A ROBUST DEEP LEARNING MODEL FOR DIRECT CT IMAGE RECONSTRUCTION

Pham Cong Thang*, Huynh Duc Anh Bao, Dinh Minh Toan, Nguyen Thanh Than, Le Minh Tri

The University of Danang - University of Science and Technology, Vietnam

*Corresponding author: pcthang@dut.udn.vn

(Received: May 07, 2025; Revised: June 18, 2025; Accepted: June 21, 2025)

DOI: 10.31130/ud-jst.2025.23(10B).628E

Abstract - Computed Tomography (CT) image reconstruction is an important research direction in the medical field. In addition to traditional reconstruction methods using iterative algorithms such as Maximum Likelihood Expectation Maximization (MLEM) or back-projection and forward-projection methods to reconstruct images from sinograms, deep learning models have emerged as a promising solution. Deep learning architectures, such as convolutional neural network (CNN), U-Net, and generative adversarial network (GAN), allow to significantly improve reconstruction accuracy from sinogram data directly. In this study, we focuses on improving deep learning models to directly reconstruct CT images from sinogram data while also applying traditional reconstruction methods to enhance the quality of the reconstructed images.

Key words - Direct image reconstruction; Computed tomography (CT); Generative adversarial network (GAN); Wasserstein generative adversarial network (WGAN); Maximum Likelihood Expectation Maximization (MLEM)

1. Introduction

Computed tomography (CT) plays a crucial role in detecting molecular activity within tissues and is widely used for tumor detection and various other medical diagnoses [1]. However, conventional CT imaging methods, while effective, present a significant risk of radiation exposure to patients, as the gamma rays used can ionize organic molecules and cause potential harm. The normal radiation dose, although effective in producing diagnostic-quality images, poses a latent risk of radiation-induced damage [2]. In response to these challenges, various algorithms have been proposed to enhance the quality of CT images and improve diagnostic capabilities. Some widely used algorithms include MLEM [3], Markov random field (MRF) [4], Total variation (TV), and other different variants [5-7], which are commonly applied to improve CT image quality from sinogram data.

Deep-learning applications have recently made great progress in medical imaging, especially in noise reduction, image segmentation, and reconstruction. Notable applications include CNNs, U-Net, GAN for CT noise reduction [8-10]. Complex CNN models have shown promising results in these areas. However, these methods typically serve as post-processing tools rather than directly applying deep-learning models to the CT reconstruction process. DeepPET [11] and DPIR-Net [12] are two new ways to reconstruct PET images that are worth mentioning. Häggström came up with DeepPET, which uses a deep encoder-and-decoder network to accurately solve the reconstruction inverse problem in PET images. This improves both the quality of the images and the speed of the reconstruction. Zhanli Hu et al. came up with DPIR-Net, which is based on the Wasserstein generative adversarial network (WGAN). This method uses GANs to make direct PET image reconstruction better and faster.

In addition, there are other methods that combine deep learning models with reconstruction techniques, such as FBP or IR, for various purposes [13]. This research aims to improve CT image reconstruction by leveraging deep learning models, such as DeepPET and DPIR-Net, for direct image reconstruction from sinograms while also integrating traditional reconstruction methods like MLEM-IR [14-18] to enhance the quality of the resulting images. The main experiments in this study focus on two input approaches: one input (sinogram) and two inputs (sinogram and MLEM-IR output from one iteration). The results indicate that enhancing the input by adding more information significantly improves the quality of the CT images reconstructed from sinogram data.

2. Related works

2.1. The Maximum Likelihood Expectation Maximization – Iteration Reconstruction (MLEM-IR)

The MLEM algorithm is a way to rebuild images in CT that uses an iterative process. The MLEM process updates the image over a series of k iterations. Each iteration improves the image based on the first input, which is usually a sinogram image that has been back-projected to fit the size of the final reconstructed image. The MLEM update blocks are used to update the image several times in this process. The method is widely used because it works well with noisy and incomplete data, which makes it a basic technique for image reconstruction tasks. Some variations, like the MLEM-IR method, add more steps or regularization techniques to the reconstruction process to make the images even better and speed up the process [14–18].

2.2. Convolutional neural network (CNN)

CNNs are the best choice for CT image reconstruction because they can naturally learn deep, layered features from the data they get. Convolutional layers are known as special filters to find features while pooling layers make the image smaller by down-sampling it and activation functions make sure the model can find non-linear relationships. The network first processes the sinogram (forward propagation) to get useful data. Then, it fine-tunes its settings (backward propagation) to improve the quality of the image and reduce

errors. You can get better results with fully connected layers, but CNNs usually work well without them. They keep getting better at making CT images by going through training cycles over and over.

