ECE 228 - Final Report - Team 22

Chris Light
UCSD - ECE
cdlight@ucsd.edu

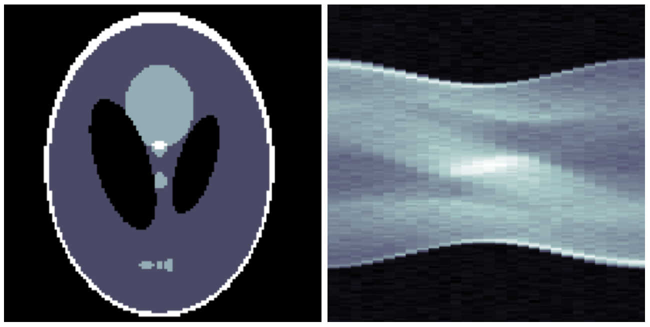
Michael Ingerman
UCSD - MAE
mingerman@ucsd.edu

1 Introduction

Computed Tomography is one of the most prevalent forms of medical imaging available in the modern world. Albeit invaluable to doctors in medical decision making, CT scanning holds a risk to patients as the use of harmful X-ray radiation poses health concerns. There is a delicate balance of not exposing the patient with too much radiation while still producing a meaningful picture that doctors can use to make informed medical decisions from. Traditional signal processing methods in CT imaging is bounded by the Nyquist rate, which is a lower limit to the number of projection angles necessary to perform image construction. Reducing the number of angles necessary for a meaningful image means less radiation exposure - and a necessity to break the Nyquist criteria via the use of a Deep Neural Network.

In this paper we adapt a method called Deep Back Projection [1] for the purposes of reconstructing CT chest images of Covid19 patients using sparse imaging with a deep convolution neural network (CNN) framework. The advantages of using a CNN is that it reduces the number of parameters of deep neural network by imposing spatial invariance on image data. Using the Harvard's "COVID19-CT-Dataset"[2] we first process each CT image by producing a set of sinogram slice projections (or back projections) to form a stack of images of parallel lines at different orientations. We then feed this stack as input into our model during training in an attempt to employ learning of the inverse radon transform.

All work performed can be accessed via the Team 22 Github Repository

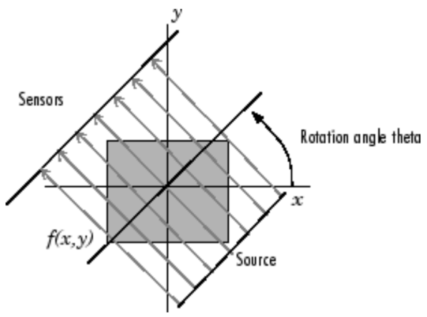


(a) Shepp-Logan phanton (b) Shepp-Logan sinogram

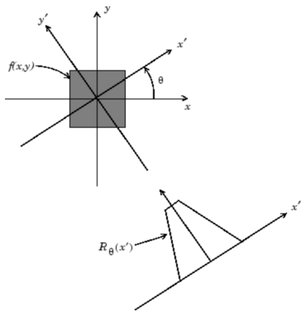
Figure 1: Left image is a Shepp-Logan Phantom. Right image is a Corresponding sinogram.

2 Background

The Radon transform function computes the projections of an image matrix along a specified direction. For a 2D image, the projections of the image data found via radon transform can be considered a set



