Malaria Detection in Blood Cells

Submitted in partial fulfilment of the requirements of the degree of

Master of Science

By

Amanda Judy Andrade

Supervisor:

Dr. Alex Sumarsono

Department of Computer Science and Engineering, School of Engineering

Santa Clara University

2023–2024

Santa Clara University

500 El Camino Real, Santa Clara CA 95053.
Department of Computer Science and Engineering, School of Engineering

CERTIFICATE

This is to certify that the project entitled "Malaria Detection in Blood Cells" is a bonafide work of
Amanda Judy Andrade
submitted to Santa Clara University in partial fulfilment of the requirement for the award of the **Graduate** degree in **Master of Science in Computer Science and Engineering**.

Date:

(Dr. Alex Sumarsono)
Supervisor, Chair of CSE Department

(Dr. Sally Wood)
Associate Dean: School of Engineering

(Dr. Elaine Scott)
Dean: School of Engineering

Santa Clara University

500 El Camino Real, Santa Clara CA 95053.

Department of Computer Science and Engineering, School of Engineering

Project Report Approval for M.S.

This project report entitled “Malaria Detection in Blood Cells” by Author Name is approved for the **Graduate** degree of **Master of Science in Computer Science and Engineering**.

External Examiner:

Supervisor: Dr. Alex Sumarsono

.

Date:

Place:

Santa Clara University

500 El Camino Real, Santa Clara CA 95053.

Department of Computer Science and Engineering, School of Engineering

Declaration

I declare that this written submission represents my ideas in my own words. Where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all academic honesty and integrity principles and have not misrepresented, fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will cause disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

Amanda Judy Andrade

Date:

ABSTRACT

Malaria remains a global health concern, particularly affecting Sub-Saharan Africa, Southeast Asia, the Eastern Mediterranean, the Western Pacific, and the Americas. With nearly half of the world's population at risk in 2020, efforts to combat the disease have intensified. This paper explores the critical need for accurate and efficient malaria diagnosis, emphasising the impact on vulnerable populations, especially children under the age of five.

The project delves into the current scenario of malaria diagnosis, highlighting the challenges associated with the manual counting of blood films and the need for standardisation. The paper proposes automating the malaria diagnosis through deep learning methods to address these issues, explicitly employing the EfficientNet B3 model. The chosen model demonstrates superior performance, achieving an overall accuracy of 98% with high precision.

The literature review provides insights into existing technologies such as the Inception Module, ResNet-50, Inception-ResNet-v2, and NasNet, comparing their advantages and limitations. EfficientNet's compound scaling method is showcased as a breakthrough in achieving optimal trade-offs between model size and accuracy.

The paper details the methodology adopted, incorporating transfer learning with EfficientNet B3, data pre-processing image data generators, and custom callbacks for dynamic learning rate adjustment. The proposed system's hardware and software requirements and design details are outlined, ensuring a comprehensive understanding of the implementation process.

The project aims to address the shortcomings of manual diagnosis by leveraging deep learning for automated and standardised malaria detection. The results showcase the efficiency of the proposed system, with implications for enhancing diagnostic accuracy, reducing costs, and ultimately improving patient outcomes. The paper concludes by emphasising the importance of continued research and development in automated malaria diagnosis for global health advancement.

CONTENTS

LIST OF FIGURES

INTRODUCTION

PROBLEM STATEMENT

Malaria [1] is a potentially fatal disease caused by parasites spread to humans through the bites of disease-ridden female Anopheles mosquitos. It is both preventable and treatable. Malaria is caused by five parasite species, two of which, Plasmodium falciparum and Plasmodium vivax, pose the greatest threat.

Nearly half of the world's population was in danger of malaria in 2020. Sub-Saharan Africa has the highest number of cases and deaths. However, a considerable number of cases and deaths have been reported in the WHO regions of Southeast Asia, the Eastern Mediterranean, the Western Pacific, and the Americas.

In 2021, there were approximately 247 million cases of malaria, with a total of 619,000 malaria fatalities. The WHO African Region bears a disproportionately large percentage of the worldwide malaria incumbrance. In 2020, the region accounted for 95% of malaria cases and 96% of malaria deaths.

Children under the age of five are the most vulnerable group to malaria, accounting for over 80% of all malaria deaths in the WHO African Region in 2021.

By analysing the blood cells which are infected and not infected, robust medical equipment can be developed to detect and diagnose patients suffering from the parasite. Other projects, such as MRI 3D Reconstruction, can also be researched through this project.

SCOPE OF THE PROJECT

Every year, hundreds of millions of blood films [2] are checked for malaria, which entails a qualified microscopist manually counting parasites and infected red blood cells. Not only are accurate parasite counts required for malaria diagnosis, but They are also helpful for testing for medication resistance, assessing drug efficacy, and categorising disease severity. However, microscopic diagnosis is not standardised and greatly depends on the microscopist's expertise and skill.

It is usual for microscopists in low-resource settings to labour in solitude, with no strict structure to maintain their abilities and, hence, diagnostic quality. This leads to inaccurate diagnostic conclusions in the field. False-negative cases result in the wasteful use of antibiotics, a second consultation, missed days of work, and, in some circumstances, progression to severe malaria. False-positive instances result in the inappropriate use of anti-malaria medications, as well as the potential adverse effects of these drugs, such as nausea, abdominal pain, diarrhoea, and potentially severe consequences.

This sober assessment of malaria diagnosis has encouraged initiatives to automate malaria diagnosis.

The potential of automating malaria diagnosis, with its apparent benefits, has piqued the interest of many experts, particularly in the recent decade. The publications cover the essential advances in automatic pattern recognition and machine learning during the last few years. Our post will provide a summary of the articles that have been published, utilising the above-mentioned processing processes as a framework and guide. This is not the first poll article on the topic. Several survey articles have already been published, attesting to the importance of automated malaria detection, research dynamics, and rapid system development.

We recommend the following surveys for more information on the history of automatic malaria diagnosis and the image processing and machine learning algorithms used for automated microscopy diagnosis of malaria.

CURRENT SCENARIO

The potential of automating malaria diagnosis, with its apparent benefits, has piqued the interest of many experts, particularly in the recent decade. The publications cover the important advances in automatic pattern recognition and machine learning during the last few years. Over 170 papers have been proposed for Malaria Detection through blood smears; however, very few models achieved an accuracy above 95%.
In this paper, the main goal is to maximise the accuracy and precision of the model so that it can be deployed in a real-world application.

