Deep Learning Framework for Inverse Problems

by
Shrishti Agarwal
(MA21MSCST11006)
Advisor: Dr. C.S. Sastry

August 16, 2025

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Organization

Introduction to Computed Tomography

- The goal of CT is to obtain a representation of the internal structure of an object by X-raying it from many different directions.

Introduction to Computed Tomography

- The goal of CT is to obtain a representation of the internal structure of an object by X-raying it from many different directions.
- Consider an object $O(x, y, z)$ as a superposition of n layers of the same thickness along z axis, all located in planes parallel to the plane (x, y) and perpendicular to z (Fig. 1). Each layer is considered as a 2D function $f(x, y)$, which is a section in the object to be reconstructed

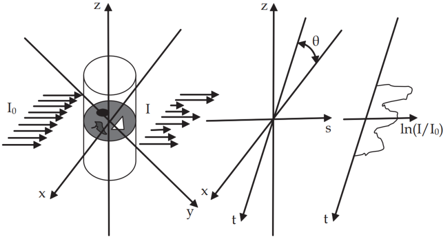
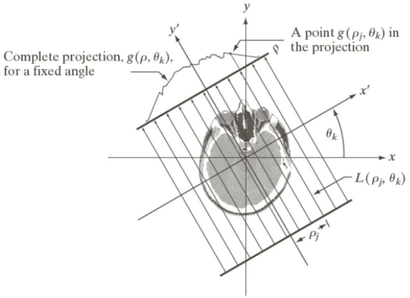


Figure: Data acquisition in CT

Radon Transform

- An arbitrary point (ρ_j, θ_k) in the projection signal is given by the ray-sum along the line $x\cos\theta_k + y\sin\theta_k = \rho_j$.
- The ray-sum is a line integral:

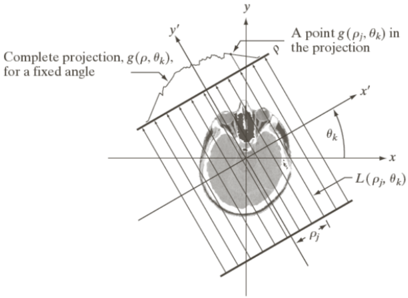
$$g(\rho_j, \theta_k) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \delta(x\cos\theta_k + y\sin\theta_k - \rho_j) f(x,y)dxdy \quad (1)$$



- For all values of ρ and θ, we obtain the Radon transform:

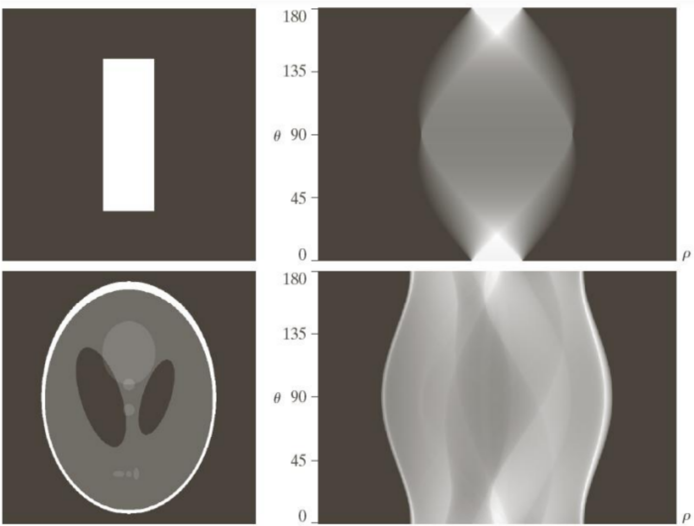
$$g(\rho,\theta) = R_\theta f = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \delta(x\cos\theta + y\sin\theta - \rho) f(x,y)\,dx\,dy. \quad (2)$$

'



Contd..

The representation of the Radon transform $g(\rho, \theta)$ as an image with ρ and θ as coordinates is called a **sinogram**.



Projection-Slice Theorem

The $1D$ Fourier transform of the projection at given angle θ is the slice of the $2D$ Fourier transform of the image.

$$G(\omega, \theta) = F(\omega \cos\theta, \omega \sin\theta), \tag{3}$$

where $G(\omega, \theta)$ is $1D$ Fourier transform of $g(\rho, \theta)$ and $F(\omega \cos\theta, \omega \sin\theta)$ is the $2D$ Fourier transform of image $f(x, y)$.

