# Embedded System-Based Malaria Detection From Blood Smear Images Using Lightweight Deep Learning Model

**Article** *in* International Journal of Imaging Systems and Technology · October 2024

DOI: 10.1002/ima.23205

CITATIONS
4

READS
255

**7 authors**, including:

Abdus Salam
Rajshahi University of Engineering and Technology
**15** PUBLICATIONS **96** CITATIONS
SEE PROFILE

Md. Jawadul Karim
BRAC University
**6** PUBLICATIONS **104** CITATIONS
SEE PROFILE

Md Shamim Anower
Rajshahi University of Engineering and Technology
**215** PUBLICATIONS **3,979** CITATIONS
SEE PROFILE

Md. Nahiduzzaman
Rajshahi University of Engineering and Technology
**60** PUBLICATIONS **1,481** CITATIONS
SEE PROFILE

*International Journal of Imaging Systems and Technology*

WILEY

RESEARCH ARTICLE

# Embedded System-Based Malaria Detection From Blood Smear Images Using Lightweight Deep Learning Model

Abdus Salam[1,2] | S. M. Nahid Hasan[1] | Md. Jawadul Karim[1] | Shamim Anower[3] | Md Nahiduzzaman[1] | Muhammad E. H. Chowdhury[2] | M. Murugappan[4,5]

[1]Department of Electrical and Computer Engineering, Rajshahi University of Engineering & Technology, Rajshahi, Bangladesh | [2]Department of Electrical Engineering, Qatar University, Doha, Qatar | [3]Department of Electrical and Electronic Engineering, Rajshahi University of Engineering and Technology, Rajshahi, Bangladesh | [4]Intelligent Signal Processing (ISP) Research Lab, Department of Electronics and Communication Engineering, Kuwait College of Science and Technology, Doha, Kuwait | [5]Department of Electronics and Communication Engineering, Vels Institute of Sciences, Technology, and Advanced Studies, Chennai, Tamilnadu, India

**Correspondence:** Muhammad E. H. Chowdhury (mchowdhury@qu.edu.qa) | M. Murugappan (m.murugappan@kcst.edu.kw)

**Received:** 20 March 2024 | **Revised:** 19 August 2024 | **Accepted:** 10 October 2024

**Funding:** The authors received no specific funding for this work.

**Keywords:** deep learning | embedded system | lightweight model | malaria parasite | SqueezNet model

## ABSTRACT

The disease of malaria, transmitted by female Anopheles mosquitoes, is highly contagious, resulting in numerous deaths across various regions. Microscopic examination of blood cells remains one of the most accurate methods for malaria diagnosis, but it is time-consuming and can produce inaccurate results occasionally. Due to machine learning and deep learning advances in medical diagnosis, improved diagnostic accuracy can now be achieved while costs can be reduced compared to conventional microscopy methods. This work utilizes an open-source dataset with 26 161 blood smear images in RGB for malaria detection. Our preprocessing resized the original dimensions of the images into 64 × 64 due to the limitations in computational complexity in developing embedded systems-based malaria detection. We present a novel embedded system approach using 119 154 trainable parameters in a lightweight 17-layer SqueezeNet model for the automatic detection of malaria. Incredibly, the model is only 1.72 MB in size. An evaluation of the model's performance on the original NIH malaria dataset shows that it has exceptional accuracy, precision, recall, and F1 scores of 96.37%, 95.67%, 97.21%, and 96.44%, respectively. Based on a modified dataset, the results improved further to 99.71% across all metrics. Compared to current deep learning models, our model significantly outperforms them for malaria detection, making it ideal for embedded systems. This model has also been rigorously tested on the Jetson Nano B01 edge device, demonstrating a rapid single image prediction time of only 0.24 s. The fusion of deep learning with embedded systems makes this research a crucial step toward improving malaria diagnosis. In resource-constrained settings, the model's lightweight architecture and accuracy enhancements hold great promise for addressing the critical challenge of malaria detection.

## 1 | Introduction and Related Works

Malaria is among the most dangerous diseases facing humanity [1]. Globally, approximately half of the population faces malaria risks, according to the World Health Organization (WHO). A staggering 619 000 lives were lost to malaria in 2021 alone, resulting in an estimated 247 million cases documented worldwide. This deadly disease disproportionately affects the African region [2]. Figure 1 summarizes malaria deaths by age group. Malaria claims the lives of a child under the age of five almost every minute, and many of these deaths are not only preventable but also treatable [3]. Around a thousand children die every day

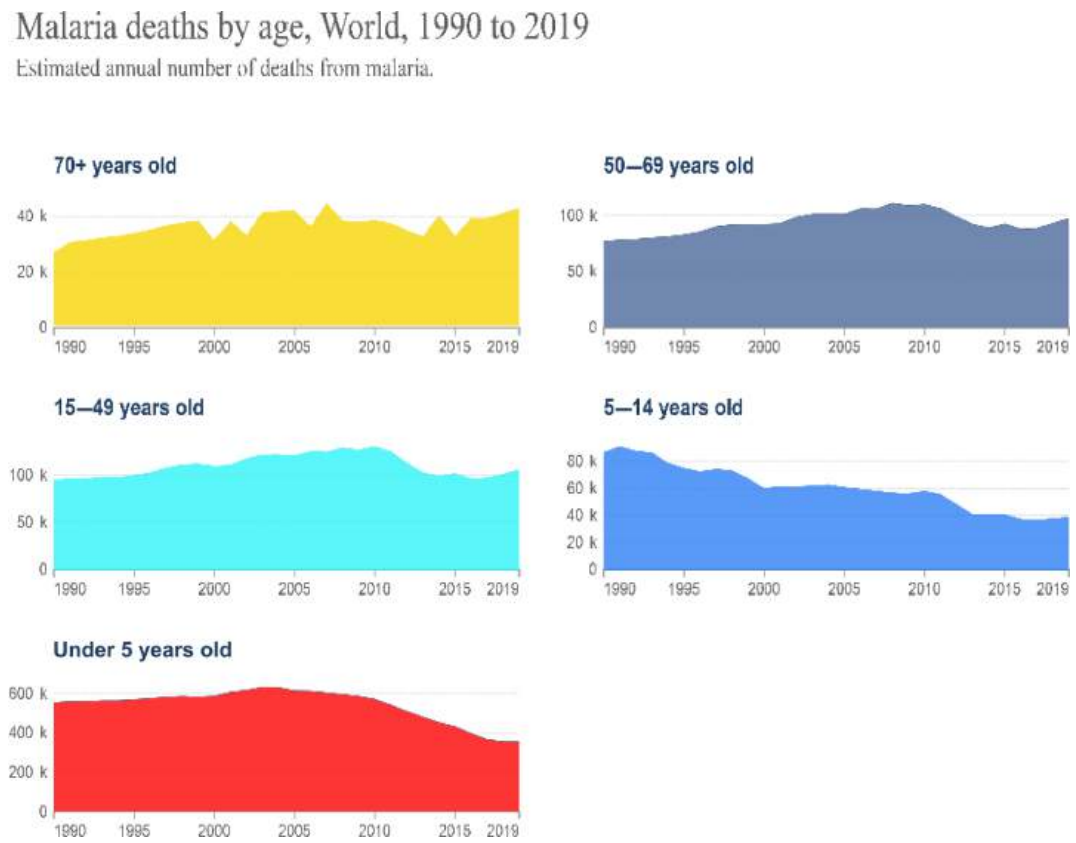


**FIGURE 1** | Malaria death rates by age [4].

because of these tragic accidents under the age of five, constituting a startling 77% of all these tragic deaths.

Plasmodium, a parasitic agent, is the root cause of malaria. Humans can be infected by multiple Plasmodium species, including Plasmodium falciparum, Plasmodium vivax, Plasmodium malaria, and Plasmodium ovale [5]. These parasites usually enter the human body through bites from female Anopheles mosquitoes [6]. Humans contract malaria when an infected mosquito bites them and introduces sporozoites. Sporozoites migrate to the liver, where they target hepatocytes and undergo schizogony. As merozoites enter the bloodstream, they infect erythrocytes and initiate asexual reproduction. Figure 2 shows the blood cells of a normal individual and those with malaria. As a result of mating, a zygote is formed, which then matures into an ookinete. The ookinete of a mosquito develops into an oocyst after passing through its midgut epithelium. Due to the remarkable and intricate life cycle of the malaria parasite, the disease is transmitted to humans by mosquitoes.

Malaria's devastating impact on human lives, especially the youngest and most vulnerable, demonstrates the need for increased research, public health interventions, and innovative strategies. Malaria parasites include *P. falciparum* and *P. vivax*, which are the most pernicious [7]. Those bitten by mosquitoes carrying these parasites can lie dormant for 10–15 days or even longer after being bitten [8]. The impact of malaria on our youngest population should not be overlooked. A study by the Institute for Health Metrics and Evaluation (IHME) found most malaria-related deaths occurred in children under five [9].

There are three types of malaria: asymptomatic (simple), complicated (severe), and asymptomatic (complicated). There are early symptoms such as fever, chills, muscle aches, and gastrointestinal issues in children. Symptoms include excessive sweating, high fever, and fatigue, and can occur suddenly. Malaria is commonly diagnosed using microscopic examination of stained blood films and rapid diagnostic tests (RDTs). The gold standard for malaria diagnosis is microscopically examining thick and thin blood films. Due to their strong diagnostic capabilities, wide availability, and affordability, RDTs are widely used. Compared with thin blood cell microscopy, antigen-based rapid diagnostic procedures are more expensive and less accurate in resource-constrained settings. A cost-effective, efficient, and accurate malaria diagnostic system is urgently needed. Machine learning (ML) and deep learning (DL) techniques have been explored to detect life-threatening diseases in the past decade. Due to its severity, automated malaria detection has received considerable attention. The effectiveness and accuracy of disease classification has improved significantly since DL models were introduced [10, 11]. Convolutional operations allow DL approaches to extract specific patterns from malaria-infected red blood cells. This section discusses several recent studies focused on malaria disease detection:

Sumagna et al. [12] proposed a model of deep convolutional neural networks (CNNs) based on the ResNet152 architecture. Their Deep Greedy Network model was trained on 27 558 images of malarial cells with balanced infected and uninfected classes. The model detected malaria with 98.25% accuracy. AOCT-NET is a lightweight CNN model that leverages transfer learning to classify malaria-infected cells developed by Alqudah et al. [13].

