Boston University

Electrical & Computer Engineering

EC463 Senior Design Project

First Prototype Testing Report

Swing On

Team 20 SwingOn

Team Members:

Yoel Beyene

Tingru Lian

Hanlin Mai

Jessica Martinez Marquez

Equipment

For the hardware part of the prototype testing, a device that can run iOS 15, iPadOS 15, or macOS Catalina (or later) was set up to run and simulate the App. An iPhone 6S (or newer for smoother processing) was also part of the setup to run the App and show it can work on the device.

For the software part of the prototype testing, XCode 13 is the development environment for setting up the testing. Swift is the language used for programming, and SwiftUI is the user interface toolkit for designing the App presented in prototype testing.

Three Tensorflow-based pose detection pre-trained models (PoseNet, MoveNet Lightning, and MoveNet Thunder), which run on the iOS CoreML framework, were used to estimate the spatial location of a person's 17 key body joints from a live video.

Additionally, a phone tripod was used for the stabilization of the iPhone for testing the prototype. One team member also acted as the user to perform several movements in front of the camera of the iPhone to demonstrate the model results.

Setup

The setup consisted in connecting the iPhone to the computer to install the App. Then, the SwingOn project was built on XCode and downloaded on the iPhone by running the code. A tripod was set up for the iPhone to ensure a stable position. This was to ensure a consistent laboratory environment to achieve better result accuracy when performing the test.

Measurements

The measurements taken were as follows. The App should be able to successfully run on any iOS device, specifically, on the iPhone that is used to test. Each screen of the App should be clearly displayed. The main screen should have the “Swing On” logo, a welcoming message, and a button directing to the camera screen. The camera screen should ask permission from users to access and open the camera. The buttons on the user interface should perform the following tasks: the “Open camera” button should open the camera screen, “Back” Button should return to the main screen. The camera screen should display a person's body poses in real-time (>30fps) as well as allow users to make adjustments to the model. There should be a total of 17 key points detected on a person’s body, including nose, left and right eye, left and right ear, left and right shoulder, left and right elbow, left and right wrist, left and right hip, left and right knee and left and right ankle. The confidence score should be shown on the camera screen and indicate the probability that a keypoint exists in that position. The time it took for the model to detect the pose of the person should also be displayed on the screen. The number of threads should be able to increase or decrease, by increasing the number of threads the App should be able to speed up the execution of the operation. However, this would make the model use more resources and power. There should be a total of 3 delegates (hardware accelerators) to choose from, which includes: CPU, GPU, and NPU. CPU is the safest and simplest choice. However, it is usually slower and consumes more power than running the model on accelerators. GPU was the most widely available accelerator and provides a decent performance boost. NPU was similar to a GPU, but instead of accelerating graphics, it accelerated neural network operations such as convolutions and matrix multiplies. There should be a total of 3 Models to choose from Posenet, Lightning, and Thunder. Posenet was the standard pose estimation model that can detect these 17 key points. Lightning was smaller and faster, but less accurate than the Thunder version. It could run in real-time on modern smartphones. Thunder was a more accurate version but also larger and slower than Lightning. The App should be able to run without an internet connection.

Conclusion

All the requirements in the measurements were met in the first prototype testing. However, there were some requirements that needed to be met for the App to perform the designated task, such as the App can only work in a perfect laboratory condition in order to detect all the points of a person with high accuracy. Which means if the background of the person is cluttered or a part of the person's body is not clearly shown, there will be a lot of misclassifications of the body points, and this will result in a low accuracy. Another issue is that the App currently performs real time body analysis. The goal is not only to have a real time analysis of a golfer's motion using a camera, but also enable users to upload a video for analysis as well. Therefore, the next step for the team is to work on developing a feature to upload pre-recorded videos to the App and analyze them by applying the pre-trained model. Since the pre-trained model is used for image analysis, the video uploaded needs to be preprocessed, that is split frame by frame, in order for the model to classify and detect body points.