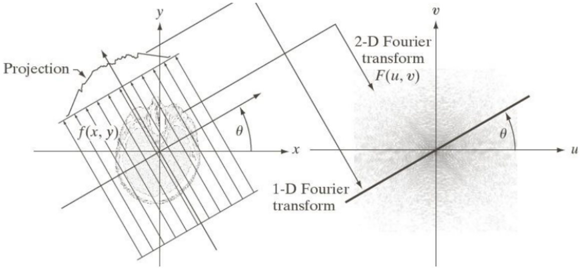


Figure: Here, $u = \omega \cos\theta$ and $v = \omega \sin\theta$

Filtered Back Projection

- The $2D$ inverse Fourier Transform of $F(\omega\cos\theta, \omega\sin\theta)$ is:

$$f(x,y) = \int_0^{2\pi}\int_0^{+\infty} F(\omega\cos\theta, \omega\sin\theta) e^{2\pi i\omega(x\cos\theta + y\sin\theta)}\omega d\omega d\theta.$$

Filtered Back Projection

- The $2D$ inverse Fourier Transform of $F(\omega\cos\theta, \omega\sin\theta)$ is:

$$f(x,y) = \int_0^{2\pi}\int_0^{+\infty} F(\omega\cos\theta, \omega\sin\theta)e^{2\pi i\omega(x\cos\theta + y\sin\theta)}\omega d\omega d\theta.$$

- By Projection-Slice Theorem and $\rho = x\cos\theta + y\sin\theta$, we have

$$f(x,y) = \int_0^{\pi}\left[\int_{-\infty}^{+\infty} |\omega| G(\omega,\theta)e^{2\pi i\omega\rho}d\omega\right]d\theta.$$

For a given angle θ, the inner expression is the $1D$ Fourier transform of the projection multiplied by a ramp filter $|\omega|$ which is equivalent to filtering the projection with a high pass filter before back-projection.

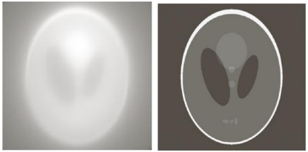


Figure: Back Projection V/S FBP Reconstruction

Deep Learning

- Deep learning algorithms are modeled on the structure and function of the human brain, and they are capable of performing complex tasks such as image and speech recognition, natural language processing, and autonomous driving.

- Deep learning algorithms are modeled on the structure and function of the human brain, and they are capable of performing complex tasks such as image and speech recognition, natural language processing, and autonomous driving.
- **MLP** consist of multiple layers of artificial neurons that are organized into a hierarchical structure. Each layer of the neural network extracts and transforms the input data into a more useful representation.

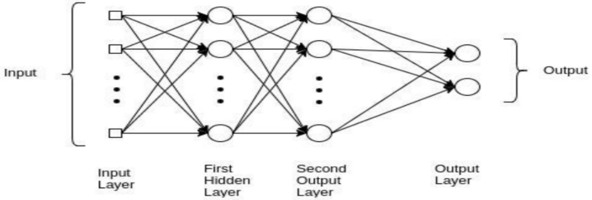


Figure: The above diagram is an example of A Multilayer Perceptron (MLP) is a feedforward artificial neural network that generates a set of outputs from a set of inputs

Need For CNN

- Consider passing an RGB image of size 200×200 through Multilayer Perceptron.

Need For CNN

- Consider passing an RGB image of size 200×200 through Multilayer Perceptron.
- Vectorizing it leads to $200 \times 200 \times 3 = 120K$ neurons in input layer.

Need For CNN

- Consider passing an RGB image of size 200×200 through Multilayer Perceptron.
- Vectorizing it leads to $200 \times 200 \times 3 = 120K$ neurons in input layer.
- A hidden layer of same size leads to $\equiv 1.44e^{10} \implies$ 58GB which leads to hardware limit, overfitting etc.

Convolution Neural Network

Need For CNN

- Consider passing an RGB image of size 200×200 through Multilayer Perceptron.
- Vectorizing it leads to $200 \times 200 \times 3 = 120K$ neurons in input layer.
- A hidden layer of same size leads to $\equiv 1.44e^{10} \implies$ 58GB which leads to hardware limit, overfitting etc.
- Flattening removes the Structure.

Need For CNN

- Consider passing an RGB image of size 200×200 through Multilayer Perceptron.
- Vectorizing it leads to $200 \times 200 \times 3 = 120K$ neurons in input layer.
- A hidden layer of same size leads to $\equiv 1.44e^{10} \implies$ 58GB which leads to hardware limit, overfitting etc.
- Flattening removes the Structure.
- **Convolution** incorporates idea of applying same linear operation at all locations and preserve the structure.

contd...

- **Convolution** is the process where we take a small matrix of numbers (called kernel or filter), pass it over an image, and then transform it based on the values from the filter.

$$S_{ij} = (I * K)_{ij} = \sum_{a=\lfloor -\frac{m}{2} \rfloor}^{\lfloor \frac{m}{2} \rfloor} \sum_{b=\lfloor -\frac{n}{2} \rfloor}^{\lfloor \frac{n}{2} \rfloor} I_{i-a,j-b} K_{\frac{m}{2}+a,\frac{n}{2}+b}$$

contd...

