iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** I **Month of publication:** January 2026

DOI: https://doi.org/10.22214/ijraset.2026.76801

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue I Jan 2026- Available at www.ijraset.com

On-Device Pneumonia Diagnosis: Benchmarking Deep Learning Architectures on Raspberry Pi

Abdul Ashhad[1], Dr. Danish Raza Rizvi[2]
Jamia Millia Islamia, Delhi 110025 India

Abstract: ***Pneumonia continues to be a serious health problem worldwide, especially for children under the age of five. As reported by the WHO, almost 740,000 children in this age group died from pneumonia in 2019, with most of these cases coming from rural areas where proper diagnostic facilities are limited. Chest X-rays are commonly used to detect pneumonia, but the accuracy of this method depends heavily on the radiologist's experience. Early diagnosis is important, but not always available in low-resource regions.***

Because of this, deep learning has gained attention as a possible alternative for automated pneumonia detection. Many models have been proposed, but it is still unclear which one works best on a low-cost, low-power device that can be deployed in rural setups. In this work, we compare several deep learning models—including a simple Sequential CNN, VGG16, ViT, MobileViT Hybrid, EfficientNet-V2S, and MobileNet-V3S—by running and evaluating them on a Raspberry Pi 4B. The goal is to find a model that gives good accuracy while still being efficient enough to run on a small device that can support healthcare in underserved areas.

Keywords: Pneumonia detection, deep learning, chest X-rays, CNN models, Vision Transformer, MobileNetV3, EfficientNet-V2S, MobileViT, Raspberry Pi, low-resource healthcare.

I. INTRODUCTION

This Pneumonia remains a leading cause of mortality worldwide, especially among children under five and in rural areas with limited access to trained doctors and diagnostic tools. Early detection is crucial, but interpreting chest X-rays (CXRs) requires expertise that may not be available in low-resource settings. Automated solutions using deep learning can assist healthcare workers by providing reliable, rapid analysis of CXRs.

While CNN-based models have shown strong performance in controlled environments, deploying them in real-world rural settings requires models that are not only accurate but also efficient enough to run on low-cost devices such as a Raspberry Pi. This raises an important question: which deep learning architecture balances high predictive performance with real-time deployability in resource-constrained environments? This study evaluates six models—from basic CNNs to VGG16, Vision Transformers, MobileNetV3, MobileViT Hybrid, and EfficientNet-V2S—directly on a Raspberry Pi 4B, comparing both accuracy and practical performance to identify the most deployment-ready model.

II. LITERATURE REVIEW

Research on automated pneumonia detection from chest X-rays has expanded rapidly due to the global need for fast and reliable diagnosis, especially in regions lacking radiology expertise. Early efforts relied on conventional CNNs, but recent literature shows a shift toward lightweight architectures, hybrid models, and Transformer-based designs that balance accuracy with efficiency and deployability.

Roy et al. [1] introduced VGG-Lite with an Edge-Enhanced module to address class imbalance, achieving performance comparable to heavier ViT models. Similarly, Xia et al. [2] proposed MedFormer, a hierarchical Vision Transformer tailored for medical imaging, showing strong results on multi-resolution CXRs. Transformer-based approaches have further evolved with region-aware mechanisms; for example, Saber et al. [3] combined multi-scale Transformers with lung segmentation to surpass 93% accuracy, while Bukhari [4] demonstrated that compact Transformer variants like MobileViT Small can outperform both CNNs and standard ViTs on pediatric datasets.

Survey works such as Siddiqi and Javaid [5] highlight that CNNs remain strong baselines on datasets like ChestX-ray14 and CheXpert, though Transformers often excel when sufficient data is available. Comparative studies by Al Reshan et al. [6] show that MobileNet architectures offer an optimal trade-off between accuracy and efficiency, reinforcing the value of lightweight networks. Modernized CNN architectures such as ConvNeXt [7] further narrow the performance gap with Transformers through updated design principles.

