University of Regina | Go far, together.

Testing Document for Eye State Detection

IFD DriverAlert

Date: Dec. 03, 2024

Project Overview

Our project involves creating a machine learning model implemented in a system to detect eye states (Open/Closed). The goal is to have the system run in real time, using a TensorFlow Lite model deployed on a Raspberry Pi, to detect if the driver is falling asleep. The current system uses a live camera feed to classify eye states and display the status on the screen.

Test Strategy

At this phase, User Testing is more appropriate, as the primary goal is to validate the end-user experience with the live eye detection functionality. Unit tests are less critical in our case because the project primarily integrates ML models, hardware (camera and speaker), and software.

Test Scenarios

- Live Eye Detection Accuracy: Verifying that the system correctly identifies the "Open" and "Closed" states for both eyes.
- Performance on Raspberry Pi: Ensuring smooth real-time detection without significant delays.
- Robustness against different cases:
    - Testing with varying distances from the camera.
    - Testing under different lighting conditions.
    - Testing with glasses and without.
    - Testing when the driver looks away or down.
    - Testing system behavior with multiple face Input.

Test Cases and Results

Test Case	Scenario	Steps Taken	Expected Result	Actual Result	Defects
Live Eye Detection Accuracy	Validate that the model correctly identifies eye states for "Open" and "Closed"	Observed the output on the screen while blinking or keeping eyes open/closed	Correct classification for both states	Correct classification for both states	None
Performan-ce on Raspberry Pi	Ensure smooth real-time processing	Ran eye_detection_rpi.py for 10 minutes and observed any lags	Real-time detection without delays	Real-time detection without delays	None
Different Distances	Test eye detection at close, average, and far distances from the camera	Moved between 25 cm and 2 meters from the camera	Eye state detected correctly	Worked well at close and average distances, but less accurate at far distances	Detection accuracy is reduced at far distances
Different Lighting Conditions	Test detection under day and night lighting for both laptop and Pi camera	Used high light, dim light, and natural light	Correct detection in all conditions	Worked well under all conditions for the Pi camera, but the laptop camera struggled in low light	Laptop camera not optimized for low-light scenarios
Detection with Glasses	Test detection with and without glasses	Wore different types of glasses while testing detection	Eye state detected consistently	Eye state detected consistently	None
Multi-Face Input	Ensure only the driver’s face is detected in multi-face scenarios	Tested with two people in the frame	Keep tracking the driver's eyes and ignore the other face	Keep tracking the driver's eyes and ignore the other face	None
Looking Down	Test detection when the driver looks down	Looking down at a phone	Should not classify as "Closed"	Detected as "Closed"	System needs improvement to handle head down position

Defects and Plans to Improve

- Reduced Accuracy at Far Distances:
    - Cause: Eye features become less distinct as the subject moves farther from the camera.
    - Plan: Not determined yet as this does not hurt our project goal because the driver can not go far to the back.
- Low-Light Performance on Laptop Camera:
    - Cause: Laptop camera is less capable in low light conditions compared to the camera we are using with the Pi and less clear.
    - Plan: Again this does not hurt our project goal and we already mentioned in our proposal that we need to use a camera which is capable of detecting in low light such as Pi Camera NOIR.
- Incorrect Detection When Looking Down:
    - Cause: When looking down, eyes seemed to be closed so the model is detecting closed.
    - Plan: This is a big concern that we have to work on in the coming semester.

Conclusion

Our main concern at the beginning was whether the Raspberry Pi would have the computational power to run a machine learning model effectively in real time. Through optimization and the use of TensorFlow Lite, we were able to eliminate this fear, achieving real-time eye state detection without significant delays.

The system performs well within the project's defined scope, including detecting eye states at close and average distances, working effectively with the RPi camera in both day and night lighting, and handling scenarios with and without glasses. Additionally, it successfully limits detection to a single face, aligning with the project goal to monitor the driver exclusively. Now, by saying "worked well," we do not mean that it was 100% accurate. Although the detection worked well in low light for the Pi camera, we still encountered occasional false negatives, though these were rare. Furthermore, the incorrect detection when the driver looks down is a significant concern that we must address. That being said, we still have to focus on improving the model's robustness to certain edge cases, particularly addressing head positioning when the driver looks down.