Distributed Deep Learning for Medical Image Denoising using U-Net and U-Net++

25th IEEE International Conference on BioInformatics and BioEngineering

[1]Sulaimon Oyeniyi Adebayo and [1,2]Ayaz H. Khan

[1]Computer Engineering Department, King Fahd University of Petroleum and Minerals (KFUPM), Saudi Arabia.
[2]SDAIA-KFUPM Joint Research Center for Artificial Intelligence, KFUPM, Saudi Arabia

November, 2025

OUTLINE

- INTRODUCTION
- PROBLEM
- BACKGROUND
- METHODOLOGY
- RESULTS AND DISCUSSION
- CONCLUSION AND FUTURE WORK

Distributed Deep Learning:

A subset of machine learning that involves training deep neural networks across multiple machines in parallel.

Distributed deep learning improves efficiency in the following ways;

- Huge Datasets
- Flexibility
- Large Models
- Resource Utilization
- Scalability

Credit For this Slide: Dr. Jagreet Kaur Gill (XENONSTACK)

INTRODUCTION
Traditional & Distributed Training

Traditional Training

- ✓ Involves training a model on a single machine with one or more GPUs or CPUs.
- ✓ The entire dataset is loaded onto a single node's memory.
- ✓ Training is sequential, and resources (CPU, RAM, GPU) on a single machine limit scalability.

✅ **Simplicity**
✅ **No Communication Overhead**
✅ **Efficient for Small Models**

❌ **Limited by Hardware**
❌ **Slow for Large Models**
❌ **Not Scalable.**

Distributed Training

- ✓ Involves multiple machines (nodes) working together to train a model.
- ✓ Used when a single machine is insufficient due to memory or compute demand/limitations.
- ✓ Requires a communication strategy to synchronize updates across nodes.

✅ **Scalability**
✅ **Faster Training**
✅ **Better Utilization**

❌ **Communication Overhead**
❌ **Complex Implementation**
❌ **Debugging Challenges.**

PROBLEM

- Centralized medical image training raises privacy and computational concerns.

- Hospitals and research centers face restrictions on data sharing due to patient confidentiality.

- Deep learning denoising requires large datasets but centralizing sensitive data is risky.

- Need for a distributed, privacy-conscious, and efficient denoising framework that preserves diagnostic quality.

BACKGROUND

Deep Learning

- ✓ Deep learning (U-Net, U-Net++) has improved medical image segmentation and denoising.
- ✓ Distributed Deep Learning (DDL) enables training across GPUs/nodes to reduce computation time.

