Malaria Classification using Custom CNN

1. Introduction

Malaria detection using deep learning has significantly advanced the field of medical image analysis. This project implements a Convolutional Neural Network (CNN) to classify malaria-infected cells from uninfected ones using grayscale microscopy images. The goal is to achieve high classification accuracy through a custom CNN architecture and hyperparameter tuning.

2. Dataset Details

- Dataset Type: Microscopy Images
- Classes: 2 (Parasitized 🦠, Uninfected ✅)
- Total Images:
  - Training Set: 27,558 images
  - Testing Set: 15,832 images
- Image Size: 224x224 (grayscale, single-channel)
- Preprocessing:
  - Rescaling (Normalizing pixel values between 0 and 1)
  - Augmentation (Rotation, Brightness, Shear Transformations, Zoom)

3. Model Architecture

This model utilizes a custom CNN architecture designed for feature extraction and binary classification.

3.1 CNN Layers and Parameters:

- Conv2D Layer 1: 32 or 64 filters, kernel size 3x3, ReLU activation, Batch Normalization, MaxPooling2D

- Conv2D Layer 2: Tunable (32, 64, or 128 filters), kernel size 3x3, ReLU activation, MaxPooling2D

- Conv2D Layer 3 (Optional): Applied based on Hyperparameter tuning

- Global Average Pooling Layer

- Dense Layer: 128 to 512 neurons (ReLU activation)

- Dropout Layer: 0.3 to 0.5 to prevent overfitting

- Output Layer: 1 neuron (Sigmoid activation for binary classification)

3.2 Model Compilation Settings:

- Optimizer: Adam (`learning_rate = 5e-5`, adjusted with `ReduceLROnPlateau`)

- Loss Function: Binary Crossentropy

- Metrics: Accuracy

4. Training Process

4.1 Data Augmentation Applied

- Rotation: ±30 degrees

- Brightness Adjustment: Range 0.5 to 1.5

- Zoom & Shift Transformations: Applied for better generalization

- Shear Transformations: Used to add variation

4.2 Training Configuration

- Batch Size: 8, 16, or 32 (Tuned)

