Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Smart Wearable Glasses for Health Diagnosis Using AI-ML

Aditya Prashant Patil, Vedang Sanjay Doley, Snehal Somnath Ambre
Department of Electronics & Telecommunication Engineering, Jayawantrao Sawant College of Engineering, Hadapsar, Pune 28

Guided By:
Ms. Trusha Wagh
Department of Electronics & Telecommunication Engineering, Jayawantrao Sawant College of Engineering, Hadapsar, Pune 28

LIST OF ABBREVIATIONS

Abbreviation	Full Form
AI	Artificial Intelligence
ML	Machine Learning
CNN	Convolutional Neural Network
DL	Deep Learning
TL	Transfer Learning
RPi	Raspberry Pi
GAP	Global Average Pooling
BN	Batch Normalization
ReLU	Rectified Linear Unit
ImageNet	Large Scale Visual Recognition Dataset
ICML	International Conference on Machine Learning
NeurIPS	Neural Information Processing Systems
ICLR	International Conference on Learning Representations
IoT	Internet of Things
CSI	Camera Serial Interface
RAM	Random Access Memory
OS	Operating System
API	Application Programming Interface
SDK	Software Development Kit
E&TC	Electronics and Telecommunication Engineering
JSCOE	Jayawantrao Sawant College of Engineering
SPPU	Savitribai Phule Pune University
BOM	Bill of Materials
H5	Hierarchical Data Format version 5 (Keras model format)
INT8	8-bit Integer (quantization format)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

CHAPTER 1: INTRODUCTION

1.1 Scope of Project

The Smart Wearable Glasses for Health Diagnosis project aims to design and develop an integrated wearable system capable of detecting common eye diseases in real time using Artificial Intelligence and Machine Learning. The system combines a Raspberry Pi Zero 2W single-board computer, a Raspberry Pi Camera Module, and a deep learning classification model built upon the EfficientNetB0 architecture. The diagnostic system captures ocular images through the wearable camera and classifies them into four clinically relevant categories: Immature Cataract, Mature Cataract, Normal, and Pterygium. The diagnostic output, along with the confidence score and personalised health recommendations, is presented through a Streamlit web application that can be accessed from any device connected to the same local network.

The primary motivation for this project is the critical gap in accessible and affordable ophthalmic screening tools, particularly in rural, semi-urban, and resource-constrained regions of India. Cataracts alone account for approximately 51% of all cases of blindness globally, and the majority of these cases occur in regions where trained ophthalmologists and clinical diagnostic equipment are scarce. By embedding the complete AI-based diagnostic pipeline within a wearable glasses form factor, this project aims to eliminate the dependency on expensive slit-lamp equipment and specialist clinical visits for preliminary eye disease screening. The system is designed as a first-level screening tool intended to facilitate early detection and timely referral to appropriate medical care.

The scope of the project encompasses: (a) collection and preprocessing of a 3,548-image ophthalmic dataset from Kaggle; (b) training and fine-tuning of the EfficientNetB0 transfer learning model; (c) deployment of the trained model on Raspberry Pi Zero 2W for on-device inference; (d) development of a Streamlit web interface for user-friendly diagnosis display; and (e) integration of all components within a wearable glasses hardware prototype.

1.2 Operating Environment — Hardware & Software

1.2.1 Hardware Environment

Table 1.1: Hardware Components Summary

Component	Specification	Purpose
Raspberry Pi Zero 2W	Quad-core ARM Cortex-A53 @ 1GHz, 512 MB RAM	Edge AI computing unit
Raspberry Pi Camera Module v2	8 MP, Sony IMX219 sensor, 1080p @ 30fps	Ocular image capture
Smart Glasses Frame	Custom 3D-printed / commercial frame	Wearable hardware housing
MicroSD Card	32 GB, Class 10 / UHS-I	OS and model storage
Li-Po Battery / Power Bank	5V, 2A output, min. 3000 mAh	Portable power supply
CSI Ribbon Cable	15-pin, 150 mm flexible cable	Camera-Pi interface

1.2.2 Software Environment

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Table 1.2: Software Tools Summary

Software / Tool	Version	Role
Raspberry Pi OS	64-bit Lite (Bookworm)	Host operating system
Python	3.10+	Primary programming language
TensorFlow / Keras	2.x	DL framework for training and inference
EfficientNetB0	ImageNet pre-trained	CNN classification backbone
OpenCV (cv2)	4.x	Image acquisition and preprocessing
Streamlit	1.x	Web-based UI for diagnosis display
NumPy	1.x	Numerical computation and array ops
rpicam-still	Latest	Raspberry Pi camera capture utility
Kaggle Dataset	3,548 images	Ophthalmic training data source

1.3 Brief Description of Technology & Tools Used

1.3.1 EfficientNetB0

EfficientNetB0 is a Convolutional Neural Network architecture developed by Tan and Le (Google Brain, 2019) that employs a principled compound scaling strategy to simultaneously optimise network depth, width, and image resolution using a fixed scaling coefficient. Unlike earlier architectures such as VGG16 (138M parameters) or ResNet50 (25.6M parameters), EfficientNetB0 achieves competitive ImageNet top-1 accuracy of 77.1% while utilising only 5.3 million parameters and occupying approximately 21 MB of storage in H5 format. These characteristics make EfficientNetB0 ideally suited for deployment on edge hardware such as the Raspberry Pi Zero 2W, where computational resources and memory are severely constrained.

The architecture consists of a series of Mobile Inverted Bottleneck Convolution (MBConv) blocks with Squeeze-and-Excitation optimisation, allowing the network to learn channel-wise feature dependencies efficiently. The compound scaling uniformly scales all network dimensions (depth, width, resolution) by a fixed ratio, unlike previous ad-hoc scaling approaches that independently scaled one dimension at a time.

1.3.2 Transfer Learning

Transfer learning involves adapting a deep learning model that has been pre-trained on a large-scale dataset — in this case, ImageNet (1.28 million images across 1,000 classes)
— to a new, related task with limited labelled data. This approach is especially valuable in medical imaging applications where annotated datasets are typically small due to the cost and expertise required for labelling.

In this project, the EfficientNetB0 model is loaded with its pre-trained ImageNet weights, and the last 30 of its 237 layers are selectively unfrozen and retrained on the ophthalmic dataset. The remaining 207 layers are frozen, preserving the low-level feature detectors (edge detectors, texture filters) that transfer well across visual domains. Only the unfrozen layers and the custom classification head are updated via gradient descent, limiting the total number of trainable parameters to approximately 1.2 million out of 5.3 million total.

1.3.3 Streamlit Web Interface

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Streamlit is an open-source Python framework developed by Streamlit Inc. (acquired by Snowflake in 2022) that enables rapid creation of interactive web applications for data science and machine learning models without requiring knowledge of HTML, CSS, or JavaScript. The Streamlit application developed for this project — titled 'Smart Eye Diagnosis' — provides three image input modalities: (1) Raspberry Pi Camera capture via rpicam-still command, (2) image upload from local storage, and (3) webcam capture. For each analysed image, the application displays the predicted diagnosis label, the confidence score as a percentage, a visual probability distribution chart across all four classes, and personalised health recommendations tailored to the diagnosed condition.

1.3.4 Raspberry Pi Zero 2W

The Raspberry Pi Zero 2W is a compact, credit-card-sized single-board computer manufactured by the Raspberry Pi Foundation. Its core specifications include a quad-core ARM Cortex-A53 processor clocked at 1 GHz, 512 MB LPDDR2 SDRAM, integrated 802.11 b/g/n Wi-Fi and Bluetooth 4.2, a dedicated 15-pin CSI camera interface, and micro-USB and mini-HDMI ports. The board measures 65 mm × 30 mm × 5 mm and consumes approximately 0.4 W at idle, making it suitable for battery-powered wearable applications. The Zero 2W runs Raspberry Pi OS (64-bit), which supports the full Python scientific computing stack including TensorFlow and Keras, enabling on-device deep learning inference without cloud connectivity.

1.3.5 OpenCV

OpenCV (Open Source Computer Vision Library) is an open-source machine vision software library primarily designed for real-time computer vision applications. In this project, OpenCV is used for reading captured images from disk, performing colour space conversion (BGR to RGB), resizing images to the required 224 × 224 pixel input dimensions, and preparing image tensors for model inference. OpenCV's optimised C++ backend with Python bindings ensures fast preprocessing on the resource-constrained Raspberry Pi.

