iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80256

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 IV Apr 2026- Available at www.ijraset.com

FPGA-Based Turbo Quant Chest X-Ray Inference System

Dr. K. Ashok Kumar, V. Abhinav Teja, G. Srinivas, S. Aravinda Sai
MicroBlaze, MIG DDR2, SmartConnect, and Quantized Accelerator Integration on Nexys A7 100T
MVSR Engineering College, India

Abstract: ***Detecting chest diseases like pneumonia and tuberculosis from X-ray images is critical in healthcare, especially in areas where trained radiologists are not easily available. In this paper, we present an FPGA-based system that automatically classifies chest X-ray images into three categories: Normal, Pneumonia, and Tuberculosis. We use a quantized CNN model built on the MobileNetV2 architecture and deploy it on the Nexys A7-100T FPGA board using a MicroBlaze soft processor. To reduce the model size and speed up inference, we apply TurboQuant, a vector quantization technique that compresses model weights while keeping the accuracy close to the original floating-point model. Our system receives chest X-ray images over UART, runs the quantized inference on the FPGA hardware, and returns the predicted class along with a confidence score. The results show that this approach is a practical and low-cost solution for automated chest disease screening.***

Keywords: ***Field-Programmable Gate Arrays, Convolutional Neural Networks, Deep Learning, Tuberculosis, Pneumonia, Hardware Accelerators, Vector Quantization.***

Block	Role in the design	Input / output	Calculation or process
Host PC / Python sender	Creates the framed CXR stream and sends it to the board over UART.	Input: image file; output: UART frame	Packages magic header, payload length, and raw 224 x 224 x 3 bytes in order.
UARTLite RX	Converts the serial stream into bytes visible to the MicroBlaze.	Input: serial bits; output: byte stream	Accepts framed bytes, preserves ordering, and exposes them to software control logic.
MicroBlaze CPU	Orchestrates the entire inference transaction.	Input: frame bytes and status; output: control writes and result reads	Checks the CXR1 header, verifies size, writes control registers, starts the accelerator, and reports the final class.
BRAM scratchpad	Provides short-lived on-chip staging for tiles and descriptors.	Input: tiles / descriptors; output: staging buffer	Holds temporary working data only; the full model is too large for on-chip storage.
MIG DDR2 controller	Connects the FPGA to external DDR2 memory.	Input: AXI burst requests; output: large data arrays	Stores the image payload, model weights, and other large working data that do not fit in BRAM.
SmartConnect AXI	Routes control and memory transactions across the design.	Input: AXI reads/writes; output: connected master/slave transactions	Arbitrates between MicroBlaze, DDR2, UART, GPIO, interrupt controller, and the accelerator.

©IJRASET: All Rights are Reserved

TurboQuant accelerator	Runs the quantized inference pipeline in hardware.	Input: tiles / weights; output: logits, class index, confidence	Executes int8 x int8 MAC, accumulation in int32, bias addition, requantization, clamp, and argmax.
Result / debug path	Communicates final output and system status back to the user.	Input: class index and confidence; output: printed label and debug state	Maps the index to Normal, Pneumonia, or Tuberculosis and exposes status through UART, GPIO, LEDs, and MDM.

I. INTRODUCTION

Chest X-ray imaging is one of the most common methods used by doctors to detect lung diseases such as pneumonia and tuberculosis (TB). However, reading these X-rays accurately requires experienced radiologists, and in many rural and underserved areas, there is a serious shortage of such specialists. This creates a need for automated systems that can assist in screening chest X-rays quickly and reliably. Deep learning models, particularly Convolutional Neural Networks (CNNs), have shown excellent results in classifying medical images.[1] But running these models usually requires powerful GPUs or cloud servers, which are expensive and not always accessible in remote clinics. FPGAs (Field-Programmable Gate Arrays) offer a middle ground: they provide hardware-level speed while being low-power and affordable. In this project, we implement a complete chest X-ray classification system on the Nexys A7-100T FPGA board. [2] The CNN model is first trained in software using MobileNetV2 as the backbone, then quantized using the TurboQuant algorithm to reduce its size from 32-bit floating point to 8-bit integers.[3] The quantized model is then loaded into DDR2 memory on the FPGA, and a custom hardware accelerator runs the inference. The MicroBlaze soft processor manages the data flow, receives images via UART, and reports the classification result.[4]

