NIIT UNIVERSITY

End Semester Report: Distracted Driving Detection

Hridaya Annuncio
U101116FCS046

Company:
IBM

Industry Mentor:
Sujoy Roychowdhury
Ashish Rao

University Mentor:
Dr. Suman Sanyal

Off campus Mentor:
Bipin Sinha

Acknowledgement

I would like to thank my Industry mentors, Ashish Rao and Sujoy Roychowdhury for their guidance and continuous support. I learnt a lot working under them. I would also like to thank my teachers at NU who helped me with my education which consequently enabled me to work on this project.

Declaration

This paper has been solely written by Hridaya Annuncio, a 4th year student from NIIT University.

Endorsement

This internship was supervised by Sujoy Roychoudhury and Ashish Rao.

Table of Contents

1. **Part A - Technical** **6-8**
    - **Introduction** **6**
    - **Analysis and Methodology** **7**
    - **Conclusion** **8**
    - **Abstract** **8**

2. **Part B - Non-Technical** **9**

Part-A Technical

Introduction

Our team was trying to build 2 use cases for Deep Learning on Edge Devices. I was working on the Distracted Driver use case. We had to create a model/algorithm that would detect if a driver was distracted or not and then deploy it on a Rasberry PI and Android phone.

I worked in the CBDS(Cognitive Business Decision Support) department in IBM. This department deals with all data science related issues. IBM is mainly a service company. However, our project was purely research based.

Our project was divided into 2 phases. In the first phase we used an already available distracted driving dataset on Kaggle for our deep learning approach to the problem. This only helped detecting explicit distraction in images/frames. In the second phase, we detected more implicit kinds of distraction which included drowsiness and yawning on video clips.

Analysis and Methodology

In phase 1, we used a Kaggle Dataset wherein there were images of people holding a cup,texting on a phone, talking to someone on the backseat etc. This data displays explicit forms of distraction. This kind of distraction could be detected with mere images.

Hence, we created a transfer learning model for this.It was a classification problem where there were 9 classes of images that were categorized as distracted and 1 class of images were categorized as not distracted.

First a ground truth was calculated wherein the images were passed straight through a transfer learning model(MobiNet). It was observed that the accuracy was less and kept on shifting. Hence, we decided to give more information to the model.
The procedure was divided into object detection and posture detection. I was responsible for the former.
I used YOLO for detecting two objects : Steering Wheel and the Driver. YOLO v2 on its own can detect a person but cannot detect a steering wheel. Hence I had to create code to make the YOLO model learn how to detect a steering wheel. THis was done by creating annotations (LabelImg) and then putting it through training. The metric used was IOU. The steering wheel was to be detected as it can give information on the hands' presence.

The coordinates of the bounding box for the driver was further sent to the posture detection algorithm. The coordinates of the posture detection algorithm(open Pose) and the coordinates of the bounding box of the steering wheel were both sent to the main transfer Learning Model. This increased the accuracy of our Model.

So as to be able to deploy the model on the Rasberry Pi, we had to decrease the size of the model. For that we used the methods of Pruning and Quantization.

Phase 2 was dedicated to the detection of Drowsiness and Yawns. I was incharge of creating a Computer Vision code to do both. A colleague of mine was checking the accuracy of an already created Deep Learning model for the same. We realised that the computer vision version worked better. The code included the use of Landmark Detection using the Dlib library. Other projects we saw for the detection of Drowsiness were based on the blinking rate. However, ours' was based on the amount of time that the eyelid and the lower part of the eye were at minimum distance from each other.
This detection was done using videos. Again, the metric used for accuracy check was 1d IOU.
A problem faced was that when mp4 videos were used for testing, the detection code behaved differently as it was devised for live webcam detection. Hence, RTSP was used to take care of this problem.

To deploy this on an android device, I had to convert the Python Code to C++. This created problems with regard to Library installations as there was less documentation on the subject.

Conclusion

Our models worked well in detecting the kinds of distraction they were based on. We were even trying to find a way to detect absent mindedness. However, not even neuro-scientists have found a way to explicitly detect that.

Abstract

The project was based on the detection of a distracted car driver or pilot. It was divided into 2 phases. The 1st phase dealt with a deep learning approach to the problem while the 2nd phase dealt with a computer Vision approach. The 1st phase detected explicit forms of distraction and the model was created using image data. The 2nd phase detected more implicit forms of distraction like drowsiness and yawns and the algorithm was created for video data. This algorithm was converted from Python to C++ for the subsequent deployment on an Android device.

Part-B

Non-Technical

The main thing I learnt was how to approach a problem: The method of dividing a big problem into sub parts and learning the skill of solving each part. I also learnt the art of debugging.
Along with the former, I learnt how to work across states with team members based in Bangalore. I learnt how to communicate efficiently and work with coordination in a team.

On the whole, all the courses I studied in NU did help in one way or the other. However, in our Deep Learning courses, we did not go into the depth of the mathematics used. This is crucial when dealing with Deep Learning models in the industry. Also, C++ is an important language to be taught as it is used when trying to deploy a deep Learning/ Computer vision model on an Android phone.