Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

# Deep Learning Ring Artifact Removal in Computed Tomography: A Two-Head U-Net Benchmarked Against Classical Radial profile Filtering

**Ekta Chahal**
Indian Institute of Technology Roorkee
**Mayank Goswami**
mayankfph@iitr.ac.in

Indian Institute of Technology Roorkee https://orcid.org/0000-0001-8829-9980

**Research Article**

**Keywords:** computed tomography, ring artifacts, U-Net, residual learning, synthetic phantoms, image restoration

**Posted Date:** November 18th, 2026

**DOI:** https://doi.org/10.21203/rs.3.rs-11739530/v1

**License:** This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

**Additional Declarations:** The authors declare no competing interests.

# Deep-Learning Ring Artifact Removal in Computed Tomography: A Two-Head U-Net Benchmarked Against Classical Radial-Profile Filtering

Ekta Chahal[1] and Mayank Goswami,[*1,2]

[1]Department of Physics, IIT Roorkee
[2]MF School of AI and DS, IIT Roorkee
mayank.goswami@ph.iitr.ac.in

November 14, 2026

**Abstract**

Ring artifacts, caused by detector pixels whose gain or offset is slightly wrong, appear in reconstructed computed tomography (CT) slices as concentric circles around the rotation centre. They can hide small features and break threshold-based segmentation. Classical ring removal relies on hand-tuned filters, and real scans have no ground truth for measuring how well a method works. This project builds a synthetic CT pipeline (phantom, parallel-beam projection, Poisson noise, detector gain and offset error, filtered back-projection) in which the true ring is known for every image, and trains a two-head U-Net that predicts the ring image, which is subtracted from the input, and a ring mask. On 1000 training, 100 validation and 100 test slices of $256 \times 256$ pixels, where the test slices come from a phantom family never seen in training, the U-Net raises the peak signal-to-noise ratio (PSNR) from 33.11 dB to 39.54 dB and the structural similarity (SSIM) from 0.868 to 0.957. A radial-profile classical baseline tuned on the validation set gains 0.77 dB on validation but loses 0.56 dB on the unseen family. The U-Net gain depends on ring type, from +13.9 dB for `dead_hot` rings to +0.4 dB for faint `band` rings. Training with 100, 500 and 1000 images gives 35.29, 38.42 and 39.54 dB; three random seeds give $39.40 \pm 0.15$ dB; removing the Dice term from the loss raises PSNR to 39.89 dB but lowers mask Dice from 0.640 to 0.605. The mask head is the weaker output (Dice 0.640 overall, 0.07 on `band` rings). Because all data are simulated, the sim-to-real gap remains the main open question.

**Keywords:** computed tomography, ring artifacts, U-Net, residual learning, synthetic phantoms, image restoration.

## 1 Introduction, Problem Objectives, and Overview

### 1.1 Motivation and Problem Objectives

In X-ray computed tomography, an object is imaged from many angles and a row of detector pixels records how much radiation passes through along each ray. A slice is then reconstructed from these projections, most commonly by filtered back-projection (FBP) [1]. Each detector pixel has its own gain, meaning how strongly it responds, and its own offset, meaning what it reads when no signal arrives. Calibration removes most of the difference between pixels, but a pixel that is slightly miscalibrated, dead or hot leaves a systematic error that shows up in the reconstruction as a ring [2, 3]. Rings matter because they can mimic or hide small structures such as pores and cracks and because they disturb automatic segmentation.

*Corresponding author: mayank.goswami@ph.iitr.ac.in

The practical difficulty is twofold. First, classical correction methods need hand-tuned parameters and tend to trade ring suppression against damage to real structure. Second, in real scans the ring-free image is unknown, so the error of a correction method cannot be measured directly. This study therefore pursues four concrete objectives:

1. **Ground-truth-verified synthetic benchmark.** Generate CT slices with a known ring component by simulating the whole measurement chain, with training and validation slices from one phantom family and test slices from a different, unseen family.
2. **Two-head U-Net surrogate.** Train a U-Net that predicts the ring image (to be subtracted from the input) and a ring mask, and compare it with a classical radial-profile filter whose two settings are tuned on validation data only.
3. **Sensitivity analysis.** Quantify the effect of ring type, training-set size, random seed, mask threshold and the Dice term of the loss.
4. **Honest characterisation of limits.** Report where the method fails, including the weak mask head and the lack of real-scan validation.

