School of Computing and Augmented Intelligence (SCAI)
CSE 535 Mobile Computing
Teaching Assistant: Aranyak Maity
Instructor: Ayan Banerjee

ASSIGNMENT 1

Android application to click a picture and upload to a local Server.

Project Group 20

- Purna Venkatesh Peddireddy (1223526954)
- Vijay Maddineni (1222982943)
- Prasidh Aggarwal (1225362125)
- Mohit Suresh Ahuja (1225915823)
- Phanindra Pabba (1226585219)

Date: September 26, 2022

Project Overview

This project consists of 3 major parts. Application, Server, and the TensorFlow Model.

- Application part is developed using Android Studio and Java. It comprises all the major work being done in the application starting from the application startup, all the way to clicking the images, and to upload them to the classified digit.
- Server is developed using vsCode and nodeJs. It holds the local directories and the individual folders where the clicked pictures are uploaded.
- ML model is created using tensorFlow and Keras on Google colab notebooks
- User clicks image of a handwritten digit and the app uses the ML model to classify it and store it in it's own directory after user clicks upload.

Environment Setup

The environment involved the use of two IDE's:

- Android Studio (for Java)
- VsCode (for nodeJs)
- Installed nodeJs on Bash.
- Gradle was used for easily packaging the dependencies in Android Studio.
- Google Colab for building, testing, and training the model. (Using Keras, TensorFlow, MatplotLib etc.)

Tasks

Major tasks Included:

- Remove the selectCategory spinner and also the selectCategory method as it is no longer needed. (UI change)
- Capture the image with phone camera and preview it as usual. After that, modify the upload button to now trigger our digit classification mechanism. (UI change)
- Load the model's tflite file under the assets directory, along-with the gradle dependency for TFLite.(Package change)
- Initialize the Digit detector class when upload is clicked, which loads the model file into a TFLite Interpreter.

- Call the detectDigit method which will first preprocess (resize, center-crop, greyscale) the bitmap (original image) and then convert the output of pre-processing to a ByteBuffer, so that we can pass it as an input to our model.
- Pass the output of pre-processing to our Interpreter (which has loaded the model file), which then stores the output into a float[][] array.
- From the output of model, check which index (digit) has the max confidence of being matched, and return that digit.
- Use this digit to display a toast notification to the user saying which digit was uploaded to the server.
- Make an async call to the NodeJs server to upload the clicked image to the directory with the same name as the digit.

Lessons Learnt

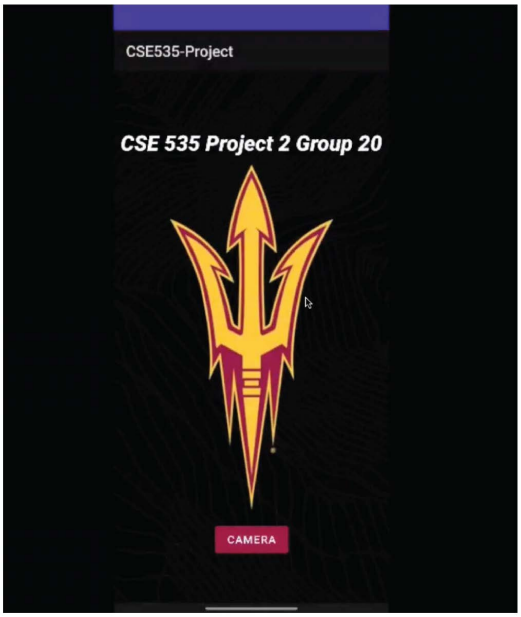
- Understood how to create a model and fine tune the weights and accuracy using Keras and TensorFlow.
- Learnt how to integrate the TensorFlow Lite models with our android application.
- Worked in a team and built the application using a SCRUM model to distribute tasks equally.
- Always account for alignment of the ~~test~~ data and the actual possible real-world data.

