Malaria Detection Task Report

Name: Karan KS

Dataset: https://www.kaggle.com/iarunava/cell-images-for-detecting-malaria

Abstract: To create a Deep learning model which classifies the given image between Infected or Uninfected and Deploy that model in a streamlit web app.

Libraries used:

★ Numpy
★ Matplotlib
★ Streamlit
★ BytesIO
★ Glob
★ Request
★ PIL
★ Tensorflow.keras.layers: Input, Lambda, Dense, Flatten,Conv2D, MaxPooling2D
★ Tensorflow.keras.models: Model, load_model, Sequential
★ Tensorflow.keras.applications.vgg19: VGG19
★ Tensorflow.keras.applications.resnet50: preprocess_input
★ Tensorflow.keras.preprocessing: image
★ Tensorflow.keras.preprocessing.image: ImageDataGenerator, load_img

Approach: Deep Learning

Github Link: https://github.com/karan00713/IQgateway.git

Deployment Platform: StreamLit Web

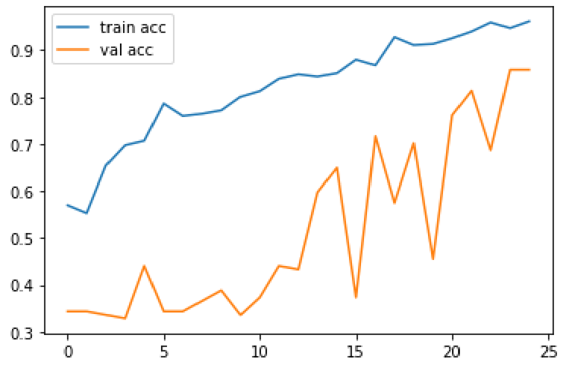
Python files: malaria.ipynb, app.py

Command to Launch Streamlit app (Anaconda prompt): streamlit run app.py

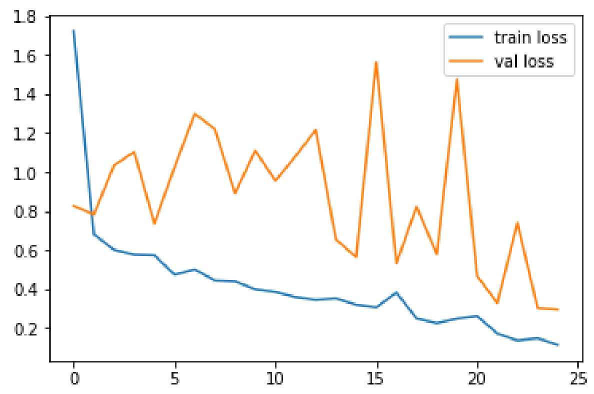
Details: The Dataset was huge (Nearly 13000 images in each set) and takes lots of computing power & time to train the model, So the dataset was reduced into a sample dataset for training. For better accuracy, we tried two kinds of pre-trained models namely VGG19 & Resnet50, but VGG19 gave better results among them, so we had done transfer learning from the VGG19 model which is pre-available in TensorFlow.Keras.applications and we blocked all other nodes of VGG19 except for the last node, so we can use it on our model while classifying the images, there are a total of 25 epochs done to train models with a batch size of 32 images, And we used sequential model for this network, with 3 pair of Convolution 2D layer & Maxpooling Layer, Single Flatten Layer, Two Dense layers. Because of this Image classification, we used two neurons at the end node with softmax activation. By testing the model with test data. ImageDataGenerator is used while training the models, it would help the model to better understand the image in various views.

Findings:

Training Accuracy and Validation Accuracy



Training Loss and Validation Loss



We have got - accuracy: 0.9609, loss: 0.1086. while observing the chat, there is a large difference between the Training and Validation makes models need to be trained with the more clear dataset. To get better validation accuracy, this will give a quite significant result.