- Epochs: 25-30

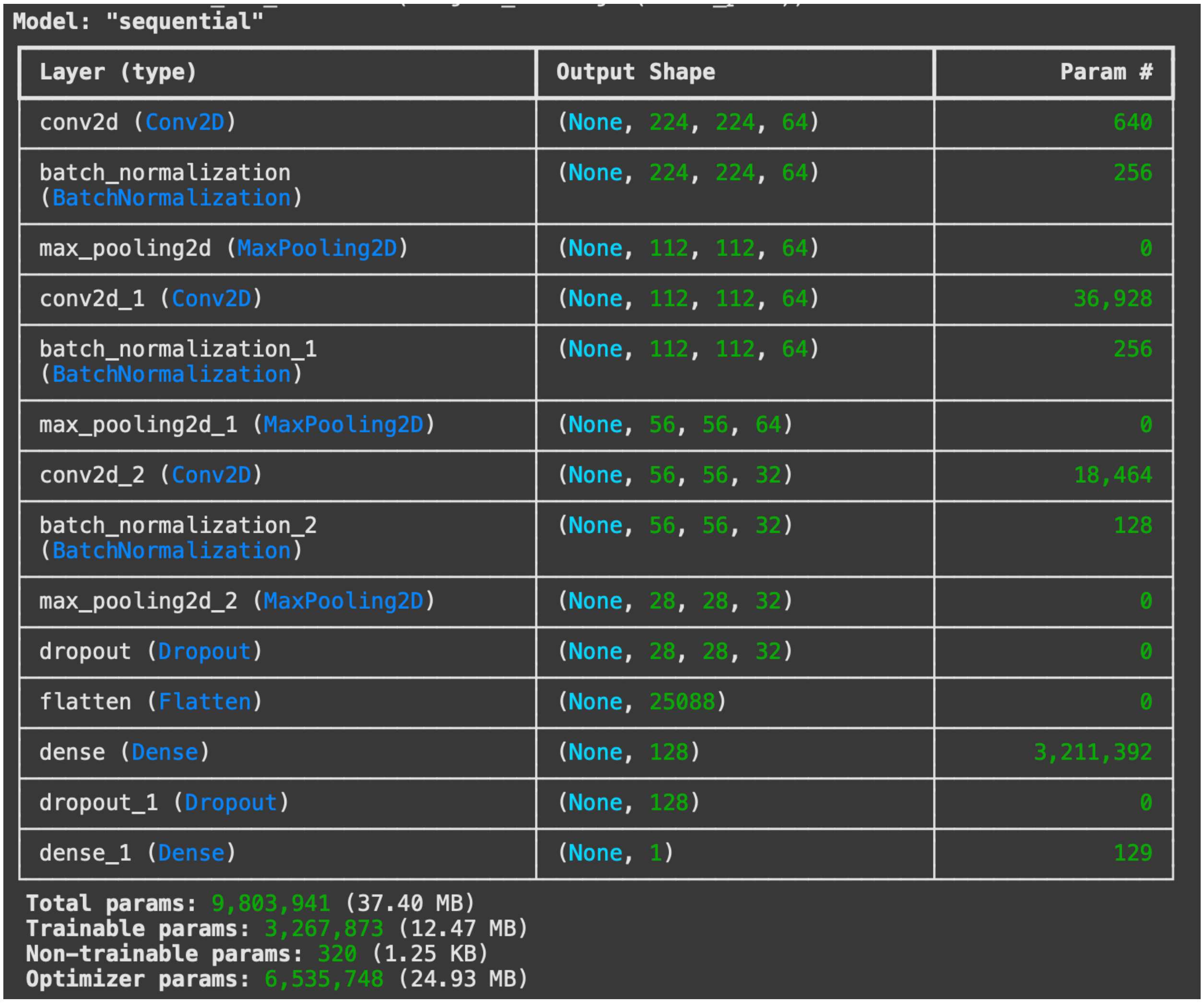
- Class Weights: Adjusted to balance the dataset

- Early Stopping: Enabled (patience = 5 epochs)

