Distributed Deep Learning for Medical Image Denoising with Data Obfuscation

Sulaimon Oyeniyi Adebayo
Computer Engineering Department,
King Fahd University of Petroleum and Minerals,
Dhahran, 31261, Saudi Arabia.
g202203440@kfupm.edu.sa

Ayaz H. Khan
Computer Engineering Department and
SDAIA-KFUPM Joint Research Center for Artificial Intelligence,
King Fahd University of Petroleum and Minerals,
Dhahran 31261, Saudi Arabia
ayaz.khan@kfupm.edu.sa

***Abstract*—Medical image denoising enhances diagnostic quality while enabling privacy-aware training on sensitive clinical datasets. This study presents a distributed deep learning framework for denoising chest X-rays from the NIH ChestX-ray14 dataset, using additive Gaussian noise as a lightweight obfuscation technique. We implement and evaluate U-Net and U-Net++ architectures under three training setups: single-GPU, multi-GPU (DataParallel), and optimized multi-GPU using PyTorch's DistributedDataParallel (DDP) with Automatic Mixed Precision (AMP). Results show that U-Net++ consistently achieves superior structural fidelity (Peak Signal to Noise Ratio and Structured Similarity Index Method), while U-Net performs better in perceptual similarity (Learned Perceptual Image Patch Similarity) at lower noise levels. Our optimized DDP+AMP pipeline reduces training time by over 60% compared to single-GPU and over 40% versus DataParallel, with minimal accuracy trade-off. This work demonstrates the practical viability of combining model design, noise-based obfuscation, and software-level acceleration to build scalable, privacy-conscious medical imaging pipelines. The full implementation is publicly available at: https://github.com/Suadey/medical-image-denoising-ddp.**

***Index Terms*—Distributed Deep Learning, Medical image denoising, Data Obfuscation, U-Net, U-Net++, DistributedDataParallel, Mixed Precision Training, Multi-GPU Optimization.**

I. Introduction

Deep learning has significantly advanced medical image analysis, particularly in segmentation and denoising tasks using convolutional neural networks (CNNs) [1]. However, training large-scale models on centralized medical datasets raises certain concerns, such as high computational power demand [2] and privacy [3]. Distributed Deep Learning (DDL) offers a promising alternative, allowing training across multiple GPUs or sites while reducing data centralization.

This study focuses on medical image denoising using chest X-ray images from the NIH ChestX-ray14 dataset [1]. To simulate privacy-preserving data sharing, we apply additive Gaussian noise [4] as a lightweight obfuscation technique. Noisy images are used as model inputs, while clean images serve as targets, mimicking a real-world scenario where sensitive data is masked before training.

The authors would like to acknowledge all support provided by King Fahd University of Petroleum and Minerals (KFUPM).

[1]NIH Chest X-rays

We investigate how distributed training configurations—including single-GPU, multi-GPU (DataParallel), and optimized DistributedDataParallel (DDP) with Automatic Mixed Precision (AMP)—affect denoising performance and training efficiency. We use U-Net [5] and U-Net++ [6], two widely used CNN architectures in medical imaging, selected for their effectiveness and computational efficiency in image-to-image restoration tasks.

Our contributions are as follows: (1) We develop a distributed deep learning framework for medical image denoising using Gaussian-noised X-ray inputs; (2) We compare U-Net and U-Net++ performance under various GPU configurations; (3) We integrate DDP and AMP for improved training efficiency; and (4) We analyze the viability of noise-based obfuscation as a privacy-preserving mechanism for clinical pipelines.

II. Literature Review

Research on distributed deep learning has grown in recent years, with applications ranging from generic image recognition to specialized domains. This section reviews literature that is related to our work. Specifically, we review prior work in three pertinent areas: (a) distributed training to improve efficiency/scalability, (b) privacy-preserving machine learning for sensitive data, and (c) deep learning models for medical image denoising and segmentation.

A. Distributed Deep Learning for Efficiency and Scalability

Distributed Deep Learning (DDL) has gained momentum for its ability to scale training processes across multiple GPUs or nodes [7]–[11]. This has in turn significantly reducing convergence time and enabling the handling of large datasets. Nurnoby et al. [7] showed up to 80% speedup in convergence time using multi-GPU training for image classification, underscoring the potential of parallelism in accelerating deep learning workflows. In medical imaging, where input sizes are often large and training can be resource-intensive, DDL presents a practical pathway for scaling. Pal et al. [12] explored both data and model parallelism, finding that hybrid schemes can further improve throughput for deep networks. However, for mid-sized models like U-Net and U-Net++, data parallelism

alone is often sufficient and offers lower implementation complexity. Our work leverages PyTorch's DistributedDataParallel (DDP) with Automatic Mixed Precision (AMP) to maximize performance while maintaining training stability.

