5 April 2022

HEYER Sébastian
sebastian.heyer@imt-atlantique.net

SELLAMI Aziz
aziz.sellami@imt-atlantique.net

WANG Yuming
yuming.wang@imt-atlantique.net

NING Zijie
zijie.ning@imt-atlantique.net

Supervisors:
Mostapha RIZK
Amer BAGHDADI
Vincent GRIPON

Rapport Projet 3A

Smart-cameras network for multi-view AI-based object detection

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Abstract

Object detection is widely used in different fields such as security, autonomous vehicles and medical imaging. There exist several real-life applications in this regard: counting vehicles, crowd detection, obstacle avoidance in autonomous vehicles. This project has for pupose to implement a smart-cameras network in order to use such algorithms in edge devices and compensate the loss in accuracy due to the usage of small neural network in these edge devices to run in real time.

Keywords : Deep Learning, Edge Device, Computer Vision, Communication Protocol (TCP/IP), Dataset

Contents

1 Context **4**
1.1 Trend . . . 4
1.2 Our project . . . 4
1.3 Challenges . . . 4
1.3.1 Technical challenges . . . 4
1.3.2 Social challenges . . . 5

2 Dataset **6**
2.1 Motivations . . . 6
2.2 Related Work . . . 6
2.3 Our dataset . . . 6
2.3.1 Acquisition Protocol . . . 6
2.3.2 Our dataset without orientation . . . 9
2.3.3 Our dataset with orientation . . . 9
2.3.3.1 The layout of the room . . . 9
2.3.3.2 The job to be done . . . 10

3 Object Detection Algorithm **11**
3.1 Overview of algorithms . . . 11
3.1.1 One-stage and Two-stage methods . . . 11
3.1.1.1 Two-stage detectors . . . 11
3.1.1.2 One-stage detectors . . . 11
3.2 Comparison of different state of the art . . . 12
3.3 How Yolo works . . . 13
3.3.1 Image grid . . . 13
3.3.2 Bounding box . . . 13
3.3.3 Intersection over union (IOU) . . . 14
3.3.4 Combination of the three techniques . . . 14
3.4 Training on our dataset . . . 15

4 Multi-view **16**
4.1 Technical Choices . . . 16
4.2 Challenges First Approach . . . 18
4.2.1 Challenges . . . 18

1 Context

1.1 Trend

Object detection remains one of the most challenging field in the modern world : autonomous vehicles, medical imaging, crowd detection, exploration... Currently, artificial intelligence (AI) is the key-trend for object detection in images and videos. Convolutional Neural Networks (CNN) architectures allow to meet the requirements of handling multiple objects in real-time. However, the implementation of the CNN comes with a cost of limited available energy.

Furthermore, the interest in having them integrated on embedded system underline the consideration of the available resources. Drones, camera, vehicles cannot rely on high computational performances because of the lack of space. Depending on the architecture of the CNN, the algorithm can be either better on the accuracy and precision or better on the power consumption. This trade-off is what will determine which CNN and hardware will be used in embedded systems.

One way of increasing accuracy and precision in the learning and inference phases is to incorporate a multi-view network. Several nodes will record the same object with different angles so that each one of them have different guesses. The goal is then to gather all inferences and to decide which one is right or wrong. The advantage of this technique is that it overcomes hurdles like occlusion or re-scaling.

1.2 Our project

The goal of this project is to build a camera network in order to show the advantage of the multi-view in having better results for the object detection. Each camera will be taken place all around the studied scene in order to have the most different angle possible.

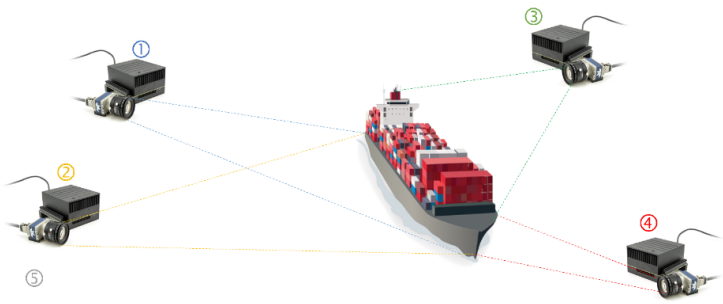


Figure 1: Boat recorded using a multi-view network

In order to have all the information that we need, we have created our own dataset. We chose objects of the everyday life that are big enough to be seen in a large room and marked their positions and orientations in each acquisition.

For the results of the detection, several configurations needs to be tested to put in perspective the role of each parameters. Among them are the algorithm used, the training with and without the orientation of the object and the algorithm used for the decision making.

1.3 Challenges

Despite the subject being very general, the project face technical and social challenges which constrain some choices in the decision of the method used.

1.3.1 Technical challenges

A Neural Network for object detection usually use a lot of computational resources and for real time detection, and are executed on the Graphical Power Units (GPU) of desktop computer. As most of the computations are done in the large amount of convolutions, such tasks can highly be parallelized and take advantage of this

aspect of GPUs. However those devices use a lot of power ; we can take the GTX 10880 Ti as a reference with 250W of power consumption under high loads, while the power consumption seems to increase on most recent GPUs (350W for RTX 3080 TI).