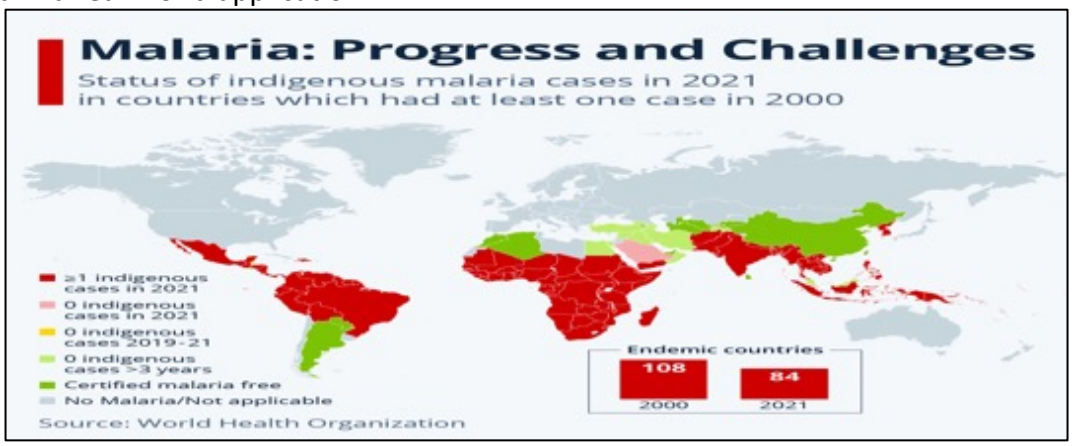

Figure 1: Malaria Indigenous Cases in 2021 (Source: WHO)

Malaria

Malaria is a potentially fatal disease transferred to people by certain mosquitoes. It is found primarily in tropical nations. It is both preventable and treatable. A parasite causes the infection, which does not transfer from person to person.
Plasmodium falciparum, Plasmodium vivax, Plasmodium malariae, Plasmodium ovale, and Plasmodium knowlesi are the five Plasmodium species that cause malaria in humans. P. falciparum and P. vivax are the two most frequent species. The most severe variety, P. falciparum, is responsible for the majority of malaria-related deaths worldwide.
P. falciparum is the prevalent malaria parasite in Sub-Saharan Africa, accounting for 99% of malaria cases reported in 2016. P. vivax is the most common parasite outside of Africa, accounting for 64% of malaria cases in the WHO Region of the Americas, more than 30% in the WHO Southeast Asia and 40% in the Eastern Mediterranean areas.
During their growth cycle (48 hours), each parasite species goes through stages that give the parasites a distinct visual appearance that may be seen under a microscope. These stages are the ring, trophozoite, schizont, and gametocyte in chronological order.
Symptoms might range from minor to life-threatening. Mild symptoms include a fever, chills, and a headache. Fatigue, confusion, seizures, and difficulty breathing are examples of severe symptoms.
Infants, children under the age of five, pregnant women, travellers, and persons living with HIV or AIDS are all at greater risk of severe infection.
Malaria can be avoided by avoiding mosquito bites and using malaria medications. Treatments can prevent mild cases from worsening.

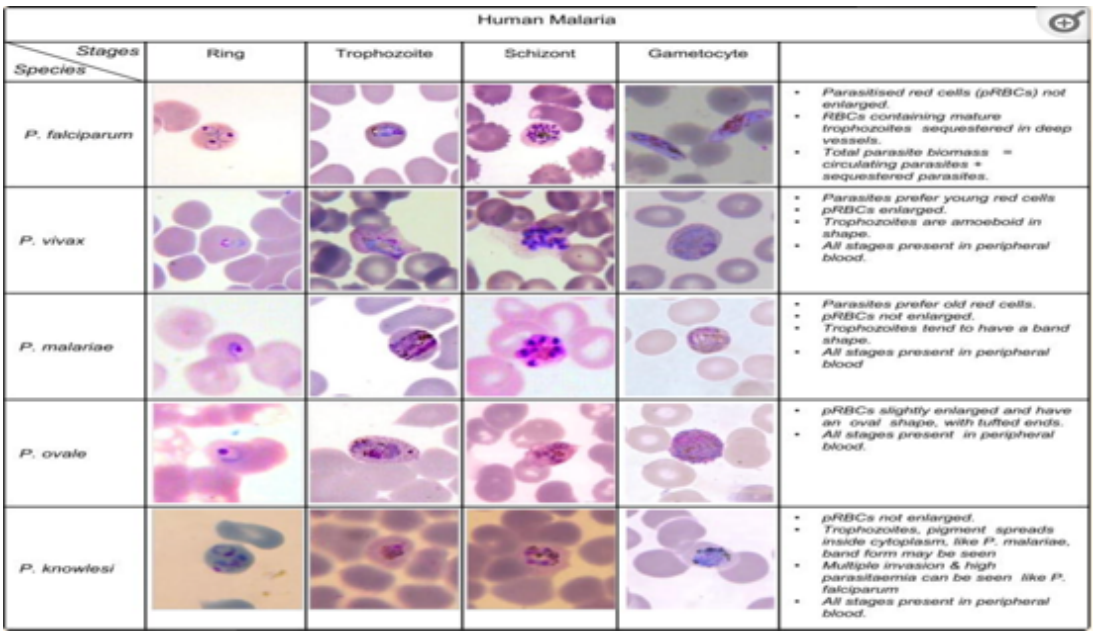

Figure 2: Five different human malaria Plasmodium species and their life stages in thin blood film (Source: K. Silamut and CDC)