### 1.2 Why a Single Bad Pixel Becomes a Ring

Consider parallel-beam geometry. A detector pixel at lateral position $s$ always sees the line $x\cos\theta + y\sin\theta = s$ for projection angle $\theta$. If the pixel has a constant error, the same wrong value is added at every angle, which is a vertical stripe in the sinogram (the plot of detector position against angle). During back-projection each projection is smeared back along these lines. All lines belonging to position $s_0$ are tangent to the circle $x^2 + y^2 = s_0^2$, so the stripe is reconstructed as a ring of radius $|s_0|$ around the rotation centre. A classical filter has to infer from the image alone which circular structure is an artifact and which belongs to the object.

### 1.3 Overview in Plain Language

Imagine photographing a slowly spinning object with a camera that has one sensor pixel that reads slightly too bright. As the object turns, that pixel adds the same bright error to every photograph. When the photographs are combined into a cross-section, the error does not stay a point: it is spread into a circle. Our pipeline teaches a neural network to recognise such circles and to estimate exactly how bright each one is, so that the circle can be subtracted and the true cross-section remains. A second output of the network marks where the circles are. Because we generate the pictures ourselves, we know the true circles and can score the network and a classical filter on exactly the same images, including images from a different kind of object than the network was trained on.

## 2 Previous Work and Literature Review

### 2.1 Classical Ring Artifact Removal

The causes and appearance of ring artifacts in CT are described in overview articles by Barrett and Keat [2] and Boas and Fleischmann [3]. Most correction methods work on the sinogram, where a miscalibrated pixel is a stripe. Münch et al. combine wavelet decomposition with Fourier filtering to suppress stripes [5], and Vo et al. propose sorting-based and filtering-based techniques for micro-tomography [6]. Sijbers and Postnov address ring artifacts directly in high-resolution micro-CT reconstructions [4]. The baseline used here is in the same spirit as image-domain methods: it works on the reconstructed slice, summarises it in polar coordinates around the centre, and removes what stands out from a smoothed radial profile. Such methods need their parameters set by hand, and their behaviour on object structures that are themselves circular is a known weak point.

### 2.2 Learning-Based Image Restoration

The U-Net [7] is an encoder-decoder convolutional network with skip connections that was introduced for biomedical segmentation and is widely used for image-to-image tasks because it combines wide context with fine detail. Residual learning, in which the network predicts the degradation instead of the clean image, was popularised by deep residual networks [8] and applied to image denoising by DnCNN [9]. For segmentation of small regions, the Dice loss, as used in V-Net [10], directly optimises overlap and is robust against class imbalance. The present work combines these ingredients for ring removal and, unlike most studies on real data, evaluates every method against an exactly known ring component and against an unseen phantom family.

## 3 Theory and Synthetic CT Dataset Pipeline

### 3.1 Detector Error Model and Image Formation

Let $p(\theta, s)$ be the ideal parallel-beam projection of a phantom. The measured projection with detector gain error $g_s$ and offset error $b_s$ is

$$\tilde{p}(\theta, s) = g_s\, p(\theta, s) + b_s, \tag{1}$$

with $g_s = 1$ and $b_s = 0$ for a perfect pixel. The error $(g_s - 1)\, p(\theta, s) + b_s$ is constant in $\theta$ for a pure offset and varies slowly with $\theta$ for a gain error, so in both cases it forms a stripe at fixed $s$. FBP reconstructs the slice as

$$f(x, y) = \int_0^{\pi} (\tilde{q}_\theta * h)\,(x\cos\theta + y\sin\theta)\, d\theta, \tag{2}$$

where $h$ is the ramp filter, and, as shown in Section 1.2, a stripe at $s_0$ turns into a ring of radius $|s_0|$. Five ring types are simulated: `band`, `dead_hot`, `many_mild`, `mixed` and `sparse_strong`. The ring type of every test slice is stored in the metadata file written by the data generator and is used for the per-type analysis in Section 5.3.

### 3.2 Phantom Families and Measurement Chain

Each slice is $256 \times 256$ pixels. Training and validation slices use phantom family A and the test slices use family B, which is never seen in training. Family A consists of a few ellipses and discs of different grey levels inside a circular field of view, while family B has smooth irregular regions with small dark inclusions (Figure 1). For every phantom the pipeline (i) computes parallel-beam projections, (ii) adds Poisson noise, (iii) applies the gain and offset errors of Eq. (1) to selected pixels, and (iv) reconstructs by FBP twice, once from the corrupted and once from the clean projections.

