Konstruksi: Publikasi Ilmu Teknik, Perencanaan Tata Ruang dan Teknik Sipil
Volume. 3 Nomor. 4 Oktober 2025
E-ISSN .: 3031-4089; P-ISSN .: 3031-5069, Hal. 96-115
DOI: https://doi.org/10.61132/konstruksi.v3i4.1115
Tersedia: https://journal.aritekin.or.id/index.php/Konstruksi

Plasmodium Falciparum and Plasmodium Vivax Malaria Detection Using Image Processing and Multi-Class CNN Classifier

Jamal M. Alrikabi *
Departemen Ilmu Komputer, Fakultas Pendidikan Ilmu Murni, Universitas Thi-Qar, Irak.
**Corespondensing author : jamal.alrikaby@utq.edu.iq*

***Abstract**. Millions of people suffer from malaria, one of the most serious parasitic diseases that threatens human life and causes high rates of morbidity and mortality, particularly in tropical and subtropical regions. Traditional diagnostic methods, such as blood smear examination, which can be performed using a microscope, face many challenges due to the inaccuracy of manual analysis and the reliance on individual skills. Therefore, the use of machine learning or deep learning algorithms to automate malaria detection offers promising solutions to improve accuracy, reduce diagnosis time, and enhance scalability. In this paper, a multi-class convolutional neural network (CNN)-based model is designed to classify cells infected with Plasmodium falciparum (P. falciparum) and Plasmodium vivax (P. vivax) and uninfected cells from blood smears, as most severe cases and deaths are caused by P. falciparum and P. vivax. This is achieved by building and training a CNN from scratch, rather than using transfer learning from pre-trained models. The proposed network was trained and tested on the Kaggle dataset, which consists of 27,558 images of infected and uninfected individuals. These images were divided into 13,779 images of uninfected individuals, 6,890 images of individuals with P. falciparum malaria, and 6,889 images of individuals with P. vivax malaria. The images were preprocessed using several operations, including blurring, denoising, and morphological processing. The proposed model achieved the best evaluation accuracy when compared with other deep learning algorithms, with an accuracy rate of 96.5%, a sensitivity rate of 95%, a specificity rate of 97.6%, and an F1-score rate of 96.5%. These results demonstrate the effectiveness of the proposed model as a tool to assist clinicians in malaria diagnosis, reducing reliance on manual analysis.*

***Keywords:** Convolutional Neural Network (CNN), Deep Learning, Feature Extraction, Image Processing, Malaria Detection, Plasmodium Infection.*

1. INTRODUCTION

Malaria is a deadly disease, according to the World Malaria Report 2019 published by the World Health Organization. Malaria is an infectious parasitic disease caused by a parasite (Plasmodium), transmitted through the saliva of female Anopheles mosquitoes, known as malaria vectors. Malaria has existed for 30 million years and was identified as a leading cause of death in ancient civilizations worldwide. Today, malaria remains a serious disease, with nearly half of the world's population at risk, although the WHO has identified the African Region as bearing a disproportionately large share of the global malaria burden. In 2023, the region accounted for approximately 94% of all malaria cases and 95% of malaria deaths. Children under five years of age accounted for approximately 76% of all malaria deaths in the region [1].

The disease is curable when detected early. Current methods for detecting malaria include microscopic detection of infected cells in the laboratory. This method is expensive, so a faster, low-cost and reliable alternative for microscopic detection of malaria is proposed by developing an algorithm for detecting malaria cells using image processing technology [2]. It

Naskah Masuk: 06 September, 2025; Revisi: 29 September, 2025; Diterima: 13 Oktober, 2025; Terbit: 16 Oktober, 2025

also gives us all the motivation to build a model that can detect cells from multiple blood cell images on standard microscope slides and classify them as infected or not with early and effective tests and perform classification on an image of an infected cell using machine learning techniques [3].

The main research problem focuses on building an automatic disease recognition model that will assist doctors in automatically diagnosing malaria by analyzing images of infected blood samples and providing an accurate diagnosis of whether the patient is infected with malaria or not by using image processing and deep learning techniques. This will save a lot of time needed by doctors for diagnosis and laboratory staff for analysis. It will also reduce the error that can occur (human error) as well as improve the accuracy of diagnosis. The use of computer automation in this medical field has become extremely important because it provides more accurate results, helping doctors improve outcomes and reduce error rates, which could help save lives [4]. The research aims to develop a new method for detecting malaria, which could replace current traditional diagnostic methods by physicians. This method contributes to reducing the number of unnecessary blood samples or smears and increasing reliability by assisting in correct diagnosis.

2. DEEP NEURAL NETWORK ARCHITECTURE

A deep neural network consists of several layers: the convolution layer, the pooling layer, and the fully connected layer as shown in Figure (1). At the first level of processing the input image, the deep neural network typically learns simple visual features such as edges or color, then at the second level, it combines the features of the previous level. Adding more levels leads to finding higher-level features [5]. Below is an explanation of the most important CNN layers.

Convolution Layer

The convolution layer is the most important part of a CNN and is always used as the first layer. The main task of this layer is to identify features present in local regions of the input image that are common to the entire dataset. Feature recognition produces a feature map by applying filtering factors. The convolution layer applies a local filter to the input image and produces the best correlation between the pixels of the input image. There are three important components of the convolution layer [6]:

1. The input image.
2. Feature detection.
3. The feature map.

The feature detector, also called kernel, is applied to the input image. By applying the convolution operator, the first value of the feature map table is obtained. The same process is then repeated to complete the first row of the feature map. After completing the first row, the next row is moved to the next row, and this process is repeated until the entire feature map is complete [6].

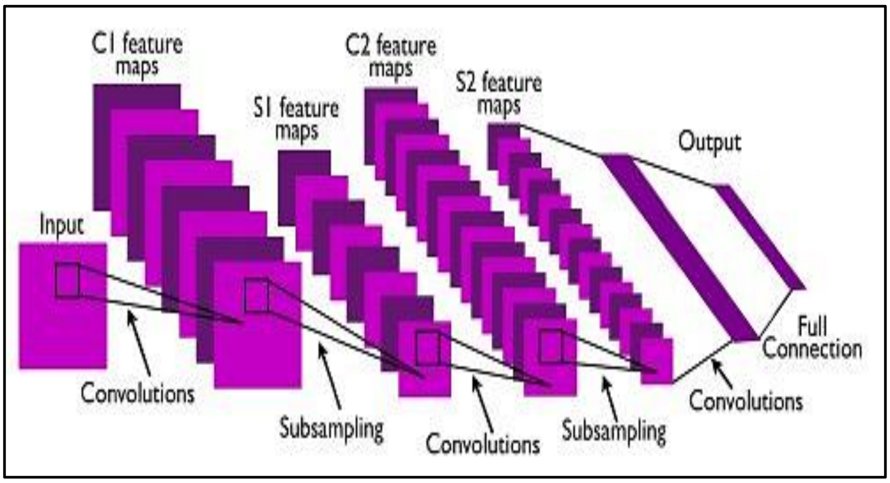


Figure 1. An overview of deep neural network architecture [5].

