深度學習
Pytorch手把手實作
物件偵測

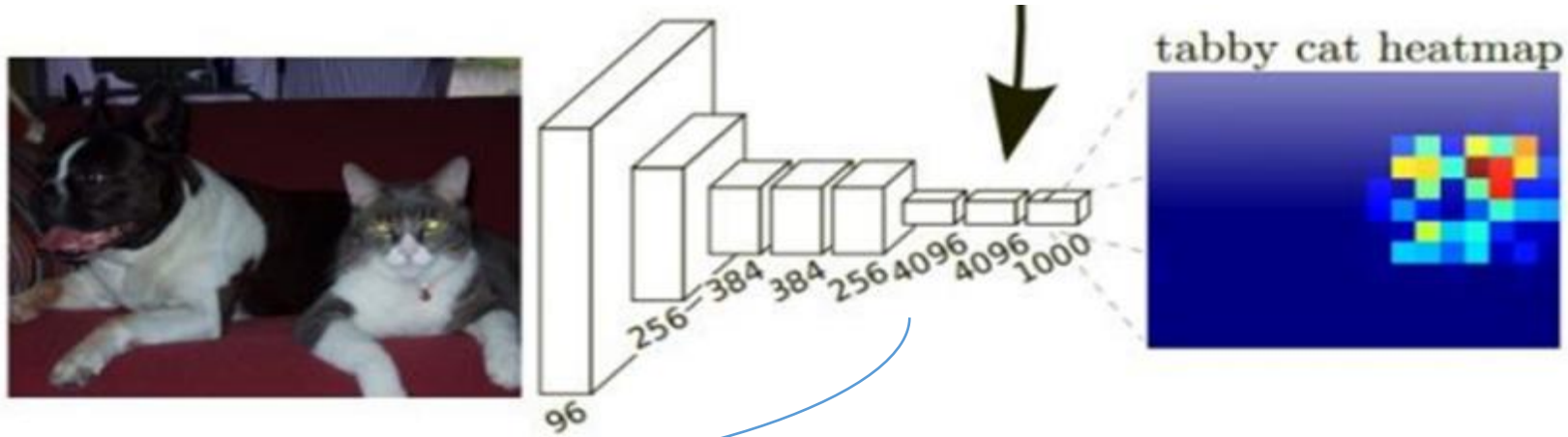
黃志勝
義隆電子人工智慧研發部
國立陽明交通大學 AI學院 合聘助理教授
國立台北科技大學 電資學院 合聘助理教授

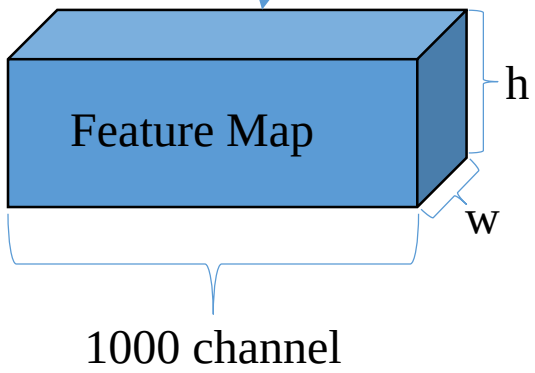
Introduction

- Brief Introduction
- 直覺的方式建立自己的物件偵測模型
- YOLO

Deep Learning: Feature Extractor

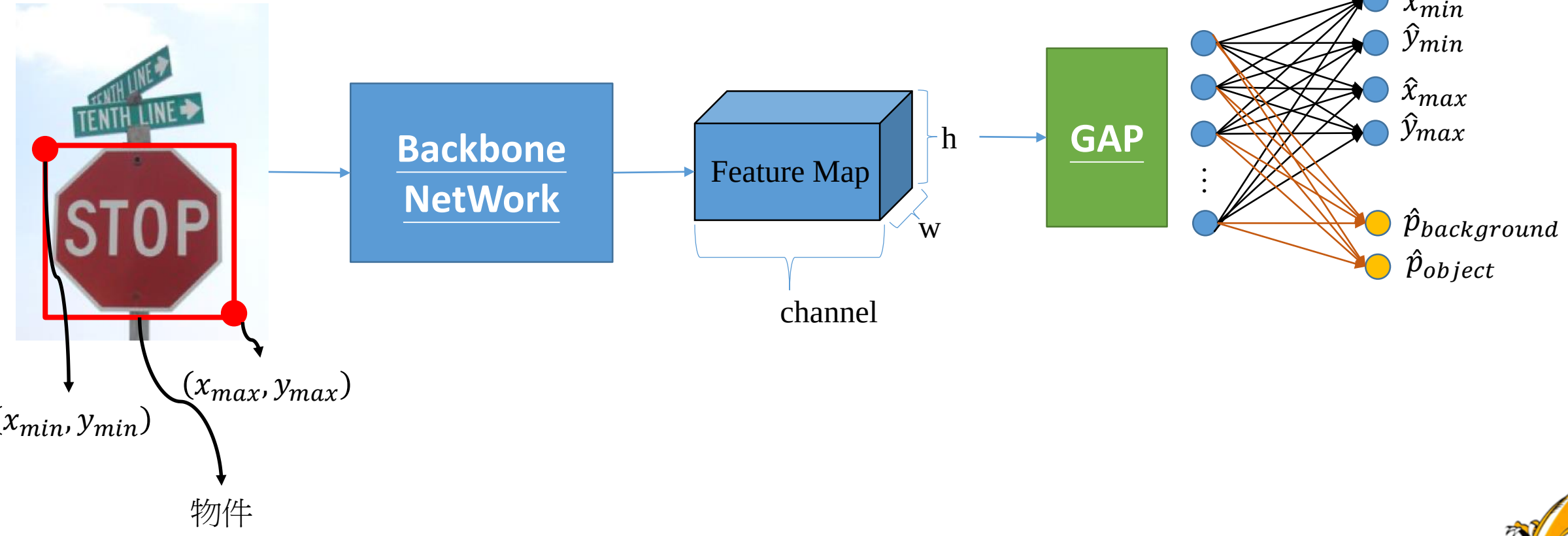
- Feature map可以做什麼?



