

Day 3: Hands-on AI

Part 2: Basic Knowledge of Object Detection

2021

Part 3: Basic Knowledge of Object Detection

Content

- ◆ 2.1 Building Deep Learning Model
- ◆ 2.2 Getting Started with TensorFlow
- ◆ 2.3 Basic Knowledge of Object Detection

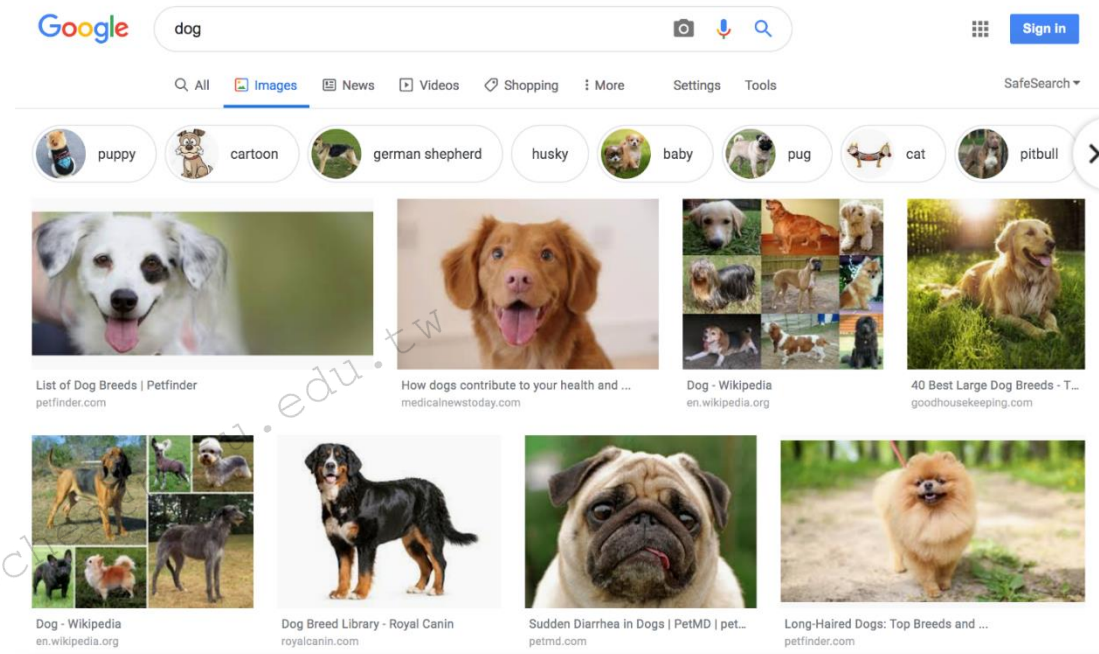
- 2.3.1 Overview
- 2.3.2 Performance metrics
- 2.3.3 Traditional methods
- 2.3.4 Two-stage detection
- 2.3.5 One-stage detection

2.3 Basic Knowledge of Object Detection

◆ 2.3.1 Overview

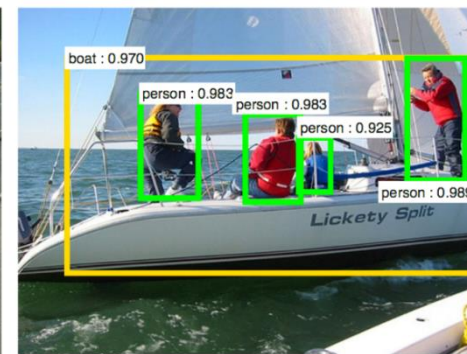
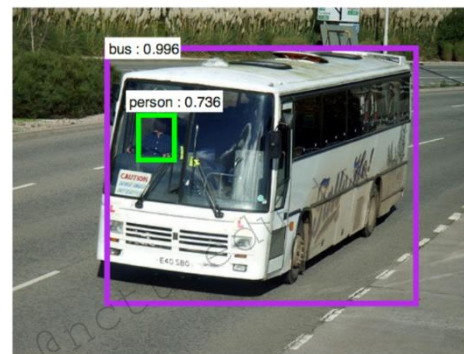
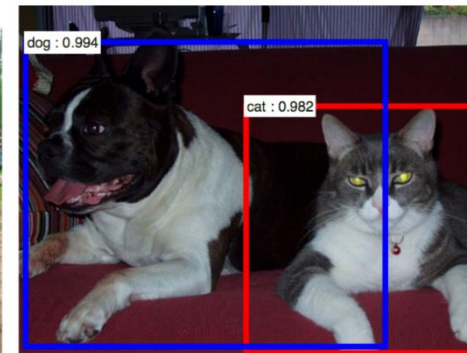
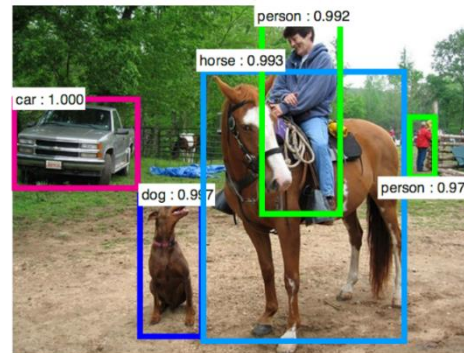
Overview

- In addition to classification and regression, detection is another important task in computer vision.
- Object detection aims to detect instances of semantic categories like animals, persons.
- Object detection is widely used in
 - image retrieval (e.g. Google image search)
 - face / pedestrian detection
 - self-driving car
 - ...



Task definition

1. Use a bounding box to mark the location of objects.
2. Predict the category of each object.
3. (Optional) The confidence score of this object



Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, arxiv 1506.01497

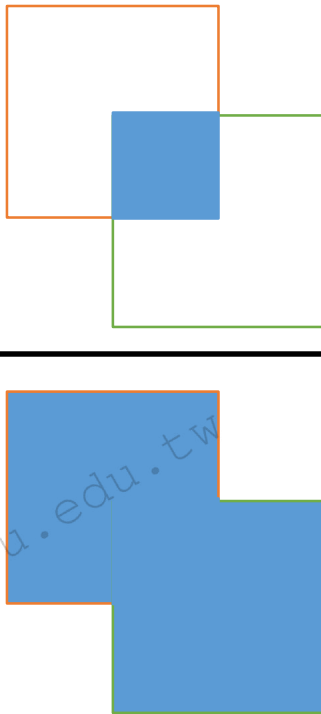
2.3 Basic Knowledge of Object Detection

◆ 2.3.2 Performance metrics

- IoU
- Confusion matrix, Precision, Recall, Accuracy
- Average Precision, mean Average Precision (mAP)

Intersection Over Union (IOU)

- Intersection Over Union (IOU, a.k.a. Jaccard index) is a statistic for measure similarity of two sets.

