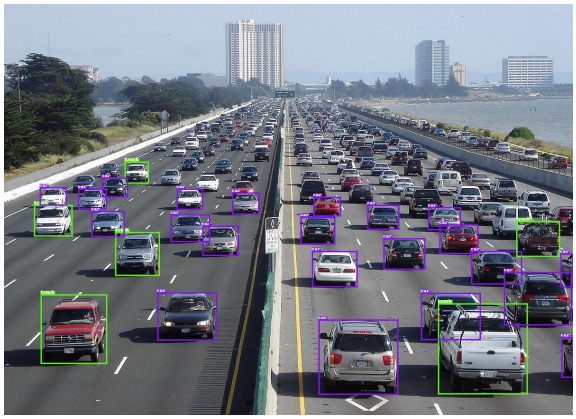
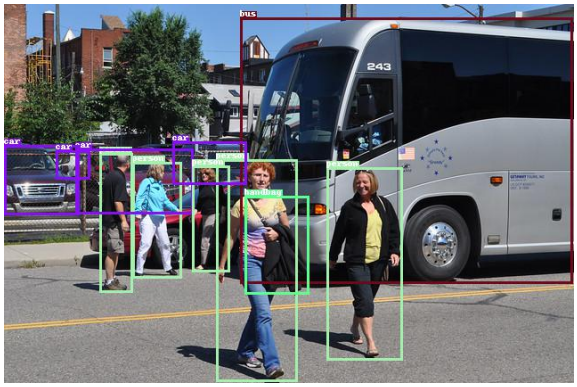
HW3 - Report

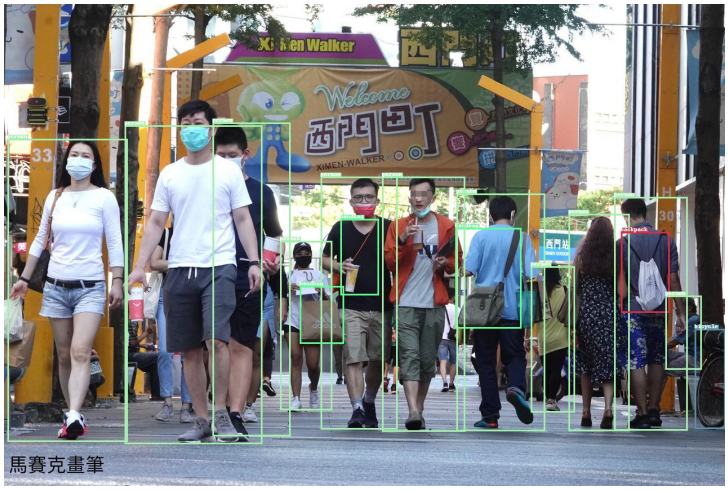
- Department: Master program of artificial intelligence, computer science and information engineering depart
- ID Number: r11922a05
- Name: 林聖硯

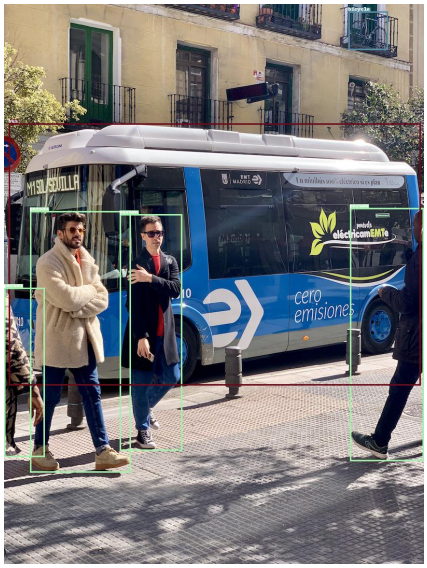
1. Implement the non_max_suppression() function in the sample code
2. Please explain how you implement the non_max_suppression() function. And discuss the results of object detection from the 4 sample images. E.g., why some objects are detected and others are not or misclassified.

