Data-driven Computed Tomography Reconstruction

A report presented

by

Andrea Sainz Bear

Departments

Department of Physics (Cavendish Laboratory)

Degree

MPhil Data Intensive Science

Supervision

Dr Ander Biguri

Word Count:

6900

Abstract

Low-dose computed tomography (LDCT) reconstruction faces a significant challenge due to the increased noise and artifacts introduced by reduced radiation exposure. Traditional analytical methods such as filtered backprojection (FBP) and iterative reconstruction degrade under these conditions, motivating the adoption of deep learning techniques that can learn data-driven priors from large-scale datasets. This project implements and evaluates four deep learning reconstruction models: Deep Filtered Back-Projection (DeepFBP), Deep Back-Projection (DBP), FusionFBP and DeepFusionBP; trained and tested on sinograms simulated from the LoDoPaB-CT dataset, converted to fan-beam geometry to reflect clinical acquisition setups. The evaluation covers normal-dose, low-dose and sparse-view configurations, using standard image quality metrics: peak signal-to-noise ratio (PSNR), structural similarity index (SSIM) and mean squared error (MSE). Among the proposed models, FusionFBP achieves the highest performance in full-view normal-dose and low-dose conditions, while DeepFusionBP outperforms others under sparse-view constraints. Quantitative results confirm that both models consistently outperform traditional reconstruction methods and baseline deep learning architectures across all tested experiments.

Contents

List of Figures

List of Tables

1 Introduction

Computed tomography (CT) is an imaging technique in which an X-ray beam is passed through an object or patient. The source and detectors rapidly rotate around the body, collecting signals that represent the amount of radiation transmitted. These signals are then processed by a computer to generate cross-sectional images [3]. CT scanners are widely used in industrial applications as well as for non-invasive diagnostic procedures in modern medicine.

The internal distribution of the scanned object is obtained by solving an inverse problem. Conventionally, this task has been addressed using methods such as filtered backprojection (FBP) or iterative reconstruction techniques, including SIRT, OS-SART, and OSEM. However, these methods produce optimal results primarily under high-dose (low-noise) conditions [4].

Radiation dose must be considered a potential health risk. It is well-established that repeated exposure to CT imaging increases the likelihood of developing cancer and other health issues, including genetic defects. Although modern scanners are designed to minimize these risks, low-dose CT imaging, when using conventional methods like FBP, often results in excessive noise and artifacts in the reconstructed images, potentially compromising diagnostic accuracy [4, 5].

Another key factor in CT image quality and reconstruction complexity is the underlying acquisition geometry. Different system configurations (CT geometries) influence both the data collected and the reconstruction algorithms required:

1. Parallel-beam geometry: where rays travel in parallel lines through the object, a setup mainly used in theoretical studies and older CT systems.

2. Fan-beam geometry: this employs a single X-ray source emitting a diverging beam, which is detected by a linear array of detectors; this is the most common configuration in 2D clinical CT scanners due to its faster acquisition and realistic design.

3. Cone-beam geometry: this geomtry extends fan-beam one into 3D by using a 2D detector array to capture conical projections, and is widely used in dental CT, C-arm systems, and some modern multi-slice scanners.

Figure 1 illustrates these three geometries.

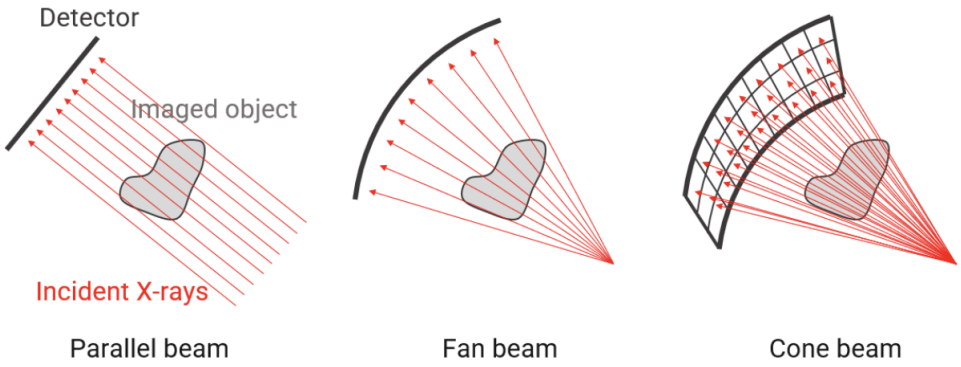


Figure 1: Schematic comparison of CT acquisition geometries: parallel-beam, fan-beam and cone-beam [1].

1.1 State of the Art

In recent years, deep learning has become a relevant approach in CT image reconstruction, especially in the context of low-dose CT (LDCT), where traditional analytical methods like FBP often struggle due to noise and artifacts. Deep learning methods are particularly attractive because of their ability to learn image representations directly from data.

Several deep learning-based reconstruction models have emerged, addressing the LDCT problem from different angles. Some methods operate in the image domain by applying convolutional neural networks (CNNs) to the output of traditional reconstruction algorithms. For instance, RED-CNN [6] and FBPConvNet [7] take the noisy FBP reconstruction and learn to suppress artifacts and enhance image quality. Other models integrate learning into the reconstruction process itself. DeepFBP [5], for example, combines the interpretability of filtered backprojection with learnable components, improving reconstruction performance within an iterative framework. More recent models such as DBP [2] aim to replace large parts of the traditional reconstruction pipeline entirely, by learning to map from backprojected or intermediate representations directly to high-quality images.

These approaches have demonstrated strong results across various datasets, especially under moderate dose reduction scenarios. However, many works still lack reproducibility or fail to compare against strong baselines. This work focus on replicating and evaluating recent reconstruction models under standardized and reproducible conditions.

1.2 Objectives

The primary objective of this work is to reproduce the results presented in [5], hereafter referred to as the original work. However, it is important to note that this study does not use the original dataset proposed by the authors. Instead, the LoDoPaB-CT dataset [4] was employed, which is originally designed for parallel-beam reconstruction tasks. To further evaluate model generalization, this dataset was converted to fan-beam geometry, introducing a different acquisition setting than the one assumed in the original work.

Beyond replication, this study introduces several key extensions: (i) it compares the reconstruction performance of the original DeepFBP architecture for sparse-view CT with a simpler convolutional neural network (CNN) model proposed in [2]; (ii) it incorporates modifications to the original model architecture to improve reconstruction quality; and (iii) it proposes two new architectures, FusionFBP and DeepFusionBP, inspired by the designs of DeepFBP and DBP, aiming to combine their strengths in a unified framework.

1.3 Structure of thesis

The structure of the report is as follows: Chapter 1 introduces the background, motivation and objectives of low-dose CT reconstruction. Chapter 2 presents the architectures of the models considered in this work. Chapter 3 details the experimental framework, including data preparation, preprocessing pipelines, evaluation metrics and training setups for each algorithm. Chapter 4 reports the reconstruction results under different acquisition conditions. Chapter 5 provides a discussion of these findings and Chapter 6 concludes with final remarks and potential directions for future research.

2 Models

This section provides an overview of the reconstruction models considered in this work. Each model combines classical CT principles with deep learning components to various degrees. The architectural designs are based on the original implementations proposed in [2, 5].

2.1 Deep Filtered Back-Projection (DeepFBP)

Filtered Back-Projection (FBP) is a classical method for CT reconstruction that involves filtering a sinogram in the frequency domain and backprojecting it to obtain an image. DeepFBP extends this approach by replacing fixed filters and interpolation with learnable components.

The model first applies a learnable filter in the frequency domain. Two configurations are considered: Filter I, a shared filter across all angles, and Filter II, one filter per angle. Both are initialized using the Ram-Lak filter. After filtering, the sinogram is processed by a nonlinear interpolator implemented as residual blocks and a shallow 1D CNN [5].

The result is then backprojected to the image domain, followed by 2D CNNs that refines the image through convolutional and residual blocks. An overview of the full model is shown in Figure 2.

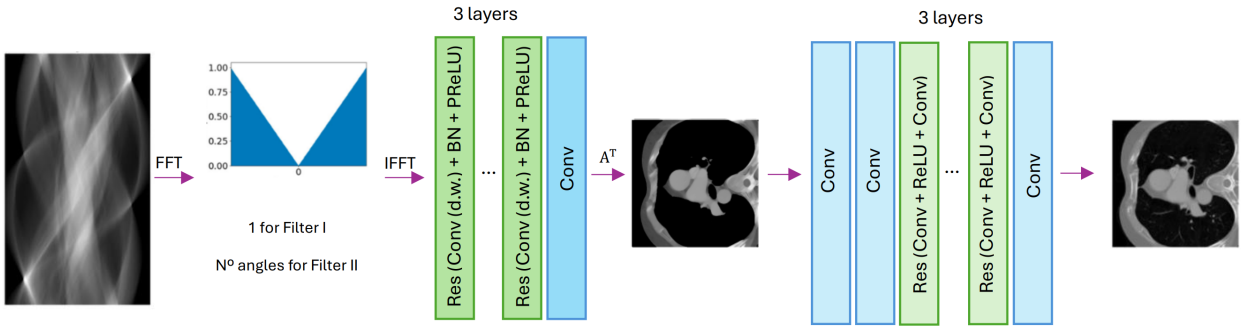


Figure 2: Schematic architecture of the DeepFBP model.