CHAPTER 2: LITERATURE SURVEY

A thorough review of the existing research literature was conducted to understand the state of the art in deep learning-based eye disease detection, transfer learning methodologies, wearable health monitoring systems, and edge AI deployment. The following section presents a structured survey of 10 seminal and recent papers that directly inform the design decisions of this project.

2.1 Review of Related Work

[1] Gulshan et al. (2016) — JAMA

Gulshan et al. published a landmark study in the Journal of the American Medical Association (JAMA) demonstrating that a deep Convolutional Neural Network, trained on over 128,000 retinal fundus photographs, could detect diabetic retinopathy and diabetic macular oedema with sensitivity and specificity at or exceeding that of certified ophthalmologists and retinal specialists. The model was trained using an ensemble of Inception-v3 networks, achieving an area under the ROC curve (AUC) of 0.991 on one validation set and 0.990 on another. This study was a pivotal demonstration of the viability of AI-assisted ocular diagnosis and established the research precedent for applying deep learning to ophthalmic screening. It directly motivated the use of deep learning for classifying anterior segment conditions such as cataracts and pterygium in our project.

[2] Tan & Le (2019) — EfficientNet, ICML

Tan and Le introduced EfficientNet at the International Conference on Machine Learning (ICML) 2019, presenting a novel compound scaling methodology for CNNs. The paper demonstrated that simultaneously scaling network depth, width, and input resolution using a compound coefficient η results in dramatically improved accuracy per parameter compared to scaling any single dimension in isolation. The EfficientNetB0 baseline model achieves 77.1% ImageNet top-1 accuracy with only 5.3 million parameters — compared to ResNet50 (76%, 25.6M parameters) and VGG16 (72.9%, 138M parameters). The model's compact size (~21 MB) and computational efficiency make it

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

uniquely suitable for edge deployment on the Raspberry Pi Zero 2W, which has only 512 MB RAM. This paper is the direct architectural foundation of our project.

[3] Zhang et al. (2019) — Computer Methods and Programs in Biomedicine

Zhang et al. proposed an automatic cataract grading system based on deep learning, utilising a ResNet-50 backbone with ImageNet pre-training. The system was evaluated on a binary classification task (normal eye vs. cataract) and achieved over 90% accuracy. The study provided critical evidence that transfer learning from ImageNet significantly outperforms training CNN models from random weight initialisation on limited medical image datasets, validating the transfer learning strategy adopted in this project. The authors also demonstrated that data augmentation (rotation, flipping, colour jitter) is essential to prevent overfitting on small medical datasets, a finding directly incorporated into our training pipeline.

[4] Zheng et al. (2021) — Biomedical Signal Processing and Control

Zheng et al. developed an automatic pterygium detection system based on MobileNet, a lightweight CNN architecture designed for mobile and edge computing applications. The system was evaluated on anterior segment photographs and reported an accuracy of 87.4%. Crucially, the study identified class imbalance — the underrepresentation of pterygium cases relative to normal cases — as a key challenge in pterygium detection datasets. The authors proposed the use of weighted loss functions as a mitigation strategy to improve sensitivity on the minority class without requiring oversampling of training data. This finding directly motivated the asymmetric class weighting scheme (pterygium weight = 2.5×) employed in our model training.

[5] Raghu et al. (2019) — NeurIPS (Transfusion Study)

Raghu et al. conducted a comprehensive study titled 'Transfusion: Understanding Transfer Learning for Medical Imaging' at NeurIPS 2019. The study systematically compared ImageNet pre-trained models against models trained from random initialisation across multiple medical imaging tasks (chest X-ray, ophthalmology, pathology). The principal finding was that ImageNet pre-training consistently improves performance on medical imaging tasks, even when the source domain (natural images) and the target

domain (medical images) differ substantially. The study also found that even partial fine-tuning (freezing early layers and training only later layers) achieves most of the benefit of full fine-tuning while reducing computational cost. This finding directly validates the selective fine-tuning strategy (unfreezing only the last 30 of 237 EfficientNetB0 layers) adopted in this project.

[6] Li et al. (2021) — Medical Image Analysis

Li et al. published a comprehensive review of deep learning applications in fundus image analysis, covering diabetic retinopathy grading, glaucoma detection, age-related macular degeneration screening, and retinal vessel segmentation. The review emphasised several key design principles for robust medical image classification systems: (a) data augmentation is essential for improving model generalisation with limited datasets; (b) multi-class classification frameworks that unify multiple conditions within a single model outperform separate binary classifiers in terms of deployment efficiency and inter-class discrimination; and (c) model interpretability through visualisation techniques such as Grad-CAM is critical for clinical adoption. These principles directly inform the multi-class unified classification approach and future Grad-CAM integration plans in our project.

[7] Majumder & Deen (2019) — Sensors

Majumder and Deen published a review of physiological sensors embedded in wearable health monitoring devices in the journal Sensors (MDPI). The paper provided a systematic analysis of sensor modalities, processing architectures, and communication protocols used in wearable health devices, ranging from smartwatches and fitness bands to smart glasses and implantable sensors. The review highlighted the emerging convergence of miniaturised embedded computing platforms (such as Raspberry Pi and NVIDIA Jetson) with on-device machine learning inference as a key enabler for the next generation of wearable health monitoring systems. The paper specifically identified smart glasses as a promising form factor for assistive diagnostics, providing direct technological context and motivation for the wearable glasses hardware design in our project.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

[8] Pan & Yang (2009) — IEEE Transactions on Knowledge and Data Engineering

Pan and Yang published a seminal survey on transfer learning in the IEEE Transactions on Knowledge and Data Engineering (TKDE) in 2009. The paper provided a comprehensive taxonomy of transfer learning approaches, categorising them by the relationship between source and target domains and tasks. The survey established inductive transfer learning — where the source and target tasks differ but the target domain has some labelled data — as the most applicable paradigm for adapting ImageNet-trained models to domain-specific medical imaging tasks. The theoretical framework provided in this survey underpins the transfer learning methodology adopted in our project, where the source task is ImageNet image classification and the target task is ophthalmic disease classification.

[9] LeCun et al. (1998) — Proceedings of the IEEE

LeCun, Bottou, Bengio, and Haffner published the foundational paper on gradient-based learning applied to document recognition in the Proceedings of the IEEE in 1998. This paper introduced the LeNet-5 architecture and formalised the core computational primitives of modern CNNs: convolutional layers, pooling layers, and end-to-end training via backpropagation with gradient descent. The theoretical principles established in this landmark paper — local receptive fields, shared weights, and spatial subsampling — underpin all modern deep learning architectures including EfficientNetB0 and directly inform the CNN-based classification approach used in this project.

[10] Kingma & Ba (2015) — ICLR (Adam Optimiser)

Kingma and Ba introduced the Adam (Adaptive Moment Estimation) optimiser at the International Conference on Learning Representations (ICLR) in 2015. Adam maintains a running average of both the first moment (mean) and the second moment (uncentred variance) of gradients, using these estimates to adapt the learning rate for each model parameter individually. This adaptive behaviour makes Adam particularly effective for fine-tuning scenarios — such as the selective layer fine-tuning employed in our project
— where different layer groups have significantly different gradient magnitudes. Adam with a learning rate of 0.0001 is the optimiser used for training the EfficientNetB0 model in this project.

2.2 Summary of Literature Survey

The reviewed literature collectively establishes the following key findings that directly inform the design of the Smart Wearable Glasses for Health Diagnosis system:

- Transfer learning with ImageNet pre-trained models is highly effective for medical image classification tasks with limited labelled data, consistently outperforming training from scratch (Raghu et al., 2019; Zhang et al., 2019).
- EfficientNetB0 provides the optimal accuracy-to-parameter ratio among standard CNN architectures for edge computing deployment, making it the appropriate choice for the Raspberry Pi Zero 2W platform (Tan & Le, 2019).
- Asymmetric class weighting (weighted loss functions) is an effective strategy for handling class imbalance in medical imaging datasets without requiring synthetic data generation or oversampling (Zheng et al., 2021).
- Selective fine-tuning — freezing early layers and training only the later, task-specific layers — achieves most of the benefit of full fine-tuning at a fraction of the computational cost (Raghu et al., 2019).
- Wearable platforms integrating cameras, edge AI processors, and wireless connectivity are technically feasible platforms for real-time health screening applications (Majumder & Deen, 2019).

