
# Indian Institute of Technology, Bombay

Report

---

# CS 736
# Medical Image Computing

---

Recurrent Feature Reasoning for Medical Image Inpainting

Ashwin Goyal (210050024)
Ravi Kumar (210010052)

*Course Instructor:* Prof. Suyash Awate

# Contents

# 1 Introduction

## 1.1 Inpainting

Image inpainting refers to the technique of reconstructing missing, corrupted, or unwanted parts of an image. This process allows for the restoration of damaged photographs, the removal of unwanted objects from a scene, and the completion of partially captured images. Inpainting finds applications in various domains, including photo restoration, image editing, and even computer graphics.

### 1.1.1 The Inpainting Process

- Defining the Missing Area: The user employs tools like brushes or selection tools to define the region of the image that requires inpainting.
- Surrounding Analysis: The inpainting algorithm analyzes the surrounding areas of the defined region. This analysis focuses on extracting information about textures, patterns, and color variations to understand the overall visual context.
- Content Generation: Based on the analysis of the surrounding regions, the inpainting algorithm generates new pixels to fill in the missing area. The objective is to create content that seamlessly blends with the surroundings, maintaining the image's natural coherence.

### 1.1.2 Popular Inpainting Techniques

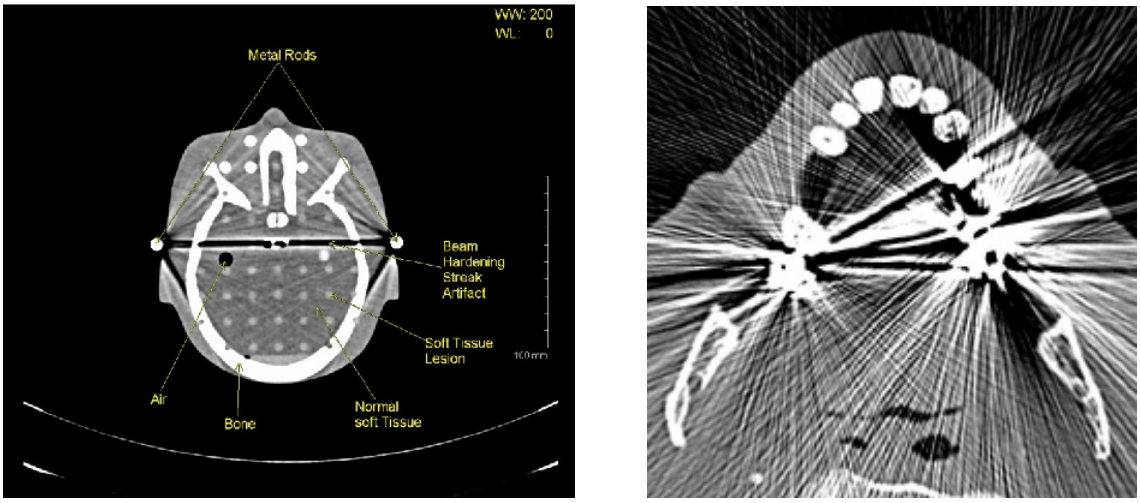
- Diffusion-based Methods: These techniques progressively propagate information from the known regions of the image towards the missing area, gradually filling it in.
- Patch-based Methods: These methods borrow textures from similar regions within the image and strategically place them to reconstruct the missing content.
- Deep Learning-based Methods: With the rise of artificial intelligence, deep learning algorithms have emerged as powerful inpainting tools. Trained on massive image datasets, these methods can learn complex image patterns and generate highly realistic inpainted content.

Existing inpainting techniques often struggle to handle large holes effectively, as they lack sufficient constraints to guide the reconstruction process in the hole's central regions.

## 1.2 Need for Medical Image Inpainting

Medical imaging plays a vital role in modern diagnosis, with technologies like CT, MRI, and PET providing crucial information to doctors. However, during image acquisition and transmission, various factors can lead to incomplete or missing data.

- Motion artifacts in MRI can cause image distortions.
- Beam hardening in CT scans creates artifacts like streaking and cupping.
- Metallic implants can introduce light and dark streaks in CT images.




(a) Beam Hardening

(b) Streaks due to dental fillings

Figure 1: Artifacts in medical images

These artifacts significantly reduce image quality and can hinder accurate diagnosis by obscuring important details (7). This is particularly critical for tasks like image segmentation, classification, and attenuation correction in PET/MRI, where complete and accurate information is essential.

This is also used when we need to analyze multi-modal data. Often, images across different datasets won't align perfectly, and hence inpainting the regions which are not present in some of them can be helpful in making inferences.

# 2 Method

## 2.1 Recurrent Feature Reasoning

This project explores a novel approach called Recurrent Feature Reasoning (RFR) proposed by (5) to address the limitations of existing inpainting techniques. The RFR module progressively reasons about the extracted feature maps of the image. This reasoning process helps to infer the boundaries of the missing region and utilize this information to accurately reconstruct the missing content.

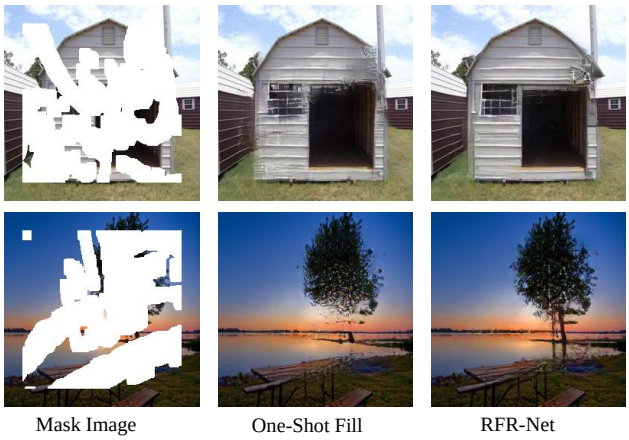



Figure 2: Illustration of semantic ambiguity that existing one-shot inpainting methods suffer from. Due to the lack of explicit constraints on the hole center, it is challenging for a network to directly inpaint on the hole center. The results of One-Shot Fill are taken from the state-of-the-art method (4).

### 2.1.1 Key Components

- Recurrent Feature Reasoning Module (RFR): This module iteratively refines the feature representation of the image, progressively incorporating knowledge about the hole boundaries. The RFR module employs a recurrent architecture that allows it to reason about the context of the hole and gradually refine its understanding as the reconstruction proceeds.
- Knowledge Consistent Attention (KCA) Module: The KCA module complements the RFR module by capturing long-range dependencies within the feature maps. It focuses on capturing information from distant locations in the feature map, essential for reconstructing intricate details within large holes.

The model has multiple modules, corresponding to Area Identification, Feature Reasoning and Feature Merging, where KCA us used within the Feature Reasoning Module. The images help understand the structure of the model, and further details can be found in the paper (5).

We used their official GitHub implementation as well for the python implementation of this model (jingyuanli001).