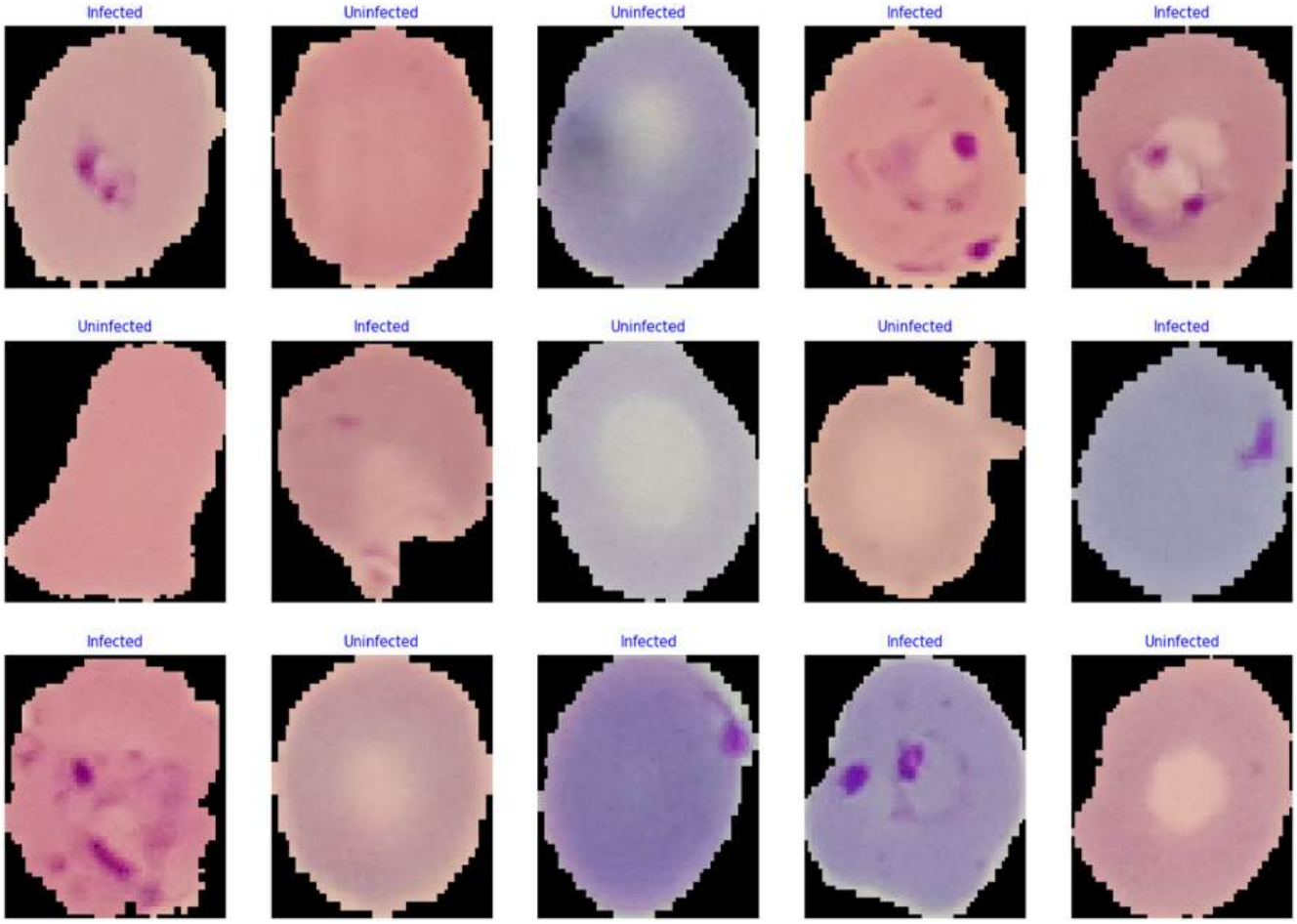


**FIGURE 2** | Infected and healthy blood cells.

Their dataset contained approximately 27 558 segmented images of RBCs. This model achieved an astounding accuracy of 98.90% with a learning rate of 0.001. Jain et al. [14] proposed a low-cost CNN that did not require preprocessing or GPU processing. In tests with 27 558 balanced classes, ResNet outperformed K-nearest neighbors (KNN) and support vector machines (SVM) as well as genetic algorithms-based models. Using stacked convolutional neural networks, Muhammad Umer et al. [15] developed an autonomous malaria detection system. The dataset included 27 558 parasitized and uninfected cells, resulting in 100% precision, 99.9% recall, and 99% F1-measure. Zhaohui Liang et al. [16] proposed a CNN model with 16 layers for the detection of malarial parasites. Models trained on around 27 000 images exhibited 97.0% accuracy, exceeding transfer learning methods. The method proposed by Gopakumar et al. [17] used a CNN based on a focus stacking algorithm. By focusing on suspected parasite positions, the computational cost was reduced. Using nine layers and a soft-max log-loss layer, this CNN architecture achieved 98.50% specificity, 97.06% sensitivity, and 0.7305 Matthew Correlation Coefficient (MCC), outperforming SVM and other patched approaches.

Transfer learning has demonstrated its value in these specific situations, improving classification performance. The use of deep networks has been demonstrated to be effective in the detection of malaria, with notable accuracy and precision [13, 18–30]. Omaer et al. [8] proposed the extreme learning machine (ELM) for the diagnosis of malaria. Their deep ELM network, which consists of two layers of 500 nodes each, achieved 97.79% accuracy. They did not achieve the same accuracy with CNN or CNN combined with ELM. Sumagna et al. [12] achieved impressive accuracy (98.25%), as well as high precision and recall (98.40% and 98.11%, respectively). It has been demonstrated that ResNet-152 and Deep Greedy Networks enhance learning and performance. A falcon architecture proposed by Banerjee et al. [14] consists of nine layers and 984 961 parameters. Alassaf and Sikkandar presented an intelligent deep transfer learning model (IDTL-MPDC) for detecting malaria parasites in blood smear images [18]. A residual neural network and the differential evolution (DE) approach were used to optimize model hyperparameters, resulting in good accuracy (95.86%), sensitivity (95.82%), specificity (95.98%), F1 score (95.69%), and precision (95.86%).

Vijayalakshmi et al. [19] combined VGG-19 with an SVM classifier to detect malaria in microscopic blood cell images. They detected malaria with a maximum accuracy of 93.13%. An accuracy of 89.21% is achieved with VGG16 and SVM. Using 10-fold cross-validation, a CNN, and a fine-tuned version of the CNN architecture helped Gautham et al. [20] detect malaria automatically with 94% and 96% accuracy. Amal H. Alharbi et al. [21] explored CNN, SVM, and XGBoost for malaria detection. XGBoost achieved 90% accuracy, while SVM achieved 94%. However, CNN that was specifically designed outperformed both, achieving 97% accuracy. Bibin et al. presented a novel model using a

deep belief network (DBN) pre-trained using the contrastive divergence technique [22]. Their 484-600-600-600-600-2 architecture outperformed other models, achieving an F-score of 89.66%. In comparison with state-of-the-art CNN models, this model achieved 95.2% accuracy without overfitting. According to Ibrar Amin et al. [23], a semi-supervised GAN-based architecture was developed for malaria classification. The model achieved 96.6% accuracy with good precision (95.53%) with less training data than standard deep learning models. Detecting malaria parasites using a multiheaded attention transformer was proposed by Robiul et al. [24]. This study used 27 588 RBC images and achieved a maximum accuracy of 99.25% using the SGD optimizer. Krishnadas et al. [25] applied YOLOv5 and YOLOv4 to malaria parasite classification and stage progression. The first dataset had an accuracy rate of 78.5% and the second had an accuracy rate of 83%.

A comparative analysis involving Naive Bayes, KNN, SVM, Artificial Neural Networks (ANN), and a hybrid model was conducted by Shuleenda et al. [26]. The authors analyzed 1302 microscopic erythrocyte images from 400 blood smears, equally distributed across seven classes. Combining these techniques resulted in the best accuracy of 96.54%. Using genetic algorithms with Principal Component Analysis (PCA) and independent component analysis (ICA) for gene sample classification, Arowolo et al. proposed an innovative approach [28]. Anopheles gambiae mosquito data with seven features and 2457 samples were used in this study. PCA with KNN achieved 90% accuracy over ICA with KNN.

Several imperative reasons make deep learning (DL) particularly effective in the health sector. A big advantage of this technique is its ability to handle and evaluate extensive, intricate biomedical datasets more efficiently than any conventional machine learning (ML) technique [29]. Rajaraman et al. [30] evaluate several pre-trained CNN models for classifying parasitized and uninfected cells, showing promising results for automated malaria screening. DL eliminates the need for labor-intensive and biased manual feature engineering by extracting features autonomously from unprocessed data [30, 31]. A notable application of DL algorithms is in medical imaging, where they have demonstrated exceptional performance in diagnosing ailments based on X-rays, MRIs, and CT scans. In some cases, DL models surpass or are equivalent to the diagnostic accuracy of seasoned radiologists [31, 32]. Furthermore, DL's ability to analyze unstructured data, such as free-text clinical notes from electronic health records (EHRs), enables it to identify crucial clinical information and predict patient outcomes more accurately than conventional approaches. The DL platform demonstrates exceptional proficiency in genomics and drug discovery by analyzing large volumes of clinical and genomic data and revealing new insights. In addition, this enhances the development of precision medicine by expediting the identification of pharmaceutical candidates and individualized treatment strategies. Moreover, its adaptability and ongoing learning enhance its predictive precision and applicability, which makes it an essential tool for developing resilient healthcare solutions.

Based on our review of recent state-of-the-art (SOTA) research, we have identified and discussed the following challenges in the detection of automated malaria-infected blood cells:

- *Challenge 1*: One of the key challenges in our research is the high-performance CPUs and GPUs needed for most image classification models. However, our research focuses on embedded systems, which often have limited computational capabilities and memory.
- *Challenge 2*: Although reducing model parameters saves time, it often reduces output accuracy. As a result, one of our major challenges is reducing the parameters of our model without compromising its accuracy.
- *Challenge 3*: Because the model will be used in pathology detection in clinical practice, precision, and recall measures are of the utmost importance in achieving the most accurate performance. Embedded software must be bug-free and lightweight so that the system runs smoothly.