- How I implement the function: I use the confidence score of model prediction to sort the images first, and then pick the bounding box with the highest confidence score. After that, I calculate IoU between the current box and the remaining boxes, then remove boxes with high overlap (IoU) using the specified threshold.
- Observation
  - In `1_pred.jpg`, there are many cars that were not detected in the end, which could be attributed to the default confidence threshold and IoU threshold being set too high (0.4 and 0.6 respectively). Additionally, the objects (cars) are relatively small or heavily obscured, causing the model to assign them lower confidence scores.
  - In `3_pred.jpg`, the model incorrectly classified the man wearing a white shirt and inaccurately determined the y-axis of the bounding box (it only includes the region with his mask, not the entire head). This could be because the model believed that the mask or the surrounding region contained sufficient visual cues or features for object classification, resulting in misclassification and an inaccurate representation of the bounding box.







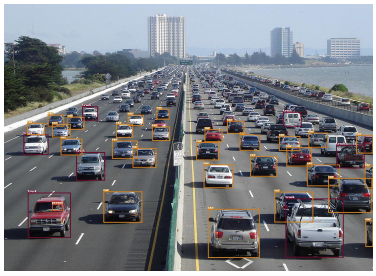


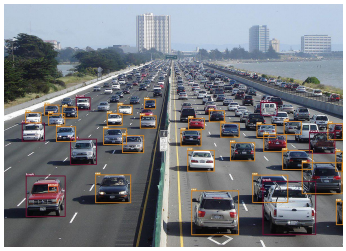
3. Try to adjust the conf_thres and iou_thres parameters and observe how they affect the final predictions. Provide your observations.

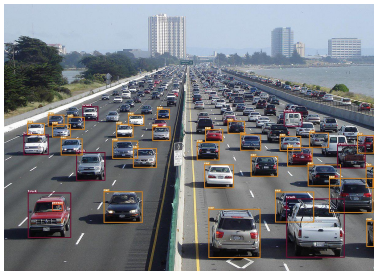
First I chose, the first picture (`1_pred.jpg`) since it seems to me that it's the most difficult picture for the model to correctly detect all the objects.

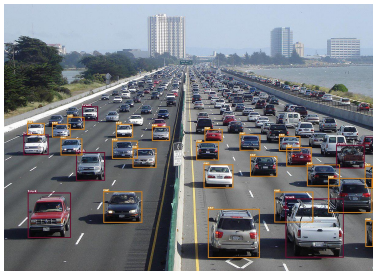
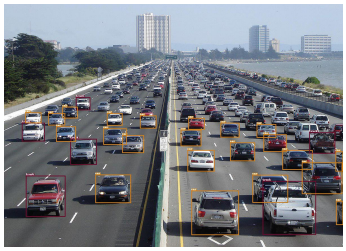
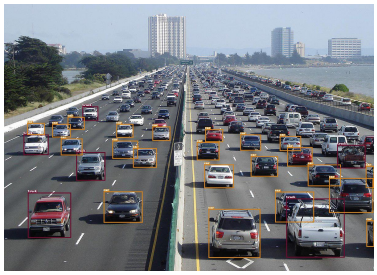
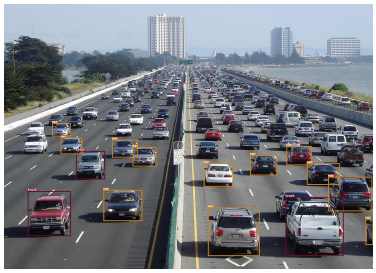
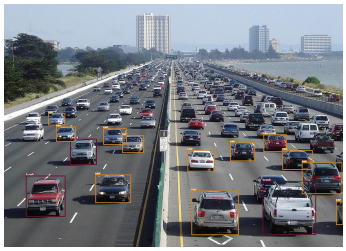
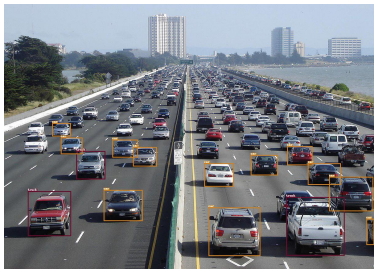
From left to the right (IoU = 0.4, 0.6, 0.8)

From top to the bottom (confidence = 0.2, 0.4, 0.6)