(a) Parallel-Beam Projection

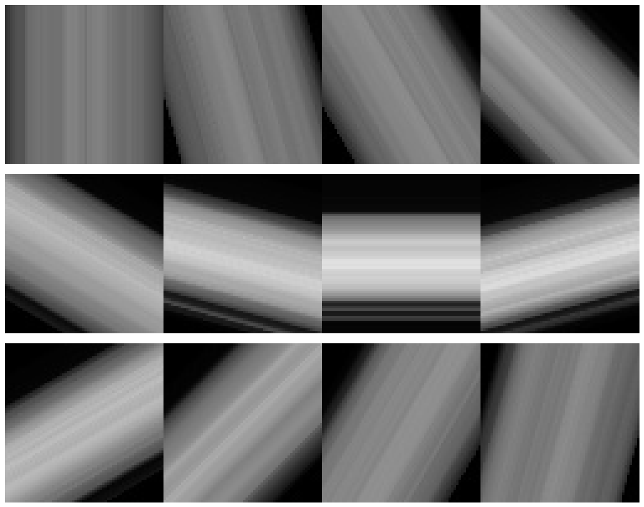


(b) Radon Transform on axis

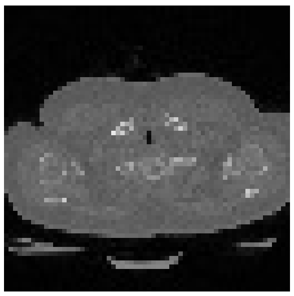
Figure 2: Left image showcases how parallel signals are sent at a given orientation of a source. Right image shows how signals are interpreted through Radon Transform to yield the integral intensities of a signals along an axis.

of line integrals of pixel intensities along parallel paths as can seen in fig. 2. By sampling axes at various orientations, a set of random transforms can be combined to form a sinogram image like in fig 1. Furthermore, for each orientation of sinogram projection computed via radon transform, the signal intensities for each projection can be cast as parallel lines of uniform intensities along the direction of the original signal as can be seen in fig. 3. The resulting images with parallel lines of uniform intensities are called back-projected slice sinogram images. Collectively, these back-projected images are useful since they represent spatial-based information in a non-spatially dependent way. This makes back-project images a convenient input for a convolution neural network style model.

Loosely speaking and in the case of square images, the number of sinogram back projections needed for perfect (or near perfect) reconstruction is about the number of pixels along a given dimension. For example, for an image of size 160×160 pixels, the corresponding sinogram will be of size [ceil($\sqrt{2} \times 160$), 160]. The vertical dimension of the sinogram is necessary for angles that result in a max distance across while performing the line integral.



(a) 12 Generated Sinogram Back Projections



(b) Truth CT Chest Scan Image

Figure 3: Radon transform is applied to a CT chest scan(truth image) using 12 angles to create a CT sinogram with fixed step size between $0° - 180°$. Single-view back projections are stacked to form the input for the convolution neural network.

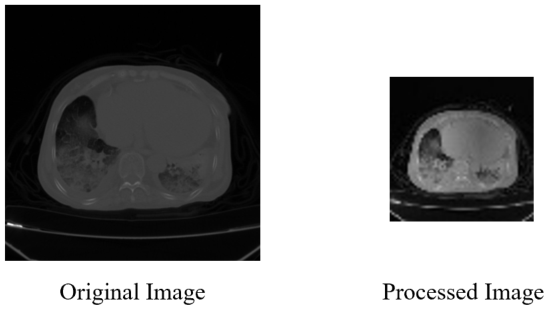


Figure 4: Left shows the original image from Harvard's "COVID19-CT-Dataset"[2]. Right is image after preprocessing.

3 Dataset and Framework

3.1 Dataset Processing

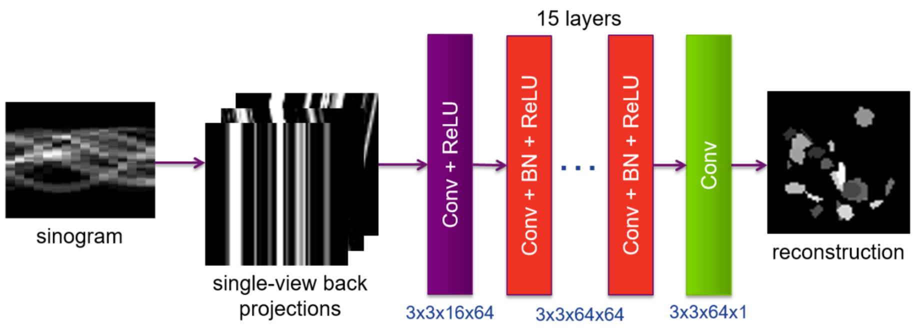
Truth images for our model were gathered from Harvard's "COVID19-CT-Dataset"[2]. Each truth image was converted to a set of sinogram back projections at different orientations θ_i. Every set of sinogram back projection images y_i was found by first computing a sinogram y from a given truth image with a set of n angles. The number of slices n corresponding to the range of angles was variably selected in order to gauge how the model preformed as a function of data sparsity. Then each set of sinogram projection $y_i \in R^n$ were generated by extracting projections along the set of angles $\theta_i \in R^n$ for a given sinogram image y.

