3D Reconstruction of 2D Medical Images using Unsupervised GANs

Adwait, Avantika, Harshvardhan and Shivam

{adk1361, avantk, hershs, sgoyal15}@bu.edu

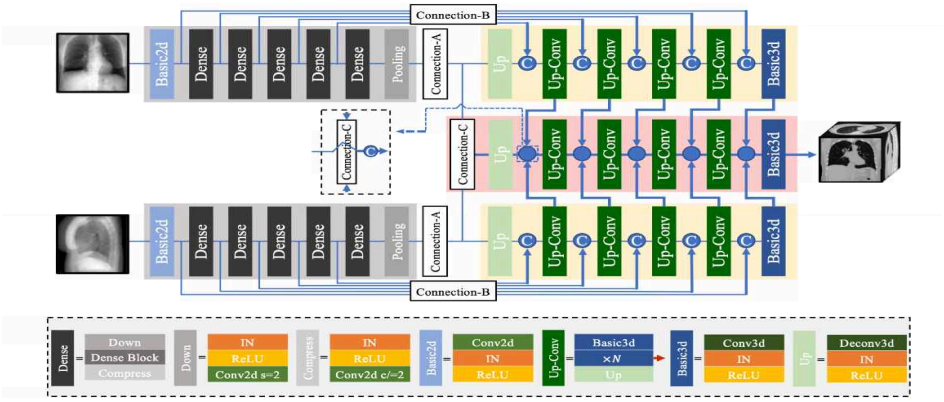

Fig 1: The generator architecture of the original X2CTGAN

1. Task

There has been a lot of recent research on Generative Adversarial Networks (GANs) and their ability to convert 2D images into 3D. Medical images would often benefit from being viewed in 3D. However, in order to efficiently use GANs to convert 2D images to 3D, we need to use supervised learning with ground truth 3D images (paired data) like Computed Tomography (CT). Such CT images are often very memory intensive and are more complex to process computationally. With our project, we have explored the possibility of using not-fully supervised learning for the same. With only 2D chest XRay input images and no ground truth or labels, we have tried answering some questions about the challenges that not-supervised learning poses in such tasks.

2. Related Work

Recent research in the area of GANs as it relates to medical images for 2D to 3D reconstruction has been spearheaded by X2CTGAN [1]. The researchers behind this framework provided an architecture for a GAN-based neural network that makes use of biplanar chest XRays to produce 3D CT-like reconstructions. They were able to produce impressive results with 26.19 dB PSNR and 0.66 SSIM. However, they made use of a supervised learning approach with CT ground truth. The dataset used by these researchers included about 70 GB of CT data. To further improve on their results, some other researchers proposed an improved framework called TRCT-GAN [2], where they made use of the same biplanar XRay images to generate 3D reconstructed images but with added Transformer blocks for enhanced resolution. Here too, the researchers made use of supervised learning with ground truth CT images. In the domain of unsupervised learning as it relates to medical images, researchers proposed another Structure Preserving Cycle-GAN framework for MRI to CT adaptation [3]. However, both CT and MRI are 3D images, and so there was no change of dimensions here, even though this involves unsupervised learning in the domain of medical images. Another approach called MADGAN, an unsupervised method to detect anomalies in brain MRI was proposed [4]. In this approach as well, while the researchers proved that unsupervised learning can be applied successfully in medical images, it was all done in the 3D domain only. Researchers of [5] proposed an unsupervised framework for converting 2D images into 3D in an unsupervised Style-GAN setup. However, their images were not medical images, which inherently come with a lot of intricacies and complexities that might be challenging in an unsupervised or semi-supervised setup.

3. Approach

3.a. Outline of the approach

Our approach was to experiment with not-fully supervised and weakly supervised learning as an alternative to supervised learning. The researchers of X2CTGAN [1], provided an in-depth framework to convert 2D biplanar images into 3D, using ground truth 3D CT images. However, the problems accompanying this approach involve extensive memory issues that come with high volume 3D CT data and the complexity of the computations behind processing these images. Our objective is to find a solution to this ground truth problem by leveraging unsupervised/weakly

supervised learning. To do this, we have carried out a series of experiments making use of only biplanar 2D images (frontal and lateral chest XRays) with the aim to produce 3D reconstructions of the same. Our experiments involved many architectures with varying input image dimensions and hyperparameters. Here we have presented the top models that performed comparatively well.

