Author: Shunyang Dong
Created date: 13 Feb 2020

Incorporate TensorFlow Lite into an Android App

Export a TensorFlow model in Python

```
In [58]: model.save('simple_model3.h5')
         new_model= tf.keras.models.load_model(filepath="/Users/shunyang/FHIR_coding/simple_model3.h5")
         tflite_converter = tf.lite.TFLiteConverter.from_keras_model(new_model)
         tflite_model = tflite_converter.convert()
         open("tf_lite_model.tflite", "wb").write(tflite_model)
```

Model can be any TensorFlow model you build. This part of code converts TensorFlow model to TensorFlow Lite model and store it as a tflite file.

Import TensorFlow Lite in Android

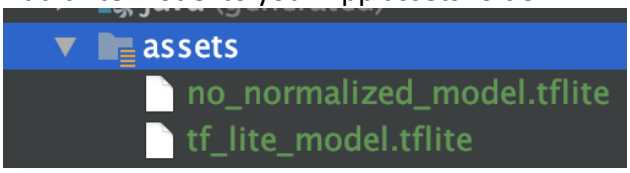
1. Add implementation to build.gradle file.

```
implementation 'org.tensorflow:tensorflow-lite:+'
```

2. Set apptOptions to noCompress for tflite.

```
aaptOptions{
    noCompress="tflite"
}
```

3. Add tflite model to your App assets folder.



4. Import TensorFlow lite interpreter in the activity class.

```
import org.tensorflow.lite.Interpreter;
```

5. Load model from assets folder.

```
private MappedByteBuffer loadModelFile() throws IOException {
    AssetFileDescriptor fileDescriptor = this.getAssets().openFd( fileName: "no_normalized_model.tflite");
    FileInputStream inputStream = new FileInputStream(fileDescriptor.getFileDescriptor());
    FileChannel fileChannel = inputStream.getChannel();
    long startOffet = fileDescriptor.getStartOffset();
    long declaredLength = fileDescriptor.getDeclaredLength();
    return fileChannel.map(FileChannel.MapMode.READ_ONLY,startOffet,declaredLength);
}
```

6. Run model

```
try{
    tflite = new Interpreter(loadModelFile());
    float[][] inputVal = new float[1][5];

    inputVal[0][0] = female;
    inputVal[0][1] = male;
    inputVal[0][2] = single;
    inputVal[0][3] = married;
    inputVal[0][4] = (float) age;

    float[][] outVal = new float[1][1];
    tflite.run(inputVal,outVal);
    final String cholesterolString = Float.toString(outVal[0][0]);

    cholesterolTv.setText(cholesterolString);
```

NOTE: When we pass the input to the neural network, we need to put them into an array. We have five values as input in this case, so we need a two-dimensional array. Our output from the neural network is a two-dimensional array with. After running tflite model, it writes the results to the output value so we can read it out as a float.

Fetch FHIR data in Android

AsyncTask Class

AsyncTask enables proper and easy use of the UI thread. This class allows you to perform background operations and publish results on the UI thread without having to manipulate threads and handlers. You must implement two methods in your AsyncTask sub class.

- doInBackground() executes on a background thread. This method gets data and returns it. In this case, this part of code retrieves patient's information.
- onPostExecute() executes on the main UI thread when doInBackground has completed its execution. In this case, this part of code starts a new activity.

Retrieve JSON format data from the web service

1. Create an HttpsURLConnection object representing a connection to the URL's server.
2. If the connection response is ok, open an input stream and create a reader.
3. Connect the input stream to a json reader which parses json format data.

```
URL webServiceEndPoint = new URL( spec: "http://hapi-fhir.erc.monash.edu:8080/baseDstu3/Patient/" + selectedPatient);
//create connection
HttpURLConnection myConnection = (HttpURLConnection) webServiceEndPoint.openConnection();
if(myConnection.getResponseCode() == 200) {
    //open a stream to it and get a reader
    InputStream responseBody = myConnection.getInputStream();
    InputStreamReader responseBodyReader = new InputStreamReader(responseBody,  charsetName: "UTF-8");
    //now use a JSON parser to decode data
    JsonReader jsonReader = new JsonReader(responseBodyReader);
```

Decode JSON format data

To access an object in JSON data, call beginObject() to consume object's opening brace. Then have a loop to process each key/value pair inside the current object. Call nextName() to move on to the next key/value pair. In the end, call endObject() when the loop finishes.

```
jsonReader.beginObject(); //consume array's opening JSON brace
while(jsonReader.hasNext()){
    //process key/value pairs inside the current object
    keyName =jsonReader.nextName();
    if(keyName.equals("name")){...}
    else if(keyName.equals("gender")){...}
    else if(keyName.equals("birthDate")){...}
    else if(keyName.equals("maritalStatus")){...}
    else if(keyName.equals("telecom")){...}
    else if(keyName.equals("address")){...}
    else{...}
}
jsonReader.endObject();
```

Based on the value's type, we have different way to extract the value.

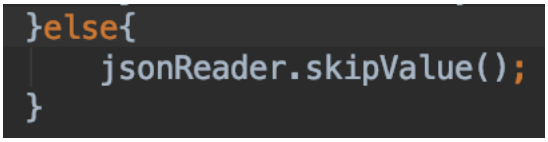
- If the value's type is an array, we use an inner loop to process each element in the array:

```
if(keyName.equals("name")){
    jsonReader.beginArray();
    while(jsonReader.hasNext()){
        jsonReader.beginObject();
        while(jsonReader.hasNext()){
            keyName = jsonReader.nextName();
            if(keyName.equals("family")){
                familyName = jsonReader.nextString();
                patient.setFamilyName(familyName.substring(0,familyName.length()-3));
            }else if(keyName.equals("given")){
                jsonReader.beginArray();
                while(jsonReader.hasNext()){
                    givenName = jsonReader.nextString();
                    patient.setGivenName(givenName.substring(0,givenName.length()-3));
                }jsonReader.endArray();
            }else{
                jsonReader.skipValue();
            }
        }jsonReader.endObject();
    }jsonReader.endArray();
```

- If the value's type is string, we can extract the string directly:

```
else if(keyName.equals("gender")){
   patient.setGender(jsonReader.nextString());
```

- If we don't want to extract the current value, skip it:



References:

https://developer.android.com/reference/android/os/AsyncTask