©IJRASET: All Rights are Reserved

Edge deployment has become a critical focus, with several studies evaluating models on embedded hardware. Pandey et al. [8] demonstrated the feasibility of quantized MobileNetV2 on Raspberry Pi, and Mehta et al. [9] achieved real-time pneumonia detection using INT8-optimized CNNs. These works collectively emphasize the importance of low-power, deployable architectures for rural healthcare settings.

Broader comparative studies (e.g., Almutairi et al. [10], EfficientNetV2 family [11]) and transfer learning advancements (Apostolopoulos and Mpesiana [12]) continue to shape the field, while foundational datasets like CheXpert [13] and ChestX-ray14 [14] remain central benchmarks. Rajpurkar et al.'s CheXNet [15] marked a landmark achievement by reaching radiologist-level performance using DenseNet-121. Crucially, Joshua et al. [16] stress that accuracy alone is insufficient; latency, throughput, and memory footprint must also guide model selection—particularly for edge devices. The widely used Kermany dataset [17] remains a standard reference for such evaluations.

III. DATASET USED

This study uses a publicly available dataset of 5,856 chest X-ray images collected from the Women and Children Medical Centre in Guangzhou, China, with each scan labelled as either Normal or Pneumonic. The data is arranged into training, validation, and test sets, each containing two class-specific folders. Since the dataset was pre-filtered at the source, unclear or low-quality scans had already been removed, so only basic manual cleaning—such as eliminating duplicates—may be needed. Before training, all images are resized and normalized according to the input requirements of each model, with grayscale normalization applied where single-channel inputs are expected. The dataset is publicly available on Mendeley Data: https://data.mendeley.com/datasets/rscbjbr9sj/2.

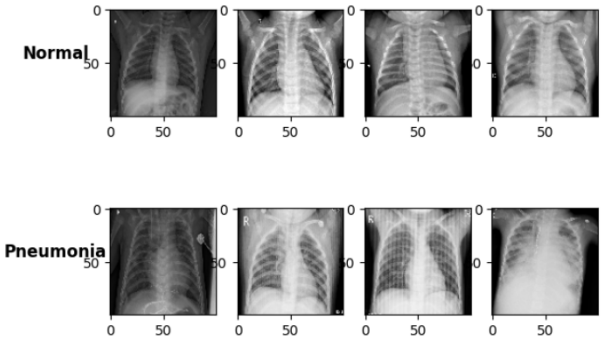


Fig. 1. Normal and Pneumonic images from the train set

IV. METHODOLOGY

The overall workflow of this study was designed to answer a simple but practical question: which deep learning model works best for pneumonia detection when deployed on a low-cost device like the Raspberry Pi 4B? To explore this, the methodology was divided into a clear sequence of steps. We began by preparing the dataset—cleaning it where needed, resizing the X-ray images, and applying model-specific normalization. After that, we trained a set of models ranging from a simple baseline CNN to more advanced architectures like VGG16, ViT, MobileNetV3-Small, EfficientNet-V2S, and a MobileViT Hybrid model. Each model followed the same training structure, but with hyperparameters adjusted so that their performances were comparable. Once the models were trained, we evaluated them using standard metrics such as accuracy, ROC-AUC, precision, recall, and F1-score.

However, performance on a powerful workstation is only half the story. To understand which model is realistic for rural deployment, we also benchmarked the models on a Raspberry Pi 4B. For this, we measured latency, FPS, memory usage, and the final quantized model size. Finally, we combined these observations using a weighted scoring system to determine the most balanced and practical model for real-world use.

A. Pre-Processing and Data Augmentation

To improve model learning and reduce overfitting, images were preprocessed and augmented. Preprocessing included resizing images to (224 × 224 × 3) and, for some models, converting to a single grayscale channel. Data augmentation was applied using Keras' *ImageDataGenerator*, with transformations such as random rotations, flips, zooms, and minor shifts. This not only helped the models generalize better but also simulated real-world variations in chest X-rays, such as differences in orientation and scale

To further address class imbalance, class weights were applied during training, calculated using:

$$b_i = (1 / f_i) \times (f_n) / 2$$

where f_i is the number of images in class i and f_n is the total number of training images.