2.3. Generative adversarial networks (GANs)

GANs [20] consist of two networks: a generator (G) that creates unreal data and a discriminator (D) that evaluates whether the data is real or fake. The G intends to fool the D, and both networks are trained together. However, GANs can suffer from the vanishing gradient problem, making training difficult.

WGAN [21] solves this problem by using the Earth Mover distance instead of the Jensen-Shannon divergence, which makes training more stable. WGAN also has a critic function that is Lipschitz-constrained. Gulrajani et al. improved WGAN even more by punishing the gradient norm instead of clipping weights. This made it converge faster and produce better samples. The primary distinction between GAN and WGAN lies in their utilization of divergent objective functions for assessing distribution distance.

2.4. DeepPET

The DeepPET network [11] has a few important parts that work together to quickly put together PET images. The first step is to use convolutional layers with kernels of different sizes, like 7x7, 5x5, 3x3, and strides, to slowly pull out and improve features from the input data. Next, ReLU activation functions and batch normalization are also used in this network which make sure that the output of each layer is the same. After the convolutional steps, upsampling layers are used to increase the spatial resolution of the feature maps.

This network's feature extraction process is divided into several steps, each with more feature layers (for example, 32, 64, 128, 256, etc.). The size of the data in space gets smaller with each step. These layers work together to capture both low-level and high-level features which helps reconstruct high-quality outputs from raw data. The architecture of the network keeps important spatial and semantic information during the reconstruction process.

2.5. DPIR-Net

DPIR-Net [12] is a deep learning model that is made to make high-quality images again. The architecture has two parts: a generator (G) and a discriminator (D). The G has three parts: the encoder, the transformation part, and the decoder. It uses convolutional layers with batch normalization and ReLU activations to get features. Then it uses the transformation part to model the input distribution. The D then uses upsampling and convolution layers to put the image back together.

The D looks at the images that were made and decides if they are real or fake. As shown in Figure 7, it uses several convolutional layers with LeakyReLU activations to analyze features at different resolutions over time. The last layer makes a binary decision about whether the image is real or not. Using WGAN to combine these two parts makes training more stable and improves the quality of PET image reconstruction.

3. Proposed method

The proposed method in this study examines two different input approaches for CT image reconstruction from sinogram data. As shown in Figure 1, the single input method uses only the sinogram (sized 127x127) as input to the deep learning model, which outputs the reconstructed CT image (sized 90x90). The second approach, referred to as the double input method, introduces additional information by combining both the sinogram and the output from one iteration of the MLEM (MLEM-IR) method. This additional input, as illustrated in Figure 1, enhances the reconstruction process by feeding both inputs into the deep learning model. A detailed description of the proposed method, the related models, and the experimental setup will be presented in this section, while the results and comparisons between the methods will be discussed in Section 4.

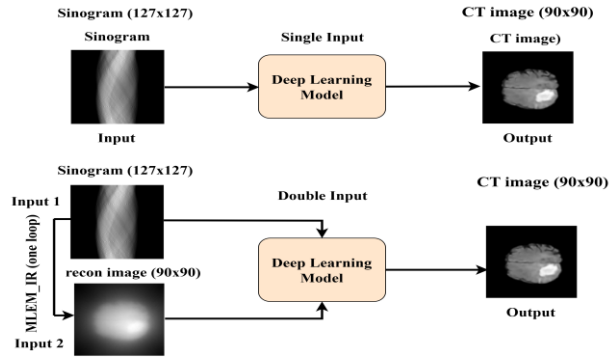

Figure 1. *The overall of proposed methods*

The letters "ic," "oc," "k," "s," and "p" are used in the figures in this study to show the parameters of the convolutional and linear layers. In particular, "ic" means the number of input channels, and "oc" means the number of output channels or filters. The "k" stands for the size of the kernel, the "s" stands for the stride, which is the size of the filter's step, and the "p" stands for padding, which adds extra pixels to the input to keep the spatial dimensions or allow edge processing.

3.1. Proposed Direct CT Reconstruction Models

The proposed Double-Input ED-CNN GAN (Encoder-Decoder Convolutional Neural Network GAN) network includes G and D as two main parts, as shown in Figure 2. The input to the G is a combination of the sinogram image and the reconstructed image from MLEM-IR (one iteration) [14], while the output of the G is the reconstructed image. This output is then passed to the D to differentiate between the generated image and the ground-truth image, allowing the D to guide the G in producing more accurate reconstructions. The parameters of both the G and D are updated based on the feedback from the D.