2.2 Deep Back-Projection (DBP)

The Deep Back-Projection (DBP) model is based on the idea that reconstructing an image from individual single-view back-projections is mathematically equivalent to reconstructing from a full sinogram using a global operator. Instead of using a sinogram as input, DBP processes a stack of single-angle back-projections represented as a 3D tensor, where each slice corresponds to a view [2].

The model is a simple convolutional architecture. It begins with a Conv2D layer with ReLU activation that maps the input to 64 channels, followed by 15 convolutional blocks (Conv2D + BatchNorm + ReLU), and ends with a Conv2D layer that reduces the output to a single-channel image. The overall architecture is illustrated in Figure 3.

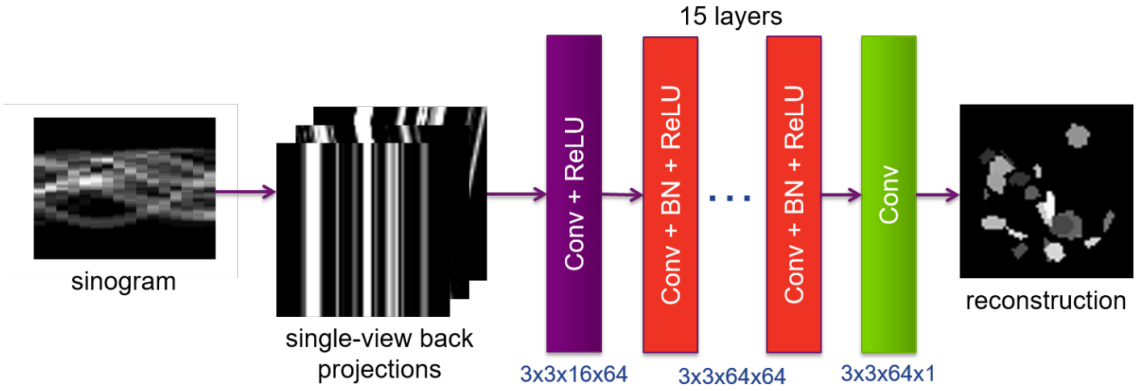


Figure 3: Schematic architecture of the DBP model [2].

3 Experimental Framework

This section describes the data preparation, processing pipeline, and implementation details used in this work. It includes the generation of fan-beam sinograms, dataset construction, noise simulation, and the actual arquitectures used for all models. All procedures are designed to ensure consistency across models and to replicate realistic CT acquisition conditions.

3.1 Data pre-processing

The LoDoPaBCT dataset [4] has been employed for this project. This dataset provides chest CT images for multiple patients acquired at standard radiation dose levels. Although the dataset includes simulated low-dose sinograms based on a parallel-beam geometry, our project focuses on a fan-beam geometry. Therefore, only the ground truth images from the dataset are used. Fan-beam geometry has been selected because it more accurately reflects the configuration used in real-world clinical CT scanners.

The dataset is split into training, validation, and test subsets, comprising 280 patients for training, 28 patients each for validation and testing.

Each patient typically has 128 CT slices. However, for the last patient, while the dataset structure still includes 128 slices, many of them are blank. Upon inspection, only about 30 slices contain valid information. Since the impact on data volume is minimal, this last patient is excluded during data loading.

To generate fan-beam sinograms, a processing pipeline was designed using the `tomosipo` library. This pipeline uses the ground truth images and simulates sinograms under fan-beam geometry by applying a custom projection operator A. The specifications for the imaging setup, based on details from the original paper [4], are summarized below:

Parameter	**Value**
Image resolution	362×362 pixels
Physical domain size	26 cm $\times$ 26 cm
Number of detector bins	513
Number of projection angles	1000
Angle range	0 to π radians

Table 1: CT scan parameters used for simulating fan-beam geometry.

This geometry was used to construct a fan-beam CT model within `tomosipo`, which allowed to simulate normal-dose sinograms. The corresponding implementation can be found in the file `sinogram_simulation.py` located in the `data_analysis` folder of the `job_files` directory of the repository.

Once the sinograms were generated, they were saved alongside their corresponding ground truth images, grouped by patient in a folder named `data_sino`. This structure ensures that the data loading function can access both the original images and their simulated sinograms for subsequent algorithm development and evaluation.

An alternative sinogram simulation pipeline, based on the method described in the original LoDoPaB-CT paper and designed to avoid the inverse crime, was initially attempted. However, due to limitations related to GPU-based implementation stability, this approach was not used in the final experiments. The details of this preliminary attempt are documented in Appendix A.

3.2 Data processing

To handle data loading and preprocessing, a custom PyTorch `Dataset` class named `LoDoPaBDataset` has been implemented. This class is designed to support both training and evaluation for different types of CT reconstruction models.

The dataloader allows for flexible configuration through a number of parameters, summarized in Tables 2 and 3.

Argument	Description
`ground_truth_dir`	Path to the directory that contains the HDF5 files
`vg`	tomosipo volume geometry
`angles`	Full set of CT projection angles
`pg`	tomosipo projection geometry (fan-beam)
`A`	Projection operator defined via tomosipo
`device`	Device for tensor operations (default: `cuda`)
`logger`	Optional logger instance

Table 2: General parameters for `LoDoPaBDataset` class.

Argument	Description
`single_bp`	Whether to compute single-angle backprojections
`n_single_BP`	Number of angles for single backprojections
`sparse_view`	Whether to use sparse-view sinograms
`indices`	Custom angle indices (optional)
`alpha`	Normalization factor for data to ensure groudn truth images are in range [0,1]
`i_0`	Incident photon count (for Poisson noise)
`sigma`	Standard deviation of Gaussian noise
`seed`	Random seed for reproducibility
`max_len`	Limit the number of samples loaded
`debug`	Enable verbose output

Table 3: Noise and mode control parameters for `LoDoPaBDataset`.

This class supports two major input formats corresponding to two model types developed in the project. For the first model the input data consists of individual backprojections for a selected subset of CT angles, and for the second one, the innput data includes full sinograms with added noise (that can also recreate low-dose sinograms) or sparse-view sinograms.

The data returned depends on the selected mode. If single view projections or sparse-view are asked, the dataloader returns the ground truth image, the full clean sinogram (for visualization purposes only), the sparse sinogram, and the stack of single-angle backprojections, for the firs scenario. If non of them are selected, the dataloader returns the ground truth image, the full clean sinogram and the full noisy sinogram.

Realistic noise is simulated in the sinograms using a combination of:

- Poisson noise, which acounts for photon counting errors from low-dose measurements, based on incident photon count I_0.
- Gaussian noise, which simulates imperfections from the detector system.

This noise addition follows the instructions described in [5].

The single-angle backprojections are generated by selecting n projection angles uniformly distributed over the original 1000 angles from $[0, \pi]$, creating a new fan-beam geometry for each angle and finally, performing a single-angle backprojection. These projections are then concatenated into a tensor and used as model input.

Since training on the entire LoDoPaB-CT dataset (around 35,000 slices) is computationally intensive, the dataset class accepts a `max_len` parameter to restrict the number of loaded images. Additionally, the last patient is always excluded to avoid inconsistencies due to empty slices, ensuring robustness for future scalability.

Figure 4 illustrates an example output from the dataset class when using the single backprojection mode. It corresponds to the 128^{th} slice from the first two patients (the first 200 images). The figure shows the clean sinogram (without noise), the sparse sinogram, a single backprojection view, and the sum of all selected single-angle projections.

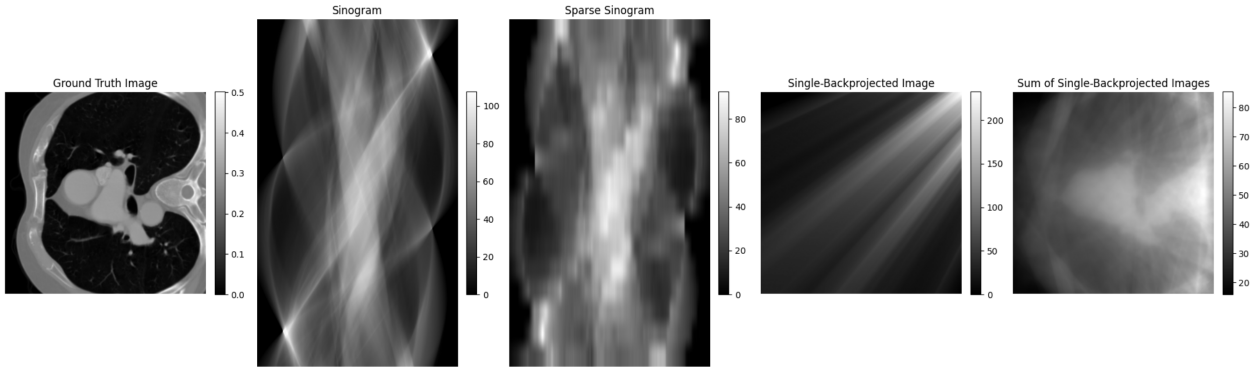


Figure 4: Example output from the dataset class in single backprojection mode.