- **Convolution** is the process where we take a small matrix of numbers (called kernel or filter), pass it over an image, and then transform it based on the values from the filter.

$$S_{ij} = (I * K)_{ij} = \sum_{a=\lfloor -\frac{m}{2} \rfloor}^{\lfloor \frac{m}{2} \rfloor} \sum_{b=\lfloor -\frac{n}{2} \rfloor}^{\lfloor \frac{n}{2} \rfloor} I_{i-a,j-b} K_{\frac{m}{2}+a, \frac{n}{2}+b}$$

- Consider $n-$ Size of Image,$k-$ Size of filter, n_c- number of channels, $p-$ padding size,$s-$ stride size, n_k-number of filters. The size of the feature map obtained is given by:

$$[n, n, n_c] * [k, k, n_c] = \left[\left\lfloor \frac{n+2p-k}{s} \right\rfloor + 1, \left\lfloor \frac{n+2p-k}{s} \right\rfloor + 1, n_k \right]$$

- **Convolution** is the process where we take a small matrix of numbers (called kernel or filter), pass it over an image, and then transform it based on the values from the filter.

$$S_{ij} = (I * K)_{ij} = \sum_{a=\lfloor -\frac{m}{2} \rfloor}^{\lfloor \frac{m}{2} \rfloor} \sum_{b=\lfloor -\frac{n}{2} \rfloor}^{\lfloor \frac{n}{2} \rfloor} I_{i-a,j-b} K_{\frac{m}{2}+a,\frac{n}{2}+b}$$

- Consider $n-$ Size of Image, $k-$ Size of filter, n_c- number of channels, $p-$ padding size, $s-$ stride size, n_k-number of filters. The size of the feature map obtained is given by:

$$[n, n, n_c] * [k, k, n_c] = \left[\left\lfloor \frac{n+2p-k}{s} \right\rfloor + 1, \left\lfloor \frac{n+2p-k}{s} \right\rfloor + 1, n_k \right]$$

Input Matrix

45	12	5	17
22	10	35	6
88	26	51	19
9	77	42	3

Kernel

0	-1	0
-1	5	-1
0	-1	0

⇒

Result

-45	12
22	10

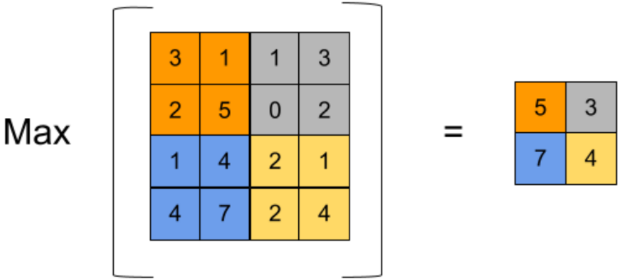
Pooling

Besides Convolution layers, CNNs often use pooling layers which are used primarily to reduce the size of the tensor and speed up calculations.

- **Max-Pooling:** The pooling operator maps a sub-region to its maximum value.

$$y_{m,n,d} = \max_{0 \leq i < H, 0 \leq j < W} x_{mH+i, nW+j, d}$$

where $0 \leq m < H_2, 0 \leq n < W_2$ and $0 \leq d < K_2 = K$

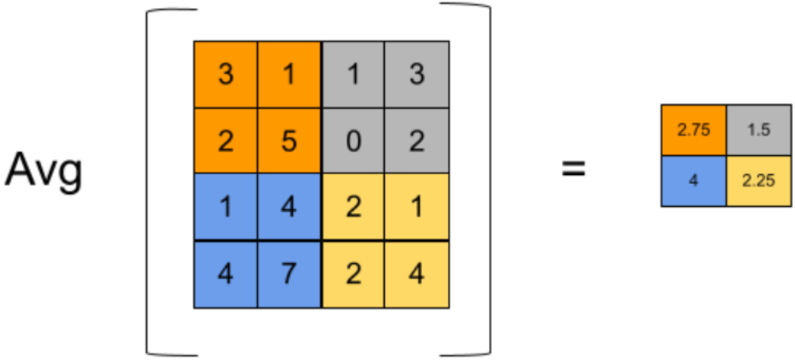


Pooling

Average Pooling: The pooling operator maps a sub-region to average value obtained by sub-region.

$$y_{m,n,d} = \frac{\sum_{0 \leq i < H, 0 \leq j < W} x_{mH+i, nW+j, d}}{H}$$

where $0 \leq m < H_2, 0 \leq n < W_2$ and $0 \leq d < K_2 = K$



Fully Connected Layers

After passing the image through the feature learning process using the CONV+POOL layers, we now have extracted all the features from this image and put them all in the long tube of features. Now, we have to classify the images based on the presence of these features in them.

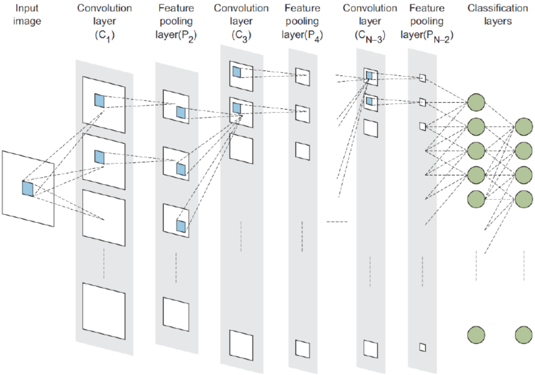


Figure: Overall CNN Architecture, the last layer represents the Fully Connected layer.