In addition to the proposed model, we also utilized the WGAN method, as previously mentioned in [12], for comparison to identify the more effective approach. The WGAN follows the same steps as the GAN but does not include the sigmoid layer in the discriminator. The architecture of the Double-Input ED-CNN WGAN is presented in Figure 3. Furthermore, two versions of the single-input ED-CNN GAN and ED-CNN WGAN were also built for comparison, as shown in Figure 4.

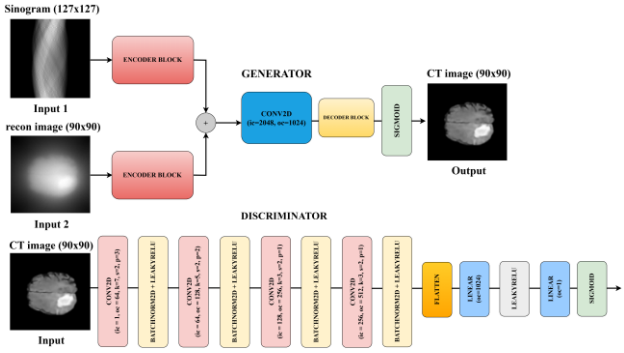

Figure 2. *Double-Input ED-CNN GAN architechture*

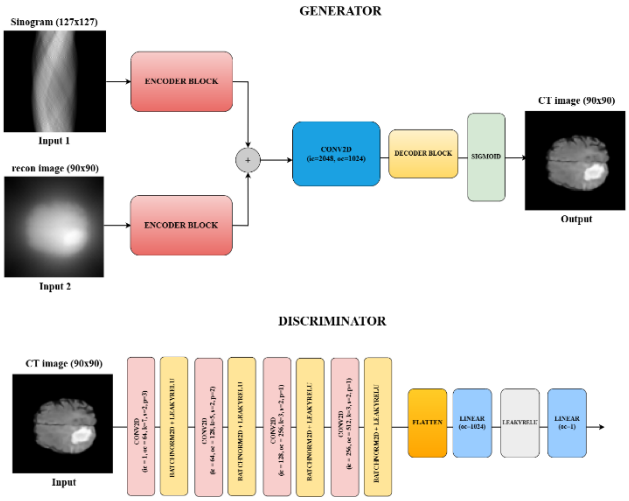

Figure 3. *Double-Input ED-CNN WGAN architechture.*

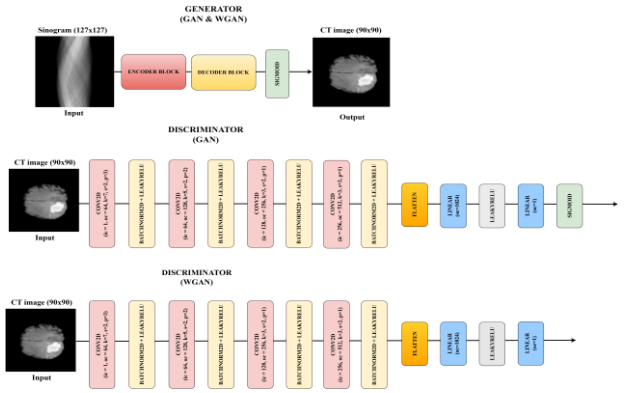

Figure 4. *Single Input ED-CNN GAN and Single Input ED-CNN WGAN Architectures*

The ED-CNN is a deep learning model that is made to help with CT image reconstruction tasks. It has an encoder-decoder structure. The encoder block takes features from the input sinogram image, and the decoder block makes the output image again. The final output then goes through a sigmoid activation function to make the image look like it did before.

The figures show that the ED-CNN model has two ways to input data. The single input model (Figure 5) only takes the sinogram data (size 127x127) as input, processes it through the encoder and decoder blocks, and then outputs the reconstructed image. The double input model in Figure 6, on the other hand, has two inputs: (1) the original sinogram (127x127) and (2) the reconstructed image (90x90) from the MLEM-IR process. A Conv2D layer combines these two inputs after they have been processed through separate encoder blocks. Then, the decoder block creates the final output. The double-input model lets the networks learn more about the MLEM-IR output which improves the quality of the reconstructed image by giving them more information to work with during training.