3.3 Data Analysis

Before applying any normalization or scaling to the ground truth images, a quantitative analysis was conducted to verify the assumption stated in [4] that all images are already normalized to the range $[0, 1]$. This step was important to ensure proper compatibility with machine learning models, which often require data within this normalized range.

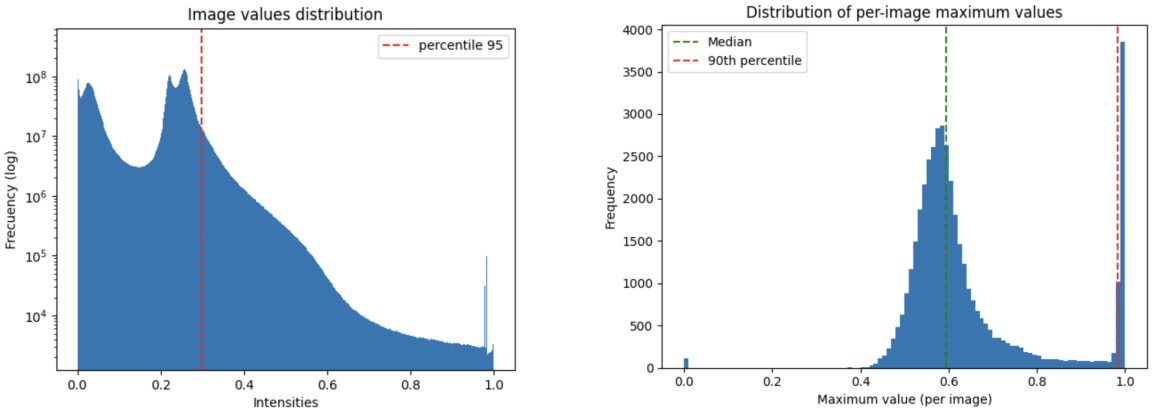
The analysis involved loading all available ground truth slices from the training, validation and test sets and substracting two statistics for each image, all individual pixel values and the maximum pixel value of each image.

These statistics were then visualized using two separate plots: one showing the distribution of all pixel values and another showing the distribution of maximum values per image.

The plots shown in Figures 5 confirm that all pixel values lie within the $[0, 1]$ interval. Furthermore, the distribution of maximum values, Figure 5b, reveals that the majority of images have the maximum values around 0.6. The few images with maximum values close to 0 are associated with the final patient entries in the dataset, which contain empty slices and are excluded during training.

Based on these findings, it was determined that an additional normalization factor (α) is unnecessary. Therefore, the normalization factor was set to 1. This choice ensures that all images remain in their original value range, which is already suitable for model training.

The code for this analysis can be found in the file `image_distribution.py` within the `data_analysis` folder of the `job_files` directory.



(a) Distribution of all pixel values.

(b) Distribution of image's maximum values.

Figure 5: Statistical analysis of ground truth images.

3.4 Metrics

This subsection is based on the methodology and definitions provided in [4, 5, 8].

To assess the quality of the reconstructed images in this project, three metrics were employed: the mean squared error (MSE), the peak signal-to-noise ratio (PSNR), and the structural similarity index (SSIM). All models were trained using the L2 loss, which corresponds to the MSE.

Mean Squared Error (MSE): The MSE measures the average squared difference between the predicted image x and the ground truth image y:

$$MSE(x, y) = \frac{1}{N} \sum_{i=1}^{N} (x_i - y_i)^2 \tag{1}$$

where N is the total number of pixels. Lower MSE values indicate higher reconstruction accuracy.

Peak Signal-to-Noise Ratio (PSNR): PSNR is a widely used metric for image quality assessment, especially in CT reconstruction. It is derived from the MSE and expressed in decibels:

$$PSNR(x, y) = 10 \cdot \log_{10} \left(\frac{\mu_{\max}^2}{MSE(x, y)} \right) \tag{2}$$

Here, $\mu_{\max}$ represents the maximum possible pixel value in the image, which is 1 in this study since all images are normalized to the $[0, 1]$ range. Higher PSNR values correspond to better image quality.

Structural Similarity Index (SSIM): SSIM evaluates the perceived quality of an image by comparing structural information, rather than just pixel-wise differences. It is computed as:

$$SSIM(a, b) = \frac{(2\mu_a\mu_b + C_1)(2\sigma_{ab} + C_2)}{(\mu_a^2 + \mu_b^2 + C_1)(\sigma_a^2 + \sigma_b^2 + C_2)} \tag{3}$$

where:

- μ_a, μ_b are the local means of images a and b
- σ_a^2, σ_b^2 are the local variances
- σ_{ab} is the local covariance between a and b
- C_1, C_2 are constants to stabilize the division

In practice, SSIM is computed over small sliding windows across the image. The final score is the average of all local SSIM values. In this project, SSIM is implemented using the `pytorch_msssim` library. The function takes the reconstructed and ground truth images as tensors of shape (batch, channel, height, width), and returns a scalar representing the average SSIM across all spatial locations and batch elements. The per-batch results are further averaged to obtain the epoch-level SSIM.

3.5 Algorithms Specifications

3.5.1 DeepFBP model

The architecture used in this work is based on the original DeepFBP design, but includes several key modifications aimed at improving performance and training stability. An overview of the final architecture is shown in Figure 6.

The DeepFBP pipeline consists of three main components: a learnable frequency-domain filter, a nonlinear 1D interpolation network and a 2D convolutional image denoiser. In thos work's implementation, all components are trained jointly from the beginning using the Mean Squared Error (MSE) loss function, in contrast to the original staged training strategy proposed in the DeepFBP paper. The article proposes a three-stage approach: phase 1, training only the learnable filter for 200 epochs with a learning rate of 10^{-3}; phase 2, training the filter and the interpolator for 100 epochs with the same learning rate; and phase 3, fine-tuning the entire model jointly for another 100 epochs using a reduced learning rate of 10^{-4}. However, this staged scheme imposes that the interpolator and the denoiser must act as identical functions during the initial training (phase 1 and/or phase 2)and it is not known if this assumption has been fulfilled in practice. Consequently, the filter may adapt to unrealistic intermediate representations of the sinogram, resulting in suboptimal filtering behaviour in the final full model.

The filtering stage applies zero padding along the detector dimension of the sinogram before transforming it to the frequency domain. This prevents boundary artifacts during FFT filtering. After applying the learnable filter, the padded region is removed to restore the original sinogram dimensions.

The interpolation module consists of three depthwise 1D convolutional residual blocks followed by a final 1D convolution layer. Each layer uses a kernel size of 3 and operates along the detector dimension with 1 input and 1 output channel. Unlike the original implementation where one interpolator was applied per angle, here a single interpolator is shared across all angles. This design treats the sinogram as a whole rather than processing each angle independently, which led to more stable training and better reconstruction results in our experiments.

The denoising network comprises an initial 1×1 convolution that expands the input from 1 to 64 channels, followed by a 3×3 convolution that maintains this dimensionality. This is followed by three residual blocks, each consisting of two 3×3 convolutions with 64 channels. A final 3×3 convolution reduces the output back to a single channel. All layers use ReLU activations except the final one, which uses a HardTanh to constrain pixel values to the [0, 1] range.

Additionally, a normalization step was introduced before the denoiser to stabilize the amplitude range across inputs. This normalization is computed by applying the back-projection operator to a uniform sinogram (filled with ones). Compared to other strategies such as Batch Normalization or learnable normalization blocks, this operator approach was found to be both more efficient and more consistent across samples.

The model was trained for 90 epochs with a batch size of 10, using the AdamW optimizer with an initial learning rate of 10^{-3}. A learning rate scheduler with a patience of 15 epochs was employed. The total number of epochs was limited due to computational constraints, as training each model required approximately 10 hours on the available GPUs.

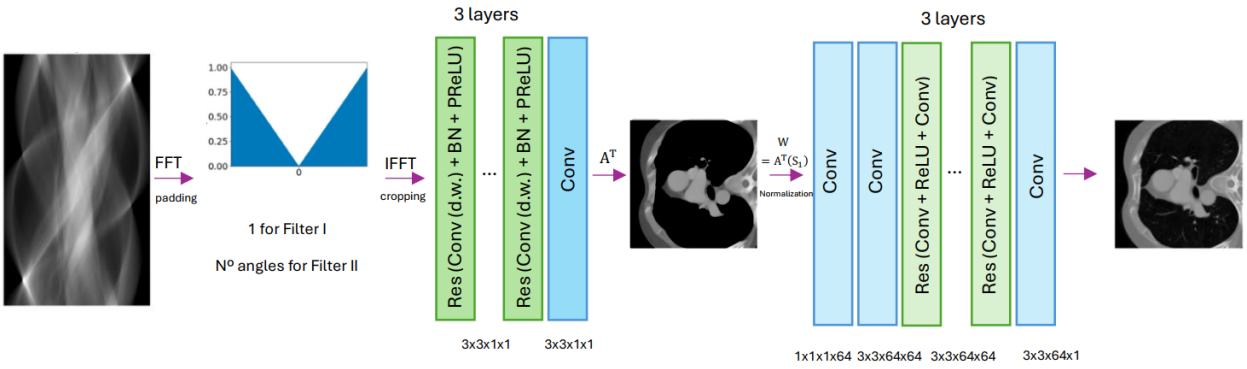


Figure 6: Schematic architecture of the DeepFBP model used.

