# HW3

ID: D11949001
Name: Miguel Angel Benalcazar Hernandez

**2. Please explain how you implement the non_max_suppression() function. And discuss the results of object detection from the 4 sample images.**

YOLO predict object detection by generating bounding boxes. Most of the bounding boxes overlaps. It is then when Non-Maximum Suppression (NMS) helps us to keep the most probable bounding boxes and deleting the lest-likely bounding boxes. In other words, the NMS delete the redundant bounding boxes only keeping the bounding boxes with maximum score.

The non_max_suppression function requires the following parameters: model predictions, confidence threshold and intersection over union (IOU) threshold.

The first threshold, called the confidence threshold, is a parameter used to filter out predictions that have confidence score below the specified threshold. This parameter can be tuned to get all of those predictions that we consider valid.
This filtering process is done at the beginning of the non_max_suppression function.

```
candidates_mask = prediction[..., 4] > confidence_threshold
for image_number, current_prediction in enumerate(prediction):
      current_prediction = current_prediction[candidates_mask[image_number]]
```

In the code, after filtering and retaining only the most confident detection for each object, the following line calculates the final confidence by multiplying the object confidence with the class confidence.

```
 # Multiply object confidence with class confidence
 current_prediction[:, 5:] *= current_prediction[:, 4:5]  # conf = obj_conf * cls_conf
```

As next step, the boxes need to be converted from x, y, height and width to x1, x2, y1 and y1, using the following line in the code.

```
# Convert bounding box format from [x, y, w, h] to [x1, y1, x2, y2]
      boxes = xywh2xyxy(current_prediction[:, :4])
```

The conversion is done by the function xywh2xyxy already defined in Utility functions.

```
conf, nonzero_confidence_j = current_prediction[:, 5:].max(1, keepdim=True)
      # Concat boxes, confidence,
      current_prediction = torch.cat((boxes, conf, nonzero_confidence_j.float()), 1)[conf.view(-1) >
confidence_threshold]
```

In the previous line, we get the maximum value of all elements in the input tensor, and nonzero_confidence_j represent the class.
In the next part, boxes, confidence and classes are concatenated as well as filtering again by getting the information greater than the confidence_threshold.

In order to complete the NMS, the following part of the code was added to the non_max_suppression function.

```
bbxs = current_prediction.detach().clone()
bestPrediction = []
while bbxs.size()[0] > 0:
   bestPrediction.append(bbxs[0])
   box_iou_Result = box_iou(bbxs[0,0:4].unsqueeze(0), bbxs[:,0:4])
   bbxs = bbxs[box_iou_Result.squeeze() < iou_thres]

outputs.append(bestPrediction)
```

Variable current_prediction is cloned in the new variable bbxs. The algorithm takes the highest score bounding box from bbxs and save it in the variable bestPrediction.
We calculate the IoU (Intersection over Union) between the bounding box selected with respect to the rest of the bounding boxes using a function provided in utilities. The IoU is defined as follows:

$$\mathrm{IoU} = \frac{\mathrm{Area(intersection)}}{\mathrm{Area(union)}}$$

Area(Intersection) represents the area of the intersection of the two bounding boxes and the Area(Union) represents the area of the union of the two bounding boxes. IoU measures whether two bounding boxes overlap or not. The range of the IoU is 0-1.
Once we get the IoU, the bounding boxes are filtered once again. Deleting all of those bounding boxes with IoU less than iou_threshold. Until the bbxs are empty, the procedure is repeated.. Once we have filtered all the bounding boxes we return the filtered data to be displayed over the image.

Results

The following image shows the result of the YOLO without NMS

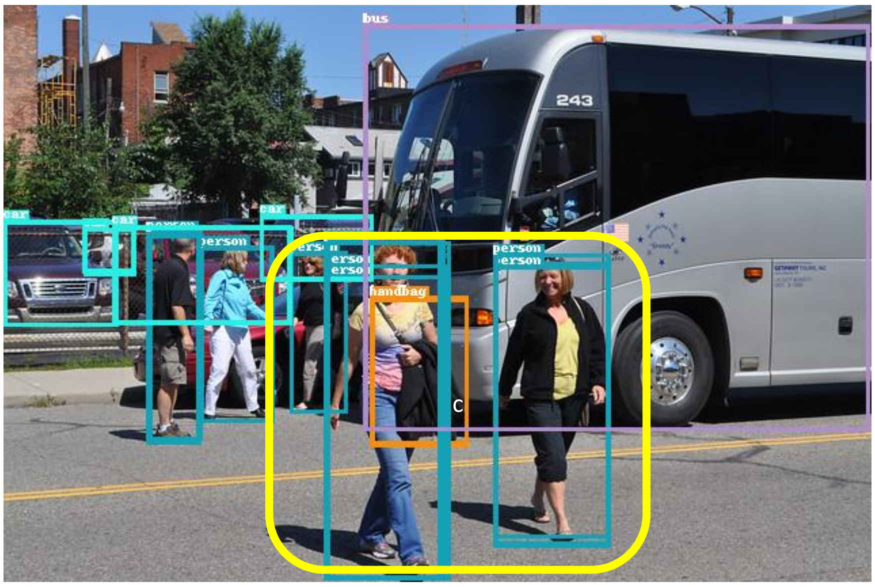

Figure 1. 2.jpg YOLO object detection without NMS

In Figure 1., inside of the yellow square, person has several bounding boxes to detect the same woman. Similarly, the person besides the yellow square shows 2 bounding boxes. These bounding boxes are going to be filtered by NMS algorithm, giving us as result Figure 2.