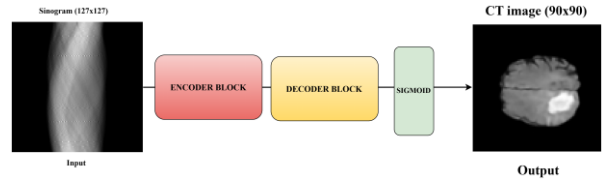

Figure 5. *Single-Input ED-CNN architechture*

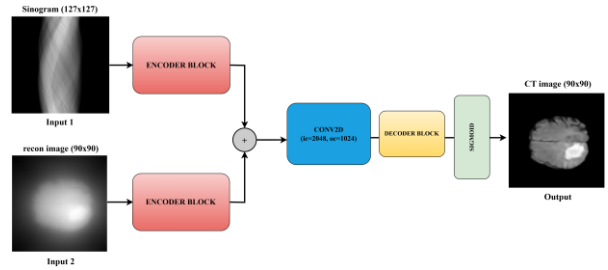

Figure 6. *Double-Input ED-CNN architechture*

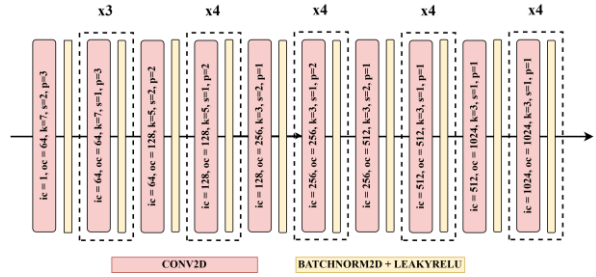

Figure 7. *Structure of Encoder block*

The proposed architecture's encoder block is based on the VGG-16 model [22], which is known for its deep, sequential convolutional layers with small receptive fields and non-linear activation functions. This architecture uses several convolutional layers with different kernel sizes to get different features from the input in this specific encoder block (Figure 7). The model uses several layers of convolutional operations with different kernel sizes (7x7, 5x5, and 3x3) to pick up features at different scales. The layers add more filters one at a time, starting with 64 and going up to 1024 in the deeper layers. The model can learn complex and hierarchical feature representations because some of the layers are repeated several times (for example, blocks of 128, 256, and 512 filters). After each convolutional layer, batch normalization and LeakyReLU activations are used to make training more stable and improve feature learning. By combining these layers, the network can learn both low- and high-level abstractions, which makes it good for tasks like image reconstruction. This is because the spatial resolution is lower and the depth of the feature maps is higher.

The D block puts the image back together using the features that the encoder found. Like the encoder block, it uses several layers of convolutional operations. However, in this case, transposed convolutions (ConvTranspose2D) are used

to upsample the feature maps and bring them back to the size of the original image. BatchNormalization and LeakyReLU activations come after the transposed convolution layers. These help keep training stable and add non-linearity. This decoder block (Figure 8) has 1024 filters, each with a 3x3 kernel, a stride of 2, and a padding of 1. This is done three times. As the network goes on, the number of filters goes down slowly, first to 512, then to 256, and finally to 128, each with a 3x3 kernel and a stride of 1. The last convolutional layer has 64 filters, each with a 3x3 kernel and a stride of 1. The final step of the decoder is to upsample the feature map to a 90x90 resolution. Then, there is a final convolution layer with one filter that makes the output image.

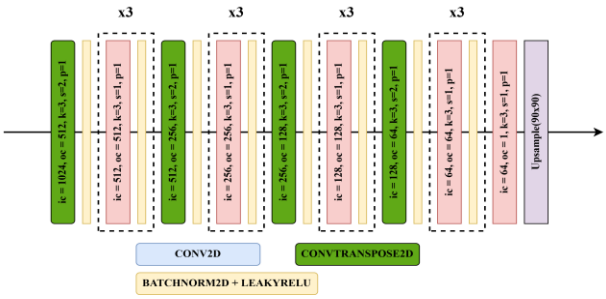

Figure 8. *Structure of Decoder block*

The encoder block in the ED-CNN is charged with feature extraction and is utilized in both the single-input and double-input CNN models that incorporate fully connected layers. It extracts features from the input data, which are subsequently transmitted to the fully connected layers for final image reconstruction.