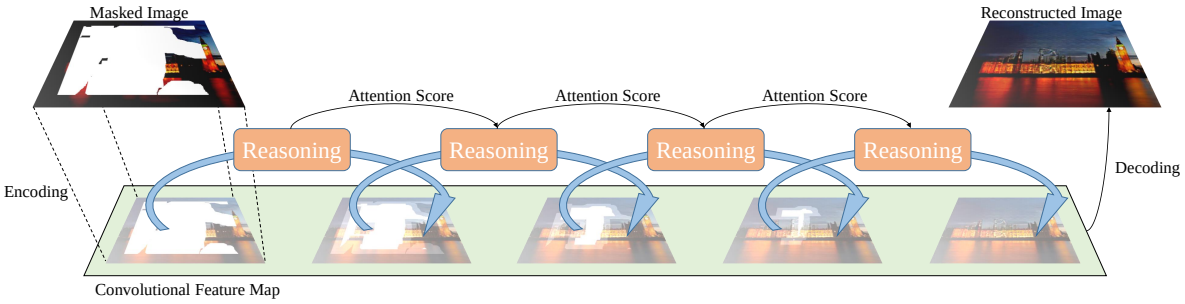



Figure 3: Overview of the RFR inpainting scheme. The masked image is first mapped into the convolutional feature space and processed by a shared Feature Reasoning module recurrently. After the feature map is fully recovered, the generated feature maps are merged together and the merged feature is translated back to an RGB image (5)

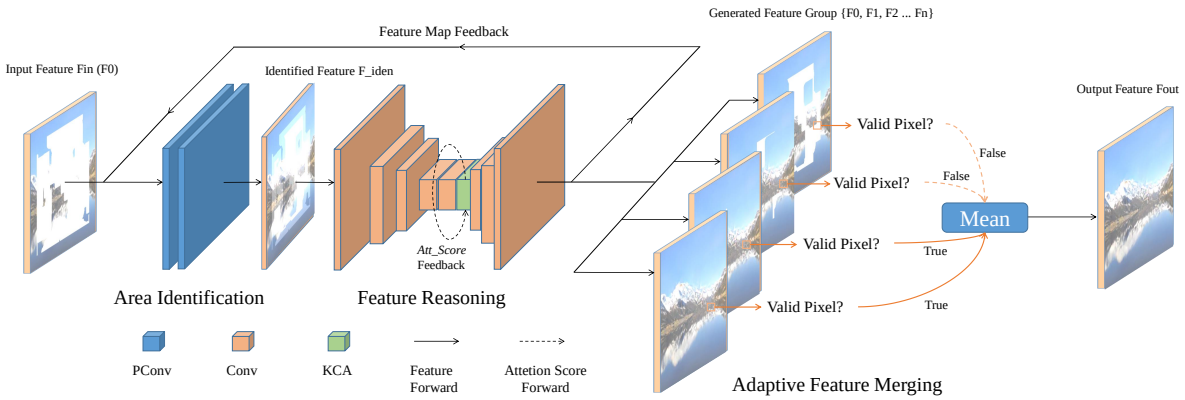
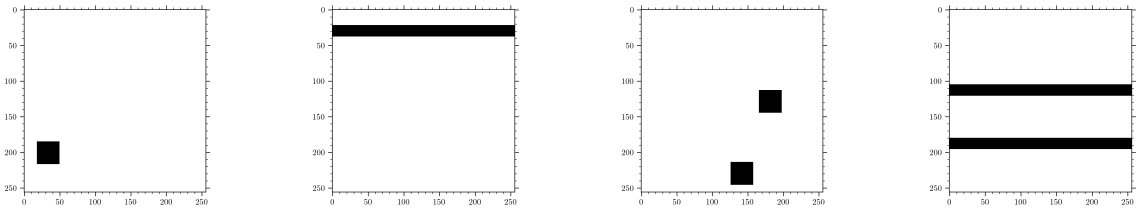



Figure 4: Illustration of the Recurrent Feature Reasoning module. The area identification process and the feature reasoning process are performed continuously. After several times of reasoning, the feature maps are merged in an adaptive fashion and an output feature map of a fixed channel numbers is generated. The module is Plug-In-and-Play and can be placed in any layer of an existing network (5)

## 2.2 Mask Generation

We generated two types of masks, a square mask of defined size to emulate regional artifacts of various sizes, and a strip mask spanning the length/height of the image of a defined thickness to simulate beaming artifacts.

NumPy arrays of size 256x256 were initialized with ones, and the mask locations were chosen randomly to not introduce any bias, then the mask pixels were set to a value of 0.



(a) 1 Hole (b) 1 Strip (c) 2 Holes (d) 2 Strips

Figure 5: Various kinds of masks

## 2.3 Datasets

We used two public datasets that are not specific to inpainting. The dataset for the masks was self generated as described in Section 2.2

- **Chest-xray-pneumonia**: This dataset contains ∼6000 chest X-rays of patients suffering from pneumonia, along with non-pneumonia X-rays. The original dataset can be found at (3) and on Kaggle. Training for the model was done with the same hyperparameters as in the official git repo, learning rate was 1e-4 and fine-tuning learning rate was 5e-5. Training was always done for about 12000 epochs + 8000 epochs fine-tuning. Train set consisted of the first 800 images in this dataset, with the next 200 being used for testing.

- **SARS-COV-2 Ct-Scan Dataset**: This dataset contains ∼2500 CT scans of COVID-19 infected patients, along with non-COVID scans. The original dataset can be found at (1) and on Kaggle. Models using this dataset were trained for 50,000 iterations and were frequently saved - every 1,000 till 20,000 iterations followed by every 10,000 iterations. Hyperparameter fine-tuning was done using a lower (by 10x) learning rate for 50% of the train iterations. Learning rates used were the same as above. The train set consisted of the first 1000 images in this dataset, with the next 400 being used for testing

***Note:*** *Images from both datasets were cutting them to the largest possible square resized to 256x256 pixels.*

## 2.4 Comparison Metrics

We used three different metrics, namely SSIM, PSNR and RMSE for evaluation of the quality of the training and model fine-tuning.

### 2.4.1 Structural Similarity Index Measure

Structural Similarity Index (SSIM, (8)) as a measure of image quality that compares local patterns of pixel intensities that have been normalized for luminance and contrast. It is based on the assumption that the human visual system is highly adapted for extracting structural information from a scene. SSIM compares the luminance, contrast and structure of the reference image with the distorted image.

The measure between two windows $x$ and $y$ of common size $N \times N$ is:

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{1}$$

where $\mu_x$ and $\mu_y$ are the average values, $\sigma_x$ and $\sigma_y$ are the variances, $\sigma_{xy}$ is the covariance of windows $x$ and $y$, $C_1$ and $C_2$ are constants to stabilize the division with weak denominator. The SSIM index ranges from -1 to 1, where 1 indicates the two images are identical.

### 2.4.2 Peak Signal-to-Noise Ratio

We often use this ratio as a measurement of quality between the original image and the resultant image. The higher the value of PSNR, the better will be the quality of the output image.

$$\mathrm{PSNR} = 10 \log_{10} \left( \frac{\mathrm{MAX}^2}{\mathrm{MSE}} \right) \tag{2}$$

where MAX is the maximum possible pixel value of the image, and MSE is the Mean Squared Error between the images.

### 2.4.3 Root Mean Squared Error

RMSE is a common statistic used to measure the error between two quantities.

$$\mathrm{RMSE} = \frac{1}{N} \sqrt{\sum_{i=1}^{N} \sum_{j=1}^{N} (I_{\mathrm{gt}}^{ij} - I_{\mathrm{rec}}^{ij})^2} \tag{3}$$

***Note:*** *We will only be calculating these metrics in the masked/reconstructed region of the images, as the images outside this region are identical.*

# 3 Experiments and Results

We perform a series of experiments to determine the capabilities of RFR network to deal with different parameters of masks and images.

## 3.1 Dataset Variability

This analysis was performed on the first dataset, Chest-xray-pneumonia. We wished to understand how this inpainting model performs when there are medical differences in images, such as x-rays with and without pneumonia.

As a baseline, we train a model on only non-pneumonia (normal) images and test it on normal images.