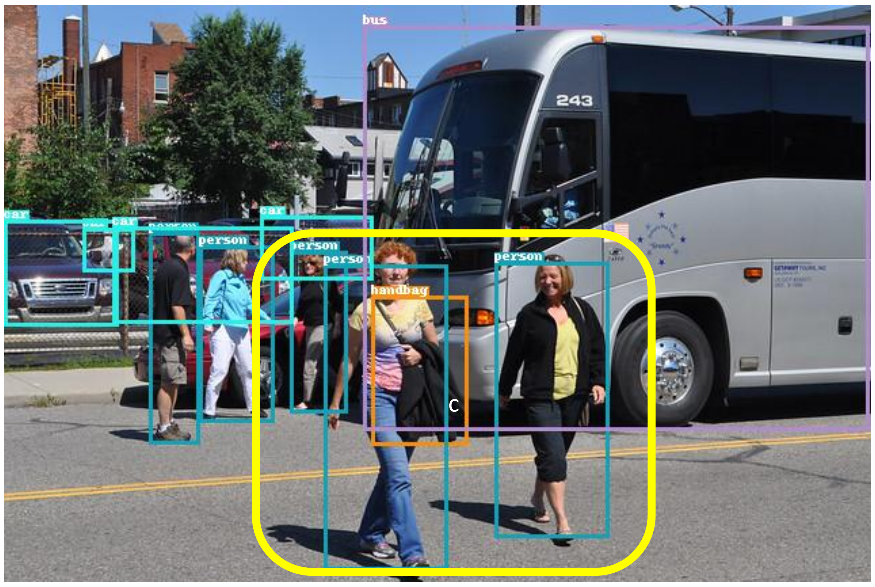

Figure 2. 2.jpg YOLO object detection after NMS

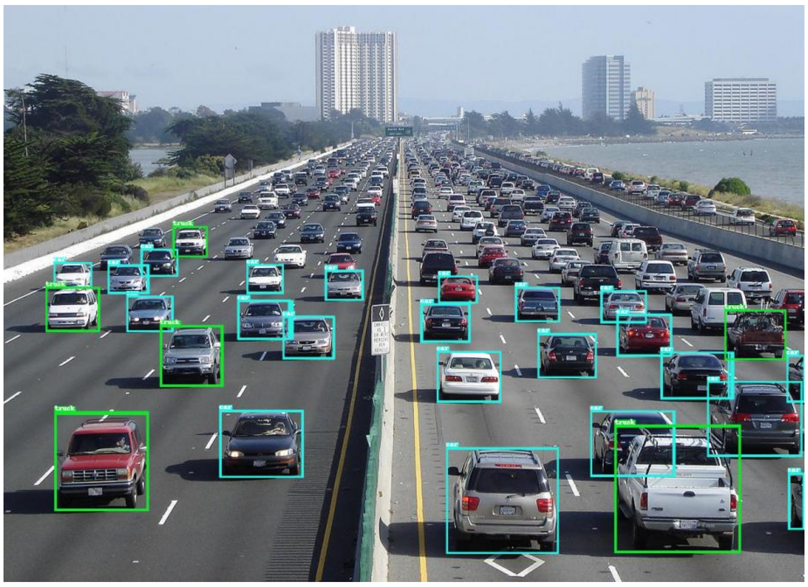
Figure 3. 1.jpg YOLO object detection without NMS

Figure 3 shows the result of the YOLO object detection without non-maximum suppression and Figure 4. shows the result of the same image but using NMS. It is really difficult to spot the differences between these two images, however, the result of the object detection in Figure 3. contains 175 bounding boxes and after NMS is reduced to 28 bounding boxes.

The same is for Figure 5. and Figure 6.. The detection for Figure 5 is 107 and after NMS the bounding boxes are reduced to be 16. This can be easily perceived if we compare Figure 5. and Figure 6.

The object detection for Figure 7. returns 48 bounding boxes and after non-maximum suppression the 48 bounding boxes is reduces 6 shown in Figure 8.

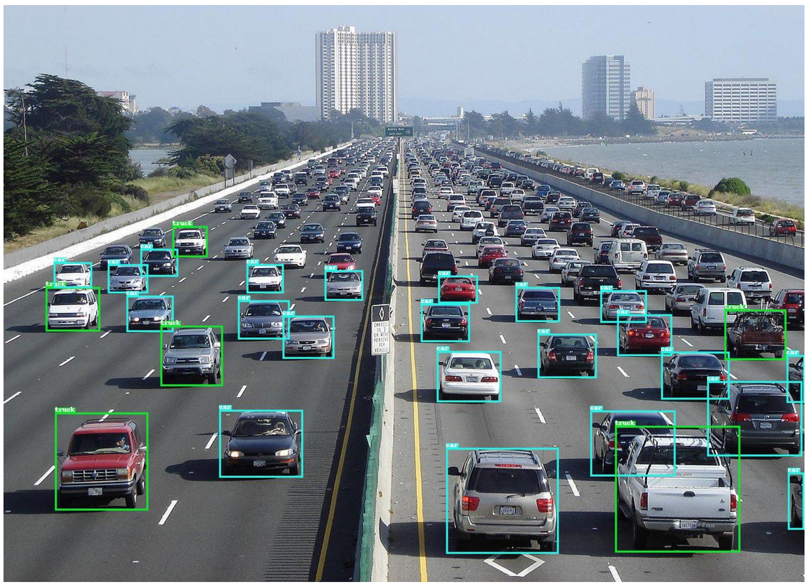
Figure 4. 1.jpg YOLO object detection after NMS