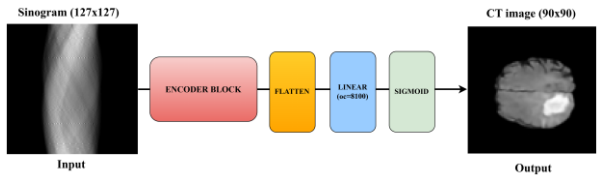

Figure 9. *Single-Input CNN architechture*

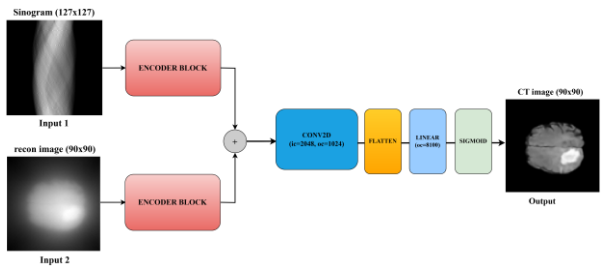

Figure 10. *Double-Input CNN architechture*

In both the single-input (Figure 9) and double-input (Figure 10) models, the linear layer is crucial for converting the extracted characteristics into the final reconstructed image. The single-input model processes the 127x127 input sinogram via the encoder to extract features. The output is then processed through a linear layer of 8100 channels, creating a flattened vector that is reconfigured into a 90x90 CT image.

In the double-input model, both the sinogram and an additional reconstructed image are processed by independent encoders. The resultant features are merged, processed through a Conv2D layer, and then traversed through a linear layer with 8100 output channels. This output is reformatted into a 90x90 CT image. The primary distinction between the two models is that the double-input model takes supplementary data from the intermediate reconstruction, therefore enhancing the reconstruction process.

3.2. Loss Function

In the proposed model, we define the loss functions for both the G and the D, which are key components in the training process of GANs and WGANs. The generator's objective is to create high-quality images that imitate the ground-truth images as closely as possible, while the discriminator's task is to distinguish between real images and the images generated by the generator.

The generator loss (L_G) in the proposed model combines two components: adversarial loss ($L_{G_{adv}}$) and reconstruction loss (L_{recon}), which are weighted by the parameter λ_{mse} - This parameter controls the offset between the adversarial loss and the reconstruction loss. This loss is computed as:

$$L_G = (12 - \lambda_{mse}) \cdot L_{G_{adv}} + \lambda_{mse} \cdot L_{recon} \quad (1)$$

Adversarial loss ($L_{G_{adv}}$): This component encourages the generator to produce images that are as realistic as possible [20]. It is defined as:

$$L_{G_{adv}} = -E_{u \sim P_g}[\log D(G(u))] \quad (2)$$

where P_r and P_g represent the real data and generative data distributions, respectively, u represents the sinogram data and $D(G(u))$ is the discriminator's output for the generated image $G(u)$. The generator aims to maximize the discriminator's probability of classifying the generated image as real.

Reconstruction loss (L_{recon}) which ensures the generator not only creates realistic images but also minimizes pixel-level differences from the ground truth, is typically represented by mean squared error (MSE) [26]:

$$L_{recon} = \frac{1}{n}\sum_{i=1}^{n}(t_i - g_i)^2 \quad (3)$$

where t is the true CT image, g is the generated image, and n is the number of image pixels.

Regarding the loss for the discriminator, we use the original discriminator loss (L_D) formula for both the GAN and WGAN models [20, 21].

For the GAN model, the discriminator loss is formulated as:

$$L_D = E_{x \sim P_r}[\log D(x)] + E_{u \sim P_g}\left[\log\left(1 - D(G(u))\right)\right] \quad (4)$$

where x represents the real CT image.

For the WGAN model, the discriminator loss, which uses the Earth Mover's (EM) distance with a gradient penalty, is expressed as:

$$L_D = -E_{x \sim P_r}[D(x)] + E_{u \sim P_g}[D(G(u))] \quad (5)$$

4. Experiments

The BraTs20 dataset [23] is used in this study to obtain CT images for image reconstruction tasks. The dataset

contains a total of 3975 training images and 100 test images, with the images resized to 127x127 pixels for uniformity. During training, the data is split into two sets: a training set and a validation one. The training one is used to train the model, and the validation one is used to check how well the model is doing and avoid overfitting. The results in the tables are derived from the mean values of the 100 test photos. From the 100 test photos, 10 images are randomly chosen to exhibit the rebuilt images in Figure 11.