Health care is becoming increasingly dependent on the Internet of Things (IoT). Traditional disease detection, health monitoring, and tracking methods are time-consuming, outdated, and offer suboptimal results. As a result, researchers are developing intelligent classification systems using IoT technology. The objective of this study is to develop a lightweight embedded system framework to detect malaria. This framework can be used to determine whether a cell is infected with malaria or not. With the technique, thin blood smear samples can be classified quickly and easily, reducing the time and effort involved in malaria pathology diagnosis. This study develops a lightweight CNN model for malaria detection using microscopic blood cell images. The lightweight model is then deployed on a Jetson Nano IoT device. The model was selected based on its size and IoT compatibility. Our model's compactness and exceptional accuracy in detecting malaria in microscopic blood images are noteworthy features.

The following contributions are noteworthy in this paper:

- We developed lightweight SqueezeNet models tailored to embedded systems for malaria detection from blood smear images.
- A real-world application of the proposed model has been successfully implemented on the Jetson Nano platform, demonstrating its practicality.
- Upon deployment in the embedded system, our proposed model demonstrated superior performance. Although the model size is small (1.72 MB), it is remarkably fast at compiling and testing, achieving impressive classification results.

## 2 | Materials and Methods

### 2.1 | Proposed Approach

During the last few decades, deep learning technologies have revolutionized the field of medicine. The purpose of this study is to address these challenges by developing a lightweight CNN model tailored especially for resource-constrained environments. In this work, we aim to build a model that reduces computational complexity and uses the least amount of memory possible. The lightweight model has been successfully implemented on the Jetson Nano IoT device, demonstrating its

practicality for deployment in compact embedded systems with limited resources. To facilitate disease detection, small, autonomous embedded systems are becoming increasingly popular as technology advances. In medical diagnosis, embedded systems are well-suited for building efficient mobile devices. Hardware limitations are the primary challenge for embedded systems, which are often compact and power efficient. Because of this constraint, there is a limited amount of storage space and computational power available. A lightweight model with reduced computational requirements is essential for seamless embedded system operation. Figure 3 illustrates the workflow of our comprehensive methodology.

A workflow can be described as follows: the process begins with the loading of the dataset from the directory. Upon initial inspection of the dataset, inaccuracies were discovered, including misclassified images. Subsequently, modifications were made to the dataset, and the refined dataset was used for subsequent training. The pre-processing of images involved resizing them to $64 \times 64 \times 3$ dimensions and scaling them. Following this, the dataset was randomly split into training, testing, and validation sets, at 80%, 10%, and 10%, respectively. Next, the lightweight SqueezeNet model was used in the training process. The best model was saved, and comprehensive classification reports and accuracy percentages were calculated.

## 2.2 | Dataset Description

A thin blood smear dataset was extracted from the National Library of Medicine archives and was derived from 148 individuals infected with Falciparum and 45 from uninfected individuals [29]. Segmentation of this dataset was performed by Rajaraman et al. [30]. Thereafter, we acquired a larger dataset comprising approximately 27 558 RBC images with an even distribution between malaria-parasitized and uninfected patients. According to the research conducted by Fuhad et al. [31], some inaccuracies in labeling were identified in this extensive data set. There were 1397 falsely labeled images, including 647 falsely labeled infected images and 750 mislabeled uninfected images. To ensure dataset integrity, erroneously labeled images were systematically removed from the original dataset. In Table 1, both the original and modified datasets are described according to their class distributions. In this study, the modified dataset by Fuhad et al. [31] is used for training and testing.

## 2.3 | Image Pre-Processing

An image-based classification system is highly dependent on the quality and dimensions of the images in the dataset. As a fixed image size is typically required for consistent training and prediction, variations in image sizes within a dataset can

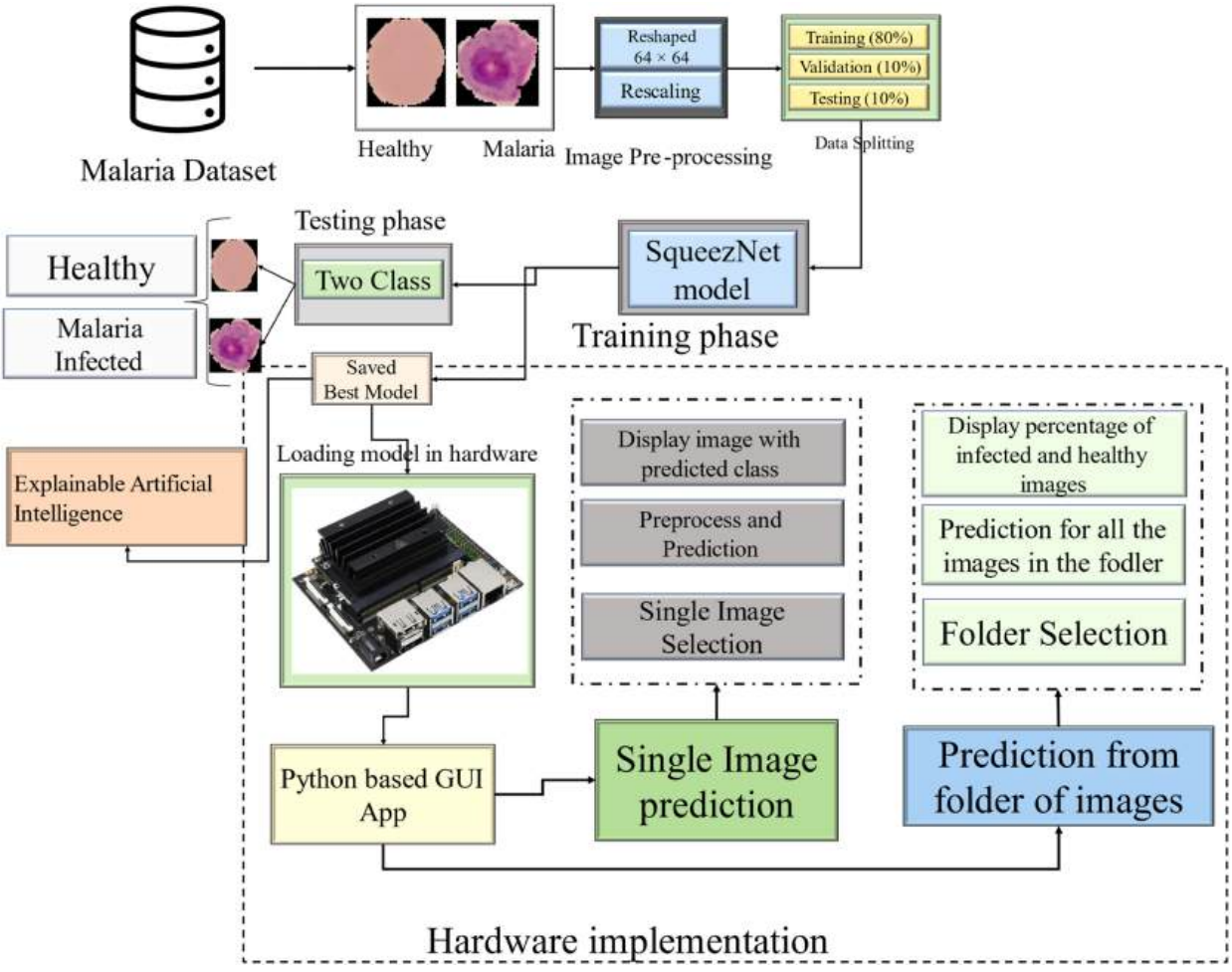


**FIGURE 3** | Overall workflow diagram with embedded system.

**TABLE 1** | Label-wise image count.

| Dataset | Infected images | Uninfected images | Total | Training (80%) | Testing (10%) | Validation (10%) |
|---|---|---|---|---|---|---|
| Original [30] | 13779 | 13779 | 27558 | 22045 | 2755 | 2756 |
| Modified [31] | 13132 | 13029 | 26161 | 20928 | 2616 | 2615 |

present challenges when used as inputs for deep learning models. Several image processing steps are streamlined in this research to optimize these images for embedded systems. Due to their limited storage and computational resources, embedded systems can use less storage and process power. To achieve this, each image is cropped to a standardized 64-pixel width and height during preprocessing. To simplify the classification process for embedded systems, all other preprocessing techniques have been omitted.

#### 2.3.1 | Image Resizing

To ensure uniformity in the input data and to optimize the computational efficiency for embedded systems, all images in the dataset were resized to dimensions of 64×64 pixels. The choice of this size reflects a balance between preserving the critical details necessary for accurate malaria detection and reducing the computational load on the model. By downscaling the images to this fixed resolution, the model can more efficiently process the data, which is particularly important in resource-constrained environments such as embedded systems. This resizing also ensures that the model can generalize better by focusing on the most relevant features while maintaining a consistent input format across the entire dataset.

#### 2.3.2 | Normalization

The images were normalized by scaling the pixel values to a range of [0,1], achieved by dividing each pixel value by 255, which is the maximum value in the RGB color space. Normalization is a crucial preprocessing step as it standardizes the input data, leading to improved convergence during the model's training phase. By bringing all pixel values into a common range, normalization helps in stabilizing the gradient descent process, thereby speeding up the training and enhancing the model's performance. It also reduces the likelihood of the model getting stuck in local minima, resulting in a more robust and accurate prediction.

#### 2.3.3 | Color Space

The images were retained in the RGB color space instead of converting them to grayscale or another color space. This decision was based on the hypothesis that color information could play a significant role in distinguishing between healthy and infected cells in blood smears. Malaria parasites can cause subtle changes in the color properties of infected cells, which might be lost if the images were converted to grayscale. By preserving the original RGB color space, the model is provided with richer information, potentially leading to more accurate detection of malaria-infected cells.

#### 2.3.4 | Batch Preparation

For efficient processing during the training phase, the images were grouped into batches of 64. Batch processing is a widely used technique in deep learning that allows the model to process multiple images simultaneously, which not only speeds up the training process but also ensures that the model's gradients are updated more smoothly.

### 2.4 | Experimental Setup

Implementation of the architecture, as well as all associated preprocessing, training, and evaluation, was carried out using the TensorFlow framework and Keras library. The model was trained on Kaggle and used an NVIDIA Tesla P100 16 GB GPU supported by an Intel Xeon E5-2690 v4 CPU and 13 GB of RAM. After the evaluation phase, the best-trained model was obtained and loaded onto the NVIDIA Jetson Nano Developer Kit-B01. This development kit boasts a Quad-core ARM Cortex-A57 CPU, an NVIDIA Maxwell GPU featuring 128 CUDA cores, 4 GB of 64-bit LPDDR4 RAM, and operates under Ubuntu 18.04 operating system.