### 3.3 Targets and Specimen-Level Partition

Let $x$ be the corrupted reconstruction and $y$ the clean reconstruction. The ring image is $r = x - y$, and the ring mask $m$ (stored with the data) is one where the ring artifact is strong. About 8% of all pixels are ring pixels. The dataset is partitioned as in Table 1. Because the test set comes from another phantom family, the test error measures generalisation and not memorisation. All random draws use a fixed seed (seed 0 for the main run).

## 4 Methodology: Workflow, Baseline, and Network

### 4.1 End-to-End Workflow

Figure 2 shows the workflow. Stage 1 simulates the measurement chain and produces the corrupted slice, the clean slice, the ring image and the ring mask. Stage 2 applies a method (no

Table 1: Dataset partition. Test slices come from an unseen phantom family.

| Split | Slices | Size (px) | Phantom family |
|---|---|---|---|
| Training | 1000 | $256 \times 256$ | A |
| Validation | 100 | $256 \times 256$ | A |
| Test | 100 | $256 \times 256$ | B (unseen) |

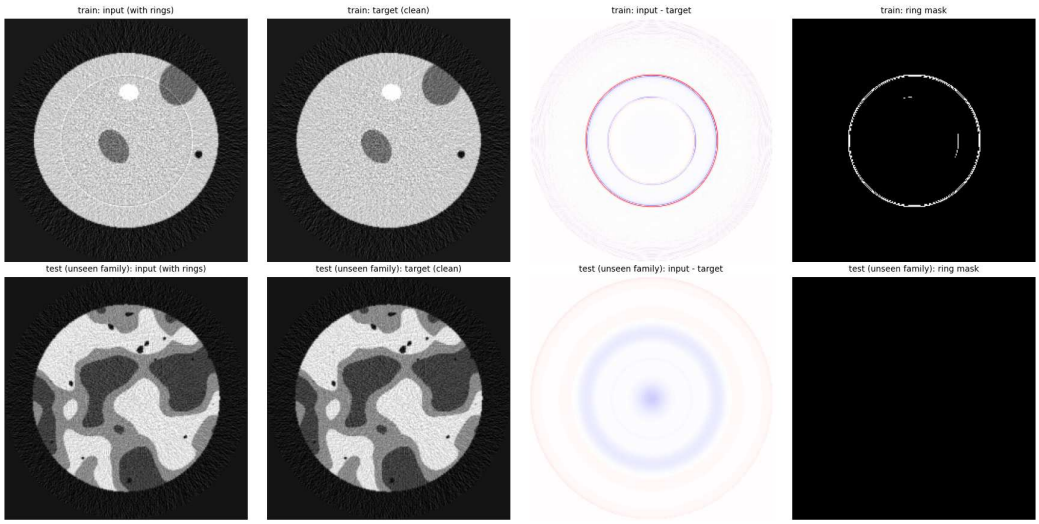



Figure 1: Data examples. Top row: a training slice. Bottom row: a test slice from the unseen family. Columns: input with rings, clean target, input minus target (the pure ring signal that the network must predict), and ring mask. In the bottom row the ring is faint and the mask is empty; for such images Dice is undefined and they are skipped when averaging Dice.

correction, classical filter or U-Net) to the corrupted slice. Stage 3 scores each method against the clean slice and the true mask with PSNR, SSIM and Dice.

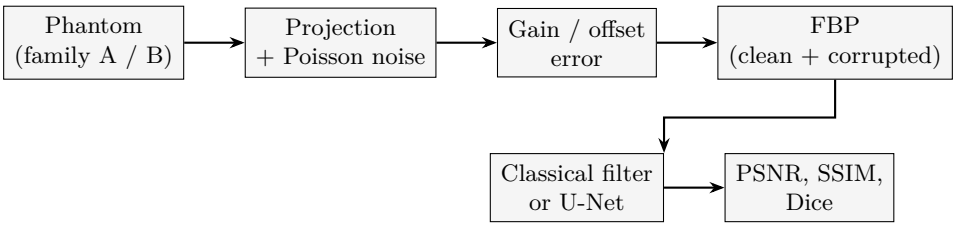



Figure 2: Workflow: synthetic data generation (top row), correction by a method, and scoring against the known truth.