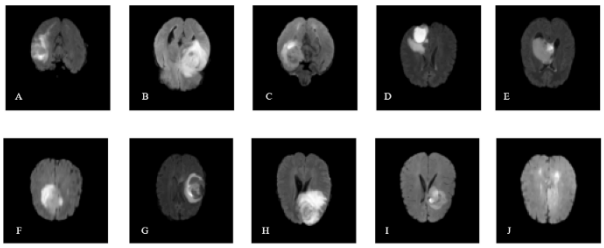

Figure 11. *Testing dataset*

The study employs three main evaluation metrics: Peak Signal-to-Noise Ratio (PSNR), relative Root Mean Squared Error (rRMSE), and Structural Similarity Index (SSIM) [24].

We trained the models in PyTorch [25] with a learning rate of 0.0001 and a batch size of 64. All other settings were left at their default values. The training lasted for 500 epochs, and every 100 epochs, the learning rate was cut in half so that the model could be slowly fine-tuned. We also use early stopping when training. If the loss value didn't get better for 30 epochs in a row, the training would stop on its own and save the model's weights. Google Colab with T4 GPU setting was used to train our models.

4.1. Single-Input ED-CNN GAN model with varying lambda-mse values

Table 1. *Performance comparison of λ_{mse} in the Single-Input ED-CNN GAN model on the testing dataset*

Config	PSNR↑	rRMSE↓	SSIM↑
lambda-mse = 0.01	20.230	1.047	0.793
lambda-mse = 0.1	25.448	0.554	0.849
lambda-mse = 0.5	29.526	0.344	0.911
lambda-mse = 0.9	**31.797**	**0.266**	**0.931**
lambda-mse = 0.99	29.714	0.343	0.914

Table 1 presents the performance comparison of the Single-Input ED-CNN GAN model with varying lambda-mse (λ_{mse}) values, evaluated on the testing dataset. The table shows that the model works much better when lambda-mse is between 0.01 and 0.9. When you raise λ_mse above 0.9, there is a small drop. In summary, the lambda-mse value of 0.9 is determined to be optimal for the ED-CNN GAN models, and this value is uniformly applied across all GAN and WGAN models for comparative analysis in this study.

4.2. Single-Input models and Double-Input models

Table 2 shows the performance of different deep learning models did at reconstructing CT images when tested on a dataset. The models are grouped based on the type of input (Single or Double) and have different architectures, such as CNN (with an Encoder block and FC layers), ED-CNN (with an Encoder block and Decoder block), ED-CNN WGAN, and ED-CNN GAN. Three metrics used to judge the models are PSNR, rRMSE, and SSIM. Higher PSNR and SSIM values and lower rRMSE values mean that the model is doing better.

Table 2. *Performance comparison of proposed models on the testing dataset*

Input	Model	PSNR↑	rRMSE↓	SSIM↑
Single	CNN	28.623	0.307	0.626
	ED-CNN	33.135	0.226	0.945
	ED-CNN + WGAN	31.494	0.278	0.93
	ED-CNN + GAN	32.930	0.23	0.942
Double	CNN	29.826	0.268	0.705
	ED-CNN	33.913	0.179	0.955
	ED-CNN + WGAN	31.601	0.228	0.941
	ED-CNN + GAN	33.991	0.177	0.955

As The ED-CNN + GAN model is better than the ED-CNN + WGAN model in both Single-Input and Double-Input configurations, as shown in Table 2. For Single-Input, ED-CNN + GAN gets a PSNR of 32.930, a rRMSE of 0.230, and an SSIM of 0.942. For Single-Input, ED-CNN + WGAN gets a PSNR of 31.494, a rRMSE of 0.278, and an SSIM of 0.930. The ED-CNN + GAN model again performs better in Double-Input, with a PSNR of 33.991, a rRMSE of 0.177, and an SSIM of 0.955. The ED-CNN + WGAN model has a PSNR of 31.601, a rRMSE of 0.228, and an SSIM of 0.941. These findings indicate that the GAN architecture improves image quality, especially in the Double-Input setup.

Figure 12 shows the results of the best reconstruction model, which used the Double-Input ED-CNN GAN architecture to create the reconstructed image.