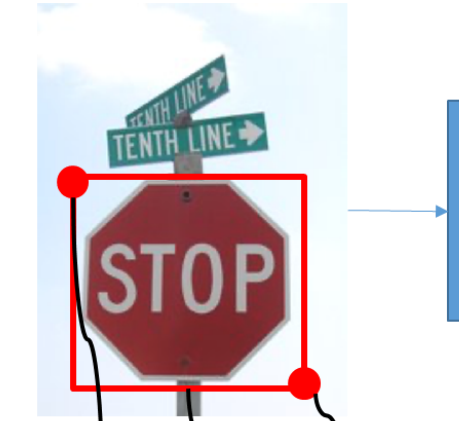


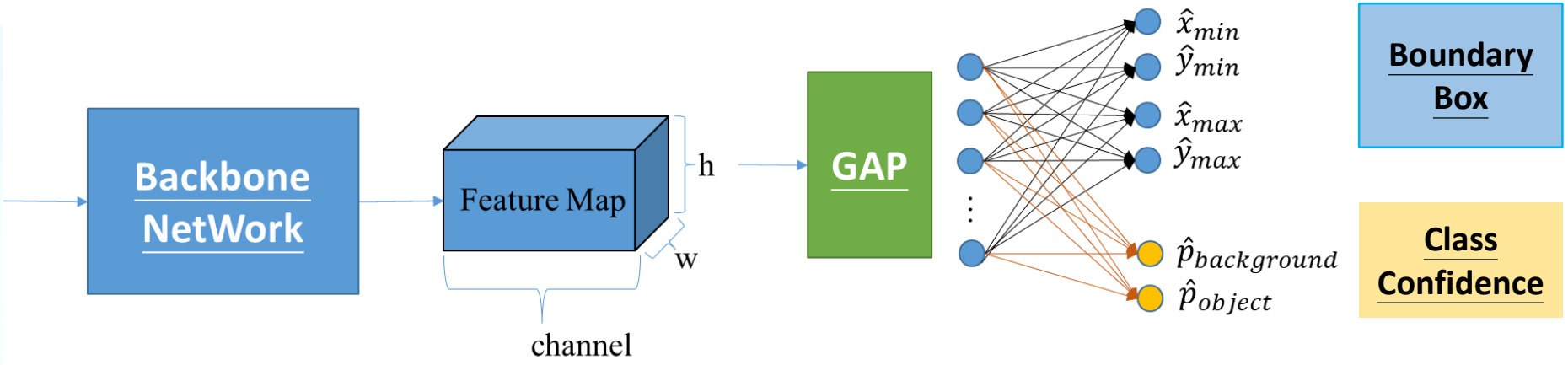
- 這張圖是貓還是夠 (Classification)
- 框出圖片內有貓和狗的位置 (Object detection)
- 從框出的物件(貓和狗)中的描繪出實際物件的輪廓 (Instance Segmentation)
- 把圖片描繪出物件(貓和狗)的輪廓 (Semantic Segmentation)

物件偵測


TENTH LINE
STOP
(x_{min}, y_{min})
(x_{max}, y_{max})
物件
Backbone NetWork
Feature Map
h
w
channel
GAP
$\hat{x}_{min}$
$\hat{y}_{min}$
$\hat{x}_{max}$
$\hat{y}_{max}$
$\hat{p}_{background}$
$\hat{p}_{object}$

物件偵測





目標:

1. $(\hat{x}_{min}, \hat{y}_{min}) \rightarrow (x_{min}, y_{min})$
2. $(\hat{x}_{max}, \hat{y}_{max}) \rightarrow (x_{max}, y_{max})$