Privacy-Preserving ML

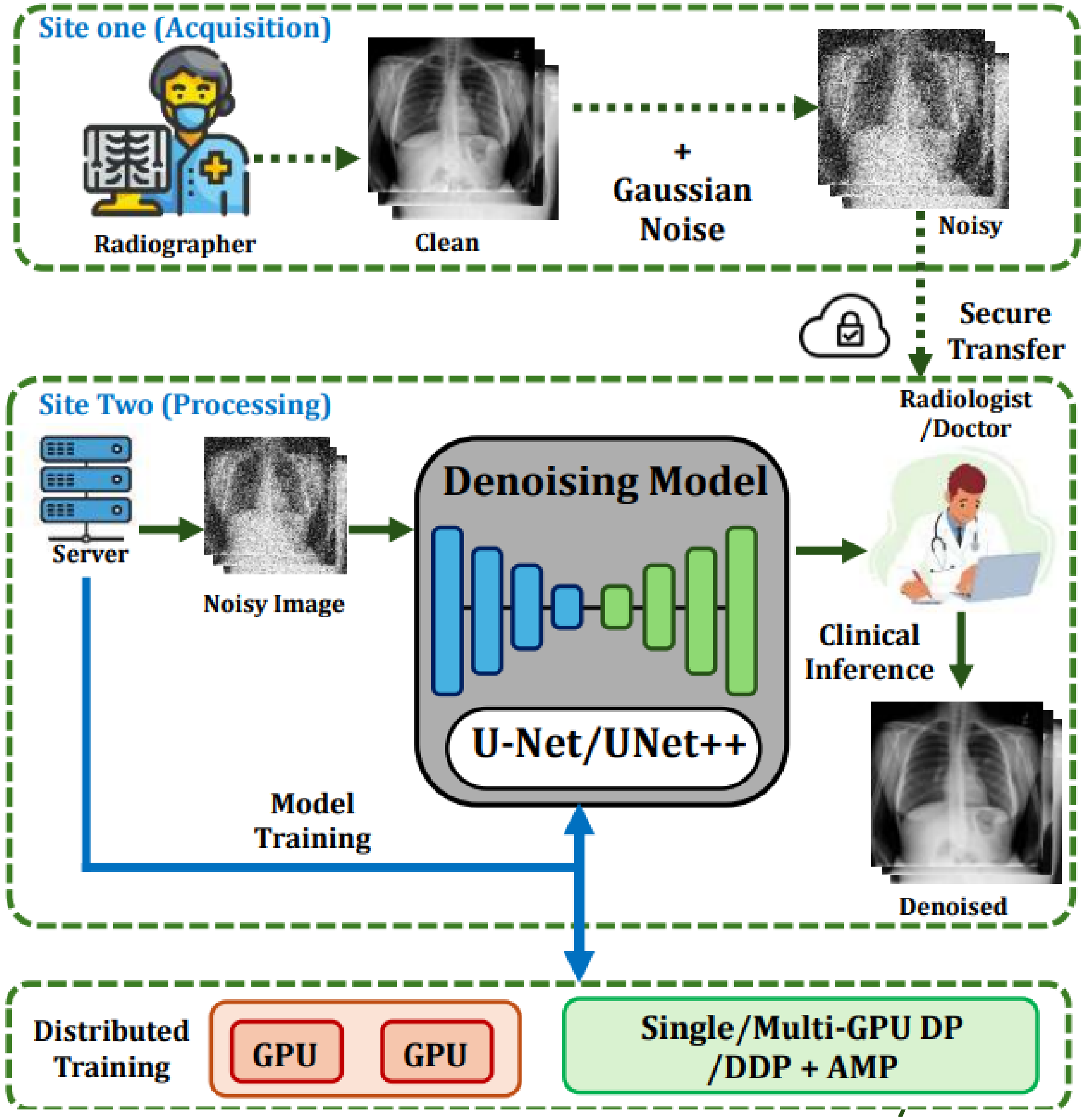
- ✓ Federated Learning, Differential Privacy, Encryption → high communication cost.
- ✓ Gaussian noise obfuscation → lightweight privacy proxy with minimal overhead.

Existing denoising filters (BM3D, TNDR, DnCNN-3) underperform compared to CNN-based methods.