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$


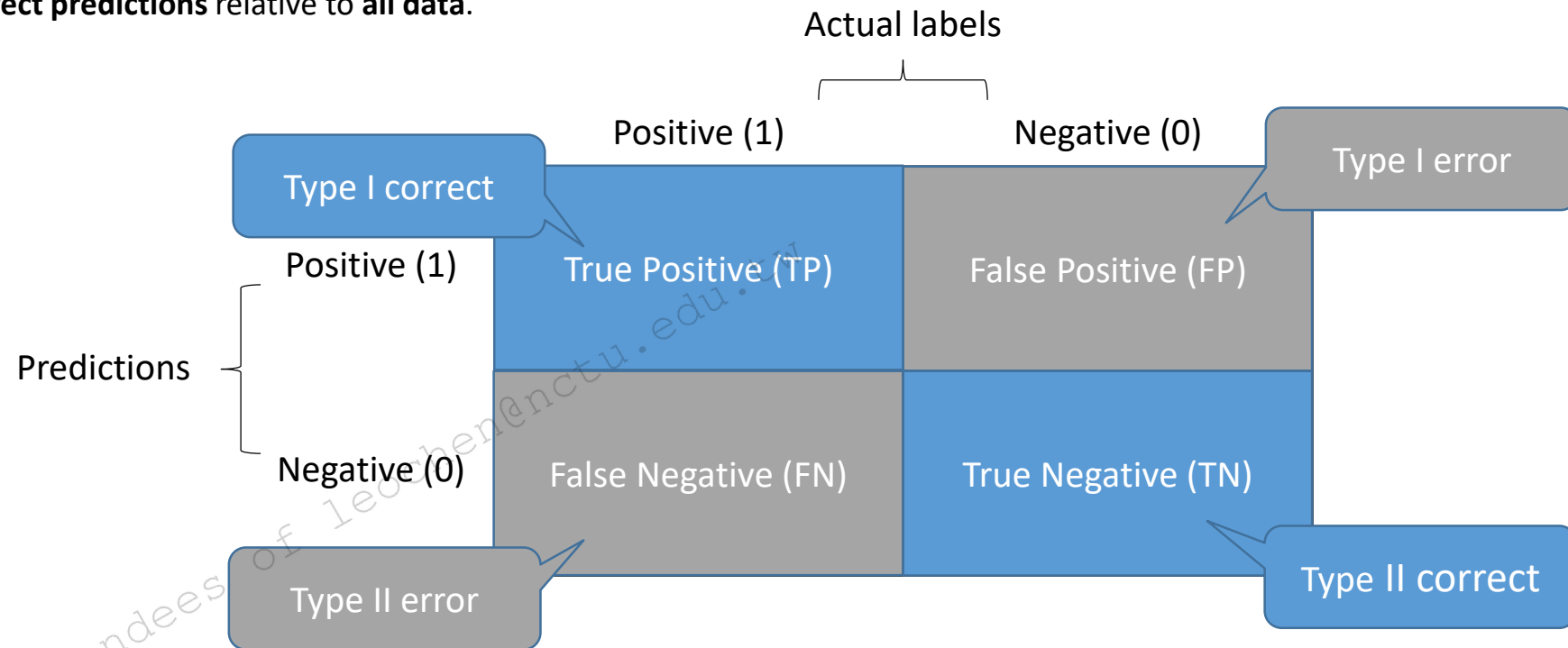
Confusion matrix

- Let's consider **binary classification**.
- 1. Precision is the fraction of **true positive** instances among all the **predicted positive** instances (including correct and wrong predictions).
- 2. Recall is the fraction of **true positive** over the total amount of **actual positive** instances (including found and not found).
- 3. Accuracy is the proportion of **correct predictions** relative to **all data**.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$



Confusion matrix

- Let's consider **binary classification**.
- 1. Precision is the fraction of **true positive** instances among all the **predicted positive** instances (including correct and wrong predictions).
- 2. Recall is the fraction of **true positive** over the total amount of **actual positive** instances (including found and not found).
- 3. Accuracy is the proportion of **correct predictions** relative to **all data**.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

		Actual labels		
		Is-Cat (1)	Not-Cat (0)	
Predictions	Is-Cat (1)	Type I correct TP: 60	Type I error FP: 20	80
	Not-Cat (0)	Type II error FN: 40	Type II correct TN: 80	120
		100	100	

Confusion matrix

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{60}{60 + 20} = 0.75$$

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{60}{60 + 40} = 0.6$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{80 + 60}{80 + 60 + 40 + 20} = 0.7$$

		Actual labels		
		Is-Cat (1)	Not-Cat (0)	
Predictions	Is-Cat (1)	Type I correct TP: 60	Type I error FP: 20	80
	Not-Cat (0)	Type II error FN: 40	Type II correct TN: 80	120
		100	100	

Confusion matrix

- For detection task, we regard **correct or wrong** of detection on each category as binary classification. But how to define “correct”?
- Multiple categories of objects usually exists in one image. Firstly, we only consider one object category at a time, all the other categories are seen as background; secondly, we define positive as the object category and negative as background.
 - - True Positive (TP): A correct detection. Predict a b-box as an object, and indeed it is. Detection with $\text{IOU} \geq \text{threshold}$.
 - - False Positive (FP): A wrong detection. Predict a b-box as an object, but it is not an object. It may be background actually ($\text{IOU} = 0$), or a part of the object ($0 < \text{IOU} < \text{threshold}$).
 - - False Negative (FN): A ground truth not detected. No bounding box at all on an object.
 - - True Negative (TN): Does not apply. If taken literally, TN means the model predict a bounding box as background, and it is indeed background. But, object detection algorithm usually don't predict background, and “boxes of background” is not labeled (infinite theoretically).