→ Regression: MSE

3. $\hat{p}_{background} \rightarrow p_{background}$
4. $\hat{p}_{object} \rightarrow p_{object}$

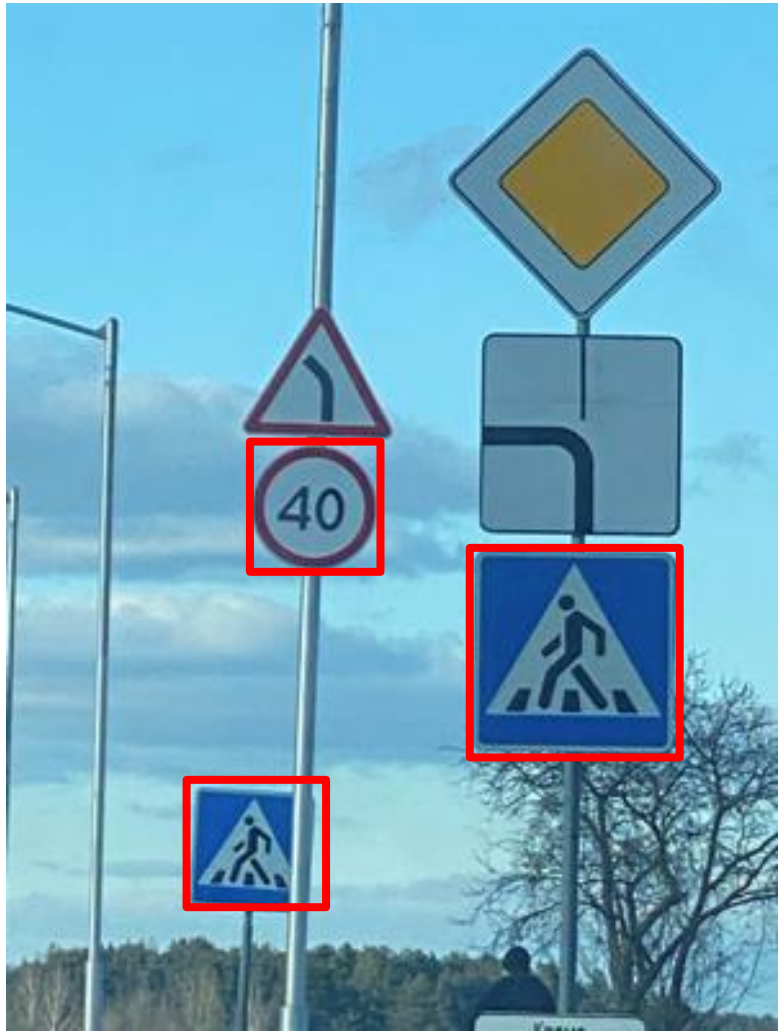
→ Classification: Cross Entropy

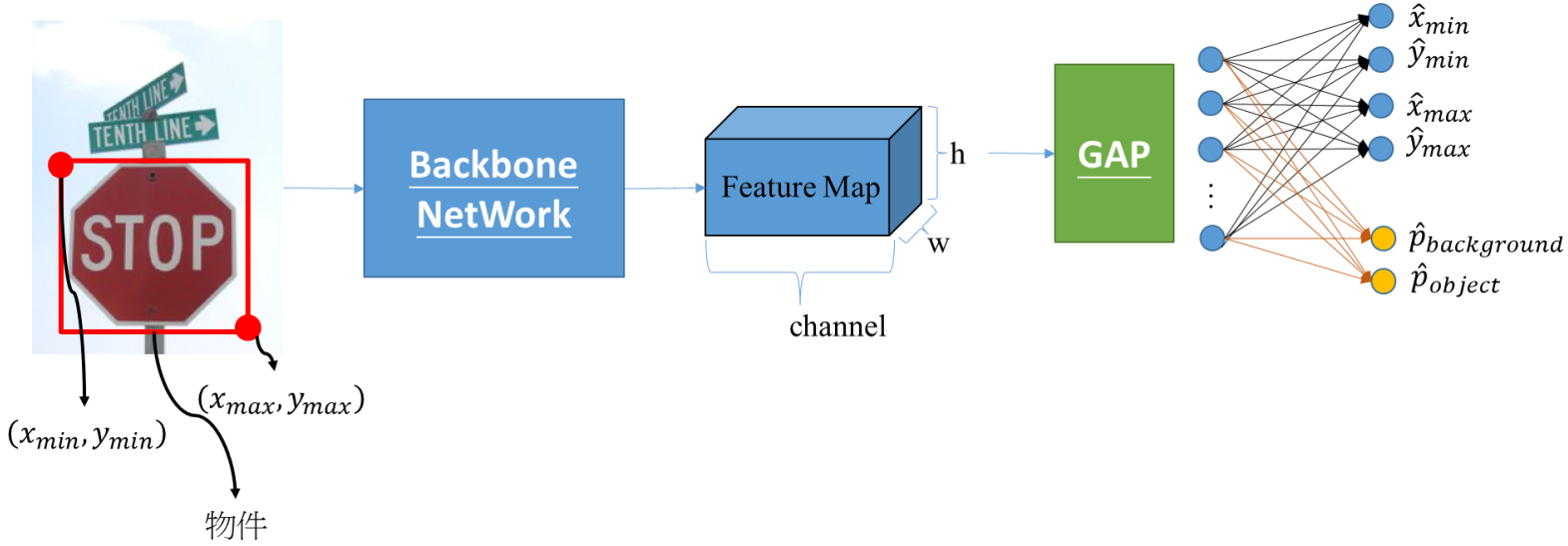
物件: One-hot Encoder

	Background	Object
x	$p_{background} = 0$	$p_{object} = 1$

最小化 {MSE (Regression)+Cross Entropy (Classification)}

上述的架構會有問題





- 物件數超過1 ????

預測框增加到B個

Backbone NetWork

GAP

$\hat{x}_{min}$ $\hat{y}_{min}$ $\hat{x}_{max}$ $\hat{y}_{max}$ Predicted box_1

$\hat{x}_{min}$ $\hat{y}_{min}$ $\hat{x}_{max}$ $\hat{y}_{max}$ Predicted box_B

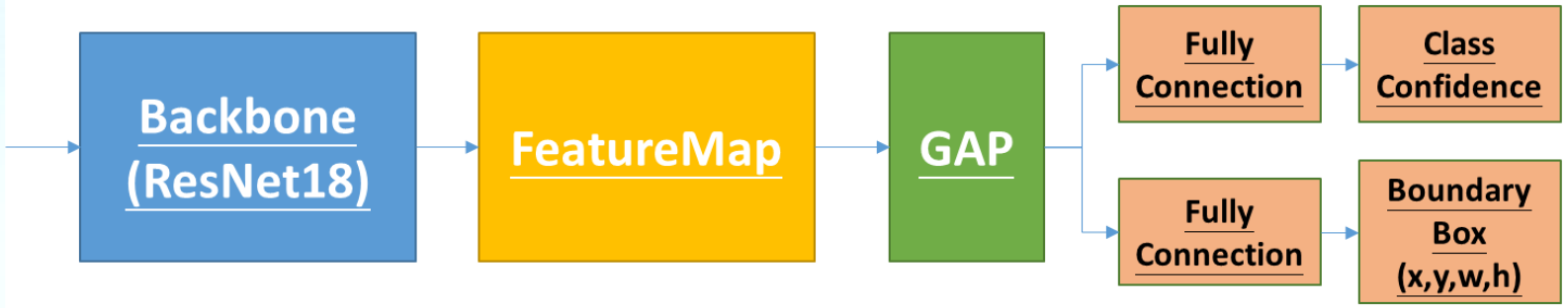
$box_1 = (x_{min}, y_{min}, x_{max}, y_{max})_1$