All datasets used for model training and verification contained 1000 randomly selected images from a dataset of 28,000 and first downsized from their original (512, 512) pixels to (160,160) pixels in an effort to reduce the number of trainable parameters as can be see in figure 4. Furthermore, in an attempt to reduce the complexity of the image, such as remove the clothing of the patient in the image, dilation and erosion techniques were applied. However, after this image processing and general image blurring, the overall contours of the image were significantly blurred making the images difficult to use for training. To mitigate this issue, an equalization histogram filter was applied to sharpen the contours of the image and allow for the image data to span the entirety of the data type, uint8.

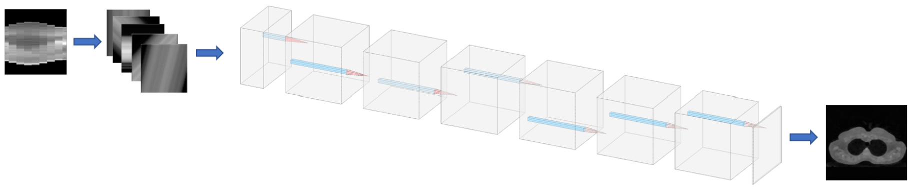
3.2 Framework Architecture

Following the work of [3], we construct a similar architecture as can be seen in fig. 5.

From the dataloader, a torch tensor of size [b,n+1,h,w] is fed into the network; For b = batch size, n = number of angles, (h,w) = dimension of the pictures. The first layer of the model conforms to n input channels (the number of slice projections) and 64 output channels with ReLU activation. 15 layers of Conv > batch normalization > ReLU with a consistent 64 channels follows and finally compressed into one channel at the last layer as the resulting image is produced. The kernel size is consistent at (3,3) and padding is set to preserve the original CT image resolution.



(a) Network found in [3]