Notably, no existing published work integrates an EfficientNetB0-based multi-class eye disease classifier directly into a Raspberry Pi-powered wearable glasses system with a web-based diagnostic interface. This gap in the literature constitutes the primary novelty and motivation for the present project.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

CHAPTER 3: OVERVIEW OF PROJECT

The Smart Wearable Glasses for Health Diagnosis Using AI-ML is an end-to-end wearable health system that integrates three core technological domains: embedded hardware, computer vision, and deep learning. The system is designed around the overarching principle of democratising preliminary ophthalmic screening — making it technologically possible to detect common eye diseases without requiring a clinic visit, specialist equipment, or even electricity grid access.

3.1 Relevance with Recent Technologies

The project aligns with several major trends currently shaping the global technology and healthcare landscape:

3.1.1 Edge Artificial Intelligence

Edge AI refers to the deployment of machine learning inference directly on embedded devices, without reliance on cloud servers or remote computing infrastructure. By running the EfficientNetB0 model directly on the Raspberry Pi Zero 2W, the Smart Wearable Glasses system eliminates network latency, protects patient data privacy by keeping diagnostic images on-device, and enables operation in areas without reliable internet connectivity. This aligns with the global Edge AI market, which is projected to grow from USD 12.1 billion in 2024 to over USD 107 billion by 2030.

3.1.2 Wearable Health Technology

Wearable health devices — from fitness trackers and smartwatches to continuous glucose monitors and ECG patches — represent one of the fastest-growing segments of consumer electronics. Smart glasses are emerging as a particularly promising wearable platform for health monitoring because they provide a hands-free form factor with a natural line-of-sight camera placement, making them ideal for non-invasive ocular imaging. The success of devices such as Google Glass Enterprise Edition and Meta Ray-Ban smart glasses demonstrates the commercial and technological viability of the glasses form factor for data acquisition applications.

3.1.3 Transfer Learning in Healthcare

Transfer learning has become the standard paradigm for deep learning in medical imaging because it allows powerful CNN models trained on large natural image datasets (ImageNet) to be efficiently adapted to medical tasks where labelled data is scarce and expensive to obtain. The ability to achieve clinical-grade performance with datasets of a few thousand images — rather than the hundreds of thousands required to train from scratch — has made deep learning practically deployable across ophthalmology, radiology, pathology, and dermatology.

3.1.4 Telemedicine and Remote Healthcare

The COVID-19 pandemic accelerated the global adoption of telemedicine platforms, creating a lasting structural shift towards remote healthcare delivery. The Streamlit web interface of the Smart Wearable Glasses system supports telemedicine workflows by providing a browser-accessible diagnostic display that can be reviewed remotely by healthcare providers, enabling patients in rural areas to receive AI-assisted preliminary diagnosis and referral guidance without physically visiting a hospital.

3.1.5 Smart Healthcare and Smart City Integration

Integration of AI-powered wearable health devices with broader healthcare infrastructure databases and electronic health record systems is a key component of smart city and smart healthcare initiatives globally. The modular software architecture of our system — with its clearly defined inference API and Streamlit web layer — is designed to be extensible to cloud-based health monitoring platforms and IoT health data aggregators in future iterations.

3.2 Focus on EfficientNetB0 Technology

EfficientNetB0 was selected as the classification backbone for the Smart Wearable Glasses system based on a systematic evaluation against alternative CNN architectures. The selection criteria included: (a) model accuracy on the target ophthalmic classification task; (b) model size in MB for on-device storage; (c) inference latency on Raspberry Pi Zero 2W hardware; and (d) number of

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

trainable parameters for efficient fine-tuning.

The compound scaling strategy of EfficientNetB0 achieves higher accuracy than models three times its size. The model's small parameter footprint (5.3M parameters, ~21 MB in H5 format) fits comfortably within the Raspberry Pi Zero 2W's 512 MB RAM, leaving sufficient memory headroom for the OS, Python runtime, and Streamlit server processes. The selective fine-tuning approach — unfreezing only the last 30 of 237 layers — further reduces the computational cost of adaptation to the ophthalmic domain while preserving the generalizable low-level visual features (edge detectors, texture filters, colour gradient detectors) learned during ImageNet pre-training.

3.3 System Pipeline Overview

The complete end-to-end system pipeline of the Smart Wearable Glasses for Health Diagnosis is as follows:

1. Camera Capture: The Raspberry Pi Camera Module v2 captures a high-resolution (1920×1080) anterior segment image of the patient's eye using the rpicam-still command.
2. Image Preprocessing: OpenCV reads the captured image, converts it from BGR to RGB colour space, and resizes it to 224×224 pixels to match EfficientNetB0's input requirements.
3. EfficientNet Normalisation: The preprocessed image is normalised using EfficientNet's built-in preprocess_input function, which applies per-channel mean subtraction and standard deviation division based on ImageNet statistics.
4. EfficientNetB0 Inference: The normalised 224×224×3 tensor is passed through the EfficientNetB0 backbone and custom classification head. The model computes a 4-dimensional softmax probability vector across the four diagnostic classes.
5. Softmax Classification: The predicted class is determined by the argmax of the probability vector, with the associated probability value representing the diagnostic confidence score.
6. Streamlit Display: The predicted diagnosis label, confidence score, probability distribution chart, personalised health recommendations, and optional downloadable diagnostic report are presented on the Streamlit web interface, accessible from any device on the local network.

CHAPTER 4: SPECIFICATIONS & SYSTEM ANALYSIS

4.1 General Specifications

Table 4.1: General System Specifications

Parameter	Specification
Model Architecture	EfficientNetB0 (Transfer Learning, ImageNet pre-trained)
Input Image Size	224 × 224 × 3 (RGB)
Number of Classes	4 (Immature Cataract, Mature Cataract, Normal, Pterygium)
Training Dataset Size	1,977 images
Validation Dataset Size	808 images
Test Dataset Size	763 images
Total Dataset Size	3,548 images
Optimiser	Adam (learning rate = 0.0001)
Loss Function	Weighted Categorical Cross-Entropy
Batch Size	32

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Maximum Epochs	20 (with Early Stopping, patience = 3)
Dropout Rate	0.3 (applied after Dense-128 layer)
Dense Layer Units	128 (ReLU activation)
Trainable Layers	Last 30 layers of EfficientNetB0 + custom head
Total Parameters	~5.3 Million
Trainable Parameters	~1.2 Million
Pterygium Class Weight	2.5× (asymmetric to handle class imbalance)
Edge Hardware	Raspberry Pi Zero 2W (Quad-core ARM Cortex-A53 @ 1 GHz, 512 MB RAM)
Camera Module	Raspberry Pi Camera Module v2 (8 MP Sony IMX219, 1080p @ 30fps)
Camera Utility	rpicam-still (1920×1080, 500 ms exposure)
Web Interface	Streamlit 1.x (local network accessible)
Model File Format	HDF5 (.h5), ~21 MB

Parameter	Specification
Test Accuracy	~93%
Macro-Averaged F1-Score	0.92

4.2 Block Diagram and Description

The system architecture of the Smart Wearable Glasses for Health Diagnosis is organised into three functional layers, each with distinct responsibilities:

4.2.1 Hardware Layer

The Hardware Layer constitutes the physical wearable component of the system. The Raspberry Pi Zero 2W, mounted inside the smart glasses frame alongside the Camera Module v2, forms the core of this layer. The camera is positioned to capture the anterior segment (front surface) of the wearer's eye or the patient's eye through close-up imaging. A portable Li-Po battery or power bank supplies 5V regulated power to the Raspberry Pi via the micro-USB port. The Wi-Fi capability of the Raspberry Pi Zero 2W enables wireless communication with the user's smartphone or laptop for the Streamlit interface.