Target	**Power Consumption**	**Benchmark tiny-yolov3**
Desktop with GPU	60 - 500 W	244 FPS (GTX1080 Ti)
Jetson Nano	10 W	48 FPS
Jetson TX2	15 W	144 FPS

Table 1: Power consumptions of different hardwares and benchmark on tiny-yolov3

As seen in the table, both Jetson Nano and TX2 are usable for our project, since they both need less than 20W consumption which is remotely providable and with a frame rate above 30 per seconds, which is enough to meet the requirements for real-time detection.

The tiny-yolov3 algorithm is also relevant since it uses a reduced CNN for real-time results in object detection. It will later be explained why YOLO was chosen amongst other algorithms. The less accurate results of such algorithm are compensated by the multi-view architecture.

1.3.2 Social challenges

Recording objects outside is not free of violating privacy rights. It is indeed possible that people would be filmed without their agreement being given. Sending these pictures to the node making the decision about the final object detection can then be a problem if it is intercepted by some malicious people.

Also in terms of communication between the cameras, sending images from one to another would have been to heavy and thus to slow for a real-time application. This is why it had been decided to only send the output of each Jetson after processing the image, which is only made of 6 numbers.

2 Dataset

2.1 Motivations

Having a multi-view data-set is necessary for training. And having a model trained by multi-view seems to have a high potential for improving object recognition accuracy. At the same time there is not a lot of related work at the moment.

2.2 Related Work

We have a data-set of 40000 images of ships with orientations that have been tagged. However, considering that subsequent tests of our model are difficult to be performed with ships, we decided to find a multi-view data-set. There are some data-sets on the internet, here is one that we found:

`https://www.epfl.ch/labs/cvlab/data/data-pose-index-php/`

This is the data-set of multi-view from EPFL of the cars, which is interesting. But it has only 20 cars with 2000 images which may be not enough for our training. Other than that, this dataset has a single image background. The light intensity is constant. These disadvantages tend to cause over-fitting of the model. But it's still a nice data-set for furthering work.

Another dats-set from ALOI (Amsterdam Library of Object Images) which contains 1000 objects and 100 photos for each object is also very interesting.

`https://zenodo.org/record/6355684`

The variables in this data-set contain different angles with different light as well as the RGB information. However this data-set uses only black background which never changes. And for each class, they have only one object. We also notified that this dataset seems to be generated with animations by PC, this inspires us to use, for example, Unity and other software, writing a script, and constantly automatically generate the scenes, adjust the camera position in Unity, light, then take photo automatically, as a way to obtain a large number of photos with exact orientations.

We then found another dataset that generated scenes with scripts.

`https://data.mendeley.com/datasets/fhzfnwsnzf/2`

This dataset does not include images, but provides all the code and materials used to reproduce the 18,000 scenes and the backgrounds, materials have been changed.However, it has only four classes, and they are mostly animals and classes that do not exist in reality.

There are also many other multi-view data-sets that we don't mention here. But they are not convenient in our case. So we decided to create our own data-set.

2.3 Our dataset

2.3.1 Acquisition Protocol

To have our data-set, we firstly defined our protocol to decide how to do the acquisition as well as the methods of related calculations.

So we firstly defined 7 classes to be included in our data-set which can be easily found in the office: mug, shoe, mouse, book, wallet, keyboard and pencil pouch. For each class we also defined its coordinate system to express their orientation by yaw, pitch, roll. Here are the expressions.

We defined the long side of the book as the y-axis and the short side as the x-axis.

We defined the short side of the keyboard as the y-axis and the long side as the x-axis.

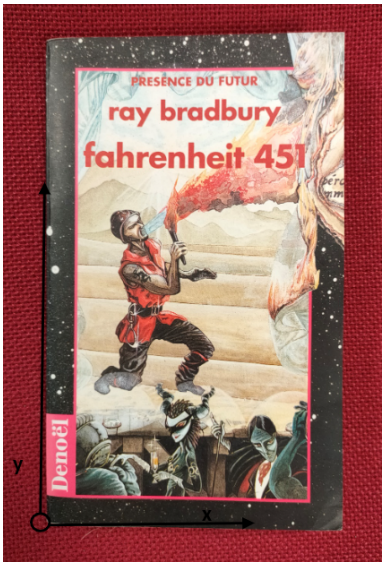


Figure 2: Book

Figure 3: Keyboard

The mouse axis definition is somewhat special, with the scroll wheel as the origin, pointing to the left when placed positively as the Y-axis, pointing forward as the X-axis.

Figure 4: Mouse

The water glass then has the center as the origin, the handle as the X-axis, the Z-axis facing outward, and the left-handed system to establish the corresponding Y-axis.