METHODOLOGY

Framework Overview

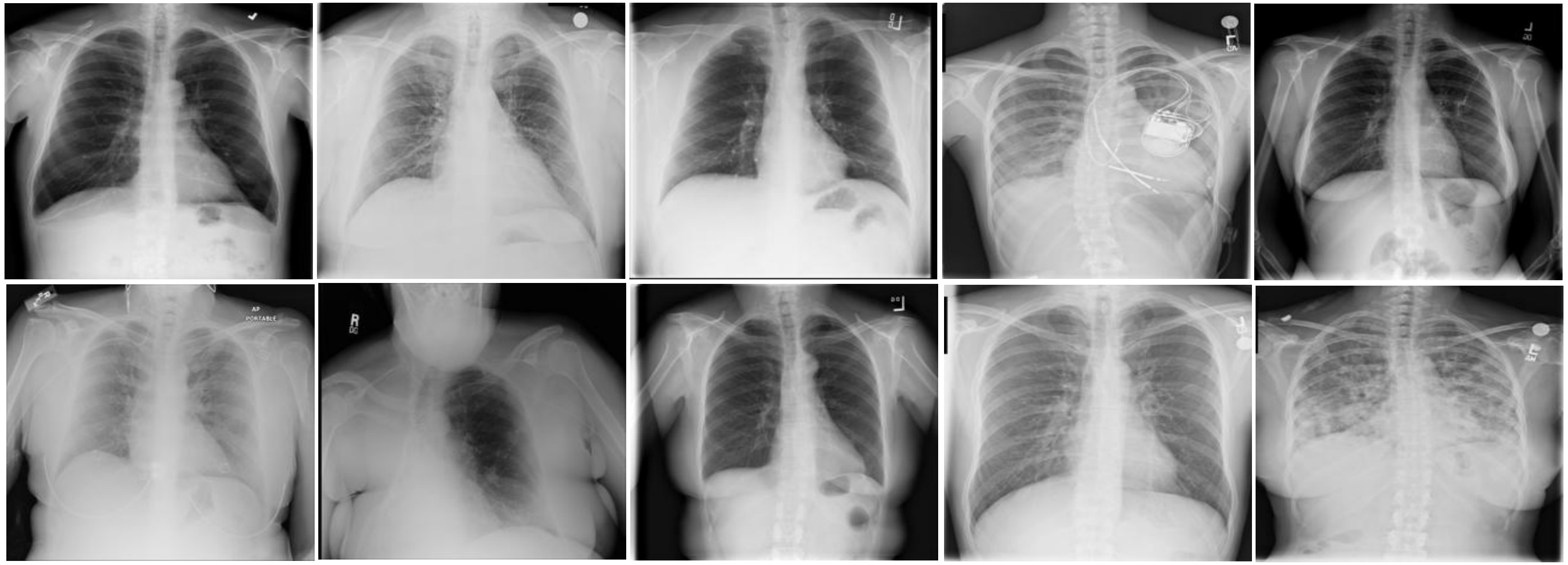
- ❖ Site 1 (Acquisition):
  Radiographer adds Gaussian noise to images → secure transfer.
- ❖ Site 2 (Processing):
  Denoising model (U-Net/U-Net++) trained on noisy-clean pairs.



METHODOLOGY
Dataset

- Chest radiographs from the NIH Chest X-ray14 dataset.
- Consists of 112,120 frontal chest X-ray images collected from 30,805 unique patients
- We sampled 15,000 images for our study.
- All images already in grayscale (single-channel) and 1024 x 1024 were resized to 256×256 pixels to standardize the input dimensions and reduce computational costs.

Samples from the dataset



SUMMARY OF U-NET AND U-NET++ ARCHITECTURES USED

Feature	U-Net	U-Net++
Input	$1 \times 256 \times 256$ grayscale image	$1 \times 256 \times 256$ grayscale image
Output	$1 \times 256 \times 256$ denoised image	$1 \times 256 \times 256$ denoised image
Depth	5 levels	5 levels with nested decoding
Param.	~ 8.6 M ($N_c = 64$)	~ 9.2 M (`base_ch=64`)
Key Features	ReLU, BatchNorm, symmetric skip connections, transposed conv upsampling	Dense skip pathways, multi-depth aggregation, multiple output heads, bilinear upsampling

Training Framework and Hardware Configuration

Training Framework and Hardware Configuration

Attribute	Details
Framework	PyTorch 1.X with `nn.DataParallel`
GPUs Used	2 $\times$ NVIDIA RTX A4500 (Each with 20470 MiB memory)
CUDA Version	12.2
Driver Version	535.171.04
GPU Utilisation	Managed by PyTorch's automatic batch splitting and gradient synchronization

METHODOLOGY
Key innovations

- Uses Gaussian noise as obfuscation instead of heavy cryptography/federated setup.
- Employs PyTorch DistributedDataParallel (DDP) + Automatic Mixed Precision (AMP) for scalability.
- Benchmarks U-Net vs U-Net++ under single-GPU, multi-GPU, and DDP setups.
- Public implementation released for reproducibility.