Precision -> AP-> mAP

- Remember? Whether a predicted bounding box is TP or FP is depending on a **threshold** value *thresh_binary_cls*. A threshold is a real number between 0 and 1.
- Precision: Each category has a precision value given a fixed threshold, calculated with formula:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- Average Precision (AP): Each category has a group of precision values if the threshold is stepping in a range with a small step size, e.g. [0.5, 0.55, 0.65, ..., 0.9, 0.95]. Average of these precision values is called Average Precision.

Precision-Recall curve

- Precision-Recall curve (PR-curve): If the threshold increases from 0 to 1, Precision and Recall are also changing. A good model should give high Precision and Recall if the threshold varies or not.
- (Suppose one category here) We sort all predicted bounding boxes by IOU with ground-truth labels. The larger IOU the box has, the farther it stands. If we choose the instances from the sorted list one by one, calculate the precision and recall, and plot them on a chart (whose horizontal axis for recall, vertical axis for accuracy) named Precision-Recall curve.
- If a PR-curve of model A is always on the upper right of the other model B, we say model A is better than B.

Precision \rightarrow AP \rightarrow mAP

mean Average Precision (mAP): mean of Average Precisions on different categories.

PASCAL VOC mAP

- PASCAL VOC is a classic and popular dataset for object detection. It sets threshold to 0.5, symbolically as mAP@0.5

COCO mAP

- COCO dataset is a recent, large dataset. For COCO dataset, AP is the average value over multiple threshold.
- For example, mAP@[.5:.95] to the average mAP for various IOU threshold from 0.5 to 0.95 with a step size of 0.05.

2.3 Basic Knowledge of Object Detection

◆ 2.3.3 Traditional methods for Object Detection

- Haar feature
- Viola–Jones object detection framework

Haar feature

- All human faces share some similar properties, if you look and think carefully. These regularities are abstracted and summarized as Haar features.

- Common characteristics of faces:

- The eye region is darker than the upper-cheeks.
- The nose bridge region is brighter than the eyes.

Similar to dot multiplication of two vectors, if Haar feature C is sliding on the image, the largest output is produced when C is overlapping the nose area.

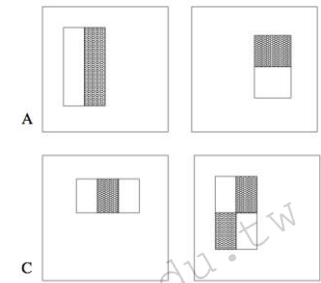


Figure 1: Example rectangle features shown relative to the enclosing detection window. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the grey rectangles. Two-rectangle features are shown in (A) and (B). Figure (C) shows a three-rectangle feature, and (D) a four-rectangle feature.

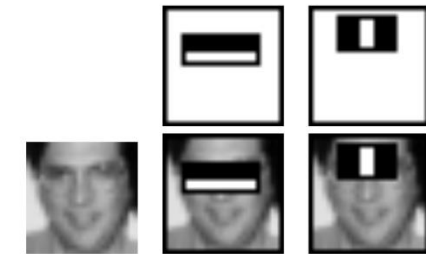
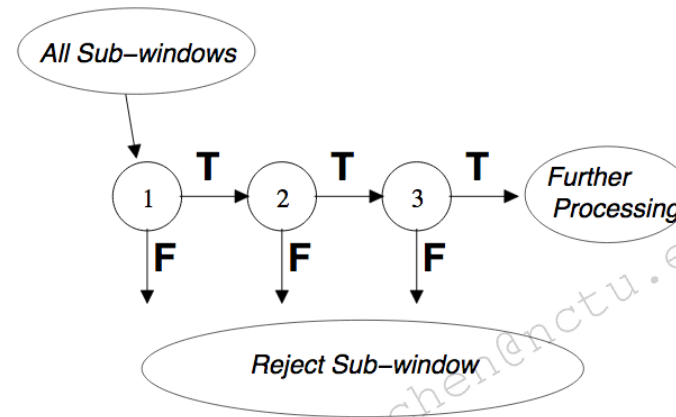


Figure 3: The first and second features selected by AdaBoost. The two features are shown in the top row and then overlaid on a typical training face in the bottom row. The first feature measures the difference in intensity between the region of the eyes and a region across the upper cheeks. The feature capitalizes on the observation that the eye region is often darker than the cheeks. The second feature compares the intensities in the eye regions to the intensity across the bridge of the nose.

Viola–Jones object detection framework

- There are a total of $M = 162,336$ possible features generated in a 24×24 pixel image. So AdaBoost algorithm is applied to both select the best features and to train classifiers that use them.
- Other tricks are also used to reducing computation time, like integral image to simplify the features extraction, cascade architecture to accelerate inference progress.
- Viola–Jones is quite complicated, so we will not introduce it more deeply.

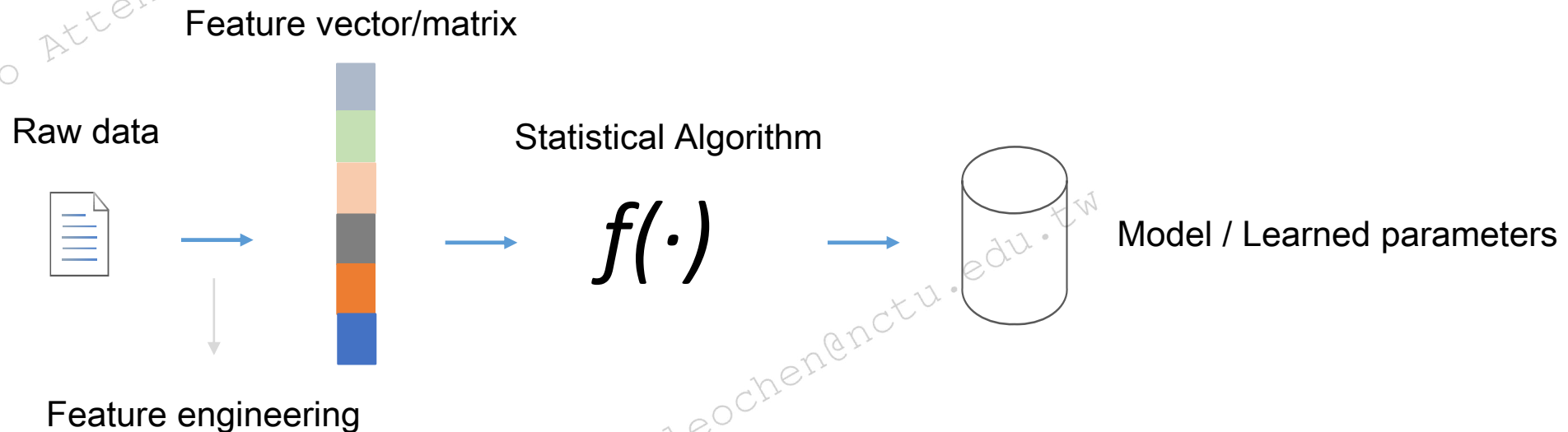


Cascade architecture

Traditional methods for Object Detection

Summary:

- Haar feature is well-designed by experts, and highly dependent on specific scenarios (face detection here). It can't apply general object detection. For example, Haar feature is not applicable to pedestrian detection. Traditional methods for Object Detection follows the **paradigm of machine learning**.



