

MALARIA

Introduction

Malaria is an infectious disease caused by protozoan parasites of the genus plasmodium and it is a major global health threat.

The standard way of diagnosing malaria is by visually examining blood smears for parasite-infected red blood cells under the microscope by qualified technicians.

This method is inefficient, and the diagnosis depends on the experience and the knowledge of the person doing the examination, hence Convolutional Neural network gives the best output. The aim of the collection of the dataset was to reduce the burden for microscopists in resource-constrained regions and improve diagnostic accuracy using an CNN algorithm to detect and segment the red blood cells.

Why CNN?

The Convolutional Neural Networks (CNNs) have been used to classify malaria parasites from blood smear images automatically and successfully gave a good result.

The use of CNN as a feature extractor shows better performance than transfer learning.

Data source:

We used archived images acquired from Kaggle Malaria classification dataset where it has nearly 28000 train and test data.

[Malaria Cell Images Dataset \(kaggle.com\)](https://www.kaggle.com/datasets/fergusmcdonald/malaria-cell-images-dataset)

SPECIFICATIONS:

IDE: Google Collab

GPU: 2.8.0

Important libraries: TensorFlow, keras, matplotlib, streamlit, h5py, pandas, NumPy,

Deployment: Streamlit, VS code

Accuracy with different epochs value:

EPOCHS	ACCURACY (%)	LOSS (%)	VALIDATION ACCURACY (%)	VALIDATION LOSS (%)
2	93.66	19.61	93.27	20.47
5	93.97	18.61	93.56	18.51
10	91.56	17.02	93.77	16.99
12	91.11	16.89	93.98	16.01

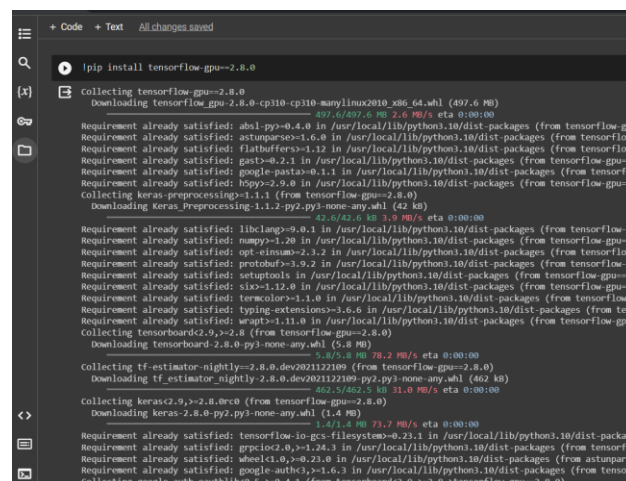
Optimiser:

The optimiser that is used is called Adaptive Moment Estimation (Adam) which combines ideas from both RMSProp and Momentum. It computes adaptive learning rates for each parameter.

EPOCHS:

```
Epoch 1/2
<ipython-input-13-fdda96df0b73>:1: UserWarning: 'Model.fit_generator' is deprecated and will be removed in a future version. Please use 'Model.fit', which supports generators.
  history = model.fit_generator(generator = train_datagen,
689/689 [=====] - 385s 558ms/step - loss: 0.2228 - accuracy: 0.9298 - val_loss: 0.2047 - val_accuracy: 0.9332
Epoch 2/2
689/689 [=====] - 386s 560ms/step - loss: 0.1961 - accuracy: 0.9366 - val_loss: 0.2072 - val_accuracy: 0.9327
```

GOOGLE COLLAB:



```
+ Code + Text All changes saved

!pip install tensorflow-gpu==2.8.0

Collecting tensorflow-gpu==2.8.0
  Downloading tensorflow-gpu-2.8.0-cp310-cp310-manylinux2010_x86_64.whl (497.6 MB)
    Requirement already satisfied: absl-py>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: flatbuffers>=1.12 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: gast>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Collecting keras-preprocessing>=1.1.1 (from tensorflow-gpu==2.8.0)
    Downloading keras_preprocessing-1.1.2-py3-none-any.whl (42 kB)
    Requirement already satisfied: libclang>=9.0.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: protobuf>=3.9.2 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Collecting tensorflow>=2.8.0 (from tensorflow-gpu==2.8.0)
    Downloading tensorflow-2.8.0-py3-none-any.whl (5.8 MB)
    Collecting tf-estimator-nightly==2.8.0.dev2021122109 (from tensorflow-gpu==2.8.0)
    Downloading tf_estimator_nightly-2.8.0.dev2021122109-py3-none-any.whl (462 kB)
    Collecting keras>=2.8.0 (from tensorflow-gpu==2.8.0)
    Downloading keras-2.8.0-py3-none-any.whl (1.4 MB)
    Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: grpcio>=2.0.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: wheel<1.0, >=0.23.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Requirement already satisfied: google-auth>=1.6.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow-gpu==2.8.0)
    Collecting google-auth-oauthlib>=0.4.1 (from tensorflow-gpu==2.8.0)
```

VISUAL STUDIO:

```
streamlit(app).py 6 X #Title Untitled-1 9+ ● img = get_img_as_base64("image.jpg") Untitled-2 1 ● streamlit.ipynb
C:\> Users > user > Desktop > streamlit(app).py > ...
1  #importing libraries
2  import streamlit as st
3  import tensorflow as tf
4  import numpy as np
5  from PIL import Image, ImageOps
6
7  #Title
8  st.write("""
9      # Malaria Cell Classification
10     """)
11
12
13  #upload file
14  upload_file = st.sidebar.file_uploader("Upload Cell images", type = "png")
15
16  #side button
17  Generate_pred = st.sidebar.button("Predict")
18
19  #importing the model
20  model = tf.keras.models.load_model("C:/Users/user/Downloads/malaria_det.h5")
21
22  #conditions
23  def import_n_pred(image_data, model):
24      size = (128, 128)
```

DEPLOYMENT:



CONCLUSION:

Malaria cell image classification uses deep learning faster than the most traditional ML models. With Streamlit, the prediction results are presenting good results.