Malaria Detection using Machine Learning

Harshit Goyal
harshit20203@iiitd.ac.in

Madhava Krishna
madhava20217@iiitd.ac.in

Shreya Bhatia
shreya20542@iiitd.ac.in

Srishti Singh
srishti20409@iiitd.ac.in

Abstract

Malaria is a life-threatening spread by infected Anopheles mosquito bites. Existing means of diagnosis include light microscopy and rapid diagnostic tests, which are used in conjuction to provide accurate results. However, the costs associated with them, in terms of human capital and time required, are immense.

We seek to provide a complementing approach to infection classification using machine learning, which is fast and inexpensive. By training different algorithms like logistic regression, boosted decision trees, support vector machines and convolutional neural networks on images of varying sizes and using image transformations to augment the dataset, we conduct a comprehensive study on model accuracy and inferencing time.

1. Introduction

Malaria is an infectious disease caused by 5 species of the Plasmodium parasite: *Plasmodium falciparum*, *Plasmodium vivax*, *Plasmodium malariae*, *Plasmodium ovale* and *Plasmodium knowlesi*, spread by bites of the Anopheles mosquito. An estimated 241 million infections and 627,000 deaths occurred in 2020-21 [1].

1.1. Testing Methods

The infection can be detected using microscopy tests, Rapid Diagnostic Tests (RDTs) and serological tests.[2]

Microscopy tests involve collecting and dyeing a thin or thick blood specimen with Giemsa or Wright's stain to detect infections visually and ascertain the percentage of infected to uninfected cells.

RDTs indicate whether the patient is infected with one of the species of the malaria-causing *Plasmodium* and provide results in about 15 minutes. However, they fail to indicate a premature infection and negative RDT results need further evaluation. Using microscopy is also advised with positive results, so that the proportion of parasitized to uninfected cells can be determined.

Serological tests examine whether antibodies for the infection are present. They are mostly used for screening blood donors, testing for questionable diagnosis accompanied with treatment.

1.2. Role of Machine Learning

Numerous machine learning models have been proposed which segment a Whole Slide Image to identify red blood cells (RBCs) and classify these RBCs with a secondary trained model using deep neural network architectures and boosted trees. Our goal is to provide a computationally modest model with a good accuracy for the latter task, and provide meaningful results on how image dimensions and colour channel affect the accuracy of the above proposed models. Once these goals have been achieved, we will delve into image segmentation techniques to isolate RBCs from wholeslide images to pipeline the whole inferencing, if time permits.

2. Literature Survey

3. Dataset

The dataset used was publicly available, courtesy of images were taken at Chittagong Medical College Hospital, Bangladesh.[3]

3.1. Dataset Description

The dataset contains 13,799 parasitized and 13,799 uninfected image samples containing 3 colour dimensions for a total of 27,588 images. The images are of varying sizes. The maximum height and width was 385 and 394 pixels respectively. The minimum height are width was 40 and 46 pixels respectively. The mean height and width was 133 and 132 pixels respectively. The median height and width was 130 pixels. The mean aspect ratio of the images is 1.0138.

Out of the 27,588 images, 647 parasitized and 750 unparasitized images were misclassified [4].

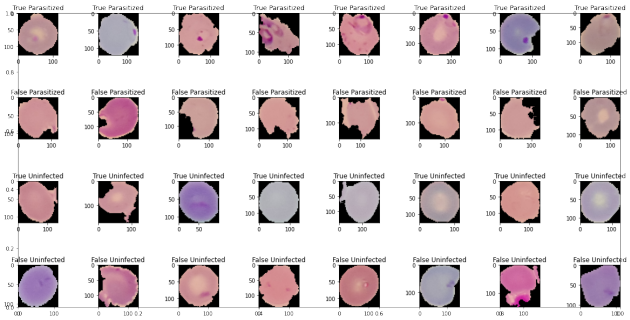

Figure 1. True and false parasitized and uninfected images.

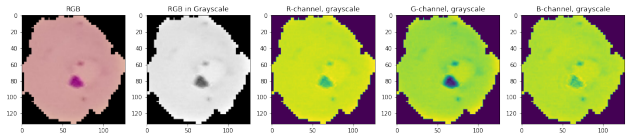

Figure 2. Comparison between various colour channels for a true parasitized cell.