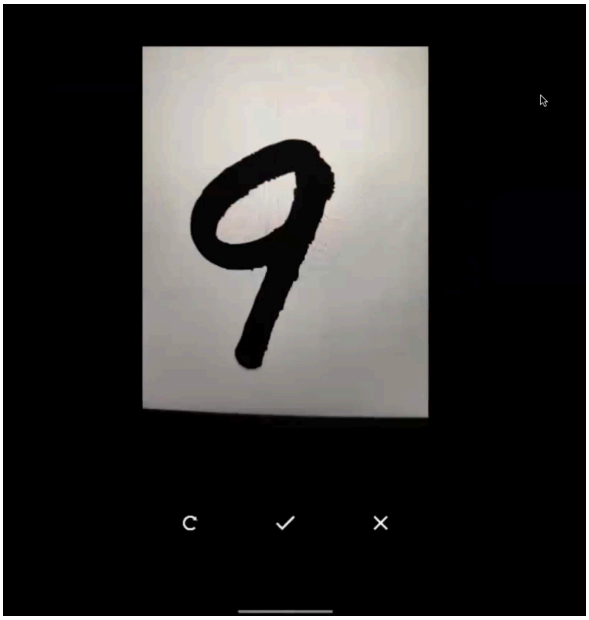
Individual Member Work

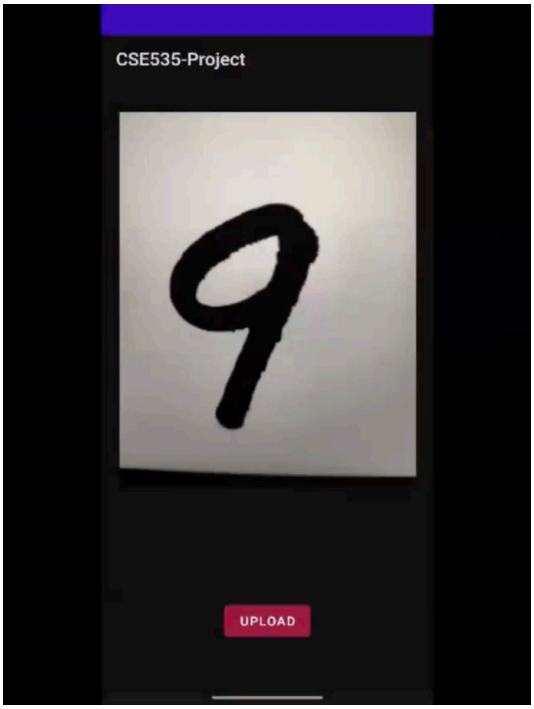
These are the ball-park task numbers even though everyone worked together to get the application running and helped wherever possible.

- Purna Venkatesh Peddireddy(6 Tasks 19-24)
- Vijay Maddineni(6 Tasks 1-6)
- Prasidh Aggarwal(6 Tasks 13-18)
- Mohit Suresh Ahuja(6 Tasks 7-12)
- Phanindra Pabba(1 Task 25)

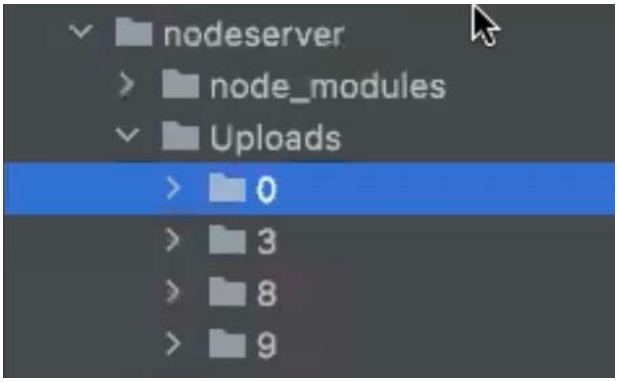
Application Working


CSE535-Project
CSE 535 Project 2 Group 20
CAMERA




CSE535-Project
UPLOAD


CSE535-Project
CSE 535 Project 2 Group 20
Successfully uploaded 9
CAMERA


nodeserver
node_modules
Uploads
0
3
8
9