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 Measure) at higher noise levels (20% and 30%), 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 30% 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.

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

Distributed deep learning (DDL) has emerged as a practical way to accelerate training by leveraging multiple GPUs or nodes, reducing convergence time for large datasets [7]–[11]. In medical imaging, where training can be resource-intensive, PyTorch's DistributedDataParallel (DDP) combined with Automatic Mixed Precision (AMP) offers an efficient balance of scalability and stability.

Privacy-preserving machine learning methods such as federated learning [12], differential privacy [13], and encryption techniques [14] aim to reduce data exposure. However, these approaches often introduce high communication or computational overhead, limiting clinical adoption. As a lightweight alternative, additive Gaussian noise can serve as a practical obfuscation proxy: it masks fine patient-specific details while avoiding the computational and communication costs of stronger methods such as federated learning or encryption. We position it here not as a comprehensive privacy solution, but as an experimental stand-in that enables controlled benchmarking of distributed training frameworks.

For denoising, convolutional neural networks (CNNs) have largely replaced traditional filters such as BM3D. U-Net [5] and its nested variant U-Net++ [6] remain widely used

[1] NIH Chest X-rays

2471-7819/25/$31.00 ©2025 IEEE
DOI 10.1109/BIBE66822.2025.00021

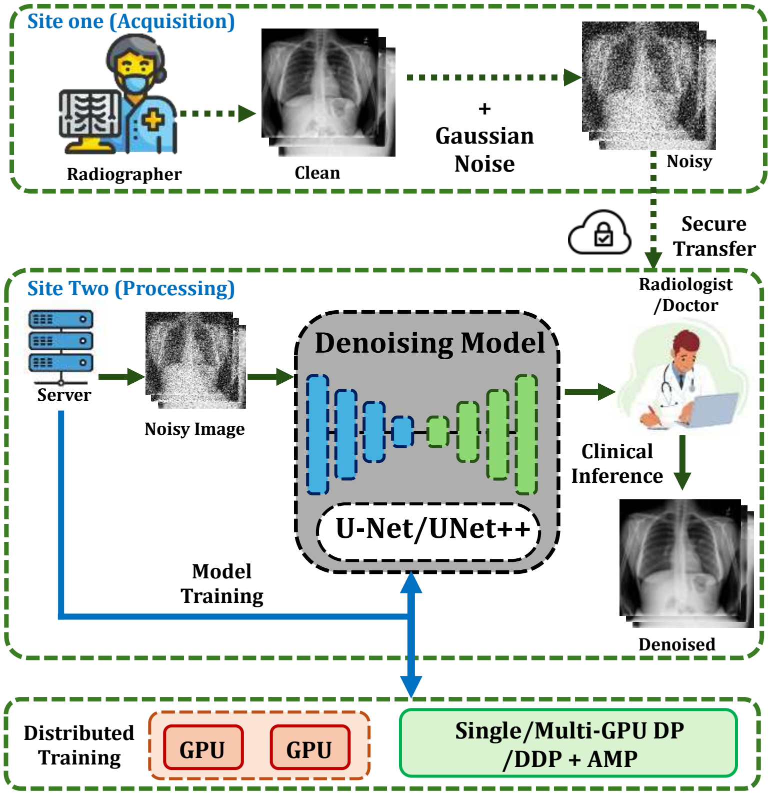

Fig. 1. Framework of the proposed distributed denoising pipeline using Gaussian noise-based data obfuscation. Figure 1 shows the pipeline: Site One injects Gaussian noise; Site Two trains and denoises using U-Net/U-Net++ across GPU setups.

for segmentation and restoration tasks due to their accuracy–efficiency trade-offs. This paper builds on these foundations by integrating distributed training with noise-based obfuscation for medical image denoising, an area that has received limited prior attention.

III. Methodology

This section outlines our distributed deep learning pipeline for medical image denoising, including data, models, training, and evaluation. The framework assesses accuracy, speed, and privacy trade-offs using DataParallel and DistributedDataParallel (DDP) with Automatic Mixed Precision (AMP).