4. Methodology

In order to reduce redundancy and obtain results in the form we desired, we focused on creating modules with specific objectives. Modules for downloading and setting up the dataset, for labelling the said dataset, to perform evaluation of models and transform images to augment the dataset were created. We have included the package requirements in a requirements.txt file, which can be used for easy installation using pip.

4.1. Exploratory Data Analysis

In order to determine which colour channel was the clearest with respect to the identification of the chromatin dot characteristic to the parasitized cell, we plotted the images in different colour channels and in grayscale.

Out of the plotted images, the green channel showed the maximum isolation of the chromatin dot. We also visualised inverted images and noticed that the green channel had isolated the chromatin dot the most.

We experimented with colour model transformations, and noticed that some models applying non-linear transformations (like HSV, HLS) captured the chromatin dot in parasitized cells better.

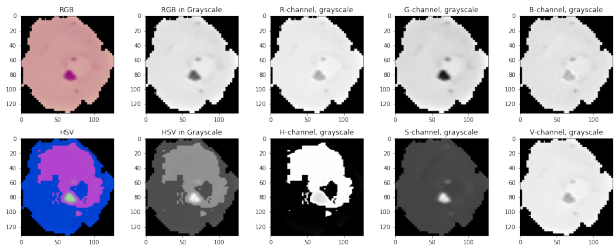

Figure 3. Conversion to HSV space from RGB.

Figure 4. KMeans and Natural Labels. The dataset was reduced to 2 dimensions using t-SNE.

4.1.1 Cluster Visualisation

To visualise them, the images were first resized to a 50x50 colour format, each pixel value rescaled by 1/255, and each image finally flattened to a 50*50*3 length array. We used Euclidean distance as the distance metric and used the t-SNE algorithm to reduce dimensions to 2. We also used KMeans clustering to determine similarity clusters, but there was no clear relation between the natural clusters and the clusters output by K-Means.

4.2. Preprocessing

The images were standardised to prespecified dimensions and colour model and each pixel scaled to a value between 0 and 1.

Most of the efforts were gone into identifying which colour transformation would be relevant for building better models, and for creating modules for transforming the data to the required format. We aim to explore more preprocessing methods after the mid-evaluation.

4.3. Data Augmentation

Modules created for augmentation provide functionality similar to TensorFlow's ImageDataGenerator [5], capable of random rotation, translation, scaling, random noise induction, colour space transformation, cropping, and denoising. We intend to use these to augment the data for model training.

4.4. Prelimnary Models

Some algorithms were implemented on unaugmented data. The 70% of data was reserved for training, 15% for validation and 15% for testing. Hyperparameters were tuned based on results observed on the validation data and final evaluation done on the testing data. The images were resized to 25x25 dimensions and colour was preserved as RGB. For training, the images were flattened to a 25*25*3 array. Images which were ambiguous [4] were removed from the dataset, leaving us with 8770 parasitized and 8701 uninfected training images, 2392 parasitized and 2373 unparasitized images for the validation dataset, and 1970 parasitized and 1955 uninfected images for the test dataset. Data augmentation was not used, and we intend to develop it more after the mid-evaluation

4.4.1 Naive Bayes

We used Gaussian Naive Bayes without prior weight initialisation.

4.4.2 Logistic regression

Logistic Regression was used with the default parameters.

4.4.3 Decision trees

Decision trees provide explainable modelling and are very fast to inference with. A decision tree with Gini entropy as the criterion with a maximum depth of 4 was constructed.

4.4.4 XGBoost

XGBoost with GPU-acceleration was used, the maximum depth was limited to 5, a forest was created from 20 trees.

4.4.5 CNNs

A basic convolutional neural network was implemented using TensorFlow. We intend to explore more on how the layers affect the model training and convergence

4.4.6 Transfer Learning

Some attempts were taken at transfer learning. We noticed that Xception performed reasonably well, coming close to the CNN, with 95% validation accuracy. A resizing layer was added to the model which upscaled the dimension of the image to 72x72 from 25x25 to be compatible with Xception.