Pooling layer

This layer is usually used intermittently between two successive convolution layers. The pooling layer's task is to reduce the size of the feature maps and extract the important features from the feature map. There are two important pooling layer methods: max and average [7].

Fully connected layer

This layer is used for data classification and is considered a traditional multi-layer layer. It uses the SoftMax activation function, although other functions may be used. The term "fully connected" refers to the fact that each neuron in the previous layer is connected to each neuron in the next layer. The output from the convolution and pooling layers represents high-level features of the input image. The fully connected layer uses these features to classify the input image into multiple classes based on the training data [8].

3. RELATED WORKS

Nayak et al. [9] in 2019, trained various deep learning models and observed which of these models provided better accuracy than previously used deep learning models. The results showed that the Resnet50 model achieved the highest accuracy of 95%.

Sifat et al. [10] in 2020, proposed an automated detection of malaria parasites from blood smears. Images of blood smears from malaria patients were gathered from an internet resource for this study. Following some preprocessing, RBCs (red blood cells) were separated from the blood smear pictures using a U-Net, RBCs infected with malaria parasites were identified using a CNN, and various phases of malaria were identified using a VGG16 deep neural network. Using the CNN model, the detection accuracy and specificity of contaminated red blood cells were 95% and 100%, respectively. The 16 VGG model's average specificity and accuracy for identifying malaria species were 94.75% and 95.55%, respectively. Using the 16 VGG model, the average accuracy and specificity for various ring-stage malaria parasite species were 96.25% and 94.82%, respectively.

Kassim et al. [11] in 2021, proposed a novel framework, PVF-Net, for analyzing thick smear microscope images for malaria diagnosis at the image and patient levels. The proposed framework is capable of identifying whether a patient is afflicted and, in the case of malaria, whether the patient is infected with P.vivax or P.falciparum. PVF-Net uses a Mask R-CNN to identify potential Plasmodium parasites, a ResNet-50 classifier to filter out false positives, and a score derived from the total number of spots found and their combined probabilities across all patient photos to identify parasite species. A manually annotated dataset of 350 patients with roughly 6,000 photos was used in this study. The overall accuracy of the suggested work was 92% at both the patient and image levels.

Turuk et al. [12] In 2022, created a solid framework for deep learning-based automated malaria parasite detection, which can be a very useful tool for doctors diagnosing illnesses. The effectiveness of pre-trained CNNs-based deep learning models, including AlexNet, ResNet50, and VGG19, as feature extractors for examining infected and uninfected cells is assessed in this study. The optimum features extraction tool for that purpose is pre-trained CNN models, according to statistical data. With a training accuracy of 95.28% and a test accuracy of 93.89%, features extracted using VGG19 were shown to be more effective than ResNet-50 and Alex-Net models for detecting a presence of malaria parasites.

Alassaf et al. [13] In 2022, created an intelligent model based on transfer learning from CNN for the detection and categorization of malaria parasites using images from blood smears. The goal of the suggested IDTL-MPDC technique is to detect malaria parasites in blood-smear images with high accuracy. then extracted feature vectors using a Res-Net model, and then employed a differential evolution (DE) approach to improve its hyperparameters. The suggested technique classified blood smear images into the proper classifications using the KNN (k-nearest neighbor) classifier. The classification results are improved by employing the DE model to determine the Res2Net hyperparameters optimally. Using a benchmark dataset, a comprehensive set of simulation evaluations was conducted on the IDTL-MPDC approach.

The suggested approach obtained precision (95.86%), F1 score (95.69%), sensitivity (95.83%), specificity (95.92%) and accuracy (95.85%).

Xu et al. [14] in 2024, suggested a malaria diagnosis system that uses the CNN LeNet-5 and YOLO algorithms to detect malaria parasites using a "look-only-once" algorithm and classify the life stages of malaria parasites using a CNN algorithm. MBB and MP-IDB are two publicly available datasets that were used. P.vivax-infected human blood smears are part of the MBB dataset. Four species of malaria parasites—P.vivax, P.ovale, P.malariae, and P. falciparum—are included in the MP-IDB dataset. The detection accuracy is 0.92, and the classification accuracy is 0.93 for the MBB dataset. The suggested methods' classification and detection accuracies for the MP-IDB dataset were 0.96 and 0.92 for P. falciparum, 0.84 and 0.94 for P.vivax, 0.82 and 0.93 for P.ovale, and 0.79 and 0.93 for Plasmodium malariae. The Models trained on P.vivax alone demonstrated good detection capability for additional malaria parasite species, according to detection results.

4. METHODOLOGY AND EXPERIMENTS

This study's primary goal is to investigate how well convolutional neural networks perform in identifying patients who have malaria and, if they do, whether they have P.falciparum or P.vivax infection. This is done by building a deep learning model from scratch for the purpose of controlling the number of layers of the model and its parameters to obtain the highest accuracy instead of using the transfer learning method that relies on the use of pretrained models such as AlexNet, ResNet, DenseNet, and others. Figure (2) illustrates the block diagram of the most important stages of this research.

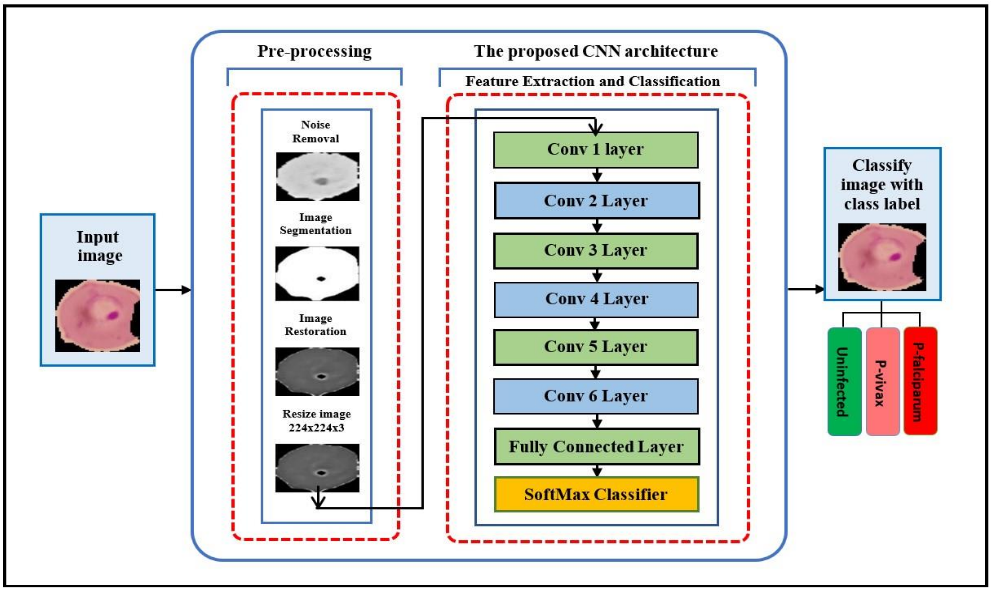


Figure 2. Block diagram of the proposed system.