U-NET

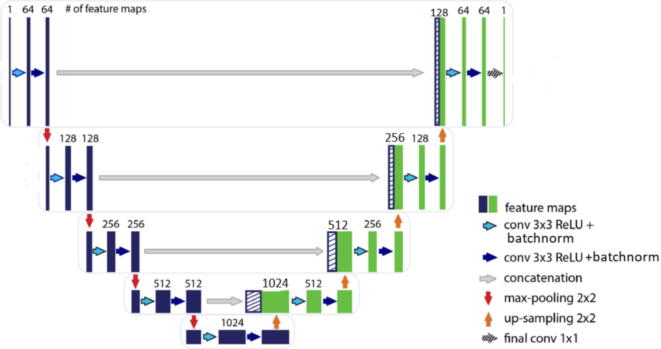


Figure: Illustration of Standard U-NET Architecture:

- Encoder - It reduces the height and width of the input images.

U-NET

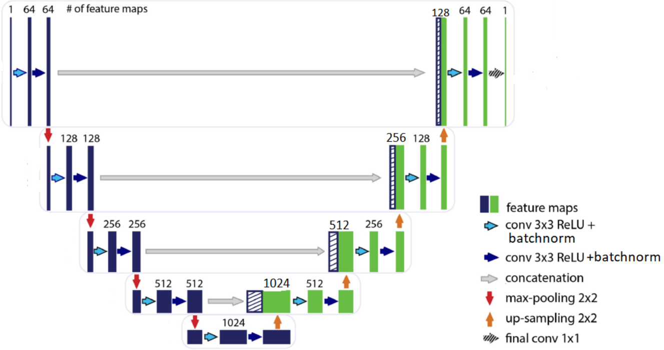


Figure: Illustration of Standard U-NET Architecture:

- Encoder - It reduces the height and width of the input images.
- Decoder - It recovers the original dimensions of the input images.

Residual U-NET

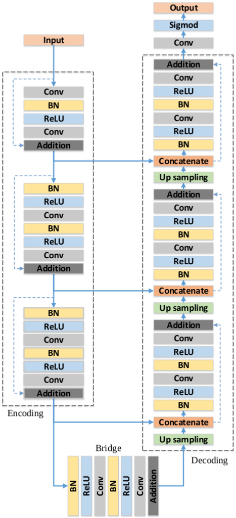


Figure: Illustration of Residual U-Net Architecture:

contd..

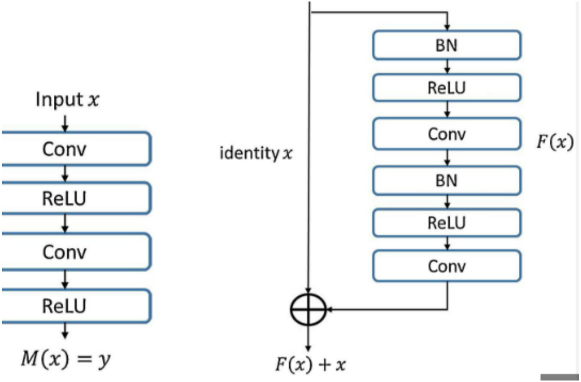


Figure: (a) The conventional feed-forward neural network used in U-Net, (b) residual unit consists of an identity map utilized in Residual U-Net

Inverse Problem: A brief introduction

Measurement Model :

$$y = \mathbb{M}x$$

Injectivity of Model :

$$\mathbb{M}x_1 = \mathbb{M}x_2 \implies x_1 = x_2$$

Inverse of Measurement Model :

$$\mathbb{M}^{-1}(y) = x$$

Stability of Inverse :

$$\left\|\mathbb{M}^{-1}(y_1) - \mathbb{M}^{-1}(y_2)\right\| \leq \omega(\|y_1 - y_2\|)$$

where ω is monotonic and $\omega(0) = 0$ (for example, $\omega(\mu) = \epsilon\mu$.

Modified Inverse Problem :

$$y = \mathbb{M}x \text{ on some restricted x.}$$

Do CNNs solve CT inverse Problem?

Measurement Model in CT

$$g = Rf \tag{4}$$

where f stands for the pixel values in vector form;
R is a discrete-to-discrete linear transform representing the action of the Radon transform on the image pixels;
g indicates the sinogram values as shown in Fig-12.

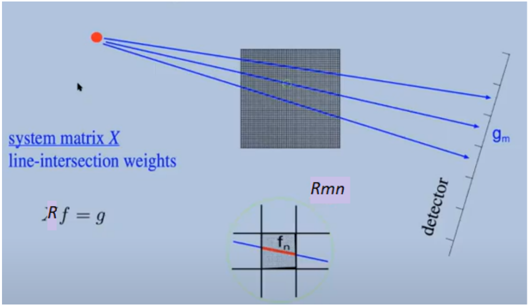


Figure: Algebraic Model for CT

Contd..

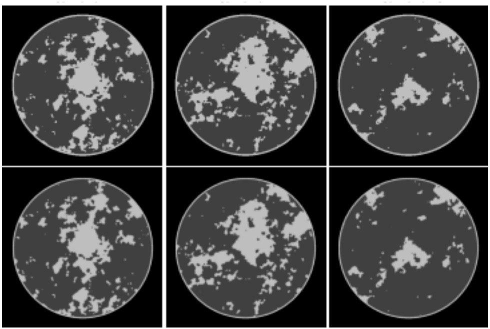


Figure: 512×512 Phantom Image. The first row shows binary class phantom realization where as second row shows smooth-edge class phantom.

