# Report second project ‘Traditional data fusion’.

1. INTRODUCTION

The aim of this project is to fuse data from different sensors looking at the same scene in order to improve the confidence and the reliability of the results. To do so, we exploited the results of the previous project and an Hungarian algorithm to perform the actual fusion.

Disclaimer: since we used the same dataset and the same network architecture of the first project. We directly refer to it and we will not write anything in this report on the structure of both the dataset and the YOLOv8 network.

1. OUR WORK
   1. TRAINING RGB NETWORK

For this second project, we needed to fuse two image detection coming from different sensors and so different neural networks. The KAIST dataset helps us on this since it has both RGB and thermal images. We decided to use the network we described in the report of the first project to acquire the detection information on the thermal images. Then we needed also to acquire such information also on the RGB images, therefore we needed a new object detection network.

To do so we use another YOLOv8 neural network, and we trained it for 25 epochs starting from a pretrained model trained on the COCO dataset, as the model chosen among the thermal ones.

The training RGB set was made by the same images used for the thermal ones, but we took the RGB version, this was made for a matter of simplicity in the dataset building phase.

The results of this training were good, we achieved a very high precision, very close to the thermal ones of 0.95, mAP = 0.97, satisfying losses and looking at the validation batch almost every subject was correctly identified.

* 1. HUNGARIAN ALGORITHM

Once we had the two neural network models trained, to accomplish the data-fusion task, we needed to find a way to fuse their outputs.

To do so we needed to perform a prediction on the same scene detected by the two different sensors. We developed a python code to perform these predictions and extract information about the bounding boxes coordinates (center x, center y, width, heigh), their confidence, and their class. Of course, these predictions were made for both the networks.

The predictions were placed into lists of tensors one per each image, where each tensor in the list reported the information about a single bounding box.

The two lists, one containing the prediction of the RGB network, and the other one containing the prediction of the thermal network, were given as input to the function performing the Hungarian algorithm.

The Hungarian algorithm function creates a cost matrix in which rows and columns represents the bounding boxes of the different lists and therefore the matrix values are the costs (1 - IoU) of the respective bounding boxes.

Starting from this matrix we found the correspondences between bounding boxes by searching for the lowest cost value. A low cost value meant a high IoU (Intersection over Union) therefore a high probability of the two bounding boxes identifying the same subject. Then, placing a threshold for both the confidence and the IoU we selected between the corresponding boxes; the one with a higher confidence was considered the valid one while the new confidence was computed as the mean value.

The python program, in the end, draws the bounding boxes around the detected subject and write the related classes on top them, saving the resulting images in the specified directory.

1. CONCLUSIONS

As could be easily imagined, the Hungarian algorithm is not the best way to perform non-learned data fusion. The need to find correspondence between the bounding boxes of the two images decreases the number of the object detected; this strategy does not increase the precision of the overall classification but increase the reliability of the detected cases. This is possible since the final detections are the ones that are present in both the starting images.

Valerio Tiri, Federico Rovighi

02/01/2024