MALARIA CELL DETECTION

Aim: To create a web app which will check whether the cell is infected or not when an image of cell is feed into it.

Step 1: Gathering the relevant Data

Data is downloaded from the Kaggle. Name of dataset is Malaria Cell Images dataset which has been uploaded by user Arunava on Kaggle for public use. Link to the data is: Malaria Cell Images Dataset | Kaggle. Dataset has total of 27.6K images. Half of them is of parasitized cells and other half is of uninfected cells.

Step 2: Creating and Training a deep learning model for classification

A very simple deep learning has been created using keras sequential API which has around 3 convolution layers and 2 dense layers. Summary of model is:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 148, 148, 16)      448
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 74, 74, 16)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 72, 72, 32)        4640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 36, 36, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 34, 34, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 17, 17, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 18496)             0
_________________________________________________________________
dense (Dense)                (None, 512)               9470464
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 1026
=================================================================
Total params: 9,495,074
Trainable params: 9,495,074
Non-trainable params: 0
_________________________________________________________________
```

Before training, data is segregated into 3 parts for training, validation and testing. Ratio of segregation is 90:5:5 i.e., 90% of data is kept for training purpose and remaining 10% is divided equally for validating and testing.

Directory Structure:

1. tmp
    a. testing
        i. infected
        ii. uninfected
    b. training

   i. infected
   ii. uninfected
c. validation
   i. infected
   ii. uninfected
2. model.py
3. app.py
4. malaria_v2.h5

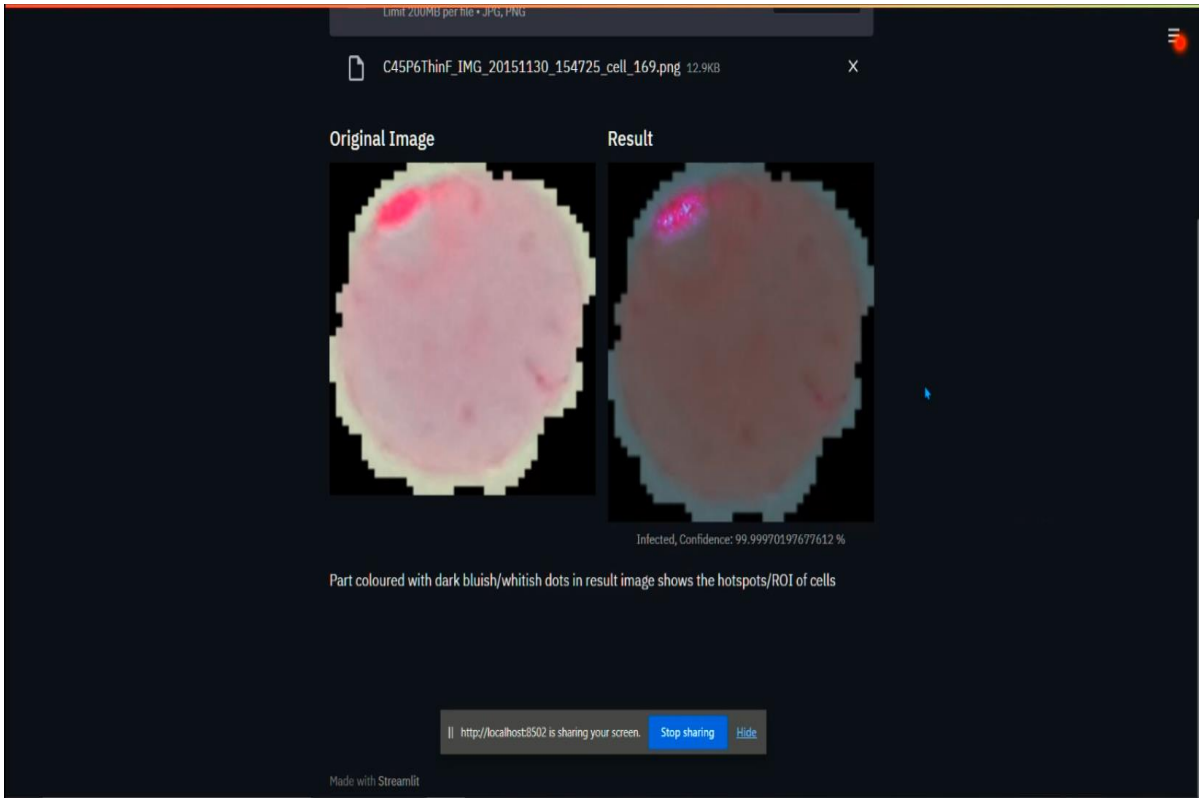
Model has been then trained for 2 epochs and achieved an accuracy of 90.21% on training data and an accuracy of 94.63% on validation data. For achieving better accuracy, model can be trained further, and different and more complex model can be tried. But training becomes a very lengthy process in that case if you do not have GPU. That's why I stopped training after 2 epochs only.

After that I evaluated the model on test dataset. On test dataset Model has an accuracy of around 94.99 % which is not that bad considering the model size and training time.

Step 3: Creating Streamlit App

Model that I trained in last step is used in creating this app. All that a user have to do is to upload an image of cell. App will check whether the cell is infected or not and will display the result along with the confidence score. App will also display an image which will have Region of Interest/Hotspots of cell highlighted.

Below is the screenshot of App:



In the we can see two images 1st is original and 2nd is result image. In the result, we can see an area in cell highlighted with bluish/whitish dots which indicates the ROI/Hotspots of cell.

Conclusion: A full-fledged working app has been designed to detect the cell infected with Malaria and to highlight the ROI/Hotspots of cells.

Tools and Technologies used are:

1) Python (Programming Language), Version: 3.8.5
2) TensorFlow (A deep learning framework for Python), version: 2.2.0
3) OpenCV (A python library for images and videos), version: 4.4.0
4) Streamlit (A python library to create web app easily), version: 0.8.6