Dataset Description

A widely used dataset was obtained for training and testing the proposed model. It was downloaded from Kaggle and contains 27,558 blood cell images [15]. The dataset is divided into two parts: 3,779 images of uninfected individuals and 13,779 images of infected individuals (6,890 with P. falciparum malaria and 6,989 with P. vivax malaria). Figure (3) shows a sample of images from the dataset used in this study.

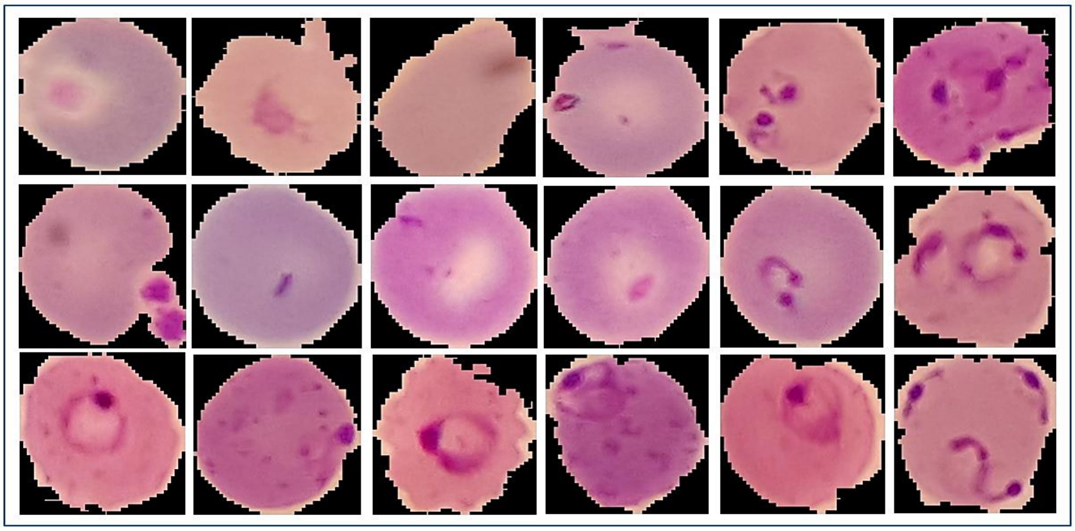


Figure 3. Sample of images from the dataset (P. falciparum, P. vivax, uninfected)

Data Preprocessing

Data preprocessing has an impact on feature extraction by the convolutional neural network and thus on the performance and accuracy of the SoftMax classifier. Therefore, the data in this study were preprocessed in several steps. First, we took an image from the dataset and applied the preprocessing steps. Then, using the same method and approach, we applied it to the entire dataset. After that, we began the preprocessing process. Since the images are units of 8, meaning they are quantized with 8 bits, the color images will be converted to grayscale images, as shown in Figure (4). All images input to the network must be of the same type, size, and color gradient [16].

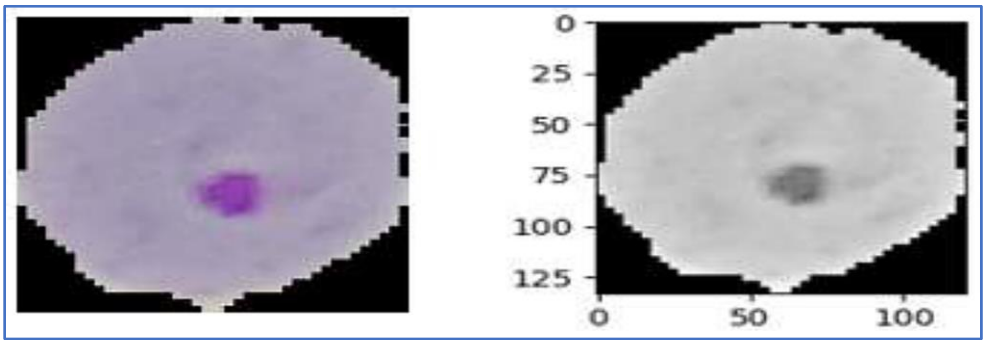


Figure 4. The distinction between the grayscale and original images.

Noise Removal using Linear Filters

These are median filters, such as the arithmetic mean filter, the geometric mean filter, and the harmonic mean filter. Each filter is suitable for a specific type of noise. For example, the harmonic mean filter is suitable for pulse noise, while arithmetic and geometric filters, such as the Gaussian filter, are suitable for random noise. The Gaussian filter is characterized by smoothing the image while preserving parts of the original image, most notably the edges. Figure (5) shows the result of applying the Gaussian filter to the image [16-17].

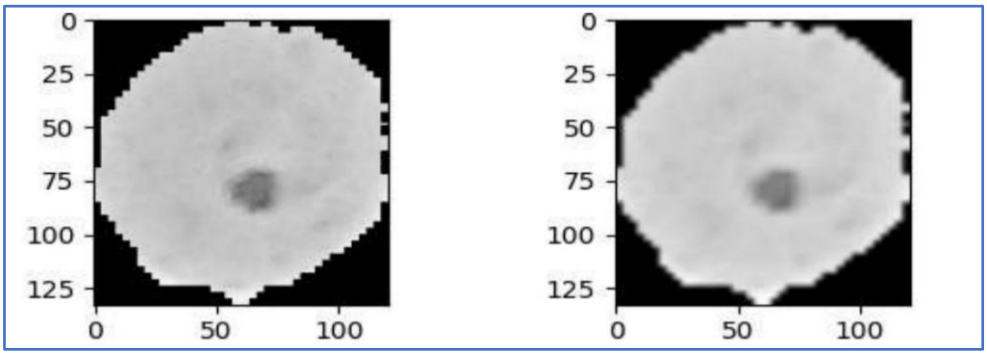


Figure 5. The distinction between the image before and after applying the filter. (the Gaussian filter).