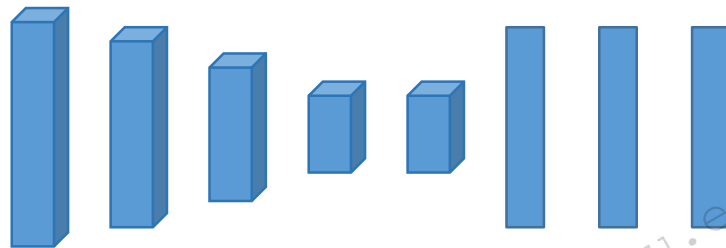
Traditional methods for Object Detection

- We will see how deep learning methods deal with object detection problem.
- Deep learning methods are proposed for general object detection. Moreover, it's easy to resolve single or specific object detection. Deep learning model trains the model **end-to-end**, rather than depending on experts-designed features.

Raw data



Network



Loss function

$loss(\cdot)$



Optimizer



2.3 Basic Knowledge of Object Detection

◆ 2.3.4 Two-stage detection

- RCNN
- Fast-RCNN
- Faster-RCNN

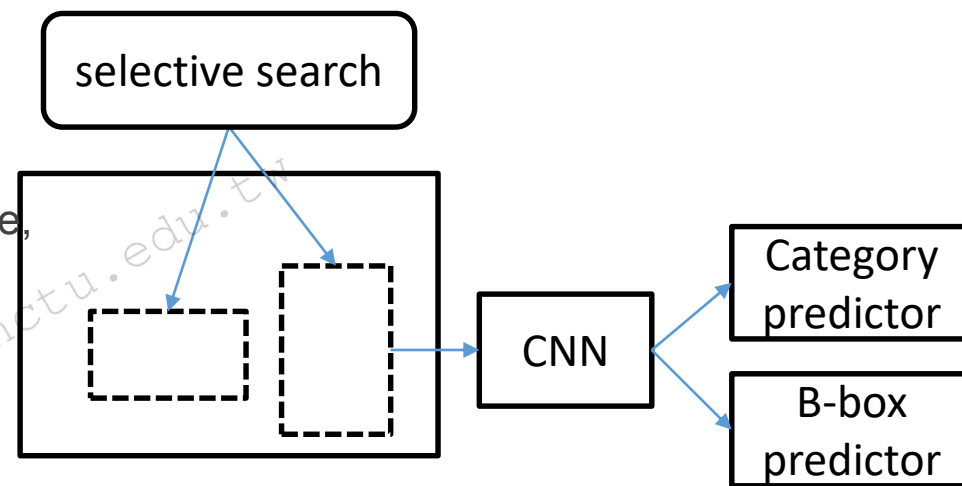
Region-based CNNs (R-CNNs)

Steps:

- 1. Select several proposed regions (a.k.a Region of Interest, ROIs) by a method named selective search.
- 2. Fed with region images, use a pre-trained CNN as feature extractor to output image features.
- 3. Use SVMs to classify category, and predict bounding box with a linear regression model.

RCNN is the masterpiece of two stage detection:

- 1st stage: Region proposal by selective search
- 2nd stage: For the sub-images generated by regions above, classify categories and predict bounding box.



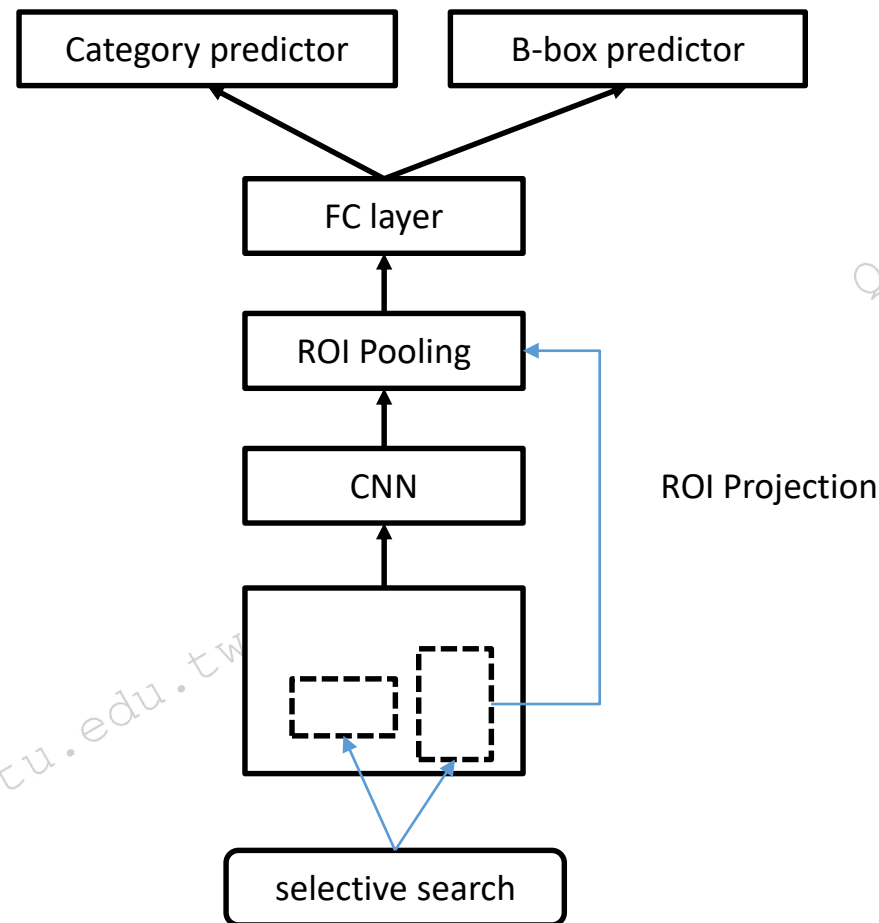
Fast RCNN

- **What's the problem of RCNN?**
- 1. Expensive in time and space: A large number of regions are selected, but most of them overlap each other. If N regions is proposed, CNN is called for N times. Save them to disk for SVMs and regression models also takes up much space.
- 2. Regions has different ratios, and traditional CNN only accepts input with fixed size. But resizing may lose some information.
- 3. The pre-training of CNN is separable from RCNN. While SVMs and regression models can be replaced by CNN in fact.