B. Model Architectures

Six deep learning architectures were selected for this study:

1) Sequential CNN (Baseline): A 22-layer model comprising Conv2D, SeparableConv2D, Batch Normalization, Max Pooling, Dropout, and Dense layers. ReLU was used for hidden layers and Sigmoid for the output. Exponentially decaying learning rates were employed to improve convergence.
2) VGG16 Autoencoder Standard VGG16 layers were retained up to the last convolutional block, while the final classification layer was removed to form an encoder for feature extraction.
3) Keras VisionTransformer(ViT): The model splits input images into patches and processes them using multiple transformer layers, with multi-head self-attention and feed-forward networks.
4) MobileNetV3-Small: Lightweight CNN using depthwise separable convolutions and attention modules, followed by global average pooling, dropout, and a dense output layer.
5) EfficientNet-V2S: Uses a series of balanced convolutional blocks with varying depth, width, and resolution. Global pooling and dropout layers precede the dense classification layer.
6) MobileViT Hybrid: Combines convolutional stems for low-level feature extraction with transformer blocks for long-range dependencies. Features are pooled and classified via a dense output layer.

C. Training Procedure

All models were trained on a Tesla P-100 GPU with the following configuration:

1) Optimizer: Adam`
2) Loss: Binary Cross-Entropy
3) Batch size: 16–32 (model-dependent)
4) Epochs: 50–100 with early stopping
5) Class weights applied to handle imbalance
6) Learning rate *reduction on plateau for faster convergence*

D. Performance Evaluation

After evaluating deployability, models were analyzed on classical predictive metrics using the held-out test set:

1) Accuracy (%): Overall classification correctness.
2) ROC-AUC: Measures the model's ability to distinguish between Pneumonic and Normal cases.
3) Precision: Proportion of correctly predicted Pneumonic cases among all predicted Pneumonic cases.
4) Recall (Sensitivity): Proportion of actual Pneumonic cases correctly identified.
5) F1-score*: Harmonic mean of precision and recall, balancing false positives and false negatives.*

This section provides a detailed comparison of predictive performance independently of deployment constraints, allowing a complete understanding of each model's strengths and weaknesses.

E. Raspberry Pi Deployment and Evaluation

A key goal of this study was to evaluate model performance on a resource-constrained device, specifically the Raspberry Pi 4B. Each trained model was converted to TensorFlow Lite (TFLite), with INT8 quantization applied where possible to reduce size and improve inference speed, while some models remained in FP32 for comparison.

On the Pi, models were tested using repeated single-image inferences to minimize measurement noise. Evaluation focused on:

1) Inference Latency (ms): time to process one image
2) Throughput (FPS): frames processed per second
3) Model Size (MB): memory footprint
4) Pi-side *Accuracy (%): performance on actual hardware*

This approach provides a realistic assessment of efficiency and reliability for edge deployment, ensuring that selected models are both accurate and practical for low-resource settings.

F. Raspberry Pi Deployment and Evaluation

To determine which deep learning model is most suitable for deployment on a Raspberry Pi, a weighted multi-criteria evaluation strategy was employed. Since classification accuracy alone cannot capture real-world usability on resource-constrained hardware, three deployment-critical metrics were considered:

- Pi Accuracy (40%) – reflecting diagnostic reliability.
- Latency and Throughput (FPS) (40%) – reflecting computational responsiveness.
- Model Size & Memory Efficiency (20%) – reflecting storage and processing affordability.