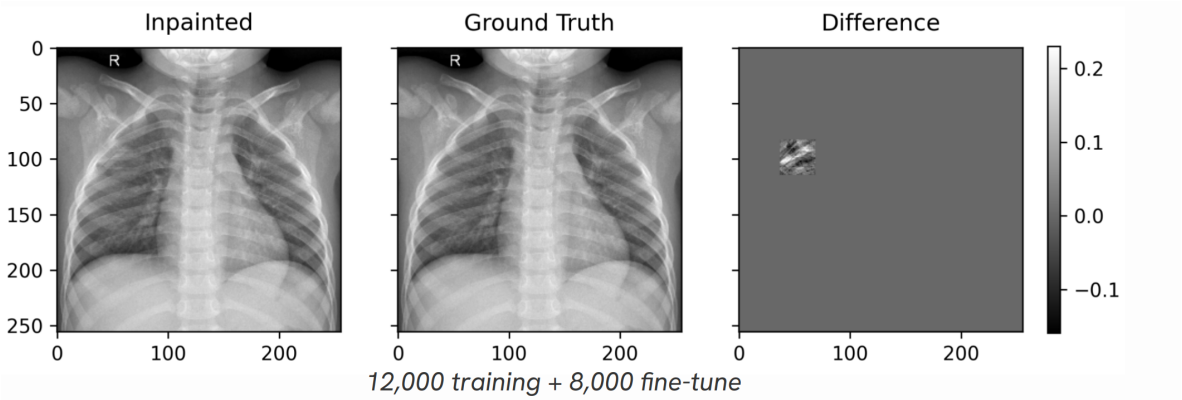



Figure 6: Normal to normal X-ray image

It is standard to compute metrics over the whole image, however since we only use small images we compute metrics over the masked region only. This makes it easier to compare masks of different sizes and shapes.

| | ssim | psnr | rmse |
|---|---|---|---|
| Over whole image | 0.992 | 41.23 | 0.008 |
| Over just mask | 0.684 | 22.34 | 0.055 |

We then train a model on normal+pneumonia images and test it on normal+pneumonia images.

| | ssim | psnr | rmse |
|---|---|---|---|
| normal+pneumonia | 0.634 | 21.67 | 0.065 |
| normal | 0.684 | 22.34 | 0.055 |

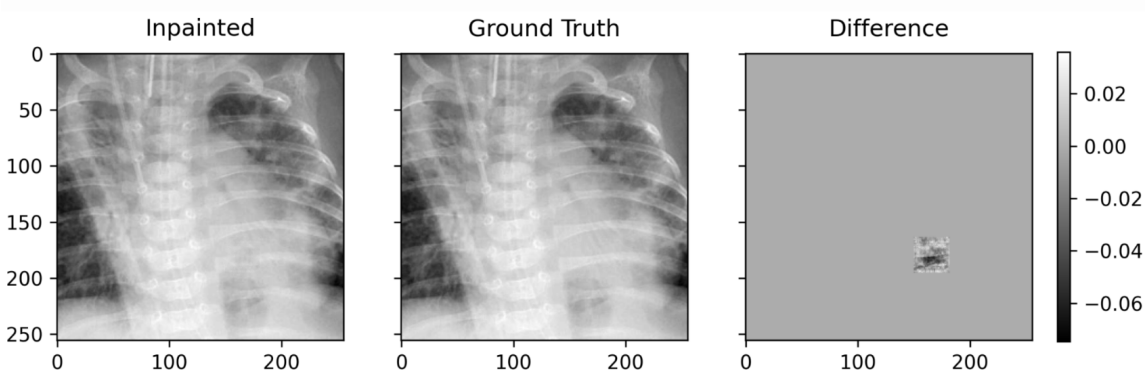



Figure 7: Normal+pneumonia to normal+pneumonia X-ray image

Lastly, we train a model on normal images and test it on non-pneumonia images. This is the OOD testing and is very important for medical image inpainting.

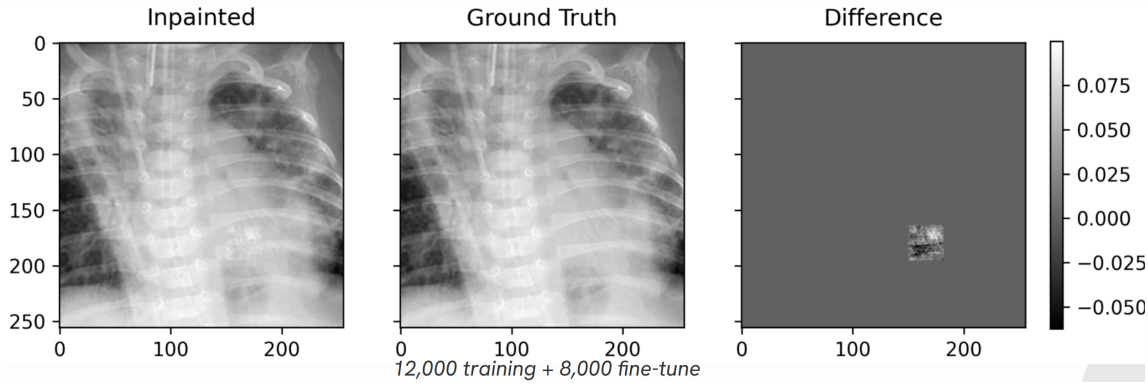



Figure 8: Normal to pneumonia X-ray images

| | ssim | psnr | rmse |
|---|---|---|---|
| normal+pneumonia | 0.684 | 22.34 | 0.055 |
| normal to pneumonia | 0.639 | 22.10 | 0.057 |

## 3.2 Hole Size

We varied hole size while keeping all other parameters constant during training. We observed the following variation in metrics wrt hole size.