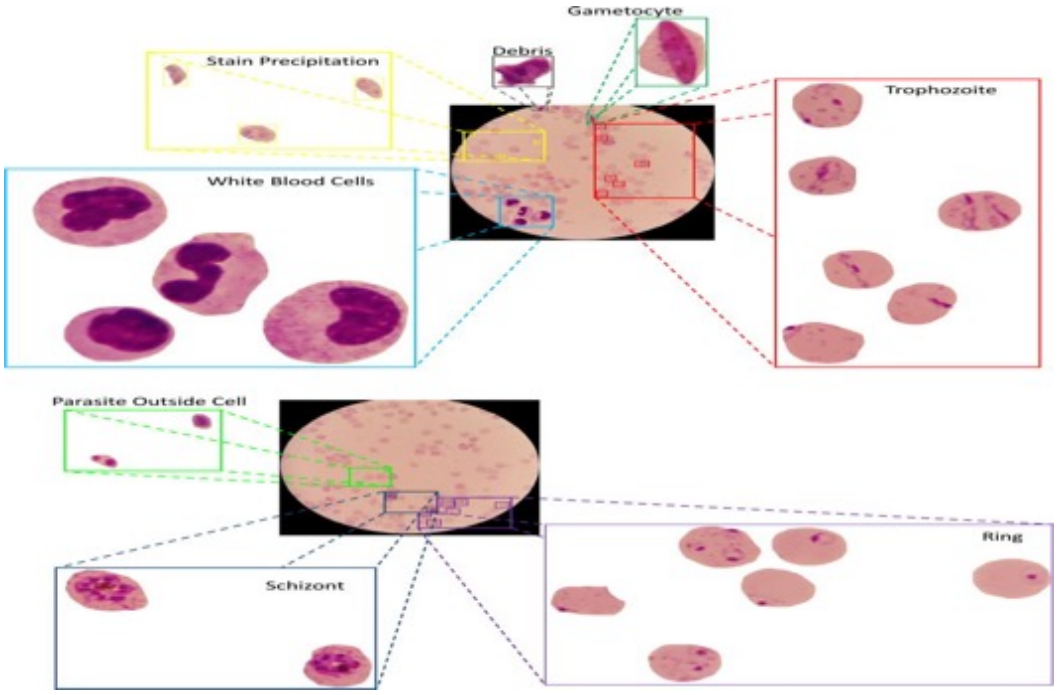

Figure 3: Parasite stages in a single thin blood smear.

Malaria Diagnosis

Malaria is a treatable disease, with medications available to help prevent malaria infections in travellers to malaria-prone areas. However, no effective malaria vaccine exists despite ongoing research and field testing. Malaria is a rapidly growing disease once infected, with a high chance of evolving into severe and cerebral malaria with neurologic symptoms in P. falciparum infections. As a result, early malaria diagnosis is critical. Although malaria can be diagnosed using various methods, existing malaria diagnostic techniques have space for cost, specificity, and convenience of use improvement.

NEED FOR THE PROPOSED SYSTEM

When opposed to manual counting, automatic parasite counting has numerous advantages:
(1) It gives a more reliable and standardised interpretation of blood films.
(2) It allows more patients to be serviced by lowering the effort of malaria field workers.
(3) It can cut diagnostic expenses.

SUMMARY OF THE RESULTS / TASK ACCOMPLISHED

This project's final classification step was accomplished using EfficientNet B3 and a self-defined Model Layer for the Neural Network. The overall accuracy of 98% was achieved with a high precision of 98%.

LITERATURE REVIEW

SUMMARY OF THE INVESTIGATION OF EXISTING TECHNOLOGY

Inception

The Inception Module [3] is a convolutional neural network (CNN) building block first described by Google researchers in their critical paper "Going Deeper with Convolution" in 2014. This architecture, commonly known as GoogLeNet, dramatically improves deep learning for computer vision tasks. The Inception Module enables a CNN to profit from multi-level feature extraction by incorporating filters of varying sizes in the same network layer.

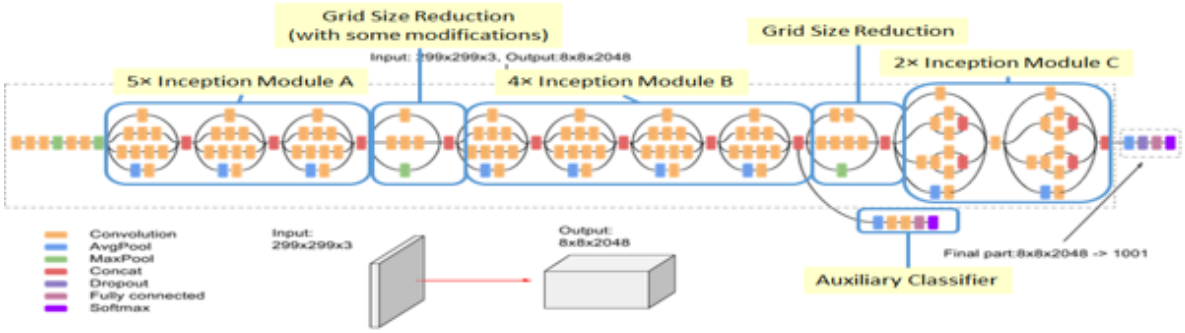

Figure 4: Inception Architecture

In comparison to standard CNN architectures, the Inception Module has numerous advantages:

i. **Efficiency:** By creating filters of various sizes, the module efficiently uses computational resources to extract valuable features without needing deeper or more extensive networks.
ii. **Reduced Overfitting:** The intricacy and depth of the architecture aid in learning more robust features, which can reduce overfitting, particularly when combined with other regularisation techniques.
iii. **Improved Performance:** Networks incorporating Inception Modules outperformed other networks on various benchmark datasets for image recognition and classification tasks.

While the Inception Module has numerous advantages, it also has several drawbacks:
Increased Complexity: The Inception Module's architecture is more complex than standard layers, making building and training difficult.
Tuning Hyperparameters: The module introduces new hyperparameters, such as the number and size of filters, that must be carefully tuned to obtain optimal performance.
Resource Intensity: Despite being built for efficiency, the Inception Module can be resource-intensive due to the massive operations and output concatenation.

ResNet-50

ResNet [4] is a type of convolutional neural network (CNN) introduced in the 2015 publication "Deep Residual Learning for Image Recognition" by He Kaiming, Zhang Xiangyu, Ren Shaoqing, and Sun Jian. CNNs are widely utilised in computer vision applications.
ResNet-50 is a convolutional neural network of 50 layers (48 convolutional layers, one MaxPool layer, and one average pool layer). Residual neural networks (RNNs) are artificial neural networks (ANNs) that build networks by stacking residual blocks.

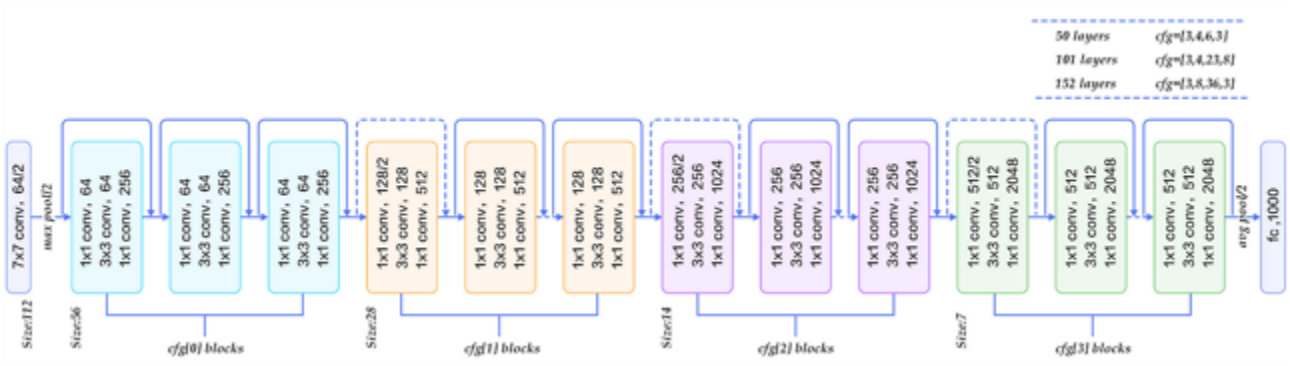

Figure 5: ResNet-50 Architecture

ResNet-50's architecture is similar to the model shown above but with one crucial distinction. The bottleneck design is used for the building block in the 50-layer ResNet. A bottleneck residual block employs 11 convolutions to limit the number of parameters and matrix multiplications. This allows for significantly faster layer training. It employs a three-layer stack rather than two levels.
The following elements are included in the 50-layer ResNet architecture, as stated in the table below:

- A 7 × 7 kernel convolution with 64 additional kernels and a 2-sized stride.
- A maximum pooling layer with a two-size stride.
- 9 more layers—3 × 3, 64 kernel convolution, 1 × 1, 64 kernel convolution, and 1 × 1, 256 kernel convolutions. These three levels are repeated three times.
- 12 more layers with 1 × 1, 128 kernels, 3 × 3,128 kernels, and 1 × 1,512 kernels, iterated 4 times.
- 18 additional layers with 1 × 1, 256 cores and 2 cores 3 × 3, 256 and 1 × 1, 1024, iterated 6 times.
- 9 more layers with 1 × 1, 512 cores, 3 × 3,512 cores, and 1 × 1, 2048 cores iterated three times (the network now has 50 layers)
- After average pooling, a fully linked layer with 1000 nodes is created using the SoftMax activation function.

ResNet [5] has several advantages that make it a popular deep-learning algorithm:

- **Deeper Networks**: ResNet makes it possible to build deeper neural networks with more than a hundred layers, which were previously unachievable due to the vanishing gradient problem. The remaining connections enable the network to learn better representations and optimise the gradient flow, making deeper networks easier to train.
- **Improved Accuracy:** ResNet has demonstrated greater accuracy than other deep neural network architectures on numerous benchmark datasets, including ImageNet, CIFAR-10, and CIFAR-100.
- **Faster Convergence:** ResNet allows faster training convergence due to residual connections that allow for improved gradient flow and optimisation. As a result, training time is reduced, and convergence to the best solution is improved.
- **Transfer Learning:** ResNet is well-suited to transfer learning, which allows the Network to reuse previously learnt features for new tasks. This is especially advantageous when the amount of Labelled data available is limited, as the pre-trained ResNet may be fine-tuned on the fresh dataset to obtain good performance.

Despite its various advantages, ResNet has a few downsides to consider:

- **Complexity:** ResNet is a sophisticated architecture that demands more memory and computing resources than shallower networks. This can be restricted in settings with low resources, such as mobile devices or embedded systems.
- **Overfitting:** ResNet is susceptible to overfitting, mainly when the Network is excessively deep, or the dataset is small. This can be addressed by utilising regularisation techniques like dropout or smaller networks with fewer layers.
- **Interpretability:** The interpretability of ResNet can be difficult since the Network learns sophisticated and abstract representations that are difficult to understand. This might be a disadvantage in situations requiring interpretability, such as medical diagnosis or fraud detection.

Inception ResNet

Inception-ResNet-v2 [6] is a convolutional neural architecture that extends the Inception family of architectures using residual connections (which replace the Inception design's filter concatenation stage).

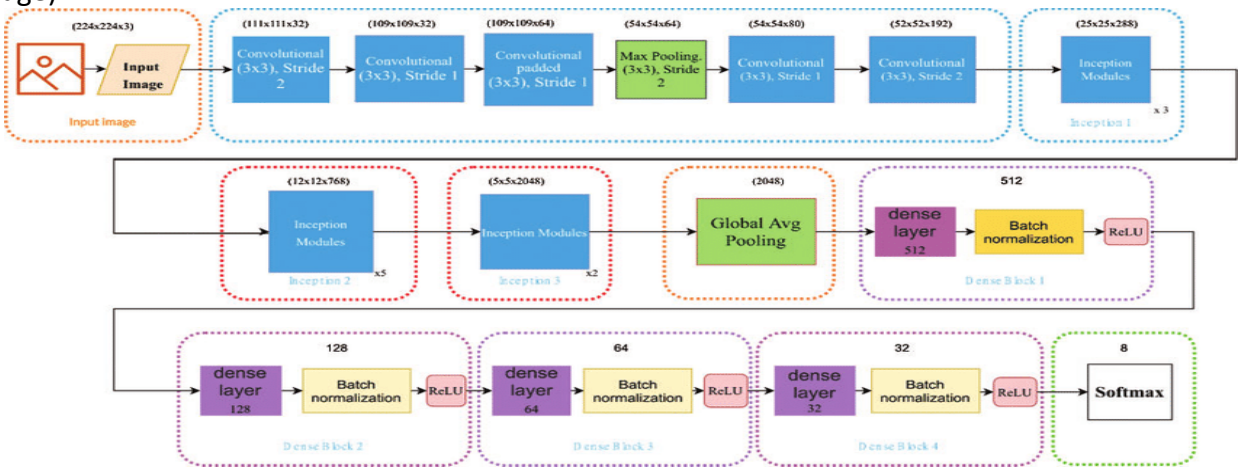

Figure 6: Inception ResNet Architecture

N-Inception [6]	25.2%	7.8%
ception-v3 [15]	21.2%	5.6%
ception-ResNet-v1	21.3%	5.5%
ception-v4	20.0%	5.0%

Figure 7: Comparison Table

NasNet

Neural architecture search (NAS) [7] is a strategy for automating the creation of artificial neural networks (ANN), a popular machine learning model. NAS has been used to create networks that perform as well as or better than hand-designed topologies. Methods for NAS are classified based on the search space, search technique, and performance estimation strategy employed:

- The type(s) of ANN that can be created and optimised are defined by the search space.
- The search strategy defines the approach utilised to explore the search space.
- The performance estimation technique assesses the performance of a potential ANN based on its design (rather than building and training it).

NAS is a subfield of automated machine learning (AutoML) related to hyperparameter optimisation and meta-learning.

Because of the costly training and evaluation phases, neural architecture search frequently necessitates considerable computational resources.

COMPARISON BETWEEN THE TOOLS/METHODS/ALGORITHMS

As one can see from the performance chart below, EfficientNet dramatically exceeds all the pre-existing models deployed for Malaria Detection in Blood Smears. Therefore, this project implements EfficientNet B3 as the final model for Model Classification.