Fast RCNN

- What does Fast RCNN do?

1. The image, not the ROIs, is directly input into CNN, and the CNN also participates in the training. The output feature map
2. Generate n ROIs on the output feature map with selective search, and use ROI pooling to process ROIs with different sizes and ratios.
3. Use Softmax regression instead of SVM and regression



ROI Pooling

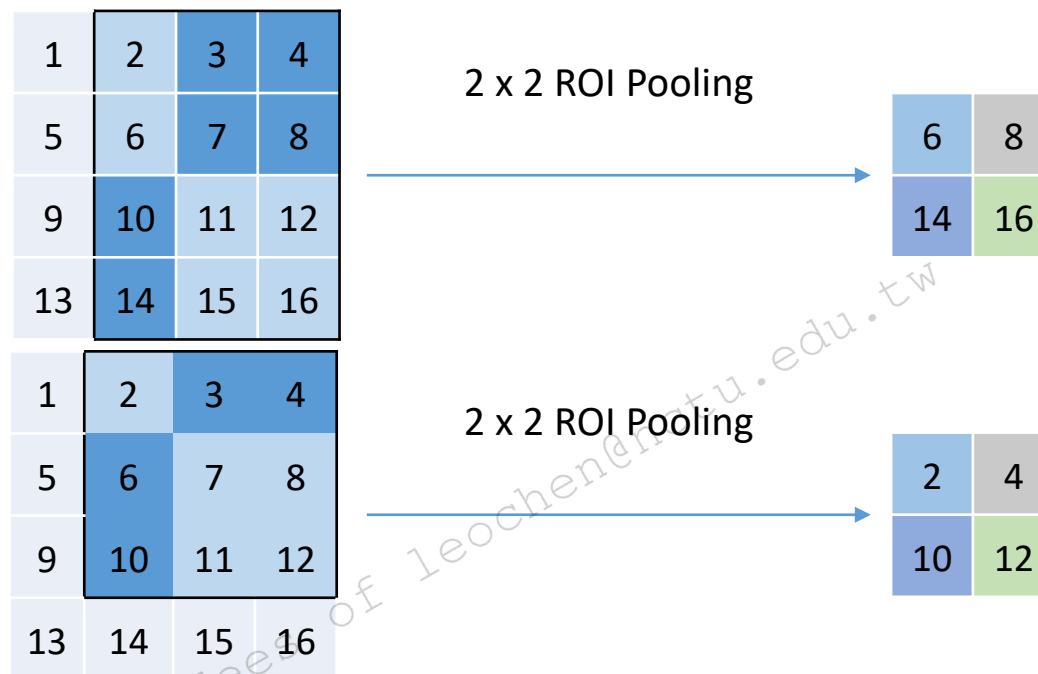
- **Review: Original Pooling**

- Input: [H, W, C]
- Hyper-parameters: pooling window size F, stride S.
- Output: [H', W', C]
 - $H' = (H-F)/S + 1$
 - $W' = (W-F)/S + 1$



ROI Pooling

- In order to resolve the problem that the input size of CNN is fixed while the shape of ROIs varies, Fast-RCNN design a new layer named ROI Pooling layer. ROI pooling layer can extract features of the same shape from ROI of different shapes.
- Given a desirable output size of ROI Pooling (Supposed $H \times W$), the input ROI is divided into $H \times W$ sub-windows, and each sub-window outputs one value, so the output is $H \times W$.



ROI Pooling

- Translation variance and ROI Projection
- For RCNN, ROIs are generated from raw image directly; CNN are called for many times to extract features of these ROIs.
- For Fast RCNN, ROIs are used in feature map, but ROIs in feature map are projections of original ROIs from raw image to feature map. (This method is inspired by SSP-Net)

```
int roi_start_w = round(bottom_rois[1] * spatial_scale_);  
int roi_start_h = round(bottom_rois[2] * spatial_scale_);  
int roi_end_w = round(bottom_rois[3] * spatial_scale_);  
int roi_end_h = round(bottom_rois[4] * spatial_scale_);
```

Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition, arxiv 1406.4729
https://github.com/rbgirshick/caffe-fast-rcnn/blob/fast-rcnn/src/caffe/layers/roi_pooling_layer.cpp
<http://www.robots.ox.ac.uk/~tvb/publications/talks/fast-rcnn-slides.pdf>