B. Privacy-Preserving Machine Learning

Medical data is inherently sensitive, requiring robust privacy measures during model training and deployment. Federated Learning (FL) frameworks aim to address this by keeping data localized at the source and aggregating only model updates on a central server [13]. Although FL reduces data exposure, it introduces challenges such as communication overhead, slower convergence, and potential model divergence across sites. Encoding-based methods like Privacy-SF [14] pre-process images into latent representations before training, ensuring raw data is never exposed. Differential Privacy (DP) methods such as DP-SGD [15] further protect individual records by adding calibrated noise to gradients, though often at the cost of model performance. Homomorphic encryption and secure multi-party computation techniques provide strong theoretical guarantees but are computationally expensive and less feasible for large-scale CNNs. We adopt a lightweight privacy mechanism (additive Gaussian noise) as a data obfuscation method that reduces identifiable information without compromising efficiency.

C. Deep Learning Models for Medical Image Denoising and Segmentation

Denoising is critical in medical imaging for enhancing diagnostic utility, especially when using low-dose acquisition protocols or archival datasets. Convolutional neural networks (CNNs) like U-Net [5] and its variants remain popular due to their strong performance on segmentation and restoration tasks. U-Net++ [6] introduces nested skip connections and intermediate layers to enable finer reconstruction of structural features, making it especially valuable in high-noise environments. DnCNN [8] introduced residual learning for denoising, and DudeNet [16] applied dual-branch designs for robust noise removal on chest X-rays. Traditional filters such as BM3D have largely been replced by CNN-based models that better preserve anatomical fidelity and generalize to variable noise conditions. Given their balance of accuracy, efficiency, and memory footprint, U-Net and U-Net++ are well-suited for distributed training pipelines in clinical scenarios. Our work builds upon these foundations by integrating DDL and obfuscation within a practical and scalable denoising framework, a combination not extensively covered in prior research, and by examining both performance gains and privacy considerations together.

III. Methodology

This section describes our distributed deep learning pipeline for medical image denoising, covering data preprocessing, model architectures, training configurations, and evaluation metrics. Our framework is designed to assess the trade-offs between accuracy, speed, and data privacy when using distributed training strategies such as DataParallel and DistributedDataParallel (DDP) with Automatic Mixed Precision (AMP). The research flowchart from the problem definition till the final result evaluation and comparison is given in Fig. 1.

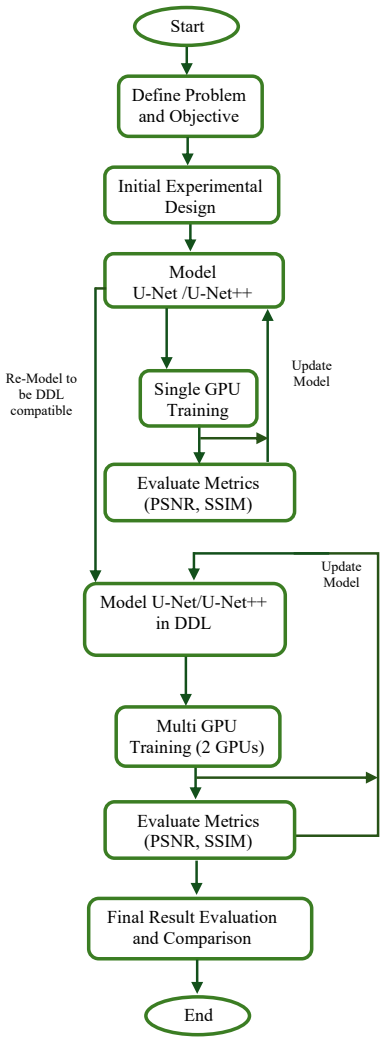

Fig. 1. Research Workflow

Fig. 2 presents the overall pipeline. The pipeline is divided into two logical locations: Site One, where chest X-ray images are acquired and noise is injected as a privacy-preserving transformation; and Site Two, where denoising is performed using a U-Net/UNet++ model. A deep learning model (U-Net or U-Net++) is trained on noisy-clean image pairs using multiple GPUs. The resulting denoised images support downstream clinical interpretation. This end-to-end setup enables efficient training and inference under constrained and privacy-conscious conditions.

A. Data Preparation and Noise Obfuscation

We used 15,000 chest radiographs sampled from the publicly available NIH ChestX-ray14 dataset, originally compris-