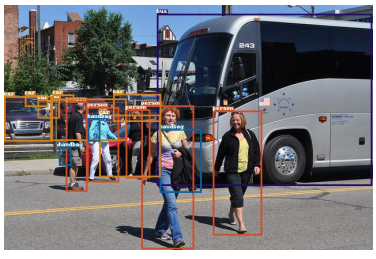
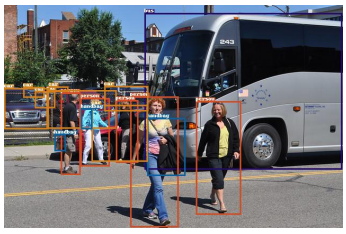
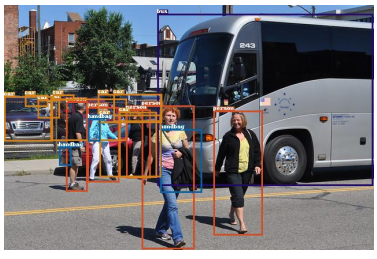







- Observation: From the pictures, it is evident that adjusting the confidence threshold significantly affects the final predictions. When the confidence threshold is set high, the number of correctly predicted cars decreases. Conversely, lowering the confidence threshold leads to more cars being detected. On the other hand, the IoU threshold appears to have a minimal impact on the predictions. This is likely because the predicted bounding boxes are close to each other, indicating that the model can accurately determine the position of the cars in the image. However, the model may not have a high level of certainty regarding the correctness of its predictions.

I also did a case study on `2_pred.jpg` to find the effect of adjusting the conf_thres and iou_thres parameters.

From left to the right (IoU = 0.4, 0.6, 0.8)

From top to the bottom (confidence = 0.2, 0.4, 0.6)

- Observation: From the top row (confidence = 0.2) of pictures, it is evident that when there are numerous bounding boxes and a high IoU threshold, the final result consists of many overlapping bounding boxes. This could be attributed to the presence of numerous objects of the same class overlapping with each other, but the degree of overlapping is not significant enough for the non-maximum suppression algorithm to remove the excess bounding boxes.

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 belongs to the **one-stage object** detection method.

In one-stage methods like YOLO, the detection network directly predicts the bounding box coordinates and class probabilities for a fixed set of predefined anchor boxes across the entire image in a single pass. Therefore, the model is trained in an end-to-end way. This makes the inference process faster compared to two-stage methods.

On the other hand, two-stage methods, such as Faster R-CNN, have a two-step approach. The first stage generates region proposals using a region proposal network, which proposes potential regions of interest. In the second stage, these proposed regions are classified and refined using a classification network.

	One-stage	Two-stage
Pros	1. simpler to implement and understand 2. faster inference times since they perform detection	1. usually achieve higher accuracy, especially for small objects or objects