### 4.2 Classical Radial-Profile Baseline

The baseline (function `remove_rings_classical`) uses the fact that a ring is constant along a circle. The slice is resampled to polar coordinates around its centre (radius $\times$ 720 angles, bilinear interpolation), and the median over angle of each circle gives a radial profile $P(\rho)$. A median filter of size *window* removes the smooth part of the profile, leaving the residual $e(\rho) = P(\rho) - \mathrm{med}_{\mathrm{window}}\, P(\rho)$, which contains the sharp spikes. With the median absolute deviation MAD of $e$, residual values are kept as ring contribution only if

$$|e(\rho)| < k \cdot 1.4826 \cdot \mathrm{MAD}(e), \tag{3}$$

where 1.4826 MAD is a robust estimate of the standard deviation. Larger spikes are set to zero because they are assumed to be real object edges and not rings. The remaining residual is interpolated back to a two-dimensional ring estimate $\hat{r}_c(x, y)$, and the corrected slice is $x - \hat{r}_c$. The parameter $k$ therefore sets, in units of a robust standard deviation, how large a spike may be and still be treated as a ring: a large $k$ accepts almost all spikes. The method needs no training. Its ring mask is $|\hat{r}_c| > 0.04\,DR$, where $DR$ is the dynamic range of the clean slice. The notebook defaults are window $= 5$, $k = 6$. I tuned both settings on the validation set only, by maximising mean PSNR over a grid, and then evaluated the chosen pair once on the test set. The first grid (window $\in \{3, 5, 9, 15, 25\}$, $k \in \{3, 6, 10\}$) had its best value at window 25, $k = 10$ (34.57 dB). The second grid (Table 2) found the best pair at window 25, $k = 30$ (34.72 dB). The best $k$ lies at the edge of the grid, but the gain per step had dropped to about 0.05 dB, so the search was stopped there; extending it after seeing test results would have used the test set for tuning. A $k$ of 30 means that almost no spike is excluded, so a plausible reason for the poor transfer to the unseen family is that the filter then also subtracts real circular structure.

Table 2: Second tuning grid: mean validation PSNR (dB) of the classical baseline. Input PSNR on validation is 33.95 dB; the default setting (5, 6) gives 34.09 dB.

| Window | $k = 10$ | $k = 15$ | $k = 20$ | $k = 30$ |
|---|---|---|---|---|
| 25 | 34.57 | 34.63 | 34.67 | 34.72 |
| 41 | 34.38 | 34.42 | 34.46 | 34.52 |
| 61 | 33.90 | 33.89 | 33.87 | 33.79 |
| 81 | 33.36 | 33.11 | 32.35 | 32.01 |

### 4.3 Two-Head U-Net

The network (class `UNet`) takes the corrupted slice $x$ ($1 \times 256 \times 256$) and has four resolution levels with 32, 64, 128 and 256 channels (base width 32). Each block consists of two $3 \times 3$ convolutions, each followed by batch normalisation [11] and ReLU. The encoder has three such blocks, each followed by $2 \times 2$ max pooling, and a bottleneck block at the lowest resolution. The decoder upsamples with $2 \times 2$ transposed convolutions, concatenates the matching encoder feature map (skip connection) and applies a block at each level. A final $1 \times 1$ convolution produces two output channels. The model has 1,927,874 trainable parameters. Head 1 (the first output channel) predicts the ring image $\hat{r}$ and the corrected slice is

$$\hat{y} = x - \hat{r}. \tag{4}$$

Predicting only the artifact is residual learning [9]: the ring is small compared with the whole image, so it is easier to learn, and the object structure in $x$ is left untouched. Head 2 (the second channel) outputs mask logits $\hat{m}$; a sigmoid and a threshold of 0.5 give the binary mask.

### 4.4 Loss, Optimisation and Training Specifications

The training loss combines a pixel-wise term for the ring image and two terms for the mask:

$$\mathcal{L} = 10\,\|\hat{r} - r\|_1 + \mathrm{BCE}(\hat{m}, m) + \left[1 - \frac{2\sum \sigma(\hat{m})\, m + \varepsilon}{\sum \sigma(\hat{m}) + \sum m + \varepsilon}\right], \tag{5}$$

where $\sigma$ is the sigmoid and $\varepsilon$ a small constant. The factor 10 balances the scales of the three terms. The Dice term is needed because ring pixels are only about 8% of the image, so an all-zero mask would already score well under cross-entropy alone. Table 3 lists all settings. Mixed-precision training [14] was used on the GPU. The weights with the lowest validation loss were kept. The run used PyTorch [18].

