Class 6: Real-Time Object Detection

Problem Addressed: Object Detection

- งานยาก เรื่องของการระบุตำแหน่งและการจำแนกประเภทของวัตถุ (locate and classify objects)
- Goal ⇒ fast & high accuracy สำคัญ
- นอกจาก detect เราต้อง classify and localize เช่น แมว เป็ด หมา คอมฯ ต้องรู้ว่าภาพนี้ประกอบไปด้วยอะไรบ้าง

☐ **Importance**

- Visual modality is very powerful
- Responsive robot system ⇒ required real-time vision based object detection

☐ **YOLO concept**

- YOLOv1 ⇒ มีการแบ่งภาพ ออกเป็น grid และเมื่อ center ของวัตถุ อยู่ใน grid cell หมายความว่า เซลล์จะมีหน้าที่ detect object นั้น จากนั้น พัฒนาเป็น
- YOLOv2 ⇒ boundary box สามารถ detect object class > 9k class
- YOLOv3 ⇒ ให้ box detect sizes object ⇒ scale ต่างกันมากขึ้น
- YOLOv4, v5, v6, tiny, PP ⇒ speed & acc
- YOLOR, X ⇒ improved times generalize
- YOLOv7 ⇒ focus small optimization
- YOLOv8, YOLONAS ⇒ improve performance trained on coco dataset ⇒ 2023

YOLO Overview

→ split photo into S*S grid : each cell, detect object โดย center point ตกอยู่ใน grid ⇒ ช่วยในการระบุตำแหน่ง
→ each grid, predict bounding box ⇒ ในแต่ละ box จะมี object อยู่ในกล่องนั้น ก็คือจะทำนายตัวแปร

- x, y : coordinate of center object that are related with cell ค่าขนาดของ grid cell จะอยู่ในช่วง [0,1] เพราะผ่านการ normalized เช่น มุมซ้ายบน กับมุมขวาล่าง เท่ากับ [1,1] then center point of this cell is [0.5, 0.5]
- w, h : แสดงถึงความกว้างและความสูงของ bounding box
- Confidence: ทำหน้าที่ระบุว่า bounding box accurate แค่ไหน ⇒ ผ่านการทำ IOU ⇒ ideal = 0

→ YOLO is one of the fastest model for object detection

☐ **YOLO training**

- YOLO is a regression algorithm. ⇒ Predict bounding box from image
- Need to understand Parameters!!!
    - X : input ⇒ image ⇒ array or matrix of pixel (w*h) values (RGB values)
    - Y : output ⇒ tensor มีขนาด S*S*(B(box)*5(parameter)*C) แต่ละ grid ทำหน้าที่ predict class and distribution for a grid block

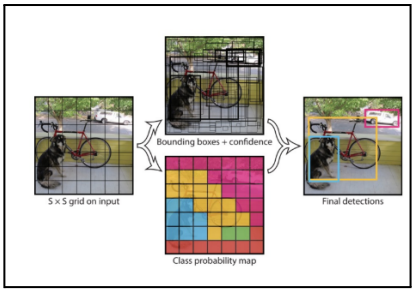
☐ **YOLO Architecture**

→ YOLO contain 7 convolution layers, ในการทำงานกับ image CNN เป็น เทคนิคที่เหมาะสม เพราะสามารถ capture ข้อมูลเชิงพื้นที่ได้ค่อนข้างดี, extract feature
→ ตัวอย่างเช่น

- input S(size) = 7 , B(bounding box) = 2, C(number of class) = 20
- Output is S*S*(5B+C) ⇒ 7*7*(5*2+20) = 7*7*30 ⇒ tensor size จาก YOLO

☐ **Non-maximal suppression**

→ case : multiple box, redundancy
→ filter bounding box and setting threshold : เอา low confidence score ออกไป → screen
→ identify class of object : ต้องคำนวณ class score และหา argmax ช่วยระบุคลาสของ object