3.5.2 DBP model

The DBP model was implemented following the original publication [2], preserving its convolutional architecture and training principles. In the original work, the model is trained for 50 epochs using a gradually decaying learning rate, and is evaluated using 16 single-view back-projections.

In this project, the number of single-view back-projections was increased to 90 in order to match the sparse-view setup used in the DeepFBP experiments. This modification ensures a fair and consistent comparison between both models under identical data conditions. An overview of the model architecture is shown in Figure 7.

The model was trained using the MSE loss function, with the Adam optimizer initialized at a learning rate of 10^{-3}. A learning rate scheduler with a patience of 10 epochs was applied. Due to computational constraints, each training run requiring approximately 10 hours, the number of epochs was limited to 25. Despite this reduction, the model reached stable performance under the given settings.

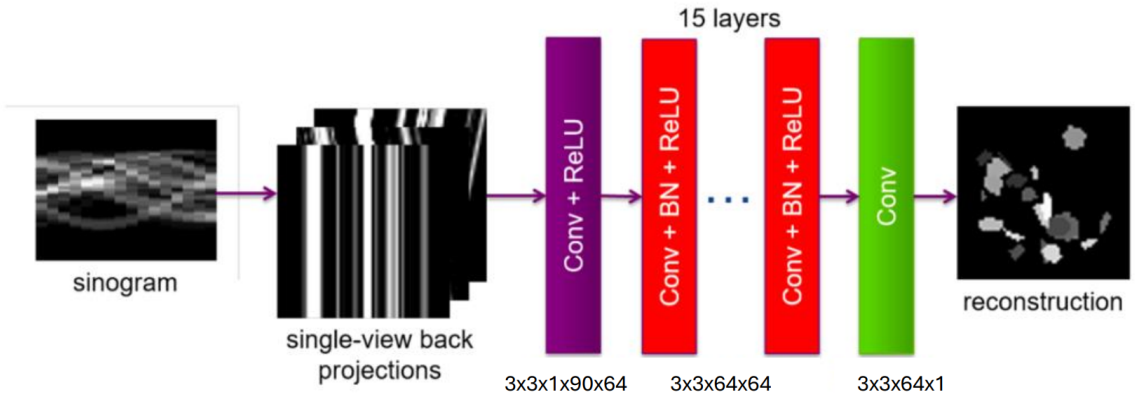


Figure 7: Schematic architecture of the DBP model used.

3.5.3 FusionFBP model

The FusionFBP model follows the same architecture as DeepFBP up to the image reconstruction stage. This includes the learnable frequency-domain filter, zero padding along the detector axis and the nonlinear interpolation module composed of three depthwise 1D residual blocks followed by a 1D convolution layer. As in DeepFBP, the interpolation module is shared across all angles, processing the sinogram globally.

After backprojection, however, the architecture diverges. Instead of the residual 2D CNN used in DeepFBP, the FusionFBP model adopts the denoising network from DBP. This network consists of 15 sequential 2D convolutional layers, each followed by Batch Normalization and ReLU activation. The first layer processes the reconstructed image as a single input channel and progressively extracts higher-level features while preserving spatial dimensions. This substitution aims to leverage the expressiveness of DBP's deep image-space denoiser, while retaining the physics-based filtering and interpolation stages of DeepFBP. The architecture of FusionFBP is illustrated in Figure 8.

The model is trained end-to-end, jointly optimizing the filter, interpolator and denoiser. Both shared (Filter I) and per-angle (Filter II) variants of the learnable filter were used in separate training runs. Training was conducted for 50 epochs with a batch size of 10, using the AdamW optimizer and an initial learning rate of 10^{-3}, with a learning rate scheduler and a patience of 15 epochs.

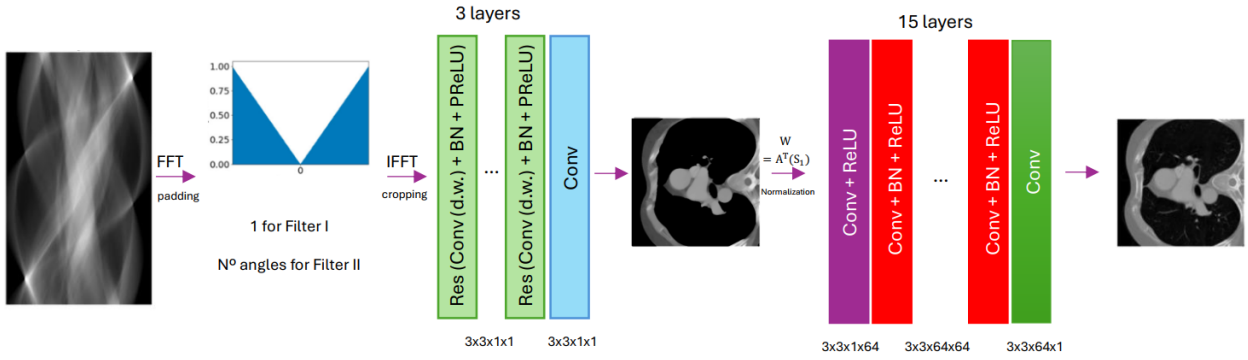


Figure 8: Schematic architecture of the FusionFBP model.

3.5.4 DeepFusionBP model

The DeepFusionBP model combines elements from both DeepFBP and DBP into a unified hybrid architecture. Like DeepFBP, the model begins with a learnable frequency-domain filter, preceded by zero padding along the detector axis to avoid FFT boundary artifacts. This is followed by the same nonlinear interpolation module used in DeepFBP, composed of three depthwise 1D residual convolutional blocks and a final shared 1D convolution. As before, the interpolation operates globally across all projection angles, using a kernel size of 3 and 1 input/output channel per layer.

After filtering and interpolation, the sinogram is backprojected to generate 90 single-view reconstructed images, one per projection angle. These 2D slices are stacked along the channel dimension. This tensor is then passed to the denoising module, which adopts the same CNN architecture originally used in DBP. The DBP denoiser consists of 15 sequential 2D convolutional layers, each followed by Batch Normalization and ReLU activations. All convolutions use 3×3 kernels and maintain the spatial resolution of the input. The architecture of DeepFusionBP is illustrated in Figure 9

The motivation behind this architecture is to preserve the physics-informed arquitecture of DeepFBP, while replacing the final denoising stage with a deep feature extractor. Unlike DeepFBP, where the denoising network operates on a single reconstructed image, DeepFusionBP exploits the full set of angular projections in image space, thus enhancing representational capacity and learning from inter-angle dependencies.

The model is trained end-to-end, jointly optimizing the filter, interpolation network and the DBP-style denoiser. Both filter variants were considered in separate runs. The training was conducted for 25 epochs using the AdamW optimizer with an initial learning rate of 10^{-3}, a batch size of 8 and a learning rate scheduler with a patience of 15 epochs.

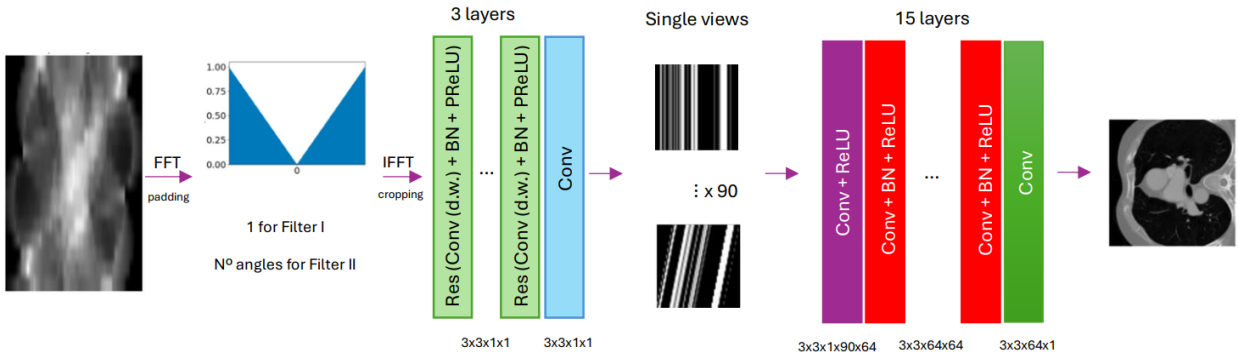


Figure 9: Schematic architecture of the DeepFusionBP model.

4 Results

4.1 Normal dose sinograms

In this section, the performance of the reconstruction methods using full-view normal-dose sinograms was evaluated. For this experiment, the LoDoPaBDataset was used with a photon count of $I_0 = 10^5$ and a Gaussian noise standard deviation of $\sigma = 0.001$. Since Poisson noise dominates under these conditions, the Gaussian component has a relatively minor impact.