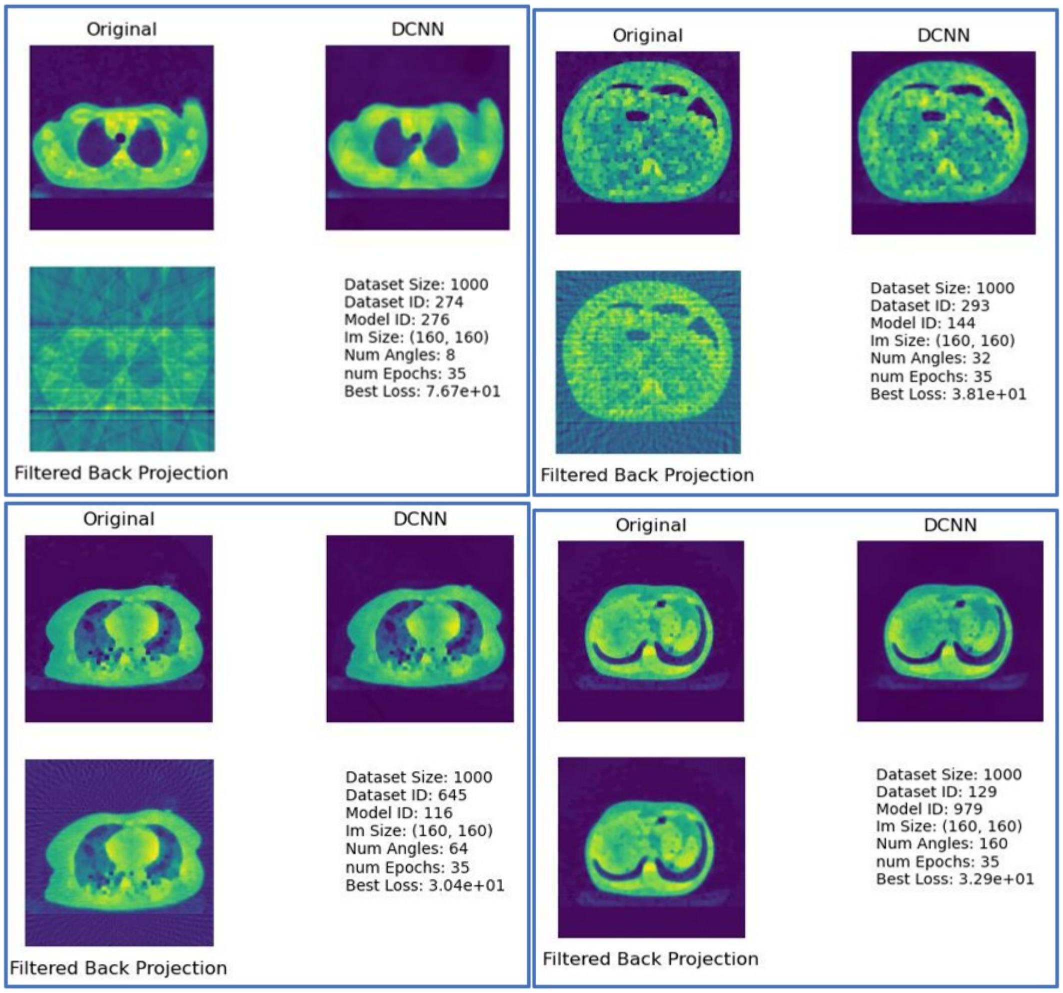
(b) Our Network

Figure 5: Network Architecture

4 Results

The results gained throughout this project were without significant hyperparameter tuning with the exception of changing the number of input images. A batch size of 10 was used along with a learning rate 5e-2, weight decay 1e-5, and MSE loss. The optimizer was AMSGrad, which is an extension of the Adam Optimizer. We are also used ExponentialLR scheduler with gamma = .912.

Our results are shown below in figure 6, with each quadrant representing a model trained with different numbers of input images in order to visualize how data sparsity affects the training and performance of our DCNN architecture. From observation it is clear that increasing the number of input images also increases the clarity and decreases the loss of a given model. Furthermore, it is interesting to note that increasing the number input images significantly improves performance when comparing the the difference between 8 input images and 32. However, when comparing the model performance between 32 and 64 images, the best loss is marginally better. This trend suggest that there exist a threshold number of input images after which model performance improvement is negligible, which is analogous to the Nyquist limit for methods such as Radon Transformation.



Dataset Size: 1000
Dataset ID: 274
Model ID: 276
Im Size: (160, 160)
Num Angles: 8
num Epochs: 35
Best Loss: 7.67e+01

Dataset Size: 1000
Dataset ID: 293
Model ID: 144
Im Size: (160, 160)
Num Angles: 32
num Epochs: 35
Best Loss: 3.81e+01

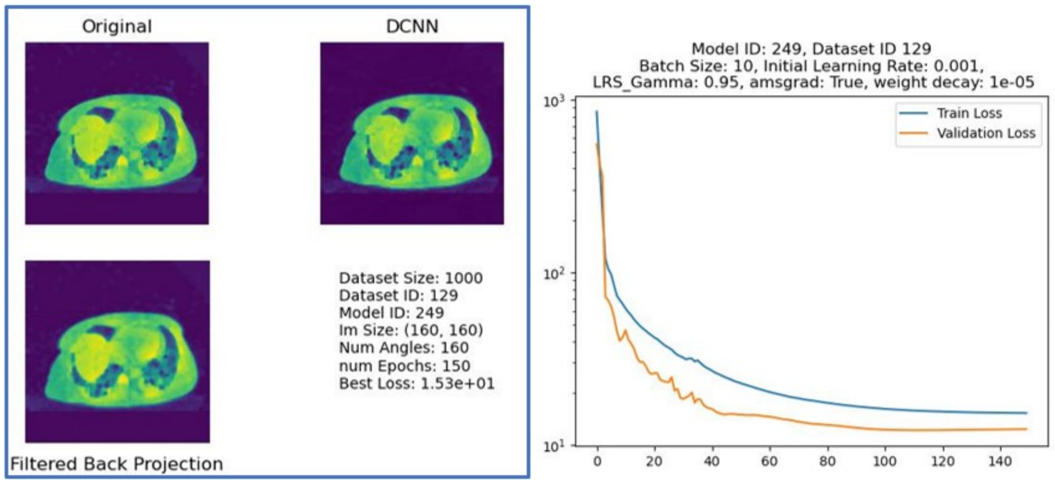
Dataset Size: 1000
Dataset ID: 645
Model ID: 116
Im Size: (160, 160)
Num Angles: 64
num Epochs: 35
Best Loss: 3.04e+01