3.b. Preliminary analysis with 3DCNNs

The Simple 3D CNN project tackles the challenge of analyzing 3D data by transforming 2D image slices, such as medical scans, into 3D volumes and using a 3D Convolutional Neural Network (CNN) for classification. The process begins with preprocessing, these volumes are normalized, and data augmentation techniques, like random rotations, are applied to enhance the model's ability to generalize. The architecture of the 3D CNN consists of two 3D convolutional layers, followed by 3D max-pooling layers. The extracted features are then flattened and passed through fully connected layers, culminating in a classification layer that predicts the final output.

Metrics such as Signal-to-Noise Ratio (SNR) and Structural Similarity Index (SSIM) are employed to evaluate the model's performance, providing insights into its ability to preserve spatial relationships and detect subtle differences in data quality. This approach is particularly valuable for medical imaging tasks, where the ability to classify conditions based on volumetric features, such as CT or MRI scans, is essential. By combining 2D slices into a cohesive 3D representation, the model captures spatial context that would be lost in 2D analyses, showcasing the strength of 3D CNNs for volumetric data applications.

3.c. CycleGANs

The CycleGAN architecture is a popular type of unsupervised GAN framework for image-to-image translation. It transforms one type of image to another without the need of any paired data. It works based on 2 generators and 2 discriminators that work in a cyclic fashion. Here, if we separate the process into 2 domains, domain A includes the biplanar 2D XRay images and domain B includes the 3D volumes that we are trying to generate. Cycle A of the generator tries to generate 3D volumes from the 2D images and cycle B tries to convert the generated 3D volumes back into 2D. Similarly, the discriminators work to challenge the 2 generators where they try to establish if the generated 3D aligns with the plausible structure of 3D data, even in its absence, and also try to check if the re-projected 2D images resemble the input frontal/lateral images. Here, the CycleGAN takes the 2D biplanar images as input.

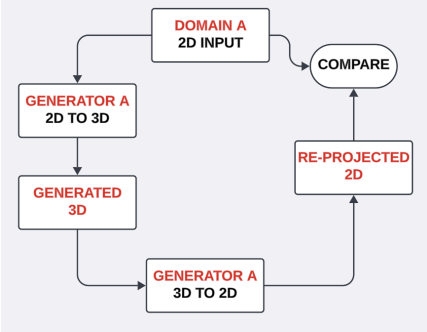

Fig 2: Basic outline of our CycleGAN Generator

Experiments were carried out twice with CycleGANs taking both biplanar images and by taking just the frontal images as input. Cycle consistency loss and adversarial loss were used during training. The generator consists of a series of convolutional layers and downsampling layers during feature extraction, 3D residual blocks and upsampling layers to transform the input into a 3D target volume. Weights were initialized using kaiming initialization. Dropout was added to prevent overfitting and ReLU was the activation layer used.

3.d. uX2CTGAN

The approach taken in this project leverages a Generative Adversarial Network (GAN) to reconstruct 3D volumes from paired 2D X-ray projections (Frontal and Lateral views). Inspired by the X2CT paper architecture[Refer to Fig 1], the Generator processes the Frontal and Lateral images independently through Dense Blocks, which extract and reuse features efficiently. These 2D features are then projected into a 3D space using a 2D-to-3D connection module, concatenated, and refined using 3D convolutional and upsampling layers to produce the final 3D volume. From this generated 3D structure, Frontal and Lateral 2D projections are recalculated, and a projection loss is computed by comparing these generated projections to the original input projections. The training was designed in a Self-Supervised manner, addressing the absence of real 3D ground truth data. Instead of directly evaluating the full 3D volume, the Discriminator takes the recalculated 2D projections (Frontal and Lateral) as input and learns to distinguish between the real projections from the dataset and the fake projections generated by the Generator. This adversarial framework, combined with the projection

loss, ensures that the Generator not only produces realistic 3D structures but also generates projections that closely match the original inputs. By optimizing both the projection loss and adversarial loss simultaneously, the Generator is encouraged to refine the global 3D reconstruction quality while maintaining fidelity to the observed 2D projections. This method draws inspiration from the X2CT architecture while introducing self-supervised training to overcome the lack of real 3D ground truth data.