Fig. 1 shows the pipeline with two sites: Site One acquires chest X-rays and applies Gaussian noise as a privacy-preserving transformation; Site Two trains U-Net/U-Net++ on noisy–clean pairs using multiple GPUs. The denoised outputs support clinical interpretation, enabling efficient and privacy-conscious training and inference.

A. Data Preparation and Noise Obfuscation

We used 15,000 chest radiographs from the NIH ChestX-ray14 dataset (112,120 frontal-view X-rays from 30,000 patients). Images were grayscale and resized to 256×256, a standard resolution in CXR studies [15]–[18] that balances detail and tractability.

Additive Gaussian noise (mean 0.1; std 0.1, 0.2, 0.3 → 10%, 20%, 30%) was applied, simultaneously masking patient features and defining a supervised denoising task with consistent brightness.

Noisy images were inputs and clean images were targets, split into 7,499 training, 4,949 validation, and 2,551 test pairs (approximately 50/33/17%). No further augmentation was used to preserve anatomical orientation, reflecting a scenario where hospitals share perturbed images while keeping originals locally.

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.	$\sim 8.6\,\mathrm{M}$ ($N_c = 64$)	$\sim 9.2\,\mathrm{M}$ (base_ch=64)
Key Features	ReLU, BatchNorm, symmetric skip connections, transposed conv upsampling	Dense skip pathways, multi-depth aggregation, multiple output heads, bilinear upsampling

B. Distributed Training Setup

To accelerate training, we used PyTorch's `DataParallel` and `DistributedDataParallel`

(DDP). `DataParallel` replicates models across GPUs, while DDP spawns one process per GPU with synchronized updates for better scaling. We also applied Automatic Mixed Precision (AMP) via `torch.cuda.amp`, casting safe operations to FP16 with gradient scaling for stability. Experiments ran on 2× NVIDIA RTX A4500 GPUs (CUDA 12.2, PyTorch; Table II), where DDP+AMP proved fastest.

TABLE II
TRAINING FRAMEWORK AND HARDWARE CONFIGURATION

Attribute	Details
Framework	PyTorch 1.X with `nn.DataParallel` and `DistributedDataParallel`
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

C. Training Protocol and Implementation Details

Models were trained with Adam [19], learning rate of 1×10^{-3} and L1 loss, which better preserved texture than L2. Training ran for 50 epochs with early stopping monitored but not triggered, selecting the best model by validation loss.

Batch sizes were 16 (single-GPU) and 32 (dual, 16/GPU). The learning rate was fixed across setups for fair comparison. AMP was combined with optimizations (pinned memory, preloading, and `DistributedSampler`). All models were implemented in PyTorch 1.X and evaluated on the same test set.

D. Evaluation Metrics and Visualization

Performance was assessed using Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS). PSNR/SSIM quantify fidelity and structure (Eqs. 1, 2), while LPIPS measures perceptual similarity from deep features.

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

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.

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

Here, μ, σ^2, and $\sigma_{I\hat{I}}$ denote mean, variance, and covariance, while C_1, C_2 are stabilizers.

Metrics were averaged over 2,551 test images, with 95% confidence intervals reported for 1-GPU only. For qualitative analysis, we compared noisy inputs, U-Net and U-Net++ outputs, and clean ground truths.

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.

A. Quantitative Performance and Speedup

Table III consolidates denoising performance (mean ± 95% CI for the 1-GPU setup) and training time across GPU configurations, including 2-GPU (DataParallel) and DDP+AMP at 10% noise. 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 setups: 1 GPU, 2 GPUs with `DataParallel`, and 2 GPUs with `DDP+AMP`. As shown in Fig. 2, DDP+AMP achieved the largest reduction ($>$ 60%), outperforming both single-GPU and `DataParallel`. Training time decreased by 36–47% when moving from 1 GPU to `DataParallel`, and by over 60% with DDP+AMP. For example, under 10% noise, U-Net training dropped from 7328 s (1 GPU) to 4665 s (`DataParallel`) and 2737s (DDP+AMP). Similar trends were observed for U-Net++.

Table III reports comprehensive results across noise levels and GPU setups. Under 2 GPUs (DataParallel), metric shifts are small-to-moderate: up to approximately 0.92 dB PSNR in the worst case (U-Net at 10%), and typically $\leq$0.3 dB otherwise; SSIM changes are within $\leq$ 0.01, with direction varying by noise level.