### 2.5 | SqueezeNet Model

#### 2.5.1 | Fire Module

A fundamental component of the SqueezeNet framework is the fire module. A key function of this module is to strike a balance between precise feature extraction and minimal computational complexity. As shown in Figure 4, it consists of two layers: the squeeze layer and the expand layer. Squeeze serves as a bottleneck layer, effectively reducing input channels with a 1×1 filter. A minimal parameter footprint allows for the extraction of spatial information using this process. The expanded layer then follows the squeeze layer and incorporates both 1×1 and 3×3 filters. In the 1×1 filter, more channels are added to the input, while in the 3×3 filter, more information is captured through a deeper dive into the data.

#### 2.5.2 | SqueezeNet Architecture

Compared to AlexNet, SqueezeNet has a 50-fold reduction in parameters and is composed of convolution layers, max-pooling layers, fire modules, global average layers, and an output SoftMax layer. A suggested SqueezeNet architecture consists of

12 layers, including a convolution layer, four max-pooling layers, five fire modules, a global average layer, and a single output SoftMax layer. Starting with an input layer with $64 \times 64$ RGB channels, the architecture employs a convolution layer with 32 filters of $3 \times 3$ size to extract features. The Rectified Linear Unit (ReLU) is an activation function in each convolutional layer that enhances the convolutional features. An essential part of the SqueezeNet model is the fire module, which consists of two expansion layers and a squeeze layer. By applying a $1 \times 1$ filter, the squeeze layer reduces the channel depth by $C/4$. As a result, the expand layer employs both $1 \times 1$ and $3 \times 3$ filters to increase the channel depth by $C/2$. ReLU activation occurs in both squeeze and expand layers. Max-pooling reduces feature map size and promotes spatial invariance by de-emphasizing less critical values and concatenating feature values from each fire module.

Each fire layer is followed by four max-pooling layers, each with a pool size of two, resulting in an 8× reduction in the original image. There is a consistent ratio between the output tensor size and the input tensor size of a fire unit. If a tensor with dimensions $H \times W \times C$ is introduced into the fire module, it will go through a squeeze layer and then an expand layer. Squeeze layers employ $1 \times 1$ filters to reduce channel depth by $C/4$, while expand layers employ $1 \times 1$ and $3 \times 3$ filters to increase the depth by $C/2$. The ReLU activation function is incorporated both in the squeeze and in the expand layers. Through concatenation, the desired features extracted from these layers are combined, thus restoring the original depth dimension. In terms of the output layer, the squeeze operation can be expressed as:

$$y^{f1} = \sum_{fm=1}^{F} \sum_{c=1}^{Ch} w_c^f x_c^{fm} \tag{1}$$

where $\{x_c^{fm}\} \in \mathbb{R}^N$ $(fm = 1, 2, 3, \ldots F, c = 1, 2, 3, \ldots, ch)$ is the input for the squeeze operation. $F$ is the feature maps and $ch$ is the channels of length $N$. $y^{f1} \in \mathbb{R}^N (f1 = 1, 2, 3, \ldots F1)$ is the output of the squeeze operation of the kernel $w \in \mathbb{R}^{ch \times 1 \times F1}$.

There are 119 154 trainable parameters out of 120 466 total parameters. This is much less than the other familiar CNN architectures such as ResNet50 has 24 120 962 parameters; VGG16 has 14 848 578 parameters; DenseNet121 has 7 304 514 parameters; MobileNetV2 has 2 591 554 parameters; EfficientNetB3 has 11 183 665 parameters, despite its higher accuracy as shown in Table 6. As a result, training and prediction are less computationally intensive. In this way, embedded systems can meet the basic requirements. The proposed model consists of 1.2 million parameters, 0.042 billion FLOPs, and 1.2 MB of memory (Figure 5). The test time for the proposed model is 10 milliseconds.

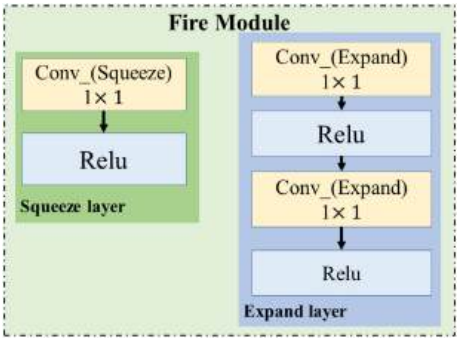


**FIGURE 4** | Fire Module.

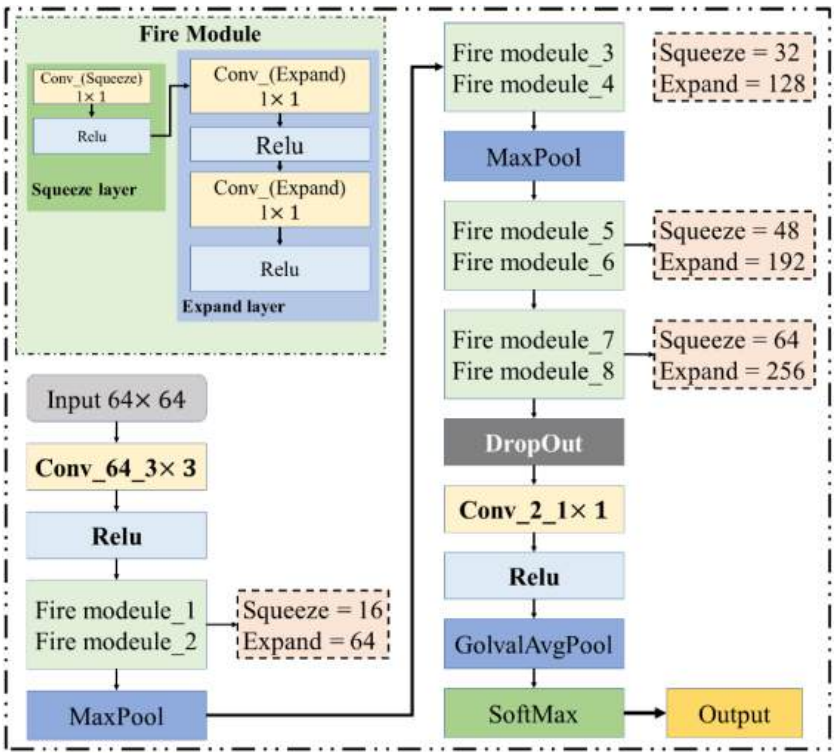


**FIGURE 5** | Model Architecture.

## 3 | Numerical Results

### 3.1 | Evaluation Matrices

Several matrices were used to evaluate the model's performance. Based on the Confusion Matrix (CM), precision, accuracy, recall, and F1-score [32–35], the proposed embedded system model was evaluated. Accuracy, precision, sensitivity, and F1-score are crucial metrics employed to assess the performance of classification models in tasks related to machine learning and data mining. These measures are essential for evaluating the efficiency of classification algorithms, especially in fields where the expenses associated with incorrect positive and negative results differ, such as medical diagnosis, fraud detection, and spam filtering. In terms of accuracy, it represents the level of accuracy with which the model can predict the class. In evaluating a model, this matrix is of importance.

*Accuracy*: Accuracy is a metric that quantifies the ratio of accurate predictions made by the model to the total number of forecasts.

$$\text{Accuracy (ACC)} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})} \tag{2}$$

Here, TP, TN, FN, and FP refer to true positive, true negative, false negative, and false positive respectively.

*Precision*: Precision is a measure that calculates the ratio of correct positive predictions to all positive predictions generated by the model. The rate of incorrectly classifying patients in medical diagnosis can lead to unnecessary treatments. It is important to evaluate a model precisely in this regard. Furthermore, if a dataset is imbalanced, the precision of a class can be reduced even if the accuracy of a particular class can be high.

$$\text{Precision (Pre)} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \tag{3}$$

*Recall*: Sensitivity, or recall, quantifies the ratio of accurately detected actual positive events by the model. In the case of malaria-infected patients, a false negative can have serious health implications.

$$\text{Recall (Spe)} = \frac{\text{TP}}{(\text{TP} + \text{FN})} \tag{4}$$

*F1-Score*: The F1-score is a mathematical average that combines precision and recall, offering a well-balanced measure of both parameters. In F1-score, recall and precision are combined into a single value. F1-score evaluation is essential when precision and recall are both crucial to a model's performance.

$$\text{F1} - \text{score (F1S)} = \frac{2 * (\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})} \tag{5}$$

As a result, the proposed model demonstrates outstanding performance across all performance metrics, including training, testing, and validation accuracies. The model's overall accuracy is 99.76%, marking a significant advancement in the medical field. In addition, training accuracy is 99.57%, and validation accuracy is 99.175%, demonstrating the model's robustness without overfitting.

### 3.2 | Batchwise Performance Analysis

A study was conducted to examine how different batch sizes impacted the model's performance, focusing on different performance metrics. Tables 2 and 3 provide a detailed overview of the results of this investigation.

#### 3.2.1 | Adam Optimizer for Original Dataset

A high degree of overall correctness is evidenced by accuracy values ranging from 95.82% to 96.37%. The precision values span

**TABLE 2** | Performance for different batch sizes with ADAM optimizer and original dataset.

| Batch Size | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) | Model Size (MB) |
|---|---|---|---|---|---|
| 8 | 95.82 | 95.62 | 96.14 | 95.88 | 1.72 |
| **16** | **96.37** | **95.67** | **97.21** | **96.44** | **1.72** |
| 32 | 96.05 | 95.00 | 97.30 | 96.14 | 1.72 |
| 64 | 96.28 | 95.50 | 97.21 | 96.35 | 1.72 |

*Note:* The bold values represent the highest performance achieved in the present work.

**TABLE 3** | Performance for different batch sizes with ADAM optimizer and modified dataset.

| Batch Size | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) | Model Size (MB) |
|---|---|---|---|---|---|
| 8 | 99.61 | 99.33 | 99.90 | 99.62 | 1.72 |
| 16 | 99.62 | 99.34 | 99.90 | 99.62 | 1.72 |
| 32 | 99.66 | 99.62 | 99.71 | 99.66 | 1.72 |
| **64** | **99.71** | **99.71** | **99.71** | **99.71** | **1.72** |

*Note:* The bold values represent the highest performance achieved in the present work.

from 95% to 95.67%, indicating a notably low false positive rate while the batch size varied from 8 to 64 (Table 2). The recall value of the model ranges from 96.14% to 97.30%, indicating its effectiveness in minimizing false negatives. There is a good balance between precision and recall in the F1 scores, which range from 95.88% to 96.44%. The model with a batch size of 16 shows promising results in malaria detection compared to other batch sizes.