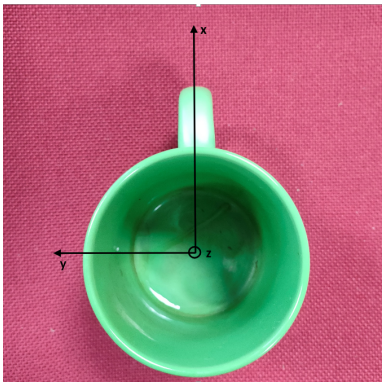


Figure 5: Mug

For pencil bag, the origin is defined as the position of the zipper lock when it is closed, and the Y-axis is along the long axis of the pencil bag.

Figure 6: Pencilcase

The origin of the shoe is defined as at the heel, with the x-axis along the body of the shoe.

The coordinate origin of the wallet is defined as the vertex of the lower left corner when the opening of the wallet is on the right side.Along the bottom edge to the right is the X-axis and up is the Y-axis.

Figure 7: Shoe

Figure 8: Wallet

Here is our rotation representation, and in our context, all objects are referenced by their own coordinate system and the order of the rotation is always yaw first than pitch and roll:

2.3.2 Our dataset without orientation

After the shooting sessions, we firstly obtained a dataset and marked the boundingbox by CVAT. This dataset allows us to train the model to identify our 7 classes, but lacks orientation information, there's only the information about the bounding box and classes in the txt file. And all the files are located in the PC-396 with the account of Zijie.

2.3.3 Our dataset with orientation

In this data-set, we recorded the following information: Class, number of the object: Obj_nbr, yaw, pitch, position. We had 3 rooms to have our acquisition sessions, we also defined the coordinate system for each room, the initial rotation of all objects follows the principle that the objects coordinate system coincide with the coordinate system of the room. The room is pre-selected with points that are measured and marked in advance for the placement of objects to facilitate the calculation of the distance between the cameras and objects if further work needs.

2.3.3.1. The layout of the room

The layout of the room 229 is shown below, the floor of this room has regularly distributed square floor tiles. Each floor tile is 60 cm long and wide, so we use this to measure the distance and mark the point where the

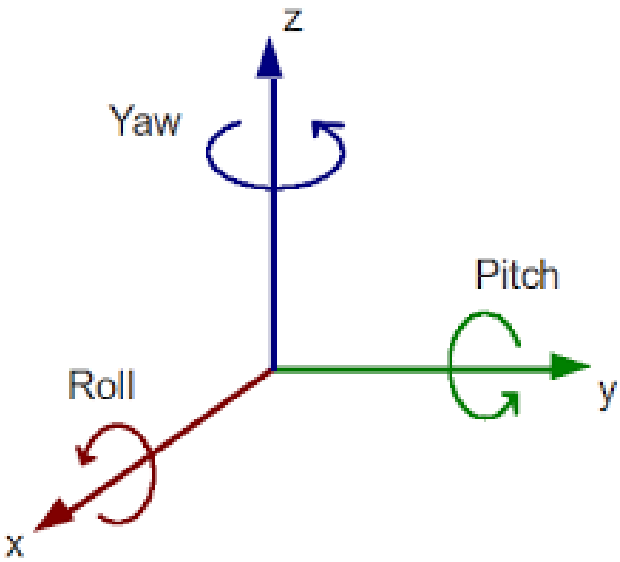


Figure 9: Expression of object rotation

objects are placed. Each square in the picture represents a floor tile in the real room.

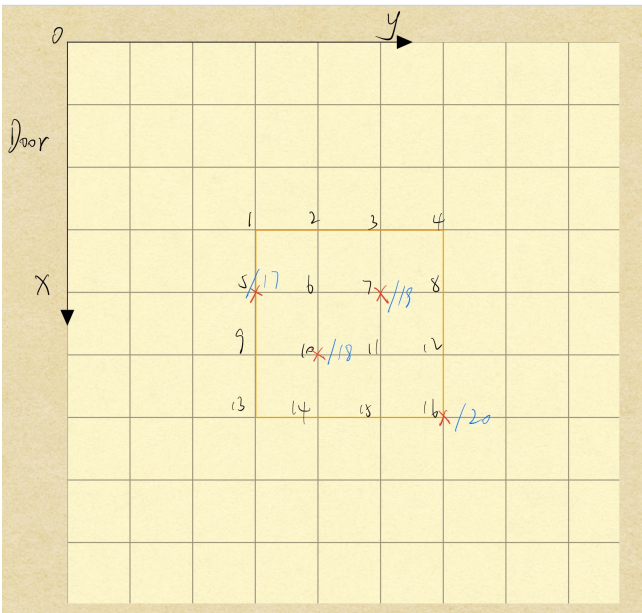


Figure 10: The layout of room 229

The placement of objects is a 4 * 4 square, in which the four painted crosses we add the height variable, that is, if the selected position is 17, 18, 19, 20, then the placement of these positions are 5, 10, 7, 16, but the height of the object placement is no longer at the ground, but of 60 cm height.

2.3.3.2. The job to be done

In the previous part of the unoriented data-set, we already have the information of the bounding box of all the objects. However, the order of the objects in the same image is not taken into account when marking the bounding box. So if we want to train a dataset with orientation, the first task is to correspond the information of the bounding box with the angle information to generate a dataset containing both information.