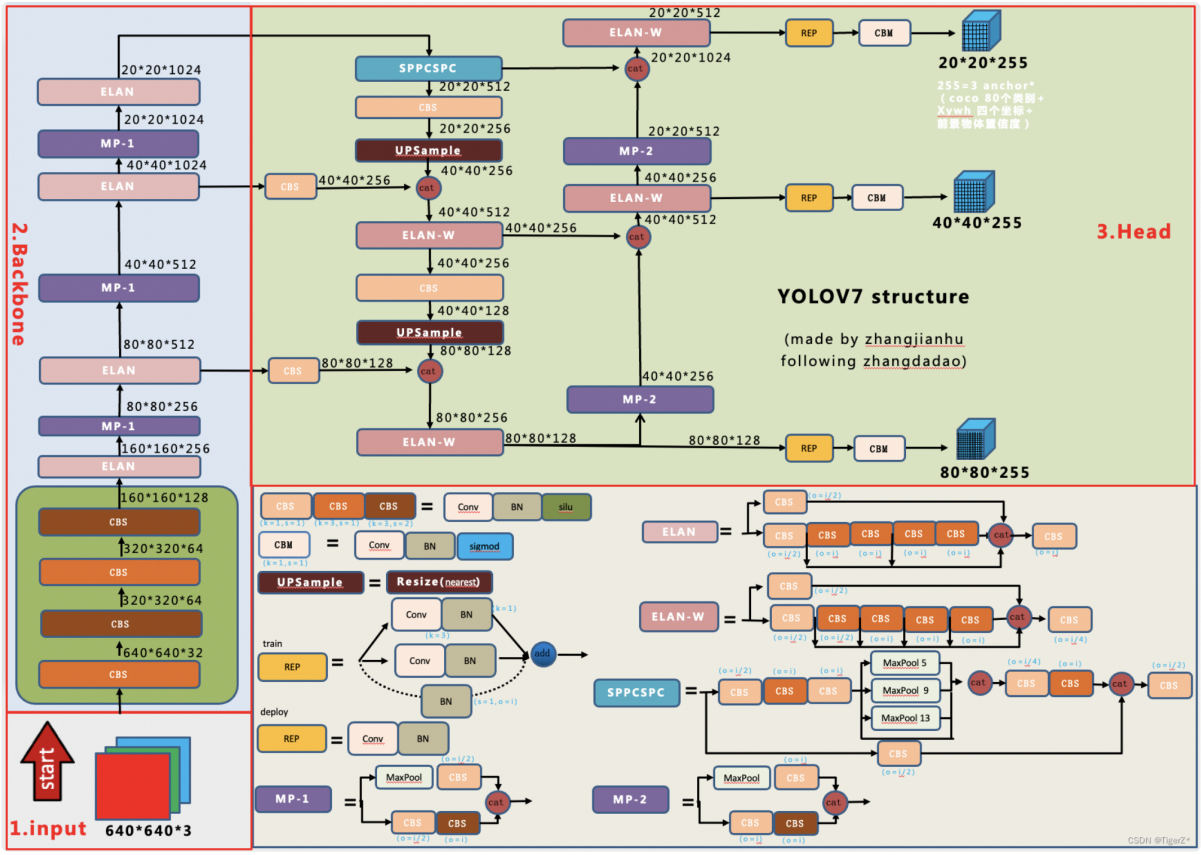
	in a single pass. 3. well-suited for real-time applications due to their fast processing speed.	with complex backgrounds 2. better handling of objects at different scales
Cons	1. may have lower localization accuracy compared to two-stage methods, especially for small and densely packed objects 2. may struggle to handle objects with a wide range of sizes since they rely on fixed anchor boxes	1. have a more complex architecture, with separate region proposal and classification stages 2. requires additional computation for region proposal generation, making it slower compared to one-stage methods 3. usually more time-consuming due to the separate stages and multiple network components

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

Reference: https://blog.roboflow.com/yolov7-breakdown/

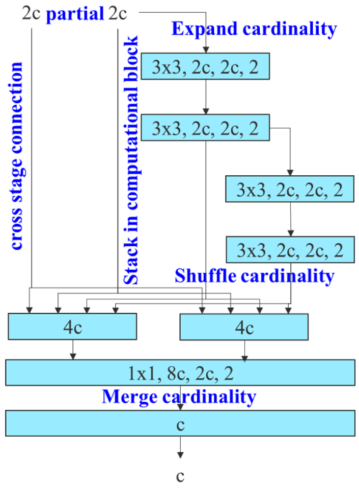
The architecture is based on YOLOv4, Scaled YOLOv4, and YOLO-R with the following modification. In general, the model architecture of YOLO family can be divided into three main components: Backbone, Neck, and Head.

Picture reference：https://blog.roboflow.com/pp-yolo-beats-yolov4-object-detectio



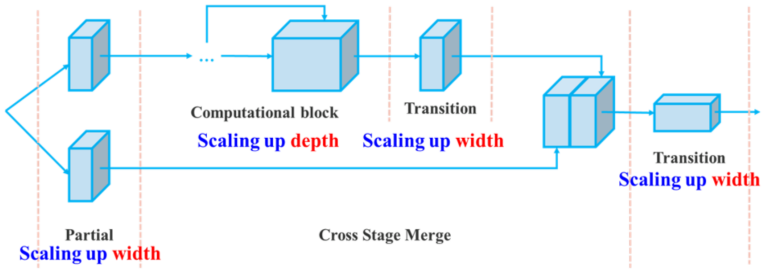
In addition, the authors of YOLOv7 propose the following improvements to the architecture:

(a.) E-ELAN (Extended Efficient Layer Aggregation Network): The authors chose E-ELAN as the final layer aggregation, an extend version of the ELAN computational block, which enables the framework to learn better.