Image Segmentation

The image segmentation process was performed in this study according to the following steps:

1. **Image Thresholding**

   We performed the thresholding process to convert the grayscale (filtered) image to binary. This was done automatically using the Otsu technique, depending on each image, as it is a fundamental and necessary process for obtaining connected components and applying morphological operations to them. Figure (6) shows the image after the thresholding process [18].

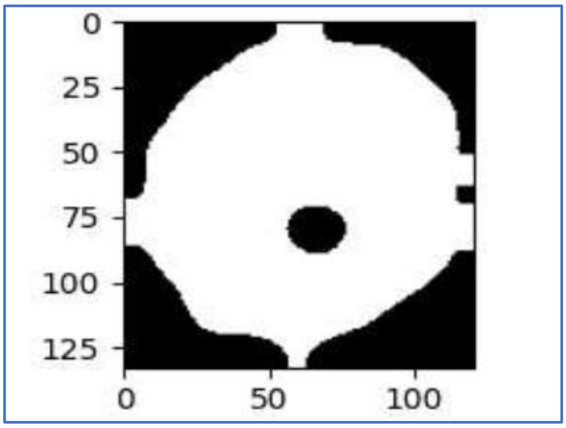


Figure 6. The image after applying the thresholding process.

2. Morphological Operations

The image used consists of a white area and a black area, and the black area (black dot) contains the disease. Through morphological transformations, the diseased area can be extracted. Morphological operations are simple operations performed on the shape of the image and are a tool for extracting useful image components. They are applied to binary images. To implement this operation, we need two inputs: the original image, and the second input is the structural element, which is the matrix responsible for determining the mask (kernel) that determines the nature of the operation. The structural element is usually chosen similarly to the object to be processed in the input image; therefore, the ellipse element was chosen. There are two basic operations: erosion and dilation [16-17].

a) **Erosion:** This is the process of shrinking objects in the image. This process smooths the boundaries of the foreground surface (which is white). This process reduces pixels with a value of 1 according to the specified mask, thereby reducing white areas and isolating elements from each other, as shown in Figure (7).

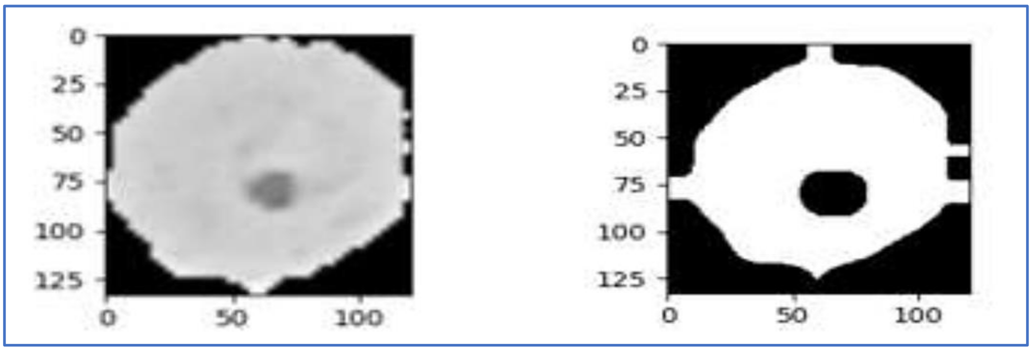


Figure 7. The image after the erosion process.

b) **Dilation:** This is the process of enlarging objects in the image, which is the opposite of the erosion process. Consequently, the size of the white area increases, as in Figure (8).

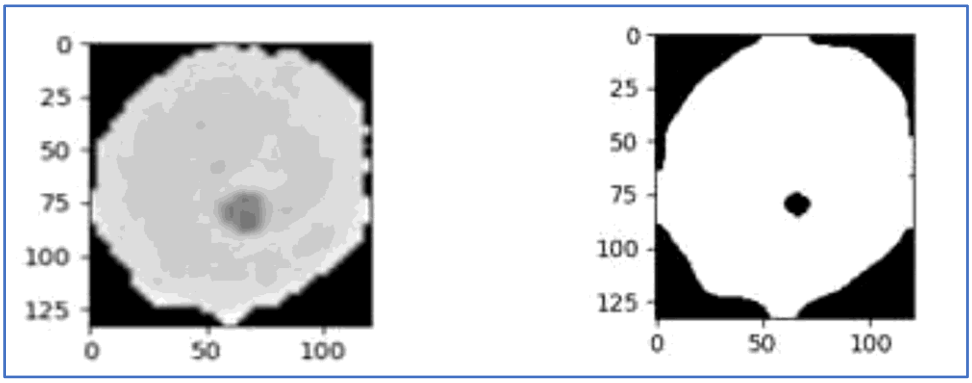


Figure 8. The image after the dilation process.

3. Image Restoration

Morphological operations can obtain the location and size of an object, but at the same time, these operations remove edges from the image. Edges are thin lines that pass between unrelated areas in the image or heterogeneous color fields that appear suddenly. Therefore, to restore an image, edges must be recovered [19].

Edge Detection Filters in the Spatial Domain

In a digital image, an edge refers to a sudden change in grayscale values. This change can be gradual or sharp. Therefore, first- and second-derivative filters are used. The first derivative shows the location of edges in the image, while the second derivative reveals more edges because it produces two edges: the first with a high value and the second with a low derivative value (next value - previous value). The second derivative takes a value of 1 when moving from the black column to the white column, indicating the presence of an edge in that region. Among the most important of these filters are the Sobel filter, which detects image edges in only one direction (vertical - horizontal - vertical - diagonal), and the Roberts filter, which features a small mask that detects angles of either 45° or -45°.

The second derivative gives a value of (1) and then a value of (-1), meaning it gives a maximum and minimum value, indicating that we crossed with zero and raised the value of the difference between the gradients, specifying the exact location of the edge. Therefore, the second derivative is more accurate. Among the most important of these filters is the Laplacian filter, which detects horizontal and vertical edges, or horizontal, vertical, and oblique edges together, and is considered one of the most powerful filters. The Canny filter extracts edges based on two basic stages [19]:

1. Image smoothing to reduce the effect of edges.
2. Edge selection based on strong and weak thresholds. A strong threshold produces edges, while a weak threshold takes weak edges associated with strong edges and ignores the rest. This results in more accurate edges. If edges have a gradient greater than the strong threshold, they are accepted. If they have a gradient less than the strong threshold, they are rejected immediately. If they are between the two thresholds, they are accepted if they are related to strong edges and their eight-way neighbors are strong. If they are weak, they are removed because they are not related to strong edges.