4.2.2 AI/ML Processing Layer

The AI/ML Processing Layer handles all image processing and deep learning inference operations. Upon image capture by the camera, the OpenCV library performs colour conversion and spatial resizing. EfficientNet's preprocess_input function normalises the image tensor to match the statistical distribution of ImageNet training data. The normalised tensor is then fed through the pre-loaded EfficientNetB0 model (best_model.h5) to generate the softmax probability prediction. This entire pipeline executes on the Raspberry Pi Zero 2W's CPU in pure Python, without requiring GPU acceleration or cloud connectivity.

4.2.3 User Interface Layer

The User Interface Layer presents the diagnostic output to the end user through the Streamlit web application. The Streamlit server

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

runs on the Raspberry Pi Zero 2W and broadcasts on port 8501. Any device (smartphone, tablet, laptop) connected to the same Wi-Fi network can access the diagnostic interface via the Raspberry Pi's IP address. The interface displays the diagnosis label, confidence percentage, class probability chart, and personalised health recommendations based on the predicted condition.

Table 4.2: Hardware System Requirements

Component	Minimum Requirement	Recommended
Raspberry Pi	Zero 2W (512 MB RAM)	Pi 4 Model B (4 GB RAM) for faster inference
Camera	Raspberry Pi Cam v1 (5 MP)	Raspberry Pi Cam v2 (8 MP, IMX219)
Storage	16 GB microSD, Class 10	32 GB microSD, Class 10 / UHS-I
Power	5V 1.5A (idle)	5V 2A (3000 mAh+ power bank)
Network	802.11 b/g Wi-Fi	802.11 n Wi-Fi for faster UI loading
Frame	Any glasses frame with mounting	Custom 3D-printed enclosure

4.3 System Analysis and Requirements

Table 4.3: Software Requirements

Software	Version	Purpose
Raspberry Pi OS	64-bit Bookworm (latest)	Host operating system
Python	3.10 or higher	Primary development and runtime language
TensorFlow	2.x	Deep learning model inference and training
Keras	Bundled with TF 2.x	High-level model API
OpenCV (cv2)	4.x	Image preprocessing pipeline
Streamlit	1.x	Web application UI framework
NumPy	1.x	Array operations and matrix manipulation
rpicam-still	Latest (Bookworm)	Camera capture utility
Pillow (PIL)	9.x+	Image format handling

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Software	Version	Purpose
Matplotlib	3.x	Training curve plotting

The system is designed to operate entirely offline after initial setup, with no internet connectivity required for inference. The trained model (best_model.h5, ~21 MB) is stored locally on the microSD card. The Streamlit server and TensorFlow runtime together require approximately 280-320 MB of the 512 MB RAM on the Raspberry Pi Zero 2W, leaving approximately 180-200 MB for the operating system and Python runtime.

CHAPTER 5: SYSTEM DESIGN

5.1 Selection of Components

5.1.1 Edge Computing Platform — Raspberry Pi Zero 2W vs. Alternatives

Table 5.1: Comparison of Edge Computing Platforms

Platform	CPU	RAM	Size (mm)	Price (USD)	Tensor Flow	Selected
Arduino Uno	ATmega328P 16 MHz (8-bit)	2 KB	68×53	~25	No	No
ESP32	Xtensa LX6 240 MHz (32-bit)	520 KB	51×28	~10	Limited	No
Raspberry Pi Zero 2W	ARM Cortex-A53 1 GHz (64-bit)	512 MB	65×30	~15	Yes (TF Lite)	YES
Raspberry Pi 4 Model B	ARM Cortex-A72 1.8 GHz	2-8 GB	85×56	~55	Full TF	Upgrade Option
NVIDIA Jetson Nano	ARM Cortex-A57 + GPU	4 GB	80×100	~99	Full TF + GPU	Too large

The Raspberry Pi Zero 2W was selected because it is the only platform that satisfies all four selection criteria simultaneously: it runs a full 64-bit Linux OS supporting the complete Python scientific computing stack (TensorFlow, Keras, OpenCV, Streamlit); it provides a dedicated 15-pin CSI camera interface for the Raspberry Pi Camera Module; its compact size (65 × 30 mm) fits within a glasses frame; and its cost (~USD 15) keeps the overall BOM affordable for deployment at scale in resource-constrained settings.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

5.1.2 Deep Learning Architecture — EfficientNetB0 vs. Alternatives

Table 5.2: Comparison of CNN Architectures

Architecture	Param s (M)	Size (MB)	ImageNet Top-1	Suitable for Pi Zero 2W	Selecte d
VGG16	138	528	72.9%	No (too large)	No

Architecture	Param s (M)	Size (MB)	ImageNet Top-1	Suitable for Pi Zero 2W	Selecte d
ResNet50	25.6	98	76.0%	Marginal	No
MobileNetV2	3.4	14	71.8%	Yes	No
EfficientNetB0	5.3	21	77.1%	Yes	YES
EfficientNetB4	19.3	75	82.9%	Marginal	Upgrad e Option

5.2 Selection of Sensors and Camera

5.2.1 Camera Module Selection

The Raspberry Pi Camera Module v2 was selected as the image acquisition sensor for the wearable glasses system. The module uses a Sony IMX219 8-megapixel CMOS image sensor capable of capturing still images at resolutions up to 3280 × 2464 pixels and video at 1080p (1920 × 1080) at 30 frames per second. The module connects to the Raspberry Pi Zero 2W via the 15-pin CSI (Camera Serial Interface) ribbon cable and is controlled through the rpicam-still command-line utility on Raspberry Pi OS Bookworm.

The camera module is mounted inside the glasses frame at an angle optimised for capturing the anterior segment of the eye at close range (approximately 3-5 cm working distance). The rpicam-still command is configured to capture images at 1920 × 1080 resolution with a 500 millisecond exposure time, providing sufficient image quality for preprocessing and classification while minimising motion blur.

5.2.2 Camera Placement Rationale

For eye disease classification, the camera must capture the anterior segment of the eye (cornea, conjunctiva, and lens) with sufficient resolution to distinguish between the visual characteristics of the four diagnostic classes. Immature cataracts appear as a grey-white opacity in the peripheral lens; mature cataracts present as a dense, opaque white opacity covering the entire lens; pterygium appears as a triangular fibrovascular growth extending from the conjunctiva onto the cornea; and normal eyes show a clear, transparent cornea and lens. The camera placement and working distance were optimised during prototyping to ensure adequate field of view and focus for these distinctions.

5.3 Circuit Diagram of Individual Blocks

5.3.1 Raspberry Pi Zero 2W — Camera Module Connection

The Raspberry Pi Camera Module v2 connects to the Raspberry Pi Zero 2W via the 15-pin CSI ribbon cable. The CSI interface on the Zero 2W is located on the board edge and uses a micro-format connector (compared to the standard-format connector

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

on larger Pi models). The ribbon cable carries differential MIPI CSI-2 data signals (clock lane + 2 data lanes), I2C control signals for camera configuration, and 3.3V power supply to the camera module.

The pin-level connection is as follows:

Camera Module Pin	Function	Raspberry Pi Zero 2W Connection
GND	Ground reference	GND (Pin 6)
VCC (3.3V)	Camera power supply	3.3V (Pin 1)
SCL	I2C clock (camera config)	GPIO 3 / SCL (Pin 5)
SDA	I2C data (camera config)	GPIO 2 / SDA (Pin 3)
CLK+	MIPI CSI-2 clock positive	CSI clock lane (+)
CLK-	MIPI CSI-2 clock negative	CSI clock lane (-)
D0+	MIPI CSI-2 data 0 positive	CSI data lane 0 (+)
D0-	MIPI CSI-2 data 0 negative	CSI data lane 0 (-)
D1+	MIPI CSI-2 data 1 positive	CSI data lane 1 (+)
D1-	MIPI CSI-2 data 1 negative	CSI data lane 1 (-)

5.3.2 Power Circuit

The Raspberry Pi Zero 2W is powered through its micro-USB port (5V, 2A). For wearable operation, a USB power bank with a minimum capacity of 3000 mAh at 5V/2A output is connected to the micro-USB data/power port. At a typical operating power draw of approximately 0.4 W (idle) to 1.5 W (peak inference), a 3000 mAh power bank provides approximately 5-8 hours of continuous operation. The power bank is stored in a glasses frame cavity or attached to the frame via a clip mount.