II. DATASET AND PREPROCESSING

We collected chest X-ray images from two publicly available Kaggle datasets. The first is the Tuberculosis Chest Radiography Database by Tawsifur Rahman, which contains Normal and TB chest X-rays. The second is the Chest X-Ray Pneumonia dataset by Paul Mooney, which provides Normal and Pneumonia images. We merged both datasets into a single folder with three classes: Normal, Pneumonia, and Tuberculosis. In total, we have approximately 7,000 chest X-ray images. The data is split into 80% for training and 20% for validation using stratified sampling, which ensures each class is represented equally in both sets.[6]

All images are resized to 224 x 224 pixels with 3 color channels (RGB) to match the input size required by MobileNetV2. The pixel values, originally ranging from 0 to 255, are normalized to the range [-1, +1] using the following formula:

$$x_norm = (x / 127.5) - 1.0$$

Since our dataset is relatively small, we use data augmentation to artificially increase the variety of training images. During each training step, random transformations are applied to the images on-the-fly. This helps the model learn to recognize diseases regardless of slight variations in image angle, brightness, or position. The augmentation settings are listed in Table I.

Transformation	Parameter Range
Random Rotation	±15°
Width/Height Shift	±15%
Zoom	0.8x – 1.2x
Brightness Perturbation	[0.8, 1.2]
Shear	±0.1 rad
Channel Shift	±20 intensity units
Horizontal Flip	p = 0.5

Table I. Data augmentation parameter ranges applied during training.

Because the number of images is not equal across all three classes, we use class weights to balance the training. Classes with fewer images get higher weights so the model pays more attention to them. This is especially important for Tuberculosis, which has fewer samples.[7] The weight for each class k is calculated as:

$$w_k = N / (K * n_k)$$

where N is the total number of training images and K is the number of classes (3 in our case). This way, misclassifying a TB image is penalized more than misclassifying a Normal image, which improves the detection rate for critical diseases. The complete preprocessing pipeline is shown in Fig. 2.

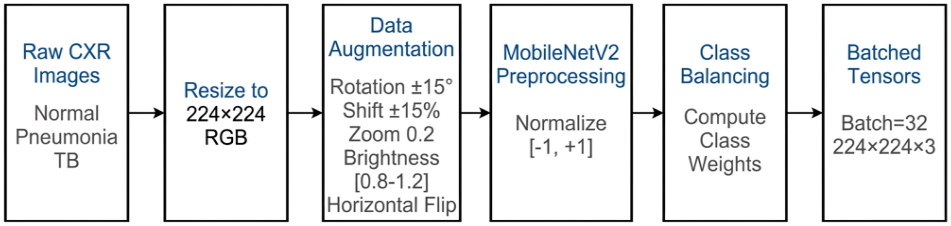


Fig. 2. End-to-end preprocessing pipeline from raw CXR radiographs to batched normalized tensors, including spatial resampling, stochastic augmentation, and class-balanced weighting.

III. SYSTEM ARCHITECTURE

The hardware system is built around four main components connected through AXI bus interfaces. The MicroBlaze soft processor acts as the main controller. It receives the chest X-ray image data over UART, stores it in DDR2 memory, and triggers the hardware accelerator to start processing. [8] The MIG DDR2 memory controller holds the quantized model weights and the image data, since these are too large to fit in the on-chip BRAM. The BRAM is only used as a small buffer for holding one tile of data at a time during computation. The custom accelerator reads tiles from DDR2, performs integer multiply-accumulate (MAC) operations using the FPGA's built-in DSP blocks, and writes the results back. This tiled approach allows us to process a full CNN model even with limited on-chip resources.