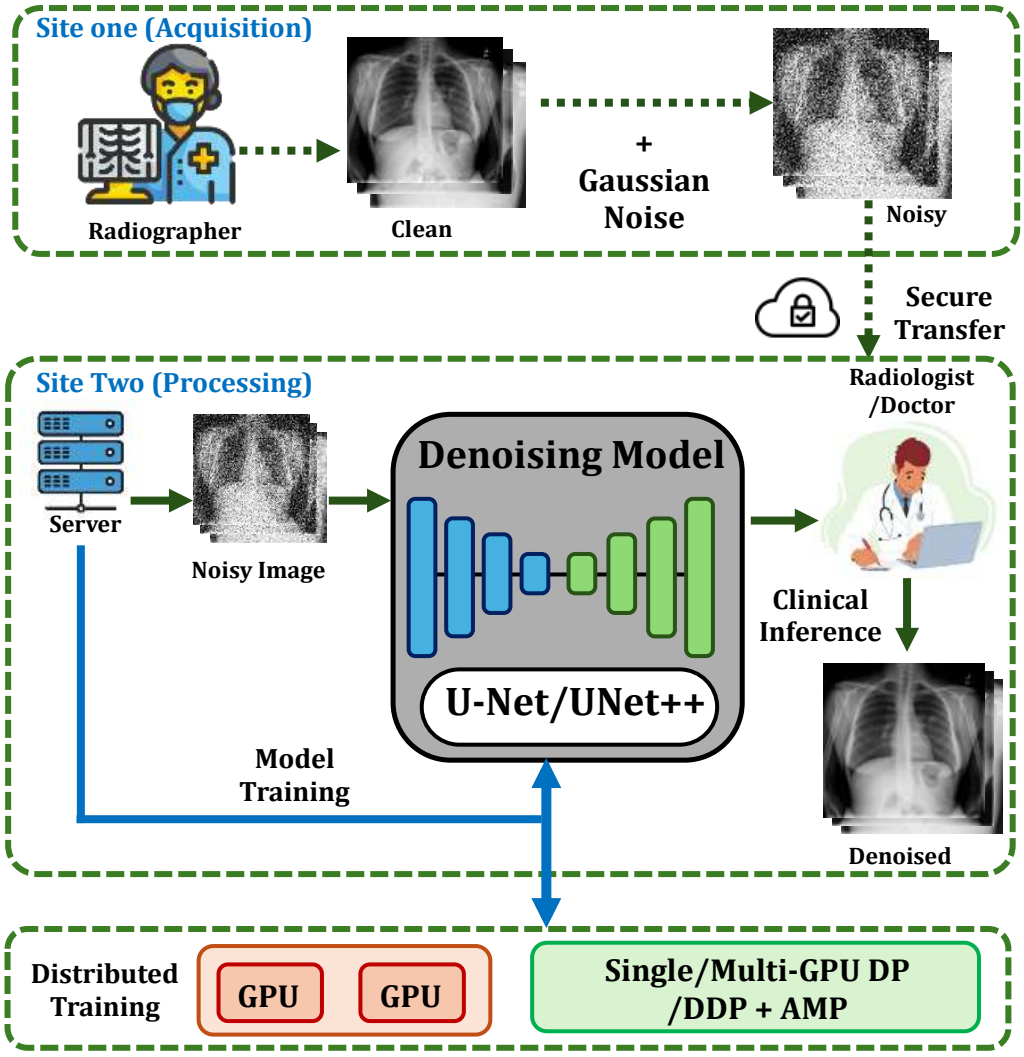

Fig. 2. Framework of the proposed distributed denoising pipeline using Gaussian noise-based data obfuscation. At Site One, clean chest X-ray images are acquired and locally obfuscated using additive Gaussian noise. The noisy images are securely transferred to Site Two, where they are processed by a U-Net/UNet++ model. Model training is performed using various GPU configurations—including single GPU, multi-GPU (DataParallel), and DDP with AMP to support scalability and diverse hardware environments. The denoised output is used for clinical interpretation by a radiologist

ing 112,120 frontal-view X-rays from over 30,000 patients. All images were grayscale and resized from 1024×1024 to 256×256 to reduce computational cost and enable multi-GPU training with practical batch sizes. This resolution is common in CXR deep learning studies [17]–[19] and provides a balance between anatomical detail and tractability [20].

To simulate a privacy-aware training setup, we applied additive Gaussian noise to the clean images. This obfuscation serves dual purposes: it masks sensitive patient features and creates a supervised denoising task. Gaussian noise with fixed mean 0.1 and standard deviations of 0.1, 0.2, and 0.3 was added to generate 10%, 20%, and 30% noise levels, respectively. By keeping the noise mean fixed at 0.1, we ensure a consistent brightness in the training data.

The noisy images were used as model inputs, and clean images served as targets. Data were split into 7,499 training, 4,949 validation, and 2,551 test pairs (approximately 50/33/17%). No other augmentations were applied to preserve anatomical orientation. This setup models a realistic scenario in which hospitals share perturbed images while retaining original data locally. The samples from the datasets is given in Fig. 3.

B. Denoising Models: U-Net and U-Net++

We evaluated two encoder–decoder convolutional network architectures (U-Net and U-Net++) for mapping noisy X-ray images to their clean counterparts. Both are widely used in medical imaging for their ability to capture fine anatomical details while remaining computationally tractable. U-Net consists of a symmetric encoder-decoder structure with skip connections to preserve spatial information. Each encoder block comprises two 3×3 convolutions followed by max pooling,

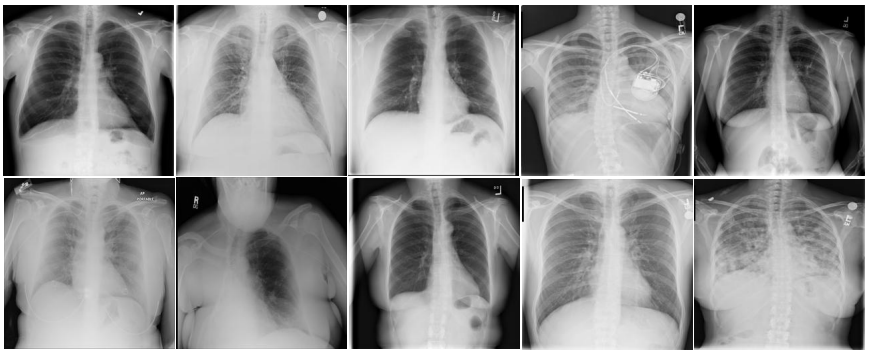

Fig. 3. Dataset Sample

and the decoder mirrors this with transposed convolutions and concatenated skip paths.

