Project Report

Valerio Tiri, Federico Rovighi

December 30, 2023

1 Introduction

The purpose of our work, as asked by professor Cerri, was to create a new ADAS or try to solve the issues of an existing one.

After thinking about it for a long time we ideate a new ADAS to improve children's safety as car passengers. The idea is to use a neural network able to identify the occupants of the front and rear seat in a car in order to:

- Trigger a warning alarm when a child or infant is detected in the car and at the same time there are no adults.
- If child/infant are detected left alone the system records the outside temperature and act consequently by heating or cooling the vehicle interiors to save the babies from hot/freezing weather.
- Automatically disable front-passenger airbags in case of infant in infant seat detection, to improve safety in case of accidents.

2 Dataset acquisition

The starting point of the project has been choosing the dataset; we have found one that fits very well our purposes, the SVIRO dataset.

SVIRO, acronym of Synthetic dataset for Vehicle Interior Rear seat Occupancy is a dataset created for detection and classification, consists of 25.000 sceneries across ten different vehicles and provides several simulated sensor inputs and ground truth data. It comprehends different types of images, coming from RGB camera and IR camera, depth images, masks, key points for posture understanding and bounding boxes. The choice of using only a part of this dataset was dictated by the computational constraints (time of HPC use =24h), the storage constraints (only 50GB in HPC), and the choice of the neural network. In fact, we downloaded the simulated thermal camera images for 5 different cars (BMW X5 random, Lexus GSF, Mercedes A-Class, Ford escape, Tesla Model 3), and their respective bounding boxes.

The SVIRO dataset can be downloaded at this like `https://sviro.kl.dfki.de/`.

The authors said that the simulated near infrared images are not physically accurate due to the method of their generation. Specifically, they simulated a dark environment, lighting it up with a red light (R=100%, G=0%, B=0%) placed next to the camera inside of the car illuminating the rear seat. They then applied a greyscale filter to simulate near infrared images.

3 Neural Network

We decided to adopt a neural network recently released by Ultralytics, the YOLOv8n. The 'nano' version is the smallest one, this translates in having a faster training at a cost of slightly worst results. We took a pretrained version of it and fine-tuned the last layers with our data.

This choice will clarify why we took only those few parts of the dataset; first of all, the YOLOv8 when in training phase, automatically turns the train images into grayscale. Thus, using the corresponding RGB images would have been a useless waste of memory space since it would not have brought any benefit to the training. Then, the YOLOv8 needs bounding boxes to continue the training, and here is where the problem starts.

4 Dataset preparation

The first thing to notice is that the contents of the bounding boxes text files provided by the SVIRO authors are not in the same format as the one required by the YOLO network.

The original format includes the class label, the upper left corner, the lower right corner in global coordinates; all these parameters are divided by a comma.

While the yolo networks require a format that contain: the class number, the center coordinates, the width, and the height of the bounding box in relative coordinates; these elements separated by a space.

To solve this problem, we developed a python program which takes all the text files in a folder and turns their content in the required YOLO format. In particular, it acquires the class and the coordinates given by the SVIRO file and calculate the bounding box center coordinates, height and width needed by the YOLO. In addition, it creates a new file reporting this information distanced by spaces instead of commas.

5 Training

5.1 First training attempt

At this point our initial attempt was to train a version of the YOLOv8n, pretrained on the COCO dataset. The training was made using the Parma HPC step by step on all the five folders containing five different car scenarios:

- BMW x5 random, 2000 images
- Tesla model 3, 2500 images
- Ford escape, 2500 images
- Lexus gsf, 2500 images
- Mercedes A-class, 2500 images

We started by training it for 15 epochs on the first folder, then we loaded the obtained model as base for the following training on the second folder; we held this trend for all the aforementioned folders.

When we looked at the final results, we noticed a big problem: the classification of certain classes was completely wrong. After an accurate analysis we found that the main problem lay in the classes specified into the label files. In fact, there were no classes 5 (Empty infant seat), and 6 (Empty child seat), because all the child and infant seats were labelled as child in child seat (class 2) and infant in infant seat (class 1), even if no child or infant was placed on the seat. On the other hand, if a child or infant was present on his seat it was labelled as an adult. This led to an absent detection of empty child/ infant seats and errors in recognizing adult class and child/infant in child/infant seats.

5.2 Labels correction

To address this issue, we wrote a python script that was able to convert the classes under certain conditions. The basic idea of this additional code was the following: scan every annotation file searching for classes 1 and 2 (infant/child in infant/child seat). The program then determined if inside the same file there was also reported a class 3 subject (adult); at this point the program checked if the bounding box area was smaller than the area of the seat and if its center was placed inside the perimeter of the seat's bounding box. These conditions allowed us to understand if the class adult actually represented an adult or if it was an infant/child positioned on its infant/child seat. We modified the class number into 5 or 6 if there was no adult class corresponding to the constraints, instead we removed the line of the adult class (class 3) if it satisfied the constraints of our code.

5.3 Second training attempt and Bug correction

Following the adjustments made to the label files, we again trained a pre-trained YOLO model with the same methodology described before.

Upon reviewing the results, we noticed some errors in the bounding boxes used for the validation batch. Further investigation revealed a bug occurring in some labels, caused by the program that we have seen before. The problem was simple and at the same time very impactful on the training process; sometimes when a line into the label text file was modified it resulted in that line being concatenated with the previous one causing a line with more than the required five elements. This made the network see the bounding box in an incorrect manner that led to less precise detections. To solve this bug, we wrote another python code, since we were not able to correct it on the first code implemented. This new code simply looks at every line in the annotations and if one had more than 5 elements it meant that such line suffers from that bug, so the code divided the line in two starting the second line from the last character of the fifth element.