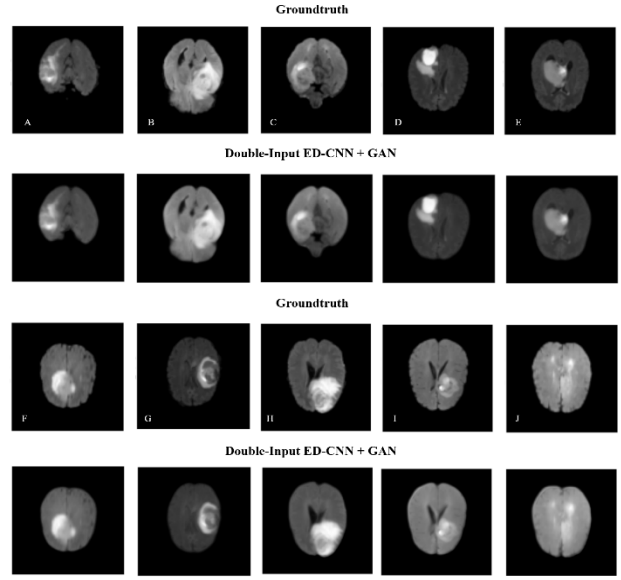

Figure 12. *Reconstructed CT images from the best-performing Double-Input ED-CNN GAN model*

5. Conclusion

In this study, we proposed many deep learning models for CT image reconstruction, emphasizing on the improvement of reconstruction quality by incorporating

advanced architectures like ED-CNN and GAN. We focused on how different setups, like models with one or two inputs, affected the performance of the GAN-related model. Presented results showed that the ED-CNN GAN model performed better than other models when it came to different evaluation metrics (PSNR, rRMSE, and SSIM). This shows that adversarial training is a good way to improve image quality.

A central focus of our research was the examination of the double input model, wherein the generator received both a sinogram image and a reconstructed image from MLEM-IR (one iteration) as input. This method was shown to effectively improve the accuracy of the process of reconstructing. The double input model let the network use both the raw sinogram data and the extra information from the MLEM-IR reconstruction. This made the output more accurate and easier to see than the single input model. The results showed that using more than one source of input for training was helpful due to it made the model better at producing high-quality reconstructions.

This study lays the groundwork for future research in CT image reconstruction, potentially enhancing deep learning models by incorporating additional features, sophisticated loss functions, and diverse network architectures. Future research will examine three domains: (1) utilizing the intermediate outputs of MLEM-IR as inputs, (2) developing hybrid systems that integrate deep learning with conventional reconstruction techniques, and (3) broadening the application of these advanced methods to diverse imaging modalities beyond standard CT.

REFERENCES

[1] C. W. Schmidt, "CT scans: balancing health risks and medical benefits", *Environmental Health Perspectives*, vol. 120, no. 3, pp. A118–A121, 2012. https://doi.org/10.1289/ehp.120-a118

[2] M. G. Hunink and G. S. Gazelle, "CT screening: a trade-off of risks, benefits, and costs", *The Journal of Clinical Investigation*, vol. 111, no. 11, pp. 1612-1619, 2003. https://doi.org/10.1172/JCI18842

[3] L. Shepp and Y. Vardi, "Maximum likelihood reconstruction for emission tomography", *IEEE Transactions on Medical Imaging*, vol. 1, no. 2, pp. 113–122, 1982. https://doi.org/10.1109/TMI.1982.4307558

[4] H. Shangguan, Q. Zhang, Y. Liu, X. Cui, Y. Bai, and Z. Gui, "Low-dose CT statistical iterative reconstruction via modified MRF regularization", *Computer Methods and Programs in Biomedicine*, vol. 123, pp. 129–141, 2016. https://doi.org/10.1016/j.cmpb.2015.10.004

[5] Y. Liu, J. Ma, Y. Fan, and Z. Liang, "Adaptive-weighted total variation minimization for sparse data toward low-dose X-ray computed tomography image reconstruction", *Physics in Medicine and Biology*, vol. 57, no. 23, pp. 7923–7956, 2012. https://doi.org/10.1088/0031-9155/57/23/7923

[6] Z. Tian, X. Jia, K. Yuan, T. Pan, and S. B. Jiang, "Low-dose CT reconstruction via edge-preserving total variation regularization", *Physics in Medicine and Biology*, vol. 56, no. 18, pp. 5949–5967, 2011. https://doi.org/10.1088/0031-9155/56/18/011

[7] E. Y. Sidky and X. Pan, "Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization", *Physics in Medicine and Biology*, vol. 53, no. 17, pp. 4777–4807, 2008. https://doi.org/10.1088/0031-9155/53/17/021

[8] Y. Wang *et al*., "3D conditional generative adversarial networks for high-quality PET image estimation at low dose", *NeuroImage*, vol. 174, pp. 550–562, 2018. https://doi.org/10.1016/j.neuroimage.2018.03.045

[9] H. Shan *et al*., "Can deep learning outperform modern commercial CT image reconstruction methods?" *arXiv:1811.03691*, Nov. 2018. [Online]. Available: https://arxiv.org/abs/1811.03691 [Accessed June 18, 2025].