5.4 Bill of Materials (BOM)

Table 5.5: Bill of Materials

S.No.	Component	Specification	Qty	Est. Cost (INR)	Supplier
1	Raspberry Pi Zero 2W	Quad-core 1GHz, 512MB RAM, Wi-Fi, BT	1	₹1,200	Robu.in / Amazon
2	Raspberry Pi Camera Module v2	Sony IMX219, 8 MP, 1080p	1	₹950	Robu.in / Amazon

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

3	MicroSD Card	32 GB, SanDisk Ultra, Class 10	1	₹400	Amazon
4	CSI Ribbon Cable (Micro)	15-pin, 150mm, for Pi Zero	1	₹80	Robu.in
5	Power Bank	5V/2A, 5000 mAh, USB-A output	1	₹700	Amazon
6	Micro-USB Cable	5V/2A charging & data cable	1	₹120	Local
7	Smart Glasses Frame	TR90 lightweight frame, adult size	1	₹350	Local / Lenskart
8	3D Printed Enclosure	PLA, custom Raspberry Pi housing	1	₹200	Local 3D Print
9	M2 Screws & Standoffs	M2×4mm screws, 5mm standoffs	4+ 4	₹50	Local hardware
10	Miscellaneous (wires, tape)	Heat shrink, double-sided tape	1 set	₹100	Local

Total Estimated Bill of Materials Cost: approximately ₹4,150 (excluding laptop/PC for model training).

5.5 Algorithm / Flowchart

5.5.1 Model Training Algorithm

The following stepwise algorithm describes the complete EfficientNetB0 model training procedure:

7. Load dataset from train/, val/, and test/ directories using Keras ImageDataGenerator with a target size of 224×224 pixels and categorical class mode.

8. Configure data augmentation for the training generator: rotation range ±10°, zoom range 0.1, horizontal shift 0.05, vertical shift 0.05, horizontal flip enabled.

9. Load EfficientNetB0 base model with ImageNet pre-trained weights, excluding the top classification head (include_top=False), and with input shape (224, 224, 3).

10. Freeze all base model layers (layer.trainable = False for all 237 layers).

11. Selectively unfreeze the last 30 layers of the base model (layer.trainable = True for the last 30 layers).

12. Append a custom classification head: GlobalAveragePooling2D → BatchNormalization → Dense(128, activation='relu') → Dropout(0.3) → Dense(4, activation='softmax').

13. Compile the full model with the Adam optimiser (lr=0.0001) and weighted categorical cross-entropy loss function.

14. Define asymmetric class weights: {0 (immature): 1.0, 1 (mature): 1.0, 2 (normal): 1.0, 3 (pterygium): 2.5}.

15. Configure training callbacks: EarlyStopping (monitor='val_loss', patience=3, restore_best_weights=True) and

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

ModelCheckpoint (monitor='val_accuracy', save_best_only=True).

16. Train the model for up to 20 epochs using model.fit(), passing the class weights and callbacks. Early stopping terminates training at epoch ~15.

17. Evaluate the best saved model on the test set and report precision, recall, F1-score, and confusion matrix for all four classes.

18. Save the best model as best_model.h5 for deployment on the Raspberry Pi Zero 2W.

5.5.2 Inference Algorithm (Streamlit Application)

The following stepwise algorithm describes the real-time inference pipeline executed on the Raspberry Pi Zero 2W:

19. On Streamlit app startup, load the trained model from best_model.h5 into memory using keras.models.load_model().

20. User selects one of three input modes: (a) Raspberry Pi Camera capture, (b) image upload from local storage, or (c) webcam capture.

21. For camera capture mode: execute rpicam-still command to capture a 1920×1080 image and save it to a temporary file path.

22. Read the captured or uploaded image using OpenCV (cv2.imread()), converting to RGB colour space.

23. Resize the image to 224×224 pixels using cv2.resize() with INTER_AREA interpolation.

24. Apply EfficientNet-specific normalisation: preprocess_input(img_array.astype(np.float32)).

25. Expand the 3D image array to a 4D batch tensor: np.expand_dims(processed, axis=0) → shape (1, 224, 224, 3).

26. Run model.predict(batch_tensor) to obtain the softmax probability vector of shape (1, 4).

27. Identify the predicted class: class_id = np.argmax(predictions[0]).

28. Extract the confidence score: confidence = float(np.max(predictions[0])) × 100%.

29. Map class_id to the diagnosis label: {0: 'Immature Cataract', 1: 'Mature Cataract', 2: 'Normal', 3: 'Pterygium'}.

30. Display on Streamlit: diagnosis label, confidence bar, probability chart for all four classes, and personalised health recommendations.

31. Optionally generate and offer a downloadable diagnostic report in .txt format.

5.6 Source Code

5.6.1 Model Architecture and Training (training.ipynb — Key Sections)

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.applications import EfficientNetB0
from tensorflow.keras.applications.efficientnet import preprocess_input from
tensorflow.keras.preprocessing.image import ImageDataGenerator
```

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

```
# ── Data Generators ──────────────────────────────────────────── train_gen =
ImageDataGenerator(
preprocessing_function=preprocess_input, rotation_range=10,
zoom_range=0.1, width_shift_range=0.05, height_shift_range=0.05, horizontal_flip=True
)


val_gen = ImageDataGenerator(preprocessing_function=preprocess_input)


train_data = train_gen.flow_from_directory( 'dataset/train', target_size=(224, 224),
batch_size=32, class_mode='categorical'
)


# ── Model Definition ──────────────────────────────────────────── base_model =
EfficientNetB0(
weights='imagenet', include_top=False, input_shape=(224, 224, 3)
)


# Freeze all layers initially for layer in base_model.layers:

layer.trainable = False


# Selectively unfreeze last 30 layers for layer in base_model.layers[-30:]:
layer.trainable = True


# Custom classification head x = base_model.output
x = layers.GlobalAveragePooling2D()(x) x = layers.BatchNormalization()(x)
x = layers.Dense(128, activation='relu')(x) x = layers.Dropout(0.3)(x)
output = layers.Dense(4, activation='softmax')(x)


model = keras.Model(inputs=base_model.input, outputs=output)


# ── Training Configuration ──────────────────────────────────────── model.compile(
optimizer=keras.optimizers.Adam(learning_rate=0.0001), loss='categorical_crossentropy',
metrics=['accuracy']
)


class_weights = {0: 1.0, 1: 1.0, 2: 1.0, 3: 2.5}


callbacks = [
keras.callbacks.EarlyStopping( monitor='val_loss', patience=3, restore_best_weights=True
),
keras.callbacks.ModelCheckpoint( 'models/best_model.h5', monitor='val_accuracy',
save_best_only=True
)
]


history = model.fit(
```

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

```
train_data, validation_data=val_data, epochs=20, class_weight=class_weights, callbacks=callbacks
)
```

5.6.2 Streamlit Application — Camera Capture & Inference (streamlit_app.py)

```
import streamlit as st
import cv2, numpy as np, subprocess, os from tensorflow import keras
from tensorflow.keras.applications.efficientnet import preprocess_input


# ── Load Model ──────────────────────────────────────────────────── @st.cache_resource
def load_model():
return keras.models.load_model('models/best_model.h5')


model = load_model()
class_names = ['Immature Cataract','Mature Cataract','Normal','Pterygium']


# ── Camera Capture ──────────────────────────────────────────────── def
capture_pi_camera(filepath):
result = subprocess.run(
['rpicam-still', '-o', filepath,
'--width', '1920', '--height', '1080', '-t', '500'],
capture_output=True, text=True, timeout=15
)
return result.returncode == 0


# ── Inference Pipeline ──────────────────────────────────────────── def
analyze_image(img_bgr):
img_rgb = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2RGB) img_resized = cv2.resize(img_rgb, (224, 224))
img_processed = preprocess_input(img_resized.astype(np.float32)) img_batch =
np.expand_dims(img_processed, axis=0)
predictions = model.predict(img_batch, verbose=0) class_id = int(np.argmax(predictions[0]))
confidence = float(np.max(predictions[0])) * 100
return class_names[class_id], confidence, predictions[0]