U-Net++ extends U-Net by incorporating nested skip connections and intermediate convolutional layers across scales. These refinements enhance multi-scale feature aggregation and gradient flow, which can improve restoration quality under high noise levels. Bilinear interpolation is used for upsampling to reduce checkerboard artifacts.

Both models were implemented in PyTorch with a base channel width of 64 and trained using identical protocols. A high-level comparison is summarized in Table I.

TABLE I
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

C. Distributed Training Setup

To accelerate training and support scalability, we implemented both `DataParallel` and `DistributedDataParallel` (DDP) strategies using PyTorch. `DataParallel` replicates the model across GPUs and aggregates gradients on the main device (typically GPU 0), while DDP launches one process per GPU with local gradient computation and synchronized updates, leading to better performance and scaling efficiency.

For additional speedup and reduced memory usage, we integrated Automatic Mixed Precision (AMP) via `torch.cuda.amp`. AMP dynamically casts computations to FP16 where safe, leveraging Tensor Cores on modern GPUs. Gradient scaling was used to maintain numerical stability.

All training was conducted on a workstation with two NVIDIA RTX A4500 GPUs (20 GB memory each), CUDA 12.2, and PyTorch 1.X. DDP+AMP was the fastest and most stable configuration, as detailed in Section IV. The details of the training framework and hardware configuration used is presented in Table II below.

TABLE II
TRAINING FRAMEWORK AND HARDWARE CONFIGURATION

Attribute	Details
Framework	PyTorch 1.X with `nn.DataParallel`
GPUs Used	2 × 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

D. Training Protocol and Implementation Details

All models were trained using the Adam optimizer [21] with an initial learning rate of 1×10^{-3} and L1 loss, which empirically preserved image texture better than L2. Each model was trained for 50 epochs, with early stopping monitored but not triggered. The best model was selected based on minimum validation loss.

Batch size was set to 16 for single-GPU and 32 for dual-GPU setups (16 per GPU). We kept the learning rate fixed across configurations to enable fair comparison, despite theoretical justifications for scaling it linearly with batch size.

We used PyTorch's native support for AMP and enabled performance optimizations such as pinned memory, preloading, and non-overlapping data loading with `DistributedSampler`. All models were implemented in PyTorch 1.X and evaluated using the same test dataset.

E. Evaluation Metrics and Visualization

We assessed model performance using three complementary metrics: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS). PSNR and SSIM (defined in Eq. 1 and 2 respectively) quantify pixel-wise fidelity and structural alignment relative to ground truth, while LPIPS reflects perceptual similarity based on deep feature embeddings.

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{\text{MSE}(I, \hat{I})} \right) \tag{1}$$

TABLE III
DENOISING PERFORMANCE (PSNR, SSIM, LPIPS) FOR U-NET AND U-NET++ AT DIFFERENT NOISE LEVELS WITH 95% CONFIDENCE INTERVALS.

Noise Level	U-Net			U-Net++		
	PSNR (dB)	SSIM	LPIPS	PSNR (dB)	SSIM	LPIPS
10%	34.95 (±0.04)	0.9168 (±0.0008)	0.1373 (±0.001)	34.39 (±0.05)	0.9123 (±0.0007)	0.1585 (±0.0011)
20%	32.26 (±0.04)	0.8907 (±0.0011)	0.2010 (±0.0013)	32.32 (±0.05)	0.8959 (±0.0010)	0.2265 (±0.0015)
30%	30.23 (±0.05)	0.8746 (±0.0012)	0.2498 (±0.0016)	30.76 (±0.05)	0.8840 (±0.0011)	0.2479 (±0.0016)

MAX_I is the maximum possible pixel value (typically 255), and $\text{MSE}(I, \hat{I})$ is the mean squared error between the ground truth and the denoised image. Higher PSNR values indicate lower reconstruction error, with ∞ representing a perfect match.

$$\text{SSIM}(I, \hat{I}) = \frac{(2\mu_I\mu_{\hat{I}} + C_1)(2\sigma_{I\hat{I}} + C_2)}{(\mu_I^2 + \mu_{\hat{I}}^2 + C_1)(\sigma_I^2 + \sigma_{\hat{I}}^2 + C_2)} \quad (2)$$

Here, μ, σ^2, and $\sigma_{I\hat{I}}$ denote the mean, variance, and covariance of image patches, respectively, and C_1, C_2 are small constants to stabilize the division.

