

An Edge-Intelligent Framework for Multi-Class Ocular Disease Detection using Optimized Transfer Learning Models

Sanghmitra Gaikwad¹ , Yogita Vaidya¹

¹Department of Electronics Engineering (Embedded Systems and Computing), COEP Technological University, Pune, India

Abstract - Millions of people worldwide lose vision to conditions that, if caught early, are treatable, yet specialized eye care remains out of reach for many low-resource regions. This work presents an edge-deployed AI framework built to classify multiple eye diseases directly on embedded hardware, closing that access gap. Eight deep learning architectures were benchmarked on a public fundus image dataset spanning four categories (normal, cataract, glaucoma, and diabetic retinopathy): VGG16, ResNet-50, Inception V3, DenseNet-169, ConvNeXt, Vision Transformer, Swin Transformer, and a CNN-Transformer hybrid (HybridVisionNet). DenseNet-169 achieved the highest test accuracy with 97.16%, ahead of ConvNeXt (96.21%), ViT (95.02%), HybridVisionNet (94.79%), Swin Transformer (92.56%), Inception V3 (90.52%), ResNet-50 (89.73%), and VGG16 (89.26%). This model was then ported to an NVIDIA Jetson Orin Nano and optimized with TensorRT using FP16/INT8 quantization. On a separate 200-image test set, the deployed engine achieved 98.50% accuracy (compared with [X]% for the FP32 model on the same images) at an average latency of under 15 ms (over 65 FPS), indicating that accuracy was preserved on the edge device. A web-based interface was also built so that users can upload a fundus image and receive an immediate screening prediction; it is intended as a screening aid, not a diagnostic device. A preliminary external test on 99 fundus images from a partner hospital gave 100% accuracy (99/99), although this small set does not establish generalizability. Because all inference happens on-device, the framework runs fully offline and keeps patient data local, pointing toward edge AI as a practical, scalable path for ophthalmic screening in underserved communities.

Keywords Edge AI · Ocular disease detection · Transfer learning · Fundus image classification · Embedded GPU · Medical image analysis · TensorRT optimization · Model quantization · Deep learning · Clinical decision support

1 INTRODUCTION

A significant amount of avoidable vision impairment and blindness globally is caused by ocular illnesses such as diabetic retinopathy, glaucoma, and cataracts, which together constitute a growing global health concern. Diabetic retinopathy alone affects more than 100 million people worldwide [1]. Glaucoma, dubbed the “silent thief of sight,” is also one of the primary causes of irreversible blindness because it is asymptomatic in the early stages. Early and accurate detection of these conditions is therefore of utmost importance, as timely clinical intervention can significantly mitigate the risk of permanent vision loss and improve long-term patient outcomes. The clinical importance of early screening is well known; however, qualified ophthalmological services are critically scarce in rural, remote, and resource-constrained parts of the world. The shortage of trained specialists, the high cost of diagnostic equipment, and insufficient healthcare infrastructure create a significant gap between the demand for and availability of timely eye care. This unmet demand underlines the need for automated, cost-effective, and accessible diagnostic solutions capable of supporting large-scale ocular disease screening without the constant presence of specialist staff.

In recent years, deep learning has revolutionized the analysis of medical images, achieving diagnostic accuracy comparable to that of clinicians in several well-studied imaging tasks [1]. For the classification of retinal fundus images, convolutional neural networks (CNNs) have demonstrated strong performance. Transfer learning, which refines models pre-trained on extensive visual datasets for domain-specific medical tasks, achieves strong generalization with comparatively little labelled data and has further accelerated progress [1–5]. Architectures such as VGG16 and VGG19 have been applied to ocular disease detection with good baseline performance [6, 7], while ResNet-based models have achieved competitive classification accuracy on fundus datasets [9, 10]. Meanwhile, attention-based architectures such as the Vision Transformer (ViT) and the hierarchical Swin Transformer introduced powerful mechanisms for modelling long-range spatial dependencies in high-resolution medical images, and have been applied to ocular image classification [23, 24, 28, 29]. The architectural landscape

has been broadened further by more recent designs such as ConvNeXt and hybrid CNN-Transformer architectures, which combine the global context modelling capability of transformers with the local feature extraction power of CNNs [11, 16].

Despite these improvements, most existing deep learning-based ocular disease classification systems are designed for cloud inference. Such architectures impose critical limitations on real-world clinical deployment: high inference latency, dependence on stable, high-bandwidth internet connections, and serious concerns regarding patient data privacy and regulatory compliance [21, 22]. These constraints render cloud-dependent systems infeasible in remote primary care settings, mobile screening units, and low-resource hospitals where network reliability cannot be assured. Edge AI, the practice of running deep learning inference on embedded or portable hardware at the point of care, provides an attractive solution. Computing locally eliminates network dependence, reduces inference latency, and ensures that sensitive patient data does not leave the clinical setting [26, 27]. However, deploying high-accuracy, multi-class ocular disease classifiers on resource-constrained embedded hardware with production-level performance remains a substantially underexplored problem in the existing literature.

Driven by these opportunities and challenges, this research presents a complete edge-intelligent diagnostic system for automated multi-class ocular disease categorization from fundus images. The framework systematically benchmarks eight deep learning architectures, spanning classical CNNs (VGG16, ResNet-50) [6, 9], modern convolution-based networks (Inception V3, DenseNet-169, ConvNeXt) [2, 7, 11], a CNN-Transformer hybrid (HybridVisionNet) [16], and pure transformer architectures (Vision Transformer and Swin Transformer) [23, 28], on a four-class fundus image dataset encompassing normal, cataract, glaucoma, and diabetic retinopathy cases. The best-performing architecture is subsequently optimized and deployed on the NVIDIA Jetson Orin Nano embedded GPU platform [27] using TensorRT acceleration and mixed-precision quantization (FP16/INT8). To close the gap between algorithmic research and clinical accessibility, a web-based diagnostic interface is developed and deployed so that clinicians and non-specialist users can perform real-time fundus image classification without specialized hardware or technical skills [21, 22]. The framework is further tested on a small set of fundus images collected at a partner hospital, which gives preliminary evidence of its behaviour beyond the public benchmark dataset.

The main contributions of this work are summarized as follows: (1) a comparative evaluation of eight architectures spanning CNNs, transformer models, and a CNN-Transformer hybrid on a four-class ocular disease fundus image dataset [1–5, 16, 23, 28]; (2) a two-stage fine-tuning protocol (frozen-backbone training followed by end-to-end fine-tuning) applied uniformly to all models; (3) an edge deployment pipeline using TensorRT graph optimization and FP16/INT8 quantization on the NVIDIA Jetson Orin Nano [27], which retained classification accuracy (98.50% on a 200-image test set) at an average latency of under 15 ms; (4) a web-based interface that allows fundus image upload and real-time multi-class screening prediction without cloud connectivity [21, 22]; and (5) a preliminary external test on 99 hospital-collected fundus images.

2 RELATED WORK

2.1 CNN-based approaches for ocular disease classification

A substantial body of research has investigated automated ocular disease detection from fundus images using convolutional neural network architectures. Gupta et al. [1] give a comprehensive overview of deep learning algorithms for ocular and non-ocular illnesses from colour fundus images, providing key baselines and evaluation metrics for the community. Transfer learning with VGG, ResNet and Inception architectures has been extensively studied for retinal disease diagnosis, consistently achieving strong baseline performance over multiple datasets [3, 5, 7]. In particular, Singh et al. [7] studied VGG19 and InceptionV3 for ocular disease detection and achieved competitive accuracy by tuning hyperparameters at the architecture level. Together, these pioneering works confirm the potential of pre-trained CNNs as effective feature extractors in the ophthalmic domain, particularly where labelled medical data is limited. Other studies have explored multi-class ocular disease classification using texture features [8], deep learning pipelines [12, 19], LBP-KNN [20], self-supervised versus supervised learning [17], and smartphone-based detection of ocular surface disease [13, 14].