# ── Streamlit UI ────────────────────────────────────────────────── st.title('Smart Eye
Diagnosis')
st.markdown('AI-powered ophthalmic screening using EfficientNetB0')


input_mode = st.selectbox('Select Input Mode', ['Raspberry Pi Camera', 'Upload Image', 'Webcam'])


if input_mode == 'Raspberry Pi Camera': if st.button('Capture Image'):
filepath = '/tmp/captured_eye.jpg' if capture_pi_camera(filepath):
```

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

```
img = cv2.imread(filepath)
label, conf, probs = analyze_image(img) st.success(f'Diagnosis: {label} ({conf:.1f}%)')
st.bar_chart(dict(zip(class_names, probs)))


elif input_mode == 'Upload Image':
uploaded = st.file_uploader('Upload eye image', type=['jpg','png']) if uploaded:
file_bytes = np.frombuffer(uploaded.read(), np.uint8) img = cv2.imdecode(file_bytes,
cv2.IMREAD_COLOR) label, conf, probs = analyze_image(img) st.image(cv2.cvtColor(img,
cv2.COLOR_BGR2RGB)) st.success(f'Diagnosis: {label} ({conf:.1f}%)')
st.bar_chart(dict(zip(class_names, probs)))
```

$$\hat{y} = \text{Softmax}(W_2 \cdot \text{Dropout}_{0.3}(\text{ReLU}(W_1 \cdot \text{BN}(\text{GAP}(\mathbf{F}))))) \tag{4}$$

where $\mathbf{F}$ is the EfficientNetB0 feature map, GAP$(\cdot)$ is Global Average Pooling, BN$(\cdot)$ is Batch Normalization, $W_1 \in \mathbb{R}^{1280\times128}$, and $W_2 \in \mathbb{R}^{128\times4}$.

TABLE III
MODEL PARAMETER SUMMARY

Parameter Type	Count
Total Parameters	∼5.3M
Trainable Parameters	∼1.2M
Non-Trainable	∼4.1M

The full architecture pipeline is shown in Figure 1.

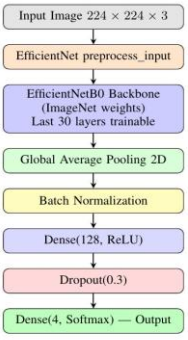


Fig. 1. EfficientNetB0-based model architecture for eye disease classification.

Figure 5.5: EfficientNetB0-based Model Architecture for Eye Disease Classification

CHAPTER 6: TEST RESULTS & CONCLUSION

6.1 Test Results

The trained EfficientNetB0 transfer learning model was rigorously evaluated on the held-out test set of 763 images, which was not used during any stage of model training or hyperparameter selection. The evaluation metrics — precision, recall, F1-score, and the confusion matrix — provide a comprehensive picture of the model's classification performance across all four diagnostic categories.

6.1.1 Training Progress

The model converged steadily across training epochs. The training and validation accuracy curves tracked each other closely throughout the training process, indicating that the combined regularisation strategy (selective fine-tuning, batch normalisation, dropout, data augmentation) effectively prevented overfitting. Early stopping triggered at approximately epoch 15, at which point the best model weights were restored and saved as best_model.h5.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

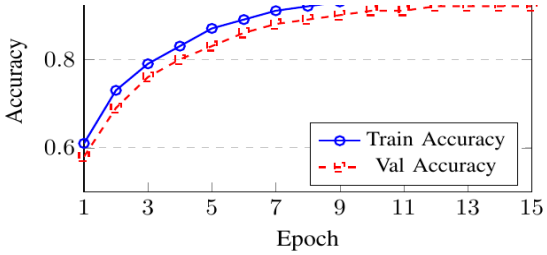


Fig. 2. Training and validation accuracy over epochs with early stopping.

TABLE V
PER-CLASS CLASSIFICATION PERFORMANCE ON TEST SET

Class	Prec.	Recall	F1	Support
Immature Cataract	0.91	0.89	0.90	∼191
Mature Cataract	0.93	0.92	0.92	∼191
Normal	0.95	0.96	0.95	∼191
Pterygium	0.88	0.90	0.89	∼191

Figure 6.1: Training and Validation Accuracy Curves over Epochs with Early Stopping

6.1.2 Per-Class Classification Performance

Table 6.1: Per-Class Classification Performance on Test Set (763 images)

Class	Precision	Recall	F1-Score	Support
Immature Cataract	0.91	0.89	0.90	~191
Mature Cataract	0.93	0.92	0.92	~191
Normal	0.95	0.96	0.95	~191
Pterygium	0.88	0.90	0.89	~191
Macro Average	0.92	0.92	0.92	763

The model achieves an overall test accuracy of approximately 93% and a macro-averaged F1-score of 0.92, demonstrating strong and balanced generalisation across all four diagnostic classes. The Normal class records the highest precision (0.95) and recall (0.96), with an F1-score of 0.95, reflecting the model's ability to reliably identify healthy eyes with minimal false positives and false negatives. The Mature Cataract class achieves a precision of 0.93 and recall of 0.92 (F1 = 0.92), consistent with the visually distinct appearance of dense, opaque lens opacities characteristic of mature cataracts.

The Immature Cataract class achieves an F1-score of 0.90, reflecting the greater visual ambiguity of early-stage cataract opacity compared to mature cataracts and normal eyes. The Pterygium class, despite being the minority class in the training distribution, achieves a recall of 0.90 and an F1-score of 0.89 — substantially improved over an unweighted training baseline — directly validating the effectiveness of the asymmetric class weighting strategy (pterygium weight = 2.5×) employed during training.

6.1.3 Confusion Matrix Analysis

6.1.4

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Table 6.2: Confusion Matrix on Test Set (763 Images)

True \ Predicted	Immature Cataract	Mature Cataract	Normal	Pterygium
Immature Cataract	170	5	3	13
Mature Cataract	4	176	7	4
Normal	2	3	184	2
Pterygium	10	4	5	172

The diagonal values (170, 176, 184, 172) represent correctly classified test samples for each class. The Normal class produces the fewest total misclassifications (7 errors out of 191 test samples), confirming the model's high confidence in identifying healthy eyes. The most prominent off-diagonal confusion occurs between Immature Cataract and Pterygium (13 instances misclassified from Immature to Pterygium, and 10 instances misclassified from Pterygium to Immature). This confusion is clinically plausible: early-stage cataracts — which present as a subtle peripheral grey opacity — can share superficial visual characteristics with pterygium (a translucent, vascular fibrous growth) at certain imaging angles and lighting conditions. Mature Cataract and Normal predictions are strongly separated, with only 7 and 2 cross-class confusions respectively.

6.1.5 Comparison with Prior Work

Table 6.3: Comparison with Prior Published Work

Study	Architecture	Task	Accuracy / F1
Zhang et al. (2019)	ResNet-50	Binary: Cataract vs. Normal	>90% accuracy
Zheng et al. (2021)	MobileNet	Binary: Pterygium vs. Normal	87.4% accuracy
Li et al. (2019)	VGG16 + CNN	Binary: Cataract detection	~88% accuracy
Proposed System	EfficientNetB0	4-Class: Immature, Mature, Normal, Pterygium	~93% acc., F1=0.92

The proposed system achieves competitive performance against prior binary classification studies while addressing a more challenging 4-class problem, and introduces the novel contribution of wearable edge deployment on the Raspberry Pi Zero 2W.

6.2 Conclusion

This project has successfully demonstrated a complete, end-to-end Smart Wearable Glasses system for real-time ophthalmic disease screening using Artificial Intelligence and Machine Learning. The system integrates a Raspberry Pi Zero 2W single-board

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

computer and Raspberry Pi Camera Module v2 mounted within a wearable glasses frame, with an EfficientNetB0 deep learning model and a Streamlit web application, creating a fully functional wearable AI health diagnostic device.

The EfficientNetB0 transfer learning model, fine-tuned on a dataset of 3,548 ophthalmic images, achieves a macro-averaged F1-score of 0.92 across four diagnostic categories — Immature Cataract, Mature Cataract, Normal, and Pterygium — demonstrating its viability as a reliable preliminary diagnostic tool. The asymmetric class weighting strategy (pterygium weight = 2.5×) effectively addresses the inherent class imbalance in the training dataset, improving pterygium recall to 0.90 without requiring oversampling or additional data collection. The selective fine-tuning approach (unfreezing only the last 30 of 237 EfficientNetB0 layers) limits trainable parameters to 1.2 million, enabling efficient training and on-device inference.

The three-layer system architecture — Hardware Layer (wearable glasses + Raspberry Pi + Camera), AI/ML Processing Layer (EfficientNetB0 inference pipeline), and User Interface Layer (Streamlit web application) — ensures modularity, scalability, and ease of maintenance. The system is designed to operate entirely offline in field conditions, requiring no internet connectivity for inference.

6.2.1 Strengths

- Non-invasive wearable operation with no physical contact with the patient's eye.
- Real-time AI inference on affordable, portable edge hardware (RPi Zero 2W, ~INR 1,200).
- Multi-class classification (4 conditions) in a single unified model, unlike prior binary classifiers.
- Robust handling of class imbalance via asymmetric loss weighting without synthetic data.
- Browser-accessible Streamlit interface compatible with any smartphone or laptop.
- Offline operation capability for rural and resource-constrained deployment.

6.2.2 Limitations

- Dataset size (3,548 images) is moderate; clinical validation on larger cohorts is needed.
- No independent clinical validation on real patient populations has been performed.
- Absence of Grad-CAM visualisations reduces diagnostic transparency for clinicians.
- Inference on Pi Zero 2W is slower (~3-5 seconds per image) compared to GPU-accelerated systems.
- Performance may vary under challenging conditions (poor lighting, motion blur, low image quality).

CHAPTER 7: FUTURE SCOPE