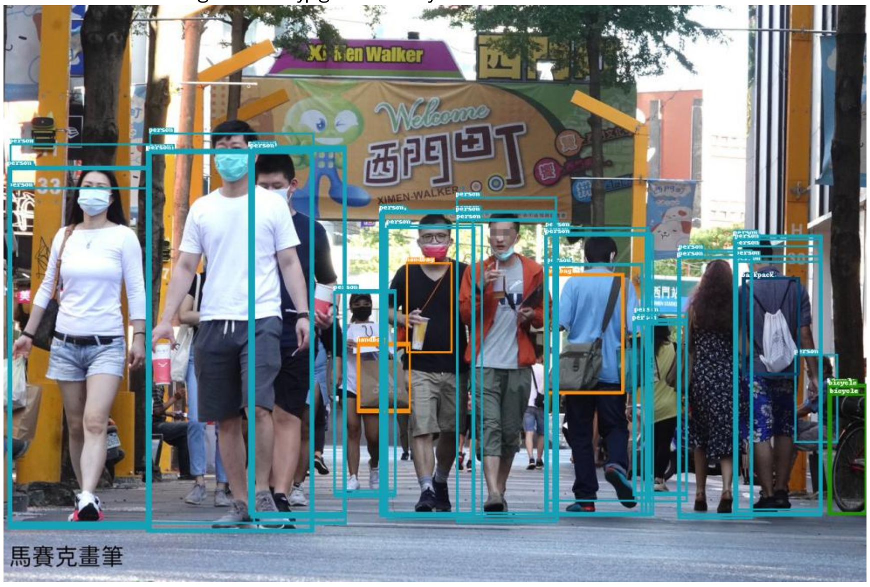

Figure 5. 3.jpg YOLO object detection without NMS

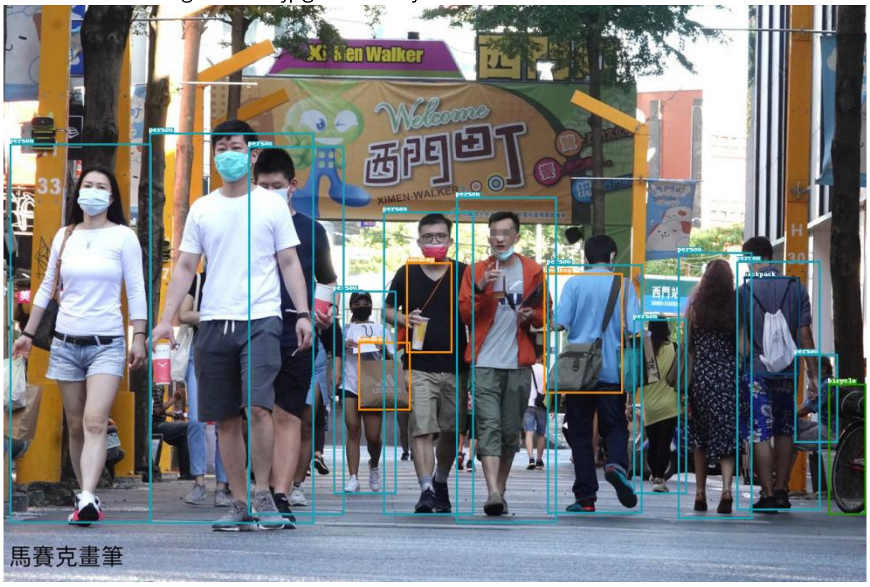

Figure 6. 3.jpg YOLO object detection after NMS

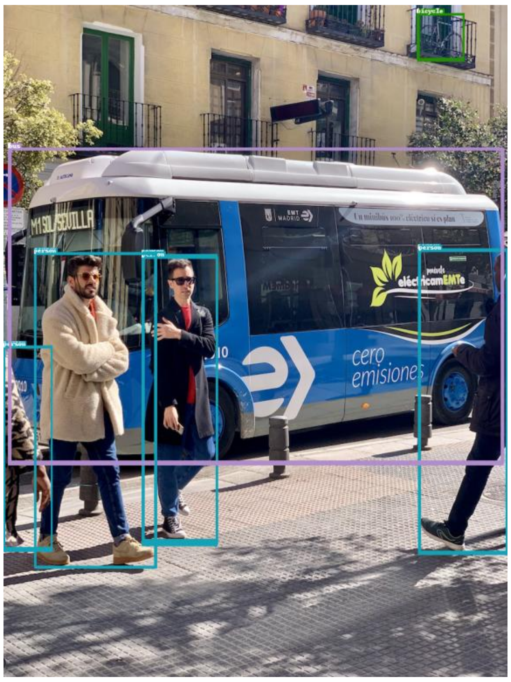
Figure 7. 4.jpg YOLO object detection without NMS