$box_2 = (x_{min}, y_{min}, x_{max}, y_{max})_2$

$\hat{p}_{background}$ $\hat{p}_{object}$ Predicted box_1

$\hat{p}_{background}$ $\hat{p}_{object}$ Predicted box_B

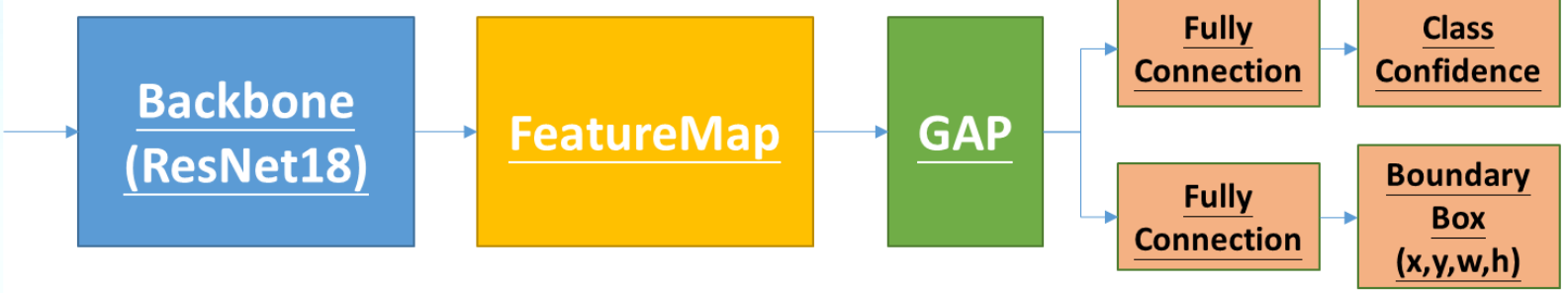
物件偵測

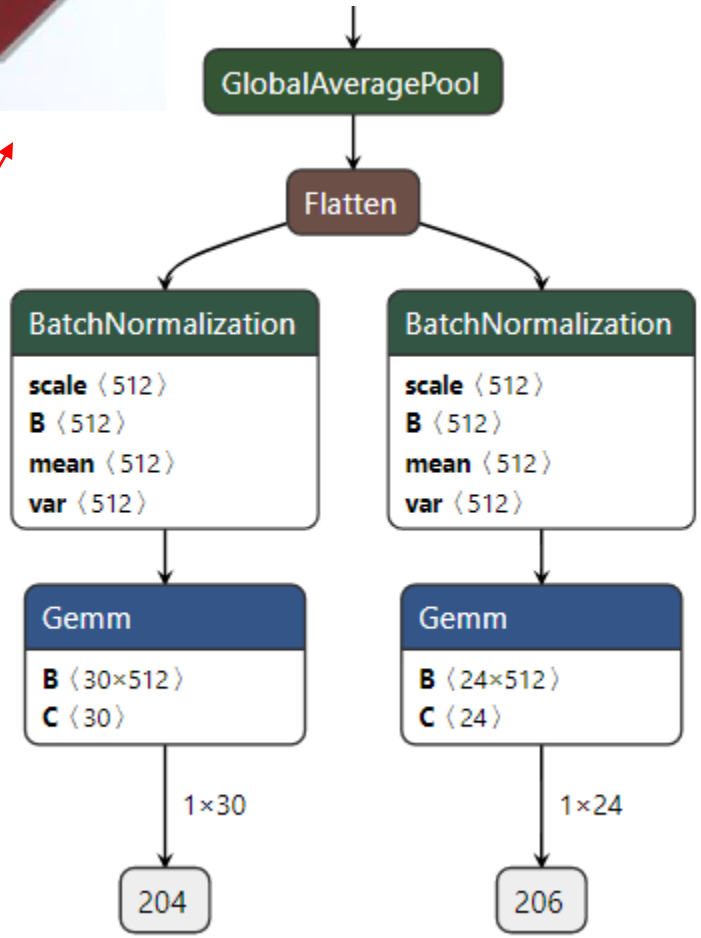


最後輸出預測6個物件框

Boundary Box輸出的特徵圖大小為