Table 3: Training specifications of the main run.

| Setting | Value |
|---|---|
| Optimiser | AdamW [12], learning rate $10^{-3}$, weight decay $10^{-4}$ |
| Schedule | OneCycleLR [13] |
| Batch size, epochs | 8, 60 |
| Augmentation | random rotation by a multiple of 90°, horizontal flip with probability 0.5 |
| Precision, hardware | mixed precision (CUDA), Colab T4 GPU |
| Run time | 681 s (about 11 min) for 60 epochs |
| Checkpoint | lowest validation loss |
| Random seed | 0 (main run); 1 and 2 for repeats |

### 4.5 Evaluation Metrics and Protocol

Three methods are scored on identical images: Input (no correction), Classical and U-Net. With dynamic range $DR = \max y - \min y$ of the clean slice,

$$\mathrm{PSNR} = 10 \log_{10}\left(\frac{DR^2}{\mathrm{MSE}(\hat{y}, y)}\right), \qquad \mathrm{Dice} = \frac{2\,|\hat{M} \cap M|}{|\hat{M}| + |M|}, \tag{6}$$

and SSIM follows Wang et al. [15]. A gain of about 6 dB in PSNR halves the error amplitude. Dice is undefined when both the true and the predicted mask are empty; such images are skipped in the average. Input has no mask, so its Dice is not a number. PSNR and SSIM are computed with scikit-image [16] using the dynamic range $DR$, and image resampling and median filtering use SciPy [17]. The mask threshold was chosen on validation data and the checkpoint was chosen by validation loss, so the test set was used only for final reporting. Additional experiments vary the training-set size (100, 500, 1000 images, 60 epochs each, same seed), the random seed (0, 1, 2), the Dice term (removed) and the mask threshold (0.3 to 0.7).

## 5 Experimental Results and Critical Analysis

### 5.1 Training Behaviour

Figure 3 shows the loss curves of the main run. Both losses fall quickly during the first ten epochs and then flatten. Validation starts below training (1.97 versus 2.63), probably because the training value is averaged over an epoch while the model is still improving quickly. Small validation bumps (for example near epochs 10 and 31) recover within a few epochs, and there is no divergence. The final losses are 0.571 (training) and 0.613 (validation), so the gap is small and there is no clear sign of overfitting. The curves are still falling slowly at epoch 60, so a longer run might help slightly; this was not tested.

### 5.2 Experiment 1: Main Benchmark

Table 4 gives the main result. On the unseen test family the input has a PSNR of 33.11 dB. The U-Net reaches 39.54 dB (gain 6.43 dB) and raises SSIM from 0.868 to 0.957 (gain 0.089). The classical baseline tuned on validation gains 0.77 dB there, but loses 0.56 dB on the test family (32.55 dB), although its SSIM rises slightly (by 0.011). With the default settings the baseline had changed test PSNR by +0.18 dB, so the loss is not caused by the tuning procedure alone: settings chosen on the training family did not carry over to the unseen family. The U-Net loses only 1.27 dB from validation to test. Dice of the classical estimate is 0.154 (validation) and 0.218 (test); the U-Net reaches 0.668 and 0.640.

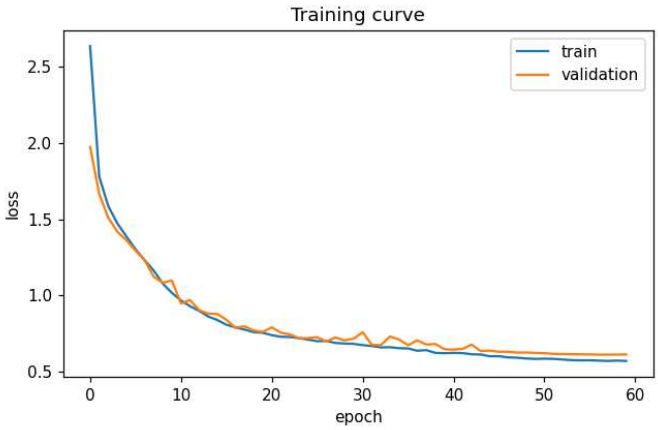



Figure 3: Training and validation loss of the main run over 60 epochs.