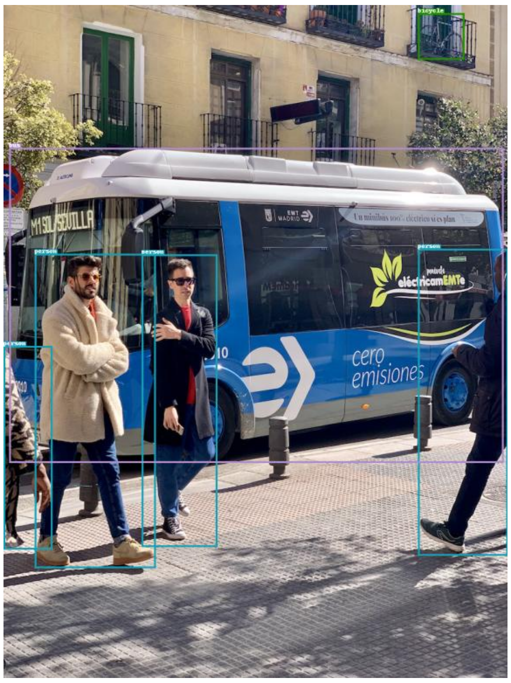
Figure 8. 4.jpg YOLO object detection after NMS

However, it looks like the YOLO object detection sometimes does not detect some object due to the position of the object or its size. The object might be partially blocked by another or it might be too small. Furthermore, sometimes  object was detected by YOLO have low score of prediction that is filtered in the non-maximum suppression part.

**3. Try to adjust the conf_thres and iou_thres parameters and observe how they affect the final predictions. Provide your observations. (Default: conf_thres=0.4, iou_thres=0.6)**

To demonstrate how the conf_thres and iou_thres parameters affect the results of the object detection 1.jpg from the data set was chosen.

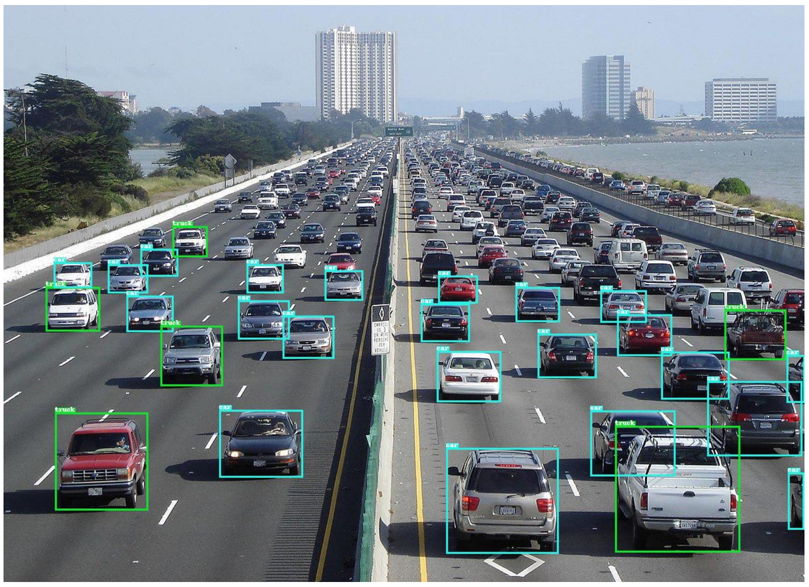
Figure 9.  conf_thres=0.4, iou_thres=0.6