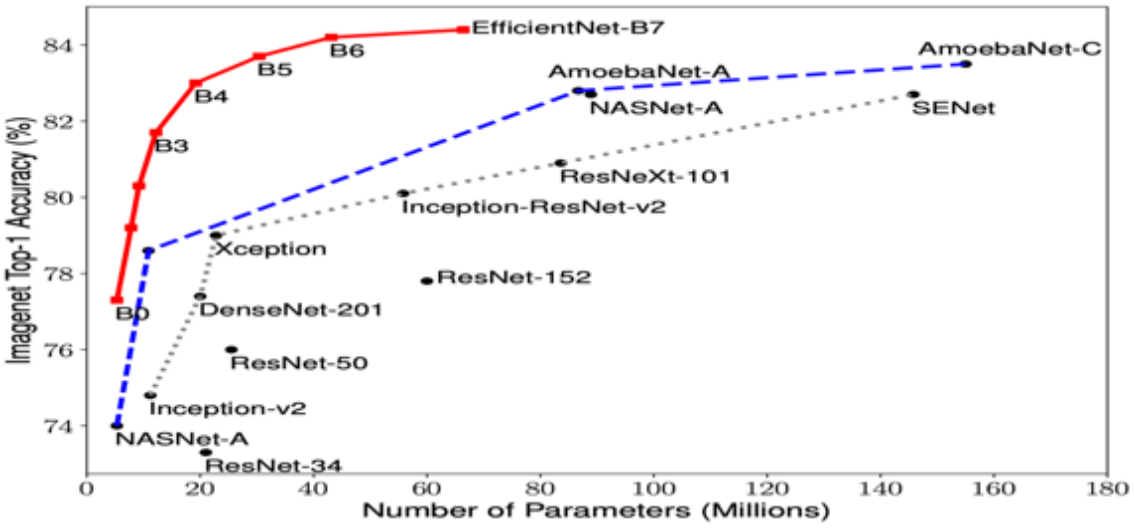

Figure 8: Comparisons of Models used

The largest EfficientNet model, EfficientNet B7, achieved cutting-edge performance on the ImageNet and CIFAR-100 datasets. On ImageNet, it achieved roughly 84.4% top-1/and 97.3% top-5 accuracy. Furthermore, the model size was 8.4 times lower, and the speed was 6.1 times faster than the previous best CNN model. The CIFAR-100 data set achieved 91.7% accuracy and 98.8% accuracy on the Flowers dataset.

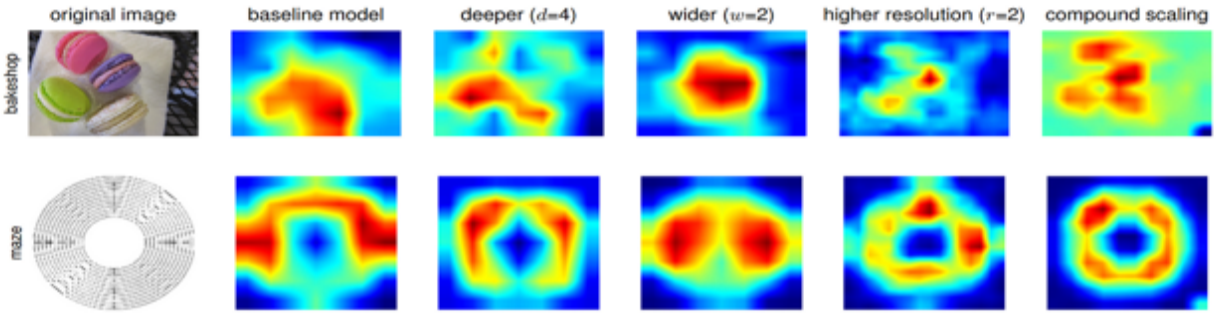

Figure 9: The compound scaling method allows the scale critical model (last column) to focus on more relevant regions with more object details (Source: image from the original paper).

The model also provided superior Class Activation Maps (CAM), which focused more on the critical regions with more object features, opening the path for improved model explainability.

ANALYSIS AND DESIGN

METHODOLOGY / PROCEDURE ADOPTED

Using Deep Learning Methods, we used a pre-trained Model as a top layer to our model. The dataset was downloaded from the National Institute of Health Website.