#### 3.2.2 | Adam Optimizer for Modified Dataset

A high level of accuracy was consistently demonstrated by the model across all batch sizes, ranging from 99.61% to 99.71% (Table 3). Based on these results, the model can effectively classify most of the samples submitted. Additionally, the precision values, ranging from 99.33% to 99.71%, were consistently high. The model's exceptional accuracy in predicting positive samples is underscored by this consistency. Additionally, the recall values remained consistently above 99.6%, illustrating the model's remarkable ability to identify positive samples.

The F1-score, which combines precision and recall, consistently exceeded 99.6% across all batch sizes. Thus, the model ensures both high accuracy and effective detection of positive samples by maintaining an equilibrium between precision and recall. Across all batch sizes, the model size remained constant at 1.72 MB. As a result, the batch size had a minimal impact on the model's size, allowing efficient use of computational resources. Based on these results, it is clear that the model consistently performs exceptionally well regardless of batch size. The information provided here will assist scientists and professionals in selecting the most appropriate batch size based on their computational capabilities and training needs.

### 3.3 | Evaluation of Original Dataset Versus Modified Dataset

Most of the recent research has been conducted using the original dataset. The evaluation matrices of the proposed model using the original dataset can be found in Table 4. The presence of mislabeled data in the original dataset inevitably adversely affected model performance. In most recent studies of malaria, the original dataset was used, which could have contained inaccuracies. Unlike the original dataset, the modified version was carefully curated to eliminate mislabeled images and therefore produced significantly higher accuracy. A 99.71% accuracy, precision, recall, and F1-score were obtained when using the modified dataset. In comparison, when working with the original dataset, these metrics were at 96.28% accuracy, 95.50% precision, 97.21% recall, and 96.35% F1-score. Figure 6a,b illustrates the effectiveness of the model with the ADAM optimizer and a batch size of 64. The confusion matrix shown in Figure 6c illustrates the number of predicted infected and uninfected samples in the test split of the dataset.

### 3.4 | Comparison of Performance With State-Of-The-Art Models

Table 5 shows the performance comparison of the proposed work with the state-of-the-art methods reported in the literature in malaria detection from blood smear images. Based on the results, the proposed model outperformed all the other models reported in the literature and achieved a maximum accuracy and F1-score of 99.71%. We conducted experiments on various architectures, including VGG16, DenseNet121, MobileNetV2, MvitV2_small, Xception and the proposed SqueezeNet model, to identify the best-performing model. We used a modified dataset consisting of 64×64 resized images, a batch size of 64, and the ADAM optimizer for these experiments. Several critical factors were weighed before selecting the proposed model, including the number of parameters and the size of the model. Based on the comparative results presented in Table 6, the proposed SqueezeNet architecture outperforms the others. It is noteworthy that the proposed model has fewer parameters, while being more efficient than VGG16, DenseNet121, MobileNet_V2, and MViTv2_small, SqueezeNet_0, Xception respectively. Additionally, the proposed CNN model has a footprint of only 1.72 MB, which is much smaller than that of other CNN models.

Table 6 shows how different deep learning models perform for malaria detection. Five performance measures are considered in this work: Accuracy (Acc), Precision (Pre), Sensitivity (Recall) (Sen), F1-score (F1S), and Specificity (Spe). Classification tasks typically use these performance measures to evaluate model performance. All evaluation measures show that the proposed model scored 99.71%, 99.71%, 99.71%, 99.71%, and 99.71%, respectively, for Accuracy, Precision, Sensitivity, F1-score, and Specificity. It is evident from these findings that the proposed model can accurately identify instances, maintain a high true positive rate (Sensitivity/Recall), limit false positives (Precision), and strike a good balance between Precision and Recall (high F1-score). The model's ability to recognize negative events is demonstrated by its high Specificity. VGG16 is the second-best model, with somewhat lower scores than the suggested model. In comparison to our proposed model and the VGG16, Xception, MobileNet_V2, DenseNet121, Multiscale Vision Transformer (MViTv2_small), and SqueezeNet1_0 perform differently. The proposed model only requires 0.042 billion FLOPs, significantly fewer than most other pre-trained models. Due to its low computational complexity, the proposed model has a faster inference time than other pre-trained models. The proposed model sacrifices some accuracy and representational power in exchange

**TABLE 4** | Evaluation metrics between uses of original and modified dataset.

| Dataset | Accuracy (%) | Precision (%) | Recall (%) | F1-score (%) |
|---|---|---|---|---|
| Original | 96.28 | 95.50 | 97.21 | 96.35 |
| **Modified** | **99.71** | **99.71** | **99.71** | **99.71** |

*Note:* The bold values represent the highest performance achieved in the present work.

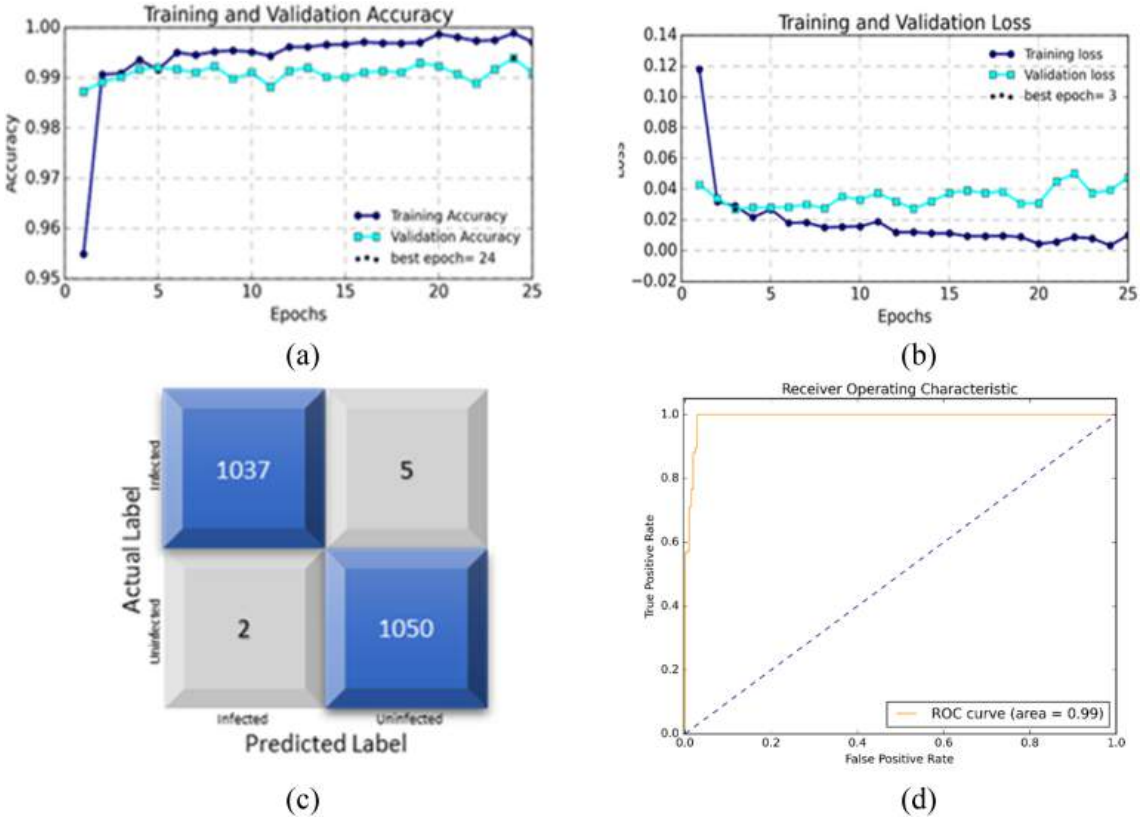


**FIGURE 6** | Training and validation accuracy curve (a), loss curve (b), Confusion Matrix (c), and (d) ROC-AUC curve.

**TABLE 5** | Comparative Analysis of previous models with the proposed model.

| Authors | Model | Accuracy (%) | F1-score (%) |
|---|---|---|---|
| Liang et al. [16] | 16-layer CNN | 97.37 | 97.36 |
| Alqudah et al. [13] | 23-layer CNN | 98.85 | 98.85 |
| Bibin et al. [22] | Deep Belief Network | 96.30 | — |
| Rajaraman et al. [30] | Ensemble CNN | 99.50 | 99.08 |
| Yuhang et al. [36] | CNN pre-trained model | 98.13 | — |
| Kumar et al. [37] | CapsNet model | 99.08 | 97.44 |
| Devi et al. [27] | ANN | 96.32 | 85.32 |
| **Proposed model** | **SqueezeNet** | **99.71** | **99.71** |

*Note:* The bold values represent the highest performance achieved in the present work.

for efficiency and a smaller memory footprint when compared to larger and more complex models like VGG16 (138.4 billion FLOPs, 528 MB size).

However, the proposed model is on par with or even outperforms the highly efficient MobileNet_V2 (0.22 billion FLOPs, 14 MB size) in terms of model size and computational complexity, so it is a compelling choice for applications that prioritize speed and resource utilization over maximizing accuracy. Although DenseNet-121 (8 billion FLOPs, 33 MB size) has a larger model size and higher computational complexity than our proposed model, it still maintains a relatively compact footprint in comparison to models like VGG16 and DenseNet121. In the end, the choice of model depends on your application's specific requirements, such as the desired balance between accuracy, speed, and resource constraints. With its exceptional efficiency, our model is a strong contender for scenarios where computational resources are limited, while larger models such as VGG16 and DenseNet121 may be more suitable for applications requiring accuracy over efficiency. Based on the investigation results, the proposed model captured key traits and patterns better than other models. The performance of a model should be evaluated concerning its computational complexity, resource requirements, and interpretability, depending on the application and limitations. To confirm performance differences and ensure results robustness, statistical significance testing, and cross-validation may be required. A clear and useful comparison of model performance is provided in the table (Table 6), demonstrating the proposed model's potential for further development and practical application.