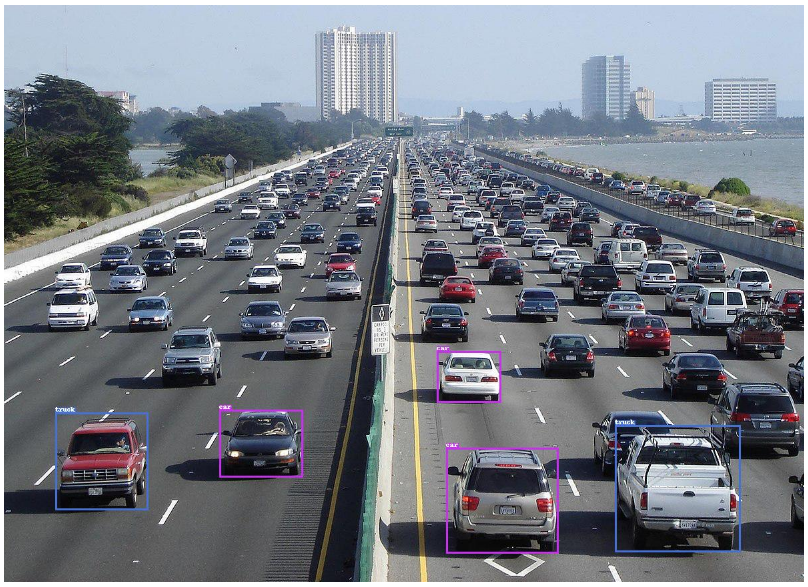

Figure 10.  conf_thres=0.75, iou_thres=0.6

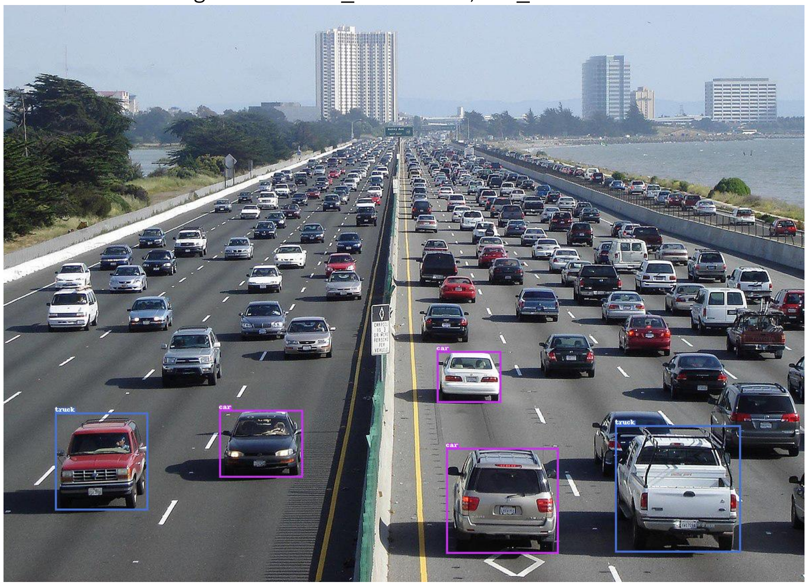

Figure 11. conf_thres=0.75, iou_thres=0.8

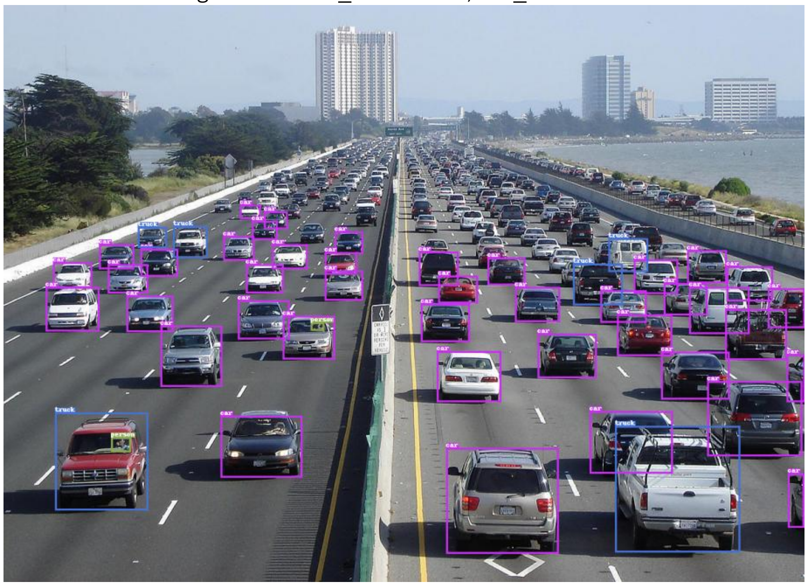
Figure 12. conf_thres=0.1, iou_thres=0.3

Figure 9. shows the result of the YOLO object detection using the default conf_thres=0.4 and iou_thres=0.6. This figure will be our base to compare with results with different thresholds.
If we compare the result from Figure 10. and Figure 9., we can easily perceive that the bounding boxes were reduced significantly. conf_thres was set to be 0.75, this means that all of those detection with confidence score less that 0.75 will be filtered, leaving just those bounding boxes with high probability.

For Figure 11., the confidence threshold was set to be 0.75 as the previous test while the iou threshold was set to 0.8 this time. As it can be seen at Figure 11., there are not changes with respect to Figure 10. This happens because the iou threshold just filters those redundant bounding boxes without taking into consideration the confidence score.

The confidence threshold was reduced to be 0.1 which filters all of those confidence scores less than 0.1. The result of this experiment is shown in Figure 12. If Figure 12. is compared with Figure 9., we can find that Figure 12. shows more bounding boxes, which means that we are seeing almost all of the detection made by YOLO. Since the threshold is set to be 0.1, it is just suppressing those bounding boxes with lower confidence score than 0.1. Thus, NMS is just filtering some noises or perhaps miss-detection.

## 4. The methods for object detection can be divided into one-stage and two-stage approaches. What type of object detection method does YOLO belong to? What is the difference between one-stage and two-stage methods, and what are their respective pros and cons?

YOLO stands for You Only Look Once, and it is a one-stage object detection algorithm.

*One-stage approach*
The algorithm only passes once over the image making predictions. It means that by a single pass, the algorithm detects and locates the objects in the image. This method is also known as single-shot object detection that processes the entire image in a single pass.

*Pros:*
Since the One-stage approach checks the entire image just once, the object detection is much faster than the two-stage object detection algorithm, thus, it is computationally efficient and fast. In terms of architecture, a one-stage approach is more simple than implementing the two-stage approach. It tends to be more flexible, since it can be implemented in less powerful devices in comparison with the two-staged object detection.

*Cons:*
The accuracy is affected since this method does not have the opportunity to refine their predictions by examining multiple region proposals.
One-stage object detection algorithms are not as good at detecting objects that are partially hidden or surrounded by other objects as two-stage object detection algorithms. This is because one-stage algorithms only make one prediction for each object, while two-stage algorithms make multiple predictions and then refine them

*Two-stage approach*
On the other hand, the two-stage approach passes twice over the image to perform object detection. The first time that the algorithm analyzes the image, generating a set of the proposal of potential object locations. The second pass refines the proposals obtained in the first stage and finally makes the final predictions

*Pros:*
The two-stage approach is more accurate. This is because this algorithm generates a set of proposals at its first stage and the second stage refines the proposal and makes final predictions. On the other hand, two-stage object detection algorithms are more robust when we need to identify objects that are partially hidden or surrounded by other objects.

*Cons:*
These methods are slower than the one-stage approach. Since these methods inspect the input image twice. First to generate the proposal and second to refine the proposals and make the final prediction. Last, these methods are complex in terms of training and implementation. To perform a real-time prediction, these methods require to be implemented in power devices.

