Android App Development with custom TensorflowLite model deployment Project Report

Abhishek Satpute, Deepak Tanwar

1 Abstract

This project presents the development of an Android application integrated with a custom TensorFlow model, utilizing ResNet50 architecture, for real-time image processing. The application enables users to capture images within the app, which are then processed by the custom model. TensorFlow Lite facilitates the deployment of the model on mobile devices, ensuring efficient and accurate image analysis. The processed images and the corresponding processing time are displayed on the app's interface, enhancing user experience and interaction. The model was trained using TensorFlow in Python on a relevant dataset, emphasizing its adaptability to various image recognition tasks. This integration of a custom model with an Android app showcases the potential of on-device machine learning, highlighting the synergy between TensorFlow's robustness and the mobile platform's accessibility. The project's success underscores the practicality and efficiency of deploying custom models on mobile devices, opening avenues for versatile and accessible image processing applications.

2 Introduction

This project amalgamates an Android application with a custom TensorFlow model based on ResNet50, aiming to enable real-time image processing on mobile devices. Integrating the capability to capture images within the app, it harnesses a bespoke TensorFlow model to analyze images seamlessly. Leveraging TensorFlow Lite for model deployment ensures swift and accurate processing while maintaining computational efficiency on Android devices.

The primary focus is on creating a user-friendly interface that seamlessly interacts with a trained TensorFlow model for image analysis tasks. The project's foundation lies in the detailed training of the model using TensorFlow in Python, enabling its versatility in recognizing various types of images. This project signifies the merging of advanced machine learning capabilities with the convenience of mobile platforms, paving the way for accessible on-device image processing applications.

3 Task 1: Image Capture and Processing

The code utilizes the ActivityResultContracts.StartActivityForResult to launch the device's camera for image capture. The captured image is converted to grayscale using a custom algorithm.

4 Task 2: Image Classification with TensorFlow Lite

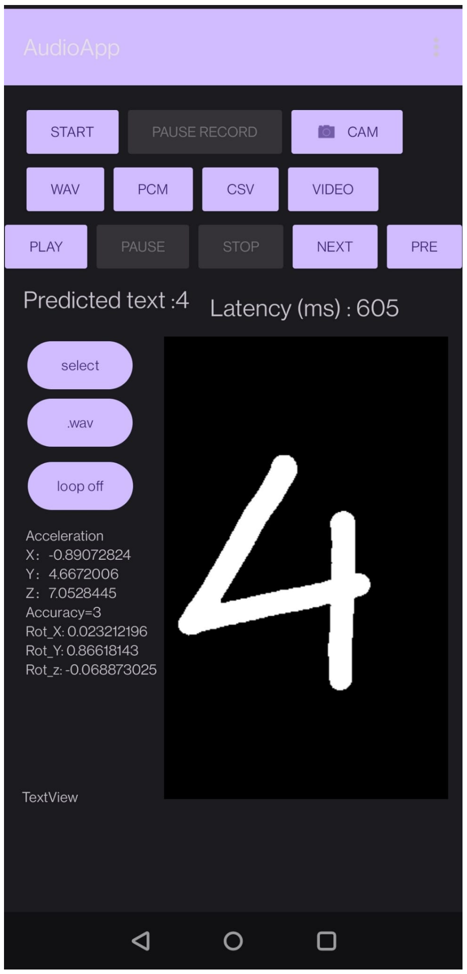
The grayscale image is processed using the TensorFlow Lite framework. The model used for inference is Rn50Lite, which is pre-trained for image classification. The result of the classification is displayed in the Android app, showing the predicted label and the latency of the process.

5 Task 3: User Interface Updates

The grayscale image is displayed in an ImageView (IVPreviewImage), providing a visual representation of the processed image. The predicted label and latency information are updated in corresponding UI elements (predicted and phrases).

6 User Interface Design

The Android app features a clean interface with a prominent camera button for image capture. Upon capturing an image, a processing indicator appears, followed by the display of the processed image and analysis results. Users can also view the time taken for image processing, ensuring a seamless and informative experience. This design prioritizes simplicity and user engagement, making image processing efficient and accessible within the app.



```
Predicted text :4    Latency (ms) : 605

Acceleration
X:  -0.89072824
Y:  4.6672006
Z:  7.0528445
Accuracy=3
Rot_X: 0.023212196
Rot_Y: 0.86618143
Rot_z: -0.068873025
```

7 Conclusion

In conclusion, this project effectively merges an Android app with a custom TensorFlow model, enabling efficient real-time image processing on mobile devices. Through the integration of a user-friendly interface and TensorFlow Lite deployment, the app facilitates seamless image capture and analysis.

The utilization of TensorFlow in Python for model training ensures adaptability across diverse image recognition tasks, showcasing the system's versatility. This project signifies the potential of mobile-based machine learning applications, bridging advanced technology with the convenience of Android platforms.

Overall, the successful integration of the TensorFlow model into the Android app sets a foundation for future advancements in on-device image processing, emphasizing both technological innovation and user-centric design.