### 3.5 | Model Visualization

The study of Explainable AI (XAI) techniques is considered a crucial area of study in the field of machine learning, especially

**TABLE 6** | Performance comparison (in %) of the proposed model with different pre-trained model.

| Model | Acc | Pre | Sen | F1S | Spe | param | mSize (MB) | Training time (s) | Testing time (s) |
|---|---|---|---|---|---|---|---|---|---|
| Xception | 99.12 | 99.12 | 99.12 | 99.12 | 99.11 | 22.9 M | 90.2 | 24 min | 0.20/img |
| MobileNet_V2 | 99.12 | 99.12 | 99.12 | 99.12 | 99.11 | 3.5 M | 14.0 | 39 min | 0.40/img |
| DenseNet121 | 99.07 | 99.07 | 99.07 | 99.07 | 99.07 | 8.0 M | 33.0 | 48 min | 0.31/img |
| MViTv2_small | 99.02 | 99.02 | 99.02 | 99.02 | 99.02 | 19.5 M | 77.0 | 27 min | 0.23/img |
| VGG16 | 99.31 | 99.31 | 99.31 | 99.31 | 99.3 | 138.4 M | 528.0 | 45 min | 0.19/img |
| SqueezeNet1_0 | 99.16 | 99.26 | 99.19 | 99.12 | 99.11 | 1.2 M | 4.8 | 24 min | 0.21/img |
| **Proposed model** | **99.71** | **99.71** | **99.71** | **99.71** | **99.71** | **1.2 M** | **1.72** | **20 min** | **0.09/img** |

*Note:* The bold values represent the highest performance achieved in the present work.
Abbreviations: img—image; Acc—accuracy; Pre—precision; Sen—sensitivity; F1S—F1-score; Spe—specificity; param—parameter count; mSize—model size.

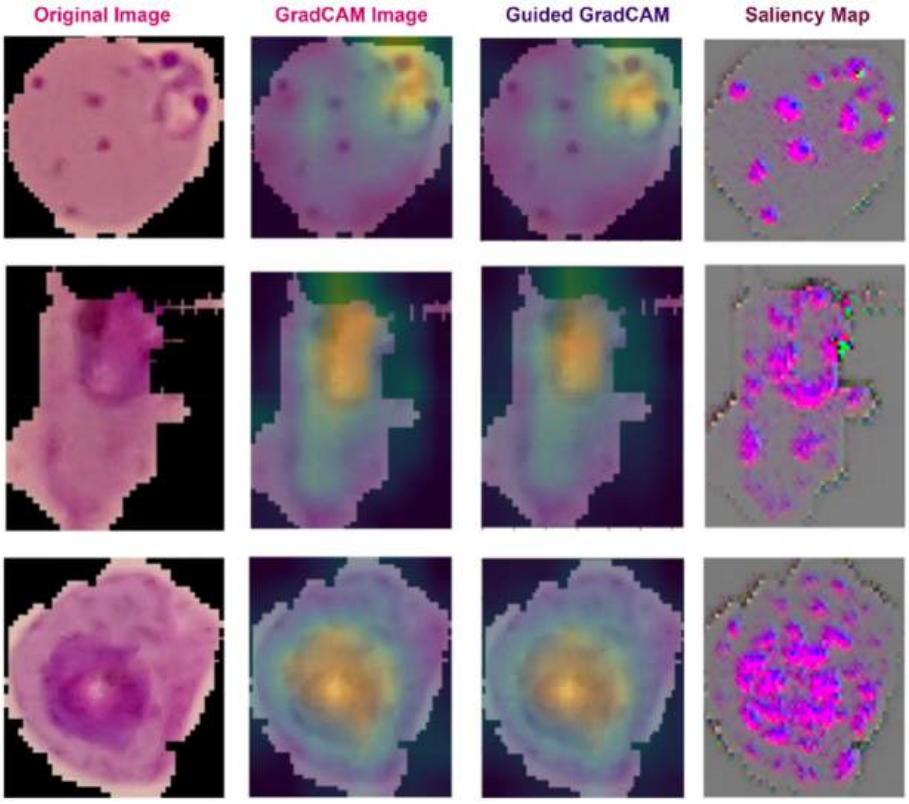


**FIGURE 7** | Original image and their corresponding Grad-Cam, Guided-Grad-Cam, and Saliency Mapping result for malaria-affected cells.

for applications requiring transparency and interpretability. Researchers and practitioners can use XAI to understand decision-making processes, detect biases or errors, and build trust and accountability through visual representations and explanations of intricate model behavior. By using methods like Grad-Cam [33], Grad-Cam++ [34], activation visualization [38, 39] saliency maps [40, 41], and architectural visualization [42], XAI provides insight into the internal processes of deep neural networks and other complex models, which facilitates model debugging, regulatory compliance, and collaboration between humans and machines. Finally, XAI and model visualization are important because they enable AI systems to unleash their full potential, promoting trust, accountability, and responsible implementation.

Deep learning methods such as saliency maps, Grad-CAM (Gradient-weighted Class Activation Mapping), and Guided Grad-CAM are widely used to visualize and interpret models, specifically convolutional neural networks (CNNs). Figure 7 illustrates the region of the image that affects more in the proposed model's prediction by using these methods. Their purpose is to highlight the areas or characteristics of the input image that are most significant for the prediction of the model. From the figure, the model is predicted based on red pixels. The illustration of healthy cells in

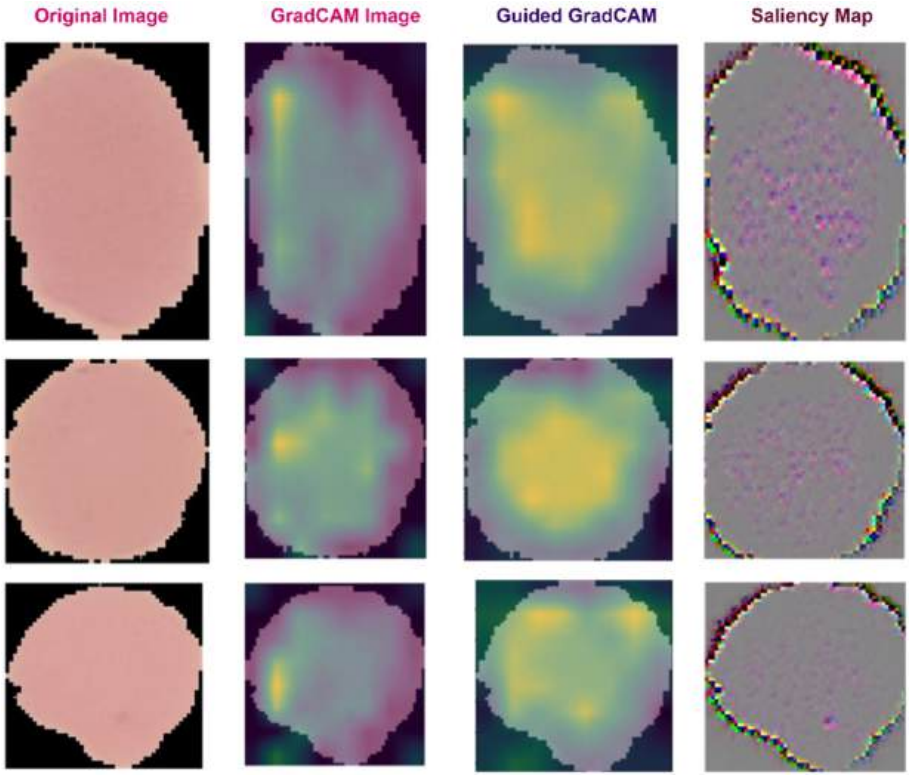


**FIGURE 8** | Original image and their corresponding Grad-Cam, Guided-Grad-Cam, and Saliency Mapping result for healthy cells.

Figure 8 makes this clearer. In the absence of reddish pixels, the reddish pixels model predicts them as healthy images. As a fundamental method, saliency maps utilize pixels or features to visually represent the significance of each input pixel or feature based on a specific output. It is necessary to determine the gradient of the output in relation to the input to perform the computation. The saliency map prioritizes the input pixels or features that have the greatest impact on the output. Saliency maps may be difficult to interpret for sophisticated models due to their chaotic nature.

Grad-CAM is a convolutional layer-specific extension of saliency maps that is implemented in CNNs. Gradients of the target output are calculated using the feature maps of the last convolutional layer. These gradients are then used to compute importance weights for each feature map; these weights are then combined to produce a coarse localization map. By using the map, the critical areas within the input image that contribute the most to the desired result are highlighted.

#### 3.5.1 | Guided Grad-CAM

Guided Grad-CAM combines guided backpropagation's high-resolution details with Grad-CAM's localization information. Guided backpropagation applies positive gradients to the backpropagation process to emphasize the areas of the input image that are positively impacted by the target image. Combining the Grad-CAM localization map and the guided backpropagation saliency map is achieved through element-wise multiplication. By integrating the high-resolution details obtained from guided backpropagation with the coarse localization information obtained from Grad-CAM, Guided Grad-CAM offers a more detailed visualization than Grad-CAM.

## 4 | Hardware Implementation and Embedded System

The hardware setup includes the Jetson Nano B01, a single-board computer designed for edge computing applications. This compact device is equipped with a quad-core ARM Cortex-A57 processor, a powerful 128-core NVIDIA Maxwell GPU, and 4 GB of LPDDR4 RAM. The Jetson Nano B01 provides the necessary computational capabilities for deep learning models. The Cortex-A57 cores in the Jetson nano have a clock speed of 1.43Ghz which can be increased to 2 GHz by overclocking. With four cores and four threads, the CPU can compute floating-point math at 2471 MOps/Sec. Figure 9 shows the graphical user interface (GUI) application developed using the Python programming language, to deliver a user-friendly interface for interacting with the malaria prediction system designed for microscopic images.

Here are the steps involved in the operation of the embedded system:

- *Training*: A lightweight SqueezeNet model is trained for malaria detection.
- *Deployment*: The trained model is deployed on an EDGE device known as Jetson Nano.
- *GUI Application:* A specialized Python GUI application is developed, utilizing libraries such as PYQT5, CV2, Tensorflow, Keras, Numpy, Matplotlib [43], and os.
- *Model Loading:* The GUI application loads the pre-trained model into memory using the Keras Model library.
- *User Choice:* Users are presented with two forecasting options.