The edges of the original image were obtained using the Canny filter, and the image was scanned to remove image elements that do not constitute an edge. This detector performs several steps during the detection process, resulting in accurate edge detection. Figure (9) shows the threshold image after edge detection.

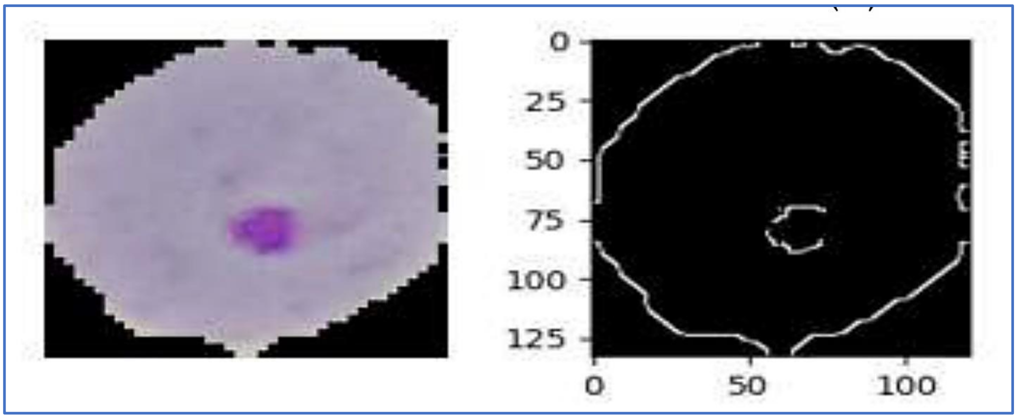


Figure 9. The threshold image after edge detection.

Merging the two images

After the edge detection process, the image is reconstructed by merging the image obtained after applying the morphological operations with the edges of the original image. Figure (10) shows the merging process and the resulting image.

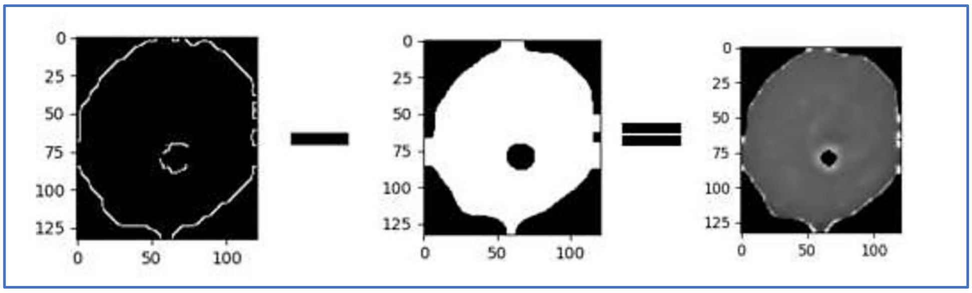


Figure 10. The result of merging the two images.

The Proposed Deep CNN Model

There are currently three main strategies for successfully using CNNs in medical image classification: The first is to learn the model from scratch, in which case the model structure is employed and trained based on a new dataset. The second method is to use transfer learning using features from pre-trained CNNs on large datasets, such as the ImageNet dataset. The third method is to use CNNs for transfer learning by retaining the original convolutional baseline and feeding its output to the classifier. The pretrained model is used as a fixed features extraction strategy when the dataset is short or the classification problem is similar [20].

Design of the proposed CNN model

In this study, we built a convolutional neural network from scratch instead of using transfer learning from well-known pre-trained deep network architectures such as AlexNet or ResNet. Choosing the correct architecture for a CNN is one of the challenges facing this type of network. This process requires careful consideration of a number of parameters, such as determining the layers number required, size of filters and the number of filters, and the correct values for the stride size. There are no fixed rules and standards for all of these parameters, as this network varies depending on the type and complexity of the data, the image size, the available hardware resources, and much more [21].

The architecture proposed in this study consists of a sequence of layers (convolutional, dropout, convolutional, max-pooling) placed sequentially. This structure is repeated three times, and the sizes of the selected filters, which represent the number of feature maps in the output for each layer, are 32, 64, and 128, respectively, for each iteration. This leads to an increase in the feature maps number, but with smaller sizes after each max pooling layer. The padding process is implemented in the convolutional layers, where padding is performed equally on the right and left, and if the number of columns to be added is odd, an additional column is added to the right. The same principle is applied vertically, with an additional row of zeros at the bottom. Finally, a number of high-density layers are added at the output end of the network to better translate the feature maps into the desired class. The designed deep network structure consists of:

1. The first layer is a convolutional layer consisting of (32) different kernels, with a (3×3) downsampling step, which gives us a single layer. This is followed by applying the Relo activation function.
2. The second layer is a convolutional layer consisting of (32) different kernels with a (3x3) down sampling step, giving us a single layer. This is followed by the application of the Relu activation function, immediately followed by a max pooling layer with a stride size of (2).
3. The third layer is a convolutional layer consisting of (64) different kernels with a (3x3) down-sampling step, giving us a single layer. This is followed by the application of the Relu activation function.
4. The fourth layer is a convolutional layer consisting of (64) different kernels with a (3x3) down-sampling step, giving us a single layer. This is followed by the application of the Relu activation function, immediately followed by a max pooling layer with a stride size of (2).
5. The fifth layer is a convolutional layer consisting of (128) different kernels with a (3x3) down sampling step, giving us a single layer. This is followed by the application of the Relu activation function.
6. The sixth layer is a convolutional layer consisting of (128) different kernels with a (3x3) down sampling step, giving us a single layer. This is followed by the application of the

Relu activation function, immediately followed by a max pooling layer with a stride size of (2).

7. The seventh stage is a fully connected layer that processes the extracted features and uses the Relu activation function.
8. The output (classification) layer is a fully connected layer and uses the SoftMax activation function.

Table (1) illustrates the basic architecture of the proposed CNN with its layers, showing the information of each layer separately.