batch * (6*4)= batch * 24

Class Confidence輸出的特徵圖大小為
batch * (6*(n_class+1)) = batch * (6*5) = batch * 30





最後輸出預測6個物件框

Boundary Box輸出的特徵圖大小為
batch * (6*4)= batch * 24

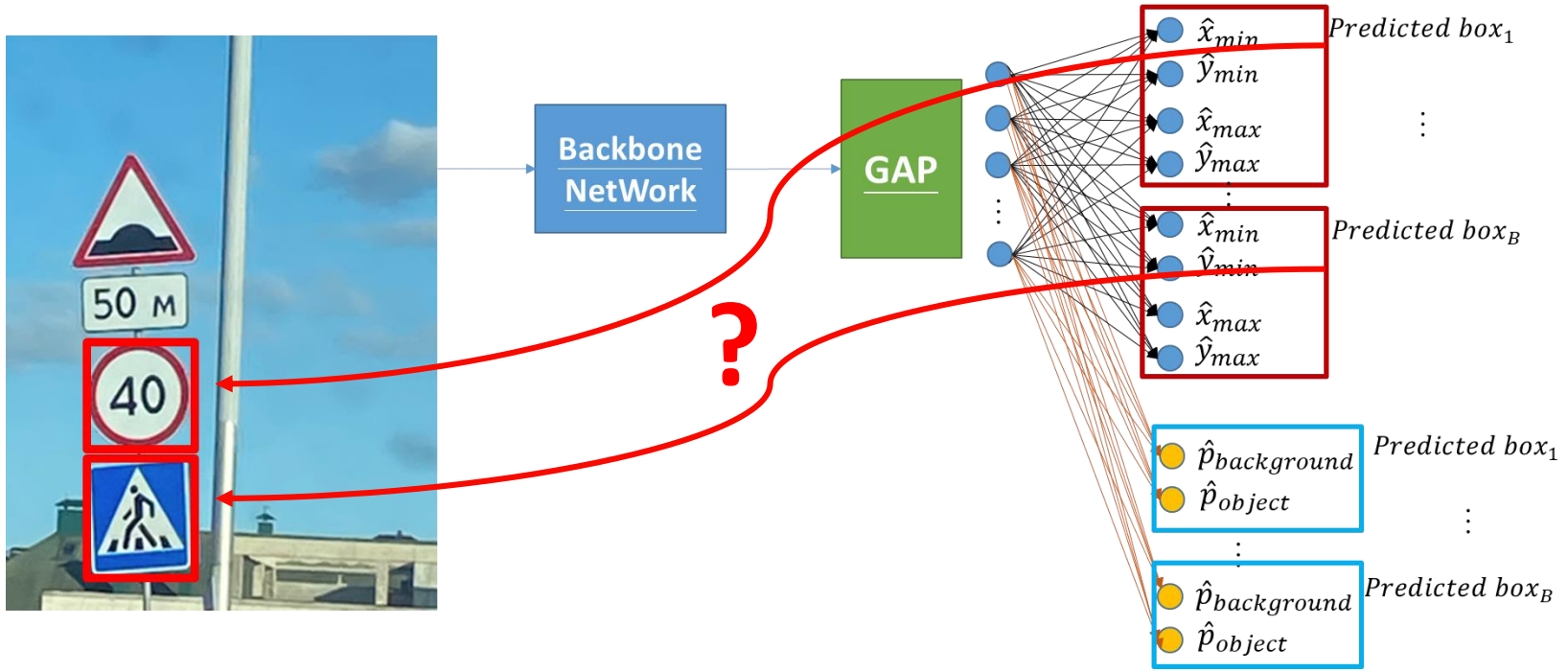
Class Confidence輸出的特徵圖大小為
batch * (6*(n_class+1)) = batch * (6*5) = batch * 30