- ReduceLROnPlateau: Reduce LR by 0.5 when validation loss stagnates

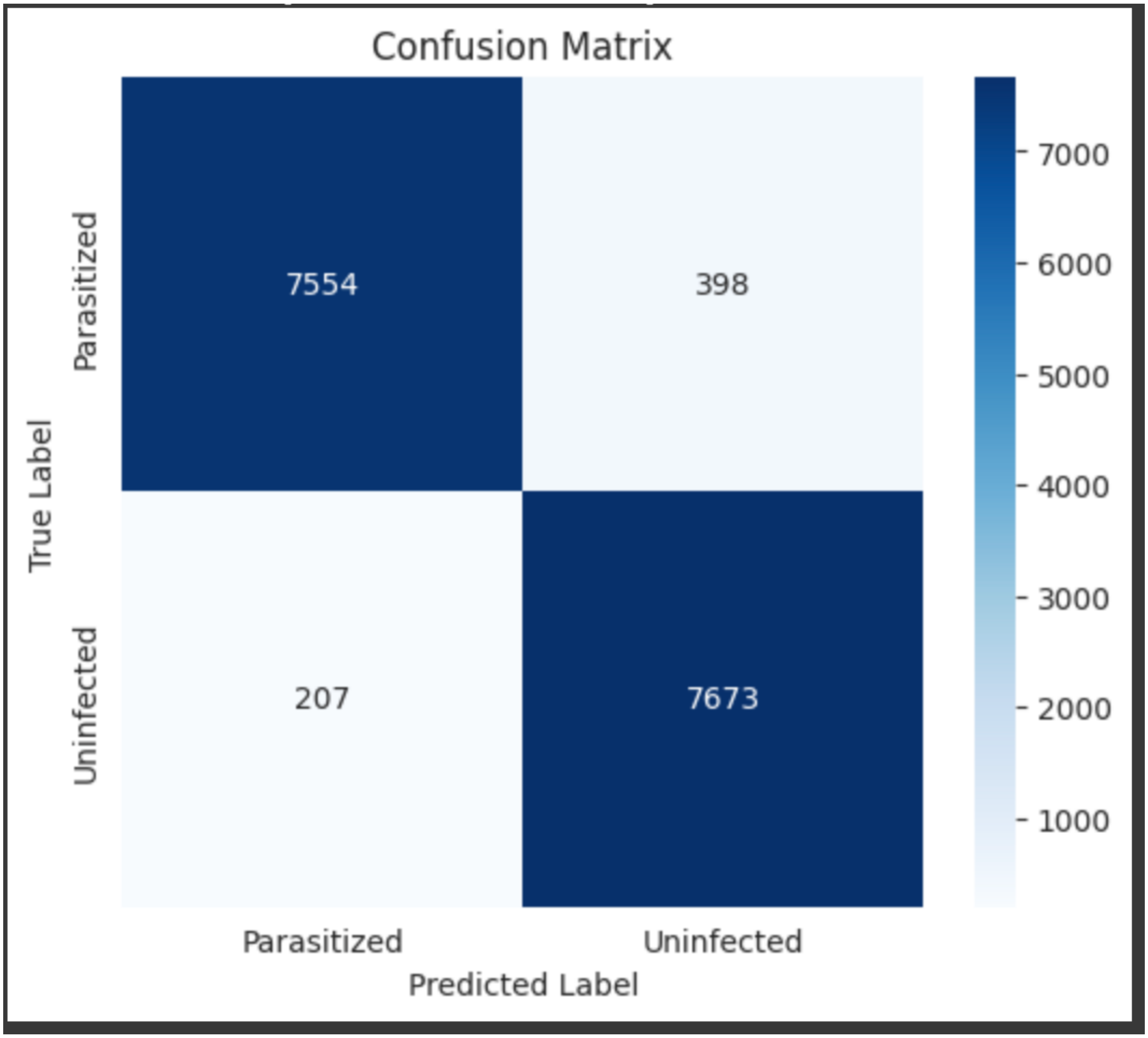
5. Results and Evaluation

5.1 Model Summary

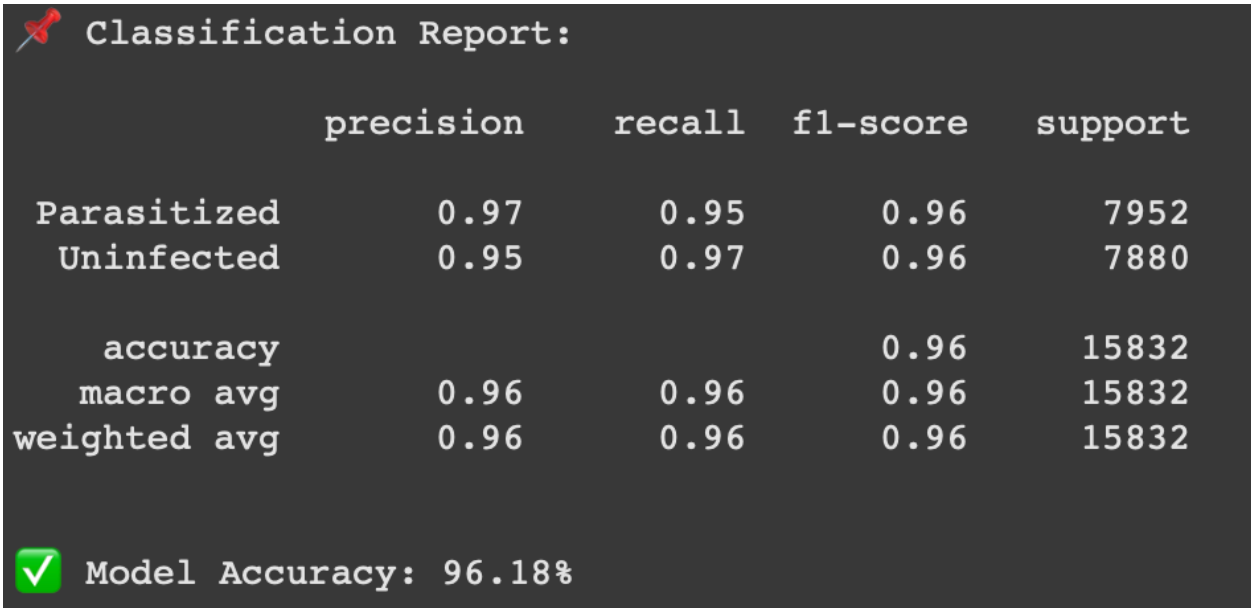


```
Model: "sequential"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┓
┃ Layer (type)                         ┃ Output Shape                ┃         Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━┩
│ conv2d (Conv2D)                      │ (None, 224, 224, 64)        │             640 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ batch_normalization                  │ (None, 224, 224, 64)        │             256 │
│ (BatchNormalization)                 │                             │                 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ max_pooling2d (MaxPooling2D)         │ (None, 112, 112, 64)        │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ conv2d_1 (Conv2D)                    │ (None, 112, 112, 64)        │          36,928 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ batch_normalization_1                │ (None, 112, 112, 64)        │             256 │
│ (BatchNormalization)                 │                             │                 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ max_pooling2d_1 (MaxPooling2D)       │ (None, 56, 56, 64)          │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ conv2d_2 (Conv2D)                    │ (None, 56, 56, 32)          │          18,464 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ batch_normalization_2                │ (None, 56, 56, 32)          │             128 │
│ (BatchNormalization)                 │                             │                 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ max_pooling2d_2 (MaxPooling2D)       │ (None, 28, 28, 32)          │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ dropout (Dropout)                    │ (None, 28, 28, 32)          │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ flatten (Flatten)                    │ (None, 25088)               │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ dense (Dense)                        │ (None, 128)                 │       3,211,392 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ dropout_1 (Dropout)                  │ (None, 128)                 │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ dense_1 (Dense)                      │ (None, 1)                   │             129 │
└──────────────────────────────────────┴─────────────────────────────┴─────────────────┘
 Total params: 9,803,941 (37.40 MB)
 Trainable params: 3,267,873 (12.47 MB)
 Non-trainable params: 320 (1.25 KB)
 Optimizer params: 6,535,748 (24.93 MB)
```