More advanced convolutional architectures have achieved higher classification accuracy. DenseNet-based models are particularly suitable for medical image analysis with limited training data [2] because their dense connectivity patterns enable efficient feature reuse and improved gradient propagation. Jibon et al. [2] demonstrated the success of ensemble transfer learning in classifying multiple types of eye diseases by combining various CNN backbones, and also addressed the detection of retinal vascular conditions. Ensemble approaches that combine multiple architectures with autoencoder-based feature enhancement have shown better robustness and reduced misclassification rates in ocular screening tasks [6, 18]. ResNet-50 has also been benchmarked for ocular disease classification by Chiranjeevi et al. [9], who report its comparative advantages over lightweight architectures such as SqueezeNet on standard fundus datasets. Islam et al. [10] extended ResNet-based modelling by adding wavelet decomposition and diffusion-based augmentation to achieve richer features in low-data settings.

2.2 Transformer and hybrid architectures

The emergence of Vision Transformers (ViT) has transformed the analysis of medical images by replacing local convolutional operations with global self-attention. Vision transformer-based architectures can capture long-range spatial dependencies in ocular images and have been reported to outperform conventional CNNs in some ocular classification studies [23, 24, 29], although the outcome depends on dataset size and pre-training. Ali and Islam [24] employed a hyper-tuned ViT model with explainability features for eye disease detection, highlighting the diagnostic interpretability benefits of attention-based architectures. Kamal et al. [29] demonstrated ViT-based classification with Grad-CAM visualization for ocular health screening, emphasizing the clinical relevance of transformer-derived attention maps. The Swin Transformer extends the ViT paradigm through a hierarchical shifted-window attention mechanism that improves computational efficiency and enables multi-scale feature extraction, making it practically viable for high-resolution retinal fundus images [28].

Feature fusion techniques that combine transformer and CNN representations have brought further diagnostic improvements by exploiting the complementary strengths of local texture modelling and global context reasoning [15, 25]. Lian et al. [25] proposed a modified conformer with sparse attention for diabetic retinopathy classification from fundus images, achieving competitive performance through attention sparsification. ConvNeXt is a recent re-exploration of pure convolutional network design that incorporates structural insights from transformer architectures, including depthwise convolutions, inverted bottlenecks, and larger kernel sizes, while maintaining the inductive bias and computational efficiency associated with CNNs [11]. The HybridVisionNet framework proposed by Kilic [16] shows the benefit of explicitly combining convolutional feature extraction with transformer-based semantic reasoning through cross-attention mechanisms, achieving competitive accuracy on multi-class fundus classification benchmarks and motivating the inclusion of hybrid architectures in comparative evaluations.

2.3 Edge AI and embedded deployment for medical imaging

Despite significant algorithmic advances, most current systems for ocular disease classification rely solely on cloud-based inference pipelines. Chandran et al. [21] and Baig et al. [22] point out the real-world limitations of such methods, including high inference latency, reliance on stable internet connectivity, and patient data privacy concerns, all of which impose fundamental constraints on deployment in remote, rural, and resource-constrained clinical settings. Smartphone-based diagnostic tools have mitigated some of the accessibility issues [22], but they remain limited by mobile hardware constraints and network reliance for model inference. Edge AI solutions are increasingly attractive for addressing these deployment challenges. For real-time biomedical signal processing and image classification on embedded platforms such as the NVIDIA Jetson family, lightweight deep learning models have been implemented effectively [26, 27]. A review of edge AI deployment techniques for healthcare by Prakash et al. [26] showed that TensorRT-based optimization coupled with FP16/INT8 quantization can drastically reduce inference latency on embedded GPUs while largely maintaining model accuracy. The NVIDIA Jetson Orin Nano platform [27] provides a potent blend of computational power and energy efficiency for these deployments, supporting both FP16 and INT8 precision modes with TensorRT acceleration.

2.4 Research gaps and positioning of this work

A thorough analysis of the current literature identifies two important and largely unfilled gaps. First, the literature on ophthalmic disease detection lacks thorough and equitable benchmarking of various model families, including

transformer architectures, hybrid models, modern convolution-based networks, and classical CNNs, within a single experimental framework, even though numerous studies have assessed individual architectures separately. Second, the combination of edge-optimized inference with an easily accessible, web-based diagnostic interface intended for clinical use by non-specialists is rarely addressed together in the ophthalmic literature. The majority of current solutions either address hardware deployment without offering clinician-accessible interfaces verified on real-world hospital data, or concentrate on algorithmic accuracy without addressing deployment restrictions. By putting forth a comprehensive pipeline that includes systematic architecture benchmarking, hardware-optimized edge deployment on the NVIDIA Jetson Orin Nano, a fully operational web-based diagnostic interface, and a preliminary external test on hospital-collected fundus data, this work addresses both gaps.

3 METHODOLOGY

3.1 Proposed framework for edge deployment

The proposed solution creates a comprehensive pathway from raw fundus images to real-time clinical diagnosis by combining an effective edge deployment with a strict architecture evaluation pipeline. Fig. 1 illustrates the overall system architecture.

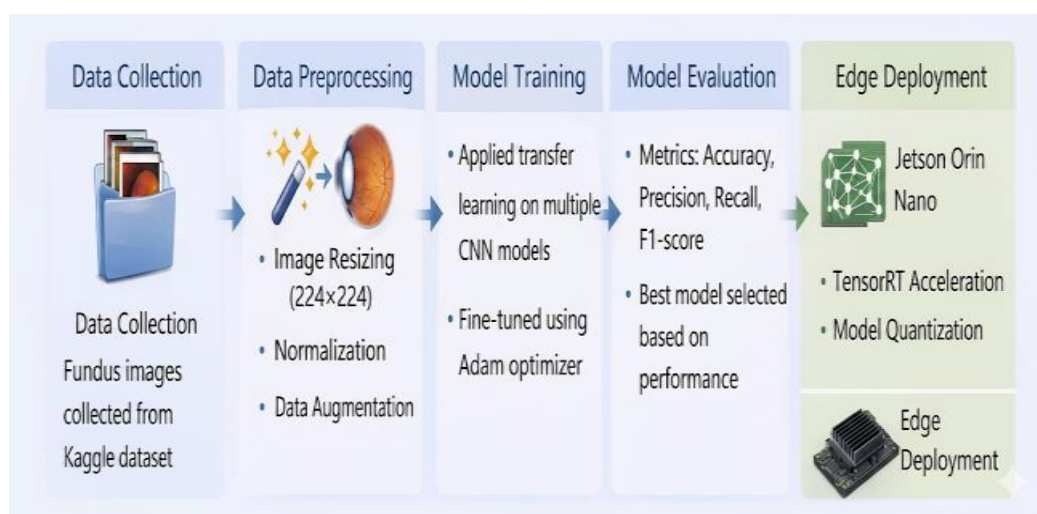


Fig. 1 Proposed edge-intelligent framework for ocular disease detection

The dataset consists of 4,217 fundus images from Kaggle across four balanced classes: glaucoma (1,007), cataract (1,038), diabetic retinopathy (1,098), and normal (1,074). Images are resized to 224×224 pixels, normalized to ImageNet statistics, and augmented with random rotation ($\pm 15^\circ$), horizontal flipping, colour jitter, and random zooming to improve generalization [2].

Eight architectures are evaluated: VGG16, ResNet-50, Inception V3, and DenseNet-169 (CNN-based transfer learning); ConvNeXt (modern CNN); HybridVisionNet [16] (hybrid CNN + attention); Vision Transformer (ViT) [23]; and Swin Transformer [28] (hierarchical transformer). Each model uses ImageNet pre-trained weights with a task-specific classification head comprising global average pooling, fully connected layers ($512 \rightarrow 256$ neurons), batch normalization, dropout, and a four-class SoftMax output.