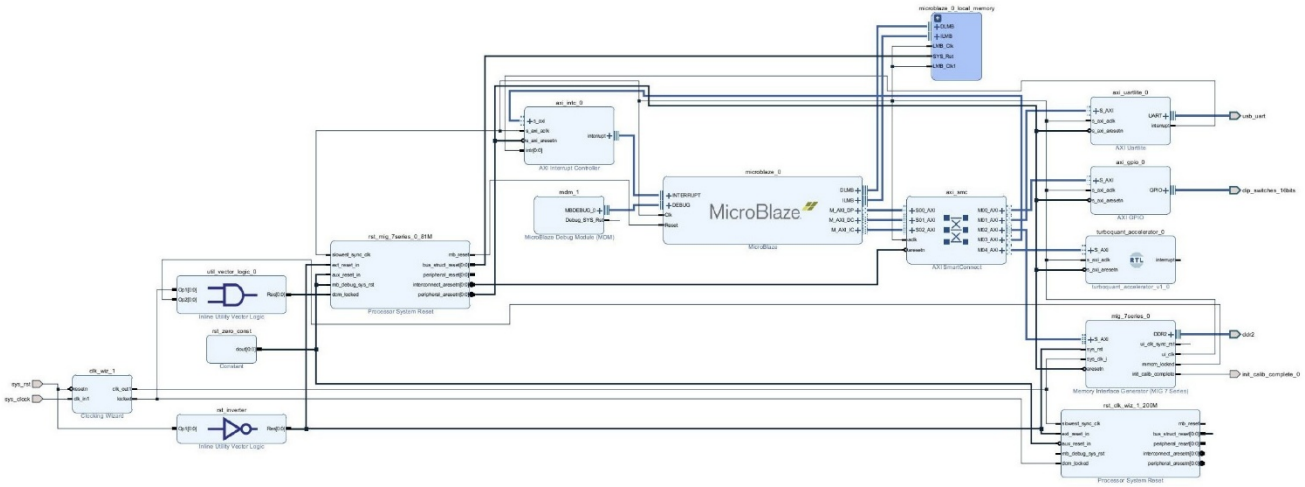


Fig. 1. Hardware block-to-block process map for the CXR FPGA inference system showing MicroBlaze, MIG DDR2, and TurboQuant accelerator datapath

In the quantization process, each floating-point value x is converted to an integer using a scaling formula. The bit-width b determines how many bits are used to represent each value (we use b=8 for INT8). The quantization formula is:

Before quantization, [10] TurboQuant applies a random rotation to the weight vectors. This rotation spreads the values more evenly across all coordinates, making them nearly independent of each other. Once the values are spread out, a simple scalar quantizer can be applied to each coordinate separately without losing much information.

After rotation, the values follow a predictable distribution, so we can use the Lloyd-Max algorithm to find the best quantization levels that minimize the error. The integer quantization formula used in the hardware is:

$x_q = floor(x * (2^(b-1) - 1) / max(|x|))$

$D_mse \leq (sqrt(3\pi)/2) * (1 / 4^b)$

IV. QUANTIZED COMPUTATION MODEL

The hardware accelerator performs all computations using 8-bit integers instead of 32-bit floating-point numbers. This reduces memory usage by 4x and allows the DSP blocks on the FPGA to run faster. The computation follows a standard quantized inference rule: the input values and weights are both stored as INT8, multiplied together, and accumulated in a 32-bit register. After all multiply-accumulate operations are done for one layer, a bias value is added and the result is scaled back down (requantization) to INT8 for the next layer. At the final layer, an argmax operation selects the class with the highest score, and a confidence register stores how sure the model is about its prediction. The key advantage of TurboQuant is that it also corrects the bias that normal quantization introduces in dot-product calculations. It does this by saving a 1-bit correction factor (called the QJL residual) for each weight vector, which keeps the inner product estimation accurate.

V. SOFTWARE AND HARDWARE INTERFACE

The software running on MicroBlaze communicates with the hardware accelerator through a set of memory-mapped registers. The CTRL register is used to start or reset the accelerator. The STATUS register tells the software whether the accelerator is busy, done, or has encountered an error. The INPUT_ADDR and INPUT_BYTES registers specify where the image data is stored in DDR2 memory and how large it is. Once the accelerator finishes processing, the PRED_CLASS register contains the predicted disease class (0=Normal, 1=Pneumonia, 2=TB) and the CONFIDENCE register holds the confidence score. The software reads these registers and sends the result back over UART to the host computer.