Table 1. Illustrates the designed architecture of the proposed CNN

No.	Layer	Type	Kernel/ Stride /Pad	Parameters & Details	Output Shape	Total Parameters	Activation
1	Input	Input Layer	-	Input size: (132, 132, 3)	132×132×3	0	-
2	Conv1	Convolutional	Kernel:3×3, stride=1, pad=1	Filters: 32	132×132×32	(3×3×3)×32+32 = 896	ReLU
3	Dropout 1	Dropout	-	Dropout Rate: 20%	132×132×32	0	-
4	Conv2	Convolutional	Kernel:3×3, stride=1, pad=1	Filters: 32	132×132×32	(3×3×32)×32+32 = 9,248	ReLU
5	MaxPool 1	Max Pooling	Pool Size: 2x2, stride=2	-	66×66×32	0	-
6	Conv3	Convolutional	Kernel: 3x3, stride=1, pad=1	Filters: 64	66×66×64	(3×3×32)×64+64 = 18,496	ReLU

7	Dropout 2	Dropout	-	Dropout Rate: 20%	66×66×64	0	-
8	Conv4	Convolutional	Kernel: 3x3, stride=1, pad=1	Filters: 64	66×66×64	(3×3×64)×64+64 = 36,928	ReLU
9	MaxPool 2	Max Pooling	Pool Size: 2x2, stride=2	-	33×33×64	0	-
10	Conv5	Convolutional	Kernel: 3x3, stride=1, pad=1	Filters: 128,	33×33×128	(3×3×64)×128+128 = 73,856	ReLU
11	Dropout 3	Dropout	-	Dropout Rate: 20%	33×33×128	0	-
12	Conv6	Convolutional	Kernel: 3x3, stride=1, pad=1	Filters: 128,	33×33×128	(3×3×128)×128+128 = 147,584	ReLU
13	MaxPool 3	Max Pooling	Pool Size: 2x2, stride=2	-	16×16×128	0	-
14	Dropout 4	Dropout	-	Dropout Rate: 20%	16×16×128	0	-
15	Flatten	Flatten	-	Converts 3D output to 1D vector	32768 (16×16×128)	0	-
16	FC1	Fully Connected	-	Units: 512,	512	32768×512+512 = 16,777,728	ReLU
17	Dropout 5	Dropout	-	Dropout Rate: 20%	512	0	-

18	FC2	Fully Connected	-	Units: 256,	256	512×256+ 256 = 131,328	ReLU
19	Dropout 6	Dropout	-	Dropout Rate: 20%	256	0	-
20	FC3	Fully Connected	-	Units: 3 (for 3 classes)	3	256×3+3 = 771	ReLU
21	Output	SoftMax	-	Output predictions for three classes	1	0	SoftMax

Training the proposed deep model

Initial initialization works well for many neural networks and against many problems. However, it is necessary to fine-tune the key parameters to make the most of the model we are building and the available dataset. Exploring optimization algorithms and key parameter values can help build intuition for optimizing networks for different tasks [22]. When searching for key parameters, it is important to understand the sensitivity of optimization to learning rate, batch size, optimizer, etc. This, along with choosing the right method, helps in finding the right model. Table (2) shows the parameters through which the designed network was fine-tuned.

Table 2. The proposed deep network parameters.

Parameter	Value of Parameter
Epochs	20
Optimizer	Adam
Batch size	32
Input shape	132, 132, 3
Pooling	Max 2×2 (convolutional layers, flatten layer)
Activation	ReLU (convolutional layers), SoftMax (final dense layer)

For training our proposed model, we split the data used in this study into two sets (80% training - 20% testing), as shown in Figure (11).

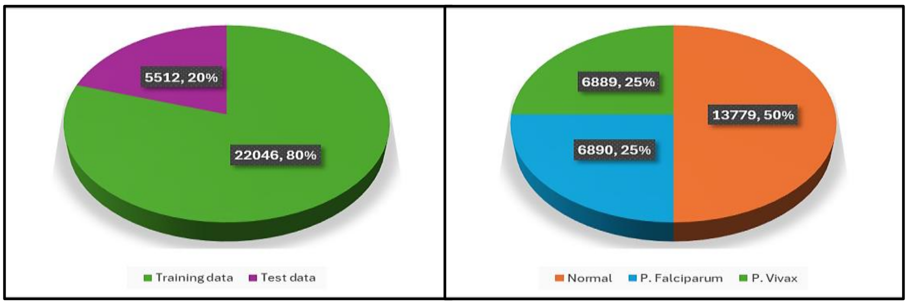


Figure 11. Data splitting to train and test the proposed model.

Experiments and results

This section includes analysis and evaluation of the performance of the proposed model and comparison with previous works.

Performance analysis

The performance of the proposed model was analyzed with all experiments based on the most common performance evaluation metrics used in the statistical testing (precision, accuracy, F-measure, and recall). The confusion matrix is a table used to evaluate the performance of a classification model. This matrix shows the number of correct and incorrect predictions made by the model compared to the actual results.

Evaluation of the Proposed Model

After completing the training and testing process, we plotted the accuracy and error curves for both the training and test data, as shown in Figure (12).

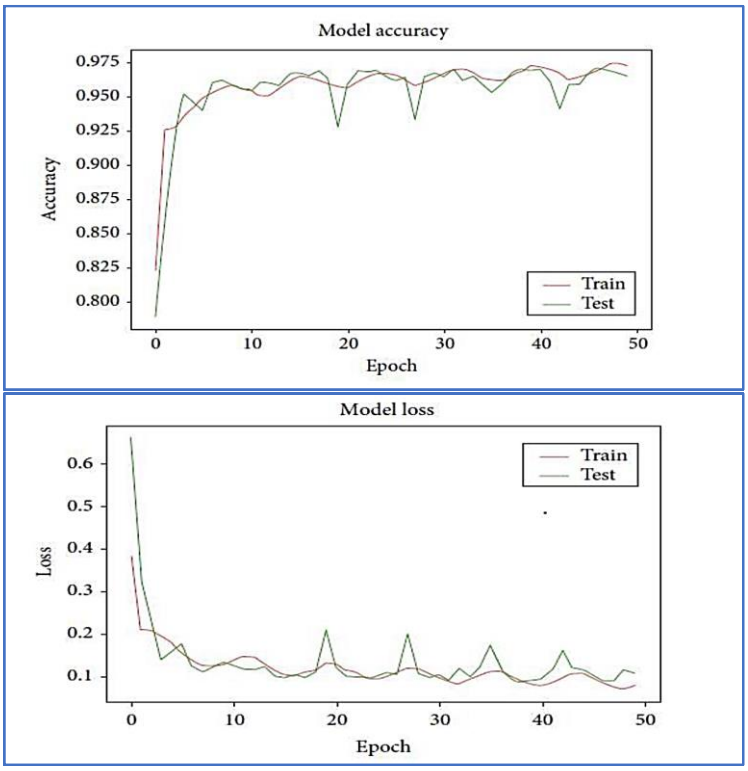
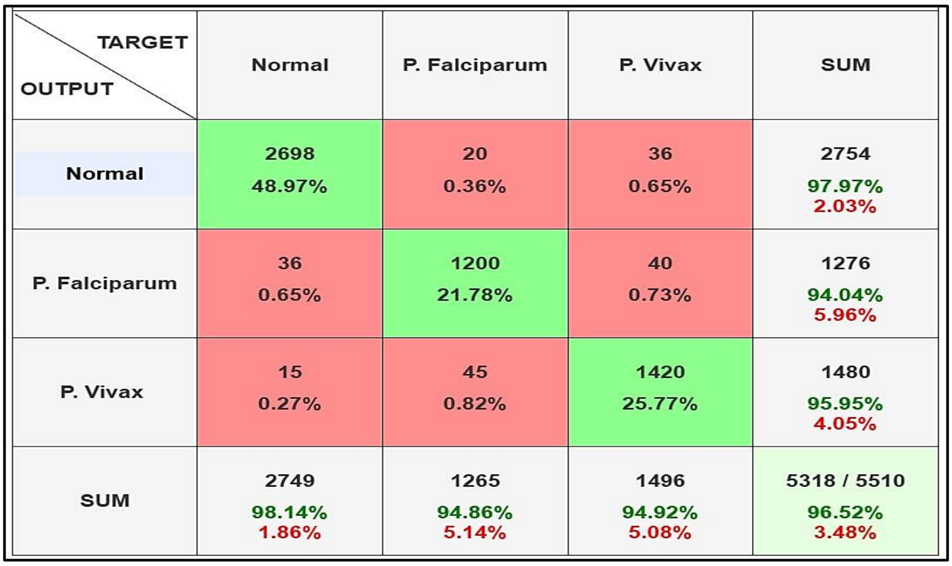


