Projeto Integrador - L.EIC

Mobile App using Object Detection for Car Driving

Filipe Campos up201905609@fe.up.pt, Francisco Cerqueira up201905337@fe.up.pt, Vasco Alves up201808031@fe.up.pt

Summary

A technology race is "on" between car manufacturers to produce automatic systems to help human drivers. However, these systems are expensive and the vast majority of the car fleet does not yet incorporate any of these systems, they typically use either RGB cameras or LiDAR sensors, or both. Neural networks are then used to translate these sensory inputs into semantic representations of the surroundings. This project brings driver assistance via object detection to a broader audience by running on any Android phone instead of requiring dedicated hardware.

Objectives

The goal of this project was to produce an Android application that generates an audible sound to warn the user in certain situations:

- When another car is in front of the user's car
- When a pedestrian or another obstacle is in the road,
- To notify on horizontal and vertical signs.

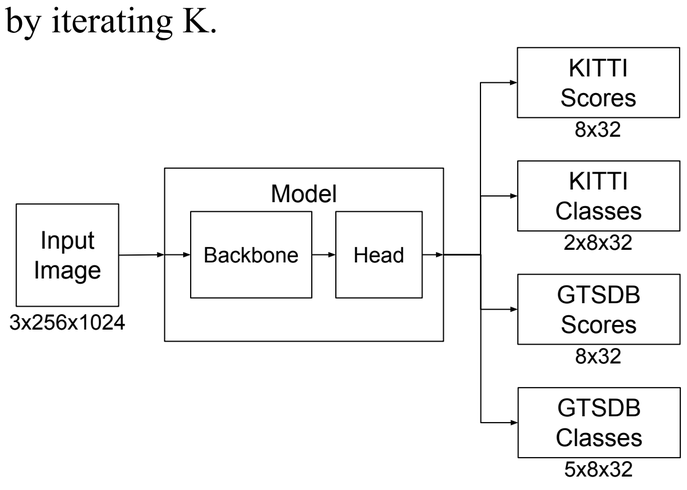
The neural network should run on the mobile phone itself to mitigate problems that accompany the usage of a network connection to an external server.

Datasets

The **KITTI Dataset** was utilized to train the model to detect cars and pedestrians. Although KITTI contains traffic signs, they are not annotated. Therefore the **GTSDB Dataset** was also used to detect and classify road signs.

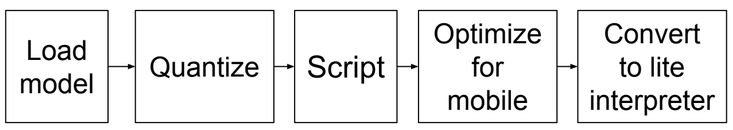
Model

The model was developed using the objdetect package, which uses **PyTorch**, and uses a pre trained torchvision backbone that connects to a head with four outputs, two sets of scores and classes arrays with predictions for each cell in a **8x32 grid**. The scores array contains floats in the range [0,1] that predict the whether or not that cell contains an object while the classes array is composed of **Kx8x32** elements where K is the number of classes. For each grid element we can determine the most likely class type by iterating K.



Conversion to Torch Mobile

After training, the model was converted to torch mobile using the following pipeline:



Application

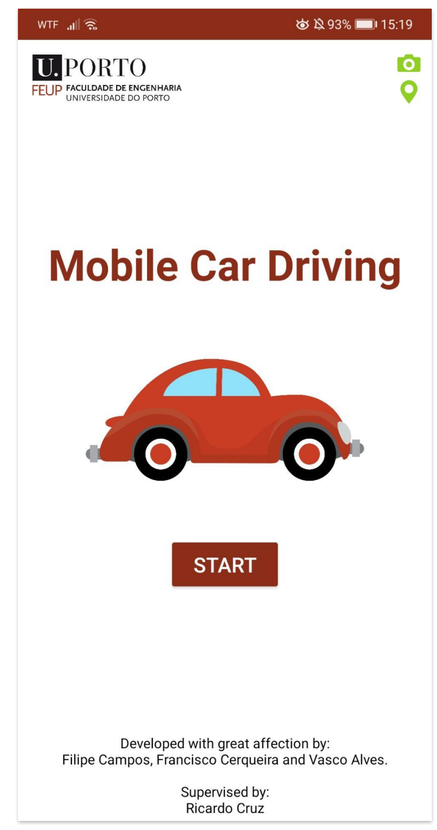
The android application was developed using Java and utilizes the PyTorch Android API to run the previously converted model. The pipeline used includes the following steps:

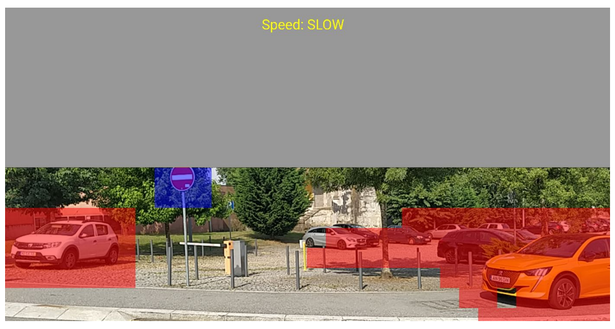
1. **Pre-processing**: the image is cropped and adjusted to the required aspect ratio (4:1);
2. **Model Evaluation**: the image is evaluated by the model and returns which classes were detected, along with the corresponding position and score;
3. **Post-processing**: the results returned by the model are now post-processed, returning a last analysis result that includes the collection of objects and warnings that should be displayed, as well as other relevant info, like the device's current speed according to the GPS.
   This analysis uses not only the information of the last frame, but also the **last 3 frames**, in order to increase the accuracy;
4. **Prediction Output**: the results obtained on the previous step are sent to the view class and sound warnings are played if necessary, using Synthetic Voice.

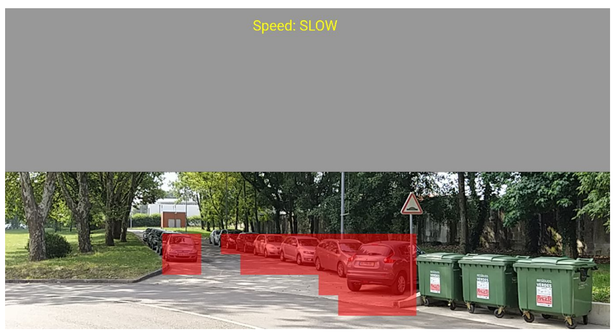
MobileNet-V3 small was chosen as the model backbone due to its high FPS and small size, making it ideal for a mobile application.

Model Backbone	Average FPS	Size (MB)
MobileNet-V3 small	7.6	3.6
MobileNet-V3 large	2.0	11.4
ResNet-50	0.4	89.7

Results







Real Use Case

Conclusion

The project was successful and we managed to create an Android application capable of warning its users about nearby vehicles, pedestrians and traffic signs using a Text to Speech synthesizer.

As planned, the application was uploaded to the Google Play Store and can be accessed using the QR Code.

GET IT ON Google Play