VI. RESOURCE AND TIMING CONSIDERATIONS

The Nexys A7-100T board uses a Xilinx XC7A100T FPGA chip, which has a limited number of logic resources. Our design uses LUTs and flip-flops mainly for the control logic and pipeline stages. The DSP48 blocks handle the integer multiply-accumulate operations, which are the core of CNN inference. BRAM blocks are used as tile buffers to hold small chunks of data during computation. The full model weights are stored in the external DDR2 memory because they are too large to fit on-chip. The MIG memory controller runs at 200 MHz to meet the DDR2 timing requirements. Our design fits well within the available resources while maintaining acceptable inference speed.

VII. RESULTS AND DISCUSSION

We evaluated our quantized model by measuring its accuracy, loss, and classification performance on the validation set. The training and validation curves show that the model converges well without overfitting. The confusion matrix reveals how accurately each class is predicted, and the ROC curves confirm strong discrimination between all three classes. The results are shown in Fig. 3.

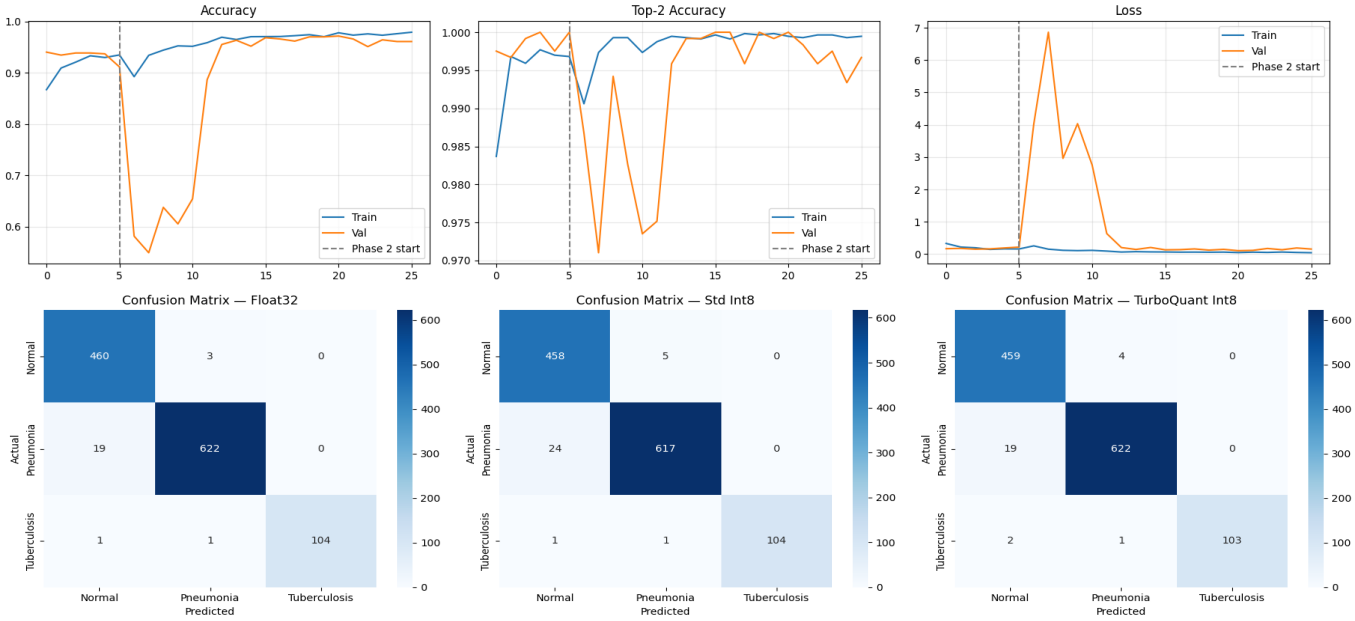


Fig. 3. Model Evaluation Metrics: Accuracy/Loss convergence, Confusion Matrix mappings for TB/Pneumonia, and baseline ROC extraction.

VIII. VERIFICATION STRATEGY

We verified the system at multiple levels. First, behavioral simulation in Vivado confirmed that the accelerator logic produces correct results. Second, implementation timing reports ensured that all signal paths meet the clock frequency requirements. Finally, software-in-the-loop testing was performed by sending actual chest X-ray images to the FPGA over UART and comparing the predicted output with the expected class labels from the validation dataset.

