ANDROID IMAGE CLASSIFICATION

Krista Chanaritthichai, Daniel Roberts, Guillermo Romero, Dat Tran, Alex Turk, Sneha Venkatesh
Advisors: Dr. Vladimir Dragovic, Dr. Branislav Kisacanin

STAT/CS/MATH 4475 / Spring 2021
The University of Texas at Dallas

Abstract

Image classification is a growing field in machine learning which can be utilized by anyone using modern technology. The objective of the project is to classify routine objects using an Android device and TensorFlow. By using TensorFlow, a popular deep learning development framework, and Android Studio we are able to repurpose the dataset and convert it into a form that can efficiently run on an Android. The application was designed by the team to run on the latest and older versions of Android SDK 8.1-11

Architecture

Google Colab: Train a classifier to recognize objects using TensorFlow Lite transfer learning

TensorFlow Lite: An open-source deep learning framework to run TensorFlow models on-device. With an object detection model, TensorFlow models detect objects in real-time from a camera feed.

Android Studio: Build and run the app on the real device using Android Studio (4.1 or above)

Canva: A design software that lets us design and create the UI elements for the app such as the logo

XML layout file:
Customize and modify the user interface in XML files.

Git: Merge all the code into the master branch.

Models

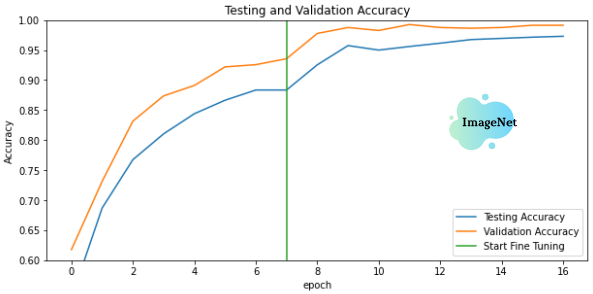
Four Models
EfficientNet used in 3 base models : ImageNet_EN1, ImageNet_EN2,Insect_EN
MobileNet used in 1 base model: ImageNet_MN

Efficient and MobileNetv2 (Refined)
- Build an input pipeline, using Keras ImageDataGenerator (rescale the images)
- Compose the base model
- Load in the pre-trained base model/weights
- Add a classifier on top and train top-level classifier
- Compile and Train model
- Unfreeze top layers of the base model
- Train weights of pre-trained base
- Compile model (keras.optimizer lower training rate)
- Train and then Evaluate model

Insect Model (Not Refined)
- Build an input pipeline, using Keras ImageDataGenerator (rescale the images.)
- Compose the base model (Efficient Net)
- Load in the pre-trained weights/base model
- Add a classifier on top and train top-level classifier
- Compile model
- Train the model
- Evaluate the model

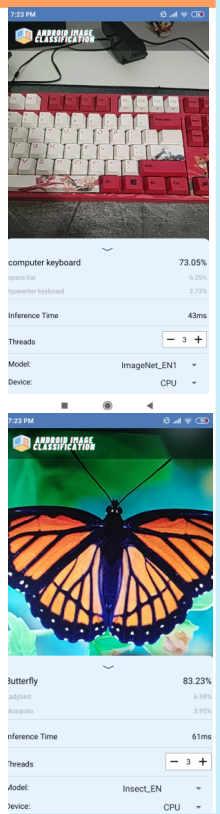
Results/Figures

EfficientNet
- Biggest .tflite size (25MB) and inference time but higher accuracy

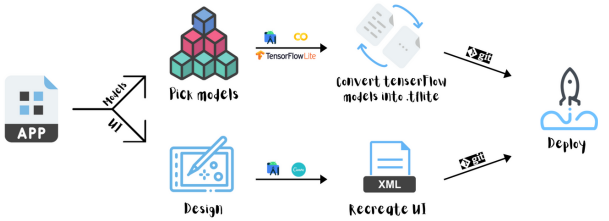


MobileNet
- Smallest .tflite size (8.54MB) and inference time

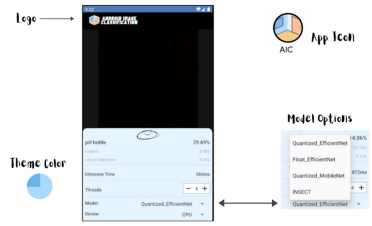
Insect model
- Fastest inference time because it only has 5 classes.



Implementation process



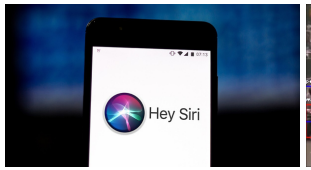
User Interface



Summary/Use Cases

We learned
- Use of Tensorflow
- In's and out of Android Studio
- Change the User Interface
- Emulator vs Android Device
- Deploy an App

Future Use Cases
- Voice/Sound recognition
- Text-based applications
- Image recognition,
- Time-series
- Video detection
- Sentiment Analysis



References

Ali, H. (2020, September 16). Insects Recognition. Kaggle. https://www.kaggle.com/hammaadali/insects-recognition.

CS231n: Convolutional Neural Networks for Visual Recognition. Stanford University CS231n: Convolutional Neural Networks for Visual Recognition. (n.d.). http://cs231n.stanford.edu/.

Google. (n.d.). Recognize Flowers with TensorFlow Lite on Android | Google Codelabs. Google. https://codelabs.developers.google.com/codelabs/recognize-flowers-with-tensorflow-on-android/#0.