RESULTS AND DISCUSSION

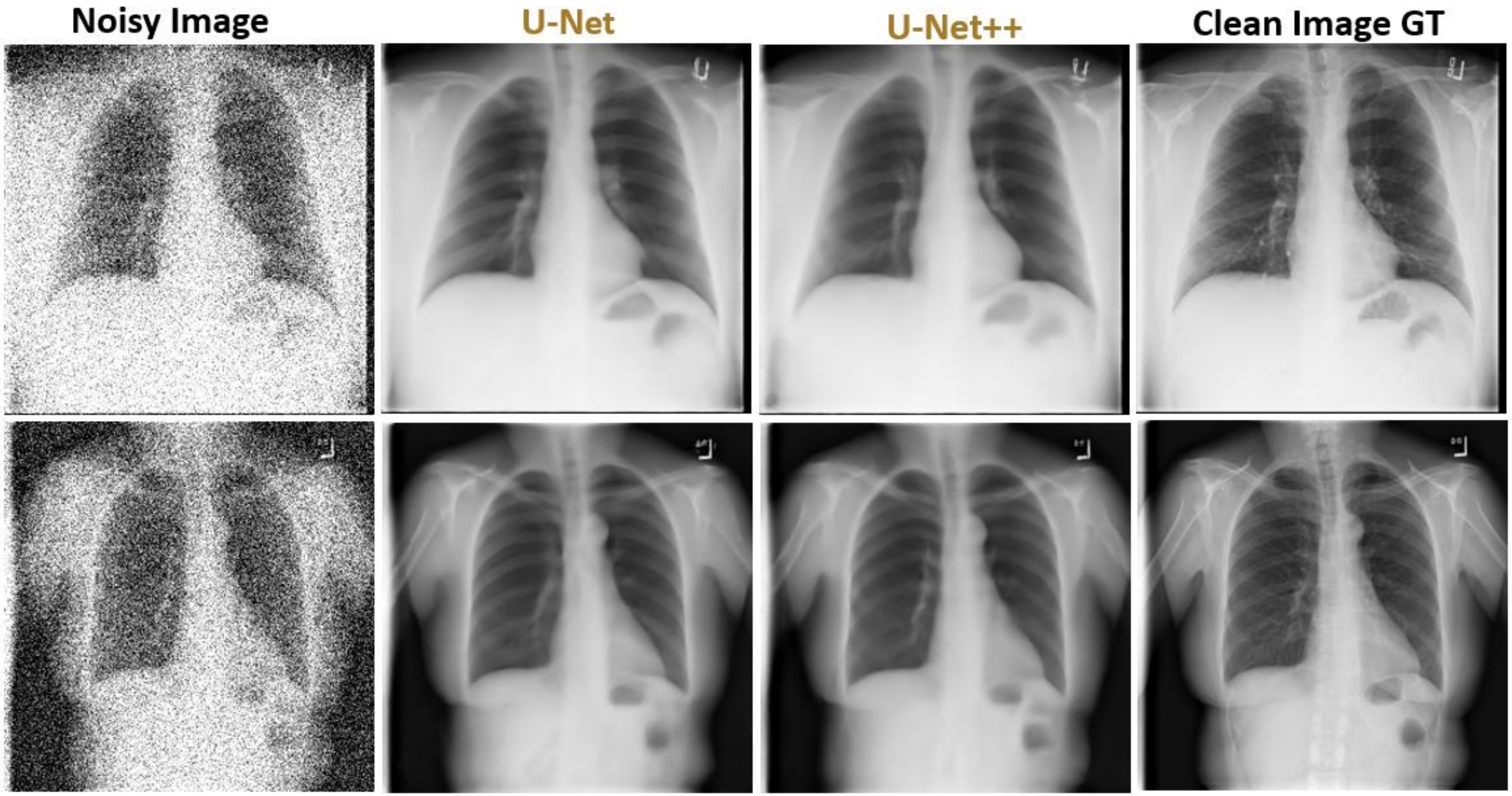
Quantitative

- ✓ U-Net++ > U-Net in PSNR/SSIM at 20–30% noise (better structure).
- ✓ U-Net > U-Net++ in LPIPS at 10% noise (better perceptual quality).

Training Speed

- ✓ DDP + AMP reduces training time by > 60 % vs single-GPU.