Because these metrics exist on different numerical scales, they were normalized to enable meaningful comparison. The normalization procedure for each metric was as follows:

1) Accuracy Score (higher is better)

Accuracy was converted directly into a 0–1 scale:

$$Acc_{Score} = \frac{Accuracy}{100}$$

2) Model Size Score (smaller is better)

TFLite model sizes were normalized using min–max inversion:

$$Size_{Score} = \frac{Size_{Max} - Size}{Size_{Max} - Size_{Min}}$$

3) Latency Score (lower is better)

Latency was normalized using min–max inversion:

$$Lat_{Score} = \frac{Lat_{Max} - Latency}{Lat_{Max} - Lat_{Min}}$$

4) Throughput/FPS Score (higher is better)

$$FPS_{Score} = \frac{FPS - FPS_{Min}}{FPS_{Max} - FPS_{Min}}$$

Since real-time performance depends jointly on **latency** and **FPS**, these two normalized values were averaged to obtain a single deployment-performance indicator:

$$Lat\ \&\ FPS = \frac{Lat_{Score} + FPS_{Score}}{2}$$

Finally, the overall weighted score for each model was calculated as:

$$Weighted\ Score = 0.4 \times Acc_{Score} + 0.4 \times Lat\ \&\ FPS + 0.2 \times Size_{Score}$$

All component scores were rounded to three decimal places in the results table to maintain clarity and interpretability. This approach ensures that the final ranking reflects diagnostic effectiveness, computational efficiency, and deployment feasibility on the Raspberry Pi platform.

G. Raspberry Pi Deployment and Evaluation

Finally, all six models are compared across the full set of evaluation metrics. This holistic assessment considers not only classification performance but also computational efficiency, deployability, and resource usage. By examining accuracy, precision-recall behavior, inference speed, throughput, and model size together, we can clearly understand how each architecture performs under real-world constraints. This integrated evaluation highlights the strengths and limitations of each model, enabling informed decisions on the most suitable architectures for practical deployment or future optimization.

V. RESULT

A. Raspberry Pi Deployment and Evaluation

The dataset used in this study is organized into three directories: training, validation, and testing. Each directory contains two subfolders: Normal, which holds images labeled as normal, and Pneumonia, which contains pneumonic cases. Any manual curation focused on removing inconsistent or duplicate images, while retaining high-quality radiographs. Since the dataset had already undergone initial screening, further image enhancement was unnecessary. Images were preprocessed and resized to match the input requirements of the models, and augmented where necessary to improve generalization.

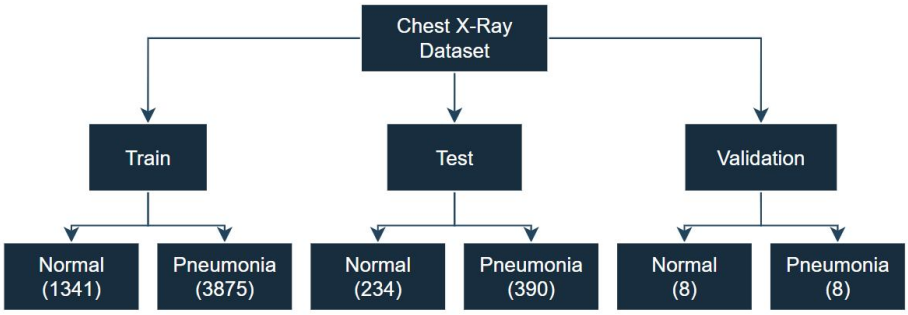


Figure 2 Dataset breakdown (before Data Augmentation)

B. Classification Results

All six models were first evaluated on the test dataset to measure their diagnostic performance. The metrics included test accuracy, F1-score, precision, recall, and ROC-AUC. This allowed us to gauge how well each model could detect pneumonia from chest X-rays under ideal conditions before deployment considerations.

Table 1 Comparative Performance of All Deep Learning Models Before TFLite Conversion.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	ROC-AUC (%)
Sequential CNN	78.04	74.56	98.46	84.85	92.53
VGG16 Autoencoder	83.49	79.95	98.2	88.14	94.91
ViT	71.63	68.84	99.74	81.45	84.82
MobileNetV3S	75.32	74.06	97.61	77.02	76.31
EfficientNet-V2S	79.16	82.33	84.97	84.45	82.54
MobileViT Hybrid	81.41	78.05	98.01	87.36	92.16

C. Deployment Performance on Raspberry Pi

To assess real-world feasibility, all trained models were converted to TensorFlow Lite format and deployed on a Raspberry Pi 4B. While some models were quantized to INT8 for faster inference and smaller size, others retained FP32 precision. The key metrics recorded were inference latency, throughput (FPS), TFLite model size, and Pi-side accuracy. These metrics capture both speed and resource constraints, which are critical for deployment in low-resource environments.