All metrics were averaged over a held-out test set of 2,551 images. We report mean values with 95% confidence intervals. For qualitative analysis, we selected representative samples showing noisy inputs, denoised outputs (U-Net and U-Net++), and the corresponding ground truths to visually evaluate preservation of anatomical details such as ribs, lung textures, and diaphragm contours.

IV. RESULTS AND DISCUSSION

We evaluated U-Net and U-Net++ across three Gaussian noise levels (10%, 20%, and 30%) and two hardware configurations (single-GPU and dual-GPU). Performance was measured using PSNR, SSIM, and LPIPS on a held-out test set of 2,551 images. We also analyzed training efficiency in terms of wall-clock time and GPU utilization.

A. Quantitative Performance and Speedup

Table III summarizes model performance with 95% confidence intervals. At 10% noise, U-Net achieved slightly better PSNR and SSIM than U-Net++, while also producing lower LPIPS values, suggesting superior perceptual fidelity. At higher noise levels (20% and 30%), U-Net++ consistently outperformed U-Net in PSNR and SSIM, demonstrating better structural preservation. LPIPS scores became comparable at 30%, where both models converged in perceptual similarity.

All metric differences were statistically significant ($p < 10^{-13}$). These results highlight a trade-off: U-Net offers better perceptual alignment at low noise, whereas U-Net++ preserves structure better under heavier degradation.

B. Training Efficiency and Multi-GPU Acceleration

We compared training times across three configurations: 1 GPU, 2 GPUs with `DataParallel`, and 2 GPUs with `DDP + AMP`. As illustrated in Fig. 4, DDP+AMP yielded the largest reduction (over 60%) in training time outperforming both the single-GPU and `DataParallel` configurations. Training time was reduced by 36–47% when moving from 1 GPU to 2 GPUs with `DataParallel`, and further reduced by over 60% with the optimized setup. For instance, under 10% noise, U-Net training time dropped from 7328.2 seconds (1 GPU) to 4665.2 seconds (2 GPUs, `DataParallel`), and further to just 2737 seconds using DDP + AMP. Similar trends was observed for U-Net++.

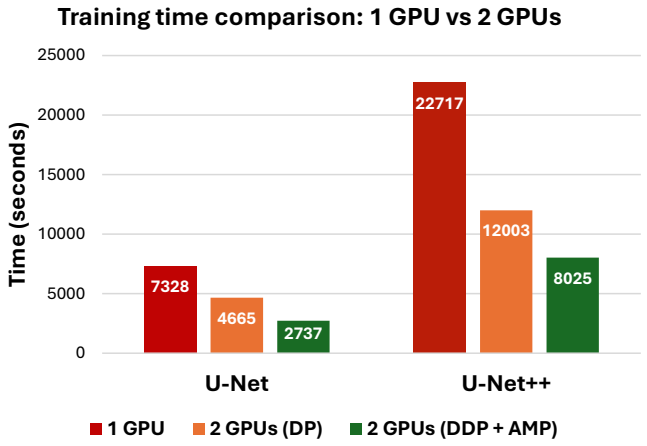

Fig. 4. Training time comparison for U-Net and U-Net++ across three configurations: 1 GPU, 2 GPUs with DataParallel, and 2 GPUs with DDP + AMP. The optimized configuration yields the shortest training time for both models.

Table IV shows full results across all noise levels and hardware configurations. U-Net++ have less performance reduction under dual-GPU settings with respect to the performance in single-GPU setup, though it remains more computationally demanding.

TABLE IV
PSNR (DB), SSIM, LPIPS, AND TRAINING TIME (IN SECONDS) FOR U-NET AND U-NET++ UNDER VARYING NOISE LEVELS AND GPU CONFIGURATIONS.

Noise	Model	GPUs	PSNR (dB)	SSIM	LPIPS	Training Time (s)
10%	U-Net	1	34.952	0.9168	0.1373	7328.2
10%	U-Net++	1	34.387	0.9123	0.1585	22717.1
10%	U-Net	2	34.034	0.9060	0.1582	4665.2
10%	U-Net++	2	34.089	0.9083	0.1564	12003.0
20%	U-Net	1	32.255	0.8907	0.2011	7326.1
20%	U-Net++	1	32.315	0.8959	0.2265	22266.0
20%	U-Net	2	32.157	0.8906	0.2143	4648.1
20%	U-Net++	2	32.151	0.8886	0.2000	11998.5
30%	U-Net	1	30.225	0.8746	0.2498	7322.0
30%	U-Net++	1	30.756	0.8840	0.2479	22343.5
30%	U-Net	2	30.481	0.8757	0.2661	4648.2
30%	U-Net++	2	30.691	0.8830	0.2638	12015.6

TABLE V
COMPARISON OF DENOISING PERFORMANCE, TRAINING TIME (TT) IN SECONDS, AND PERCENTAGE TIME SAVINGS (TS) FOR U-NET AND U-NET++ MODELS UNDER 10% GAUSSIAN NOISE USING DIFFERENT GPU CONFIGURATIONS