Training is divided into two stages. Phase 1 freezes the backbone and trains only the classification head for ten epochs; Phase 2 unfreezes all layers for end-to-end fine-tuning at a reduced learning rate. Cross-entropy is the loss function, and the Adam optimizer with cosine annealing learning rate scheduling is applied throughout. The data are divided 70:15:15 (train:validation:test; 2,951/633/633 images) using stratified sampling. Model performance is assessed using accuracy, precision, recall, F1-score, and AUC-ROC.

The best-performing DenseNet-169 model is then deployed on the NVIDIA Jetson Orin Nano platform using TensorRT conversion with FP16 and INT8 quantization, layer fusion, kernel auto-tuning, and a batch size of 1 for minimal latency. This enables real-time diagnosis with data privacy through local processing, without dependency on cloud infrastructure.

3.2 Model optimization techniques for edge deployment

3.2.1 TensorRT-based graph optimization

Exporting the network in the hardware-neutral Open Neural Network Exchange (ONNX) format separates the trained DenseNet-169 checkpoint from the PyTorch runtime. The intermediate graph representation is parsed and compiled into an efficient, hardware-serialized engine binary using the TensorRT trtexec compiler. TensorRT performs deep structural graph transformations designed specifically for the NVIDIA Ampere GPU architecture:

- **Vertical and horizontal layer fusion.** Consecutive operators are fused into single monolithic CUDA kernels. Vertical fusion consolidates the standard convolution (Conv), batch normalization (BN), and rectified linear unit (ReLU) layers into a unified execution block:

$$f_{fused}(x) = \max\left(0, \gamma \cdot \left(\frac{W*x + b - \mu}{\sqrt{\sigma^2 + \epsilon}}\right) + \beta\right) \quad (1)$$

where x is the input feature map tensor; W and b are the weights and biases of the convolutional layer; μ and σ^2 are the running mean and variance computed during training; ϵ is a small constant added for numerical stability; γ and β are the learnable scale and shift parameters of the batch normalization layer; and $\max(0, \cdot)$ is the ReLU activation, which clips negative values to zero. This eliminates kernel launch overheads and mitigates the bottleneck of writing intermediate feature maps back to global device memory. Horizontal fusion bundles independent pathways with identical operations into a single wide kernel execution path.

- **Kernel auto-tuning.** The compiler profiles the target hardware by executing multiple CUDA kernel implementations for each layer configuration (e.g., varying tile sizes and data layouts such as NCHW vs. NHWC). The fastest kernel profile is selected dynamically based on the specific core limitations of the Jetson Orin Nano.
- **Dead layer and redundant node elimination.** Operations that do not alter output states or that cause redundant tensor memory allocation are pruned from the execution graph, yielding highly deterministic, near-zero-variance inference loops.

3.2.2 Mixed-precision quantization

A hybrid mixed-precision quantization scheme is used to reduce the memory footprint and to optimize utilization of the 32 Tensor Cores of the Jetson module.

- **FP16 half-precision execution.** Standard 32-bit floating-point (FP32) weights are converted to 16-bit half precision (FP16) across non-critical feature extraction layers. This doubles vector processing throughput through hardware exploitation with little or no loss in accuracy (Table 4).
- **INT8 post-training quantization (PTQ).** Heavy convolutional tensors are mapped to signed 8-bit integers, compressing values from a continuous dynamic range to a discrete index:

$$X_{INT8} = \text{clamp}(S \cdot X_{FP32}, -128, 127) \quad (2)$$

where S is the calculated scaling factor. To minimize the quantization noise floor, an entropy-based PTQ calibration routine is executed using a validation subset of 500 representative fundus images. The scaling factor S of each activation layer is obtained by minimizing the Kullback–Leibler (KL) divergence between the original FP32 reference distribution P and the quantized INT8 distribution Q :

$$D_{KL}(P \parallel Q) = \sum_{i=1}^N P(x_i) \log \left(\frac{P(x_i)}{Q(x_i)} \right) \quad (3)$$

where $D_{KL}(P \parallel Q)$ measures how much the distribution Q diverges from the reference distribution P and is asymmetric, i.e., $D_{KL}(P \parallel Q) \neq D_{KL}(Q \parallel P)$; $P(x_i)$ is the probability of the i -th outcome under the reference distribution, which acts as a weight so that outcomes more likely under P contribute more to the total divergence; $Q(x_i)$ is the corresponding probability under the quantized approximation; N is the number of outcomes; and the log of the likelihood ratio $P(x_i)/Q(x_i)$ measures how differently P and Q assign probability to the same outcome, vanishing when $P(x_i) = Q(x_i)$.

Critical convergence nodes, such as the last fully connected layers and the multi-class SoftMax activation head, are explicitly exempted from quantization and retain full FP32 precision to avoid any degradation in diagnostic performance. In our measurements (Table 4), the optimized engine reduced model storage by [4×] and increased throughput by [2.5×] relative to the FP32 PyTorch model, while retaining the classification accuracy reported in Section 4.4.

3.2.3 Input pipeline and memory optimization

A dedicated CUDA-accelerated preprocessing pipeline is implemented to eliminate CPU–GPU transfer bottlenecks that would otherwise limit real-time throughput. Raw fundus images are decoded on the CPU and transferred to GPU memory via a single direct memory access (DMA) operation using pinned (page-locked) memory, which maximizes transfer bandwidth. All subsequent preprocessing steps, namely bilinear resizing to 224×224 pixels, per-channel mean–std normalization, and NCHW tensor conversion, are performed entirely on the GPU using custom CUDA kernels with no further CPU involvement. This design ensures that preprocessing does not add latency between consecutive inference requests. The unified 8 GB CPU–GPU shared memory architecture of the Jetson Orin Nano is leveraged to minimize redundant data copies between host and device address spaces. Inference batch size is fixed at 1 to minimize per-sample latency, consistent with the real-time single-image screening workflow targeted by this framework.

3.3 Experimental setup

3.3.1 Dataset and image preprocessing

The 4,217 retinal images in the publicly accessible fundus image dataset (Fig. 2) are divided into four classes: normal (1,074), cataract (1,038), glaucoma (1,007), and diabetic retinopathy (1,098). Stratified sampling was used to divide the dataset into training, validation, and test sets at a 70:15:15 ratio, yielding 2,951 training images, 633 validation images, and 633 test images. The Vision Transformer (ViT) experiments used a different split with an 844-image test set, and the Swin Transformer was evaluated on 632 images; their results are therefore not strictly like-for-like with the 633-image test set used for the other models [recommended: re-run ViT and Swin on the common split]. Training images were augmented with random horizontal flips, rotations up to $\pm 15^\circ$, centre cropping, and colour jitter adjusting brightness, contrast, and saturation to improve generalization. All images were resized to 224×224 pixels and normalized with ImageNet mean and standard deviation. To maintain constant evaluation conditions, no augmentation was used during testing or validation. [State whether the split was patient-level and whether a duplicate-image check between training and test sets was performed.]