It is important to note that the models DBP and DeepFusionBP are not included in this experiment. These methods are based on single-angle backprojections, which are designed for sparse-view settings. In a full-view scenario with 1000 angles, using 1000 individual backprojections with TomoSimpo operators is effectively equivalent to a standard full backprojection, but much more computatinally expensive. Therefore, it was considered unnecessary to apply these models in the full-view case.

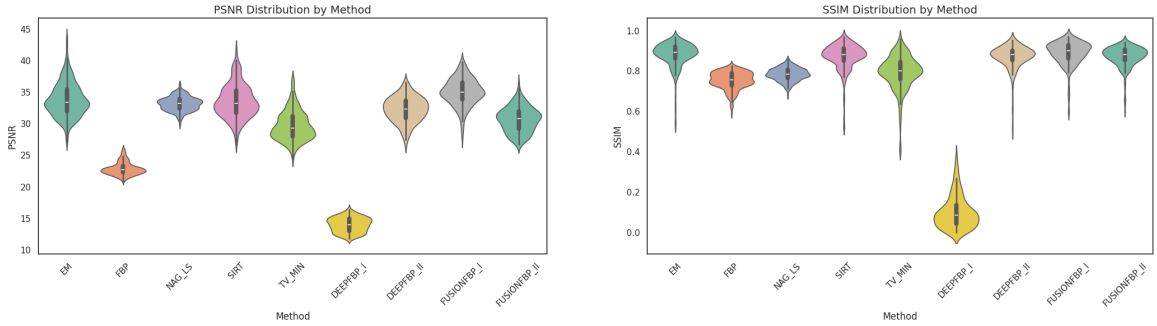
Figure 10 shows the distribution of PSNR and SSIM values across all methods. It can be observed that the proposed methods, except for DeepFBP I, outperform the traditional methods in terms of both PSNR and SSIM. The detailed numerical results are summarized in Table 4.

Figure 11 presents visual examples of reconstructions for each method. As can be seen, the DeepFBP I model fails to reconstruct meaningful images and returns an almost completely black result. This failure is not due to the absence of output, but rather because the network does not learn to produce images within the normalized intensity range of [0, 1]. All reconstructed images are plotted within this same range, which is why DeepFBP I appears entirely black despite technically producing a CT image (see Appendix).

Overall, the results show that the proposed approaches can yield substantial improvements under normal-dose conditions, with FusionFBP I achieving the highest quantitative metrics across all measures.

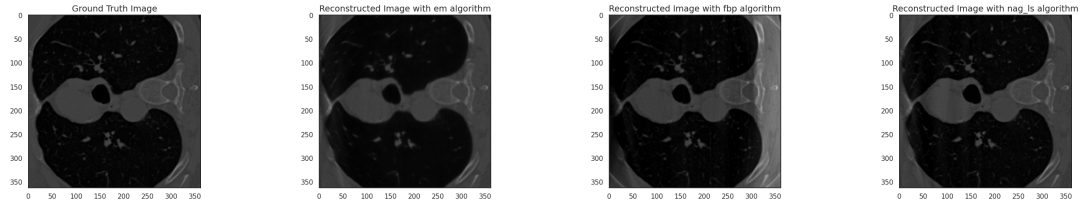
Method	**PSNR ± std**	**SSIM ± std**	**MSE ± std**
EM	33.8354 ± 2.6899	0.8820 ± 0.0626	0.0005 ± 0.0003
FBP	22.8887 ± 1.0185	0.7542 ± 0.0436	0.0053 ± 0.0011
NAG-LS	33.2092 ± 1.1133	0.7843 ± 0.0322	0.0005 ± 0.0001
SIRT	33.5809 ± 2.6728	0.8726 ± 0.0652	0.0005 ± 0.0003
TV-MIN	29.7258 ± 2.3436	0.7946 ± 0.0853	0.0012 ± 0.0006
DeepFBP I	13.9939 ± 1.1397	0.1019 ± 0.0803	0.0413 ± 0.0109
DeepFBP II	32.2884 ± 1.9893	0.8709 ± 0.0592	0.0007 ± 0.0003
FusionFBP I	**35.0735 ± 2.1501**	**0.8893 ± 0.0585**	**0.0004 ± 0.0002**
FusionFBP II	30.6965 ± 1.9627	0.8729 ± 0.0496	0.0009 ± 0.0004

Table 4: Quantitative results (mean ± std) for full-view normal-dose reconstruction.



(a) PSNR Distribution (b) SSIM Distribution

Figure 10: Distribution of PSNR and SSIM for full-view normal-dose reconstruction methods.

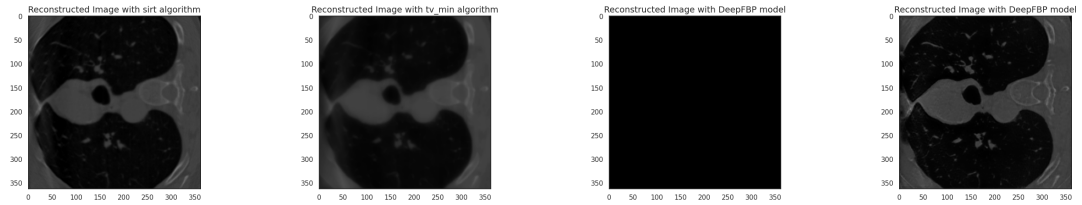


(a) Ground Truth

(b) EM
PSNR: 33.19 dB
SSIM: 0.90

(c) FBP
PSNR: 21.93 dB
SSIM: 0.75

(d) NAG-LS
PSNR: 33.71 dB
SSIM: 0.81

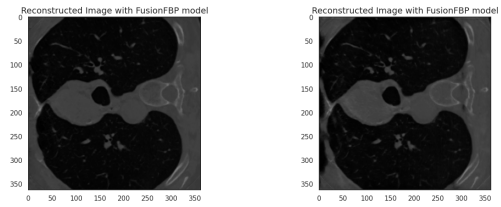


(e) SIRT
PSNR: 33.53 dB
SSIM: 0.91

(f) TV-MIN
PSNR: 28.42 dB
SSIM: 0.83

(g) DeepFBP I
PSNR: 15.89 dB
SSIM: 0.07

(h) DeepFBP II
PSNR: 30.25 dB
SSIM: 0.90



(i) FusionFBP I
PSNR: **36.26 dB**
SSIM: **0.95**

(j) FusionFBP II
PSNR: 29.32 dB
SSIM: 0.92

Figure 11: Reconstructed images using traditional and proposed methods for full-view normal dose sinograms.

4.2 Low-dose sinograms

In this section, the performance of the models on full-view low-dose sinograms is analyzed. Unlike the normal-dose case, the paper [5] does not specify a concrete value for the photon count I_0, but instead refers to the dataset used, the 2016 NIH-AAPM-Mayo Clinic Low Dose CT Grand Challenge [9]. Based on the documentation associated with that challenge, the scans correspond approximately to 25% of the original radiation dose. Therefore, in these experiments a photo count of $I_0 = 2.5 \times 10^4$ have been used, while keeping the Gaussian noise level at $\sigma = 0.001$ to maintain consistency with the normal-dose setting.

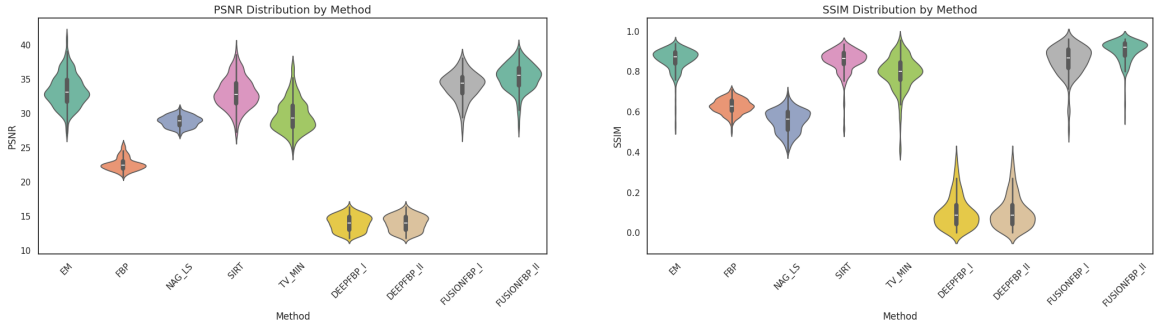
Similarly to the previous section, results for the DBP and DeepFusionBP models were not included. These methods rely on the use of individual backprojection operators, which offer no advantage in a full-view setup.

Figure 12 shows the PSNR and SSIM distributions for all methods under the low-dose setting. The numerical summary of these results is reported in Table 5. It can be observed that the FusionFBP II model significantly outperforms all other methods, including traditional approaches such as EM and SIRT. Its performance is also consistently better than the rest of the proposed methods, achieving the best results in all three metrics.

As seen in the qualitative examples presented in Figure 13, both DeepFBP I and DeepFBP II fail to produce visually meaningful reconstructions. As in the normal-dose case, this is not because the network fails entirely, but rather because the output is not mapped correctly to the normalized range [0, 1], making the reconstructed image appear entirely black. All visualizations are normalized within this fixed range for consistency across methods.