Table 2 Edge-Device Performance of All Models After TFLite Conversion on Raspberry Pi 4B.

Model	Total Params	TFLite Size	Latency (ms) *(Pi 3B)*	Throughput (FPS)	Pi Accuracy (%)
VGG16	15.1M	57.8MB FP32	312 ms	3.2 FPS	91.86
Sequential CNN	1.1M	4.23MB INT8	41 ms	24.3 FPS	77.216
MobileViT Hybrid	93K	0.38MB INT8	29 ms	34.4 FPS	78.33
EfficientNet-V2S	20.3M	76.8MB INT8	506 ms	1.97 FPS	76.46
MobileNetV3 Small	939K	1.30MB INT8	22 ms	45.4 FPS	71.65
ViT	~86M	~300MB+	NA	NA	N/A (Not runnable)

©IJRASET: All Rights are Reserved

D. Weighted Scoring and Ranking

To determine the most deployment-appropriate model for Raspberry Pi, a weighted multi-criterion scoring system was applied. Each model was evaluated based on Pi accuracy, latency & FPS, and model size. Normalization was used to ensure metrics on different scales could be combined. The final weighted score provides a clear ranking, balancing diagnostic performance with computational efficiency and resource constraints.

Table 3 Weighted Scoring and Final Ranking of All Models.

Model	Accuracy Score	Size Score	Latency Score	FPS Score	Lat & FPS Combined	Weighted Score	Rank
MobileViT Hybrid	0.783	1	0.986	0.747	0.866	0.86	1
MobileNetV3 Small	0.716	0.988	1	1	1	0.82	2
Sequential CNN	0.772	0.95	0.961	0.514	0.737	0.782	3
VGG16 Autoencoder	0.918	0.249	0.401	0.028	0.215	0.62	4
EfficientNet-V2S	0.765	0	0	0	0	0.153	5
ViT (Not runnable on Pi)	N/A	N/A	N/A	N/A	N/A	N/A	6

E. Comparitive Analysis

Among the six evaluated models, VGG16 Autoencoder and Sequential CNN delivered strong accuracy but suffered from larger size or moderate efficiency. EfficientNet-V2S and ViT were too computationally heavy for Raspberry Pi, with ViT failing to run altogether. In contrast, MobileNetV3-Small and MobileViT Hybrid demonstrated excellent edge performance, achieving low latency, high FPS, and minimal storage requirements. MobileViT Hybrid stood out by providing the best overall balance of Pi-side accuracy, speed, and compactness, outperforming all other architectures in deployment-focused metrics.

VI. CONCLUSION AND FUTURE WORK

This study evaluated six deep learning models for pneumonia detection, considering both classification performance and deployment feasibility on Raspberry Pi 4B. While models like VGG16 Autoencoder and ViT achieve high accuracy, their larger size and slower inference limit practical deployment. Lightweight models such as MobileNet V3S and MobileViT Hybrid offer a better balance, and based on our weighted scoring of accuracy, latency, throughput, and model size, **MobileViT Hybrid** is identified as the most suitable for edge deployment.

Future work could explore model optimization techniques like pruning, quantization, or knowledge distillation to further reduce latency and memory footprint. Incorporating attention-based pre-processing, multi-modal patient data, or evaluating other low-cost hardware platforms can enhance robustness and real-world applicability of pneumonia screening systems.