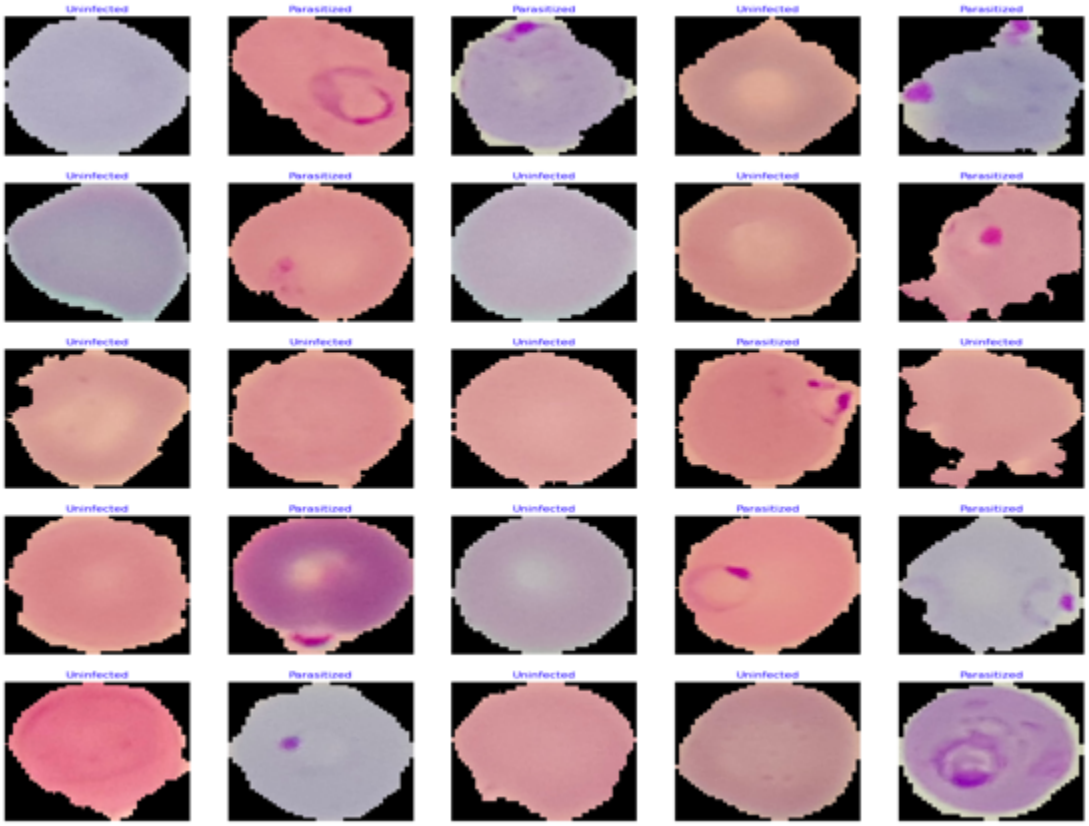

Figure 10: Sample Images from the Training Data

EFFICIENTNET

EfficientNet [8] scales up models in a simple but effective way using a technique known as compound coefficient. Compound scaling uniformly scales each dimension with a predetermined set of scaling coefficients rather than randomly scaling up width, depth, or resolution. The authors of EfficientNet created seven models of varying dimensions using the scaling approach and AutoML, which outperformed the state-of-the-art accuracy of most convolutional neural networks while being far more efficient.

$$\text{Depth } d = \alpha^{\emptyset}, \text{Width } w = \beta^{\emptyset}, \text{Resolution } r = \gamma^{\emptyset}, (1)$$

$$\text{such that } \alpha.\beta^2.\gamma^2 \approx 2$$

$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

Figure 11: Compound Scaling

The network's intuition is that if the input image is larger, the network will require more layers to extend the receptive field and more channels to catch more fine-grained patterns on the larger image. When compared to other random scaling methods, the compound scaling strategy improved

the model efficiency and accuracy of prior CNN models, such as MobileNet and ResNet, by roughly 1.4% and 0.7%, respectively.
EfficientNet is built on the baseline network generated by the AutoML MNAS framework's neural architecture search. The network is fine-tuned for optimum accuracy but is penalised if it is computationally costly. When the network takes time to create predictions, it is penalised for slow inference time. The architecture employs a mobile inverted bottleneck convolution, similar to MobileNet V2; however, it is significantly more significant due to the rise in FLOPS. This baseline model is scaled up to create the EfficientNets family.

PROPOSED SYSTEM

Hardware Requirements

2.7 GHz CPU
16 core GPU
16 GB RAM

Software Requirements

Python 3.8.10
Tensorflow 2.9.1
Keras 2.4 or above

DESIGN DETAILS

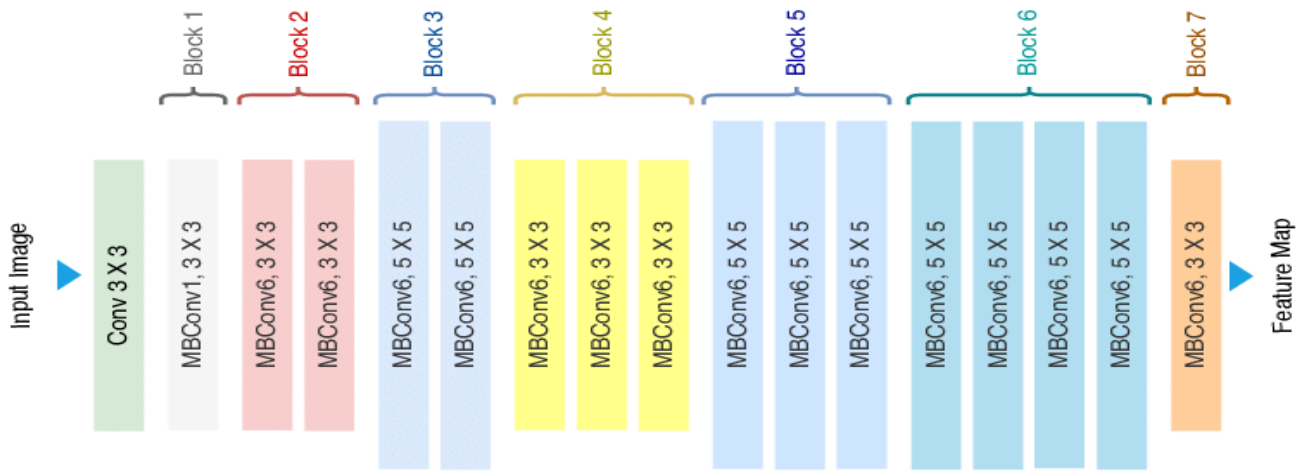

Figure 12: EfficientNet Architecture

EfficientNet [9] employs Mobile Inverted Bottleneck (MBConv) layers, which combine depth-wise separable convolutions and inverted residual blocks. Furthermore, the model architecture employs Squeeze-and-Excitation (SE) optimisation to further improve the model's performance.
The MBConv layer is a critical component of the EfficientNet design. It is inspired by MobileNetV2's inverted residual blocks but with various modifications.
The MBConv layer begins with a depth-wise convolution, then a point-wise convolution (1x1 convolution) that increases the number of channels, and another, 1x1 convolution that decreases the number of channels back to the original number. This bottleneck design enables the model to learn quickly while retaining a high level of representational capability.
In addition to MBConv layers, EfficientNet includes the SE block, which teaches the model to focus on essential characteristics while suppressing less important ones. The SE block employs global

average pooling to compress the feature map's spatial dimensions to a single channel, followed by two fully connected layers.
These layers enable the model to learn channel-wise feature dependencies and generate attention weights multiplied by the original feature map to highlight significant information.
EfficientNet is available in several forms with differing scaling coefficients, such as EfficientNet-B0, EfficientNet-B1, etc., reflecting a different trade-off between model size and accuracy, allowing users to choose the best model variant for their needs.