3 Object Detection Algorithm

In this section we will train the model on our own dataset for later deployment to each camera.

3.1 Overview of algorithms

Deep learning methods for object-detection gained a lot of popularity from 2014. These algorithms can be separated in two main categories: the one-stage methods and two-stage methods.

3.1.1 One-stage and Two-stage methods

The detection algorithm can be divided in two main tasks:

- Find an arbitrary number of objects
- Classify those objects and estimate the bounding boxes

3.1.1.1. Two-stage detectors

Two-stage detectors relies on two separate neural networks that correspond to the two tasks of object-detection.

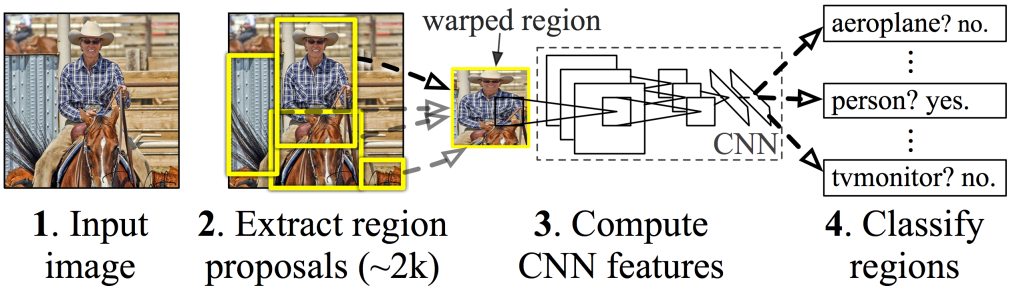


Figure 11: Process of RCNN Breakdown

As explained by the above figure, the first Neural network extract different region on the input image and the second neural network is used to classify the region. Two-stage methods are usually slower than one-stage detectors due to the many inference steps. Besides, the training of this architecture is harder because the two neural networks needs to be trained separately.

3.1.1.2. One-stage detectors

One-stage detectors predict bounding boxes over the image without the region proposal step. Therefore these techniques are faster than the two-stage detectors while achieving great performances.

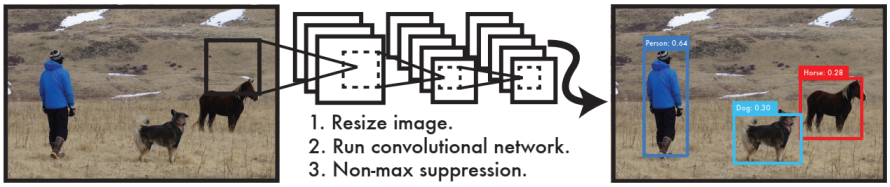


Figure 12: A sample of one stage detection System

3.2 Comparison of different state of the art

In this section, we compared several detectors, especially their speed and accuracy. As mentioned above, the architectures are Maxwell and Pascal respectively, so we focus on the results on these two architectures.

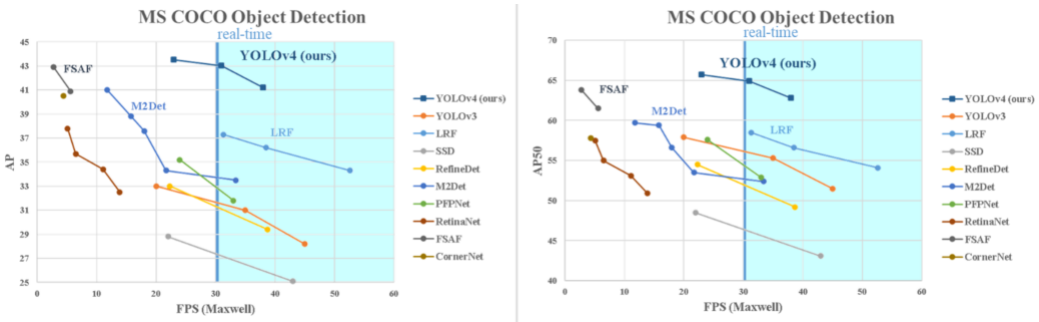


Figure 13: Comparison of the speed and accuracy of different object detectors based on Maxwell

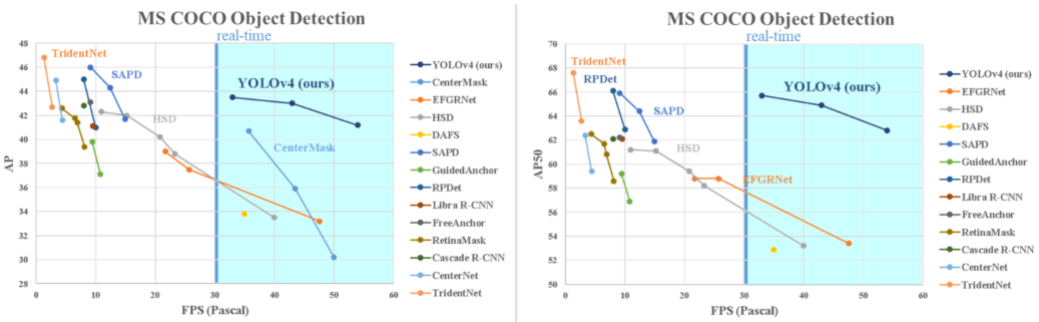


Figure 14: Comparison of the speed and accuracy of different object detectors based on Pascal.

