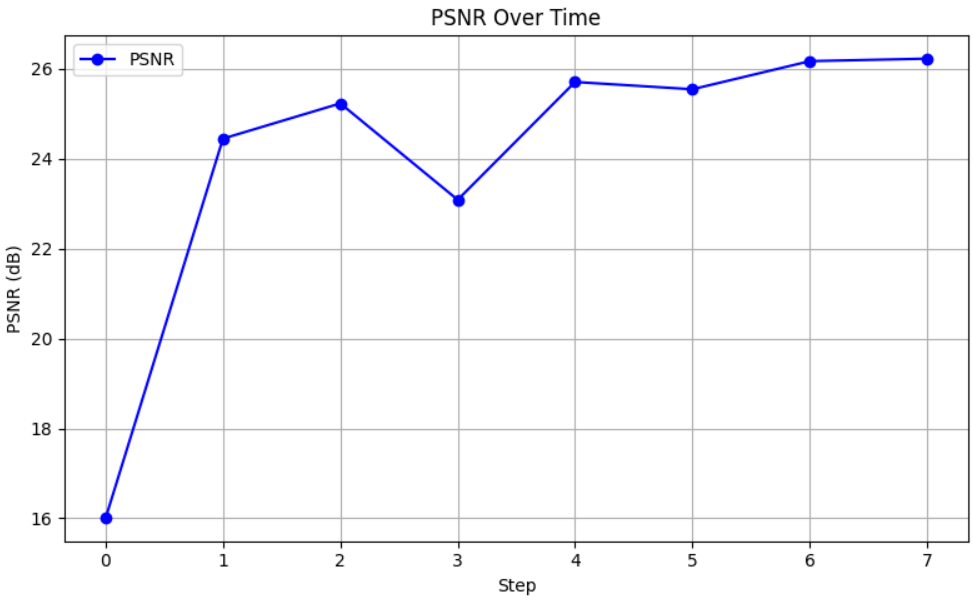
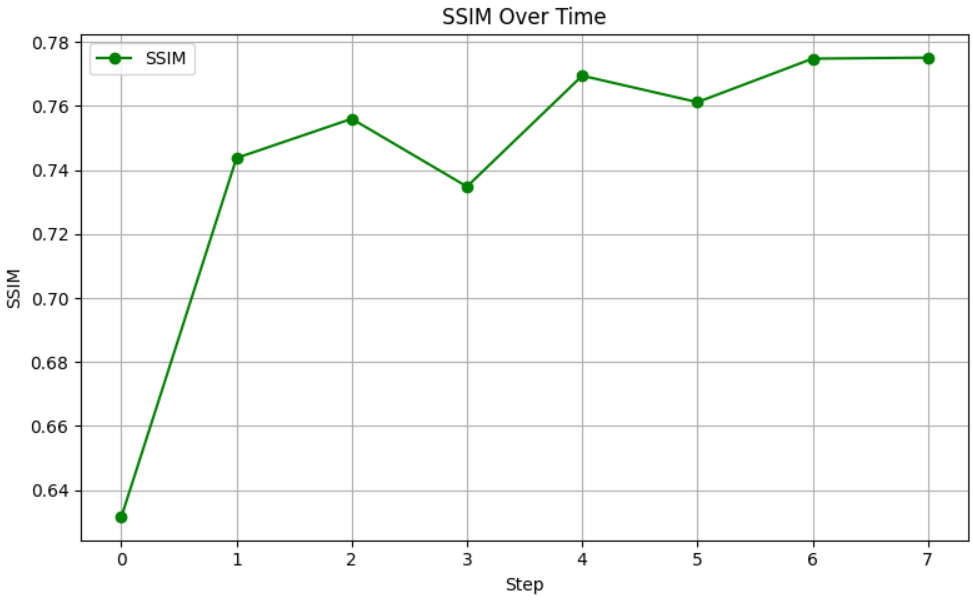


Figure 5: Final inpainting predictions after model convergence.

Figure 6: Final inpainting predictions on Out of Distribution images after model convergence.






```
⌛ Evaluating the Model

🔵 Running PSNR and SSIM evaluation on entire validation set...
📈 Mean PSNR: 26.23 ± 3.96 dB
📈 Mean SSIM: 0.7752 ± 0.0854

✅ Evaluation Successful!
```

Figure 7: Evaluation Metrics over training steps

7. References

1 tanimutomo/partialconv - GitHub Repository

2 Kaggle - Aerial Landscape Dataset

3 Diffusion Models - Lilian Weng

4 Image Inpainting for Irregular Holes Using Partial Convolutions - arXiv:1804.07723

5 Shivkumar25/Image-Inpainting - GitHub Repository

6 db2003/image-inpainting - GitHub Repository

7 Generative Inpainting with Contextual Attention - arXiv:1803.07422

8 Pushing the Limits of Deep Image Inpainting - Towards Data Science