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 25, 25, 16)        448

 max_pooling2d_4 (MaxPooling  (None, 9, 9, 16)         0
 2D)

 conv2d_5 (Conv2D)           (None, 9, 9, 32)          2080

 max_pooling2d_5 (MaxPooling  (None, 5, 5, 32)         0
 2D)

 flatten_2 (Flatten)         (None, 800)               0

 dense_6 (Dense)             (None, 512)               410112

 dropout_4 (Dropout)         (None, 512)               0

 dense_7 (Dense)             (None, 256)               131328

 dropout_5 (Dropout)         (None, 256)               0

 dense_8 (Dense)             (None, 2)                 514

=================================================================
Total params: 544,482
Trainable params: 544,482
Non-trainable params: 0
_________________________________________________________________
```

Figure 5. CNN model architecture

Accuracy			
Model	Training	Validation	Testing
Naive Bayes	0.639	0.640	0.632
Log. Reg.	0.750	0.709	0.707
Decision Trees	0.704	0.702	0.694
XGBoost	0.974	0.865	0.856
CNN	0.991	0.991	0.988
Trans. Learn.	0.970	0.948	0.953

Table 1. Accuracy

5. Results and Analysis

Below are prelimnary results on unaugmented datasets used for training across 6 different models. Hyperparameters were tuned based on the validation set, and the models tested on the test set. The results can be noted in tables 1, 2, 3 and 4, which correspond to accuracy, precision, recall and f1-score respectively.

6. Conclusion

We notice that the CNN model has the best performance when it comes to accuracy, precision, recall and f1-score and shows low bias and low variance. Meanwhile, the XGBoost model performs extremely well on the training set,

Precision			
Model	Training	Validation	Testing
Naive Bayes	0.685	0.683	0.677
Log. Reg.	0.763	0.716	0.720
Decision Trees	0.740	0.737	0.726
XGBoost	0.985	0.878	0.869
CNN	0.996	0.995	0.996
Trans. Learn.	0.971	0.946	0.955

Table 2. Precision

Recall			
Model	Training	Validation	Testing
Naive Bayes	0.512	0.529	0.511
Log. Reg.	0.730	0.698	0.679
Decision Trees	0.634	0.639	0.679
XGBoost	0.962	0.850	0.839
CNN	0.986	0.986	0.980
Trans. Learn.	0.970	0.952	0.952

Table 3. Recall

F1-Score			
Model	Training	Validation	Testing
Naive Bayes	0.591	0.596	0.583
Log. Reg.	0.746	0.707	0.699
Decision Trees	0.683	0.679	0.674
XGBoost	0.973	0.863	0.854
CNN	0.991	0.990	0.988
Trans. Learn.	0.971	0.949	0.953

Table 4. F1-score

but fails to perform as well on the validation and test sets, indicating high variance. The rest of the models: Naive Bayes, Logistic Regression and Decision trees portray high bias and low variance.

Image space transformations can accentuate features and we intend to explore more regarding that.

6.1. Remaining Tasks

We intend to evaluate models on images of varying sizes, from 25x25 to 100x100. We also aim to apply dataset augmentation techniques and nonlinear colour space transformations (eg. RGB to HSV) and determine their impact, while tuning the models further, implementing more architectures and determining optimal hyperparameters. Finally, we will attempt wholeslide image segmentation to isolate RBCs, if time permits.

6.2. Learnings

6.3. Individual Contributions

References

[1] W. H. Organization, *World malaria report 2021*. World Health Organization, 2021. p. 23 Death and Infection Statistics.

[2] C. for Disease Control and Prevention, "Cdc - malaria - diagnostic tools," 2020. Malaria Testing.

[3] N. L. of Medicine, "Thin smear single-cell dataset." Link 1; Link 2; Link 3. Dataset.

[4] F. KMF, T. JF, S. MRA, M. S, M. N, and R. T., "Deep learning based automatic malaria parasite detection from blood smear and its smartphone based application," *Diagnostics (Basel)*, vol. 10, no. 5, p. 329, 2020. Misclassified images.

[5] G. TensorFlow, "Imagedatagenerator." documentation.