You only look once (YOLO) is a state-of-the-art, real-time object detection system.

It can be seen that as for Maxwell architecture, YoloV4 has the highest AP and AP50 in terms of reaching 30fps, at 43% and 65 % respectively. Compared to other important one stage detectors, such as RetinaNet, which cannot achieve sufficient FPS value, SSD whitch are less accurate at the same FPS value.The performance of the CenterMask detector is notable when using the Pascal architecture, but its accuracy decays rapidly when FPS increases and its overall accuracy is not as good as that of YOLOV4.

Therefore, given the hardware constraints of the system (Jetson Nano, Tx2) and the requirement for real-time object detection, a one-stage detector is required to achieve higher recognition speeds. The YOLOV4 detector was finally adopted as the detector for the system, as well as for the accuracy requirements.

After the paper research and comparison mentioned above, we chose the Yolov4 algorithm. Thanks to its well-established architecture, high accuracy and multiple selectable model depths, we think it is the most suitable for our low-power system.

3.3 How Yolo works

3.3.1 Image grid

First, the image is divided into various grids. Each grid has a dimension of S x S. The following image shows how an input image is divided into grids. Every grid cell will detect objects that appear within them. For example, if an object center appears within a certain grid cell, then this cell will be responsible for detecting it.

Figure 15: Image into grids

3.3.2 Bounding box

A bounding box is an outline that highlights an object in an image.Every bounding box in the image consists of the following attributes:

- Width (bw)
- Height (bh)
- Class (for example, person, car, traffic light, etc.) Witch is represented by letter C
- Bounding box center (bx,by)

The following image shows an example of a bounding box. The bounding box has been represented by a yellow outline.

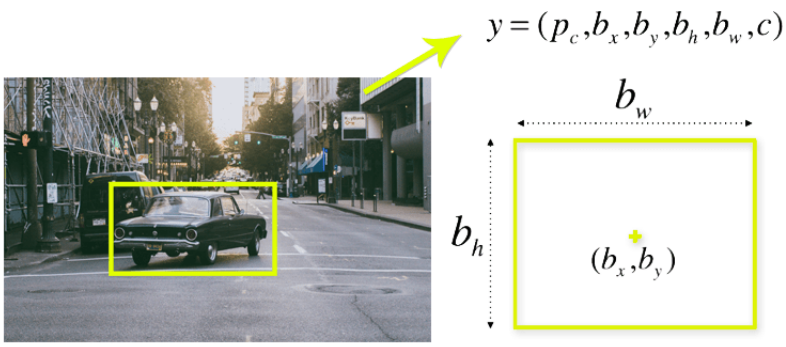


Figure 16: Bounding Box

YOLO uses a single bounding box regression to predict the height, width, center, and class of objects. In the image above, represents the probability of an object appearing in the bounding box.

3.3.3 Intersection over union (IOU)

Intersection over union (IOU) is a phenomenon in object detection that describes how boxes overlap. YOLO uses IOU to provide an output box that surrounds the objects perfectly.Each grid cell is responsible for predicting the bounding boxes and their confidence scores. The IOU is equal to 1 if the predicted bounding box is the same as the real box. This mechanism eliminates bounding boxes that are not equal to the real box. The following image provides a example of how IOU works.

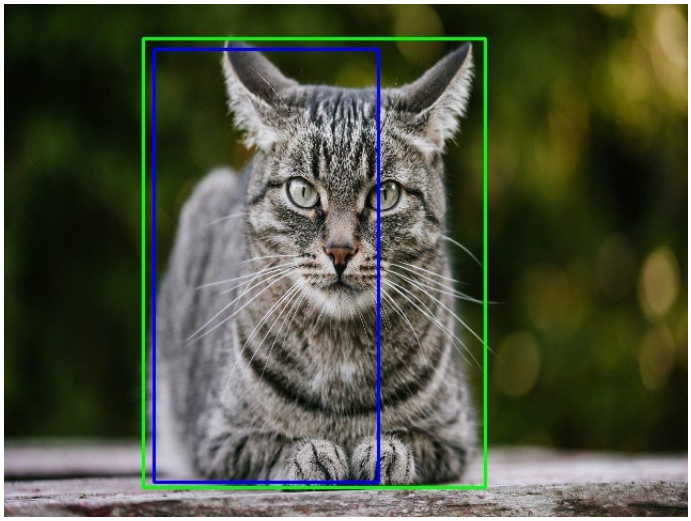


Figure 17: IOU

In the image above, there are two bounding boxes, one in green and the other one in blue. The blue box is the predicted box while the green box is the real box. YOLO ensures that the two bounding boxes are equal.

3.3.4 Combination of the three techniques

The following image shows how the three techniques are applied to produce the final detection results.

