YOLO

What is YOLO?

YOLO (You Only Look Once) is an Object Detection Algorithm

What is Object Detection?

The task of Object Detection is to find all the interesting objects in the image, determine their category and location

Application example

- Face detection
- Vehicle detection
- Obstacle detection in on-board camera images in autonomous driving
- etc.

Classification of Algorithms

"two-step" object detection

- Region recommendation is performed first, and then target classification.
- A preprocessing step for region proposals is included, making the overall process two-stage.
- R-CNN, SPP-Net, Fast R-CNN, etc.

"end-to-end" object detection

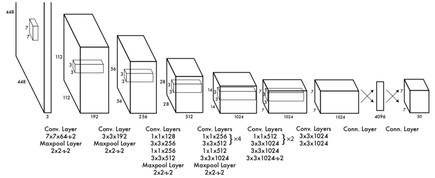
- Extract features directly in the network to predict object classification and location.
- This is a separately proposed method that does not separate detection proposals, making the whole process one-stage
- OverFeat, YOLOv1, YOLOv2, YOLOv3, etc.

Differences from other algorithms

How does YOLO work?

1. The system divides the input image into an S×S grid. If the center of the target falls within a grid cell, that grid cell is responsible for detecting the target
2. Each grid cell predicts B bounding boxes and confidence scores for those boxes
3. Perform NMS screening, screening probability and IoU

Structure of YOLO



Scale the image to 448X448

Running CNN on Graphs

Network has 24 Conv. layers, 2 Conn. laywes

Threshold the detection results based on the confidence of the model

Loss Function (sum-squared error)

bounding box loss

$$\lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^{B} \ell_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i{}^2] \quad \text{center loss}$$

$$\lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^{B} \ell_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + \sqrt{h_i} - \sqrt{\hat{h}_i})^2] \quad \text{bound loss}$$

$$\sum_{i=0}^{s^2} \sum_{j=0}^{B} \ell_{ij}^{obj} [(C_i - \hat{C}_i)^2] + \lambda_{noobj} \sum_{i=0}^{s^2} \sum_{j=0}^{B} \ell_{ij}^{noobj} [(C_i - \hat{C}_i)^2] \quad \text{confidence loss}$$

$$\sum_{i=0}^{s^2} \ell_i^{obj} \sum_{c \in classes} [(p_i(c) - \hat{p}_i(c))^2] \quad \text{classes loss}$$