$f_{smooth-edge} = G(\omega_0) f_{binary}, \ \omega_0 = 1$
Data acquisition set-up: 128 fan beam Projection, 512 detector pixel.

TV min reconstruction for Sparse CT

- **Measurement Model:**

$$g = Rf$$

where $f \in F_{GMI-sparse}$ and

$$F_{GMI-sparse} = \left\{ f \middle| \; \| \left(|Df|_{mag} \right) \|_0 \leq s \right\}$$

TV min reconstruction for Sparse CT

- **Measurement Model:**

$$g = Rf$$

where $f \in F_{GMI-sparse}$ and

$$F_{GMI-sparse} = \left\{ f \middle| \; \| \left(|Df|_{mag} \right) \|_0 \leq s \right\}$$

- **Proposed Inverse:**

$$f^\star = \arg\min_f \| \left(|Df|_{mag} \right) \|_1 \quad \text{such that } g = Rf$$

TV min reconstruction for Sparse CT

- **Measurement Model:**

$$g = Rf$$

where $f \in F_{GMI-sparse}$ and

$$F_{GMI-sparse} = \left\{ f \middle| \; \| \left(|Df|_{mag} \right) \|_0 \leq s \right\}$$

- **Proposed Inverse:**

$$f^\star = \arg\min_f \| \left(|Df|_{mag} \right) \|_1 \quad \text{such that } g = Rf$$

- $\| \left(|Df|_{mag} \right) \|_1$ is the total variation (TV) of image. This optimization problem is solved by CPPD.

CNN-based Sparse-view CT image reconstruction

Image reconstruction for a 2D sparse-view problem, where a 512×512 image is reconstructed from a sinogram of dimension 128 views by 512 detector bins.

- 4000 simulation phantoms are generated, 2000 of binary class phantom and 2000 for smooth-edge phantom.

CNN-based Sparse-view CT image reconstruction

Image reconstruction for a 2D sparse-view problem, where a 512×512 image is reconstructed from a sinogram of dimension 128 views by 512 detector bins.

- 4000 simulation phantoms are generated, 2000 of binary class phantom and 2000 for smooth-edge phantom.
- 128-view sinogram is generated by eq-4 which is further processed by FBP.

CNN-based Sparse-view CT image reconstruction

Image reconstruction for a 2D sparse-view problem, where a 512×512 image is reconstructed from a sinogram of dimension 128 views by 512 detector bins.

- 4000 simulation phantoms are generated, 2000 of binary class phantom and 2000 for smooth-edge phantom.
- 128-view sinogram is generated by eq-4 which is further processed by FBP.
- **Objective:** Obtain a "model" yielding true phantom from FBP reconstruction.

CNN-based Sparse-view CT image reconstruction

Image reconstruction for a 2D sparse-view problem, where a 512×512 image is reconstructed from a sinogram of dimension 128 views by 512 detector bins.

- 4000 simulation phantoms are generated, 2000 of binary class phantom and 2000 for smooth-edge phantom.
- 128-view sinogram is generated by eq-4 which is further processed by FBP.
- **Objective:** Obtain a “model” yielding true phantom from FBP reconstruction.
- **Input:** 128-view FBP reconstructed image.

CNN-based Sparse-view CT image reconstruction

Image reconstruction for a 2D sparse-view problem, where a 512×512 image is reconstructed from a sinogram of dimension 128 views by 512 detector bins.

- 4000 simulation phantoms are generated, 2000 of binary class phantom and 2000 for smooth-edge phantom.
- 128-view sinogram is generated by eq-4 which is further processed by FBP.
- **Objective:** Obtain a "model" yielding true phantom from FBP reconstruction.
- **Input:** 128-view FBP reconstructed image.
- **Target:** Corresponding true phantom image.

CNN-based Sparse-view CT image reconstruction

Image reconstruction for a 2D sparse-view problem, where a 512×512 image is reconstructed from a sinogram of dimension 128 views by 512 detector bins.

- 4000 simulation phantoms are generated, 2000 of binary class phantom and 2000 for smooth-edge phantom.
- 128-view sinogram is generated by eq-4 which is further processed by FBP.
- **Objective:** Obtain a "model" yielding true phantom from FBP reconstruction.
- **Input:** 128-view FBP reconstructed image.
- **Target:** Corresponding true phantom image.
- **Method:** Standard U-Net Architecture [3]

CNN-based Sparse-view CT image reconstruction

Image reconstruction for a 2D sparse-view problem, where a 512×512 image is reconstructed from a sinogram of dimension 128 views by 512 detector bins.