Faster RCNN

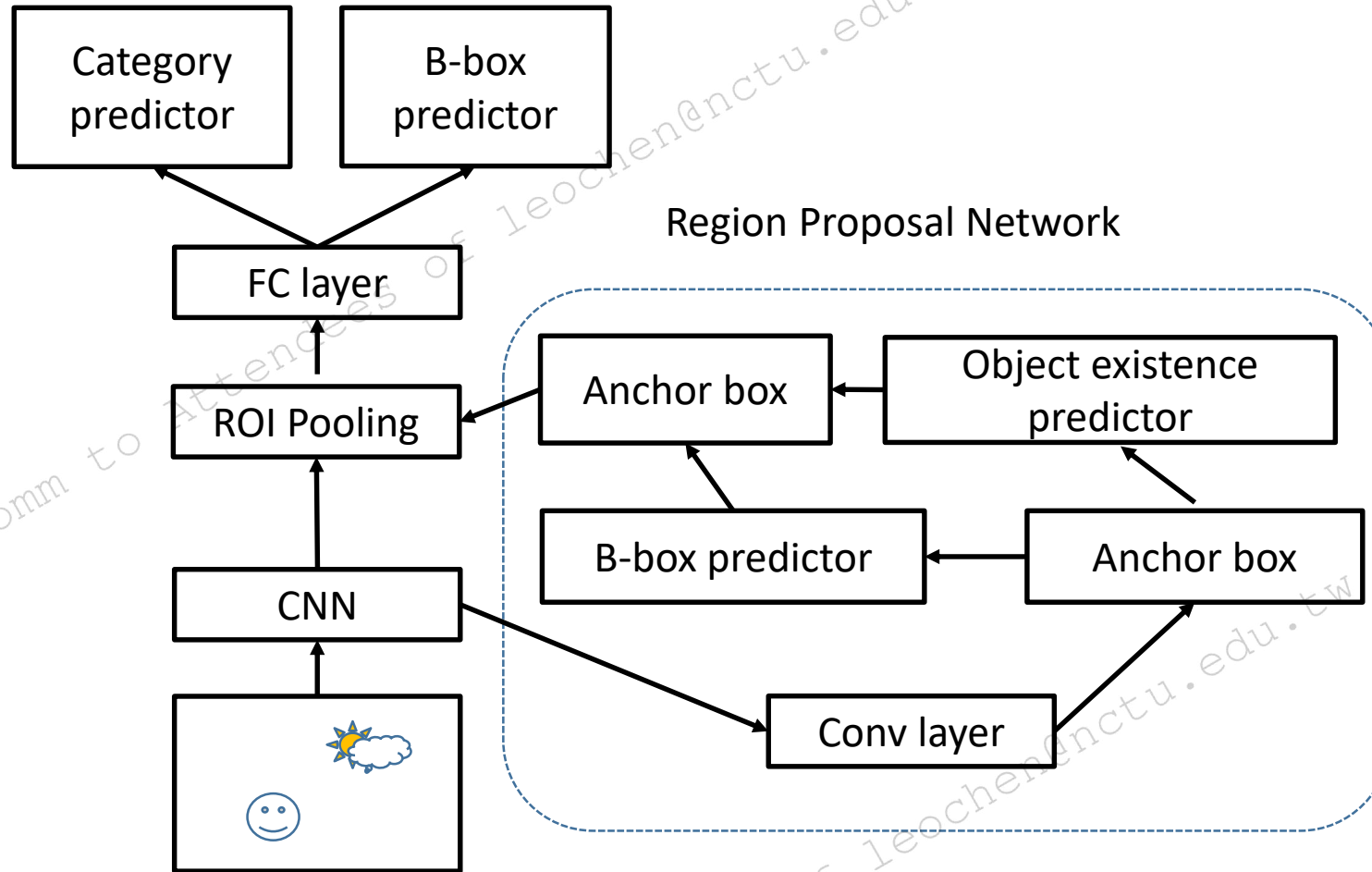
- **What's the problem of Fast RCNN?**

- Selective search is still used;
- 1) It is separable with the main part of the model.
- 2) It generates a large number of regions; most of those regions are background.

- **What does Faster RCNN do?**

- Replace selective search with Region Proposal Network (RPN)

Faster RCNN



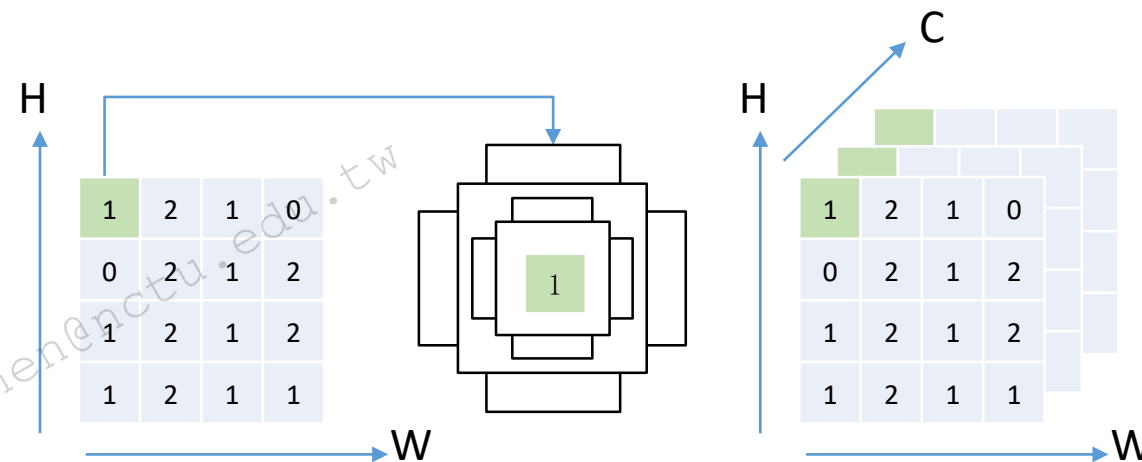
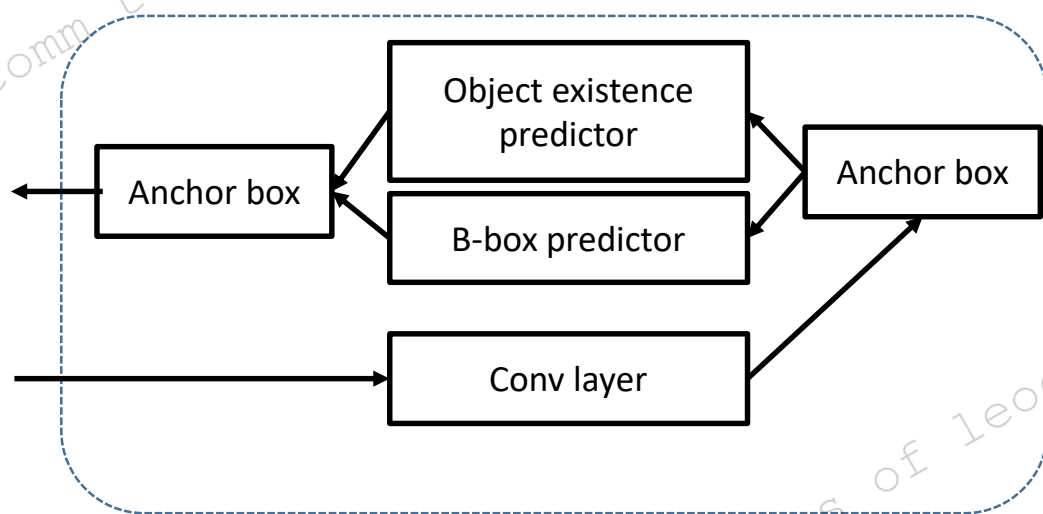
Faster RCNN replaces selective search with Region Proposal Network (RPN) to generate candidate ROIs. RPN is a sub-network of the whole network and it is jointly trained with other parts.