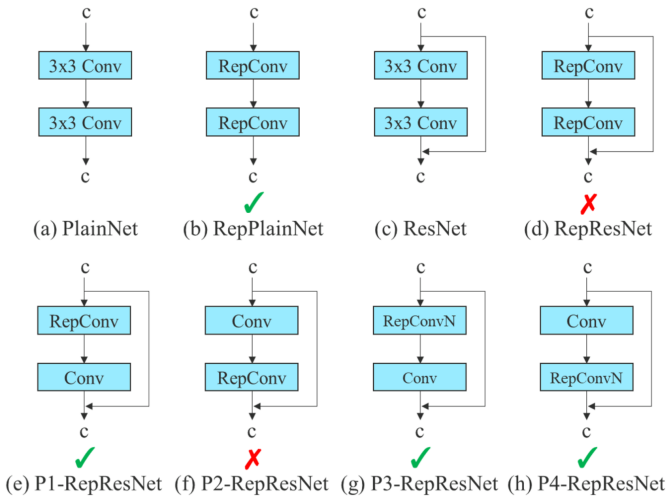


(d) E-ELAN

(b.) Model Scaling for Concatenation-based Models: the authors scale the network depth and width in concert while concatenating layers together. They showed that this technique keep the model architecture optimal while scaling for different sizes.



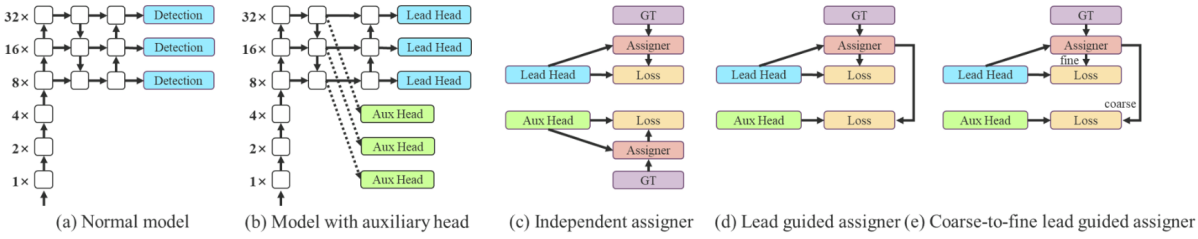
(c.) Planned re-parameterized convolution: Re-parameterization involves averaging a set of model weights to enhance the model's robustness to general patterns it aims to learn. The YOLOv7 authors specifically focus on module-level re-parameterization, where different parts of the network have their own re-parameterization strategies. The authors conducted experiments by switching or replacing the positions of RepConv, 3×3 Conv, and Identity connection.



(d.) Coarse for auxiliary and Fine for lead loss

To improve the performance of the YOLO network, it can be beneficial to incorporate an auxiliary head within the network at an intermediate stage, despite the final predictions being made by the downstream network head. During the training process, both the detection head responsible for final predictions and the auxiliary head are supervised. The weights of the auxiliary heads are updated using an assistant loss, facilitating deep supervision and improving the model's learning process.

In addition, the label assigner mechanism plays a crucial role in YOLOv7. It takes into account both the network's prediction results and the ground truth annotations. Based on this information, the Label Assigner assigns soft labels to the training samples instead of using hard labels. Soft labels are more flexible and provide a coarse representation of the desired outputs, allowing for a more robust and informative training signal for the model.



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.

The model that I surveyed is from the following paper - End-to-End Object Detection with Transformers (also known as DETR), which is a model that I've implemented in

another computer vision course.

DETR, also known as detection transformer, is an object detection system that utilizes a Transformer model built on top of a convolutional neural network (CNN) backbone. It employs a traditional CNN backbone to extract a 2D representation of the input image. This representation is flattened and combined with positional encoding before being fed into a transformer encoder.

The transformer decoder component of DETR takes a fixed number of learned positional embeddings, referred to as object queries, as input. It also attends to the output of the encoder. Each output embedding from the decoder is then passed through a shared feed forward network (FFN) that predicts either a detection (including class and bounding box information) or a class indicating the absence of an object ("no object").

Models	YOLOc7	DETR
Pros	1. Faster (since it’s a one-stage model) 2. The architecture design is more intuitive	Better performance in general
Cons	Performance is not as good as DETR in general	1. Need to manually choose the number of object queries, which means you have to guess the number of objects in a picture, and the meaning of object query is ambiguous 2. DETR converges very slowly