Method	**PSNR ± std**	**SSIM ± std**	**MSE ± std**
EM	33.3370 ± 2.3516	0.8605 ± 0.0591	0.0005 ± 0.0003
FBP	22.6131 ± 0.9624	0.6283 ± 0.0371	0.0056 ± 0.0011
NAG-LS	28.8690 ± 0.8277	0.5558 ± 0.0577	0.0013 ± 0.0003
SIRT	32.9777 ± 2.2809	0.8540 ± 0.0621	0.0006 ± 0.0003
TV-MIN	29.7134 ± 2.3391	0.7942 ± 0.0851	0.0012 ± 0.0006
DeepFBP I	13.9939 ± 1.1397	0.1019 ± 0.0803	0.0413 ± 0.0109
DeepFBP II	13.9939 ± 1.1397	0.1019 ± 0.0803	0.0413 ± 0.0109
FusionFBP I	34.0483 ± 1.9514	0.8552 ± 0.0749	0.0004 ± 0.0002
FusionFBP II	**35.3107 ± 1.9374**	**0.9029 ± 0.0560**	**0.0003 ± 0.0002**

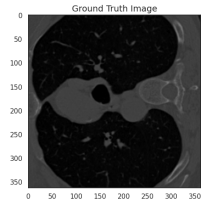
Table 5: Quantitative results (mean ± std) for full-view low-dose reconstruction.



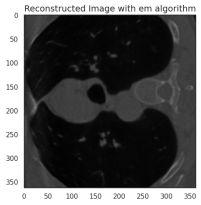
(a) PSNR Distribution

(b) SSIM Distribution

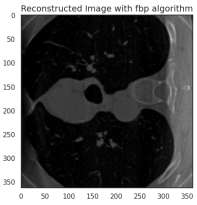
Figure 12: Distribution of PSNR and SSIM for full-view low-dose reconstruction methods.



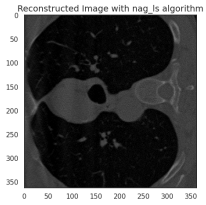
(a) Ground Truth



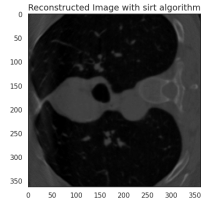
(b) EM
PSNR: 32.87 dB
SSIM: 0.89



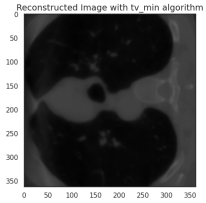
(c) FBP
PSNR: 21.77 dB
SSIM: 0.65



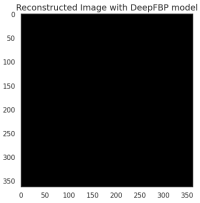
(d) NAG-LS
PSNR: 29.82 dB
SSIM: 0.60



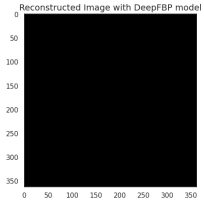
(e) SIRT
PSNR: 33.02 dB
SSIM: 0.89



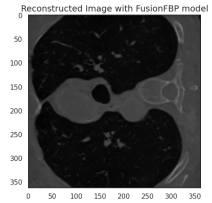
(f) TV-MIN
PSNR: 28.42 dB
SSIM: 0.83



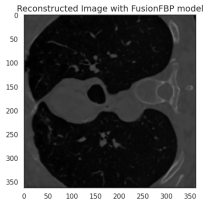
(g) DeepFBP I
PSNR: 15.89 dB
SSIM: 0.07



(h) DeepFBP II
PSNR: 15.89 dB
SSIM: 0.07



(i) FusionFBP I
PSNR: 34.77 dB
SSIM: 0.91



(j) FusionFBP II
PSNR: **35.90 dB**
SSIM: **0.93**

Figure 13: Reconstructed images using traditional and proposed methods for full-view low-dose sinograms.

4.3 Sparse-view low-dose sinograms

This is the last experiment mentioned in the paper [5]. In a manner consistent with the low-dose context, the parameters $I_0 = 2.510^4$ and $\sigma = 0.001$ were used to simulate the noise. Furthermore, the number of projection angles was reduced to just 90, following the sparse-view configuration described in the original work.

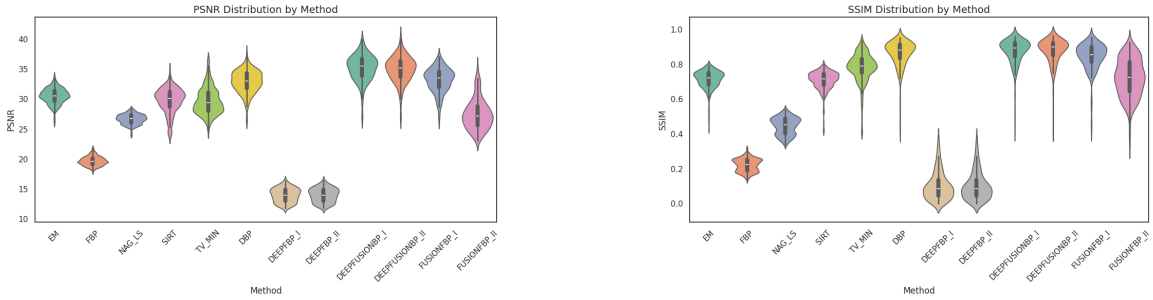
In this case, it was observed that the algorithms specifically designed for sparse-view data, namely DBP and DeepFusionBP, achieved the best results. Their performance even surpasses that of FusionFBP, although the latter still shows excellent consistency across different settings.

As in the previous experiments, DeepFBP I and II failed to correctly map the reconstructed images to the [0,1] range. Consequently, although they generate CT images internally, the output appears black in the visualizations.

Figure 14 shows the distribution of PSNR and SSIM across all methods, and Table 6 summarizes the quantitative metrics. In Figure 15, example reconstructions for all evaluated models are presented.

Method	**PSNR ± std**	**SSIM ± std**	**MSE ± std**
EM	30.5609 ± 1.2945	0.7146 ± 0.0528	0.0009 ± 0.0003
FBP	19.6834 ± 0.7591	0.2220 ± 0.0370	0.0109 ± 0.0019
NAG-LS	26.6834 ± 0.8370	0.4470 ± 0.0496	0.0022 ± 0.0004
SIRT	29.7956 ± 2.1013	0.7085 ± 0.0552	0.0012 ± 0.0007
TV-MIN	29.7207 ± 2.2525	0.7835 ± 0.0825	0.0012 ± 0.0006
DBP	32.9578 ± 1.8297	0.8616 ± 0.0799	0.0006 ± 0.0003
DeepFBP I	13.9939 ± 1.1397	0.1019 ± 0.0803	0.0413 ± 0.0109
DeepFBP II	13.9939 ± 1.1397	0.1019 ± 0.0803	0.0413 ± 0.0109
DeepFusionBP I	**35.2398 ± 2.1637**	0.8746 ± 0.0784	**0.0003 ± 0.0002**
DeepFusionBP II	35.0330 ± 2.1587	**0.8757 ± 0.0794**	0.0004 ± 0.0003
FusionFBP I	33.2120 ± 2.0090	0.8454 ± 0.0800	0.0005 ± 0.0003
FusionFBP II	27.4828 ± 2.4974	0.7228 ± 0.1091	0.0021 ± 0.0010

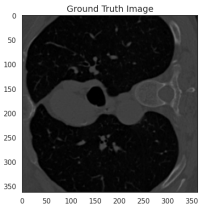
Table 6: Quantitative results (mean ± std) for sparse-view low-dose reconstruction.



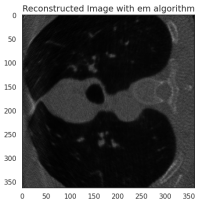
(a) PSNR Distribution

(b) SSIM Distribution

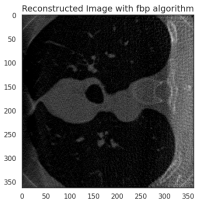
Figure 14: Distribution of PSNR and SSIM for sparse-view low-dose reconstruction methods.