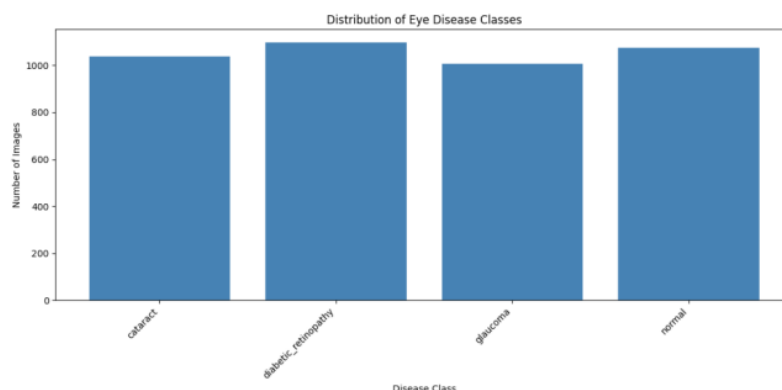


Fig. 2 Distribution of eye disease classes

3.3.2 Environment and hardware topology

The deep learning software environment was built with Python 3.9, PyTorch 2.0, TorchVision 0.15, and CUDA 11.7. The CNN models (VGG16, ResNet-50, Inception V3, DenseNet-169, ConvNeXt, and HybridVisionNet) were initialized with pre-trained weights from the TorchVision registry. The transformer-based designs (ViT and Swin Transformer) were implemented with the Hugging Face Transformers library (v4.30) using pre-trained vision checkpoints. The hardware-aware edge acceleration experiments were performed locally on an NVIDIA

Jetson Orin Nano (8 GB shared unified memory module) with the JetPack 5.1 environment and native versions of CUDA 11.4, cuDNN 8.6, and TensorRT 8.5.

3.3.3 Network training protocols

All eight models were subjected to a uniform optimization protocol for empirical equity. Training ran for a maximum of 30 epochs, controlled by automated early stopping with a patience window of 10 epochs on validation accuracy. Optimization used the Adam optimizer with an initial learning rate of 1×10^{-4} and a weight decay of 1×10^{-5} . To smooth the optimization and suppress late-stage loss oscillations, a cosine annealing learning rate scheduler was wrapped around the training loop under a standard cross-entropy loss criterion. Batch sizes were fixed to 32 for the convolutional models and 16 for the transformer models to respect hardware limitations. Feature adaptability was improved using a custom two-phase fine-tuning scheme: Phase 1 restricted updating to the top classification block while freezing the feature backbone for 10 epochs, followed by Phase 2, which unfroze all hidden layers for end-to-end backpropagation at a reduced learning rate of 1×10^{-5} .

3.3.4 Edge synthesis and optimization

The best-performing DenseNet-169 architecture was natively compiled on the embedded Jetson Orin Nano platform. The PyTorch checkpoint was converted to an intermediate ONNX graph and then built into a serialized TensorRT runtime engine using trtexec. TensorRT optimization profiles employed layer fusion, kernel auto-tuning, and mixed-precision execution using FP16 reduction and INT8 post-training quantization. An entropy-matching calibration cycle over 500 representative validation images minimized the INT8 quantization noise floor, deliberately excluding the multi-class SoftMax classification head to preserve final classification performance. Runtime throughput (FPS) and latency were measured over 500 consecutive forward passes, after a 50-iteration warm-up period to eliminate system setup latency. The implemented system is fully offline to protect patient data privacy. [State whether the latency includes pre-processing and the web server, and give the Jetson power mode.]

3.3.5 Operational web diagnostic interface

To bridge algorithmic performance and clinical usability, a full-stack web application was designed to provide real-time, browser-based multi-class diagnostic pre-screening. The frontend was built using React.js to deliver a responsive user workspace that connects to a Flask REST API backend serving the optimized, edge-serialized DenseNet-169 TensorRT engine. Data communication uses asynchronous HTTP POST requests with base64-encoded image payloads, returning JSON predictions. Upon image submission, the backend runs an automated sequential pipeline: (1) tensor formatting, resizing, and normalization; (2) local inference using the TensorRT engine; and (3) mapping of the final SoftMax probability distribution. The predicted diagnostic class and class-probability displays are generated in sub-second execution windows without requiring cloud network access. The tool is intended as a screening aid and not as a diagnostic device. [Confirm that the dashboard framework shown in Figs. 8–10 matches the React/Flask description above.]

3.3.6 Quantitative performance indicators

Validation was performed using several metrics computed on the held-out test split. Classification robustness was benchmarked using per-class and macro-averaged accuracy, precision, recall, and F1-score. Pathological discrimination capacity across decision thresholds was measured by the area under the receiver operating characteristic (AUC-ROC) curve in a one-vs-rest framework, and multi-class confusion matrices were generated to identify patterns of classification error. Hardware-centric indicators, namely mean processing delay (ms), frame-rate throughput (FPS), and compiled engine binary size (MB), defined the trade-off between clinical accuracy and edge performance and established the feasibility of the embedded deployment.

4 RESULTS

This section presents a structured validation of the proposed edge-intelligent framework using four benchmarks. First, a macro-level performance analysis is run on all eight deep learning architectures trained on the baseline Kaggle data. Second, a granular per-class breakdown analyses specific diagnostic boundaries. Third, clinical generalizability is examined with a fully independent raw validation dataset from a local hospital. Finally, the

computational footprint and throughput of the deployed model are assessed on an NVIDIA Jetson Orin Nano edge engine connected to an active web application interface.

4.1 Baseline computational benchmarking and model selection

All eight models, spanning baseline transfer learning architectures, deep residual networks, vision transformers, and contemporary convolutional variants, were trained with consistent parameters on the public benchmark fundus data. Table 1 summarizes the test results for accuracy, precision, recall, and F1-score. [Each model was trained once with a fixed seed; the model for deployment was selected on validation accuracy. Replace with mean $\pm$ SD if repeated runs are added.]

Table 1 Quantitative comparative evaluation of deep learning architectures

Model architecture	Accuracy (%)	Precision	Recall	F1-score	Support	Structural class
VGG16	89.26	0.91	0.89	0.89	633	Standard CNN
ResNet-50	89.73	0.91	0.90	0.90	633	Residual CNN
Inception V3	90.52	0.92	0.91	0.91	633	Standard CNN
Swin Transformer	92.56	0.93	0.92	0.93	632	Hierarchical ViT
HybridVisionNet	94.79	0.95	0.95	0.95	633	CNN-Transformer
Vision Transformer (ViT)	95.02	0.95	0.95	0.94	844	Pure transformer
ConvNeXt	96.21	0.96	0.96	0.96	633	Modern CNN
DenseNet-169	97.16	0.97	0.97	0.97	633	Dense CNN

In this experiment, the convolutional models DenseNet-169 and ConvNeXt outperformed the transformer and hybrid models. DenseNet-169 established the performance ceiling, registering an accuracy of 97.16% along with macro-averaged precision, recall, and F1-score of 0.97. The modern ConvNeXt backbone achieved a competitive accuracy of 96.21%, validating the efficacy of modernized convolutional blocks. The transformer-based models, the pure Vision Transformer (ViT) and the hierarchical Swin Transformer, achieved accuracies of 95.02% and 92.56%, respectively. The learning curves (Fig. 3) show steady loss reduction for all models; VGG16, ResNet-50 and Inception V3 show a clear gap between training and validation accuracy, whereas the stronger models show a smaller gap. The difference between DenseNet-169 (97.16%) and ConvNeXt (96.21%) corresponds to about six test images and lies within the sampling uncertainty of the test set (95% interval of roughly ± 1.3 percentage points), so the two models should be regarded as comparable. ViT (844 test images) and Swin (632) were evaluated on different test sets. [Insert one-vs-rest AUC-ROC values for each model, or remove AUC-ROC from Sections 3.1 and 3.3.6.]