VII. ACKNOWLEDGMENT

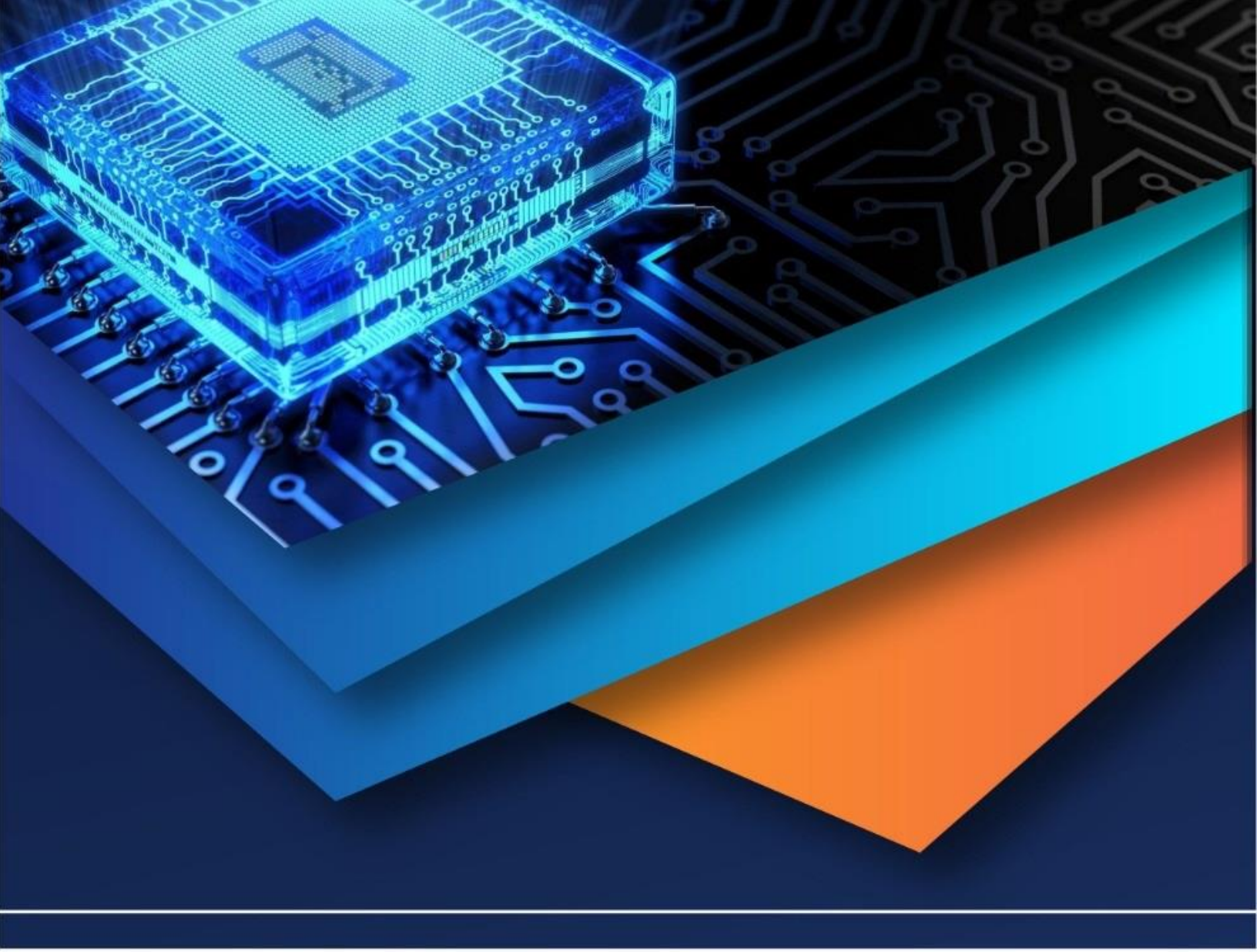
The author sincerely expresses gratitude to the Head of the Department, Dr. Mohammad Amjad, Jamia Millia Islamia, for his valuable guidance, encouragement, and continuous support throughout this research work. Appreciation is also extended to the faculty members and the institution for providing the necessary resources and a conducive research environment.

REFERENCES

[1] Roy, S., Suresh, A., Sahu, P., & Gupta, T. R. (2025). Novel pooling-based VGG-Lite for pneumonia and Covid-19 detection from imbalanced chest X-ray datasets. arXiv preprint arXiv:2504.07468

[2] Xia, Zunhui & Li, Hongxing & Lan, Libin. (2025). MedFormer: Hierarchical Medical Vision Transformer with Content-Aware Dual Sparse Selection Attention. 10.48550/arXiv.2507.02488.