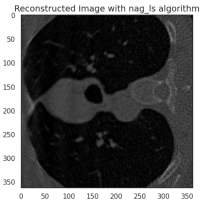
(a) Ground Truth



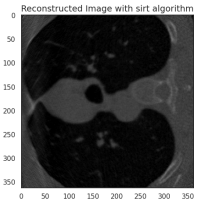
(b) EM
PSNR: 30.45 dB
SSIM: 0.76



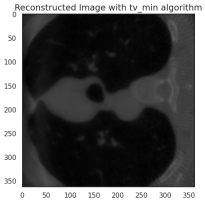
(c) FBP
PSNR: 19.53 dB
SSIM: 0.25



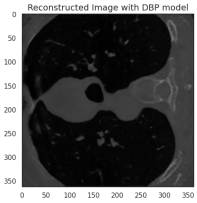
(d) NAG-LS
PSNR: 27.43 dB
SSIM: 0.50



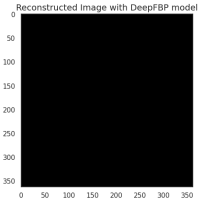
(e) SIRT
PSNR: 31.15 dB
SSIM: 0.76



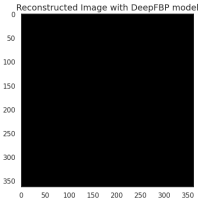
(f) TV-MIN
PSNR: 28.43 dB
SSIM: 0.82



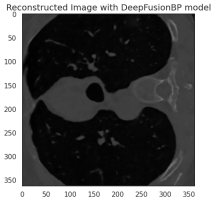
(g) DBP
PSNR: 32.83 dB
SSIM: 0.89



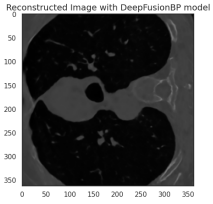
(h) DeepFBP I
PSNR: 15.89 dB
SSIM: 0.07



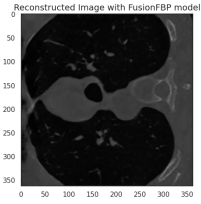
(i) DeepFBP II
PSNR: 15.89 dB
SSIM: 0.07



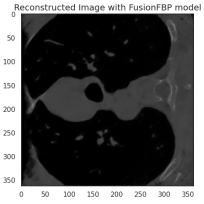
(j) DeepFusionBP I
PSNR: **35.61 dB**
SSIM: **0.92**



(k) DeepFusionBP II
PSNR: 34.43 dB
SSIM: 0.90



(l) FusionFBP I
PSNR: 31.66 dB
SSIM: 0.90



(m) FusionFBP II
PSNR: 25.77 dB
SSIM: 0.57

Figure 15: Reconstructed images using traditional and proposed methods for sparse-view low-dose sinograms.

5 Discussion

This project uses three metrics, MSE, PSNR, and SSIM, to evaluate the quality of the reconstructed images, presented in Section 3.4 and following the original papers [2, 5]. While this choice ensures consistency, it is important to highlight that none of the referenced works provide a critical justification for selecting these metrics, nor do they discuss their limitations. This omission weakens the strength of their evaluation frameworks.

Although MSE and PSNR are widely used due to their simplicity, they are known to correlate poorly with perceptual image quality. In particular, they fail to capture structural distortions and may assign similar values to images that are visually very different. SSIM, on the other hand, incorporates luminance, contrast, and structural similarity, offering a better approximation of human visual perception. However, it is sensitive to minor geometric misalignments and implementation choices, such as window size and constant values.

The DeepFBP model builds upon the classic FBP algorithm by introducing learnable components: a filter, a nonlinear interpolation operator and a denoiser. The overall structure preserves the physical interpretability of FBP, which is one of the model's strengths. Nevertheless, based on the implementation and behavior of the network, it is believed that the so-called nonlinear interpolation block functions more like a second filtering stage, applied in the spatial domain rather than in the frequency domain. This observation does not affect the model's structure but raises questions about the naming and interpretation of this component.

The authors propose initializing the learnable filter with the Ram-Lak filter, a common choice in classical FBP. However, this default option may not be optimal in this context. Exploring alternative initializations could potentially accelerate convergence and improve results. In fact, the training strategy described in the paper is surprisingly long: 200 epochs to train the filter, 100 for the filter and interpolator together and another 100 for the full model. Such a setup (400 epochs in total) suggests either high training complexity or potentially suboptimal learning rate selection. The authors do not comment on these aspects, nor do they offer any ablation study to support their design.

While the dataset used in DeepFBP is well known, the paper does not explore whether the model generalizes to other datasets or acquisition settings. In particular, no experiments are conducted to test robustness to variations in angular sampling, such as uneven spacing or angle offsets. Although during the development there was not time to explore this fully, the necessary code to perform these tests is implemented to test it in the future.

The experimental comparison presented in the DeepFBP paper is also limited. Only four models are used as baselines: two traditional (FBP and TV) and two deep learning methods (FBPConvNet and RED-CNN). Other established iterative approaches such as SIRT or EM are not considered. In addition, the authors do not report standard deviations or any other measure of variability. This is problematic, especially considering that some of the reported performance differences between models are small. Without statistical validation, claims of superiority remain questionable.

Reproducing the DeepFBP model also revealed several practical difficulties. The training procedure described in the paper involves multiple stages, but the initialization strategy is not explained in detail. In this project, it was chosen to train the models end-to-end, without separating phases. Since both the interpolator and the denoiser are based on convolutional layers with residual connections, training them separately without proper initialization (e.g., identity functions for convolutions and zero initialized residuals) may result in suboptimal learning trajectories. Notably, the original paper does not clarify how the parameters were initialized.

Architecturally, the DeepFBP paper lacks important implementation details. While the filtering block is clearly described, the interpolator and denoiser components are not. Key parameters such as the number of channels, kernel sizes and padding values are omitted. The denoiser is described as being based on another model [**?**], but the architecture is not specified, forcing to make design assumptions based on related work. In general, reproducing the full model exactly as intended was not feasible with the information provided.

One notable feature mentioned in the paper is that the interpolation block applies angle-wise convolutions. It was attempted to implement this idea using Tomosipo, but the results were not successful. The hypothesis is that these outcomes were due to the implementation of phased training, in conjunction with inadequate initialization procedures. With full training and proper initialization, this approach might still be viable and could improve the model's capacity.

In terms of performance, the DeepFBP paper claims that the model outperforms traditional methods in all tested settings: normal dose, low dose and sparse-view low dose. However, in these experiments experiments, this was only partially confirmed. With Filter I, the denoiser component was too limited to correctly adjust the dynamic range of the output image to $[0, 1]$. While the backprojection step produced a structurally valid CT image (see Appendix B), the denoiser was not effective. Filter II performed better in the normal dose case, but again failed to correct intensity values in more challenging settings. These discrepancies suggest that additional capacity or structural changes may be needed, possibly by revisiting the use of per-angle convolutions.

An initial misunderstanding between papers with similar titles resulted in the implementation of the DBP model [2], a CNN-based reconstruction method that utilises 16 projection angles and a stack of single-view backprojections as input. Although this was not the target model, it served as a useful test to validate my training and evaluation pipeline.

Subsequently, the DBP model was adapted to a 90 views configuration in order to align with the sparse-view scenario employed in DeepFBP. While this prevents direct comparison with the original results, the DBP paper was found to be much clearer and easier to reproduce. The authors provide a comprehensive description of the network architecture, including the number of channels, kernel sizes and an informative diagram. Although padding is not explicitly mentioned , it can be inferred from the output dimensions. The principal deficiency is the absence of a dataset description, which restricts the reproducibility and interpretability of the results.

The authors explicitly state that they used synthetic, noise-free images to generate the sinograms through the Radon transform. This choice implies the use of a parallel-beam geometry, although this is never directly discussed or justified. No real CT data, noise modeling or acquisition parameters (e.g., dose, resolution, number of detectors) are included, which severely restricts the clinical relevance of their results. Furthermore, the model is only compared to traditional FBP, without considering any iterative or deep learning methods. This limited scope may be due to the paper's early publication date (2018), when fewer deep learning models were available. Nevertheless, it weakens the strength of their conclusions.

It is also worth noting that the DBP paper only evaluates the model using PSNR and SSIM, although the authors do report standard deviations, a positive detail often omitted in other works. In the present implementation, the original number of channels (64) was maintained despite an increase in the number of input angles to 90. The efficacy of this design was demonstrated; however, its performance could be enhanced through the adaptation of the number of channels to the number of input projections. Furthermore, the original study did not assess the system's robustness to variations in angular configuration, such as the use of offset or non-uniform angles.

Interestingly, the DBP model served as inspiration for further development. It was observed that the denoiser employed in DeepFBP lacked sufficient capacity. Consequently, an experiment was conducted to replace it with the architecture proposed in DBP. This led to the creation of two hybrid models: DeepFusionBP and FusionFBP (described in Sections 3.5.3 and 3.5.4). These models differ not only in performance but also in architectural design and intended use. DeepFusionBP, which retains the angle-wise processing characteristic of DBP, is particularly suited for sparse-view acquisition scenarios. However, this specialization also makes it less robust when applied to more complete datasets.

In contrast, FusionFBP is structurally closer to the original DeepFBP model but incorporates a modified version of the DBP architecture as its denoising block. It is designed to accept a single channel input rather than separate channels per view, making it more flexible and generalizable. Among all the models tested, FusionFBP achieved the best overall performance in the normal-dose and low-dose settings. In the sparse-view case, it was slightly outperformed by DeepFusionBP, though the difference in performance was not substantial.

6 Conclusions

This work has explored the implementation, evaluation and critical analysis of several deep learning-based CT reconstruction models, with a particular focus on the DeepFBP architecture. A central aspect of the study has been the challenge of scientific reproducibility. In the case of DeepFBP, reproducing the model faithfully proved difficult due to missing architectural details and design decisions that were not clearly justified. This lack of transparency limited the ability to verify and extend the proposed approach. In contrast, the DBP paper, although narrower in scope and based on synthetic data, offers a clear and detailed description of the model architecture.

Among all the models tested in this project, the most robust and versatile was FusionFBP. Its architecture, based on a modified DBP denoiser adapted for single channel input, allowed it to perform consistently across different dose levels. Unlike DeepFusionBP, which was specifically tailored for sparse-view acquisitions, FusionFBP demonstrated strong performance in normal and low-dose cases and remained competitive even in the sparse-view scenario.