Table 4: Experiment 1: validation and test results. Input has no mask, so Dice is not applicable (n/a).

| | Validation (family A) | | | Test (family B, unseen) | | |
|---|---|---|---|---|---|---|
| Method | PSNR (dB) | SSIM | Dice | PSNR (dB) | SSIM | Dice |
| Input | 33.95 | 0.8702 | n/a | 33.11 | 0.8678 | n/a |
| Classical, default (5, 6) | 34.09 | 0.8735 | 0.003 | 33.29 | 0.8723 | 0.005 |
| Classical, tuned (25, 30) | 34.72 | 0.8877 | 0.154 | 32.55 | 0.8783 | 0.218 |
| **U-Net** | **40.81** | **0.9591** | **0.668** | **39.54** | **0.9565** | **0.640** |

### 5.3 Result by Ring Type

Table 5 splits the test set by ring type. The U-Net gain follows the damage: it is largest where the input is worst (`dead_hot`, from 20.8 to 34.7 dB) and small where the input is already almost clean (`band`, 45.9 to 46.3 dB). The tuned classical filter degrades `band` images by 3.6 dB, which explains why its overall test PSNR falls below the input, and helps `mixed` by 0.9 dB. The mask Dice of the U-Net lies between 0.63 and 0.81 on four ring types but is only 0.066 on `band`, where faint rings leave very few mask pixels close to the noise level.

Table 5: Test results by ring type (PSNR in dB, tuned classical baseline). Gain is U-Net minus Input.

| Ring type | $n$ | Input | Classical | U-Net | Gain | Dice cl. | Dice U-Net |
|---|---|---|---|---|---|---|---|
| `band` | 21 | 45.91 | 42.32 | 46.32 | +0.41 | 0.001 | 0.066 |
| `dead_hot` | 17 | 20.80 | 21.15 | 34.69 | +13.89 | 0.302 | 0.813 |
| `many_mild` | 26 | 34.02 | 34.01 | 38.39 | +4.37 | 0.157 | 0.631 |
| `mixed` | 13 | 28.55 | 29.40 | 35.78 | +7.23 | 0.319 | 0.724 |
| `sparse_strong` | 23 | 32.07 | 32.20 | 40.36 | +8.29 | 0.261 | 0.798 |

### 5.4 Visual Comparison

Figure 4 shows four test slices. In row 1 the classical result keeps the small central ring, which the U-Net removes; its mask finds the centre and the inner circles but only part of the outer ones. In row 2 the classical filter adds faint circular halos that were not in the input, and the U-Net image is clean, but the U-Net mask almost misses the central disc: the image correction

can succeed while the mask fails. In row 3 the classical method keeps the thin ring and the dark central dot, the U-Net removes both and the masks agree well. In row 4 both strong rings survive the classical method, which also leaves a dark smear near the centre; the U-Net removes them, and its mask follows the thick rings but adds speckle outside them.