5.2 Confusion Matrix Analysis

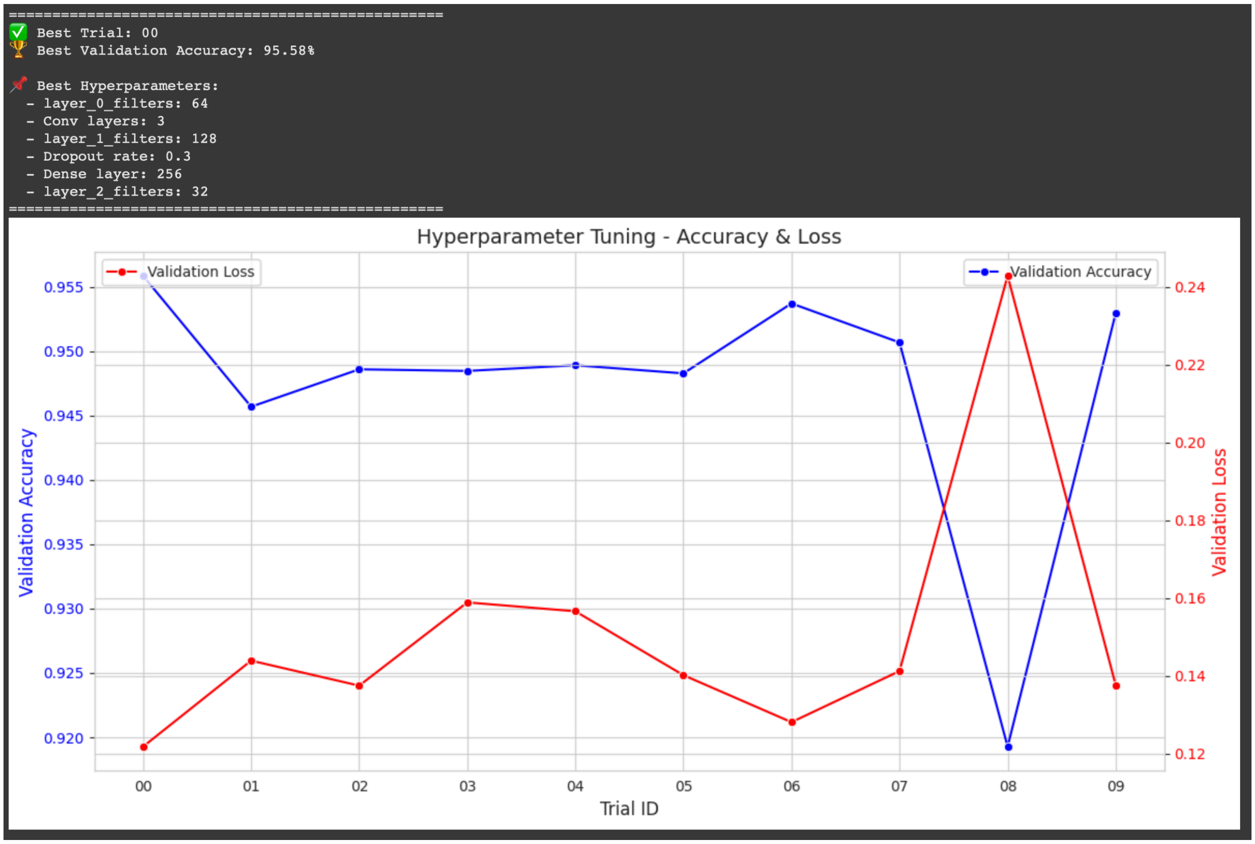


5.3 Classification Report and Final Model Accuracy



```
📌 Classification Report:

              precision    recall  f1-score   support

 Parasitized       0.97      0.95      0.96      7952
  Uninfected       0.95      0.97      0.96      7880

    accuracy                           0.96     15832
   macro avg       0.96      0.96      0.96     15832
weighted avg       0.96      0.96      0.96     15832


✅ Model Accuracy: 96.18%
```

5.4 Best Trials and Hyperparameters over-time and Accuracy Loss Graph



```
==================================================
✅ Best Trial: 00
🏆 Best Validation Accuracy: 95.58%

📌 Best Hyperparameters:
  - layer_0_filters: 64
  - Conv layers: 3
  - layer_1_filters: 128
  - Dropout rate: 0.3
  - Dense layer: 256
  - layer_2_filters: 32
==================================================
```

6. Conclusion

This deep learning model successfully classifies malaria-infected cells with high accuracy. The key factors contributing to its success include:

- Carefully designed CNN architecture with Batch Normalization
- Fine-tuning hyperparameters for optimal performance
- Data augmentation to improve generalization
- Learning rate optimization with ReduceLROnPlateau

Future Work & Improvements

- Test model on external malaria datasets for robustness
- Deploy as a web-based or mobile application
- Experiment with deeper architectures or ensemble models for further improvements

7. References

https://github.com/aryannnn07/DS_Assignment.git

Streamlit app: app.py

Jupyter Notebook: Aryan_ML.ipynb

Model: malaria_best_model.keras