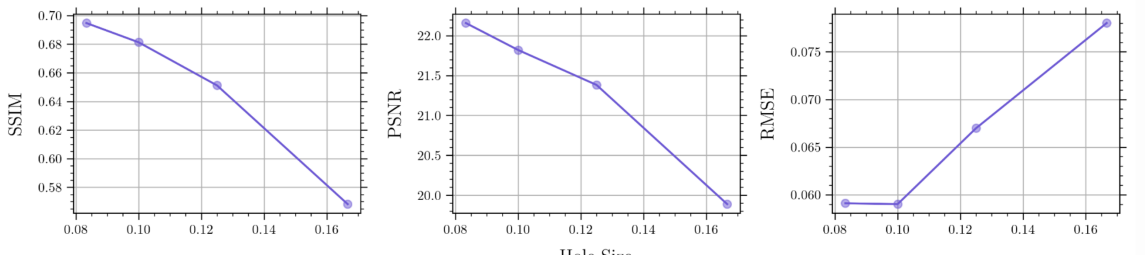



Figure 9: Variation of metrics with hole size in Chest X-ray 'normal' dataset

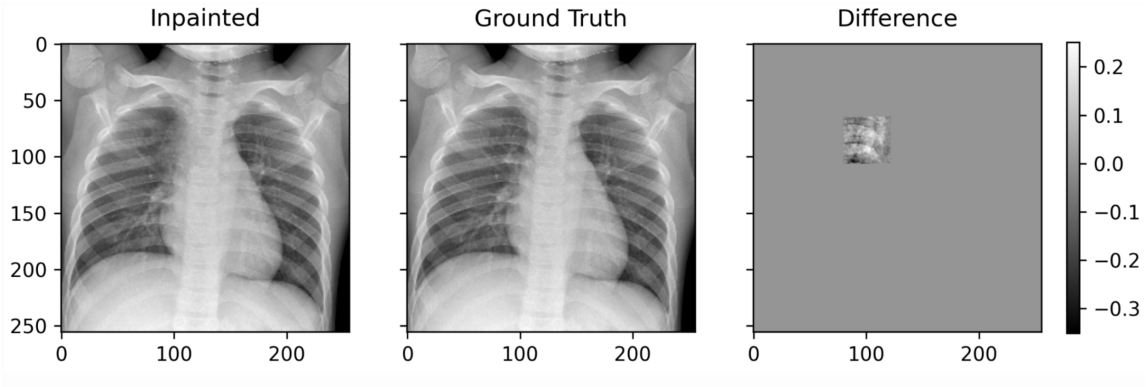



Figure 10: An OOD result, trained on hole size = 1/12, test on hole size = 1/6

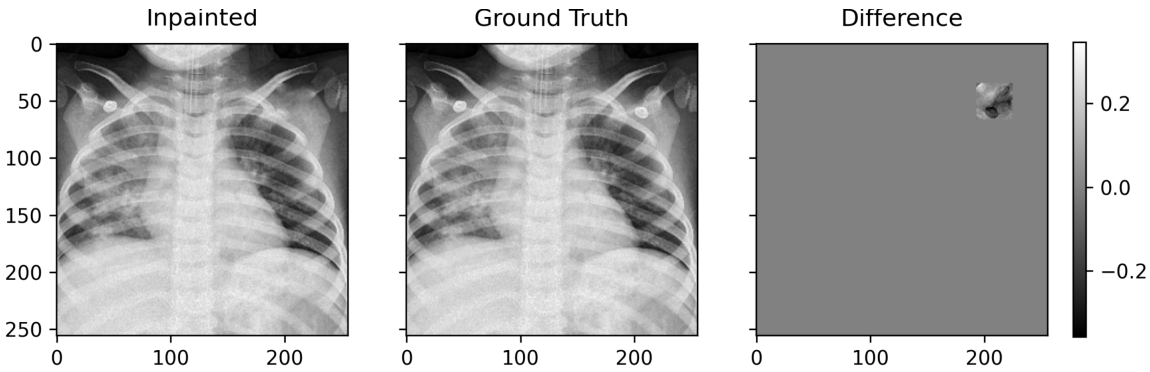



Figure 11: An example where a certain rare feature is not fitted well

***Note:*** *The below results were obtained by training the model on the "SARS-COV-2 Ct-Scan" dataset.*

## 3.3 Types of Masks

We used 2 different kinds of masks for training and compare the results below.

### 3.3.1 Square Holes

Square holes were used to simulate regional artifacts possibly present in the image.

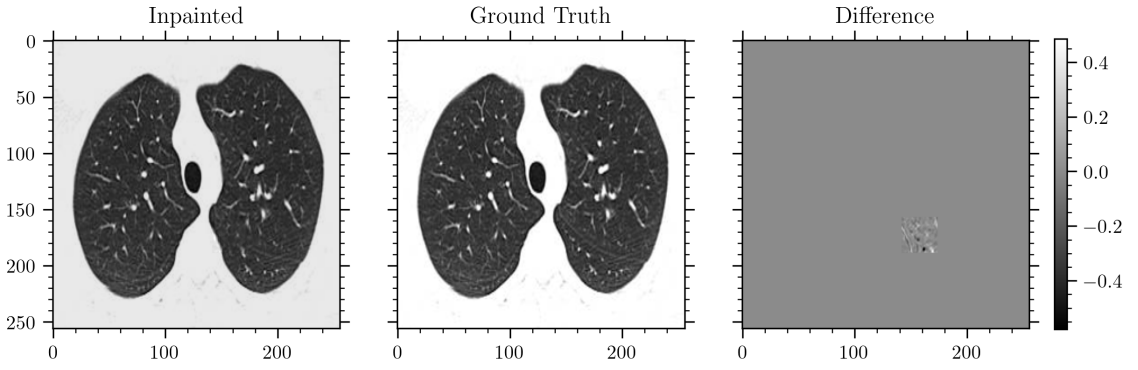



Figure 12: Inpainting result for square hole masks (16,000 training + 8,000 fine-tune)

### 3.3.2 Strips

Image wide (256px) and 16px thick strips were generated to simulate beaming artifacts.

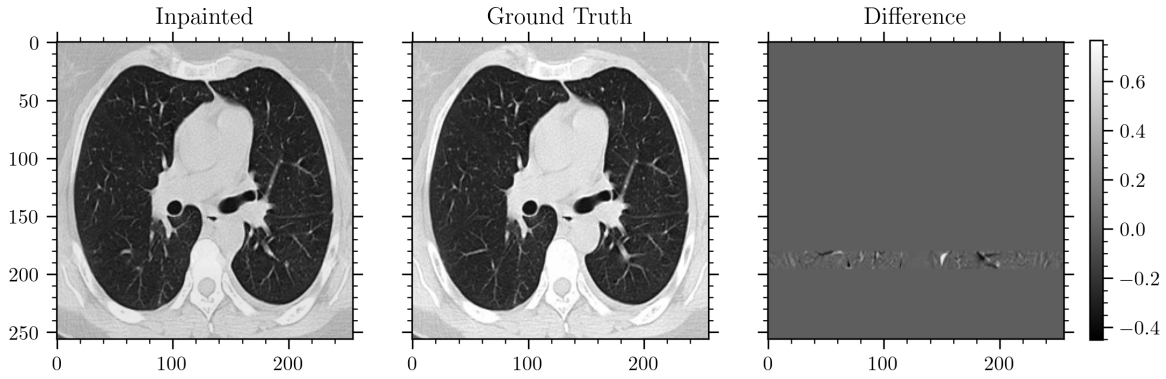



Figure 13: Inpainting result for strip masks (15,000 training + 7,500 fine-tune)

### 3.3.3 Comparison

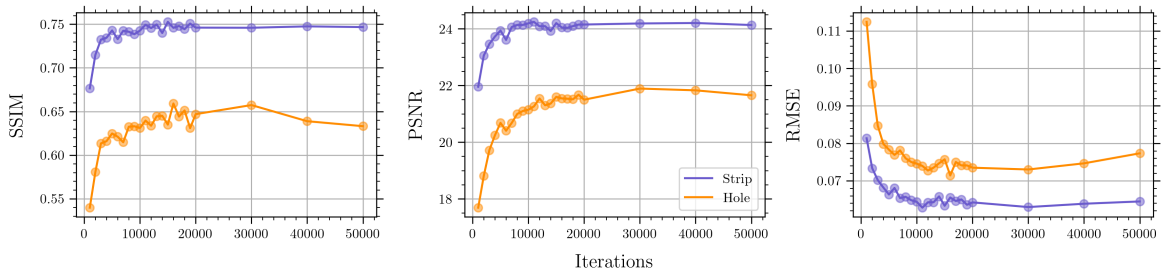
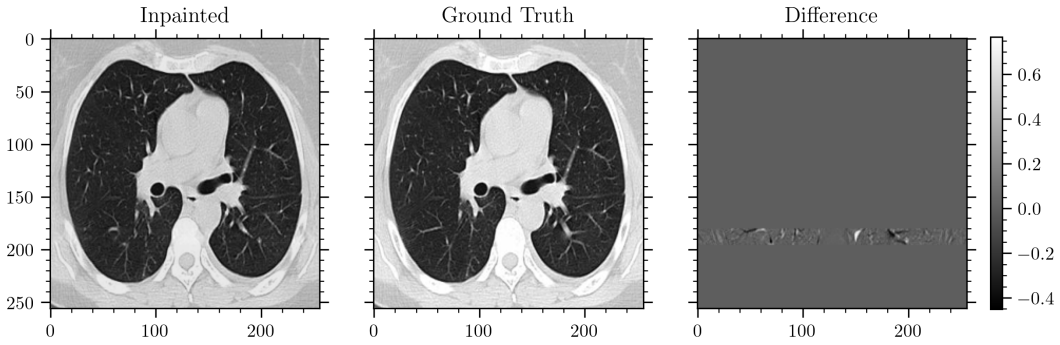



Figure 14: Metrics vs Iterations for models trained using hole and strip masks

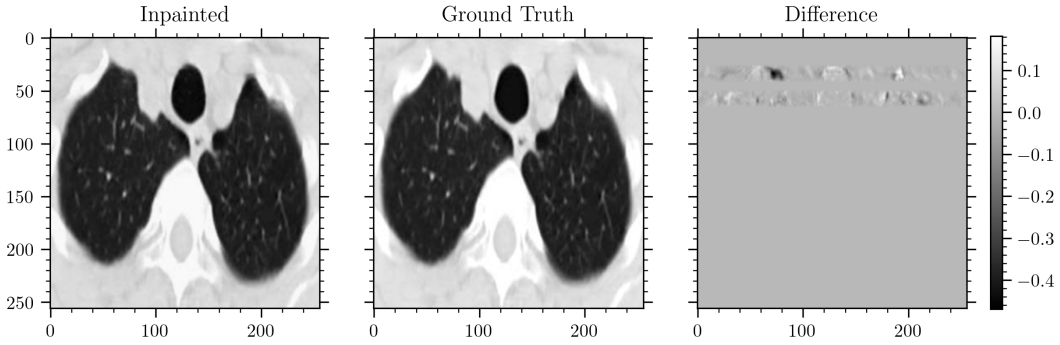
## 3.4 Number of Holes/Strips

The models were trained on datasets consisting of the same images, but masks containing 1, 2, 4, 8 holes and 1, 2, 4 strips.