RESULTS AND DISCUSSION



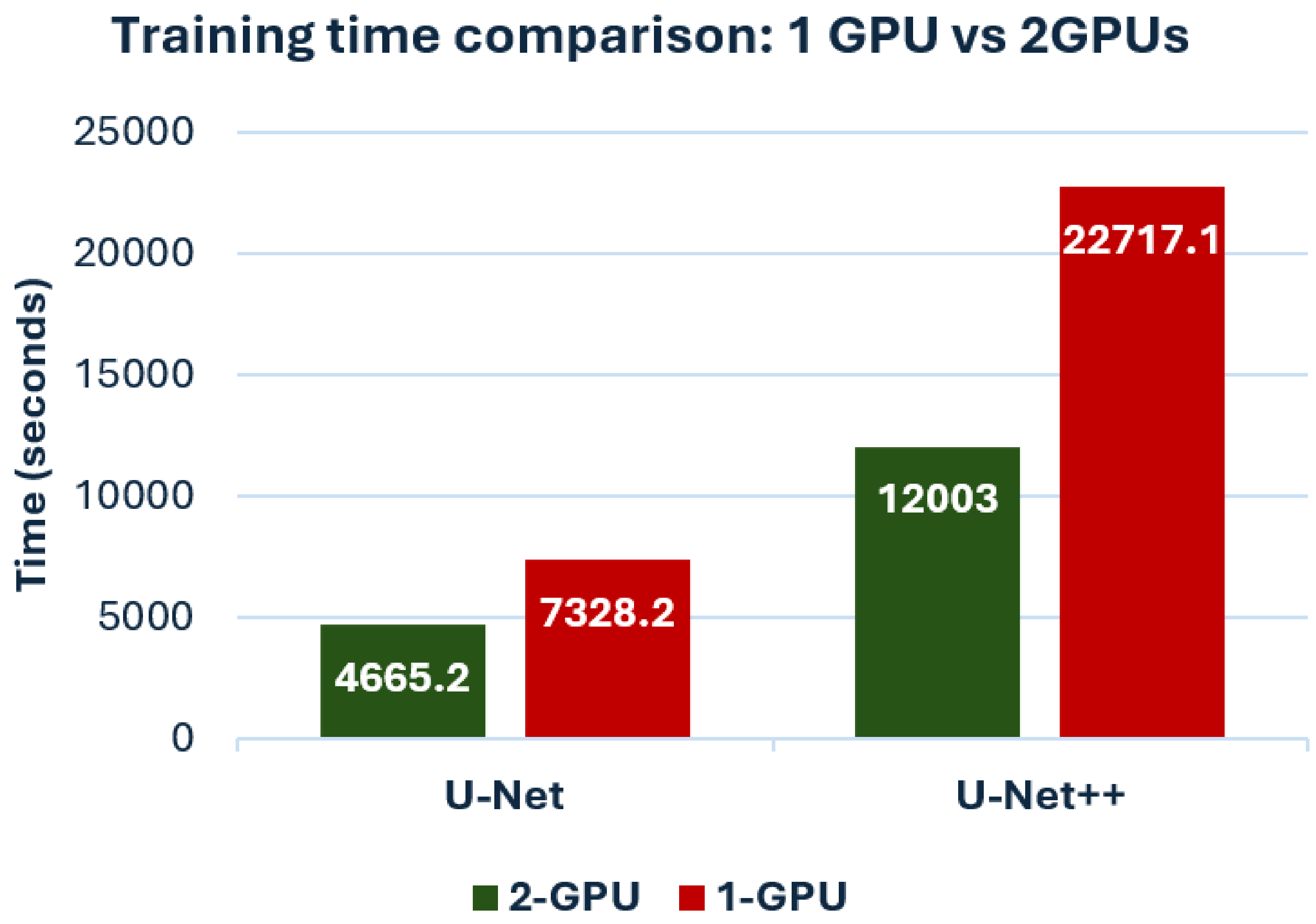
Representative qualitative comparison at 20% Gaussian noise.