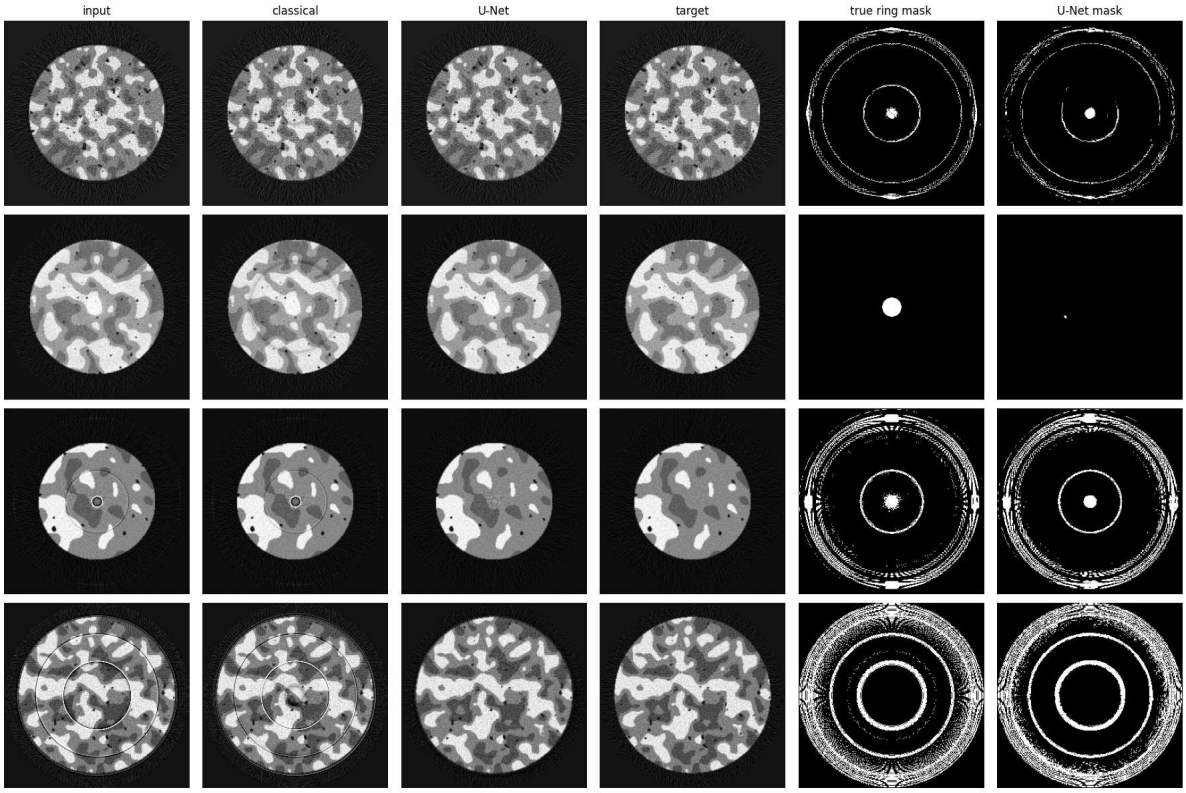



Figure 4: Four test slices (rows). Columns: input, classical (tuned), U-Net, clean target, true ring mask, U-Net mask.

### 5.5 Effect of Training-Set Size

Table 6 shows that more data helps, with diminishing returns. Even 100 images give +2.18 dB over the input, more than the tuned classical filter manages on the test family. Going from 100 to 500 images adds 3.13 dB, and from 500 to 1000 adds 1.12 dB. Every run used 60 epochs, so smaller sets also saw fewer optimisation steps, and the curve is still rising at 1000 images.

Table 6: Test performance versus number of training images.

| Training images | PSNR (dB) | SSIM | Dice | Training time |
|---|---|---|---|---|
| 100 | 35.29 | 0.9039 | 0.371 | 116 s |
| 500 | 38.42 | 0.9422 | 0.560 | 366 s |
| 1000 | 39.54 | 0.9565 | 0.640 | 681 s |

### 5.6 Seed Variation and Loss Ablation

With seeds 0, 1 and 2 the test PSNR is 39.54, 39.24 and 39.42 dB, so the mean is 39.40 dB with a sample standard deviation of 0.15 dB. This spread is small compared with the 6.3 dB gain over the input. Dice and SSIM were not recorded for the seed repeats, so their spread is unknown. Without the Dice term (Table 7) PSNR rises from 39.54 to 39.89 dB, but mask Dice falls from

0.640 to 0.605, a trade-off. Both are single runs with one seed each, so only the trend should be read from this comparison. Validation loss values of the two runs are not comparable because the loss terms differ.

Table 7: Seeds (left) and Dice-term ablation (right), test set.

| Seed | PSNR (dB) | Loss | PSNR (dB) | SSIM | Dice |
|---|---|---|---|---|---|
| 0 | 39.54 | With Dice (seed 0) | 39.54 | 0.9565 | 0.640 |
| 1 | 39.24 | Without Dice (seed 0) | 39.89 | 0.9602 | 0.605 |
| 2 | 39.42 | | | | |
| Mean ± SD | $39.40 \pm 0.15$ | | | | |

### 5.7 Mask Threshold

The mask threshold was swept on the validation set (Table 8). The curve is almost flat, and 0.6 beats 0.5 by only 0.006, so 0.5 was kept. The mask problem is therefore not a threshold problem but a ring-type problem: faint rings leave almost no mask, as the Dice of 0.066 on `band` and row 2 of Figure 4 show.

Table 8: Validation Dice of the U-Net mask for different thresholds.

| Threshold | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 |
|---|---|---|---|---|---|
| Dice | 0.657 | 0.660 | 0.668 | 0.674 | 0.668 |

## 6 Conclusions and Practical Guidelines

A two-head U-Net trained on simulated CT slices removes ring artifacts far better than a radial-profile classical baseline and generalises to a phantom family not seen in training (PSNR 33.11 to 39.54 dB, SSIM 0.868 to 0.957). Three guidelines follow from the experiments.

1. **Expect gains in proportion to the damage.** Severe rings (`dead_hot`, `sparse_strong`) gain 8 to 14 dB, faint ones very little. A learned corrector is most useful where a classical filter does least.

