Approach and Findings: Building Malaria Detection Model and Dashboard

1. Data Preparation:

I used a dataset containing images of malaria-infected and uninfected cells. The dataset was divided into two classes: "Parasitized" and "Uninfected." The images were preprocessed, resized to (33, 33), and converted to numpy arrays. Invalid files were handled using exception handling.

2. Model Architecture:

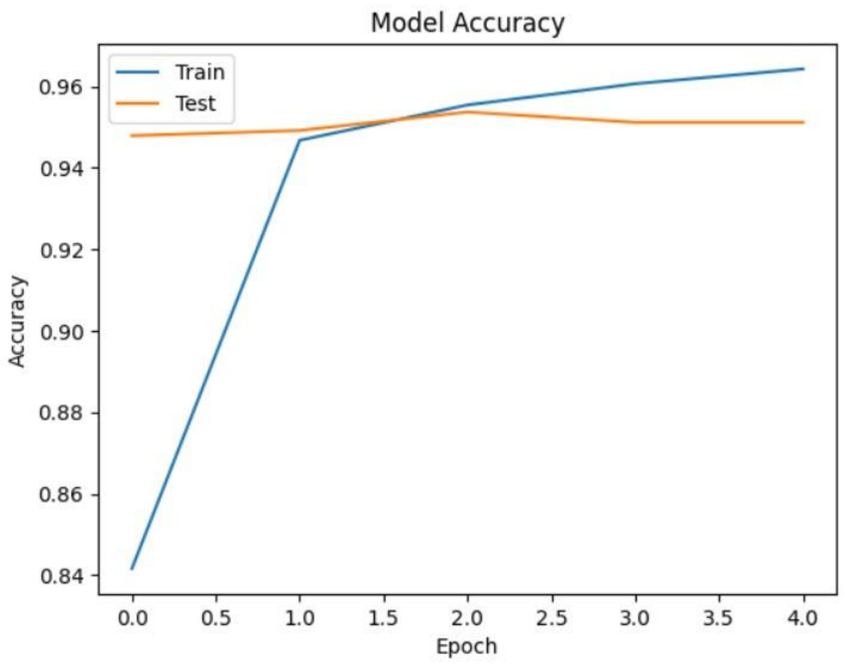
I created a Convolutional Neural Network (CNN) using TensorFlow and Keras. The model architecture included convolutional layers, max-pooling layers, flattening, and dense layers. The output layer had a softmax activation function since it was a classification problem. The model was compiled using sparse categorical cross-entropy as the loss function and the Adam optimizer.

3. Training the Model:

The model was trained on the prepared dataset. The training was done for 5 epochs with a batch size of 10. Training and validation accuracy and loss were monitored during the training process.

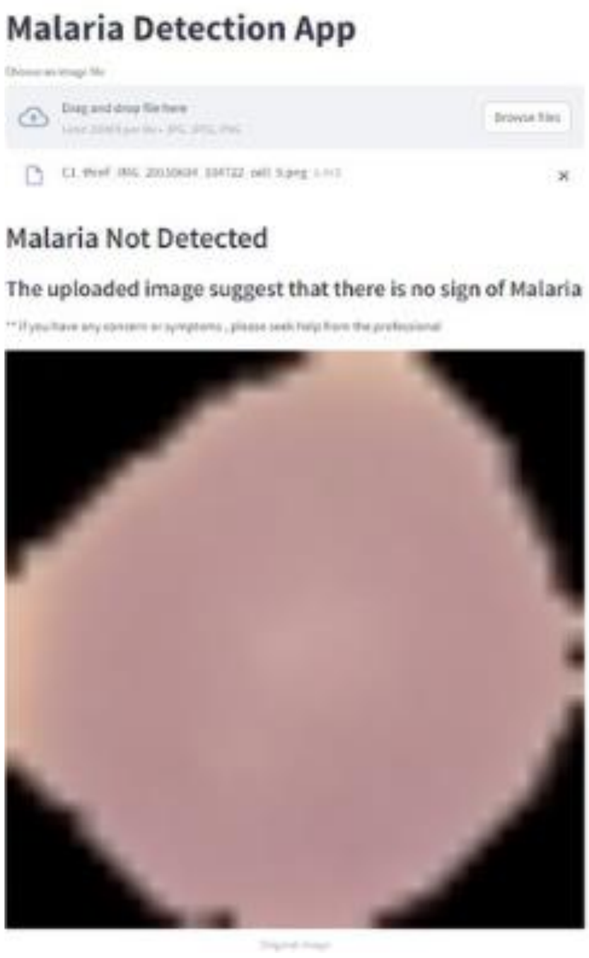
4. Model Evaluation:

After training, the model was evaluated on the test set. The accuracy achieved on the test set was approximately 95.12%.