[3] Saber, A., Parhami, P., Siahkarzadeh, A., Fateh, M., & Fateh, A. (2024). Efficient and accurate pneumonia detection using a novel multi-scale transformer approach. arXiv preprint arXiv:2408.04290
[4] Bukhari, M. T. (2024). Efficacy of lightweight Vision Transformers in diagnosis of pneumonia. medRxiv.
[5] Siddiqi R, Javaid S. Deep Learning for Pneumonia Detection in Chest X-ray Images: A Comprehensive Survey. J Imaging. 2024 Jul 23;10(8):176. doi: 10.3390/jimaging10080176. PMID: 39194965; PMCID: PMC11355845.
[6] Al Reshan, R., Alzubaidi, L., Santamaría, J., Albahri, O. S., Fadhel, M. A., & Albahri, A. S. (2023). A comprehensive review of deep learning-based chest X-ray analysis for detecting thoracic diseases. Artificial Intelligence in Medicine, 140, 102483.
[7] Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., & Xie, S. (2022). A ConvNet for the 2020s (ConvNeXt). Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 11976–11986. [Often cited across 2022–2023 for medical imaging backbones.
[8] Pandey, S., Gunjan, V. K., & Nandi, G. C. (2022). Lightweight deep learning approach for pneumonia detection on embedded devices using quantized MobileNetV2. IEEE Access, 10, 120345–120357.
[9] Mehta, H., Pandey, S., & Nandi, G. C. (2022). Real-time low-power pneumonia detection on Raspberry Pi using quantized convolutional neural networks. Journal of Real-Time Image Processing, 19, 1781–1795.
[10] Almutairi, A., El Bashir, M. K., Alharbi, R., & Alsubaie, N. (2022). Deep learning models for COVID-19 and pneumonia detection from chest X-ray images: A comparative study. Computers in Biology and Medicine, 141, 105153.
[11] Tan, M., & Le, Q. V. (2021). EfficientNetV2: Smaller models and faster training. Proceedings of the 38th International Conference on Machine Learning (ICML), 10096–10106. PMLR.
[12] Apostolopoulos, I. D., & Mpesiana, T. A. (2020). COVID-19: Automatic detection from X-ray images utilizing transfer learning with convolutional neural networks. Physical and Engineering Sciences in Medicine, 43, 635–640.
[13] Irvin, J., Rajpurkar, P., Ko, M., Yu, Y., Ciurea-Ilcus, S., Chute, C., Marklund, H., Haghgoo, B., Ball, R. L., Shpanskaya, K., Seekins, J., Mong, D. A., Halabi, S. S., Sandberg, J. K., Jones, R., Larson, D. B., Langlotz, C. P., Patel, B. N., Lungren, M. P., & Ng, A. Y. (2019). CheXpert: A large chest radiograph dataset with uncertainty labels and expert comparison. Proceedings of the AAAI Conference on Artificial Intelligence, 33, 590–597.
[14] Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., & Summers, R. M. (2017). ChestX-ray8: Hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 3462–3471.
[15] Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., Ding, D., Bagul, A., Ball, R. L., Langlotz, C., Shpanskaya, K., Lungren, M. P., & Ng, A. Y. (2017). CheXNet: Radiologist-level pneumonia detection on chest X-rays with deep learning. arXiv:1711.05225.
[16] Joshua, C., Karkala, S., Hossain, S., Krishnapatnam, M., Aggarwal, A., Zahir, Z., Pandhare, H. V., & Shah, V. (2025). Latency-Accuracy Trade-off Analysis in Edge-Based Object Detection Pipelines. Published June 12, 2025.
[17] Kermany, Daniel; Zhang, Kang; Goldbaum, Michael (2018), “Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification”, Mendeley Data, V2, doi: 10.17632/rscbjbr9sj.2



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)