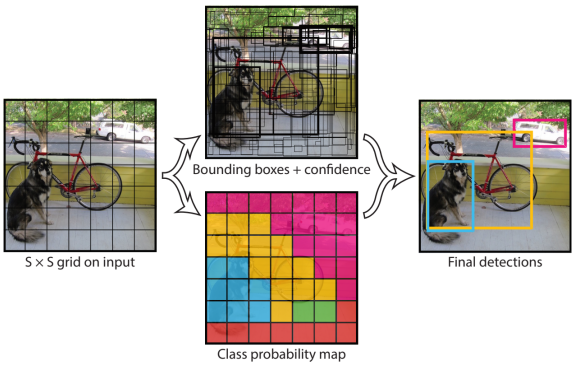


Figure 18: How to get final detectionws

First, the image is divided into grid cells. Each grid cell forecasts B bounding boxes and provides their confidence scores. The cells predict the class probabilities to establish the class of each object.

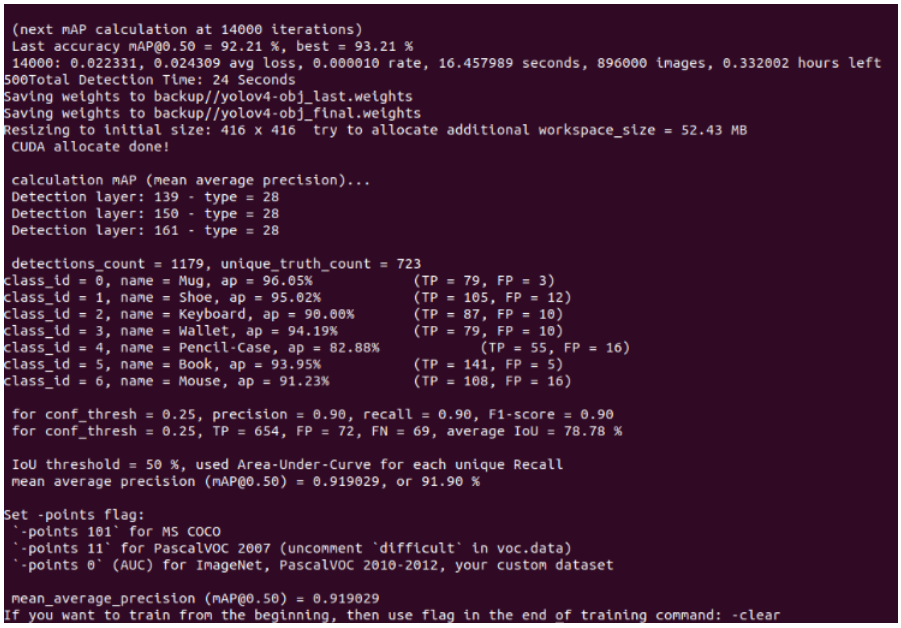
For example, we can notice at least three classes of objects: a car, a dog, and a bicycle. All the predictions are made simultaneously using a single convolutional neural network.

Intersection over union ensures that the predicted bounding boxes are equal to the real boxes of the objects. This phenomenon eliminates unnecessary bounding boxes that do not meet the characteristics of the objects (like height and width). The final detection will consist of unique bounding boxes that fit the objects perfectly.

For example, the car is surrounded by the pink bounding box while the bicycle is surrounded by the yellow bounding box. The dog has been highlighted using the blue bounding box.

3.4 Training on our dataset

We chose the Darknet version of the model and adjusted the relevant configurations according to the tutorial instructions[7]. We randomly select 80% of our dataset as the training set and the remaining 20% as the validation set. Based on the recommendations of the tutorial, we carried out 14000 epochs of training. We performed the training on a computer with an Nvidia GTX 1060 and got the result as shown in the Figure 19.



```
 (next mAP calculation at 14000 iterations)
 Last accuracy mAP@0.50 = 92.21 %, best = 93.21 %
 14000: 0.022331, 0.024309 avg loss, 0.000010 rate, 16.457989 seconds, 896000 images, 0.332002 hours left
500Total Detection Time: 24 Seconds
Saving weights to backup//yolov4-obj_last.weights
Saving weights to backup//yolov4-obj_final.weights
Resizing to initial size: 416 x 416  try to allocate additional workspace_size = 52.43 MB
 CUDA allocate done!

 calculation mAP (mean average precision)...
 Detection layer: 139 - type = 28
 Detection layer: 150 - type = 28
 Detection layer: 161 - type = 28

 detections_count = 1179, unique_truth_count = 723
class_id = 0, name = Mug, ap = 96.05%           (TP = 79, FP = 3)
class_id = 1, name = Shoe, ap = 95.02%          (TP = 105, FP = 12)
class_id = 2, name = Keyboard, ap = 90.00%      (TP = 87, FP = 10)
class_id = 3, name = Wallet, ap = 94.19%        (TP = 79, FP = 10)
class_id = 4, name = Pencil-Case, ap = 82.88%           (TP = 55, FP = 16)
class_id = 5, name = Book, ap = 93.95%          (TP = 141, FP = 5)
class_id = 6, name = Mouse, ap = 91.23%         (TP = 108, FP = 16)

 for conf_thresh = 0.25, precision = 0.90, recall = 0.90, F1-score = 0.90
 for conf_thresh = 0.25, TP = 654, FP = 72, FN = 69, average IoU = 78.78 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.919029, or 91.90 %

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

 mean_average_precision (mAP@0.50) = 0.919029
If you want to train from the beginning, then use flag in the end of training command: -clear
```

Figure 19: Training result

The whole training process took about 40 hours. We obtained the accuracy (mAP) and loss function curves as shown in the Figure 20.