5. Streamlit Dashboard:

I created a Streamlit app to deploy the trained model. The app allows users to upload an image, and the model predicts whether malaria is detected or not. The original image is displayed along with the prediction.



Malaria Detection App

Choose an image file

Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG

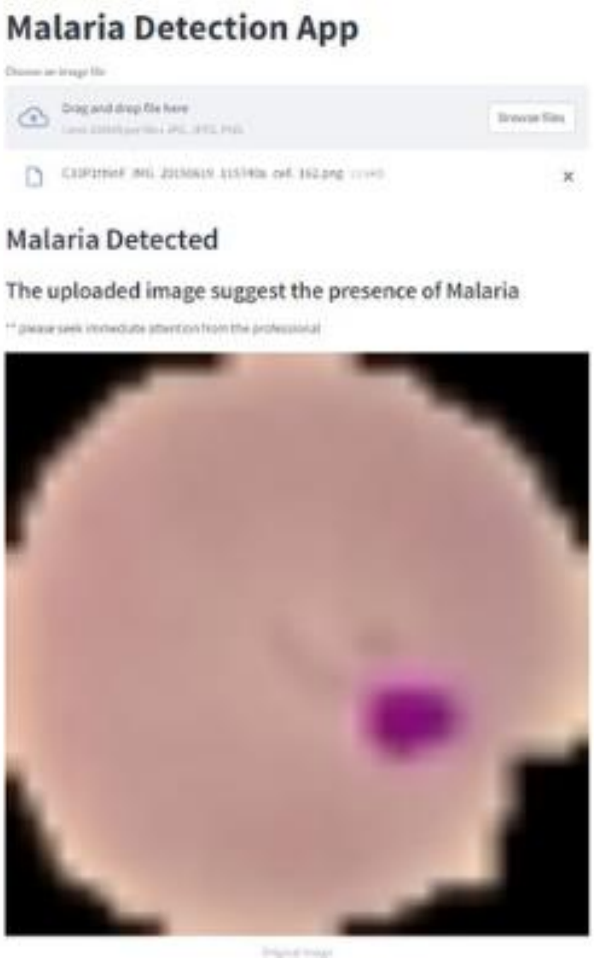
C1_thinF_IMG_20150604_104722_cell_9.png

Malaria Not Detected

The uploaded image suggest that there is no sign of Malaria

** If you have any concern or symptoms , please seek help from the professional

Original image



Malaria Detection App

Choose an image file

Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG

C33P1thinF_IMG_20150619_115740a_cell_162.png

Malaria Detected

The uploaded image suggest the presence of Malaria

** please seek immediate attention from the professional

Original image

6. Streamlit App Code:

The Streamlit app consists of code that loads the trained model, allows users to upload an image, preprocesses the image, and uses the model to make predictions. Depending on the prediction, the app displays appropriate messages and the original image.

7. Results:

The app successfully predicts whether an uploaded cell image suggests the presence of malaria or not. It provides clear messages and encourages users to seek professional attention if malaria is detected.

8. Improvement Opportunities:

- Fine-tuning the model by adjusting hyperparameters or trying different architectures.
- Exploring other techniques for model interpretability to enhance understanding of predictions.
- Enhancing the UI/UX of the Streamlit app for a better user experience.

Overall, the approach involved standard steps in building and deploying a deep learning model using AI tools(ChatGPT, Google Bard) for image classification, and the Streamlit app provides an accessible interface for users to interact with the model.

9. Conclusion:

The project successfully culminated in the development of a deep learning model for malaria detection and its deployment in a Streamlit app. The model's accuracy and the user-friendly dashboard contribute to its potential utility in real-world applications. Further refinements and continuous monitoring will ensure its relevance and effectiveness in the field of medical image analysis.