Generic Model Creation

```
# Create Model Structure
img_size = (224, 224)
channels = 3
img_shape = (img_size[0], img_size[1], channels)
class_count = len(list(train_gen.class_indices.keys())) # to define number of classes in dense layer

# create pre-trained model (you can built on pretrained model such as :  efficientnet, VGG , Resnet )
# we will use efficientnetb3 from EfficientNet family.
base_model = tf.keras.applications.efficientnet.EfficientNetB3(include_top= False, weights= "imagenet", input_shape= img_shape, pooling= 'max')

model = Sequential([
    base_model,
    BatchNormalization(axis= -1, momentum= 0.99, epsilon= 0.001),
    Dense(256, kernel_regularizer= regularizers.l2(l= 0.016), activity_regularizer= regularizers.l1(0.006),
                bias_regularizer= regularizers.l1(0.006), activation= 'relu'),
    Dropout(rate= 0.45, seed= 123),
    Dense(class_count, activation= 'softmax')
])

model.compile(Adamax(learning_rate= 0.001), loss= 'categorical_crossentropy', metrics= ['accuracy'])

model.summary()
```

Figure 13: Model Code

IMPLEMENTATION

1) **Data Pre-processing:** The code reads images from the "cell_images" directory and organises them into a data frame with file paths and corresponding labels. The dataset is split into training, validation, and test sets using the `train_test_split` function.
2) **Image Data Generator:** Image data generators are created for training, validation, and test sets. These generators handle data augmentation for the training set. The code uses the EfficientNetB3 model from TensorFlow's Keras applications as the base model. It adds a custom dense layer on top for classification. Callbacks are defined for dynamic learning rate adjustment and the ability to halt training based on user input. The model is trained using the `fit` function, and the training history is stored in the `history` variable.
3) **Model Performance Visualisation:** Training and validation loss, as well as accuracy, are plotted over epochs. The best epochs based on validation loss and accuracy are highlighted in the plots.
4) **Model Evaluation:** The trained model is evaluated on the training, validation, and test sets, and the loss and accuracy are printed.
5) **Confusion Matrix and Classification Report:** The code generates a confusion matrix and a classification report for the test set.
6) **Model Saving:** The trained model is saved as an HDF5 file, and its weights are saved separately.
7) **Custom Callback for Training:** The code defines a custom callback (`MyCallback`) for dynamic learning rate adjustment and user interaction during training.
8) **EfficientNetB3 Model Architecture:** The EfficientNetB3 model is used as a base model, and additional layers are added for classification.
9) **Use of EfficientNet Pre-trained Weights:** The EfficientNetB3 model is initialised with pre-trained weights from ImageNet.
10) **Data Augmentation:** Data augmentation is applied during training using the `ImageDataGenerator` for the training set.

The code is designed for a binary classification problem related to malaria detection, and it employs transfer learning with the EfficientNetB3 model. The code also includes extensive monitoring and logging during training to facilitate model evaluation and tuning.

Deploy and train the model for a maximum of 40 epochs. Using the Callback Function Class, if there is no significant improvement in the Validation Loss, stop the training and stick to the Learning Rate at the Last Epoch. Here, the training stopped after 13 epochs. Thus, it takes a training time of 01:00:28 hours. We choose the epoch with the lowest validation loss, i.e., epoch 12.

Epoch	Loss	Accuracy	V_loss	V_acc	LR	Next LR	Monitor	% Improv	Duration
1 /40	2.755	95.296	0.92905	97.793	0.00100	0.00100	val_loss	0.00	320.43
2 /40	0.552	97.387	0.29341	97.853	0.00100	0.00100	val_loss	68.42	275.46
3 /40	0.225	97.777	0.17059	98.004	0.00100	0.00100	val_loss	41.86	275.54
4 /40	0.150	98.390	0.13721	97.732	0.00100	0.00100	val_loss	19.57	276.93
5 /40	0.125	98.648	0.14209	97.611	0.00100	0.00050	val_loss	-3.56	275.57
6 /40	0.094	99.424	0.13323	98.034	0.00050	0.00050	val_loss	2.90	275.72
7 /40	0.085	99.487	0.12784	97.641	0.00050	0.00050	val_loss	4.04	275.06
8 /40	0.072	99.809	0.11963	97.944	0.00050	0.00050	val_loss	6.42	274.67
9 /40	0.066	99.796	0.12084	97.944	0.00050	0.00025	val_loss	-1.01	275.19
10 /40	0.060	99.891	0.11774	98.095	0.00025	0.00025	val_loss	1.58	275.38
11 /40	0.056	99.936	0.11786	98.065	0.00025	0.00013	val_loss	-0.10	276.05
12 /40	0.054	99.955	0.11861	98.125	0.00013	0.00006	val_loss	-0.74	275.90
13 /40	0.052	99.959	0.11964	98.095	0.00006	0.00003	val_loss	-1.61	274.77

training has been halted at epoch 13 after 3 adjustments of learning rate with no improvement
training elapsed time was 1.0 hours, 0.0 minutes, 28.01 seconds)

Figure 14: Training Time

Set Callback Parameters

```
batch_size = 64   # set batch size for training
epochs = 40   # number of all epochs in training
patience = 1   #number of epochs to wait to adjust lr if monitored value does not improve
stop_patience = 3   # number of epochs to wait before stopping training if monitored value does not improve
threshold = 0.9   # if train accuracy is < threshold adjust monitor accuracy, else monitor validation loss
factor = 0.5   # factor to reduce lr by
ask_epoch = 5   # number of epochs to run before asking if you want to halt training
batches = int(np.ceil(len(train_gen.labels) / batch_size))    # number of training batch to run per epoch

callbacks = [MyCallback(model= model, patience= patience, stop_patience= stop_patience, threshold= threshold,
            factor= factor, batches= batches, epochs= epochs, ask_epoch= ask_epoch )]
```

Figure 15: Callback Parameters

This code defines several parameters and configurations for implementing Epochs during the training of a neural network.

1. **batch_size:** The number of samples to be used in each training batch. In this case, it is set to 64.
2. **Epochs:** The total number of training epochs, representing the number of times the entire dataset is passed through the neural network for training. Here, it is set to 40.
3. **Patience:** The number of epochs to wait without improvement in the monitored value before adjusting the learning rate. The learning rate will be adjusted if there is no improvement for the specified number of epochs. In this case, it is set to 1.
4. **stop_patience:** The number of epochs to wait before stopping training if the Threshold value does not improve. The training will be halted if there is no improvement for the specified number of epochs. Here, it is set to 3.
5. **Threshold:** If the training accuracy is less than the threshold, the monitored value for adjusting the learning rate will be the training accuracy; otherwise, it will be the validation loss. If the training accuracy is less than 0.9 (90%), the monitored value is the training accuracy.
6. **Factor:** The factor by which to reduce the learning rate. If adjustments to the learning rate are necessary, it will be reduced by this factor. Here, it is set to 0.5.
7. **ask_epoch:** After this number of epochs, the training process will ask whether to continue training or halt. In this case, it is set to 5.
8. **Batches:** The number of training batches to run per epoch. It is calculated based on the length of the training data and the specified batch size.
9. **Callbacks:** A list containing an instance of the **MyCallback** class. This callback is used during training to implement specific actions, such as adjusting the learning rate, based on the defined parameters.