Faster RCNN

Region Proposal Network (RPN) is the alternative of Selective search implemented by CNN.

1. The output of CNN is fed into a conv layer and output shape is $[H, W, C]$
2. Each pixel in 2D projection of feature map, multiple anchors (candidates of ROI) are generated.
3. Use the values along the c-axis direction as feature, predict object existence and bounding box.
4. Use non-maximum suppression(NMS) to remove similar bounding box results that correspond to category predictions of "object".

Region Proposal Network



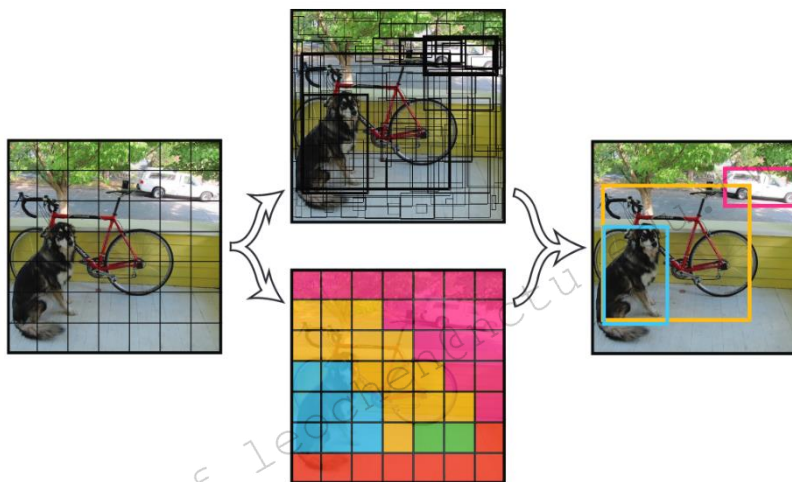
2.3 Basic Knowledge of Object Detection

◆ 2.3.5 One-stage detection

- YOLO
- SSD
- Non-Maximum Suppression (NMS)
- Hands-on Object Detection

YOLO (You only look once)

- YOLO defines detection as regression.
- 1. Divides the input image into a $S \times S$ grid.
- 2. Each grid cell predicts B bounding boxes (defined by x, y, w, h and confidence) and confidence scores for those boxes.
- 3. Merge duplicate bounding boxes and discard ones with low confidence by non-maximum suppression.



You Only Look Once: Unified, Real-Time Object Detection, arxiv 1506.02640

SSD (Single Shot Detector)

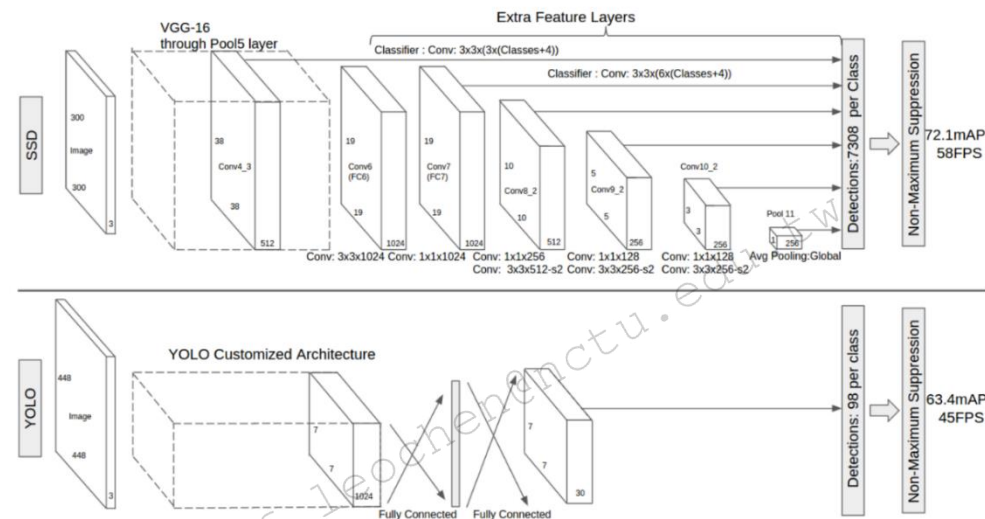
◆ SSD detects objects in these steps:

- 1. Use FC layer reduced classic classification models as feature extractor (a.k.a. base net).
- 2. Use multiple convolution and pooling layers to get feature map in different scale, in order to capture object with various sizes.
- 3. For each feature map of multi-scale layers, generate fixed number of anchors (e.g. 10) in every pixel.
- 4. For each feature map of multi-scale layers, classify all the categories and predict all the bounding boxes as well confidence scores of all generated anchors in this layer.
- 5. Merge duplicate bounding boxes and discard ones with low confidence by non-maximum suppression.

SSD

◆ Comparing with YOLO, SSD has the following differences:

- SSD introduces anchor (originally proposed by Faster-RCNN), which constrains bounding boxes with different ratios and scales..
- Anchors are generated in multi-scale feature map and projected to original image.
- SSD removes fc layers to reduce computation.