Model	Setup	PSNR (dB)	SSIM	LPIPS	TT (s)	TS (%)
U-Net	1 GPU	34.952	0.9168	0.1373	7328	0.00%
U-Net	2 GPUs (DP)	34.034	0.9060	0.1582	4665	36.34%
U-Net	2 GPUs (DDP + AMP)	**34.483**	**0.9067**	**0.1562**	**2737**	**62.65%**
U-Net++	1 GPU	34.387	0.9123	0.1585	22717	0.00%
U-Net++	2 GPUs (DP)	34.089	0.9083	0.1564	12003	47.16%
U-Net++	2 GPUs (DDP + AMP)	**33.416**	**0.8927**	**0.2175**	**8025**	**64.69%**

C. Optimized Setup: DDP + AMP

Table V summarizes the performance of our optimized DDP+AMP implementation compared to baseline single- and dual-GPU training under 10% Gaussian noise. Using DDP+AMP led to major speedups: U-Net training time dropped from 7328 to 2737 seconds, and U-Net++ from 22717 to 8025 seconds.

U-Net's performance held steady across all setups, while U-Net++ saw mild degradation under DDP+AMP. This suggests deeper models may require finer hyperparameter tuning in distributed training.

D. Visual Results and Structural Fidelity

To qualitatively assess image reconstruction fidelity, we compare denoising outputs at 10%, 20%, and 30% Gaussian noise levels using representative test samples. Each comparison (Done under 1-GPU configurations) includes the noisy input, U-Net output, U-Net++ output, and the clean ground truth.

Fig. 5, 6, and 7 illustrate results at 10%, 20%, and 30% noise levels, respectively.

As shown in Fig. 5, U-Net produces relatively clean reconstructions, though some residual speckle noise remains. U-Net++, by contrast, yields sharper edge definitions and restores anatomical continuity more faithfully. Structures such as ribs, clavicles, and lung boundaries exhibit enhanced contrast. In particular, the apical regions show smoother transitions and more precise rib contours. The visual SSIM appears higher for U-Net++, with better preservation of brightness consistency and edge contrast.

At 20% noise (Fig. 6), U-Net begins to blur finer anatomical details. Mediastinal contours and soft gradients become less distinct. U-Net++, however, maintains better spatial consistency across lung lobes and diaphragm curvature. Noise artifacts that appear in the lower lobes under U-Net are largely suppressed in U-Net++, which retains soft tissue definition.

At 30% noise (Fig. 7), U-Net visibly struggles. Texture smearing and structural loss become prominent, with ambiguous regions such as the vertebral shadow and costophrenic angle. U-Net++ remains relatively robust, with only modest blurring and clear preservation of chest wall and lung structures. Despite strong noise perturbation, U-Net++ maintains perceptual realism and continuity in critical anatomical areas.

E. Comparison with Prior Methods

We compare our results with other works that employed Gaussian Noise for image denoising and recorded the results as given in Table VI.

TABLE VI
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 [22]	15%	31.08/0.8722	25%	28.57/0.8017
TNDR [23]	15%	31.42/0.8826	25%	28.92/0.8157
DnCNN-3 [24]	15%	31.46/0.8826	25%	29.02/0.8190

Compared to several established denoising methods, our *U-Net++* approach demonstrates superior performance, particularly under higher noise levels. While prior models such as *BM3D*, *TNDR*, and *DnCNN-3* report PSNR values around 31.1–31.5 dB and SSIM scores of 0.87–0.88 under 15% Gaussian noise, our *U-Net++* model achieves a higher PSNR of 32.32 dB and SSIM of 0.8959 at 20% noise, indicating better resilience to increased corruption. The performance gap becomes even more pronounced at heavier noise levels: at 25–30% noise, *U-Net++* achieves 30.76 dB / 0.8840 SSIM, outperforming the closest competitor, *DnCNN-3* (29.02 dB / 0.8190), by more than 1.2 dB in PSNR and over 5 percentage points in SSIM. These results highlight the robustness of our nested architecture in preserving structural details and recovering image fidelity, even under more challenging noise conditions where traditional and early deep learning models begin to degrade noticeably.

V. CONCLUSION AND FUTURE WORK

This study highlights the efficacy of distributed deep learning for medical image denoising and explores the trade-offs between model complexity and training efficiency. Through systematic experiments, we demonstrated that U-Net++ consistently delivers better structural fidelity and robustness under medium to high Gaussian noise levels, owing to its nested skip connections and dense multi-scale aggregation. In contrast, U-Net provides faster convergence and competitive results, making it a strong candidate for low-noise or real-time scenarios.