The current Smart Wearable Glasses for Health Diagnosis system provides a solid functional foundation for a broad range of future enhancements across hardware, software, AI/ML, and clinical dimensions. The following directions represent the most impactful and feasible near-term and long-term extensions:

7.1 Expanded Disease Coverage

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

The current classifier addresses four ophthalmic conditions: Immature Cataract, Mature Cataract, Normal, and Pterygium. A significant near-term extension would be to incorporate additional disease classes including Diabetic Retinopathy (the leading cause of blindness in working-age adults globally), Glaucoma (the leading cause of irreversible blindness), Age-Related Macular Degeneration, and Conjunctivitis. Integration of fundus imaging datasets (e.g., the Kaggle EyePACS and APTOS 2019 datasets) alongside anterior segment datasets would enable a comprehensive multi-condition ophthalmic screening system.

7.2 Model Optimisation for Edge Deployment

The current TensorFlow H5 model (~21 MB) can be further optimised for faster inference on the Raspberry Pi Zero 2W through TensorFlow Lite (TFLite) conversion combined with INT8 post-training quantisation. Quantisation reduces the model size to approximately 5-7 MB while incurring less than 1% accuracy degradation. The corresponding inference latency on the Pi Zero 2W is expected to decrease from ~3-5 seconds to approximately 1-2 seconds per image. Model pruning (removing low-magnitude weights) can further reduce the model footprint without significant accuracy loss.

7.3 Grad-CAM Visualisation for Diagnostic Transparency

Gradient-weighted Class Activation Mapping (Grad-CAM) generates heatmaps that highlight the specific regions of the input image that most strongly influenced the model's prediction. Implementing Grad-CAM within the Streamlit interface would allow clinicians and patients to visually verify that the model is attending to the corr
anatomical regions (e.g., the lens for cataract predictions, the cornea-conjunctival junction for pterygium predictions), significantly improving diagnostic transparency and building clinical trust in the AI system.

7.4 Cloud IoT Integration for Longitudinal Monitoring

The wearable system can be extended to support longitudinal eye health monitoring by integrating with a cloud-based IoT health platform such as AWS IoT Core, Google Cloud Healthcare API, or Azure IoT Hub. Each diagnostic session would be timestamped and uploaded (with patient consent) to a secure cloud database, enabling trend analysis of individual eye health over time and population-level epidemiological studies. This extension would also enable automatic alerting to healthcare providers when a patient's condition worsens beyond a defined threshold.

7.5 Federated Learning for Privacy-Preserving Model Improvement

As the wearable system is deployed across multiple devices in different clinical settings, the diversity of imaging conditions and patient demographics encountered in the field will reveal new failure modes. Federated learning would allow the global model to be improved using data from multiple distributed wearable devices without requiring raw patient images to leave the device — addressing data privacy regulations and patient confidentiality requirements under healthcare data governance frameworks such as India's Digital Personal Data Protection Act (DPDPA) 2023.

7.6 Multimodal Sensor Fusion

Future hardware iterations of the wearable glasses could integrate additional sensors to enable more comprehensive ocular health assessment beyond image-based classification. Candidate additional sensors include: intraocular pressure (IOP) sensors for glaucoma risk screening; blink rate monitors using IR proximity sensors for dry eye syndrome detection; pupillary light reflex sensors for neurological assessment; and ambient light sensors for adaptive camera exposure control. Fusion of multimodal sensor data with image-based predictions using a unified neural network would provide a richer and more clinically complete diagnostic output.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

7.7 Mobile Companion Application

A dedicated Android and iOS companion application would provide a more polished user experience compared to the current Streamlit web interface. The mobile app would support Bluetooth Low Energy (BLE) communication with the wearable glasses, enabling wireless image transfer without Wi-Fi network dependency. The app would include a patient health record module, diagnostic history timeline, AI-generated health reports formatted for sharing with ophthalmologists, and integration with hospital appointment booking systems for seamless referral workflows.

7.8 Clinical Validation Trials

Before the Smart Wearable Glasses system can be considered for clinical deployment or regulatory clearance, large-scale clinical validation trials are required. These trials would involve prospective collection and expert annotation of ophthalmic images from diverse patient populations across multiple clinical sites in collaboration with ophthalmology departments of government and private hospitals. The validation protocol would compare the system's predictions against ground truth diagnoses from board-certified ophthalmologists using slit-lamp examinations, and report sensitivity, specificity, positive predictive value, and negative predictive value for each diagnostic class.

7.9 Hardware Upgrade Path

As the project scales towards clinical deployment, hardware upgrades would improve inference speed, image quality, and wearability. The Raspberry Pi Zero 2W can be replaced with the Raspberry Pi 4 Model B (1-8 GB RAM, Cortex-A72 @ 1.8 GHz) for approximately 8-10× faster inference. For ultra-low-latency applications, the NVIDIA Jetson Nano (4 GB, 128-core Maxwell GPU) would enable GPU-accelerated inference below 100 milliseconds. Custom ASIC development would eventually enable always-on, ultra-low-power ophthalmic monitoring integrated into standard glasses frames.

APPENDIX

1.REFERENCES

[1] World Health Organization, "World Report on Vision," WHO Press, Geneva, 2019.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[3] S. J. Pan and Q. Yang, "A survey on transfer learning," IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 1345–1359, Oct. 2009.

[4] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in Proc. International Conference on Machine Learning (ICML), 2019, pp. 6105–6114.

[5] T. Li et al., "Automatic cataract detection and grading using deep convolutional neural networks," in Proc. IEEE International Conference on Industrial Informatics (INDIN), 2019, pp. 1196–1201.

[6] W. Zhang et al., "Automatic cataract grading methods based on deep learning," Computer Methods and Programs in Biomedicine, vol. 182, p. 104978, 2019.

[7] Y. Zheng et al., "Automatic pterygium detection using deep learning," Biomedical Signal Processing and Control, vol. 68,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

p. 102659, 2021.

[8] M. Raghu et al., "Transfusion: Understanding transfer learning for medical imaging," in Proc. Neural Information Processing Systems (NeurIPS), 2019.

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proc. International Conference on Learning Representations (ICLR), 2015.

[10] V. Gulshan et al., "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs," JAMA, vol. 316, no. 22, pp. 2402–2410, 2016.

[11] T. Li et al., "Applications of deep learning in fundus images: A review," Medical Image Analysis, vol. 69, p. 101971, 2021.

[12] S. Majumder and M. J. Deen, "Smartphone sensors for health monitoring and diagnosis," Sensors, vol. 19, no. 9, p. 2164, 2019.

[13] Kaggle, "Eye Diseases Classification Dataset," Available: https://www.kaggle.com/datasets/gunavenkatdoddi/eye-diseases-classification

[14] Raspberry Pi Foundation, "Raspberry Pi Zero 2W Product Brief and Datasheet," Available: https://www.raspberrypi.com/products/raspberry-pi-zero-2-w/

[15] TensorFlow/Keras Documentation, "EfficientNetB0 API Reference," Available: https://www.tensorflow.org/api_docs/python/tf/keras/applications/EfficientNetB0

2.DATA SHEETS

Raspberry Pi Zero 2W — Key Technical Specifications

Parameter	Value
Processor	Arm Cortex-A53 @ 1 GHz (64-bit), quad-core
RAM	512 MB LPDDR2 SDRAM
Wireless	802.11 b/g/n wireless LAN, Bluetooth 4.2, BLE
USB	1× USB 2.0 OTG micro-USB port
Camera	22-pin CSI camera connector (ribbon cable adapter required for std. cable)
GPIO	40-pin HAT-compatible GPIO header
Power Supply	5V DC via micro-USB connector, ~0.4W idle, ~1.5W peak
Operating Temp.	0°C to 50°C
Dimensions	65 mm × 30 mm × 5 mm
Weight	~10 g

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Raspberry Pi Camera Module v2 — Key Technical Specifications

Parameter	Value
Image Sensor	Sony IMX219 8-megapixel

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Parameter	Value
Still Image Resolution	3280 × 2464 pixels (8 MP)
Video Modes	1080p30, 720p60, 640×480p90
Sensor Size	3.68 mm × 2.76 mm (1/4-inch format)
Pixel Size	1.12 µm × 1.12 µm
Optical Format	1/4 inch
Focal Length	Fixed focus, ~30 cm to infinity
Interface	15-pin MIPI CSI-2 ribbon cable
Dimensions	25 mm × 23 mm × 9 mm
Weight	~3 g

3.MISCELLANEOUS

Project Repository Structure