Notice that the accuracy no longer increases in the late training period. We selected the model corresponding to the highest accuracy (a model with 93% accuracy) and applied it to an image and found that the recognition worked well as shown in the Figure 21.

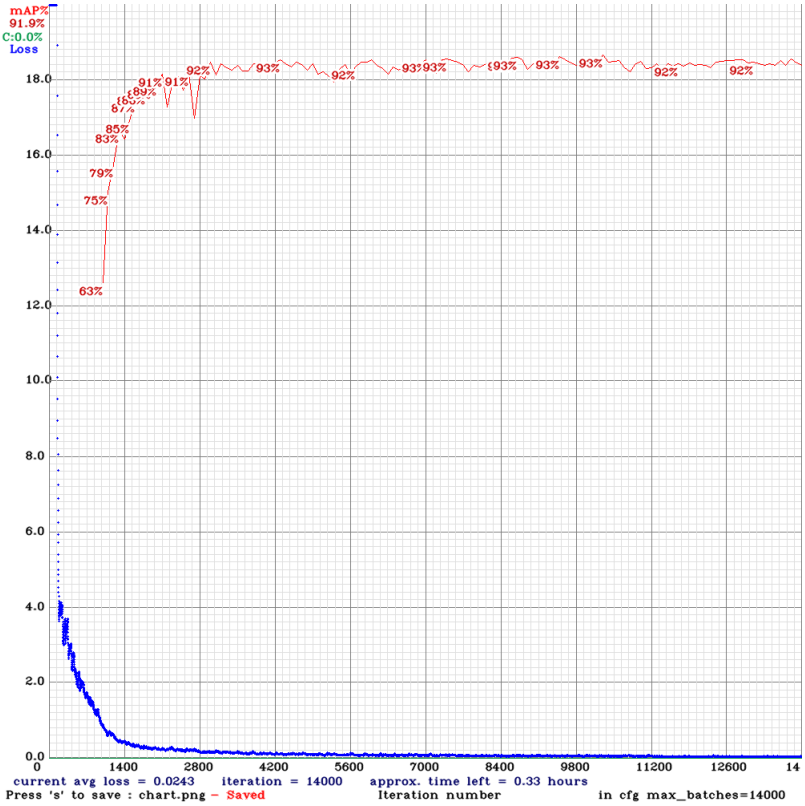


Figure 20: Loss function and mAP curve

4 Multi-view

4.1 Technical Choices

As stated in [1.3.1], we will use Jetson Single-Board Computer as the computation device for the neural network for each cameras. Jetson Nanos are really suitable for our use case, as it has networking capabilities (bluetooth, wifi,...) and are greatly supported due to its wide usage. It also is equipped with a low power GPU and has CUDA support, which can increase the performances of the computation.

The official implementation of YOLOv4 is adapted to the Jetson Nano and has been tested on this device, thus making this device a target of choice for low-power embedded computation card.

For communication, we decided to exploit the TCP/IP protocol as it is very simple to implement and a lot of library exist to use it. We can test the communication by creating dummy message and use the local server address a a way to test the interfaces locally and implement complex communication. Also there is easy ways to implement a graphical interface (Flask SocketIO) which is a great way to debug our code and visualize in real time the prediction of each nodes.

The setup is straight forward as it only needs the devices to be connect on the same wifi network (wireless connection or wired connection) and to specify the manager's IP (host).

The multi-view approach will allow a more reliable prediction as the small version of YOLOv4 has a loss of aabout 20% in mAP compared to the full model(65.7% mAP for full model and 40.2% for tiny-yolov4 on COCO). Thus using a polling system will allow to have more points of view and more sources which will increase the reliability accordingly.

The use of multiview also has certain advantages. First of all, it allows to avoid occlusion problems, which can deteriorate the predictions of the algorithm. For example, in the figure below, the passage of the person in front of the car generally brings a loss in the prediction confidence score. We can also add the fact that the training of the class representations can bring a bias on the image representation: if during the training the classifier is trained on the front representation of the object, it will have difficulties to recognize the back of this object. The addition of the multi-view feature then brings an added value by allowing to meet in a more certain way the faces on which the classifier has been trained.