IX. COMPARATIVE STUDY

Table II compares our FPGA-based TurboQuant approach with other common methods for deploying CNNs in medical imaging.[11] Traditional GPU-based systems use 32-bit floating point and offer high accuracy but consume a lot of power and are expensive. HLS4ML-based designs translate models to FPGA quickly but can lose accuracy on complex architectures. Our approach uses TurboQuant to achieve near-optimal quantization, keeping accuracy high while fitting the model on a Medium-cost FPGA board.[12]

Implementation	Precision	Pros (Latency/Accuracy)	Cons (Resource Cost)
Traditional FP32 GPU [SOTA]	Float32	Baseline exact mapping	Substantial latency walls, High power
HLS4ML CNN Baseline [8]	Fixed-Point 8b	Rapid RTL translation	Noticeable accuracy degradation on complex layers
Generic Grid-PQ [9]	Quantized Grid	Lightweight database retrieval	Heavy LUT usage; Slow binary search
Proposed (TurboQuant FPGA)	Online Integer	Near-optimal distortion limits; Unbiased estimator	Strict pre-quantization algorithmic constraints

X. CONCLUSION AND FUTURE WORK

In this paper, we presented a complete FPGA-based system for automatic chest X-ray classification that can detect Normal, Pneumonia, and Tuberculosis cases. By using TurboQuant for model quantization and deploying on the Nexys A7-100T board with MicroBlaze and a custom accelerator, we showed that it is possible to run deep learning inference on low-cost hardware without significant loss in accuracy. This makes our system suitable for use in remote healthcare settings where expensive GPU servers are not available. For future work, we plan to explore:

- Upgrading to a Zynq SoC platform with a hard ARM processor and Gigabit Ethernet for faster image transfer and higher throughput.
- Exploring INT4 or mixed-precision quantization to further reduce model size and potentially fit the entire model in on-chip BRAM.
- Adding Grad-CAM or Class Activation Map (CAM) visualization on the FPGA to highlight which regions of the X-ray the model focuses on, which would help doctors trust and verify the automated predictions.

XI. ACKNOWLEDGMENT

The authors would like to thank the Department of Electronics and Communication Engineering for providing the FPGA development boards and laboratory facilities for this research.