3.e. Diffusion Model

Our 3D U-Net-based diffusion model up-samples 3D medical images at higher resolution. It applies a simplified U-Net architecture with residual blocks and skip connections to preserve the spatial features and ensure stable training. The model iteratively refines the noisy input during inference to produce cleaner high-resolution outputs. This method is particularly effective for tasks like image denoising. Although GAN-based approaches indeed work for photo-realistic image generation, they usually fail to preserve the fine-grained details, especially for such complex 3-D data as medical scans. The latter results in different artifacts, loss of texture, or at least some subtle features that may be crucial in medical imaging. The diffusion model overcomes these limitations by progressively refining the GAN output in several steps. In essence, through its noise-based denoising process, the diffusion model learns to add high-frequency details and improve the structural fidelity of the image. It acts like a post-processing step, enhancing resolution and smoothing out inconsistencies while preserving critical spatial features. The GAN uses this combination to generate a coarse but realistic image that the diffusion model further refines into a clean and higher-resolution output suitable for medical analysis and visualization. The results from the trained diffusion model did not produce the expected result.Thus we went ahead with a pretrained model, the results of which are shown in Fig3. It shows the original image and the enhanced image after running through the model.

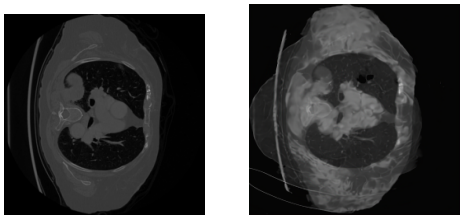

Fig 3: Original & Enhanced Image

4. Datasets

We had worked with 2 different datasets: a dataset of biplanar 2D chest XRay images provided by researchers at Indiana University and the dataset provided by our main paper of reference, i.e., the X2CT dataset. While we did not extensively use the X2CT dataset, some preliminary analysis was done using the same code and dataset provided by the researchers to replicate their results. The X2CT dataset was very memory intensive comprising 9 files of 7-8 GB each. However, the rest of our project involved the use of the Indiana University dataset which consisted of biplanar 2D chest XRay images in the frontal and lateral configurations, with a total size of 1.38 GB.

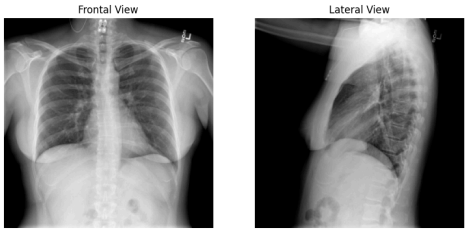

Fig 4: Input images

There were 3818 frontal and 3648 lateral images and the size range for these images was (512,362) to (512, 873). The researchers who developed this dataset acquired them from an NIH database in the DICOM standard and converted them into PNG format. For us, the first step was to make sure every frontal XRay image had a corresponding lateral image. In case some files were missing one of these, they were removed from the analysis. We eventually worked with 3193 frontal and 3193 lateral images. Preprocessing included normalizing to [0,1] and resizing to a uniform size according to our analyses: 64x64 for a few experiments and 128x128 for the other experiments. Augmentations done included random rotations. Finally the images were converted to tensors and the data was split into a 90-10 train-test split.

5. Evaluation Metrics

The chosen metrics were in line with the main paper of reference, the X2CTGAN [1]: Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index (SSIM). PSNR refers to the ratio of the maximum possible signal power of the reconstructed image to the maximum corrupting noise signal in the image. The higher this value is, the better. It is measured in decibels, dB. SSIM is a metric used to evaluate the

similarity between 2 images: the original image and the reconstructed image in this case.