- 4000 simulation phantoms are generated, 2000 of binary class phantom and 2000 for smooth-edge phantom.
- 128-view sinogram is generated by eq-4 which is further processed by FBP.
- **Objective:** Obtain a "model" yielding true phantom from FBP reconstruction.
- **Input:** 128-view FBP reconstructed image.
- **Target:** Corresponding true phantom image.
- **Method:** Standard U-Net Architecture [3]
- **Output:** The difference between the 128-view FBP images and its phantom counterpart.

Results reported in “Do CNNs solve CT inverse problem ?”[1].

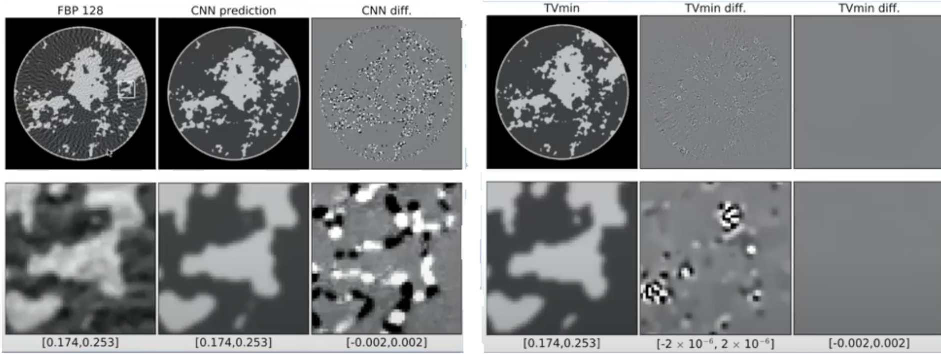


Figure: The RMSE for the TVmin result is $1.5 \times 10^{-6} cm^{-1}$ whereas RMSE reported for CNN reconstruction method is $6.76 \times 10^{-4} cm^{-1}$

[1] IEEE Trans. Biomedical Engg, 2021.

About Dataset

Data Set is same as provided for **AAPM Grand Challenge: "Deep Learning for Inverse Problems: Sparse-View Computed Tomography Image Reconstruction"**.
A fan beam geometry with 128 projections over 360 degrees was used to create sinograms and FBPs.

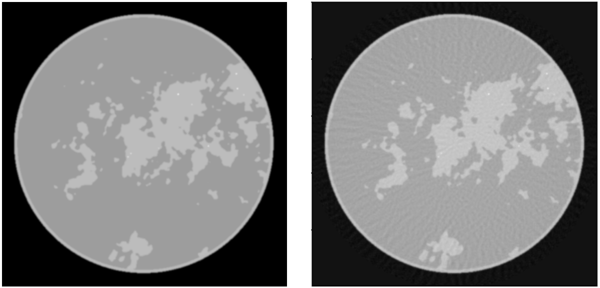


Figure: Example of 512×512 ground-truth Phantom image and its corresponding 128-view FBP Reconstruction

Implementation

- **Here's the Pseudo-code for the Forward-Path of U-Net architecture:**

Given Notations:

$CONV(\cdot)$:- $2D$ convolution with 3 filter size including zero padding.

$OUTCONV(\cdot)$:- $2D$ convolution with 1 filter size including zero padding.

$CONV^T(\cdot)$:- $2D$ deconvolution with 3 filter size including zero padding.

$\mathcal{R}(\cdot)$:- ReLU.

$\mathcal{B}(\cdot)$:- Batch-Norm.

$\mathcal{M}(\cdot)$:- Max-Pooling (2×2).

$\mathcal{CAT}(\cdot)$:- Concatenation along channel direction.

$\mathcal{DBLCONV}(\cdot) = \mathcal{R}(\mathcal{B}(CONV(\mathcal{R}(\mathcal{B}(CONV(\cdot))))))$

$\mathcal{DOWN}(\cdot) = \mathcal{M}(\mathcal{DBLCONV}(\cdot))$

$\mathcal{UP}(\cdot) = \mathcal{DBLCONV}(\mathcal{CAT}(CONV^T(\cdot)), \cdot)$

Contd..

Input:

$X_i \in \mathbb{R}^{1\times1\times\mathrm{n}\times\mathrm{n}}$:- input FBP image

$X_d \in \mathbb{R}^{1\times1\times n_d\times n_d}$:- input image for next operation.

Output:

$X_o \in \mathbb{R}^{1\times1\times\mathrm{n}\times\mathrm{n}}$:- Output of the architecture

Forward:

Phase-I : Encoder Path

$X_1 = \mathcal{DBLCONV}(X_i)$

for j in range(1,5)

$X_{j+1} = \mathcal{DOWN}(X_j)$

Phase-II : Decoder Path

for j in range(5,8)

$X_{j+1} = \mathcal{UP}(X_j)$

$X_o = OUTCONV(X_9)$

Pseudo-code for the Forward-Path of Residual U-Net architecture:

Given Notations:

$CONV(\cdot)$:- $2D$ convolution with 3 filter size including zero padding.

$CONV^T(\cdot)$:- $2D$ deconvolution with 3 filter size including zero padding.

$\mathcal{R}(\cdot)$:- ReLU.

$\mathcal{B}(\cdot)$:- Batch-Norm.

$\mathcal{M}(\cdot)$:- Max-Pooling (2×2).