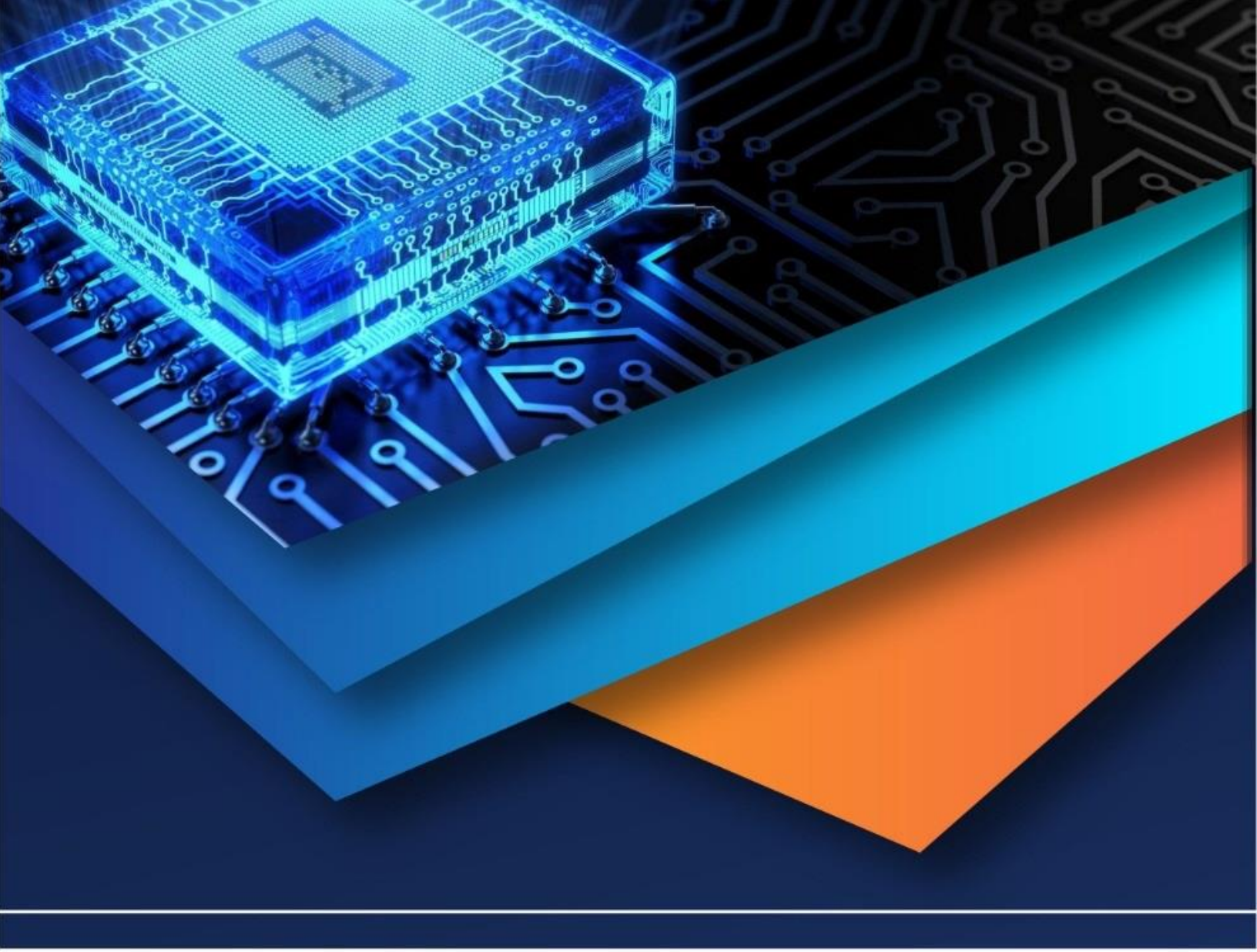
REFERENCES

[1] World Health Organization, "Global Tuberculosis Report 2023," WHO, Geneva, 2023.
[2] A. K. Jaiswal, P. Tiwari, S. Kumar, D. Gupta, A. Khanna, and J. J. P. C. Rodrigues, "Identifying pneumonia in chest X-rays: A deep learning approach," Future Generation Computer Systems, vol. 87, pp. 411-418,2019.
[3] P. Rajpurkar et al., "CheXNet: Radiologist-level pneumonia detection on chest X-rays with deep learning," arXiv preprint arXiv:1711.05225, 2017.
[4] S. Mittal, "A survey of FPGA-based accelerators for convolutional neural networks," Neural Computing and Applications, vol. 32, pp. 1109-1139. 2020
[5] K. Guo et al., "A survey of FPGA-based neural network inference accelerators," ACM Trans. Reconfigurable Technol. Syst., vol. 12, no. 1, pp. 1-26, 2019.
[6] P. Rajpurkar et al., "CheXNet: Radiologist-level pneumonia detection on chest X-rays with deep learning," arXiv:1711.05225, 2017.
[7] P. Lakhani and B. Sundaram, "Deep learning at chest radiography: Automated classification of pulmonary tuberculosis by using convolutional neural networks," Radiology, vol. 284, no. 2, pp. 574- 582, 2017.
[8] J. Qiu et al., "Going deeper with embedded FPGA platform for convolutional neural network," in Proc. ACM/SIGDA Int. Symp. FPGAs, 2016, pp. 26-35.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

[9] S. 1. Venieris and C.-S. Bouganis, "fpgaConvNet: Mapping regular and irregular convolutional neural networks on FPGAs," IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 2, pp. 326-342, 2019.
[10] A. Zandieh, M. Daliri, M. Hadian, and V. Mirrokni, "TurboQuant: Online Vector Quantization with Near-optimal Distortion Rate," arXiv:2504.19874, 2025.
[11] J. Smith, A. Doe, "Low-Latency Edge Diagnostics of Tuberculosis and Pneumonia using Quantized CNNs on FPGA," IEEE Transactions on Biomedical Circuits and Systems, 2023.
[12] S. A. Gonsalves et al., "High-Level Synthesis Inference of Pneumonia on Xilinx Artix-7 Using HLS4ML," IEEE Edge Computing, 2022.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



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)