PSNR (Peak Signal-to-Noise Ratio)

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

Where:

- MAX: The maximum possible pixel intensity (e.g., 1.0 for normalized data).
- MSE: The Mean Squared Error between the clean and enhanced images.

Fig 5: PSNR formula

SSIM (Structural Similarity Index)

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

Where:

- x: Clean (ground truth) image.
- y: Enhanced (model output) image.
- μ_x, μ_y: Mean intensities of x and y.
- σ_x^2, σ_y^2: Variances of x and y.
- σ_{xy}: Covariance of x and y.
- C_1, C_2: Small constants to stabilize the division.

Fig 6: SSIM formula

Here in the case of unsupervised/weakly supervised learning, SSIM was calculated on a slice-by-slice basis where the 2D frontal slice of the generated 3D image was compared with the 2D frontal input image.

6. Results

Table 1: Evaluation of the models

MODEL	PSNR (dB)	SSIM
X2CTGAN	26.19	0.66
CycleGAN-B	11.22	0.26
CycleGAN-F	5.91	0.011
u-X2CTGAN	10.91	0.07
3DCNN	29.46	0.31
cGAN-128	18.12	0.23
Diffusion	40.15	-

6.a. 3DCNNs

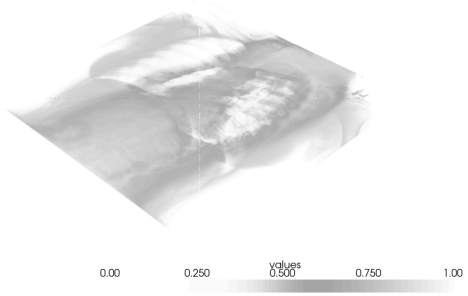

Fig 7: 3D generated Volumetric File

These volumes preserve spatial relationships across slices. This approach allows the model to leverage the spatial continuity in the data, making it particularly effective for tasks like identifying patterns or anomalies in medical scans.

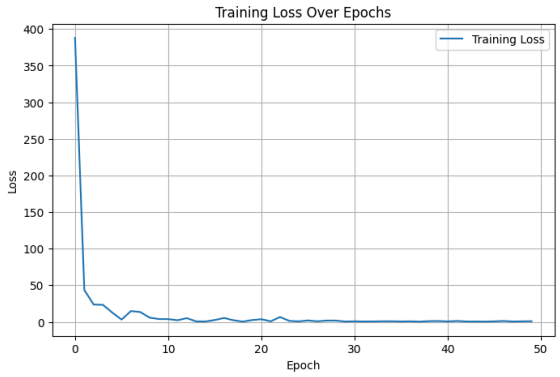

Fig 8: Loss for 50 Epochs for 3D CNN

The **loss function** used is the **cross-entropy loss**, which measures the difference between the predicted output and the true labels. It is particularly suitable for classification tasks, as it penalizes incorrect predictions more heavily when the model's confidence is high. During training, the optimizer (Adam) minimizes this loss, allowing the model to improve its predictions over multiple epochs.

6.b. CycleGANs

The CycleGANs were trained using 2 different inputs: once with biplanar images and once with just the frontal image. The loss plots of both the experiments were analyzed. As expected in many GAN-based frameworks, the losses showed very erratic patterns.