C. Optimized Setup: DDP + AMP

As shown in Table III, at 10% noise DDP+AMP cuts training time from 7328→2737s (U-Net) and 22717→8025s (U-Net++). Quality shifts are modest for U-Net and more pronounced for U-Net++, suggesting the deeper model may require tighter hyperparameter tuning under distributed training.

D. Visual Results and Structural Fidelity

To qualitatively assess image reconstruction fidelity, we provide a representative comparison at 20% Gaussian noise (Fig. 3). Each panel shows the noisy input, the U-Net output, the U-Net++ output, and the clean ground truth.

As shown in Fig. 3, U-Net++ preserves sharper structures, while U-Net produces smoother but less detailed outputs. At

TABLE III

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.

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%

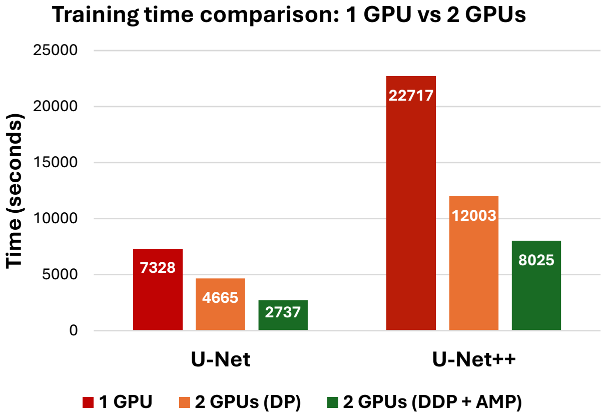

Fig. 2. Training time comparison for U-Net and U-Net++ across three configurations: 1 GPU, 2 GPUs with DataParallel, and 2 GPUs with DDP + AMP. The optimized configuration yields the shortest training time for both models.

TABLE IV

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

10% noise, U-Net offers slightly better perceptual scores, while at 30% noise, U-Net++ preserves structure more robustly. These observations align with the quantitative trends reported in Table III

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 IV.

Compared with reported ranges at 15–25% noise for BM3D/TNDR/DnCNN-3, our 20–30% results are higher (e.g., U-Net++ 32.32 dB / 0.8959 at 20%; 30.76 dB / 0.8840 at 30%), noting the noise-level mismatch. (Table IV).

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 reduced training time (62.68% for U-Net and 64.67% for U-Net++) 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

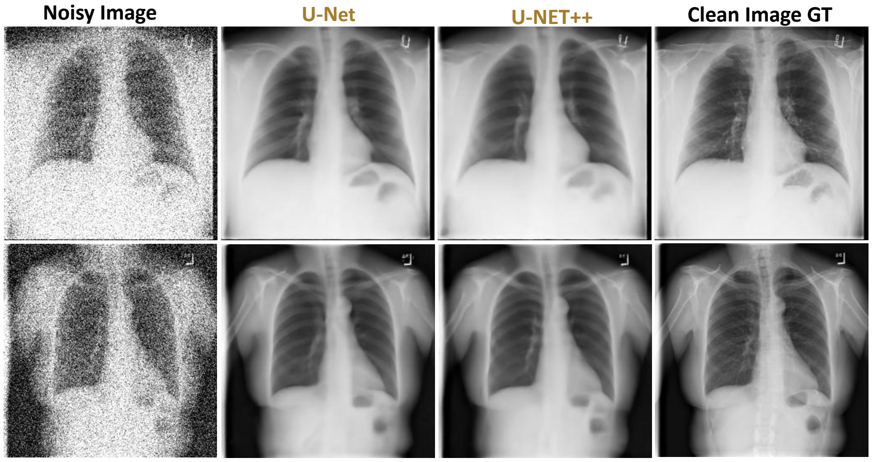

Fig. 3. Representative qualitative comparison at 20% Gaussian noise: noisy input, U-Net denoised, U-Net++ denoised, and clean ground truth.

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] 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.

[13] 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.

[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] 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.

[16] 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.

[17] 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.

[18] 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.

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

[20] 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.

[21] 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.

[22] 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.