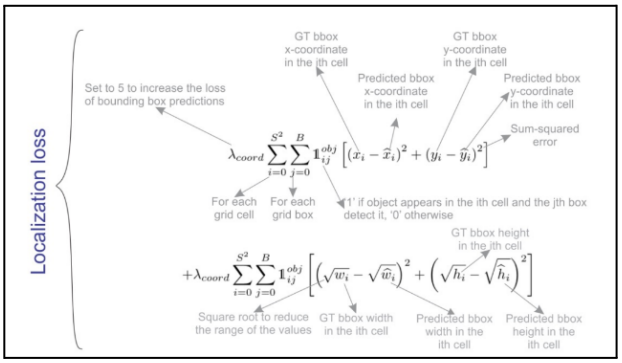


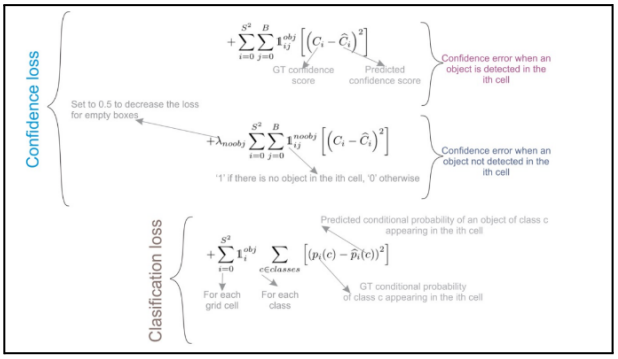
$$\Pr(\text{Class}_i|\text{Object}) * \Pr(\text{Object}) * \text{IOU}^{\text{truth}}_{\text{pred}} = \Pr(\text{Class}_i) * \text{IOU}^{\text{truth}}_{\text{pred}}$$

- [ ] **YOLO Prediction**

→it still possible to get redundant boxed but after we finalize with the high overlap (keep only highest confidence) ⇒ adds 2-3% on final MAP score

YOLO Objective Function





- [ ] **Localization Loss**

→ คำนึงถึง ขนาดของ bounding box

→ ensure bounding box from object predict มัน match กับ box ground truth (GT) หา error

→ คำนึงถึง position (x,y): mean square error loss

→ size (w,h): mean square root error loss

→ อยากให้ loss ต่ำ ค่าใกล้เคียง GT

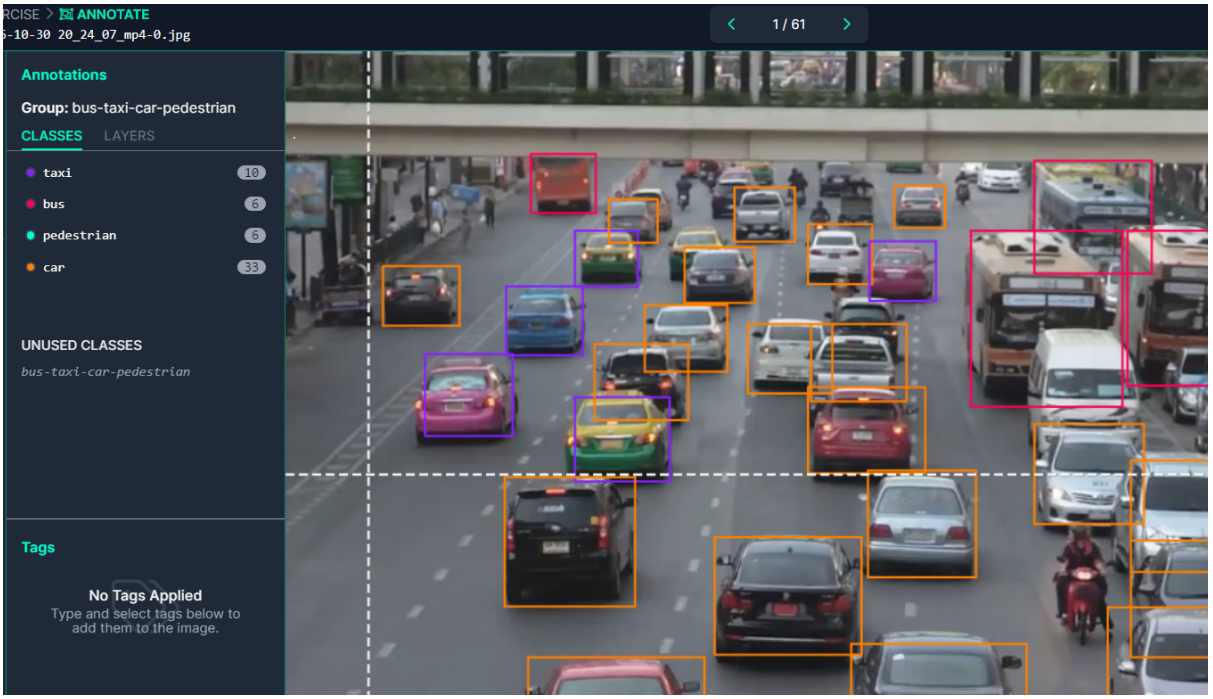
- [ ] **Classification Loss**

→ คำนึงถึง การ identify object class

- [ ] **Confidence Loss**

→ คำนึงถึง การ predict bounding box

- [ ] **Exercise**



- [ ] **YOLOv8 architecture**