Dataset Size: 1000
Dataset ID: 129
Model ID: 979
Im Size: (160, 160)
Num Angles: 160
num Epochs: 35
Best Loss: 3.29e+01

Figure 6: Each quadrant represents a different model trained with the same set of hyper parameters but using different numbers of slices image inputs. Furthermore, each quadrant contains not only the original truth image and the output from the corresponding model, but also the equivalent computed image using the same number of slice projection images using conventional radon transform methods. This is to illustrate how the DCNN is superior in dealing with data sparsity compared to conventional signal processing methods.

Additionally, in an attempt observe how many epochs it would require for the training loss to converge, we trained a model with 150 epochs. From figure 7, it can be observed that the loss for our architecture for this particular dataset converges around 70 epochs. Furthermore, it's interesting to note that the validation error is lower than the training loss. Our only explanation for this phenomenon is that the validation set contain images from the same set of patient images as the training set.

During construction and testing of our DCNN, there was a critical issue that was interfering with the network learning. Our original procedure was to have separate scripts, one to pre-process data and save back projections as .png files into folders, then another script to load the back projection .png files into a tensor for training. We surmise that this process imposed a learning barrier as the data types changed multiple times with read/write functions. We saw significant improvement when our datasets were generated in a single script from truth image to back projections to data-stacked tensors saved as a .pt file type.



Dataset Size: 1000
Dataset ID: 129
Model ID: 249
Im Size: (160, 160)
Num Angles: 160
num Epochs: 150
Best Loss: 1.53e+01

Model ID: 249, Dataset ID 129
Batch Size: 10, Initial Learning Rate: 0.001,
LRS_Gamma: 0.95, amsgrad: True, weight decay: 1e-05

Figure 7: The left image shows the original image, and resulting image from our DCNN model, as well as an image computed with the same amount of input data using conventional radon transform methods. The right image is shows training and validation loss for this model.

5 Future Improvements

The work showcased in [3] is a testament to the model's optimality for purposes such as ours. However, further consideration could be placed on training models without compressing the original image as was done in our experimentation. This would endow the model with significantly more information, and thus more parameters, which would require more processing capabilities. Another consideration to be made is observing how changing hyperparameters could affect overall model learning and performance. And finally, a study on more minimal preprocessing could be done, where the fine details are not simplified and left to be reconstructed by the model.

6 Conclusion

Overall, we were able to showcase that a deep convolution neural network can be used to achieve similar results with sparse data to that of conventional CT image reconstruction techniques on compressed and simplified images. Furthermore, with our preprocessed dataset of (160,160) pixel size images, we observed that increasing the number of input images, from 64 to 160, had a negligible effect on image clarity which suggest an analagous effect to that of the Nyquist parameter for our model.

7 Individual Contributions

Chris Light

I had a major role on developing scripts relevant to dataset generation, training, and the model itself. All the hyperparameter tuning, troubleshooting, and experimentation was performed on my local machine and by my actions.

Michael Ingerman

I had a major role in selecting and preprocessing the data appropriately before it could be input into the model. This included doing the radon transformations in order to get slice projection images from the raw data images and applying computer vision techniques (dilation, erosion, histogram filter) to prep the images for training.

References

[1] Jonas Adler and Ozan Öktem. Learned primal-dual reconstruction. *IEEE Transactions on Medical Imaging*, 37(6):1322–1332, 2018.

[2] Sayyed Mostafa Mostafavi. COVID19-CT-Dataset: An Open-Access Chest CT Image Repository of 1000+ Patients with Confirmed COVID-19 Diagnosis, 2021.

[3] Dong Hye Ye, Gregery T. Buzzard, Max Ruby, and Charles A. Bouman. Deep back projection for sparse-view ct reconstruction, 2018.