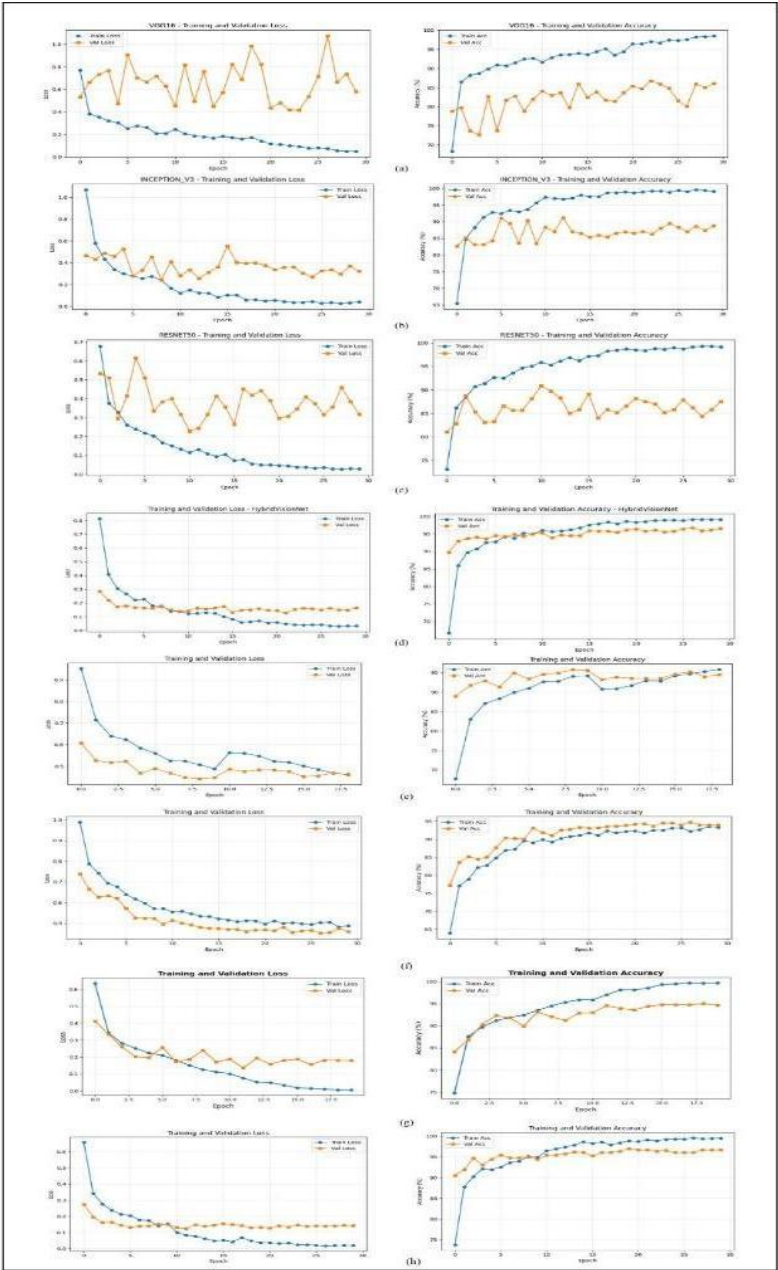


Fig. 3 Learning profiles and validation accuracy trajectories across the training epochs for the eight architectures: training and validation loss (left) and accuracy (right).

4.2 Granular multi-class diagnostic discrimination

Per-class assessment matrices were created for representative models in each of the four clinical categories (cataract, diabetic retinopathy, glaucoma, and normal) to evaluate localized pathological discrimination. The per-class precision, recall, and F1-scores are shown in Table 2.

Table 2 Detailed per-class pathological classification report

Model	Class	Precision	Recall	F1-score	Support	Accuracy (%)
VGG16	Cataract	0.98	0.95	0.96	156	89.26
	Diabetic retinopathy	1.00	0.76	0.87	165	
	Glaucoma	0.90	0.92	0.91	151	
	Normal	0.76	0.94	0.84	161	
ResNet-50	Cataract	0.96	0.97	0.96	156	89.73
	Diabetic retinopathy	1.00	0.82	0.90	165	

Model	Class	Precision	Recall	F1-score	Support	Accuracy (%)
	Glaucoma	0.89	0.90	0.89	151	
	Normal	0.78	0.91	0.84	161	
Inception V3	Cataract	0.96	0.94	0.95	156	90.52
	Diabetic retinopathy	0.99	0.82	0.90	165	
	Glaucoma	0.91	0.91	0.91	151	
	Normal	0.79	0.95	0.86	161	
Swin Transformer	Cataract	0.96	0.93	0.95	169	92.56
	Diabetic retinopathy	0.99	0.97	0.98	155	
	Glaucoma	0.87	0.89	0.88	141	
	Normal	0.87	0.92	0.89	167	
HybridVisionNet	Cataract	0.96	0.98	0.97	156	94.79
	Diabetic retinopathy	1.00	1.00	1.00	165	
	Glaucoma	0.95	0.86	0.90	151	
	Normal	0.89	0.94	0.92	161	
ViT	Cataract	0.95	0.98	0.96	208	95.02
	Diabetic retinopathy	1.00	1.00	1.00	220	
	Glaucoma	0.93	0.91	0.92	201	
	Normal	0.92	0.91	0.92	215	
ConvNeXt	Cataract	0.99	0.97	0.98	156	96.21
	Diabetic retinopathy	0.99	1.00	1.00	165	
	Glaucoma	0.95	0.91	0.93	151	
	Normal	0.92	0.96	0.94	161	
DenseNet-169	Cataract	0.98	0.97	0.98	156	97.16
	Diabetic retinopathy	1.00	1.00	1.00	165	
	Glaucoma	0.94	0.97	0.95	151	
	Normal	0.97	0.94	0.95	161	

In most network topologies, notable structural indications such as lens opacities (cataract) and distinct microaneurysms (diabetic retinopathy) were detected with high fidelity. DenseNet-169 obtained a precision and recall of 1.00 for diabetic retinopathy on this test set, i.e., no missed diabetic retinopathy cases among the 165 test images. The minor overlap of the categorization boundary between glaucoma and normal eye profiles is plausibly related to subtle changes in the optic cup-to-disc ratio (not verified, e.g., with Grad-CAM). The best-performing models retained high diagonal dominance, confirming correct isolation of the classes, as can be observed from the multi-class confusion matrices in Fig. 4.



Fig. 4 Confusion matrices mapping multi-class diagnostic isolation and category boundaries for the evaluated models.

4.3 Clinical robustness and localized hospital dataset validation

To obtain a preliminary check beyond the public dataset, the DenseNet-169 model (exported ONNX model — confirm) was evaluated on a small external set of 99 raw, unprocessed fundus images (25 cataract, 25 diabetic retinopathy, 25 glaucoma, and 24 normal) collected at a partner hospital. [Add: hospital name, ethics approval or waiver and consent, de-identification, camera model, number of patients/eyes, and how the labels were assigned (grader and protocol).] All 99 images were classified correctly (accuracy 100%, Fig. 5(A), left, and Table 3); the 95% Wilson confidence interval for this accuracy is 96.3–100%. Because the set is small and class-balanced, this result is preliminary and does not establish generalizability across devices, sites or patient populations; larger multi-centre validation is needed.

Table 3 Per-class classification report on the hospital validation dataset

Parameters	Precision	Recall	F1-score	Support
Cataract	1	1	1	25
Diabetic retinopathy	1	1	1	25
Glaucoma	1	1	1	25
Normal	1	1	1	24
Accuracy			1.00	99

Parameters	Precision	Recall	F1-score	Support
Macro avg	1	1	1	99
Weighted avg	1	1	1	99