**5. Please draw the architecture of YOLOv7. Explain how each component is designed and its functionality.**

YOLOv7 model is a single stage object detector. YOLOv7 is formed by 3 stages that are:
*Backbone:* This stage is in charge of extracting the essential features from the image.
*Neck:* This stage combines all of the features maps from the previous stage and creates features pyramid.
*Head (Dense Prediction):* information created by the neck is used in this stage to predicts the locations and classes of the objects to create the bounding boxes.

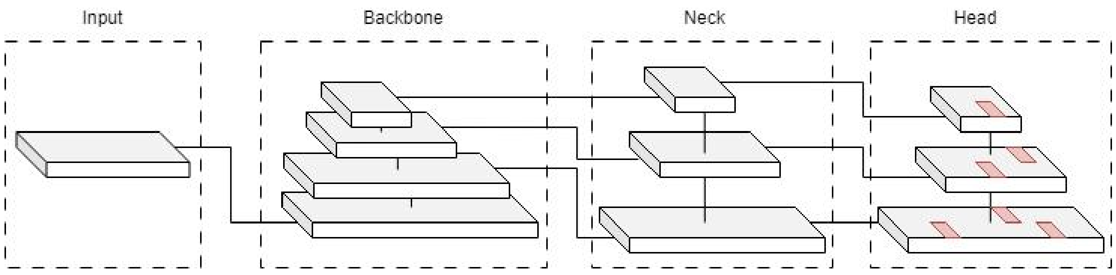

Figure 13. YOLO base architecture

The architecture of the YOLOv7 is derived from YOLOv6 and YOLOR.

<u>*Extended efficient layer aggregation networks*</u>

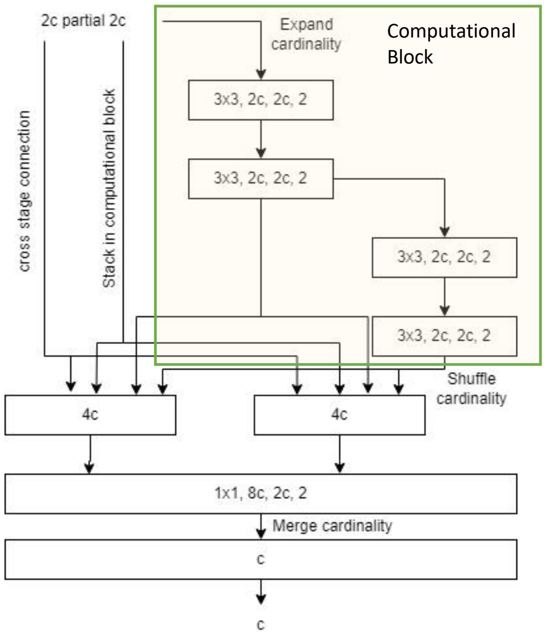

Figure 14. E-ELAN

E-ELAN is an extension of the ELAN and is the way for optimizing the feature extraction in the backbone stage. E-ELAN was designed to increase the network's capacity for learning while preserving gradient path stability and parameter use rate. This improves the network's capacity to learn by enlarging the computing blocks, rearranging and joining feature maps, and using merge cardinality. Figure 14. shows the extended efficient layer aggregation network which is the base of the backbone.

*Model scaling for concatenation-based models*

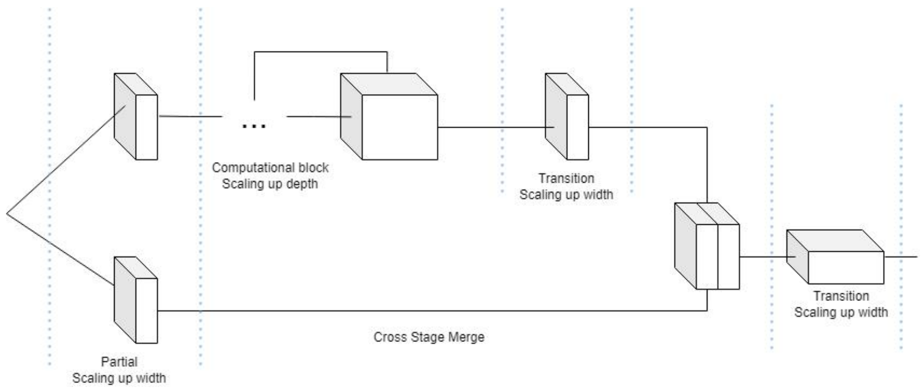

Figure 15. Compound scaling up depth and width for concatenation-based model

Scaling regulates a number of the model attributes to generate comprised models of different scales to increase different inference speeds. The compound scaling in YOLOv7 is robust since it keeps the characteristics of the model's original design and maintains the optimal structure as it can be seen in Figure 15. The compound scaling shows that if the depth factor of a computational block is adjusted, it is also necessary to determine the change in the output channel of that block. Furthermore, to maintain consistency, it is necessary to scale the transition layers' width factor by the same amount of change. The compounding scaling is part of the neck stage.