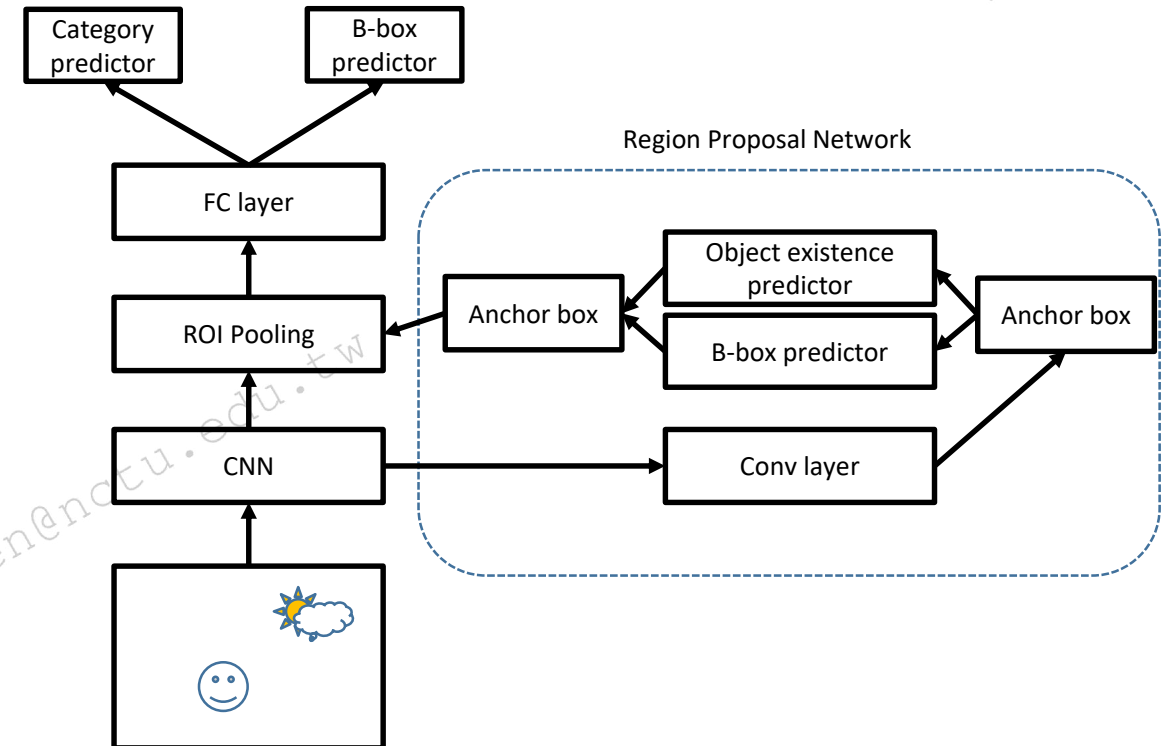
SSD: Single Shot MultiBox Detector, arxiv 1512.02325

SSD

- Comparing with Faster-RCNN, SSD has the following differences:

1. Faster-RCNN is a two-stage method, RPN is performed to generate anchor boxes (or ROI) during each inference. It includes category and bounding box predictor in core network as well as binary category and bounding box predictor in PRN.

2. SSD is inspired by Faster-RCNN, but still one-stage network. Anchor boxes are defined before an image is fed into.



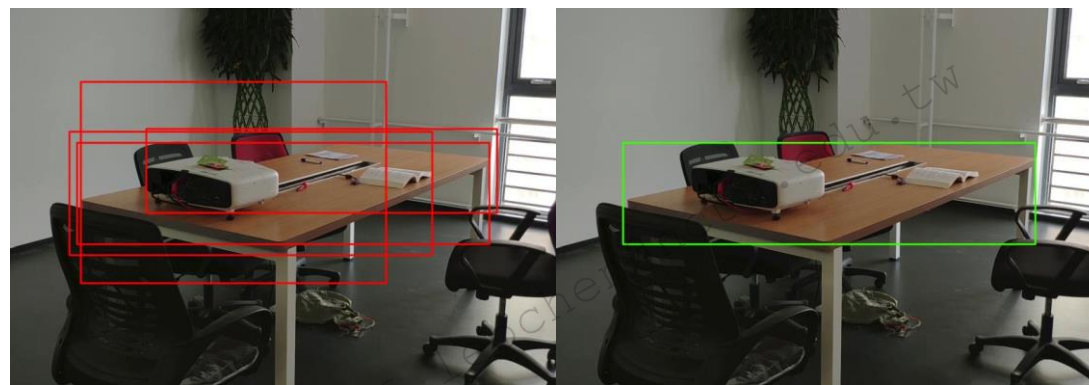
SSD: Single Shot MultiBox Detector, arxiv 1512.02325

Non-Maximum Suppression (NMS)

- ◆ Last but not least, we have to introduce post-processing of object detection.
- ◆ Attributed to anchors, there are a large number of bounding boxes generated. For example, 8732 anchor boxes are set in advance in SSD-VGG-300:
 - There are 6 feature maps in multi-scale layers, and their resolution is [38, 19, 10, 5, 3, 1].
 - Centered on each pixel in feature maps, anchors with different ratios and scales are generated. The number of anchors in each layer is [4,6,6,6,4,4].
 - So the total number of anchors is $38 \times 38 \times 4 + 19 \times 19 \times 6 + 10 \times 10 \times 6 + 5 \times 5 \times 6 + 3 \times 3 \times 4 + 1 \times 1 \times 4 = 8732$.
- ◆ 8732 bounding boxes are far more than the number of objects in an image. So first, we select bounding boxes with confidence threshold thresh_conf.

Non-Maximum Suppression (NMS)

- Even after selection by confidence score **threshold** $thresh_conf$, post-processing is still unfinished.
- A single object may have several predicted bounding boxes. We need to merge them all.
- Look, these bounding boxes of the same object share a common feature: they overlap each other with a large area. How to deal with it?



Results before and after NMS

Non-Maximum Suppression (NMS)

1. Set hyper-parameter: threshold of IOU = *thresh_nms*.
2. Define output list O as empty. Set the candidate list C with all bounding boxes. Sort C by confidence scores.
3. Loop until C is empty:
 - 3.1 Select bounding box b with the highest confidence score from C ;
 - 3.2 add b to O , and remove b from C ;
 - 3.3 Calculate the IOU between b and all bounding boxes to in C ;
 - 3.4 Delete the bounding box whose IoU with b is greater than *thresh_nms*;
4. Finish.

After NMS, boxes in output list has the following characteristics:

1. The confidence scores of all boxes are greater than *thresh_conf*, which means that we can safely assume that they are objects, not backgrounds.
2. The IOU between any two of them are larger than *thresh_nms*, they can be considered as independent objects.
3. For a certain object, we select the box with the largest confidence score.

Hands-on Object Detection

- The principles of algorithms above are easy to understand, but the code implementation is beyond the scope of our course. However, we can still run an object detection demo by hand.
- TensorFlow Object Detection API is the official implement by Google TensorFlow, which makes it easy to construct, train and deploy object detection models. Combined with OpenCV, we can develop an object detection program in minutes.

Please refer to Jupyter notebook for detail.

Thank You