Our implementation of an optimized multi-GPU training pipeline using PyTorch's `DistributedDataParallel` (DDP) and Automatic Mixed Precision (AMP) significantly

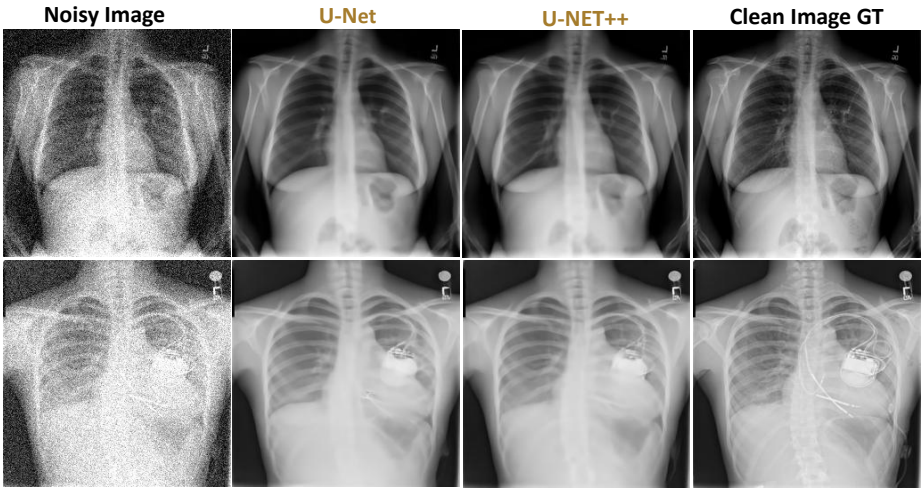

Fig. 5. Visual comparison of noisy input, U-Net and U-Net++ denoised outputs, and ground truth (10% noise levels).

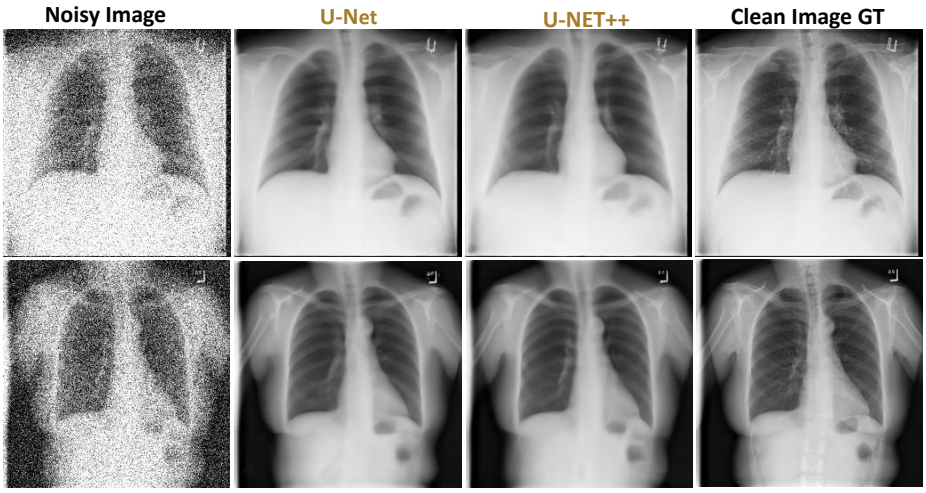

Fig. 6. Visual comparison of noisy input, U-Net and U-Net++ denoised outputs, and ground truth (20% noise levels).

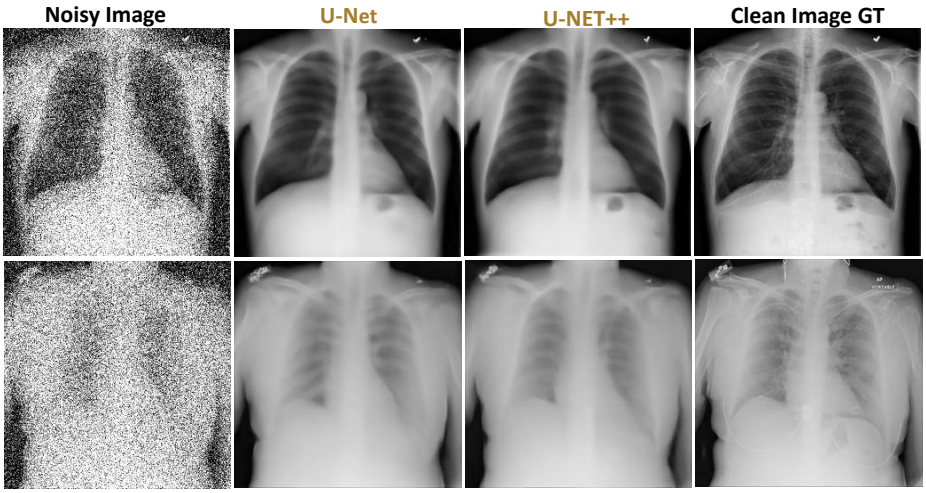

Fig. 7. Visual comparison of noisy input, U-Net and U-Net++ denoised outputs, and ground truth (30% noise levels).

reduced training time (up to 64%) without major loss in denoising quality. These improvements affirm the practical value of software-level acceleration techniques in scaling deep learning workflows for clinical imaging, particularly where time or hardware resources are limited.

While U-Net++ incurs a higher computational cost, its perceptual and structural advantages may justify the expense in diagnostic imaging tasks where detail preservation is critical. Selecting between U-Net and U-Net++ should be guided by the clinical context, noise level, and available compute budget.