Objection Detection

Pytorch 操作

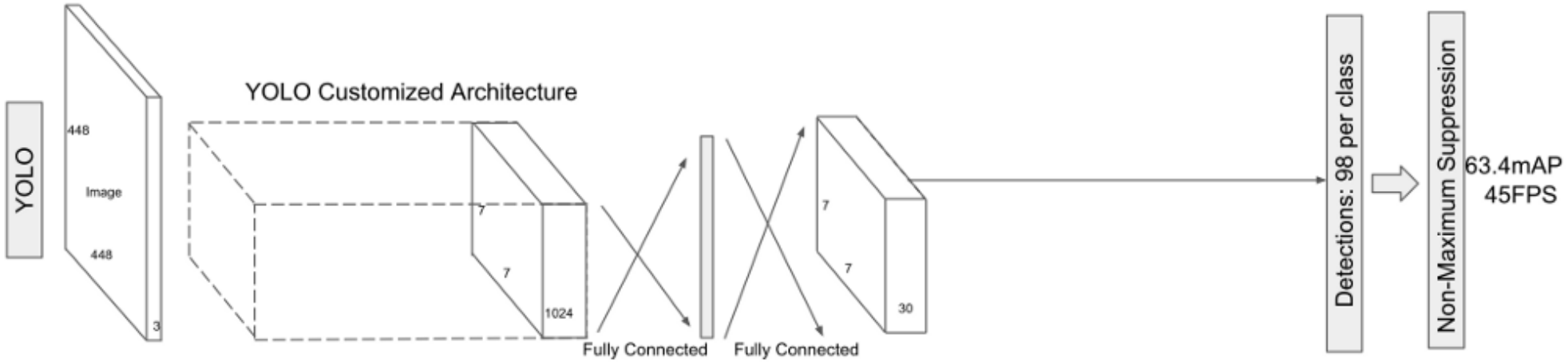
Jupyter Notebook

問題



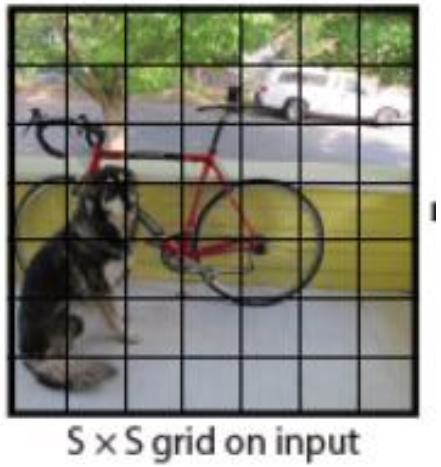
學習的時候，$Predicted\ box_i, i = 1,2,\ldots,B$該預測哪一個物件框。
造成混亂學習，所以會學不起來。

