Ira A. Fulton Schools Of Engineering — School Of Computing And Augmented Intelligence
Arizona State University

CSE 535 Mobile Computing Project 3 (Fall 2022)

Aatish Dineshbhai Chaudhari
Drishti Vijaybhai Patel
Harsh Sanjaykumar Nagoriya
Mohak Sanjay Acharya
Rahul Ashwin Sheth

Tempe, Arizona December 5, 2022

Contents

1 Problem Statement

Create an Android Application to divide the images into parts and send the handwritten digit pictures to the client mobiles and use client mobiles to classify the digits. The digits should be placed in their respective folders after the classification of the images on the master mobile.

The project has two parts,
1) Modify the Mobile Application used in Assignment 2 to divide the image into 4 parts and send each part to different client mobile.
2) Train a basic deep-learning framework from scratch on different parts of the images in MNIST dataset classify different handwritten digits. Then use the trained model to classify the images. The classified images will then need to be stored in their respective folder on the master mobile.

2 Resources Requirements

– Frameworks and Tools used for both parts are as below:

- Application Side: Android Studio (Bumblebee 2021.1.1), JAVA (jdk 11), Google API Volley to send POST request to API
- Server Side: Flask, Python 3.9, Base64 to JSON decoder, PyTorch, PIL, TensorFlow, Keras, TFLite

– System requirements:

- PC: 8 GB RAM, 128 GB Storage
- Mobile: 1 GB RAM, 16 GB storage, Android OS > 5.0, Bluetooth

– Stable Internet connection that allows to connect more than 5 devices, PC, and server

3 Development Approach

The aim is to create an app which asks user for camera permissions, clicking the photo after providing appropriate permissions and divide the image into 4 parts and send each part to 4 different client mobiles. Then client mobiles can classify digit and store it on the master mobile.

The front end in the master mobile was designed into 2 fragments. In the 1st fragment, user was asked for permission to access, camera of the phone and in the 2nd fragment, user could click the photo of the digit, delete it, or save it. Then it will divide the image into 4 parts and send each part to the different client mobile.

At Client mobiles, user was asked for permission to access, Bluetooth, location and camera of the phone. Then it will classify the images accordingly and send the predicted digits to master mobile. At master mobile, the digit will be predicted and displayed using the majority percentage of the responses received from client mobiles.

Detailed approach is mentioned below:

For the front-end, the code can be explained as follows:(Reference: McAndroidApp)

- Created a new project using Basic Activity template.
- In the frontend on master mobile, we added ImageView and Button to Open camera and display click Image icon in the ImageView. The app would first ask for camera and storage permissions from the User.
- Camera view as all default options like flash on/off/auto. After capturing an image 2 options are given Discard or Save.
- we used Nearby Connections API to connect with multiple clients that ask user to client to connect with master mobile using Bluetooth. It requires user to choose Strategy that determines how app can connect to nearby devices. We used P2P_STAR strategy to connect with master mobile. We declared all the required permission in the AndroidManifest.xml in order to establish connection.
- We used "Advertising" process to make our master mobile discoverable and used "Discovery" process for client mobiles to find out nearby devices.
- When the connection is established at the master mobile, we have stored all the endpoints in array called "otherEndPoint". We made sure that endpoints don't overflow while sending image data to client mobiles by handling several conditions. To acknowledge the transfer of data we displayed a message "PayLoad sent at *endpointID*"
- When the connection is established at client mobile, it displayed message "Parent found at EndpointID" along with endpointID that is unique to each client device. Then we clicked on "Predict Photo" button on master mobile to send the image data to client devices and predict the the digit.
- We then converted the colored image into black and white single channel image and split the same into four quadrants using splitImage() function.
- To send the image and quadrant data between clients and master mobile, we used "PayLoadCallBack" function, these image and quadrant data was used to classify the handwritten digit within *getPrediction()* function and send the output to connected devices through "sendPayLoad" function.
- After receiving the predicted outputs from all the client mobiles, we used combineOutput() function to to merge the prediction and return the most frequent digit predicted by the client mobiles. We then displayed the predicted digit as "Image Predicted as *digit*"
- Then we stored the predicted image in the folder at the master mobile.