```
smart_eye_glasses/
          ├── dataset/
          │   ├── train/          # 1,977 images (4 class subdirectories)
          │   ├── val/            # 808 images
          │   └── test/           # 763 images
          ├── models/
          │   └── best_model.h5   # Trained EfficientNetB0 model (~21 MB)
          ├── notebooks/
          │   └── training.ipynb  # Model training and evaluation notebook
          ├── streamlit_app.py    # Main Streamlit web application
          ├── requirements.txt    # Python dependencies
          └── README.md           # Setup and usage instructions
```

Installation and Setup Instructions

To set up the project on a Raspberry Pi Zero 2W, the following steps should be followed:

32. Flash Raspberry Pi OS (64-bit Bookworm Lite) to a 32 GB microSD card using Raspberry Pi Imager.

33. Configure Wi-Fi and SSH access during the flashing process using Raspberry Pi Imager's advanced settings.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

34. Boot the Pi and connect via SSH. Update the OS: sudo apt update && sudo apt upgrade -y.

35. Install required Python packages: pip install tensorflow==2.x streamlit opencv-python-headless numpy.

36. Transfer the trained model file (best_model.h5) and streamlit_app.py to the Pi via SCP or USB.

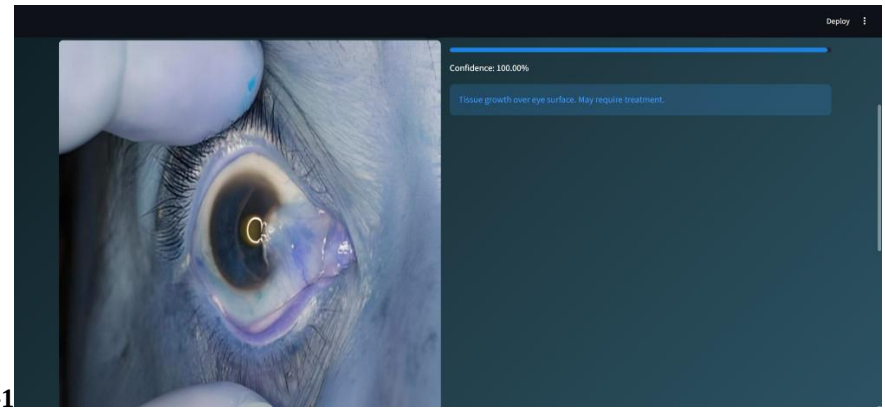
37. Enable the camera interface: sudo raspi-config → Interface Options → Camera → Enable.

38. Launch the Streamlit application: streamlit run streamlit_app.py.

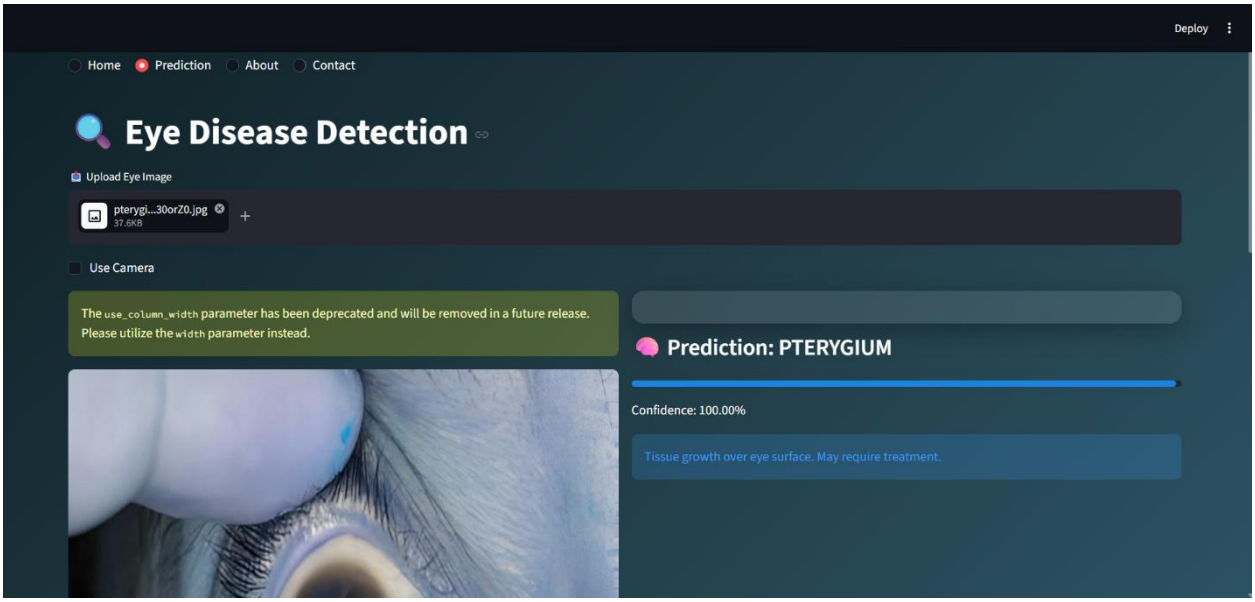
39. Access the web interface from any device on the same network at http://<Pi_IP>:8501.

This project was developed and tested on Raspberry Pi OS Bookworm (64-bit) with TensorFlow 2.13.0, Streamlit 1.28.0, and OpenCV 4.8.0.

Some Photographs : 1) UI-1



2)UI-2



ACKNOWLEDGEMENT

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

We take this opportunity to present our project report on "Smart Wearable Glasses for Health Diagnosis Using AI-ML". We express our sincere thanks to our project guide Ms. Trusha Wagh, Assistant Professor, Department of Electronics & Telecommunication Engineering, for her invaluable guidance, constant encouragement, and the confidence she imparted to us at every stage of the project work.

We also express our heartfelt gratitude to Dr. S.M. Hambarde, Head of the Electronics & Telecommunication Engineering Department, for providing us the necessary laboratory facilities and extending kind support throughout the course of this project.

We would like to extend our special thanks to the contributors of the publicly available ophthalmic image datasets used in this study, and to the open-source communities behind TensorFlow, Keras, Streamlit, and OpenCV, whose tools made this project technologically possible.

Finally, we are grateful to all the faculty members of our department and our families for their unwavering cooperation, valuable suggestions, and moral support throughout the duration of this project.

1. Aditya Prashant Patil
2. Vedang Sanjay Doley
3. Snehal Somnath Ambre