Figure 21: Prediction of model

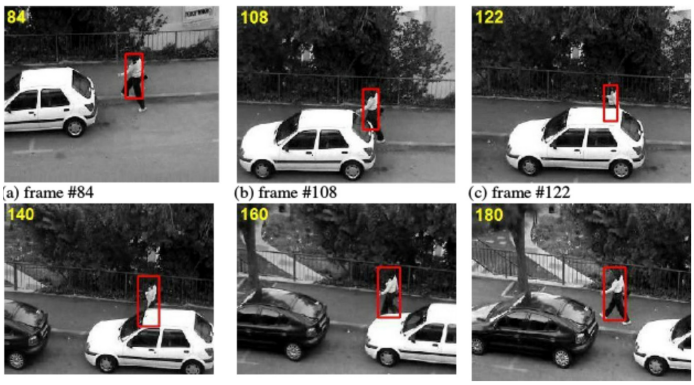


Figure 22: Occlusion

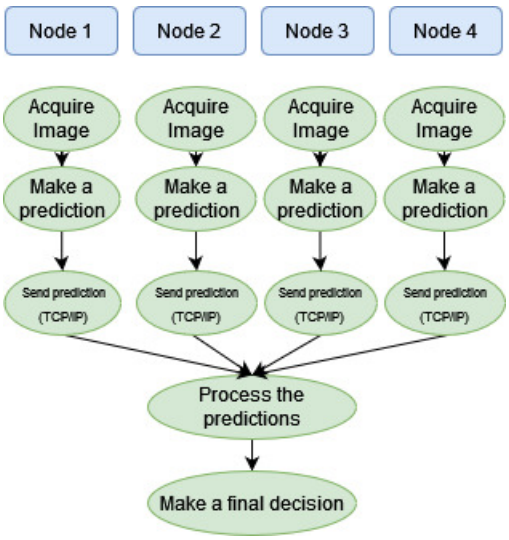


Figure 23: Flow chart for multi-view

4.2 Challenges First Approach

4.2.1 Challenges

The idea of multi-view brings several issues. This problem is similar to that of sensor networks. It is therefore necessary to be able to judge the quality of the predictions of each of the nodes (camera+Jetson): this difference in terms of prediction can be due to several factors. First of all, the difference in points of view, the lighting of the scene from a certain point of view, the distance to the object, the presence of occlusion or not. The decision node should also be able to differentiate predictions and link the prediction of each node. This action can be very complex depending on the scene. If we imagine the presence of several objects of the same class in the scene, differentiating them can be complex. Another possible concern is for the presence of several objects of different classes in the scene, so we will have to be able to avoid confusion between these objects. Moreover, it will be necessary to manage the cases where the object is only visible by a certain number of cameras and thus not to impact the confidence that we have for a certain camera. A first consideration of the problem would be to simply consider the position of the objects during the detection and to link the predictions using the relative positions of the cameras, however this approach could not be implemented due to a lack of time.

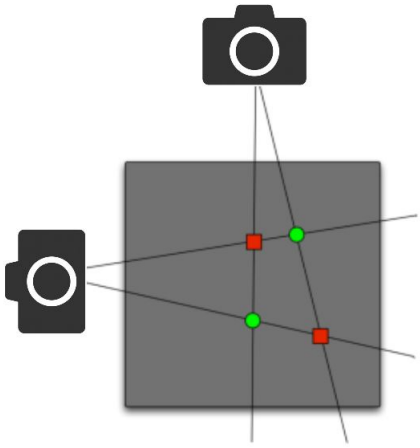


Figure 24: Multi-view: Scene Diagram

We will therefore take as a first approach the case where each camera can see a single object and simply average the decisions of each node.

References

[1] Joseph Redmon, Santosh Divvalay, Ross Girshick, Ali Farhadi, You Only Look Once : Unified, Real-Time Object Detection, arXiv:1506.02640v5 [cs.CV] 9 May 2016.

[2] Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao, YOLOv4: Optimal Speed and Accuracy of Object Detectionj, YOLOv4: Optimal Speed and Accuracy of Object Detection, arXiv:2004.10934v1 [cs.CV] 23 Apr 2020

[3] Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, arXiv:1311.2524v5 [cs.CV] 22 Oct 2014

[4] Kaidong Li, Wenchi Ma, Usman Sajid, Yuanwei Wu, Guanghui Wang, Object Detection with Convolutional Neural Networks, arXiv:1912.01844v1 [cs.CV] 4 Dec 2019

[5] https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/.

[6] https://viso.ai/deep-learning/object-detection/

[7] https://github.com/AlexeyAB/darknethow-to-train-to-detect-your-custom-objects

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

IMT Atlantique Bretagne - Pays de la Loire - www.imt-atlantique.fr

Campus de Brest
Technopôle Brest-Iroise
CS 83818
29238 Brest Cedex 03
T +33 (0)2 29 00 11 11
F +33 (0)2 29 00 10 00

Campus de Nantes
4, rue Alfred Kastler - La Chantrerie
CS 20722
44307 Nantes Cedex 03
T +33 (0)2 51 85 81 00
F +33 (0)2 51 85 81 99

Campus de Rennes
2, rue de la Châtaigneraie
CS 17607
35576 Cesson Sévigné Cedex
T +33 (0)2 99 12 70 00
F +33 (0)2 99 12 70 08