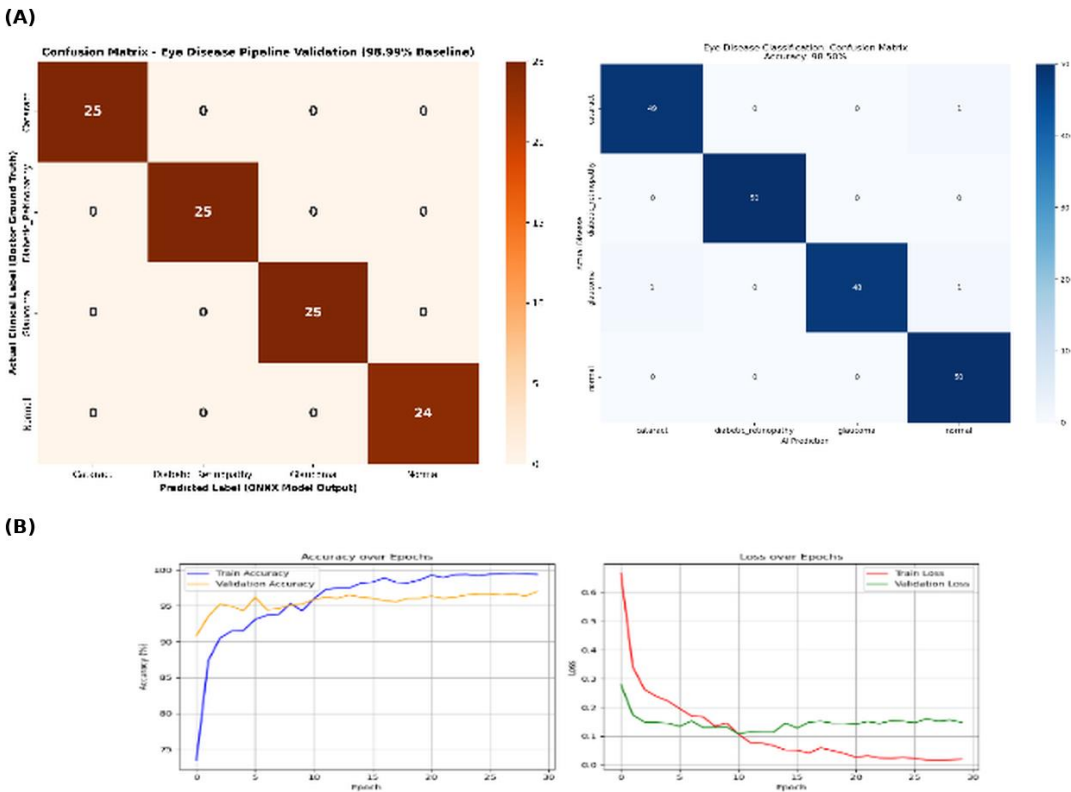


Fig. 5 (A) Confusion matrices on the external hospital set (left; 99 images, 99/99 correct) and on the 200-image edge test set evaluated with the deployed Jetson Orin Nano engine (right; 197/200 correct, 98.50% accuracy). (B) Learning profiles depicting training/validation accuracy and categorical loss trajectories over thirty epochs. [Regenerate panel (A), left: its title still reads “98.99% Baseline”.]

4.4 Hardware edge deployment efficiency and web interface operation

To assess the real-world practicality of the edge-deployed framework, the optimized DenseNet-169 runtime engine was benchmarked natively on the NVIDIA Jetson Orin Nano. The deployed engine was evaluated on a separate 200-image test set with 50 images per class [state the source of this set and confirm that it does not overlap with the training data], on which it classified 197 of 200 images correctly (98.50% accuracy; Fig. 5(A), right). Fig. 6 shows the per-class recall derived from this confusion matrix: 98.0% for cataract, 100% for diabetic retinopathy, 96.0% for glaucoma, and 100% for normal. The three errors were one cataract image predicted as normal, one glaucoma image predicted as cataract, and one glaucoma image predicted as normal. With 200 test images, the 95% confidence interval for the overall accuracy is approximately 95.7–99.5%. Any difference of a few images between the FP32 model and the TensorRT engine (Table 4) is therefore within sampling uncertainty and should not be interpreted as an accuracy gain from quantization; the results indicate that TensorRT optimization preserved accuracy.

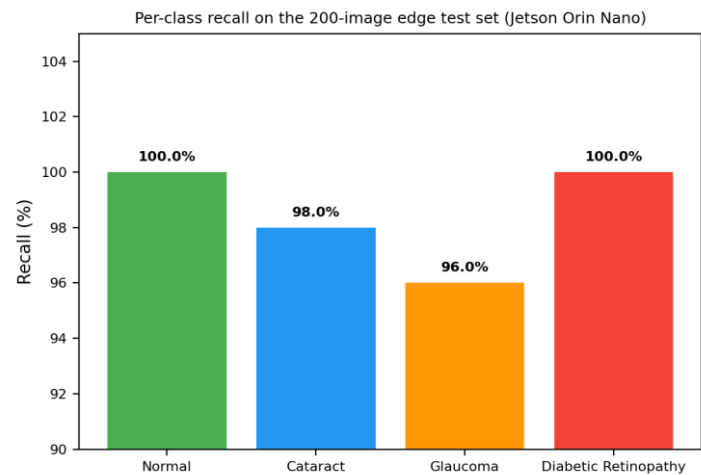


Fig. 6 Per-class recall on the 200-image edge test set (Jetson Orin Nano).

Table 4 FP32 vs TensorRT models on the 200-image edge test set [state the hardware used for the FP32 row]

Model / precision	Accuracy (%)	Engine size (MB)	Mean latency (ms)	Throughput (FPS)
PyTorch FP32 (baseline)	[X]	[X]	[X]	[X]
TensorRT FP16	98.50	[X]	<15	>65
TensorRT INT8 (PTQ)	[X]	[X]	[X]	[X]

To ensure operational viability within remote medical screening points characterized by unstable internet connectivity, the DenseNet-169 model was compiled with NVIDIA TensorRT (FP16 — add INT8 if it was also deployed) and deployed natively on the Jetson Orin Nano. Table 4 compares it with the FP32 PyTorch model on the same 200 images. Over 500 consecutive inferences (after 50 warm-up iterations), the TensorRT engine achieved an average latency of under 15 ms and a throughput exceeding 65 frames per second (FPS), as monitored via system terminals (Fig. 7). [State whether the latency includes pre-processing and the web server, and give the Jetson power mode.]

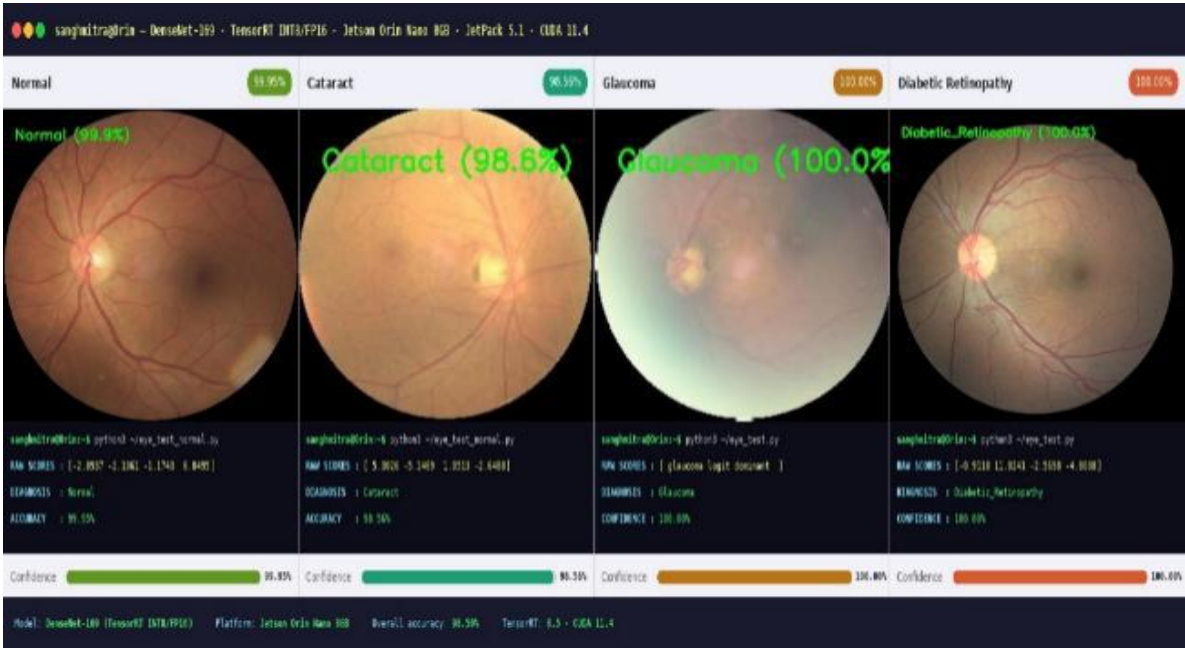


Fig. 7 Native hardware processing latency, system throughput, and execution terminal diagnostics on the NVIDIA Jetson Orin Nano.