Looking forward, we aim to enhance both performance and privacy aspects of our framework. Future directions include integrating attention mechanisms to improve fine-grained reconstruction, exploring perceptual loss functions (e.g., VGG or adversarial losses), and conducting human-in-the-loop evaluations with radiologists. We also plan to extend this work into a federated learning paradigm, enabling decentralized training across multiple institutions without exposing sensitive image data.

In conclusion, our work provides a scalable, privacy-conscious, and efficient approach to medical image denoising. These insights contribute to advancing the deployment of distributed deep learning in real-world medical systems and support the development of secure, generalizable AI solutions in healthcare.

ACKNOWLEDGMENT

The authors would like to acknowledge all support provided by King Fahd University of Petroleum and Minerals (KFUPM).

REFERENCES

[1] 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.

[2] M. Lai, "Deep learning for medical image segmentation," *arXiv preprint arXiv:1505.02000*, 2015.

[3] O. Aouedi, A. Sacco, K. Piamrat, and G. Marchetto, "Handling privacy-sensitive medical data with federated learning: challenges and future directions," *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 2, pp. 790–803, 2022.

[4] 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.

[5] 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.

[6] Z. Zhou, M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," in *International Workshop on Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 3–11, Springer, 2018.

[7] M. F. Nurnoby, K. A. A. Shawarib, and A. U. H. Khan, "Distributed deep learning-based model for large image data classification," in *Proceedings of the 7th International Conference on Future Networks and Distributed Systems*, pp. 283–291, 2023.

[8] H. Zhang, Z. Zheng, S. Xu, W. Dai, Q. Ho, X. Liang, H. Wang, and E. P. Xing, "Poseidon: An efficient communication architecture for distributed deep learning on GPU clusters," in *USENIX Annual Technical Conference (USENIX ATC)*, pp. 181–193, 2017.

[9] D. Schaa and D. Kaeli, "Exploring the multiple-gpu design space," in *2009 IEEE International Symposium on Parallel & Distributed Processing*, pp. 1–12, IEEE, 2009.

[10] Y. Sun, T. Baruah, S. A. Mojumder, S. Dong, X. Gong, S. Treadway, R. Ausavarungnirun, and D. Kaeli, "Mgpusim: Enabling multi-gpu performance modeling and optimization," in *Proceedings of the 46th International Symposium on Computer Architecture*, pp. 197–209, ACM, 2019.

[11] M. Zhu and Q. Chen, "Big data image classification based on distributed deep representation learning model," *IEEE Access*, vol. 8, pp. 133890–133904, 2020.

[12] S. Pal, E. Ebrahimi, A. Zulfiqar, Y. Fu, V. Zhang, S. Migacz, *et al.*, "Optimizing multi-gpu parallelization strategies for deep learning training," *IEEE Micro*, vol. 39, no. 5, pp. 91–101, 2019.

[13] D. C. Nguyen, Q.-V. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, J. Li, and W. H. Hwang, "Federated learning for smart healthcare: A survey," *ACM Computing Surveys (CSUR)*, vol. 55, no. 3, pp. 1–37, 2022.

[14] L. Chen, L. Song, H. Feng, R. T. Zeru, S. Chai, and E. Zhu, "Privacy-sf: An encoding-based privacy-preserving segmentation framework for medical images," *Image and Vision Computing*, vol. 151, p. 105246, 2024.

[15] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

[16] A. Sahu, K. Rana, and V. Kumar, "An application of deep dual convolutional neural network for enhanced medical image denoising," *Medical & Biological Engineering & Computing*, vol. 61, no. 5, pp. 991–1004, 2023.

[17] O. H. Khater, A. S. Shuaibu, S. U. Haq, and A. J. Siddiqui, "Attcdcnet: Attention-enhanced chest disease classification using x-ray images," in *2025 IEEE 22nd International Multi-Conference on Systems, Signals & Devices (SSD)*, pp. 891–896, IEEE, 2025.

[18] A. Jain, A. Bhardwaj, K. Murali, and I. Surani, "A comparative study of cnn, resnet, and vision transformers for multi-classification of chest diseases," *arXiv preprint arXiv:2406.00237*, 2024.

[19] H. Nakrani, E. Q. Shahra, S. Basurra, R. Mohammad, E. Vakaj, and W. A. Jabbar, "Advanced diagnosis of cardiac and respiratory diseases from chest x-ray imagery using deep learning ensembles," *Journal of Sensor and Actuator Networks*, vol. 14, no. 2, p. 44, 2025.

[20] Z. Huang, J. Lin, L. Xu, H. Wang, T. Bai, Y. Pang, and T.-H. Meen, "Fusion high-resolution network for diagnosing chestx-ray images," *Electronics*, vol. 9, no. 1, p. 190, 2020.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[22] H. C. Burger, C. J. Schuler, and S. Harmeling, "Image denoising: Can plain neural networks compete with bm3d?," in *2012 IEEE conference on computer vision and pattern recognition*, pp. 2392–2399, IEEE, 2012.

[23] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1256–1272, 2016.

[24] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE transactions on image processing*, vol. 26, no. 7, pp. 3142–3155, 2017.