YOLOv1的detector怎麼處理



最後一層是7*7*30

7*7就是的grid cell



30 = 2×5+20

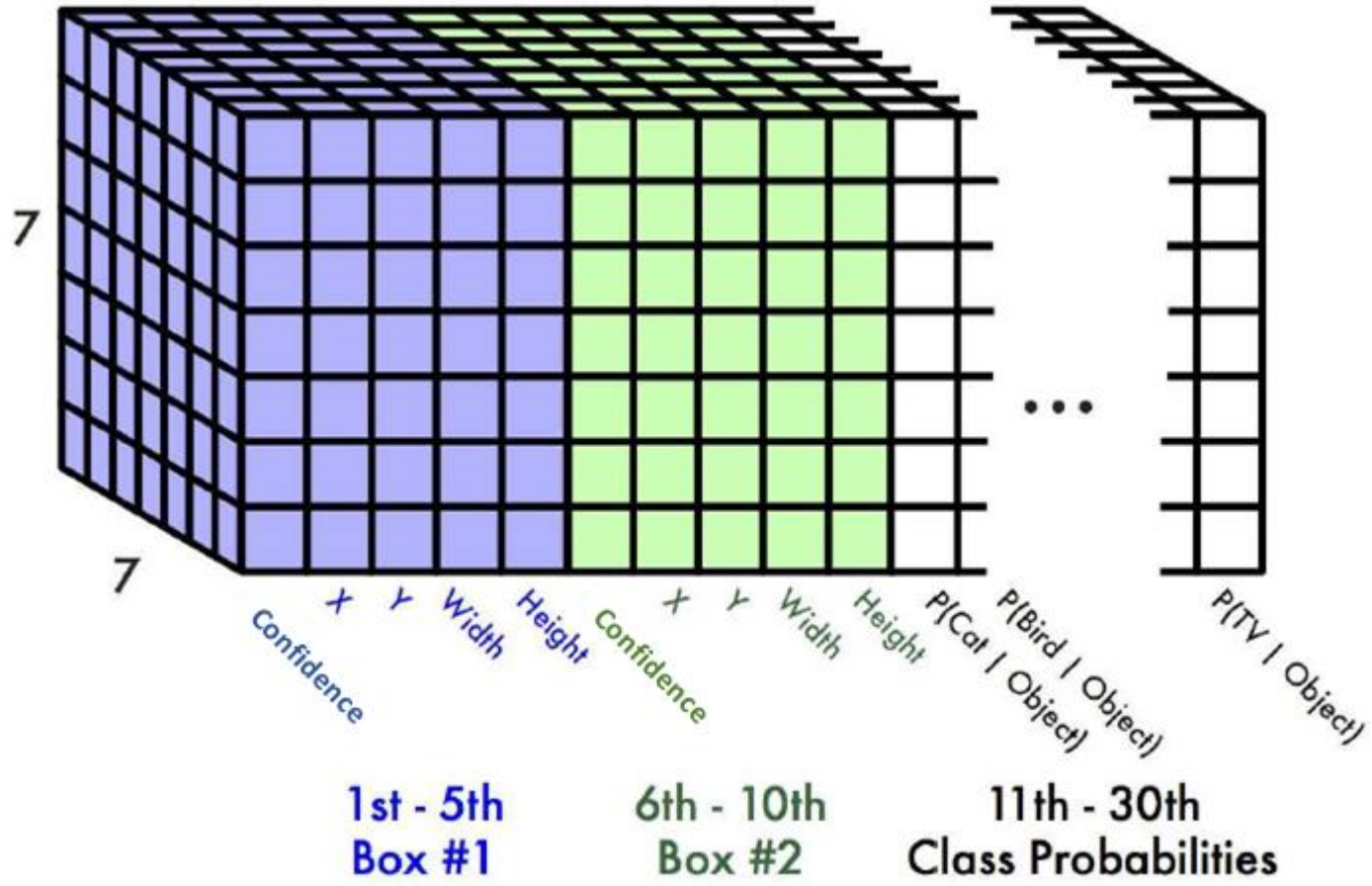
2: 2個Boundarybox

5: 每個Boundarybox (x, y, w, h, confidence)

20:屬於20個類別的機率。

YOLOv1的detector怎麼處理

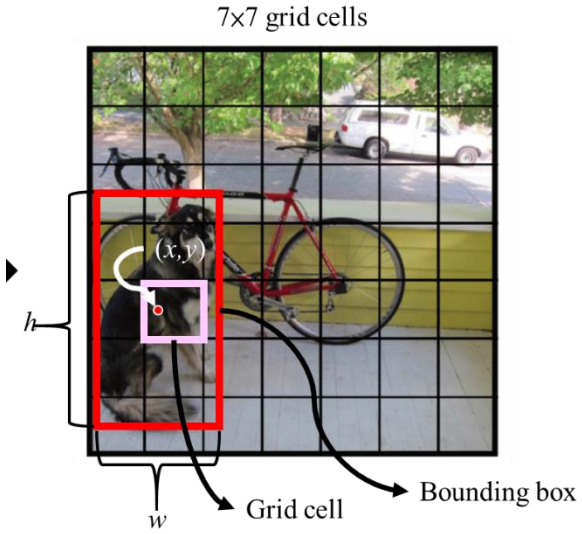
- 最後一層是7*7*30



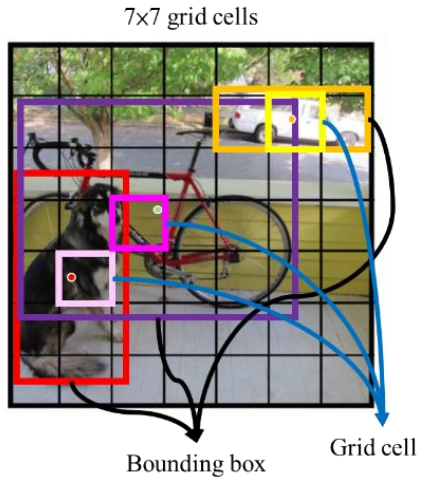
YOLOv1的detector怎麼處理

5: 每個Boundary box (x, y, w, h, confidence)