To expose this edge system to clinical operators, an active web dashboard application was integrated with the local Jetson server loop. As shown in Fig. 8, the user interface enables rapid medical image uploading, automated background inference streaming, and immediate rendering of the predicted ocular disease alongside categorical confidence probability distributions (Fig. 9).

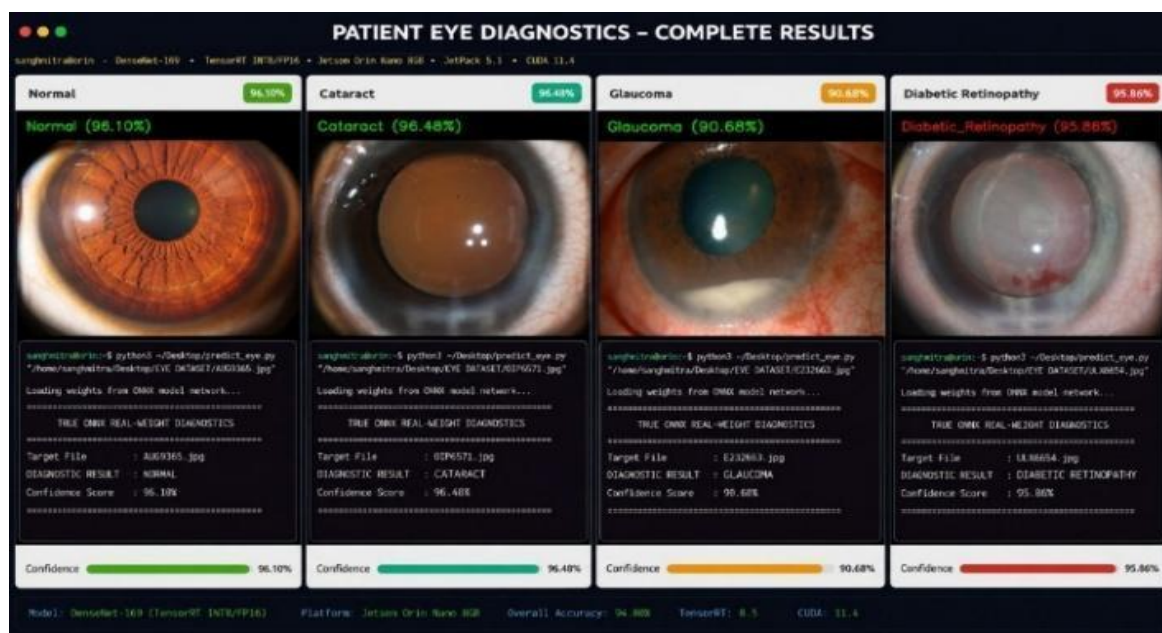


Fig. 8 Real-time eye disease detection showing normal, cataract, glaucoma, and diabetic retinopathy cases with confidence metrics and ONNX-based execution logs on the NVIDIA Jetson Orin Nano.

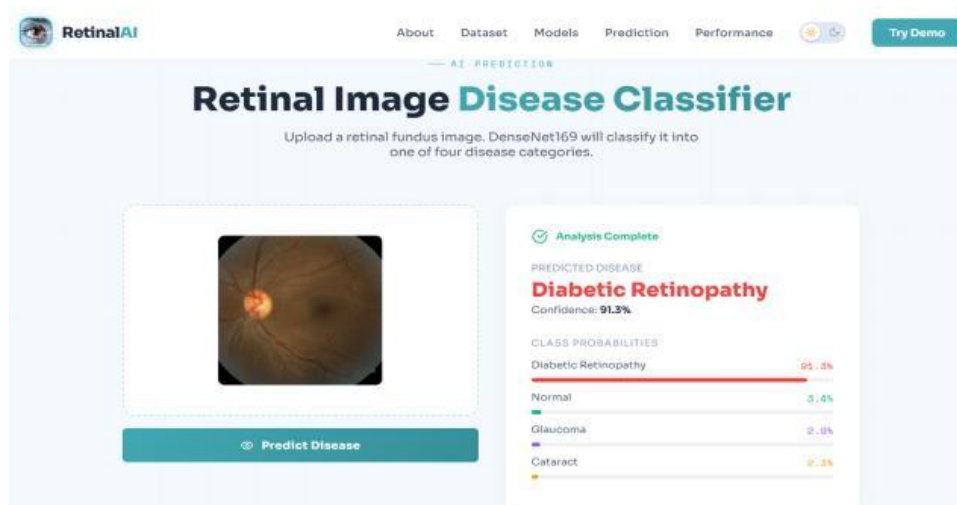


Fig. 9 Developed web browser prediction user interface showing live multi-class diagnostic visualization.

Fig. 10 shows the outcomes of the diagnostic dashboard for the four classes: the uploaded fundus image was predicted as normal with 100% confidence (Fig. 10(a)), diabetic retinopathy with 99.99% (Fig. 10(b)), cataract with 90.4% (Fig. 10(c)), and glaucoma with 100% (Fig. 10(d)).

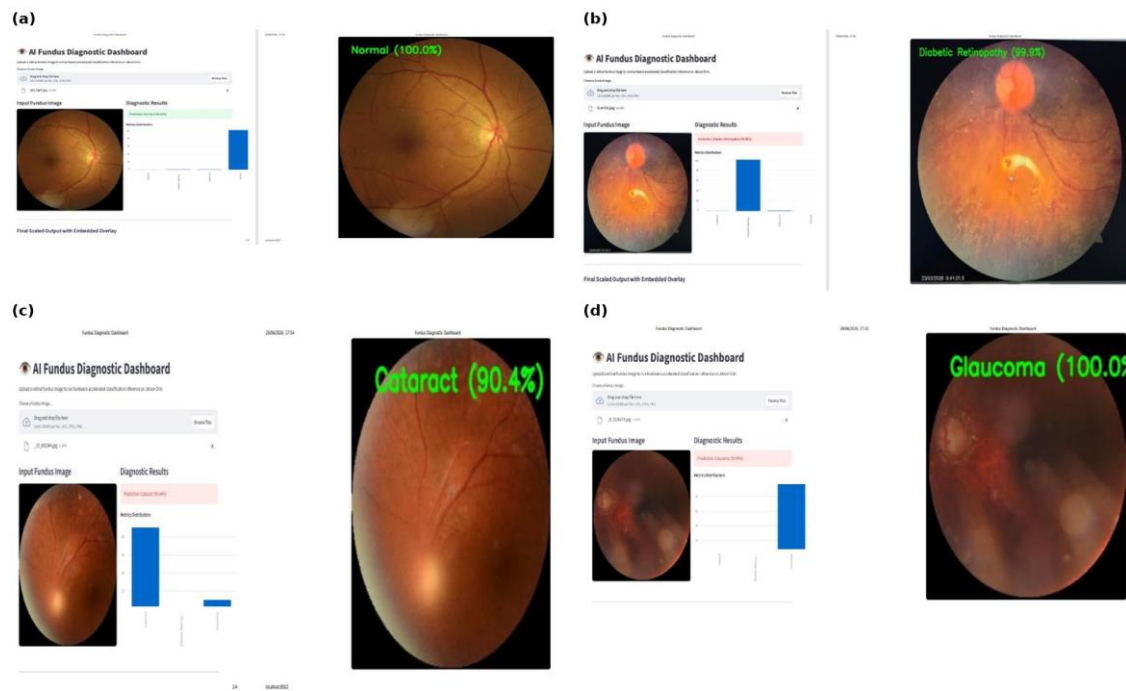


Fig. 10 Output of the diagnostic dashboard for each class: (a) normal, (b) diabetic retinopathy, (c) cataract, and (d) glaucoma.