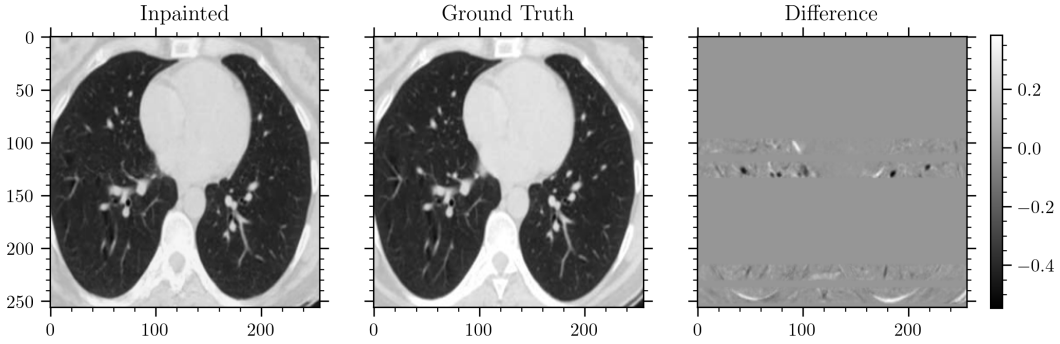
### 3.4.1 Strips



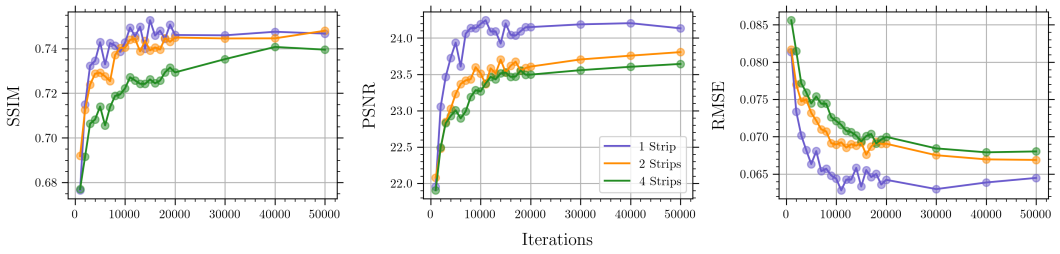

(a) 1 Strip: 15,000 training + 7,500 fine-tune




(b) 2 Strips: 18,000 training + 9,000 fine-tune



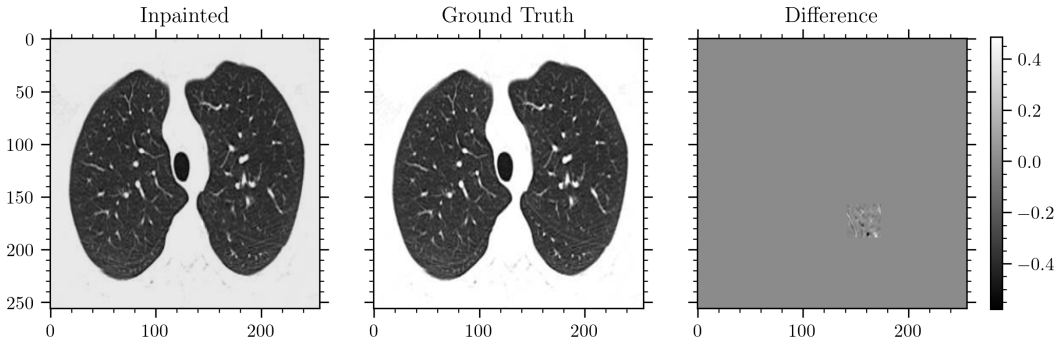

(c) 4 Strips: 40,000 training + 20,000 fine-tune



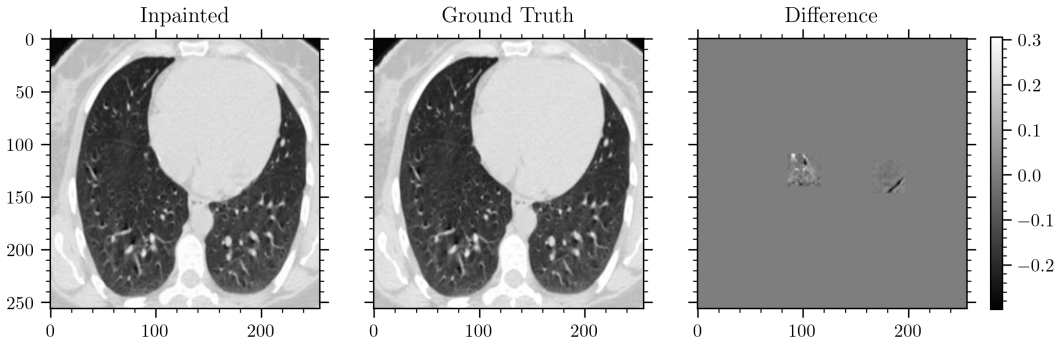

(d) Metrics vs Iterations

Figure 15: Comparison of inpainting on 1, 2, and 4 strips per mask

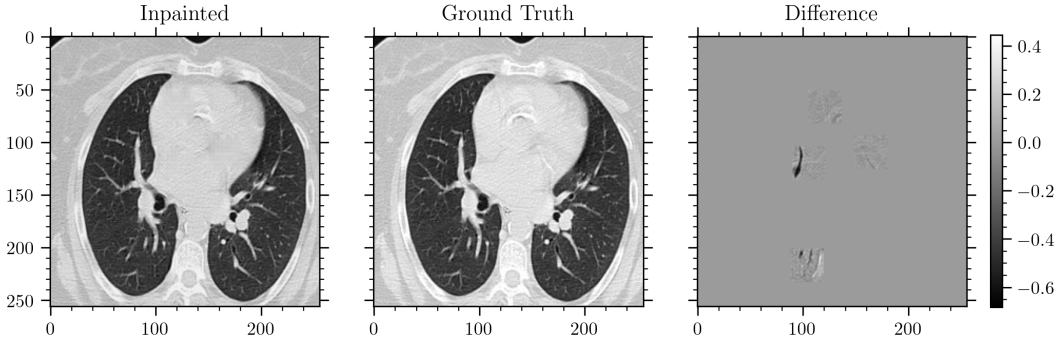
### 3.4.2 Holes



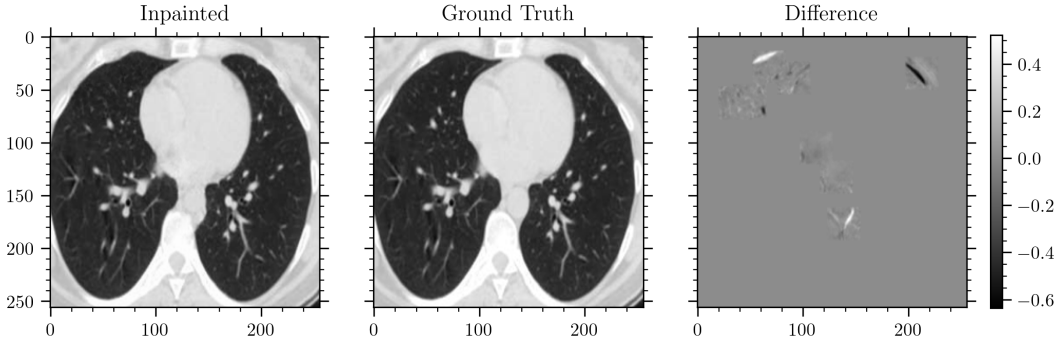

(a) 1 Hole: 16,000 training + 8,000 fine-tune




(b) 2 Holes: 13,000 training + 6,500 fine-tune




(c) 4 Holes: 20,000 training + 10,000 fine-tune




(d) 8 Holes: 50,000 training + 25,000 fine-tune