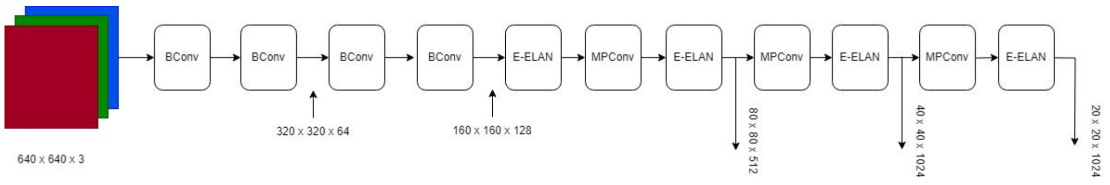

Figure 16. YOLOv7 Backbone

Figure 16. shows the backbone of YOLOv7. It is composed of several BConv that are displayed in Figure 17., E-Elan layers (Figure 14) and MPConv Layers (Figure. 18). The BConv layer is composed of a convolutional layer + a Batch Normalization layer + an activation function ReakyReLu. On the other hand, MPConv halves the length and width, doubles the channels, and extracts features.

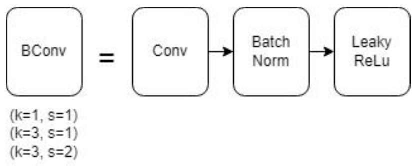

Figure 17. BConv = (Convolution + Batch Normalization + Leaky ReLu

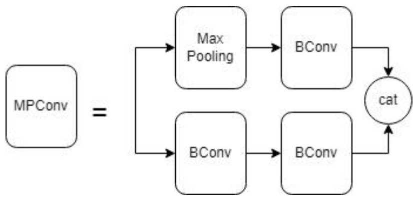

Figure 18. MP Convolution Layer

Figure 19, represents the combination of the neck and head in the YOLOv7 architecture. It is composed of SPPCSPC layer (Figure 20.), several BConv layers, several MPConv layers, several Catconv layers, and the RepVGG block layer that subsequently outputs three heads.

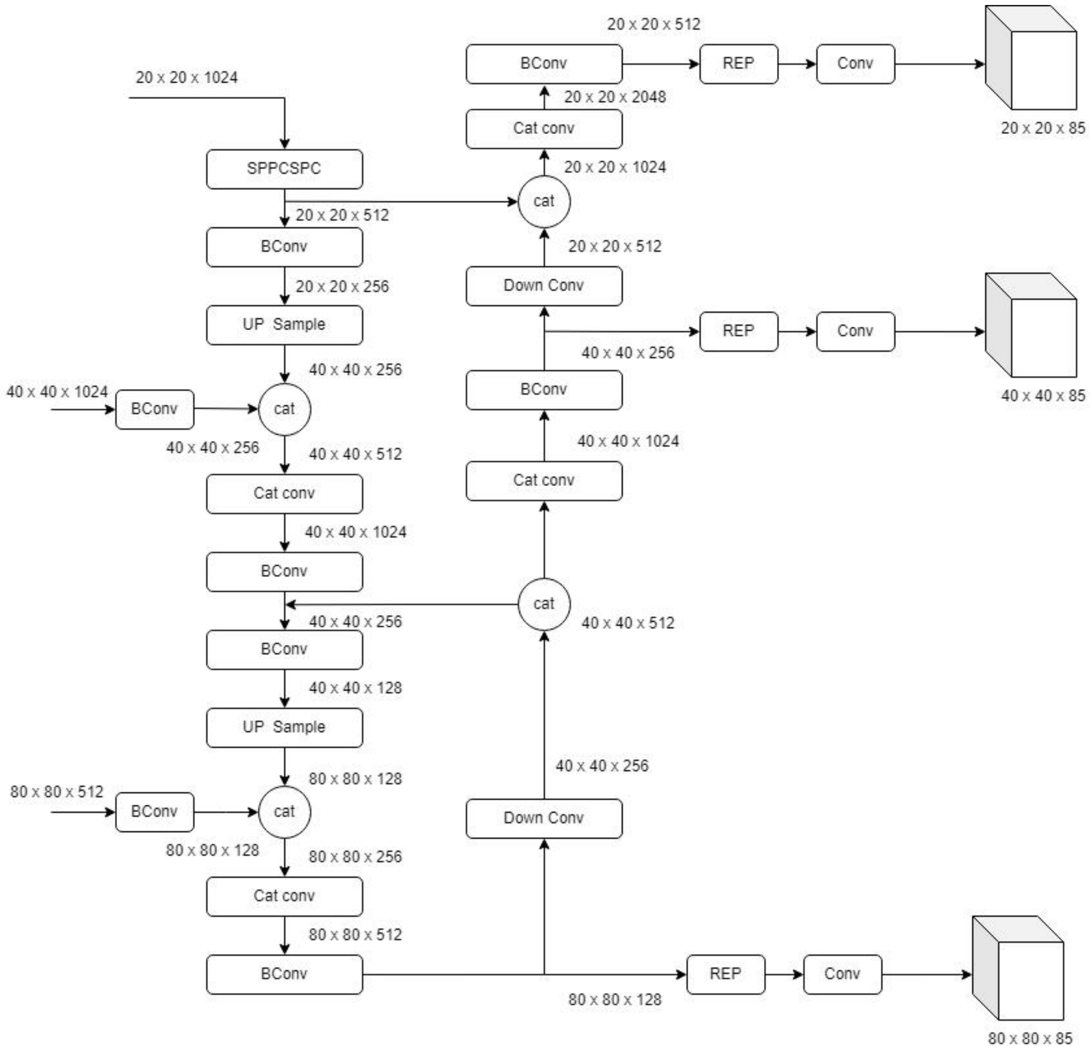

Figure 19. Neck and Head YOLOv7

A feature pooling layer called the SPPCSPC (Spatial Pyramid Pooling Cross Stage Partial Convolution ) layer is used to extract multi-scale data from the input feature map. It combines the Spatial Pyramid Pooling (SPP) and the Cross Stage Partial Convolution (CSPC). This layer enhances the network's ability to handle objects of different sizes and improves the efficiency and effectiveness of feature extraction.