RESULTS

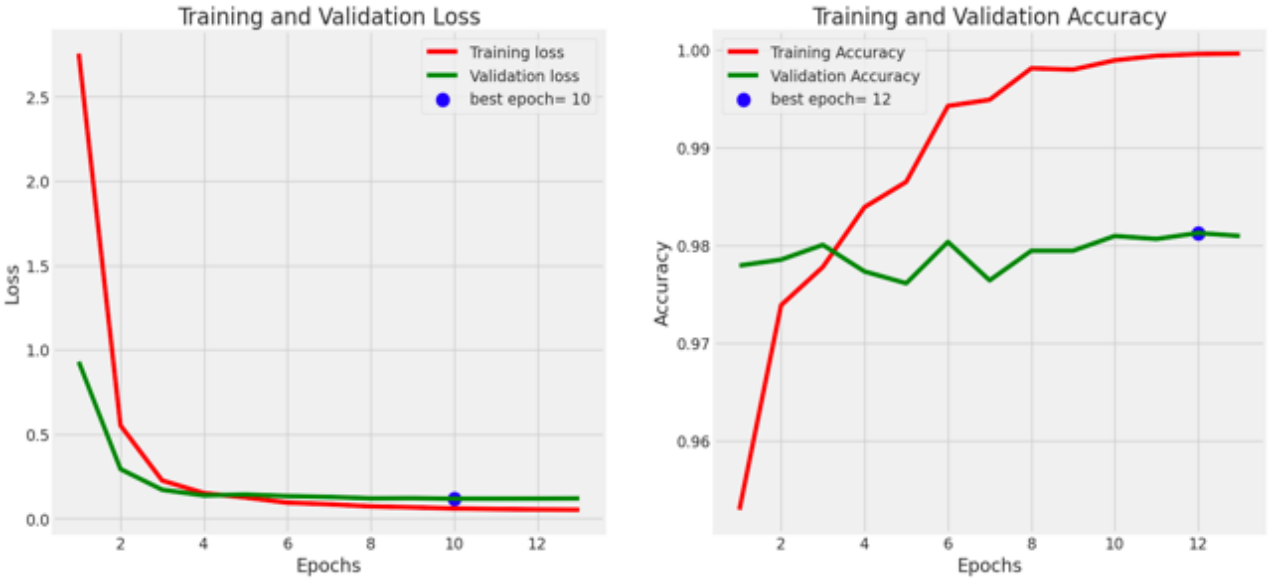

Figure 16: Model Performance

```
1103/1103 [==============================] - 65s 59ms/step - loss: 0.0478 - accuracy: 1.0000
1103/1103 [==============================] - 10s 9ms/step - loss: 0.1177 - accuracy: 0.9809
1103/1103 [==============================] - 23s 21ms/step - loss: 0.1477 - accuracy: 0.9755
Train Loss:  0.047815050929784775
Train Accuracy:  1.0
--------------------
Validation Loss:  0.11773886531591415
Validation Accuracy:  0.9809495210647583
--------------------
Test Loss:  0.14772573113441467
Test Accuracy:  0.9755213260650635
```

Figure 17: Model Evaluation

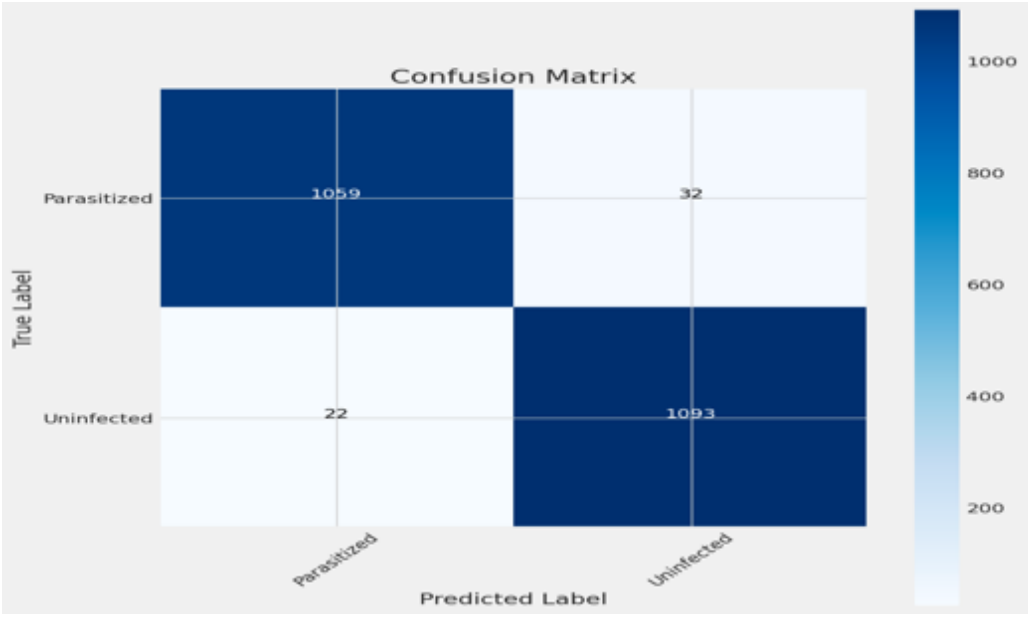

Figure 18: Confusion Matrix

	precision	recall	f1-score	support
Parasitized	0.98	0.97	0.98	1091
Uninfected	0.97	0.98	0.98	1115
accuracy			0.98	2206
macro avg	0.98	0.98	0.98	2206
weighted avg	0.98	0.98	0.98	2206

Figure 19: Classification Report

CONCLUSION

The model achieved an overall accuracy of 98%, which performed better than the other models utilised earlier (ResNet: 90%, VGG-19: 76%, NasNet: 96%, etc.)

BIBLIOGRAPHY

[1] World Health Organisation (WHO), “Malaria,” 02 October 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/malaria.

[2] K. S. R. J. M. S. J. a. G. T. Mahdieh Poostchi, “Image analysis and machine learning for detecting malaria,” *National Library of Medicine,* vol. 4, no. 194, pp. 36-55, April 2018.

[3] DeepAI, “DeepAI,” [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/inception-module. [Accessed 15 November 2023].

[4] DataGen, [Online]. Available: https://datagen.tech/guides/computer-vision/resnet-50/#.

[5] javatpoint., [Online]. Available: https://www.javatpoint.com/resnet-residual-network.

[6] [Online]. Available: https://paperswithcode.com/method/inception-resnet-v2.

[7] Wikipedia, “Wikipedia,” [Online]. Available: https://en.wikipedia.org/wiki/Neural_architecture_search.

[8] Medium.com, “Medium.com,” [Online]. Available: https://medium.com/mlearning-ai/understanding-efficientnet-the-most-powerful-cnn-architecture-eaeb40386fad.

[9] [Online]. Available: https://blog.roboflow.com/what-is-efficientnet/.