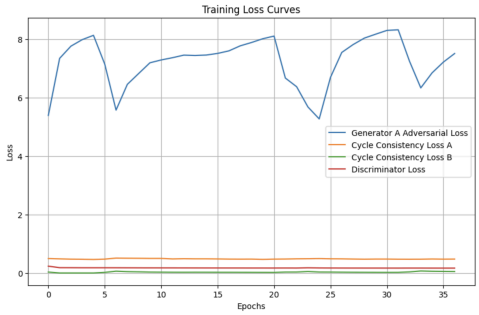

Fig 9: Loss plot of CycleGAN-B

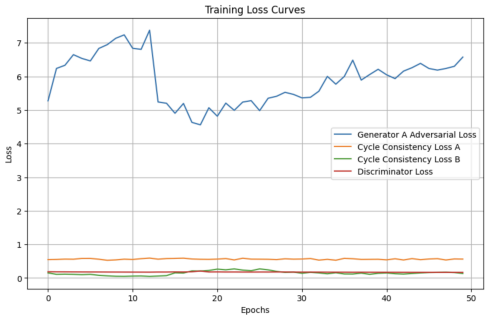

Fig 10: Loss plot of CycleGAN-F

The highly erratic nature of loss plots in a GAN are expected since GANs follow a very competing style of training where the generator and discriminator compete with each other and so the losses. Here however, the losses are high for the generator and low for the discriminator which mean that the generator is weak at producing viable outputs and the discriminator is too strong at distinguishing. The stagnant nature of the losses of the discriminator are also a point of concern.

6.c. u-X2CTGAN

Generator Loss Calculation

The Generator loss[Refer to Fig 11] in this project is calculated as a weighted sum of three key components: **Projection Consistency Loss**, **Shape Coherence Loss**, and **Adversarial Loss**. These losses ensure that the generated 3D volumes produce accurate 2D projections, exhibit structural smoothness, and appear realistic to the Discriminator. Additionally, dynamic loss weighting is applied to balance their contributions during training.

1. Projection Consistency Loss: The Projection Consistency Loss ensures that the recalculated 2D projections (Frontal and Lateral views) from the generated 3D volume align with the input Frontal and Lateral X-ray images. From the generated 3D volume:

Frontal projection is obtained by summing along the depth axis. Lateral projection is obtained by summing along the width axis. These projections are resized to match the input image resolution (512x512), and the loss is computed using the Mean Squared Error (MSE).

$$L_{proj} = MSE(frontal\_projection, frontal\_input) + MSE(lateral\_projection, lateral\_input)$$

2. Shape Coherence Loss: The **Shape Coherence Loss** regularizes the generated 3D volume to ensure smoothness and structural coherence. It uses **Total Variation (TV) Loss**, which penalizes abrupt changes along the spatial dimensions (x, y, and z axes): Mathematical Expression:

$$L_{shape} = mean(|x_{i+1} - x_i|) + mean(|y_{i+1} - y_i|) + mean(|z_{i+1} - z_i|)$$

This encourages the Generator to produce 3D volumes with smooth transitions and minimal artifacts.

3. Adversarial Loss: The Adversarial Loss ensures that the Generator produces realistic projections that can "fool" the Discriminator. The Discriminator evaluates the generated Frontal and Lateral projections, and the loss is computed using Binary Cross-Entropy (BCE) with target labels set as real (1): Mathematical Expression:

$$L_{adv} = \frac{1}{2}\left[BCE(D(fake\_frontal), 1) + BCE(D(fake\_lateral), 1)\right]$$

4. Dynamic Weighting: To balance the contributions of the three loss components, **dynamic loss weighting** is applied based on their magnitudes at each training iteration. The weights are defined as:

$$\lambda_{proj} = \frac{1}{L_{proj} + \epsilon}, \quad \lambda_{shape} = \frac{0.1}{L_{shape} + \epsilon}, \quad \lambda_{adv} = \frac{1}{L_{adv} + \epsilon}$$

where ϵ\epsilon is a small constant to prevent division by zero. The weights are normalized to ensure their sum equals 1.

5. Total Generator Loss: The final Generator loss is computed as the weighted sum of the three components:

$$L_G = \lambda_{proj} L_{proj} + \lambda_{shape} L_{shape} + \lambda_{adv} L_{adv}$$

Additionally, an **L2 regularization term** is added to penalize large parameter values in the Generator and prevent overfitting.