Hence, it makes multi-scale feature pooling possible and improves the network's capacity to accurately and efficiently recognize objects of different sizes.

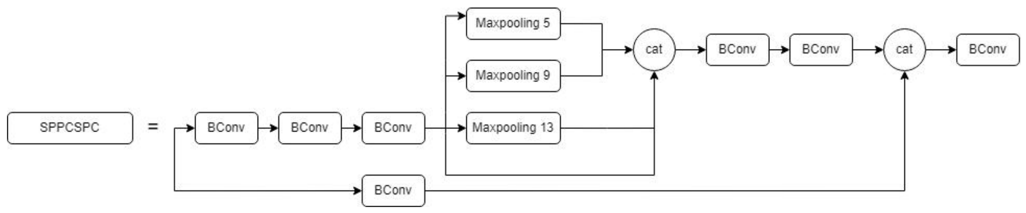

Figure 20. SPPCSPC layer

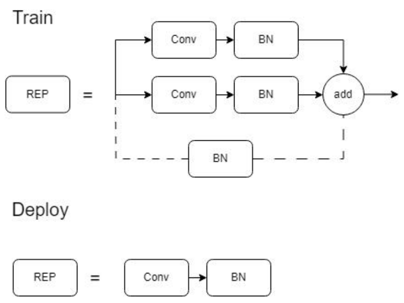

Figure 21. REP layer YOLOv7

REP is shown in Figure 21., as it can be seen, two REP are shown, the first is for training and the other is for deploying. This layer stands for re-parameterized convolution which introduces a new design that achieves the benefits of both residual connections in ResNet and concatenation in DenseNet while providing increased gradient diversity for different feature maps.

At the end of the head, 3 outputs are presented. The first output (20 x 20 x 85) represents the smallest scale, the second output (40 x 40 x 85) represents the medium scale, and the third output (80 x 80 x 85) represents the largest scale. Each of the above outputs contains a set of bounding boxes and confidence scores for each object in the image. The bounding boxes are predicted according to their center coordinates, width, and height, relative to the size of the input image. Hence, having the three outputs, allows YOLOv7 to handle occluded objects that might not be visible at one scale. This is a key part of the model's ability to detect objects at a wide range of scales and with high accuracy. In the end, these outputs are combined and by using the Non-Maximum Suppression algorithm get a filtered output with the detected object highlighted by a bounding box.

## 6. Please survey one of the other object detection methods ( e.g, CNN-based, transformer-based, one-stage or two-stage approaches, other versions of YOLO, etc.) and briefly explain it. Also, compare its pros and cons with YOLOV7.

For this part, I have decided to use a two-stage method Fast R-CNN. Fast R-CNN is an old object detection algorithm proposed in 2015. As it was mentioned before, the two-staged method first generates a region proposal and the then classifies each proposal as either containing an object or not containing an object.

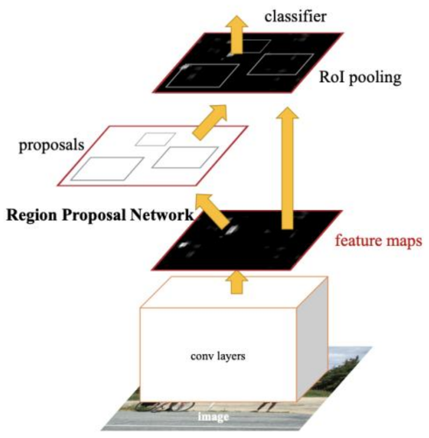

Figure 22. Faster RCNN architecture

The main components of the Faster R-CNN are shown in Figure 16.
- Region Proposal Network (RPN): This is a NN module that generates potential object regions or proposal regions in an image. A small window called an anchor slides over the convolution feature map, predicting whether the anchor contains an object or not. One of the advantages of (RPN) is that it does not only generate the region proposals but also adjusts the coordinates to align with objects.
- Region of Interest (RoI) Pooling: This is the last stage of the fastest R-CNN. This layer extracts fixed-size feature maps from the convolution feature map for each RPN.

The RoI features are fed into a network that does two operations: bounding box regression and object categorization, which refines the precise position of items inside of the suggested area.

At this point, Non-maximum-suppression is used to remove overlapping detection. As it was already mentioned, NMS is an algorithm that is used to identify the most confident detection for each object class. By removing overlapping detection, NMS ensures that the algorithm only outputs detection that is confident and accurate.

Let’s compare the pros and cons of using Faster R-CNN and YOLOv7

Faster R-CNN
- Pros:
    - More accurate
    - robust to noise
    - robust to occlusion
    - handle a wider range of object sizes and scales
    - two-stage object detection
- Cons:
    - Slower than YOLOv7
    - Hard to train
    - requires high computing power
    - Low frame rate detection

YOLOv7
- Pros
    - Faster than Faster R-CNN
    - One-stage object detection
    - Simpler to train
    - Can be used in real-time applications
    - can be implemented in devices with low computing power
- Cons:
    - Less accurate than Faster R-CNN
    - Less robust to noise
    - Less robust to occlusion