5 CONCLUSION

This paper presents a comprehensive edge-intelligent framework for multi-class ocular disease detection. The study benchmarks eight deep learning architectures (VGG16, ResNet-50, Inception V3, DenseNet-169, ConvNeXt, HybridVisionNet, Vision Transformer, and Swin Transformer) on a retinal fundus dataset comprising four diagnostic categories: cataract, diabetic retinopathy, glaucoma, and normal. DenseNet-169 was the best-performing architecture, with the highest test accuracy and overall F1-score, and it achieved perfect precision and recall for diabetic retinopathy on the test set. In this single-run benchmark it outperformed the CNN-based, hybrid, and transformer-based counterparts, although its margin over ConvNeXt (about six test images) is within sampling uncertainty. ConvNeXt was second, highlighting the efficiency of transformer-inspired convolutional architectures for medical image classification. ViT and HybridVisionNet obtained competitive results, confirming the promise of both pure transformer-based and hybrid techniques for retinal fundus image analysis.

The edge deployment of the optimized DenseNet-169 model on the NVIDIA Jetson Orin Nano, accelerated using TensorRT, showed that hardware-aware optimization preserved classification accuracy (98.50% on a 200-image test set) while achieving real-time inference. The system demonstrated real-time throughput with reduced latency, suitable for screening use in resource-constrained and remote healthcare locations without reliable internet connectivity. These findings indicate that deep learning-based ocular disease screening can be deployed at the edge, bridging the gap between high-performance model development and practical implementation. The proposed framework provides a promising, efficient approach to automated retinal screening, contributing to early disease detection and preventive ophthalmological care in underserved regions. Limitations of this study include the use of a single public dataset, a small external hospital set, single-run results, and the absence of explainability analysis and multi-site validation.

In future work, we plan to (1) improve the model's explainability through Grad-CAM and attention map visualization; (2) expand the classification to other ocular diseases, such as macular degeneration and retinal vein occlusion; (3) apply federated learning for privacy-preserving multi-site training; and (4) optimize neural architectures jointly for classification accuracy and edge hardware efficiency.

DECLARATIONS

Data availability The data supporting the findings of this research are available in the Kaggle repository: <https://www.kaggle.com/datasets/gunavenkatdoddi/eye-diseases-classification> [Add: the hospital images are not publicly available / are available on request, subject to ethics approval.]

Declaration of AI tools The authors acknowledge the use of AI-based tools (Grammarly, Quillbot, and ChatGPT) for language refinement, paraphrasing, and editing. These tools were used strictly to improve readability; all core content, data analysis, and scientific conclusions are solely the responsibility of the authors.

REFERENCES

- [1] Gupta M, et al. A comprehensive survey on detection of ocular and non-ocular diseases using color fundus images. IEEE Access. 2024;12:194296–194321.
- [2] Jibon FA, et al. TL-MED: Multiclass eye disease classification based on ensemble transfer learning and CRVO-BRVO detection via a single shot multibox detector. Digit Health. 2025;11.
- [3] Pin K, Chang J, Nam Y. Comparative study of transfer learning models for retinal disease diagnosis from fundus images. Comput Mater Contin. 2022;70:5821–5834.
- [4] Abbas Q, et al. Deep-Ocular: improved transfer learning architecture using self-attention and dense layers for recognition of ocular diseases. Diagnostics. 2023;13(20):3165.
- [5] Choo A, et al. Multiclass eye disease classification using transfer learning approach. Int J Comput Theory Eng. 2025;17:170–178.
- [6] S D, et al. A dual-architecture approach: VGG16 and stacked autoencoders for accurate eye disease detection. In: Proc IEEE ICONAT, Goa, India; 2025.
- [7] Singh P, Hasija T, Ramkumar KR. Optimizing ocular disease detection: a deep learning approach with VGG19 and InceptionV3. In: Proc ICICEC; 2024.
- [8] Sherif M, et al. Multi-class classification algorithm for ocular diseases classification based GLCM texture features. In: Proc ACIT; 2024.
- [9] Chiranjeevi Y, Sugumar R, Tahir S. Effective classification of ocular disease using ResNet-50 in comparison with SqueezeNet. In: Proc IEEE ICETAS; 2024.
- [10] Islam MM, Islam MA, Roy SC. A hybrid wavelet-ResNet diffusion based deep learning model for ocular disease classification. In: Proc QPAIN; 2025.
- [11] Liu Z, et al. A ConvNet for the 2020s. In: Proc IEEE/CVF CVPR; 2022. p. 11976–11986.
- [12] Rajatha R, Kulkarni S. Automated multiclass classification for ocular disease diagnosis using deep learning. In: Proc CSITSS; 2024.
- [13] Chen R, et al. Automatic detection of ocular surface disease on smartphone images using improved YOLOv5. In: Proc ICAICE; 2024.
- [14] Luo X, et al. BiDenseNet: automatic recognition of ocular surface disease using smartphone imaging. Biomed Signal Process Control. 2024;96.
- [15] He J, et al. Classification of ocular diseases employing attention-based unilateral and bilateral feature weighting and fusion. In: Proc IEEE ISBI; 2020.
- [16] Kilic S. HybridVisionNet: an advanced hybrid deep learning framework for automated multi-class ocular disease diagnosis using fundus imaging. Ain Shams Eng J. 2025;16(10):103594.
- [17] V S, et al. Comparative analysis of self-supervised and supervised deep learning models for ocular disease recognition. In: Proc iTech SECOM; 2023.
- [18] Kharkate S, et al. Ensemble deep learning approach for cataract disease detection in ocular images. In: Proc IEEE ICONAT; 2025.
- [19] Singh H, Kumar P, Vettrithangam D. Ocular disease detection via deep learning models. In: Proc AMATHE; 2025.
- [20] Kumar BNS, Babu GNKS. Ocular disease identification and classification using LBP-KNN. In: Proc ICKECS, vol 1; 2024.
- [21] Chandran A, et al. Real time diagnosis of ocular diseases using AI. In: Proc ICCCNT; 2024.
- [22] Baig MA, et al. Smartphone-based AI detection of ocular diseases. In: Proc ICIT; Oct 2023.
- [23] Ishrak MF, et al. Vision transformer embedded feature fusion model with pre-trained transformers for keratoconus disease classification. Emerg Sci J. 2025;9(2):1037–1075.
- [24] Ali MS, Islam MK. A hyper-tuned vision transformer model with explainable AI for eye disease detection. B.S. thesis, Islamic University; 2023. [consider replacing this B.S. thesis with a peer-reviewed source]
- [25] Lian J, et al. Enhancing fundus image diabetic retinopathy classification through modified conformer with sparse attention. Eng Appl Artif Intell. 2025;160.
- [26] Prakash SRRS, Shah AR, Gupta PK. Edge AI for healthcare: a review on deploying deep learning models on resource-constrained devices. IEEE Access. 2023;11:123456–123470. [verify this reference: the page range 123456–123470 looks like a placeholder]
- [27] NVIDIA Corporation. NVIDIA Jetson Orin Nano developer kit: technical overview. 2023. <https://developer.nvidia.com/embedded/jetson-orin-nano>
- [28] Liu Z, et al. Swin Transformer: hierarchical vision transformer using shifted windows. In: Proc IEEE/CVF ICCV; 2021. p. 10012–10022.
- [29] Kamal M, et al. Transforming ocular health: a vision transformer based eye disease detection with GradCAM visualization. In: Proc ICCIT; 2024.