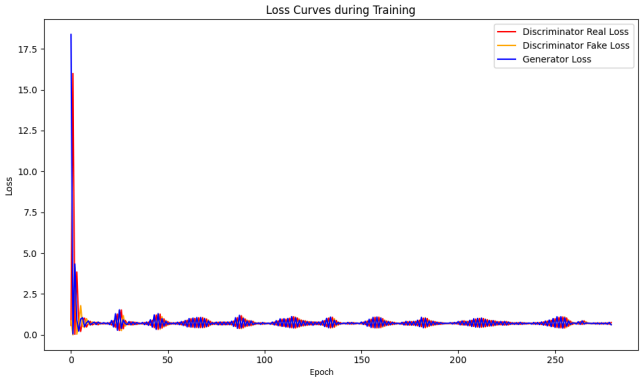

Fig 11: Loss Plot (u-X2CTGAN) for 150 epochs.

7. Conclusion

Employing not fully supervised learning in a task like 3D reconstruction brought with it several challenges. The lack of ground truth 3D images presented some very unique problems: a lot of more training is needed for the models to learn underlying patterns, i.e., increased training time. Along with increased time, the models would also benefit from having a lot more data, even if it is just 2D biplanar images. One common problem that we faced during training was the stagnating-loss issue: the losses stagnated and remained stuck at a value without reducing or increasing further. This would also happen pretty early on in the training process. To tackle this problem, we experimented by tweaking hyperparameters, but the results did not improve greatly. Upon further inspection, it was found that the input images also contained extra unnecessary components such as the body frame of the patient and the background. With more data, proper preprocessing and more robust architectures, it can be beneficial to explore more research in this area. As part of our future work, we can perform frontal and lateral lung masking (to extract only the region of interest) and then load the model with either a pretrained model that generates 3D images from 2D XRay biplanar images as input or give the generator a really small portion of the 3D ground truth data in order for it to understand what it has to generate.

References

1) Ying, X., Guo, H., Ma, K., Wu, J., Weng, Z. Zheng, Y., 2019. X2CT-GAN: Reconstructing CT from Biplanar X-Rays with Generative Adversarial Networks. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10611–10620. Available at: https://doi.org/10.1109/CVPR.2019.01087
2) Y. Wang, Z.-L. Sun, Z. Zeng, and K.-M. Lam, "TRCT-GAN: CT reconstruction from biplane X-rays using transformer and generative adversarial networks," Digital Signal Processing, vol. 140, p. 104123, 2023. doi: 10.1016/j.dsp.2023.104123
3) Iacono, Paolo & Khan, Naimul. (2023). Structure Preserving Cycle-GAN for Unsupervised Medical Image Domain Adaptation. 10.32920/22734377.v1.
4) Han, C., Rundo, L., Murao, K. et al. MADGAN: unsupervised medical anomaly detection GAN using multiple adjacent brain MRI slice reconstruction. BMC Bioinformatics 22 (Suppl 2), 31 (2021). https://doi.org/10.1186/s12859-020-03936-1
5) Liu, Feng and Xiaoming Liu. “2D GANs Meet Unsupervised Single-view 3D Reconstruction.” ArXiv abs/2207.10183 (2022): n. pag.
6) Ma, J., Zhu, Y., You, C., & Wang, B. (2023). Pre-trained diffusion models for plug-and-play medical image enhancement. In H. Greenspan, A. Madabhushi, P. Mousavi, S. Salcudean, J. Duncan, T. Syeda-Mahmood, & R. Taylor (Eds.), *Medical Image Computing and Computer Assisted Intervention -- MICCAI 2023* (pp. 3–13). Springer Nature Switzerland.

Appendix A. Detailed Roles

Table 1. Team member contributions

Name	Task	File names	No. Lines of Code
Adwait	Preprocessing, u-X2CTGAN	u-X2CTGAN	600
Avantika	Preprocessing, CycleGAN-B, CycleGAN-F	CycleGAN-B, CycleGAN-F	590
Harshvardhan	Preprocessing, Diffusion, Postprocessing	Diffusion Model	350
Shivam	Preprocessing, 3D CNN, cGAN-128	3DGans, 3DCNN	400

Appendix B. Code repository

https://github.com/WalnutEagle/EC-523-Deep-Learning-Project