Figure 16: Comparison of inpainting on 1, 2, 4, and 8 holes per mask

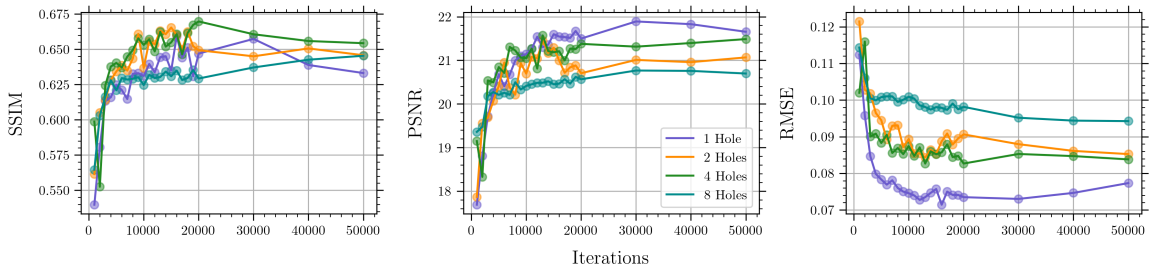



Figure 17: Metrics vs Iterations

## 3.5 Orientation of Masks

The orientation changes only apply to strip masks. We took a combination of horizontal and vertical strips to test how well the model performs with this variability.

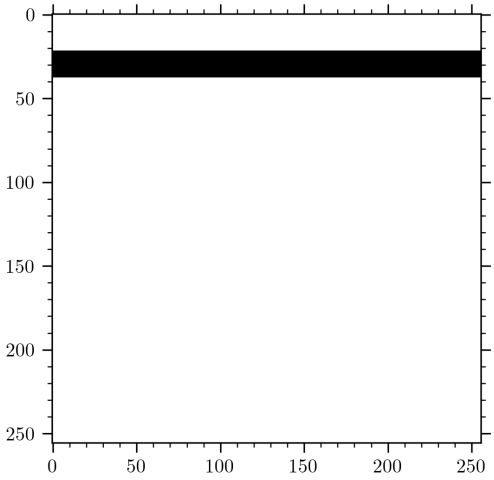


Figure 18: 1 horizontal strip

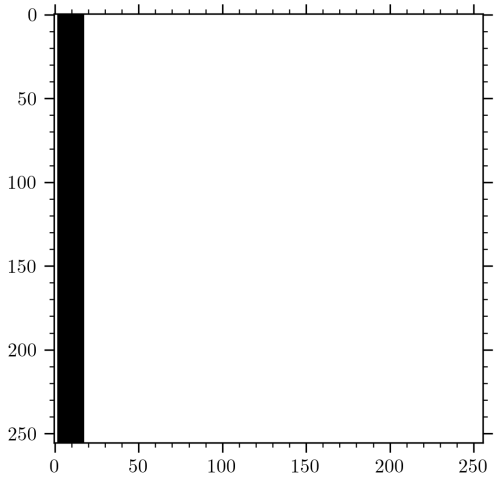


Figure 19: 1 vertical strip

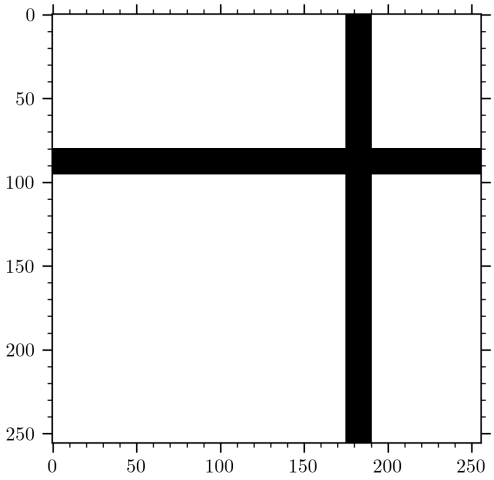


Figure 20: 1 hoizontal + 1 vertical strips

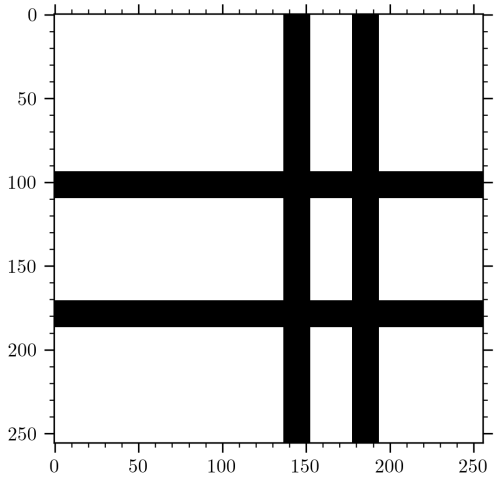
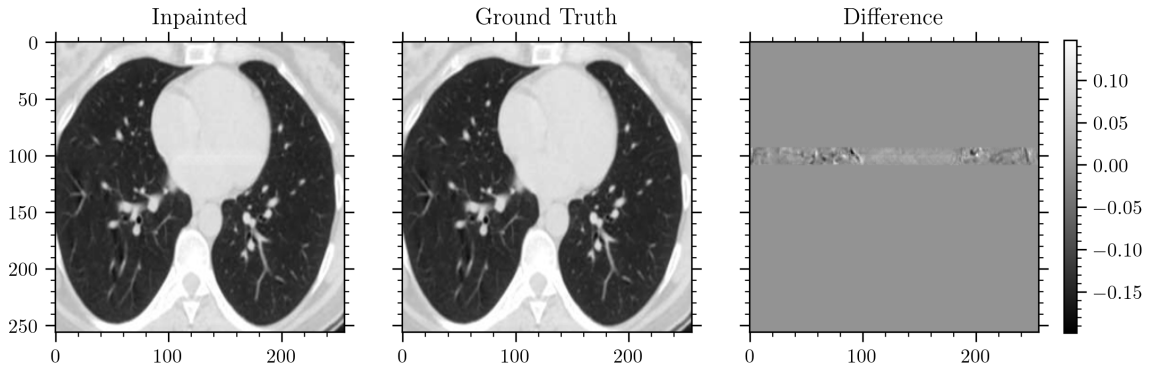
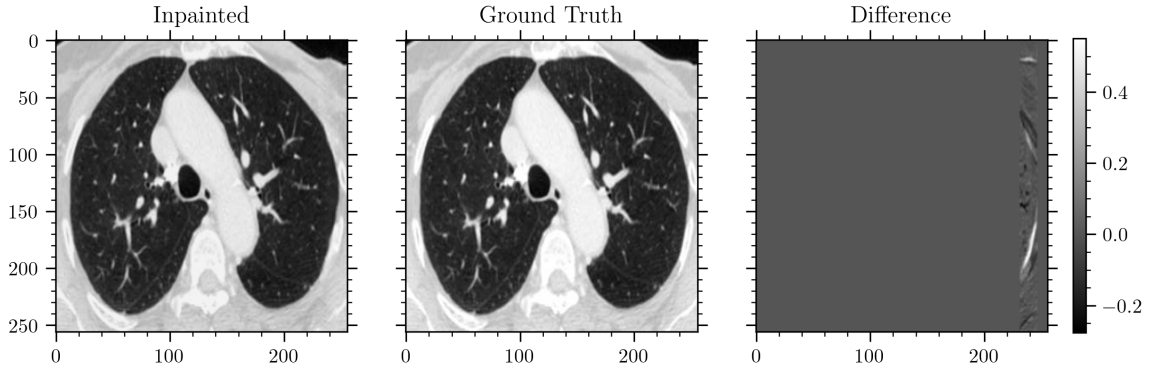