$\mathcal{CAT}(\cdot)$:- Concatenation along channel direction.

$INPUTLAYER(\cdot) = CONV(\mathcal{R}(\mathcal{B}(CONV(\cdot))))$

$INPUTSKIP(\cdot) = CONV(\cdot)$

$RESCONV(\cdot) = CONV(\mathcal{R}(\mathcal{B}(CONV(\mathcal{R}(\mathcal{B})(\cdot)))))$

$RESSKIP(\cdot) = \mathcal{B}(CONV(\cdot))$

$OUTPUTLAYER(\cdot) = \sigma(CONV(\cdot))$

Contd..

Input:

$X_i \in \mathbb{R}^{1\times1\times\mathrm{n}\times\mathrm{n}}$:- input FBP image

$X_d \in \mathbb{R}^{1\times1\times n_d\times n_d}$:- input image for next operation.

Output:

$X_o \in \mathbb{R}^{1\times1\times\mathrm{n}\times\mathrm{n}}$:- Output of the architecture

Forward: *Phase-I : Encoder Path*

$$X_1 = INPUTLAYER(X_i) + INPUTSKIP(X_i)$$
$$X_2 = RESCONV(X_1) + RESSKIP(X_1)$$
$$X_3 = RESCONV(X_2) + RESSKIP(X_2)$$
$$X_4 = CONV^T(RESCONV(X_3) + RESSKIP(X_3))$$
$$X_5 = \mathcal{CAT}(X_4, X_3)$$

Phase-II : Decoder Path

$$X_6 = CONV^T(RESCONV(X_5) + RESSKIP(X_5))$$
$$X_7 = \mathcal{CAT}(X_6, X_2)$$
$$X_8 = CONV^T(RESCONV(X_7) + RESSKIP(X_7))$$
$$X_9 = \mathcal{CAT}(X_8, X_1)$$
$$X_o = OUTPUTLAYER(RESCONV(X_9) + RESSKIP(X_9))$$

Training

- The training set consists of 4000 tuples of ground-truth phantom images and their corresponding 128-view FBP reconstructions.
- Our one training procedure includes training 400 randomly chosen images and observing their loss values. We have repeated this 20 times and considered average performance.
- Each training instance has 390 images split up into training and validation images.
- The models are trained by minimizing the Root Mean Square Error (RMSE) between the residual labels and predictions.
- The optimization algorithm is Stochastic Gradient Descent Algorithm (SGD) with momentum, learning rate decay is used.
- Each model training has been executed for 40 Epochs on IITH maths server.
- One training instance of U-net took 39 minutes and total it took 13 hrs.
- One training instance of Residual-net took 50 minutes and total it took 16 hrs.

Results

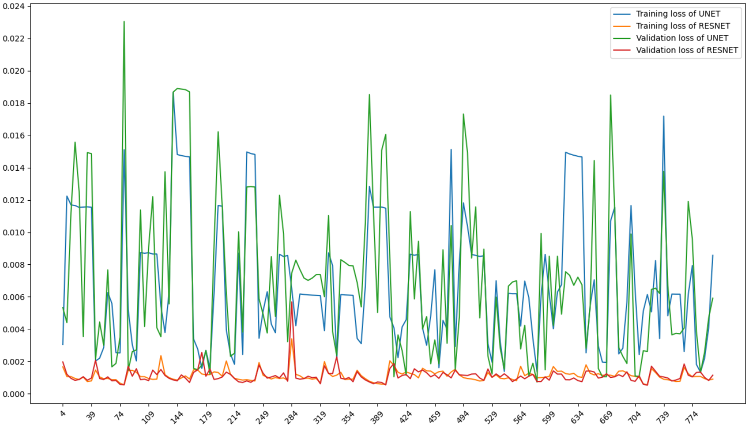
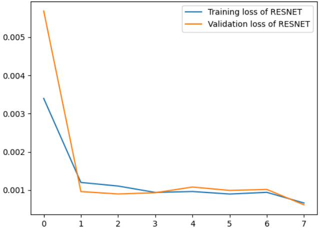


Figure: Graph B/w Loss and Number of epochs

Contd..



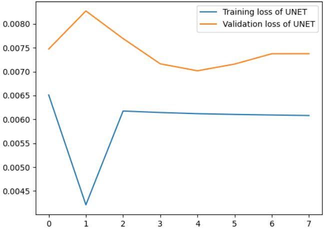


Figure: Graph B/w Loss and Number of epochs of one Residual Unet and Unet model

- To apply the reconstructed method via TV minimization method, We have built the Radon Matrix R using the Matlab Binary matrix function.

  function: [U, BinLR] = Binary matrix(N,L)

  where N is the image size, L is the parameter used to provide the number of views, BinLR is the struct type and U is the Radon matrix.

- To apply the reconstructed method via TV minimization method, We have built the Radon Matrix R using the Matlab Binary matrix function.

  function: [U, BinLR] = Binary matrix(N,L)

  where N is the image size, L is the parameter used to provide the number of views, BinLR is the struct type and U is the Radon matrix.