5.4 Third training attempt

We trained again our YOLOv8n neural network, modifying the dataset and the training strategy. First, we joined all the folders together to have a more various dataset during training (this increased the entropy of the dataset). However, this solution has some drawbacks, since the dataset was larger the HPC was not able to perform all the 15 epochs in less than 24 hours (and as students we had this computation time limit). To overcome this problem, we divided the training into steps of 10 epochs for three times and a final training of 20 epochs for a total of 50 epochs. With this strategy we were able to overcome the 24h computation-time limit.

The results obtained from this last training are very satisfying. We get a precision of more than 0.99 and the results reported in the confusion matrix are noteworthy. The actual results on the validation batch provided by the network are also excellent. We must note that all the validation images are taken from the training set so their precision might be deceptive, so, there is the necessity to test the network also on totally unseen images always coming from the SVIRO dataset. In this phase we still got very good predictions since on a batch of 10 images it got everything correctly.

6 Testing on real images

The final validation step was to analyze the network's performances on real life images. We wanted to prove that even if the network was trained only on synthetic images, it could also effectively detect real people, children, objects etc. To do so we shot some images in our cars where different subject were present. Since we have limited access to children/infants we simulated their presence in the car with photoshopped images and dolls.

In order to detect the occupancy of the real images we used the last network we trained and that we have seen before (pretrained, 50 epochs). We obtained promising results detecting almost all the subjects. On a test batch containing 29 subjects, 27 were correctly identified, so it got 93.1% of precision.

These outcomes are considered promising, especially given the constraints of time and resources. It's worth noting that the object detection network used is the nano version, the smallest one, and the dataset is not expansive.

7 Application

To develop the last part of our project, we wrote a python script structured in the following steps:

- The program takes as input the neural network model we trained earlier and a couple of images. These images simulate a single instant in the two image's streams coming from the front and rear cameras of the vehicle.
- The function 'bbox_extractor' is then invoked, having as input a single image at time, this function uses the Ultralytics' libraries to submit the image to the network. All the predictions of the network on that image are saved on a list of tensors. Each tensor contains the information of a single bounding box such as the coordinates of the upper-left vertex, bottom-right vertex, the confidence, and the class of the prediction. This list is then returned as output of the function.
- Once the extraction is made for both the images, the program draws the bounding boxes, and write the class labels, on the images, saving them in a specific repository.
- The last step consists in parsing which classes were detected in both image in order to provide information about the internal state of the vehicle. If a child or an infant are detected in the car and at the same time no adult is present, the program provides a warning message and simulates another action that the car could take. To decide we used a random variable simulating the outside temperature and the actions are weighted on it. For example, it could start the heating/AC or roll down the windows. We also developed the recognition of an

infant in the front seats, if this condition is fulfilled the program displays a message saying that the passenger airbag is disabled. The last step is the recognition of an everyday object, and if no adult is present in the car, it shows a message, saying that maybe something has been left in the car.

8 Real environment implementation

Implement this type of ADAS in production cars might not be very difficult. Our idea is to place a pair of near-infrared cameras in the passenger compartment in order to gather information about the seat occupancy both for front and rear ones. The real-time fusion of information from these cameras can significantly enhance safety thanks to the active airbag management system and even if infant and child are left alone into the car.

The cameras integration is simple, we suppose to use the Mini 256/384/640 LWIR Micro Thermal Camera Module produced by Infiray, which is only 21x21x15 mm and weight less than 8 g. The integration of such camera for the front seat occupancy detection can be easily accomplished on the rearview mirror. While the rear occupancy detection can be made by integrating the infrared camera into an already existing internal lighting systems or by placing it between the roof and its covering which is normally made of polyester fabric; this layer should be as thin as possible in order to avoid high absorption of the near infrared rays which will lead to worse camera performances.

In a real implementation of the safety system when a child/infant is left alone, the system should trigger the alarm after a certain amount of time (i.e. five minutes); it should also communicate with the air conditioning/ heating control systems, and the windows' ones. Meanwhile the honk could also be triggered, while rear/headlamps might start blinking. Moreover, a radio signal could be sent to the mobile phone app of the car owner making the phone ring continuously until the child is no longer in the car or until an adult is present. If after a certain time no measures are taken the vehicle should alert the authorities.

The last part of the ADAS is the automatic disabling of the front-passenger airbag in case of detection of an infant in infant seat, the Central control unit should communicate to the airbag control system to do this. In case of an everyday object detection and no adult is inside the car, the system can send a one-shot message to car owner's app, reminding him that he may left something in the car.

9 Conclusions

We believe that such a system might be very useful for busy families, young parents or families with only one child. The parents of these children have a higher probability to leave them in their car. Alarmingly, the American average is about 37 children dead every year solely due to heat strokes. It is evident that our system has the potential to improve safety and save many lives every year.

To the best of our knowledge, some car manufacturers tried to solve this problem using systems based on basic technology, such as weight or temperature sensors. The literature we analyzed showed that none of them used a system comprehensive of neural networks to solve the inferences. Under this new light we can say that we successfully accomplished the task requested by the professor in developing/prototyping a new ADAS.

About ourselves, we practiced with the training of neural networks and learned something new thanks to a new dataset. We faced some challenges, but we were able to overcome them and

obtain satisfying results both in means of metrics and in personal satisfaction, having even some fun developing this new ADAS.