Figure 12. Accuracy and error curves for both the training and test data.

From Figure (12), we notice the absence of overfitting. This is because the curve representing the training accuracy in the dataset shows a significant increase in the model's accuracy after each training stage. The curve representing the test accuracy shows an increase in the test accuracy rate with the increase in the training accuracy. The same applies to the curves indicating the error rate produced by the model. It is clear that the error rate for the training set decreases after each epoch. We also note that the error rate for the test set decreases with the decrease in the error rate for the training set. Therefore, the model is capable of generalization because it was trained well during the training stage. When tested on new data that it was not trained on, the proposed model produces good results.

To illustrate the results of our model, we plotted a confusion matrix, which shows the percentage of cases in which the model correctly predicted the output, as well as the cases in which the model failed to predict the correct output. Figure (13) shows the confusion matrix for the proposed network, and Table (3) shows the results of the model's performance after training.



TARGET OUTPUT	Normal	P. Falciparum	P. Vivax	SUM
Normal	2698 48.97%	20 0.36%	36 0.65%	2754 97.97% 2.03%
P. Falciparum	36 0.65%	1200 21.78%	40 0.73%	1276 94.04% 5.96%
P. Vivax	15 0.27%	45 0.82%	1420 25.77%	1480 95.95% 4.05%
SUM	2749 98.14% 1.86%	1265 94.86% 5.14%	1496 94.92% 5.08%	5318 / 5510 96.52% 3.48%

Figure 13. Confusion matrix for the proposed model.

We note that out of 1,256 patients infected with malaria (P. falciparum), the model was able to correctly predict 1,200 patients. The model incorrectly classified 40 patients infected with P. falciparum as infected with P. vivax and classified 36 infected patients as not infected (normal). The proposed model correctly classified 1,420 people infected with malaria (P. vivax), but misclassified 45 people as infected with P. falciparum and misclassified 15 people as "normal" when they were actually infected. The model also correctly classified 2,698 people as "uninfected" and misclassified 56 people as "uninfected."

Table 3. Model performance results after training.

Accuracy	**Recall Or Sensitivity**	**Specificity**	**F1-Score**
96.5%	95%	97.6%	96.5%

Comparison with Previous Studies

The results obtained from the proposed model were compared with a set of previous reference studies that use convolutional neural networks to detect malaria, as shown in Table (4).

From the table (4), we note that the proposed model achieved the highest accuracy in diagnosing malaria patients compared to previous studies, with a classification accuracy of 96.5%. The proposed model also identifies the type of malaria a patient suffers from, whether P. falciparum or P. vivax. Figure (14) show the results of the proposed model with other models.

Table 4. Comparison of Results with Previous Reference Studies.

Reference	**Model**	**Accuracy**
Nayak et al. [9] (2019)	Resnet50	95%
Sifat et al. [10] (2020)	VggNet16	94.82%
Kassim et al. [11] (2021)	PVF-Net	92%
Turuk et al. [12] (2022)	VGG19	93.89%
Alassaf et al. [13] (2022)	Res2Net+KNN	95.86%
Xu et al. [14] (2024)	YOLO and LeNet-5	94%
Our proposed model (2025)	**CNN from scratch**	**96.5%**

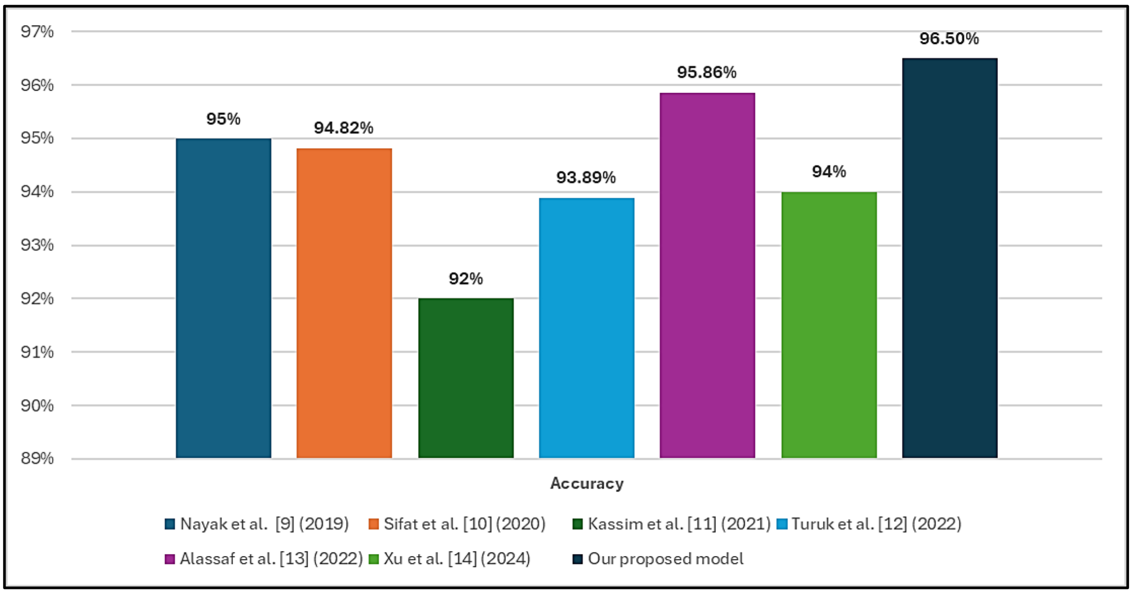


Figure 14. Comparing between the proposed model and other models.