[10] Y. Han and J. C. Ye, "Framing U-Net via deep convolutional framelets: Application to sparse-view CT", *IEEE Transactions on Medical Imaging*, vol. 37, no. 6, pp. 1418-1429, 2018. https://doi.org/10.1109/TMI.2018.2823768

[11] I. Häggström *et al*., "DeepPET: A deep encoder–decoder network for directly solving the PET image reconstruction inverse problem", *Medical Image Analysis*, vol. 54, pp. 253-262, 2019. https://doi.org/10.1016/j.media.2019.03.013

[12] Z. Hu et al., "DPIR-Net: Direct PET image reconstruction based on the Wasserstein generative adversarial network", *IEEE Transactions on Radiation and Plasma Medical Sciences*, vol. 5, no. 1, pp. 35-43, 2021. https://doi.org/10.1109/TRPMS.2020.2995717

[13] L. R. Koetzier *et al*., "Deep learning image reconstruction for CT: technical principles and clinical prospects", *Radiology*, vol. 306, no. 3, e221257, 2023. https://doi.org/10.1148/radiol.221257

[14] C. T. Pham *et al.,* "CT image reconstruction: integrating iterative methods with ML-EM algorithm and deep learning models", *Cybernetics and Physics*, vol. 13, no. 2, pp. 130–141, 2024. https://doi.org/10.35470/2226-4116-2024-13-2-130-141

[15] A. J. Reader, "Self-supervised and supervised deep learning for PET image reconstruction", in *Proc. AIP Conf.*, vol. 3061, no. 1, p. 030003, AIP Publishing, 2024. https://doi.org/10.1063/5.0203321

[16] H. Fan, H. Zhu, X. Zhao, J. Zhang, D. Wu, and Q. Han, "Ultrasonic image reconstruction based on maximum likelihood expectation maximization for concrete structural information", *Computers & Electrical Engineering*, vol. 62, pp. 293–301, 2017. https://doi.org/10.1016/j.compeleceng.2017.02.014

[17] H. Yang *et al*., "Transmission reconstruction algorithm by combining maximum-likelihood expectation maximization and a convolutional neural network for radioactive drum characterization", *Applied Radiation and Isotopes*, vol. 184, pp. 1–12, 2022. https://doi.org/10.1016/j.apradiso.2022.110172

[18] M. J. Rodriguez-Alvarez *et al.,* "Expectation maximization (EM) algorithms using polar symmetries for computed tomography (CT) image reconstruction", *Computers in Biology and Medicine*, vol. 43, no. 8, pp. 1053–1061, 2013. https://doi.org/10.1016/j.compbiomed.2013.04.015

[19] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, and F.-Y. Wang, "Generative adversarial networks: introduction and outlook," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 4, pp. 588–598, Oct. 2017. https://doi.org/10.1109/JAS.2017.7510583

[20] I. Goodfellow *et al.,* "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process*. Syst., vol. 27, Montreal, Quebec, Canada, 2014, pp. 2672–2680, https://doi.org/10.48550/arXiv.1406.2661

[21] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *arXiv preprint arXiv:1701.07875*, 2017. https://doi.org/10.48550/arXiv.1701.07875

[22] S. Tammina, "Transfer learning using VGG-16 with deep convolutional neural network for classifying images", *International Journal of Scientific and Research Publications*, vol. 9, no. 10, pp. 143–150, 2019. https://doi.org/10.29322/IJSRP.9.10.2019.p9420

[23] University of Pennsylvania Center for Biomedical Image Computing and Analytics (CBICA), "BraTS 2020 Dataset," *www.med.upenn.edu,* [Online]. Available: https://www.med.upenn.edu/cbica/brats2020/data.html [Accessed January 20, 2025].

[24] Z. Wang and A. C. Bovik, "Modern Image Quality Assessment", *Synthesis Lectures on Image, Video, and Multimedia Processing. San Rafael*, CA, USA: Morgan & Claypool, 2006. [Online]. Available: https://doi.org/10.1007/978-3-031-02238-8

[25] PyTorch, "PyTorch: An Open Source Machine Learning Framework," *PyTorch*, [Online]. Available: https://pytorch.org/ [Accessed April 25, 2025].

[26] P. J. Bickel and K. A. Doksum, "9.1.5 Mean Squared Error (MSE)," *Probability Course,* [Online]. Available: https://www.probabilitycourse.com/chapter9/9_1_5_mean_squared_error_MSE.php. [Accessed April 25, 2025].