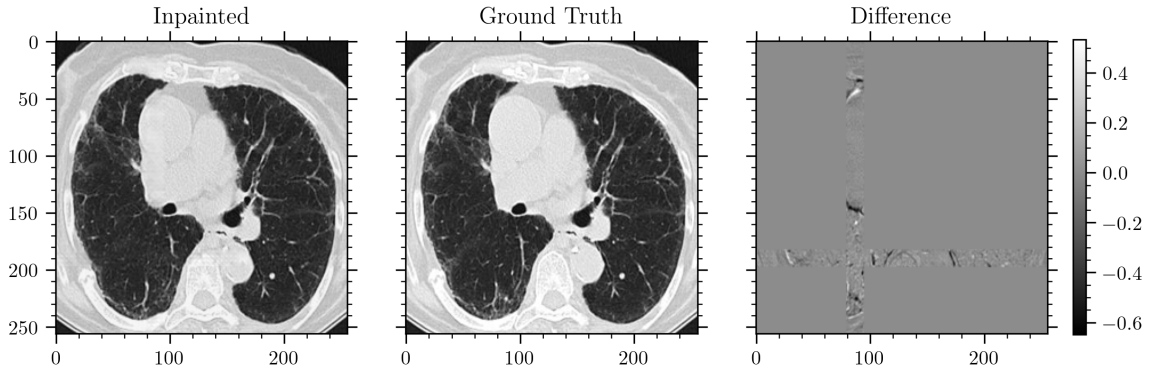
Figure 21: 2 hoizontal + 2 vertical strips



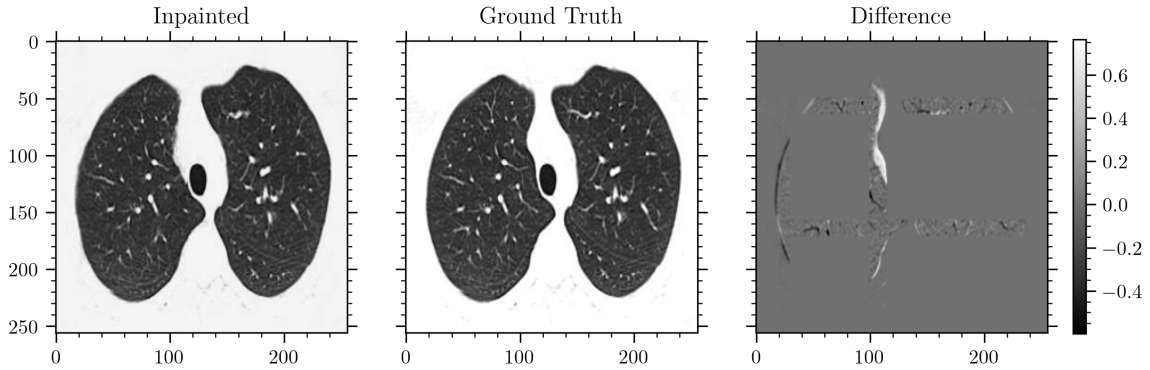

(a) 1 Strip - Horizontal+Vertical: 11,000 training + 5,500 fine-tune




(b) 1 Strip - Horizontal+Vertical: 11,000 training + 5,500 fine-tune

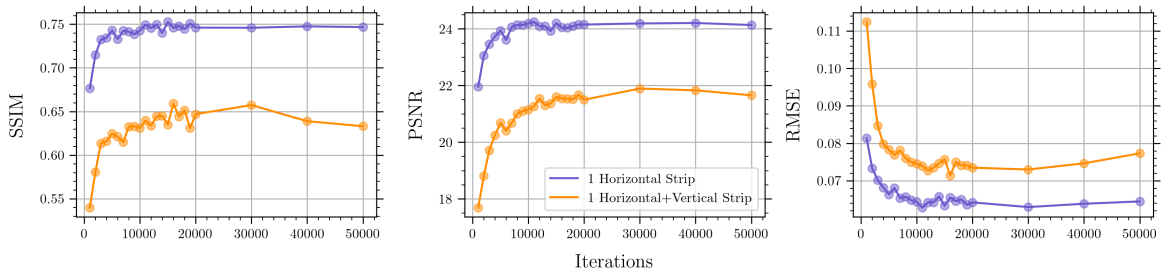



(c) 2 Strips - Horizontal+Vertical: 40,000 training + 20,000 fine-tune

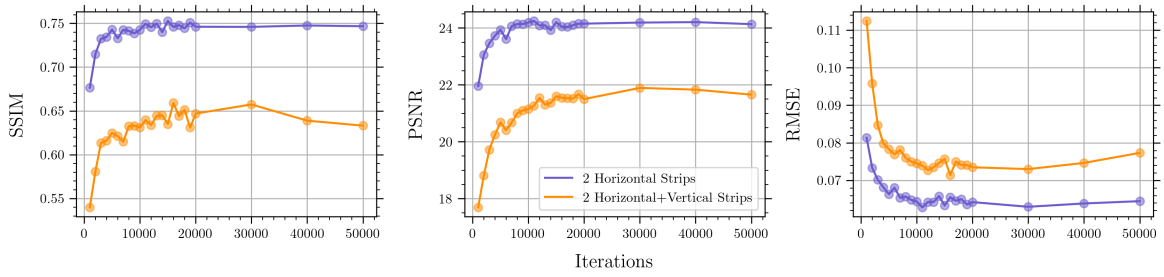



(d) 4 Strips - Horizontal+Vertical: 40,000 training + 20,000 fine-tune

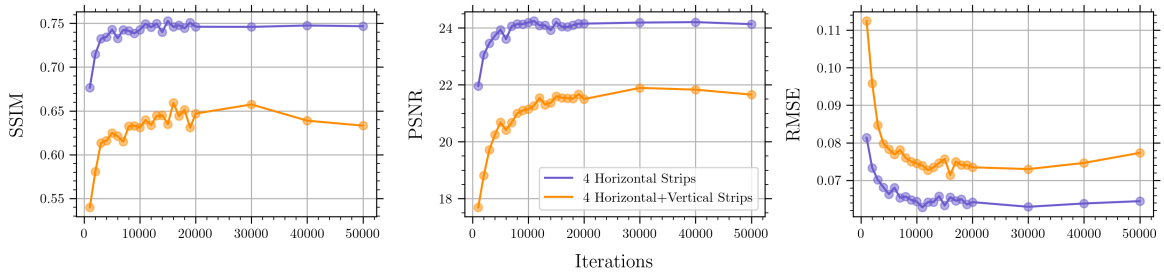
Figure 22: Comparison of inpainting on 1, 2, 4 Horizontal+Vertical Masks