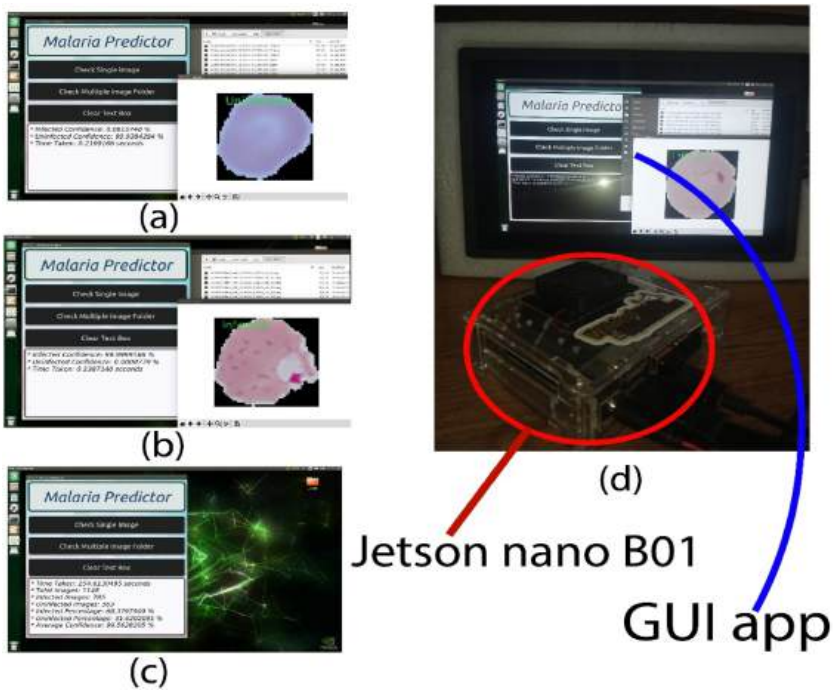


**FIGURE 9** | Proposed hardware implementation with Jetson Nano and GUI app support.

- *Option 1:* Manual selection of a picture for malaria infection assessment.
  - Image Processing: The chosen image is read and resized.
  - Prediction: The preprocessed image is used for model prediction.
  - GUI Output: The results, including class indices, are displayed on the GUI.
- *Option 2:* Selection of a folder containing unclassified cell photos.
  - Image Cycling: The GUI application lists and cycles through each image in the selected folder.
  - Processing and Prediction: Each image is processed and predicted using the trained model.
  - Infection Statistics: The application keeps a record of infected and uninfected cell photos.
  - Statistical Information: After analyzing all the photos, the application provides statistical insights, including the total image count, the tally of infected and uninfected images, and their respective percentages.
  - Integration: This step combines the training of the SqueezeNet model, deployment on Jetson Nano, and the development of a user-friendly GUI application.
  - Malaria Detection: Users are equipped to accurately predict the presence of malaria infection in individual photos or batches of unclassified cell images.

The application encompasses two primary functionalities:

a. *Single Image Prediction:* This feature empowers users, especially professionals, to select an image from their local storage or capture an image using an integrated camera. The chosen image undergoes processing by a pre-trained deep-learning model on the Jetson Nano B01. Based on the model‘s training classes or labels, the image is analyzed, and a prediction is made. The GUI then presents the predicted class along with a confidence score, offering real-time insights into individual photos.

b. *Batch Prediction:* Batch prediction comes into play when multiple images need to be processed simultaneously. Users can designate a directory containing the images to be analyzed. The GUI application processes photos from the specified directory, running them through the deep learning model in parallel. Users can evaluate the results of the entire batch by reviewing the projected classes and confidence scores for each image. These scores are presented in tabular format.

### 4.1 | Embedded system's Performance Analysis

The work leveraged the capabilities of the Nvidia Jetson Nano to deploy a lightweight malaria diagnostic model. As a result of this model's excellent performance in real-time, malaria infections were detected swiftly and accurately. The average prediction time for each image was only 0.24 s, indicating its capability to accurately categorize images as healthy or malaria-infected. Additionally, the model consistently delivered 99.71% confidence scores, indicating its dependability in accurately and quickly classifying images. As demonstrated by the model's outstanding real-time performance on Nvidia Jetson Nano, the model delivers highly efficient image processing and prompt results. Its high confidence level of 99.71% inspires trust among healthcare professionals and stakeholders, making it suitable for deployment in resource-constrained environments without access to experienced medical experts. Combining these attributes demonstrates the lightweight model's remarkable ability to provide accurate and rapid predictions. A real-time detection model for malaria is

implemented on the Nvidia Jetson Nano platform, which facilitates early diagnosis and treatment initiation. Based on these results, lightweight models may be useful for disease detection applications on edge devices. Point-of-care diagnostics are set to advance further thanks to the Nvidia Jetson Nano's processing capabilities. It has been demonstrated that the lightweight model can be implemented in edge computing environments, providing accurate and swift malaria diagnosis.

### 4.2 | Cost Analysis

The model's development and deployment were driven by the need for cost-effectiveness. During the development phase, we utilized free cloud-based platforms like Kaggle and Google Colab, which provided access to high-performance GPUs (A100 and T4). This allowed us to train and optimize our model without significant financial investment.

For deployment, we focused on using the NVIDIA Jetson Nano B01, a budget-friendly edge computing device priced at $99. The Jetson Nano was chosen for its affordability and suitability for resource-constrained environments. The successful deployment of our lightweight model on this device demonstrates that effective malaria detection can be achieved without expensive hardware. This makes the solution both accessible and practical for real-world applications, particularly in low-resource settings.

## 5 | Discussion and Future Work

This study sheds light on the vast possibilities that can arise from the use of lightweight models on edge devices for disease detection applications. Taking advantage of the generous processing capabilities provided by the Nvidia Jetson Nano, we not only achieved remarkable performance but also ensured a high degree of precision. A significant step has been taken towards the future of point-of-care diagnostics with these achievements.

As a result of our research, a real-time malaria detection model can be deployed on the Nvidia Jetson Nano with notable success. An average prediction time of 0.24 s per image was achieved, and an average confidence level of 99.71% was consistently achieved. A combination of these results confirms that the model can provide rapid and accurate malaria diagnoses, making it an invaluable tool in the detection of infectious diseases. Despite the impressive achievements of this study, it is important to acknowledge some limitations and opportunities for improvement. Despite a limited storage capacity, edge devices can often handle large volumes of images at once, and our research recommends caution. As a result of our study, predicting 1000 images took approximately 4 min, indicating that practical considerations need to be taken when working with edge devices.

In addition, the lightweight nature of the model provides an opportunity for enhancement. Training the model exclusively on malaria-related data allows it to gain a deeper understanding of the nuanced traits and patterns associated with malaria infection. As a result of this specialized approach, the model's detection accuracy can be significantly improved, making it even more efficient for deployment on edge devices. Expanding the dataset with a focus on optimizing storage in edge devices will enhance the model's generalizability and suitability for edge computing.

Based on the findings of this study, lightweight models can be deployed on edge devices to detect diseases. In the future, point-of-care diagnostics are likely to benefit from Nvidia Jetson Nano's exceptional performance and precision. In medical diagnostics, edge computing has demonstrated significant potential, but it will require further refinement and adaptation to reach its full potential. This research paves the way for the detection of diseases early and efficiently using lightweight models on edge devices in the future. There are limitations and a need for further development of the proposed approach, but it has shown promise. The computing power of edge devices limits their ability to handle large image volumes. Although the model could predict individual images very quickly, it may be slow to process many images at once in real-time.

In addition, using a handpicked dataset as a training dataset has several drawbacks. A variety of malaria-infected and healthy cell images could improve the model's generalization and robustness to imaging circumstances. A future study should investigate parallelization or memory management strategies to improve the model's performance on edge devices. To improve the malaria detection system's flexibility, continuous learning or transfer learning could be used to utilize the model's knowledge as it relates to new data distributions or imaging modalities. In this study, we proposed a comprehensive framework for the Squeeze net model and conducted extensive validation using retrospective data. Although the framework has been rigorously tested and validated in a controlled environment, clinical trials were not within the scope of this work. To bridge the gap between research and clinical application, we have developed a user-friendly software tool designed for healthcare professionals. This tool enables doctors to test the model on their datasets, thereby facilitating its practical evaluation in a clinical setting. Clinical trials are a critical next step, and we intend to pursue this as part of our future work.

## 6 | Conclusion

This study showed a new method for detecting malaria in blood smear images by using a lightweight SqueezeNet model designed for embedded computers. The proposed solution included dataset preprocessing, a SqueezeNet architecture implementation with a unique fire module, performance evaluation using several metrics, and hardware deployment on the NVIDIA Jetson Nano edge device. We carefully designed the SqueezeNet model to balance computational efficiency and malaria detection accuracy. A key design component, the fire module, extracted discriminative characteristics while reducing parameter footprints. A lightweight model with 119 154 trainable parameters and a size of 1.72 MB was proposed for resource-constrained embedded systems. In addition to testing the model on the original NIH malaria dataset, the model was also tested on a modified dataset addressing label inconsistencies. Based on the adjusted dataset, the model had 99.71% accuracy, precision, recall, and F1-score. In real-time, the model predicted single images in 0.24 s with a 99.71% confidence level. With minimal computational resources, the

model can accurately and efficiently diagnose malaria in remote or resource-constrained areas. The proposed approach outperformed top deep learning models such as ResNet50, VGG16, DenseNet121, MobileNetV2, and EfficientNetB3. Compared to these architectures, the proposed SqueezeNet was more accurate and had fewer parameters, while its model size was more efficient, making it an effective embedded device architecture. The use of explainable AI (XAI) methods like Grad-CAM and Guided Grad-CAM improved model interpretation and trust. These visualization techniques revealed the model's decision-making process by emphasizing areas of the input image that predicted malaria. AI-driven diagnostic solutions require this trait of interpretability to be trusted by healthcare practitioners.

### Author Contributions

**A.S.:** methodology, software, writing the original draft. **S.M.N.H.:** validation, formal analysis. **M.J.K.:** software, investigation, data curation. **S.A.:** investigation, writing the original draft. **M.N.:** conceptualization, supervision, formal analysis. **M.E.H.C.:** supervision, writing – review and editing, formal analysis, software. **M.M.:** methodology, visualization, investigation, writing – review and editing.

### Ethics Statement

The authors have nothing to report.

### Consent

The authors have nothing to report.

### Conflicts of Interest

The authors declare no conflicts of interest.

### Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

### References


1. A. F. Cowman, J. Healer, D. Marapana, and K. Marsh, "Malaria: Biology and Disease," *Cell* 167 (2016): 610–624.

2. S. I. Hay, C. A. Guerra, A. J. Tatem, A. M. Noor, and R. W. Snow, "The Global Distribution and Population at Risk of Malaria: Past, Present, and Future," *Lancet Infectious Diseases* 4 (2004): 327–336, https://doi.org/10.1016/S1473-3099(04)01043-6.

3. A. V. S. Hill, "Vaccines Against Malaria," *Philosophical Transactions of the Royal Society B: Biological Sciences* 366 (2011): 2806–2814, https://doi.org/10.1098/rstb.2011.0091.

4. M. Roser and H. Ritchie, "Our World in Data, Malaria," accessed April 2024, https://ourworldindata.org/malaria.

5. N. J. White, "Determinants of Relapse Periodicity in Plasmodium Vivax Malaria," *Malaria Journal* 10 (2011): 1–36.

6. "Clincial Guidance: Malaria Diagnosis & Treatment in the U.S," accessed June 5, 2024, https://www.cdc.gov/malaria/hcp/clinical-guidance/index.html.

7. C. Naing and M. A. Whittaker, "Severe Thrombocytopaenia in Patients With Vivax Malaria Compared to Falciparum Malaria: A Systematic Review and Meta-Analysis," *Infectious Diseases of Poverty* 7 (2018): 10, https://doi.org/10.1186/s40249-018-0392-9.

8. M. O. F. Goni, M. N. I. Mondal, S. M. R. Islam, et al., "Diagnosis of Malaria Using Double Hidden Layer Extreme Learning Machine Algorithm With CNN Feature Extraction and Parasite Inflator," *IEEE Access* 11 (2023): 4117–4130.

9. "Institute of Health Metrics and Evaluation," 2024, https://www.healthdata.org/research-analysis/health-risks-issues/child-health.

10. S. Asif, S. U. R. Khan, X. Zheng, and M. Zhao, "MozzieNet: A Deep Learning Approach to Efficiently Detect Malaria Parasites in Blood Smear Images," *International Journal of Imaging Systems and Technology* 34 (2024): e22953, https://doi.org/10.1002/ima.22953.

11. S. Asif, M. Zhao, F. Tang, and Y. Zhu, "A Deep Learning-Based Framework for Detecting COVID-19 Patients Using Chest X-Rays," *Multimedia Systems* 28 (2022): 1495–1513, https://doi.org/10.1007/s00530-022-00917-7.

12. S. Dey, P. Nath, S. Biswas, S. Nath, and A. Ganguly, "Malaria Detection Through Digital Microscopic Imaging Using Deep Greedy Network With Transfer Learning," *Journal of Medical Imaging* 8 (2021): 054502.

13. A. Alqudah, A. M. Alqudah, and S. Qazan, "Lightweight Deep Learning for Malaria Parasite Detection Using Cell-Image of Blood Smear Images," *Rev. d'Intelligence Artif* 34 (2020): 571–576.

14. T. Banerjee, A. Jain, S. C. Sethuraman, S. C. Satapathy, S. Karthikeyan, and A. Jubilson, "Deep Convolutional Neural Network (Falcon) and Transfer Learningbased Approach to Detect Malarial Parasite," *Multimedia Tools and Applications* 81 (2021): 13237–13251.

15. M. Umer, S. Sadiq, M. Ahmad, S. Ullah, G. S. Choi, and A. Mehmood, "A Novel Stacked CNN for Malarial Parasite Detection in Thin Blood Smear Images," *IEEE Access* 8 (2020): 93782–93792.

16. Z. Liang, A. Powell, I. Ersoy, et al., "CNN-Based Image Analysis for Malaria Diagnosis," in *2016 IEEE international conference on bioinformatics and biomedicine (BIBM)* (2016), 493–496.

17. G. P. Gopakumar, M. Swetha, G. Sai Siva, and G. R. K. S. Subrahmanyam, "Convolutional Neural Network-Based Malaria Diagnosis From Focus Stack of Blood Smear Images Acquired Using Custom-Built Slide Scanner," *Journal of Biophotonics* 11 (2017): e201700003.

18. A. Alassaf and M. Yacin Sikkandar, "Intelligent Deep Transfer Learning Based Malaria Parasite Detection and Classification Model Using Biomedical Image," *Computers, Materials & Continua* 72 (2022): 5273–5285.

19. A. Vijayalakshmi and B. Rajesh Kanna, "Deep Learning Approach to Detect Malaria From Microscopic Images," *Multimedia Tools and Applications* 79 (2019): 15297–15317.

20. G. Shekar, S. Revathy, and E. K. Goud, "Malaria Detection Using Deep Learning," in *2020 4th Int. Conf. Trends Electron. Informatics* (Tirunelveli, India: IEEE, 2020).

21. A. H. Alharbi, C. V. Aravinda, M. Lin, B. Ashwini, M. Y. Jabarulla, and M. A. Shah, "Detection of Peripheral Malarial Parasites in Blood Smears Using Deep Learning Models," *Computational Intelligence and Neuroscience* 2022 (2022): 3922763.

22. D. Bibin, M. S. Nair, and P. Punitha, "Malaria Parasite Detection From Peripheral Blood Smear Images Using Deep Belief Networks," *IEEE Access* 5 (2017): 9099–9108.

23. I. Amin, S. Hassan, S. Brahim Belhaouari, and M. Hamza Azam, "Transfer Learning-Based Semi-Supervised Generative Adversarial Network for Malaria Classification," *Computers, Materials & Continua* 74 (2023): 6335–6349.

24. M. R. Islam and M. Nahiduzzaman, "Complex Features Extraction With Deep Learning Model for the Detection of COVID19 From CT Scan Images Using Ensemble Based Machine Learning Approach," *Expert Systems with Applications* 195 (2022): 116554.

25. P. Krishnadas, K. Chadaga, N. Sampathila, S. Rao, and S. Prabhu, "Classification of Malaria Using Object Detection Models," *Inform* 9 (2022): 76.

26. S. Shuleenda Devi, S. A. Sheikh, A. Talukdar, and R. H. Laskar, "Malaria Infected Erythrocyte Classification Based on the Histogram Features Using Microscopic Images of Thin Blood Smear, Indian," *Journal of Science and Technology* 9 (2016): 1–10.

27. S. S. Devi, R. H. Laskar, and S. A. Sheikh, "Hybrid Classifier Based Life Cycle Stages Analysis for Malaria-Infected Erythrocyte Using Thin Blood Smear Images," *Neural Computing and Applications* 29 (2017): 217–235.

28. M. O. Arowolo, M. O. Adebiyi, A. A. Adebiyi, and O. Olugbara, "Optimized Hybrid Investigative Based Dimensionality Reduction Methods for Malaria Vector Using KNN Classifier," *Journal of Big Data* 8 (2021): 29.

29. Y. M. Kassim, K. Palaniappan, F. Yang, et al., "Clustering-Based Dual Deep Learning Architecture for Detecting Red Blood Cells in Malaria Diagnostic Smears," *IEEE Journal of Biomedical and Health Informatics* 25 (2021): 1735–1746, https://doi.org/10.1109/JBHI.2020.3034863.

30. S. Rajaraman, S. K. Antani, M. Poostchi, et al., "Pre-Trained Convolutional Neural Networks as Feature Extractors Toward Improved Malaria Parasite Detection in Thin Blood Smear Images," *PeerJ* 6 (2018): e4568.

31. K. M. F. Fuhad, J. F. Tuba, M. R. A. Sarker, S. Momen, N. Mohammed, and T. Rahman, "Deep Learning Based Automatic Malaria Parasite Detection From Blood Smear and Its Smartphone Based Application," *Diagnostics* 10 (2020): 329.

32. K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," 2014, arXiv Preprint arXiv:1409.1556.

33. R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization," *Proceedings of the IEEE international conference on computer vision* (2017), 618–626, https://doi.org/10.1109/ICCV.2017.74.

34. A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks," in *Proc.—2018 IEEE Winter Conf. Appl. Comput. Vision, WACV 2018* (Lake Tahoe, USA: IEEE, 2018), 839–847, https://doi.org/10.1109/WACV.2018.00097.

35. K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps," *2nd International Conference on Learning Representations*, April 14-16 (Banff, AB, Canada: ICLR, 2014).

36. Y. Dong, Z. Jiang, H. Shen, et al., "Evaluations of Deep Convolutional Neural Networks for Automatic Identification of Malaria Infected Cells," in *2017 IEEE EMBS international conference on biomedical & health informatics (BHI)* (2017): 101–104.

37. S. A. Kumar, M. K. Muchahari, S. Poonkuntran, L. S. Kumar, R. K. Dhanaraj, and P. Karthikeyan, "Application of Hybrid Capsule Network Model for Malaria Parasite Detection on Microscopic Blood Smear Images," *Multimedia Tools and Applications* (2024): 1–27, https://doi.org/10.1007/s11042-024-19062-6.

38. M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why Should I Trust You?': Explaining the Predictions of any Classifier," *NAACL-HLT 2016–2016 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. Proc. Demonstr. Sess* (2016): 97–101, https://doi.org/10.18653/v1/n16-3020.

39. A. Salam, M. Naznine, N. Jahan, E. Nahid, M. Nahiduzzaman, and M. E. H. Chowdhury, "Mulberry Leaf Disease Detection Using CNN-Based Smart Android Application," *IEEE Access* 12 (2024): 83588, https://doi.org/10.1109/ACCESS.2024.3407153.

40. J. Adebayo, J. Gilmer, M. Muelly, et al., "Sanity Checks for Saliency Maps," *Advances in Neural Information Processing Systems* 31 (2018): 1–11, https://goo.gl/hBmhDt.

41. M. D. Zeiler, and R. Fergus, *Visualizing and Understanding Convolutional Networks*, Vol. 8689, eds. D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars (Zurich, Switzerland: Springer, 2014), https://doi.org/10.1007/978-3-319-10590-1_53.

42. M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu, "Towards Better Analysis of Deep Convolutional Neural Networks," *IEEE Transactions on Visualization and Computer Graphics* 23 (2017): 91–100, https://doi.org/10.1109/TVCG.2016.2598831.

43. J. D. Hunter, "Matplotlib: A 2D Graphics Environment," *Computing in Science & Engineering* 9 (2007): 90–95, https://doi.org/10.1109/MCSE.2007.55.