- As in our dataset original image is of size 512×512 with 128 views angle.

- To apply the reconstructed method via TV minimization method, We have built the Radon Matrix R using the Matlab Binary matrix function.

  function: [U, BinLR] = Binary matrix(N,L)

  where N is the image size, L is the parameter used to provide the number of views, BinLR is the struct type and U is the Radon matrix.
- As in our dataset original image is of size 512×512 with 128 views angle.
- Resultant size of U will be $65,536 \times 262,144 \Rightarrow$ **Error: Out of Memory.**

- To apply the reconstructed method via TV minimization method, We have built the Radon Matrix R using the Matlab Binary matrix function.

  function: [U, BinLR] = Binary matrix(N,L)

  where N is the image size, L is the parameter used to provide the number of views, BinLR is the struct type and U is the Radon matrix.

- As in our dataset original image is of size 512×512 with 128 views angle.
- Resultant size of U will be $65,536 \times 262,144 \Rightarrow$ **Error: Out of Memory.**
- In order to avoid this error we have resized our dataset images to 128×128 size with 32 views angle.

Results

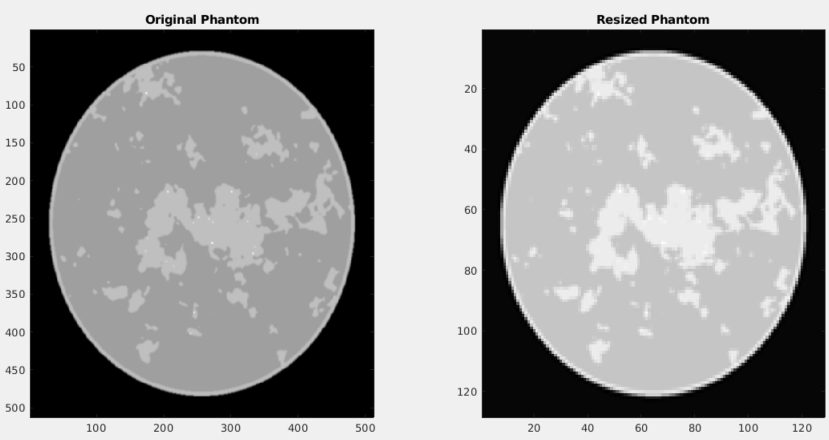
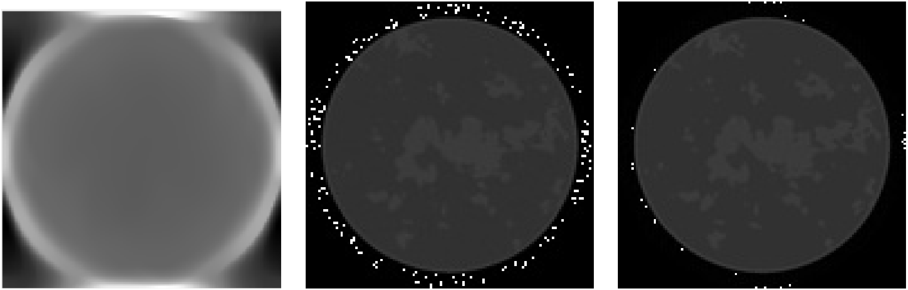


Figure: Resized Image from $512 \times 512 \Rightarrow 128 \times 128$

contd..

TV Norm | **U-Net** | **Res-Net**



RMSE: 0.1502 | RMSE: 0.5197 | RMSE: 1.0397

Figure: Reconstructed Images based on TV -Norm,U-net and Res-net architectures and their corresponding RMSE values.

Conclusions

- For solving CT Inverse problems end to end Deep Networks lack theoretical formulations in comparison to the classical back projection based method.

Conclusions

- For solving CT Inverse problems end to end Deep Networks lack theoretical formulations in comparison to the classical back projection based method.
- We have used ResUNet and UNet end-to-end deep network and TV minimization classical method for CT image reconstruction, though we have obtained RMSE value of TV norm method less than comparison to the other two, but the reconstructed image quality of UNet and ResUNet are much better than TV norm.

Road Ahead...

- We propose to use ResNet on complete database and compare our results with those available in the literature.

References

- **Sidky, E., Lorente, I., Brankov, J. G., and Pan, X.** Do CNNs solve the CT inverse problem?-IEEE Trans. Biomed.Eng 2021
- **Kharfi, F.** Mathematics and Physics of Computed Tomography (CT): Demonstrations and Practical Examples-2013.
- **O. Ronneberger, P. Fischer, and T. Brox,** U-Net: Convolutional networks for biomedical image segmentation,, 2015,pp. 234–241.
- **K. He, X. Zhang, S. Ren, and J. Sun**,Deep residual learning for image recognition," Jun. 2016, pp. 770-778.
- **G. Bal ,** \Introduction to inverse problems," 2012,URL:https://statistics.uchicago.edu/ guillaume-bal/PAPERS/IntroductionInverseProblems.pdf.
- **K. H. Jin, M. T. McCann, E. Froustey, and M. Unser**,Deep convolutional neural network for inverse problems in imaging,2017.

Thank You