Nonetheless, several avenues for improvement remain. First, it would be beneficial to modify the architecture of DeepFBP, DeepFusionBP and FusionFBP to incorporate, again, angle-wise convolutions in the interpolator. This would allow for a more rigorous comparison with the original design of DeepFBP and potentially improve reconstruction quality.

Second, a more thorough robustness study should be carried out. For example, testing whether the models can handle angular offsets between training and testing data would provide insights into their generalization capabilities. Similarly, applying small spatial shifts to the input sinograms could help determine how sensitive each model is to slight misalignments.

Another promising direction would be to experiment with different initializations of the learned filter. The Ram-Lak filter, while commonly used, may not be the most suitable choice for all training configurations. Finding a better initialization could help the model converge faster and achieve improved results.

In addition, future work should consider incorporating alternative evaluation metrics beyond MSE, PSNR and SSIM. These classical metrics, though convenient, do not always correlate well with human perception of image quality. Exploring perceptual or task-specific measures could offer a more meaningful evaluation of reconstruction performance.

A further improvement concerns the generation of sinograms. Although a GPU-compatible simulation pipeline was successfully used to generate fan-beam sinograms for the experiments, an alternative procedure designed to avoid the inverse crime was initially attempted, following the methodology described in the original LoDoPaB-CT paper. However, due to numerical instabilities in the GPU-based projection operators, this method could not be reliably used at the time. Revisiting and correctly implementing this approach in future work would be highly beneficial, as it would allow for more realistic and unbiased data simulation, thereby strengthening the robustness and

clinical relevance of the reconstruction models.

Finally, it would be worthwhile to test the performance of FusionFBP in ultra-low-dose scenarios, such as 5% or even 1% of the standard dose. Investigating whether the model remains stable and accurate under such extreme conditions could reveal its true potential for clinical applications where radiation exposure must be minimized.

Overall, this project demonstrates that, despite the challenges of replicating existing models, carefully designed adaptations and critical evaluation can lead to robust and high-performing alternatives for CT image reconstruction.

References

[1] Rigaku Corporation. How does ct reconstruction work?, 2023. URL https://rigaku.com/products/imaging-ndt/x-ray-ct/learning/blog/how-does-ct-reconstruction-work. Accessed: 2025-06-08.

[2] Dong Hye Ye, Gregery T. Buzzard, Max Ruby, and Charles A. Bouman. Deep back projection for sparse-view ct reconstruction. 2018. URL https://arxiv.org/abs/1807.02370.

[3] National Institute of Biomedical Imaging and Bioengineering (NIBIB). Computed tomography (ct). https://www.nibib.nih.gov/science-education/science-topics/computed-tomography-ct, 2022. Accessed: 2025-05-25.

[4] Johannes Leuschner, Maximilian Schmidt, Daniel Otero Baguer, and Peter Maass. Lodopab-ct, a benchmark dataset for low-dose computed tomography reconstruction. *Scientific Data*, 8(1):109, 2021. doi: 10.1038/s41597-021-00893-z. URL https://doi.org/10.1038/s41597-021-00893-z.

[5] Xi Tan, Xuan Liu, Kai Xiang, Jing Wang, and Shan Tan. Deep filtered back projection for ct reconstruction. *IEEE Access*, 12:20962–20972, 2024. doi: 10.1109/ACCESS.2024.3357355.

[6] Hu Chen, Yi Zhang, Mannudeep K. Kalra, Feng Lin, Yang Chen, Peixi Liao, Jiliu Zhou, and Ge Wang. Low-dose ct with a residual encoder-decoder convolutional neural network. *IEEE Transactions on Medical Imaging*, 36(12), 2017. ISSN 1558-254X. doi: 10.1109/tmi.2017.2715284. URL http://dx.doi.org/10.1109/TMI.2017.2715284.

[7] Kyong Hwan Jin, Michael T. McCann, Emmanuel Froustey, and Michael Unser. Deep convolutional neural network for inverse problems in imaging. *IEEE Transactions on Image Processing*, 26(9):4509–4522, 2017. doi: 10.1109/TIP.2017.2713099.

[8] PyTorch-Ignite Contributors. ignite.metrics.ssim — pytorch-ignite v0.4.12 documentation. https://docs.pytorch.org/ignite/generated/ignite.metrics.SSIM.html, 2024. Accessed: 2025-06-08.

[9] Cynthia H. McCollough, Amy C. Bartley, Rickey E. Carter, Baochang Chen, Timothy A. Drees, Patrick Edwards, David R. Holmes, Ai E. Huang, Faisal Khan, Shuai Leng, Kyle L. McMillan, Gregory J. Michalak, Katherine M. Nunez, Lifeng Yu, and Joel G. Fletcher. Low-dose ct for the detection and classification of metastatic liver lesions: Results of the 2016 low dose ct grand challenge. *Medical Physics*, 44 (10):e339–e352, 2017. doi: 10.1002/mp.12345.

Appendices

A Sinogram Simulation Based on LoDoPaB-CT paper

An initial attempt was made to reproduce the sinogram simulation pipeline as described in the original LoDoPaB-CT paper [4], with the goal of generating full-dose and low-dose sinograms consistent with the dataset's acquisition model. The procedure followed the data generation methodology outlined in the original work, which includes:

- Upsampling the ground truth images to a higher resolution (1000×1000 pixels) to prevent the inverse crime.
- Using tomosipo's cone operator to simulate fan-beam projections from the upsampled images.
- Adding Poisson noise with $N_0 = 4096$ and applying the Beer–Lambert law to simulate low-dose conditions.

However, the results obtained using this approach were not satisfactory. The simulated sinograms did not align well with the characteristics of the original dataset and the reconstructions based on them produced bad quality image quality (or any images at all). It was observed that the original publication already mentions that the most accurate results were obtained using the CPU backend (`astra_cpu`), due to numerical instabilities in the GPU-based implementation (`astra_cuda`) at certain projection angles and detector positions.

Due to hardware limitations and the impracticality of using the CPU backend for large-scale data generation on available resources, it was decided to use an alternative simulation pipeline described in Section 3.1, which is simplier and compatible with GPU acceleration, altought it does not account for the inverse crime.

The results of this approach were not used in the final experiments. Nevertheless, for transparency and reproducibility, the full implementation used in this initial experiment is included in the file `jobs_files/data_analysis/sinogram_simulation_try.py`.

B Intermediate CT Output Before and After Denoising for DeepFBP model

This appendix provides a visual example of the intermediate CT reconstruction obtained after the interpolation step (backprojection), and before the denoising stage. This intermediate image illustrates that the model is able to produce a structurally valid CT image even before applying the final denoiser. However, as shown, the intensity range is not yet correctly adjusted, and the output lies outside the desired $[0, 1]$ interval.

For comparison, the denoised output is also shown. These visualizations support the claim made in the discussion section: that the denoiser in the original DeepFBP model may not be strong enough to fully correct the interpolated output.

(a) Before denoising (max = 60.68) (b) After denoising (max = 1)

Figure 16: Intermediate and final CT reconstructions obtained with DeepFBP. The denoising block corrects the image range and slightly improves visual quality.

C Declarations of Use of Autogeneration Tools

Throughout the development of this project, both ChatGPT and DeepL Write were used as assistants to support code generation, debugging, documentation, and text refinement. Their use was carefully managed to ensure that all substantive technical contributions remained under the control of the author, with these tools serving primarily to enhance clarity, consistency, and development efficiency.

Text assistance: ChatGPT was employed as a writing assistant to improve the structure, fluency and readability of the report. All technical content, ideas, analyses and conclusions were fully developed by the author. ChatGPT was used to transform initial drafts, originally written in Spanish, into well-structured and natural-sounding English text. In this context, the tool acted similarly to a linguistic editor or translator, helping to better convey the author's intended meaning without altering or contributing to the underlying ideas.

In addition, ChatGPT was used during the development process to improve the structure and readability of code. In particular, it was helpful for generating and refining docstrings across all files, classes and functions, ensuring consistency with the style adopted throughout the project. It also provided assistance in resolving implementation issues such as tensor shape mismatches, data loader errors and bugs in the training pipeline. Additionally, ChatGPT helped to optimize plotting utilities for visualizing training and test performance and contributed to the construction of modular classes for custom deep learning models like DeepFBP, DeepFusionBP and FusionFBP. Suggestions from ChatGPT were also incorporated into the design of the `configure_logger` function and the `EarlyStopping` callback, improving maintainability and robustness. Furthermore, ChatGPT played a role in refining the README file and automating parts of the Sphinx-based documentation generation process.

DeepL Write was used to improve the linguistic quality of the report and other project documentation. It served as a stylistic tool to enhance clarity, tone and grammatical correctness, particularly in formal and academic writing contexts. It is important to note that DeepL Write does not generate new content; rather, it reformulates existing text to make it more fluent and professional. It was not used to produce any technical explanation or content from scratch.

Together, these tools contributed to the overall presentation and readability of the project without replacing the author's critical role in its conceptualization, implementation and evaluation.