(a) Comparision between 1 Strip H+V and only H




(b) Comparision between 2 Strips H+V and only H




(c) Comparision between 4 Strips H+V and only H

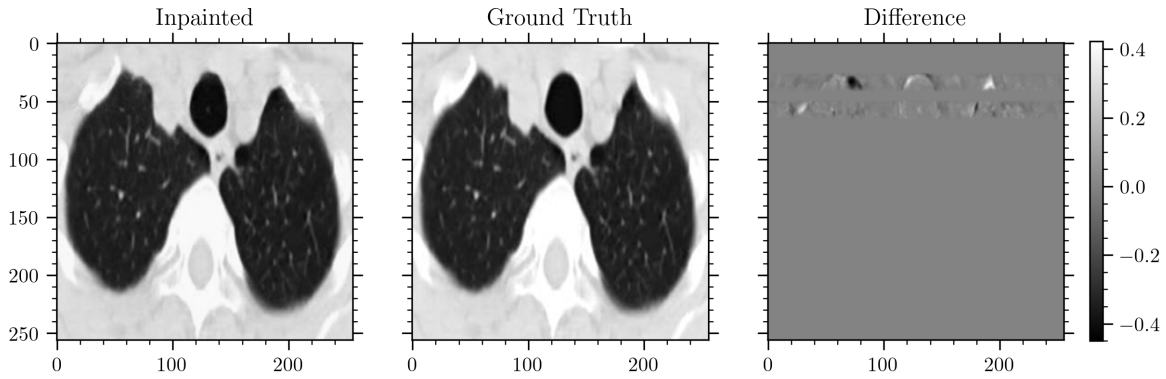
Figure 23: Metrics vs Iterations on 1, 2, 4 Horizontal+Vertical Masks

## 3.6 Out of Distribution Tests

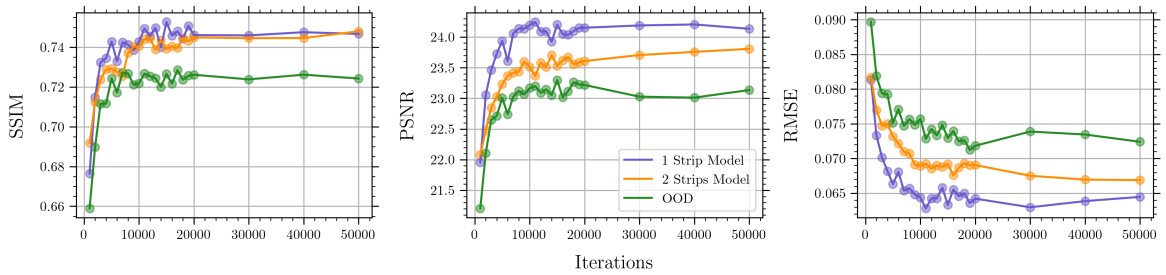
It is necessary to test inpainting models on distributions that they weren't trained on to check adaptability and learning of general features.

### 3.6.1 1 Strip Model + 2 Strips Mask

The models trained on masks containing only 1 strip were tested with masks containing 2 strips.




(a) Inpainting result of 1 Strip Model + 2 Strips Mask




(b) Metrics vs Iterations

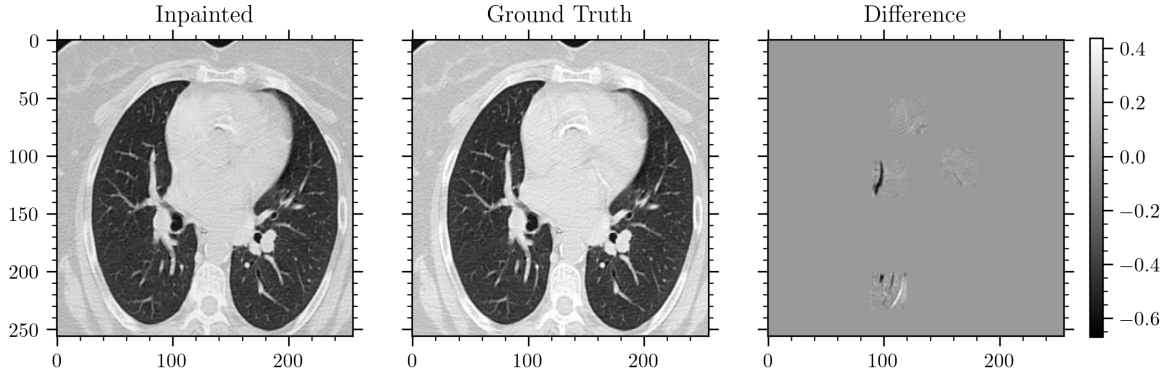
Figure 24: OOD results for 1 Strip Model + 2 Strips Mask

The Metrics vs Iterations plots compares model (1 Strip Model) trained and tested on 1 Strip Masks, trained and tested on 2 Strips Mask (2 Strips Model), and the OOD result of training on 1 Strip Masks and testing on 2 Strips Masks.

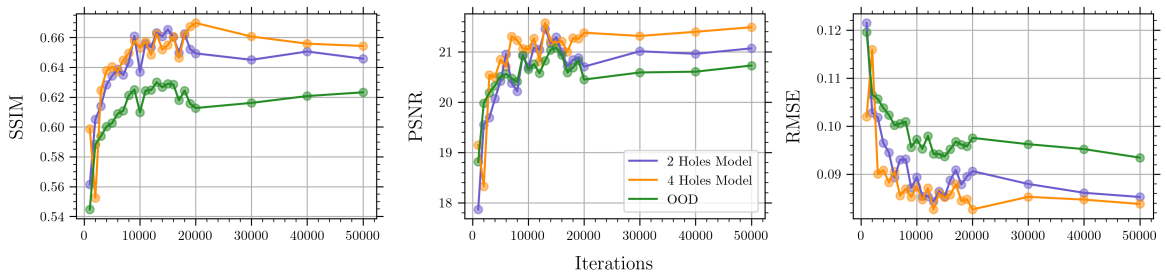
We can see clearly that the OOD result performs poorer than models trained and tested on their respective masks (ID).

### 3.6.2 2 Holes Model + 4 Holes Mask

The models trained on masks containing 2 holes were tested with masks containing 4 holes.




(a) Inpainting result of 2 Holes Model + 4 Holes Mask




(b) Metrics vs Iterations

Figure 25: OOD results for 2 Holes Model + 4 Holes Mask

The Metrics vs Iterations plots compares model (2 Holes Model) trained and tested on 2 Holes Masks, trained and tested on 4 Holes Mask (4 Holes Model), and the OOD result of training on 2 Holes Masks and testing on 4 Holes Masks.

We can see clearly that the OOD result performs poorer than models trained and tested on their respective masks (ID).

# 4 Conclusion

RFR seems to fit CT and X-ray well visually, and changing hole shape or number doesn't affect the ability of a model to predict by much. Similarly, the presence of pneumonia does not hurt model accuracy by a large amount, even if the model is not trained to inpaint images with pneumonia.

However, when looking at the image differences, we can see structure. Which implies that there are changes to the model which have taken place between the ground truth and inpainted image. This implies that we cannot use the inpainted images for diagnostic purposes.

This defeats the purpose of medical image inpainting and hence is not a satisfactory result. We conclude that methods such as this, which attempt to inpaint realistic images but provide no guarantees of accuracy to the ground truth should not be used for the purpose of medical image inpainting. Models such as (7) and (6) are better suited for these purposes, as they take medical image structure of organs into account.

***Note:*** *The code can be found at the following repository:*

https://github.com/ravioli1369/CS736-Project/

# References


[1] Angelov, P. and Soares, E. (2020). Explainable-by-design approach for covid-19 classification via ct-scan.

[jingyuanli001] jingyuanli001. Rfr-inpainting. https://github.com/jingyuanli001/RFR-Inpainting/.

[3] Kermany, D. (2018). Large dataset of labeled optical coherence tomography (OCT) and chest X-Ray images.

[4] Li, J., He, F., Zhang, L., Du, B., and Tao, D. (2019). Progressive reconstruction of visual structure for image inpainting. In *Proc. ICCV*, pages 6721–6729.

[5] Li, J., Wang, N., Zhang, L., Du, B., and Tao, D. (2020). Recurrent feature reasoning for image inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[6] Tran, M.-T., Kim, S.-H., Yang, H.-J., and Lee, G.-S. (2021). Multi-task learning for medical image inpainting based on organ boundary awareness. *Applied Sciences*, 11(9).

[7] Wang, Q., Chen, Y., Zhang, N., and Gu, Y. (2021). Medical image inpainting with edge and structure priors. *Measurement*, 185:110027.

[8] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612.