RESULTS AND DISCUSSION

Noise	Model	Setup	PSNR (dB)	SSIM	LPIPS	TT (s)	TS (%)
10%	U-Net	1 GPU	34.95 ± 0.04	0.9168 ± 0.0008	0.1373 ± 0.0010	7328.2	0.00%
10%	U-Net	2 GPUs (DP)	34.034	0.9060	0.1582	4665.2	36.34%
10%	U-Net	2 GPUs (DDP+AMP)	**34.483**	**0.9067**	**0.1562**	**2737.0**	**62.68%**
10%	U-Net++	1 GPU	34.39 ± 0.05	0.9123 ± 0.0007	0.1585 ± 0.0011	22717.1	0.00%
10%	U-Net++	2 GPUs (DP)	34.089	0.9083	0.1564	12003.0	47.16%
10%	U-Net++	2 GPUs (DDP+AMP)	**33.416**	**0.8927**	**0.2175**	**8025.0**	**64.67%**
20%	U-Net	1 GPU	32.26 ± 0.04	0.8907 ± 0.0011	0.2010 ± 0.0013	7326.1	0.00%
20%	U-Net	2 GPUs (DP)	32.157	0.8906	0.2143	4648.1	36.55%
20%	U-Net++	1 GPU	32.32 ± 0.05	0.8959 ± 0.0010	0.2265 ± 0.0015	22266.0	0.00%
20%	U-Net++	2 GPUs (DP)	32.151	0.8886	0.2000	11998.5	46.11%
30%	U-Net	1 GPU	30.23 ± 0.05	0.8746 ± 0.0012	0.2498 ± 0.0016	7322.0	0.00%
30%	U-Net	2 GPUs (DP)	30.481	0.8757	0.2661	4648.2	36.52%
30%	U-Net++	1 GPU	30.76 ± 0.05	0.8840 ± 0.0011	0.2479 ± 0.0016	22343.5	0.00%
30%	U-Net++	2 GPUs (DP)	30.691	0.8830	0.2638	12015.6	46.22%

Performance (± 95% confidence interval (ci) for 1-gpu only), training time, and speedup across setups. TT = training time (s). TS = time saving (%) relative to the corresponding 1-gpu. DDP+AMP metrics/time are reported for 10% noise only.

RESULTS AND DISCUSSION
Training Time Comparison



RESULTS AND DISCUSSION
Comparison

DENOISING PERFORMANCE COMPARISON (PSNR/SSIM) ACROSS METHODS AND GAUSSIAN NOISE LEVELS.

Method	Noise Level	PSNR/SSIM	Noise Level	PSNR/SSIM
OURS (U-Net++)	20%	32.32/0.8959	30%	30.76/0.8840
OURS (U-Net)	20%	32.26/0.8907	30%	30.23/0.8746
BM3D [20]	15%	31.08/0.8722	25%	28.57/0.8017
TNDR [21]	15%	31.42/0.8826	25%	28.92/0.8157
DnCNN-3 [22]	15%	31.46/0.8826	25%	29.02/0.8190

❖Our models outperform traditional denoisers even at higher noise levels.

❖Demonstrates robustness and scalability of our models under distributed settings.

CONCLUSION

❖ Distributed Deep Learning enables efficient, privacy-aware medical image denoising.

❖ U-Net++ shows higher structural fidelity under heavy noise; U-Net performs better at low noise.

❖ DDP + AMP cuts training time by ≈ 60 % with minimal quality loss.

❖ The framework is scalable, secure, and clinically viable for real-world deployment.

Future Work

❖ Add attention and perceptual losses.
❖ Include radiologist-in-loop evaluation.
❖ Extend to federated learning for multi-site training.

THANK YOU

REFERENCES

M. A. Abdou, “Literature review: Efficient deep neural networks techniques for medical image analysis,” Neural Computing and Applications, vol. 34, no. 8, pp. 5791–5812, 2022.

M. Lai, “Deep learning for medical image segmentation,” arXiv preprint arXiv:1505.02000, 2015.

O. Aouedi, A. Sacco, K. Piamrat, and G. Marchetto, “Handling privacysensitive medical data with federated learning: challenges and future directions,” IEEE Journal of Biomedical and Health Informatics, vol. 27, no. 2, pp. 790–803, 2022.

F. Russo, “A method for estimation and filtering of gaussian noise in images,” IEEE Transactions on Instrumentation and Measurement, vol. 52, no. 4, pp. 1148–1154, 2003.

O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in International Conference on Medical image computing and computer-assisted intervention, pp. 234– 241, Springer, 2015.