2. **Do not trust classical settings tuned on one family.** Settings that gained 0.77 dB on validation lost 0.56 dB on the unseen family, and the filter damaged almost-clean images.

3. **Use the image correction, not yet the mask.** The ring-image output is reliable; the mask output (Dice 0.640) misses faint rings and should not drive automatic decisions.

## 7 Limitations and Inherent Algorithmic Biases

### 7.1 Methodological and Dataset Limitations

1. **Simulation only.** All data use parallel-beam geometry, a simple gain and offset error model and 2D slices. Real scanners use fan-beam or cone-beam geometry and have other detector effects such as nonlinear response, afterglow and drift. The sim-to-real gap is untested.

2. **Baseline tuning.** The classical baseline was tuned over a small grid on validation data, and its best $k$ lies at the edge of the grid. A different or stronger classical method might do better; there is no evidence it would approach the 6 dB gain of the U-Net.

3. **Small sample.** There are 100 test slices and three seeds, with no confidence intervals. Differences of a few tenths of a dB, such as the 0.5 dB `band`-ring gain or the 0.35 dB effect of the Dice term, should not be over-interpreted.

4. **Dice protocol.** Dice skips images in which both masks are empty, and the seed-to-seed spread of Dice was not measured.

5. **Two phantom families only.** Generalisation to other object types, noise levels and ring strengths was not tested.

6. **Limited training schedule.** Training used 60 epochs with the loss still falling slowly, and no hyperparameter search was made for the U-Net.

### 7.2 Algorithmic Biases and Responsible Use

A network trained on simulated rings has learned the simulator's ring statistics. If applied uncritically to real scans it may remove real circular structures (for example the outline of a cylindrical sample holder) or leave rings of a type it has never seen. Because the output is a subtracted correction, the risk of inventing structure is lower than for a network that outputs the full image, but it cannot be excluded. Results should be checked against physical validation scans before use in quantitative work.

## Data and Code Availability

The notebook, the saved model weights, the result tables and the figures are openly available at: Google Drive folder.

## Video

https://youtu.be/eUmfp4iJpwc

## References

## References

[1] Kak, A. C., & Slaney, M. (2001). *Principles of Computerized Tomographic Imaging*. SIAM.

[2] Barrett, J. F., & Keat, N. (2004). Artifacts in CT: recognition and avoidance. *RadioGraphics*, 24(6), 1679–1691.

[3] Boas, F. E., & Fleischmann, D. (2012). CT artifacts: causes and reduction techniques. *Imaging in Medicine*, 4(2), 229–240.

[4] Sijbers, J., & Postnov, A. (2004). Reduction of ring artefacts in high resolution micro-CT reconstructions. *Physics in Medicine and Biology*, 49(14), N247–N253.

[5] Münch, B., Trtik, P., Marone, F., & Stampanoni, M. (2009). Stripe and ring artifact removal with combined wavelet–Fourier filtering. *Optics Express*, 17(10), 8567–8591.

[6] Vo, N. T., Atwood, R. C., & Drakopoulos, M. (2018). Superior techniques for eliminating ring artifacts in X-ray micro-tomography. *Optics Express*, 26(22), 28396–28412.

[7] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. *MICCAI*, LNCS 9351, 234–241.

[8] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *CVPR*, 770–778.

[9] Zhang, K., Zuo, W., Chen, Y., Meng, D., & Zhang, L. (2017). Beyond a Gaussian denoiser: residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, 26(7), 3142–3155.

[10] Milletari, F., Navab, N., & Ahmadi, S.-A. (2016). V-Net: Fully convolutional neural networks for volumetric medical image segmentation. *International Conference on 3D Vision*, 565–571.

[11] Ioffe, S., & Szegedy, C. (2015). Batch normalization: accelerating deep network training by reducing internal covariate shift. *ICML*, 448–456.

[12] Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. *ICLR*.

[13] Smith, L. N., & Topin, N. (2019). Super-convergence: very fast training of neural networks using large learning rates. *arXiv:1708.07120*.

[14] Micikevicius, P., et al. (2018). Mixed precision training. *ICLR*. arXiv:1710.03740.

[15] Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4), 600–612.

[16] van der Walt, S., et al. (2014). scikit-image: image processing in Python. *PeerJ*, 2, e453.

[17] Virtanen, P., et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3), 261–272.

[18] Paszke, A., et al. (2019). PyTorch: an imperative style, high-performance deep learning library. *NeurIPS*, 32.