5. CONCLUSIONS

In this study, a multi-classification deep learning model was built and evaluated for P. falciparum and P. vivax malaria detection. This model is important because it classifies malaria types in a single model. Correctly diagnosing these diseases is essential for determining appropriate treatment. From the results obtained, the proposed CNN model outperformed other models in reference studies. The CNN model achieved an accuracy of 96.5% and was able to train on multiple classifications. We also note a high convergence between the training and test curves, which is a good indicator of the absence of overfitting. We recommend combining more than one type of deep neural network to achieve better performance and using other AI techniques such as fuzzy logic and genetic algorithms, then comparing their performance with deep neural networks and training the model used to classify images of different diseases.

REFERENCES

Alassaf, A., & Sikkandar, M. Y. (2022). Intelligent deep transfer learning based on malaria parasite detection and classification model using biomedical image. *Computers, Materials & Continua, 72*(3), 5273-5285. https://doi.org/10.32604/cmc.2022.025577

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shammah, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data, 8*(1). https://doi.org/10.1186/s40537-021-00444-8

Chegeni, M. K., Rashno, A., & Fadaei, S. (2022). Convolution-layer parameters optimization in Convolutional Neural Networks. *Knowledge-Based Systems, 261*, 110210. https://doi.org/10.1016/j.knosys.2022.110210

Gu, N. K., Zhai, N. G., Lin, N. W., Yang, N. X., & Zhang, N. W. (2015). No-Reference image sharpness assessment in autoregressive parameter space. *IEEE Transactions on Image Processing, 24*(10), 3218-3231. https://doi.org/10.1109/TIP.2015.2439035

Jdey, I., Hcini, G., & Ltifi, H. (2023). Deep Learning and Machine Learning for Malaria Detection: Overview, challenges and future directions. *International Journal of Information Technology & Decision Making, 23*(05), 1745-1776. https://doi.org/10.1142/S0219622023300045

Kassim, Y. M., Yang, F., Yu, H., Maude, R. J., & Jaeger, S. (2021). Diagnosing Malaria Patients with Plasmodium falciparum and vivax Using Deep Learning for Thick Smear Images. *Diagnostics, 11*(11), 1994. https://doi.org/10.3390/diagnostics11111994

Khoei, T. T., Slimane, H. O., & Kaabouch, N. (2023). Deep learning: systematic review, models, challenges, and research directions. *Neural Computing and Applications, 35*(31), 23103-23124. https://doi.org/10.1007/s00521-023-08957-4

Maduri, P. K., Shalu, N., Agrawal, S., Rai, A., & Chaubey, S. (2021). Malaria detection using image processing and machine learning. 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), 1789-1792. https://doi.org/10.1109/ICAC3N53548.2021.9725557

Malaria Cell Images Dataset. https://www.kaggle.com/datasets/iarunava/cell-images-for-detecting-malaria. Last visited at 28-05-2025.

Mohammed, F. A., Tune, K. K., Assefa, B. G., Jett, M., & Muhie, S. (2024). Medical Image Classifications using Convolutional Neural Networks: A survey of current methods and statistical modeling of the literature. *Machine Learning and Knowledge Extraction, 6*(1), 699-736. https://doi.org/10.3390/make6010033

Qi, L., Wu, J., Li, X., Zhang, S., Huang, S., Feng, Q., & Chen, W. (2021). Photoacoustic tomography image restoration with spatially measured variable point spread functions. *IEEE Transactions on Medical Imaging, 40*(9), 2318-2328. https://doi.org/10.1109/TMI.2021.3077022

S. Nayak, S. Kumar, & M. Jangid. (2019). Malaria Detection Using Multiple Deep Learning Approaches. 2019 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT), Jaipur, India, 292-297. https://doi.org/10.1109/ICCT46177.2019.8969046

Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D. J., & Norouzi, M. (2022). Image Super-Resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1-14. https://doi.org/10.1109/TPAMI.2022.3204461

Shrestha, A., & Mahmood, A. (2019). Review of Deep learning Algorithms and Architectures. *IEEE Access, 7*, 53040-53065. https://doi.org/10.1109/ACCESS.2019.2912200

Sifat, M. M. H., & Islam, M. M. (2020). A Fully Automated System to Detect Malaria Parasites and their Stages from the Blood Smear. 2017 IEEE Region 10 Symposium (TENSYMP). https://doi.org/10.1109/TENSYMP50017.2020.9230761

Siłka, W., Wieczorek, M., Siłka, J., & Woźniak, M. (2023). Malaria detection using advanced deep learning architecture. *Sensors, 23*(3), 1501. https://doi.org/10.3390/s23031501

Sora-Cardenas, J., Fong-Amaris, W. M., Salazar-Centeno, C. A., Castañeda, A., Martínez-Bernal, O. D., Suárez, D. R., & Martínez, C. (2025). Image-Based Detection and Classification of Malaria Parasites and Leukocytes with Quality Assessment of Romanowsky-Stained Blood Smears. *Sensors, 25*(2), 390. https://doi.org/10.3390/s25020390

Taye, M. M. (2023). Theoretical understanding of convolutional neural network: concepts, architectures, applications, future directions. *Computation, 11*(3), 52. https://doi.org/10.3390/computation11030052

Turuk, M., Sreemathy, R., Kadiyala, S., Kotecha, S., & Kulkarni, V. (2022). CNN Based Deep Learning Approach for Automatic Malaria Parasite Detection. *IAENG Int. J.Comput. Sci.*, *49*, 745-753

Wali, A., Naseer, A., Tamoor, M., & Gilani, S. (2023). Recent progress in digital image restoration techniques: A review. *Digital Signal Processing, 141*, 104187. https://doi.org/10.1016/j.dsp.2023.104187

Xu, T., Theera-Umpon, N., & Auephanwiriyakul, S. (2024). Staining-Independent Malaria Parasite Detection and life stage classification in blood smear images. *Applied Sciences, 14*(18), 8402. https://doi.org/10.3390/app14188402

Z. Cai, & C. Peng. (2021). A study on training fine-tuning of convolutional neural networks. 13th International Conference on Knowledge and Smart Technology (KST), Bangsaen, Chonburi, Thailand, 84-89. https://doi.org/10.1109/KST51265.2021.9415793