The working of deep learning model to classify handwritten digits:(Reference: *MLModelCreator.py*)

- At first, we imported all the needed modules for training our model like tensorflow, seaborn, numpy, pandas and matplotlib. Then we imported MNIST dataset to get training data with its labels and also the testing data with its labels. The Machine Learning pipeline we used for the recognition of the digits is trained on MNIST dataset. It includes 60,000 training images and around 10,000 testing images of handwritten numbers from 0 to 9.

- The validation set is 10% of the training set.
- The deep learning model, built has several convolutional layers and in-between them max pooling layers and dropout layers are inserted to avoid overfitting.
- The activation units used in all the convolutional layers is ReLU (Rectified Linear Unit) except the final layer where Softmax activation is used to identify the class of the digit present in the image.
- We divided the input set of our image into four parts in order to specify four quadrants.
- To train the model, we called the model.fit() function of Keras. It takes the training data, validation data, epochs, and batch size as the parameter. The model is trained on categorical cross-entropy loss via RMSprop (Root Mean Squared Propagation) optimization.
- After the training, we used TFLiteConverter to deploy the models to edge devices. Then the model has been tested to give 86% accuracy on the testing images.

4 Conclusion

This project helped us in gaining an insight regarding android app development. We learnt how to create basic android app and connect to different client mobiles using Nearby Connections API. and use the master mobile to divide the image into different parts and sent them to each client mobiles to classify the handwritten digits. Also we got a great knowledge regarding MNIST dataset and how to train a deep learning model. We explored various libraries such as Tensorflow, Keras, TensorFlow Lite, etc. We got insights regarding data preprocessing and normalization, creating neural network architecture and finally training the model on dataset using cross validation. Working as a team we learnt about how to contribute in a collaborative environment and used GitHub for the same. Apart from that we were able to brush up our basic programming skills on Java and Python.

5 Contributions

Drishti and Harsh helped in development of backend of the project creating the server and implementing the deep learning model on both master and client mobiles. Aatish and Mohak were involved in frontend design, implementing logic in Android studio as well as integrating backend and frontend. Rahul helped in GitHub operations regarding push, pull, creation and merging of branches etc. He also helped in final code review, appropriate documentation and video.

Acknowledgement

We would like to thank Dr. Ayan Banerjee (Instructor), Arizona State University for his constant encouragement towards the realization of this work.

References

Get the data received in a Flask request,
Available at: https://stackoverflow.com/questions/10434599/get-the-data-received-in-a-flask-request

How To Process Incoming Request Data in Flask,
Available at: https://www.digitalocean.com/community/tutorials/processing-incoming-request-data-in-flask

How to send a POST request using volley with string body?,
Available at: https://stackoverflow.com/questions/33573803/how-to-send-a-post-request-using-volley-with-string-body

How to build Convolutional Neural Network with tensorflow?
Available at: https://www.kaggle.com/code/amyjang/tensorflow-mnist-cnn-tutorial/notebook

How to convert trained model in Tensorflow Lite
Available at: https://www.tensorflow.org/api_docs/python/tf/lite/TFLiteConverter

Connect client mobile devices using Nearby Connection API
Available at: https://developer.android.com/codelabs/nearby-connections3

How to Post Data to API using Volley in Android?,
Available at: https://www.geeksforgeeks.org/how-to-post-data-to-api-using-volley-in-android/

android - save image into gallery,
Available at: https://stackoverflow.com/questions/8560501/android-save-image-into-gallery

Take photos,
Available at: https://developer.android.com/training/camera/photobasics

Accessing the Camera and Stored Media,
Available at: https://guides.codepath.com/android/Accessing-the-Camera-and-Stored-Media