Deploying Image Classification Model on Google Pixel Device

Vishnuvardhan Janapati & Yasir Modak
07/14/2020

TABLE OF CONTENTS

1.0 Objective

Objective of the work is to develop a classification model to classify an image into one of the five categories of flowers (daisy, tulip, rose, sunflower, dandelion) using a mobile.

For more details on the classification, check our **presentation**.

2.0 Data for Model

We used **Flowers Recognition dataset from Kaggle** to build an image classification model and deploy on an android operating system. The dataset is fairly clean and contains 5 classes: daisy, tulip, rose, sunflower, dandelion.

ImageDataGenerator class was used to create artificial noise in the dataset to curb overfitting of our model and perform better on the unknown data.

3.0 Model development

Further we used **pre trained MobileNetV2** architecture as our base model and performed training from scratch to make our model learn images specific to 5 classes. The trained model was tuned for achieving higher performance.

The training was done using gpu and training loss was visualized using tensorboard.
In the **tf.keras.models.fit()** method we used callbacks argument to set tensorboard parameters.

After training we save the model in multiple formats such as **h5** and **tf** format.

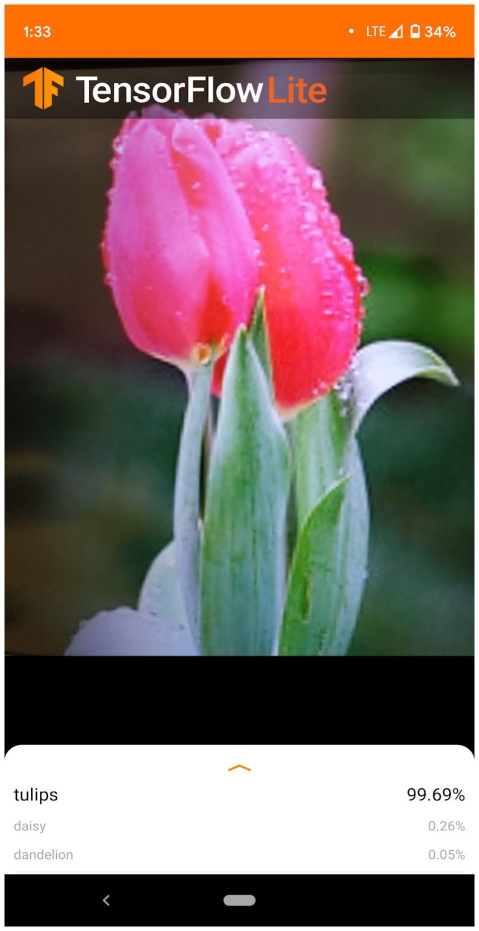
4.0 TFLite Converter

As our goal is to deploy this model on a mobile device, we need to convert the TensorFlow model to a TFLite model. **TF.LiteConverter** was used to convert the keras model into tf lite format for further deployment in an android platform.

During the conversion, quantization techniques optimize the model while reducing the size of the TensorFlow model.

After converting the model to tflite flat buffer format, the model was deployed on a Google Pixel phone using Android Studio.

During the inference, the deployed model accesses the real-time image from the camera and runs the tflite model to classify the image and display the classified flower type and the predicted accuracy of the model. Screenshot of a model prediction is shown below.



tulips	99.69%
daisy	0.26%
dandelion	0.05%

5.0 Conclusions

A pretrained model was adapted for classifying 5 target labels, trained from scratch, and fine tuned the model for higher performance. The TF model was converted into TFLITE format and deployed on Google Pixel device through Android Studio. Finally, the application was successfully demonstrated.

We can improve the model by

- tuning ImageDataGenerator even further
- Add more classes (currently 5)
- Reduce the inference time by using a fully quantized or quantization aware model.

6.0 Resources

1. Our Model repo https://github.com/jvishnuvardhan/sample-models
2. Our presentation
3. https://www.tensorflow.org/lite/guide/get_started