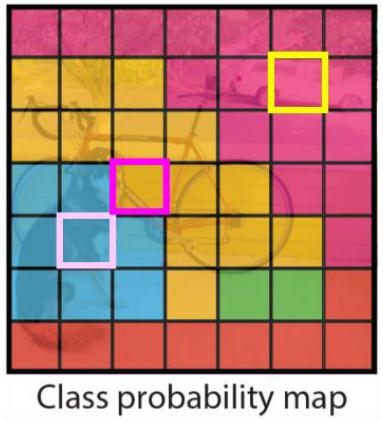
20:屬於20個類別的機率。



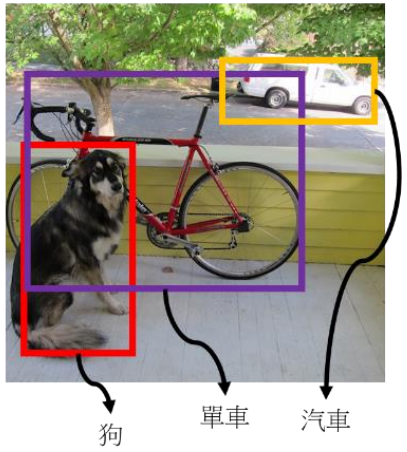
這個物件落在粉紅色框的grid cell
座標為這個grid cell內紅色框這個
Bounding box的中心(x,y)，高寬為h,w。



先利用閾值和NMS選出確定是物
件的**Bounding box**



看選出物件所屬的grid cell屬於哪
一類的機率最大。
水藍色:「狗」的機率最大
黃色:「單車」的機率最大
粉紅:「汽車」的機率最大
橘色:「地板」的機率最大



YOLOv1 loss function

物件的中心座標(x, y)和模型預測BBOX座標$(\hat{x}, \hat{y})$的均方差和

物件的長寬(w, h)和模型預測BBOX長寬$(\hat{w}, \hat{h})$的均方差和

這個grid cell有沒有物件的信心度

這個物件被判斷成每個類別的機率

$$\text{loss}_{YOLO} = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left[\left(x_i - \hat{x}_i\right)^2 + \left(y_i - \hat{y}_i\right)^2 \right]$$

$$+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2 \right]$$

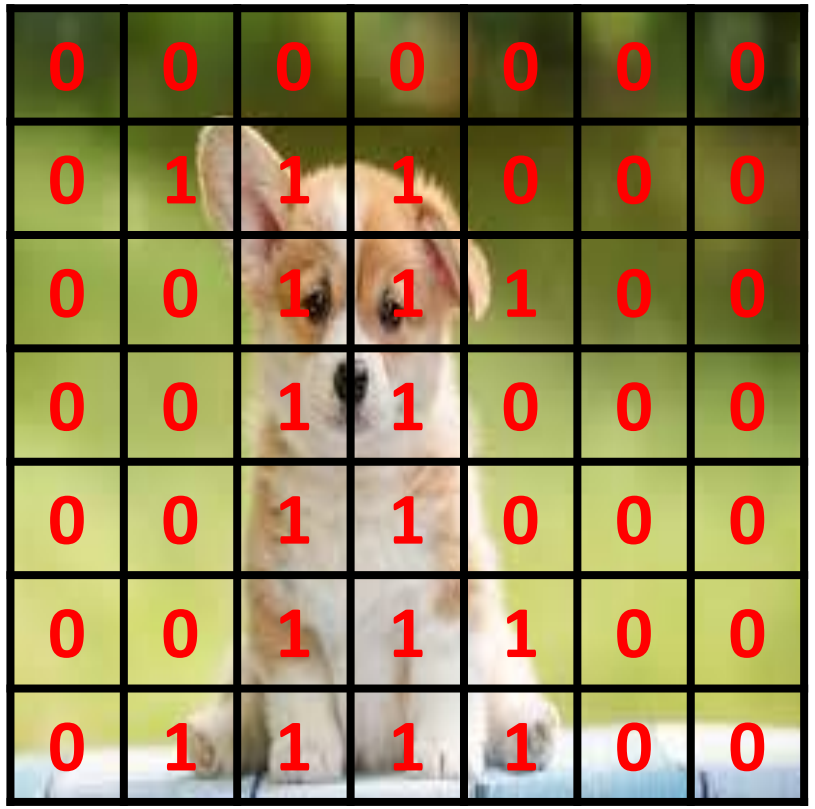
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left(C_i - \hat{C}_i\right)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{noobj} \left(C_i - \hat{C}_i\right)^2$$

$$+ \sum_{i=o}^{S^2} 1_i^{obj} \sum_{c \in classes} \left(p_i(c) - \hat{p}_i(c)\right)^2$$

1_i^{obj}: 是物件有出現在 grid cell i。

1_{ij}^{obj}:是在第 i 個 grid cell 的第 j 個 Bounding Box 負責做預測。

是不是有物件



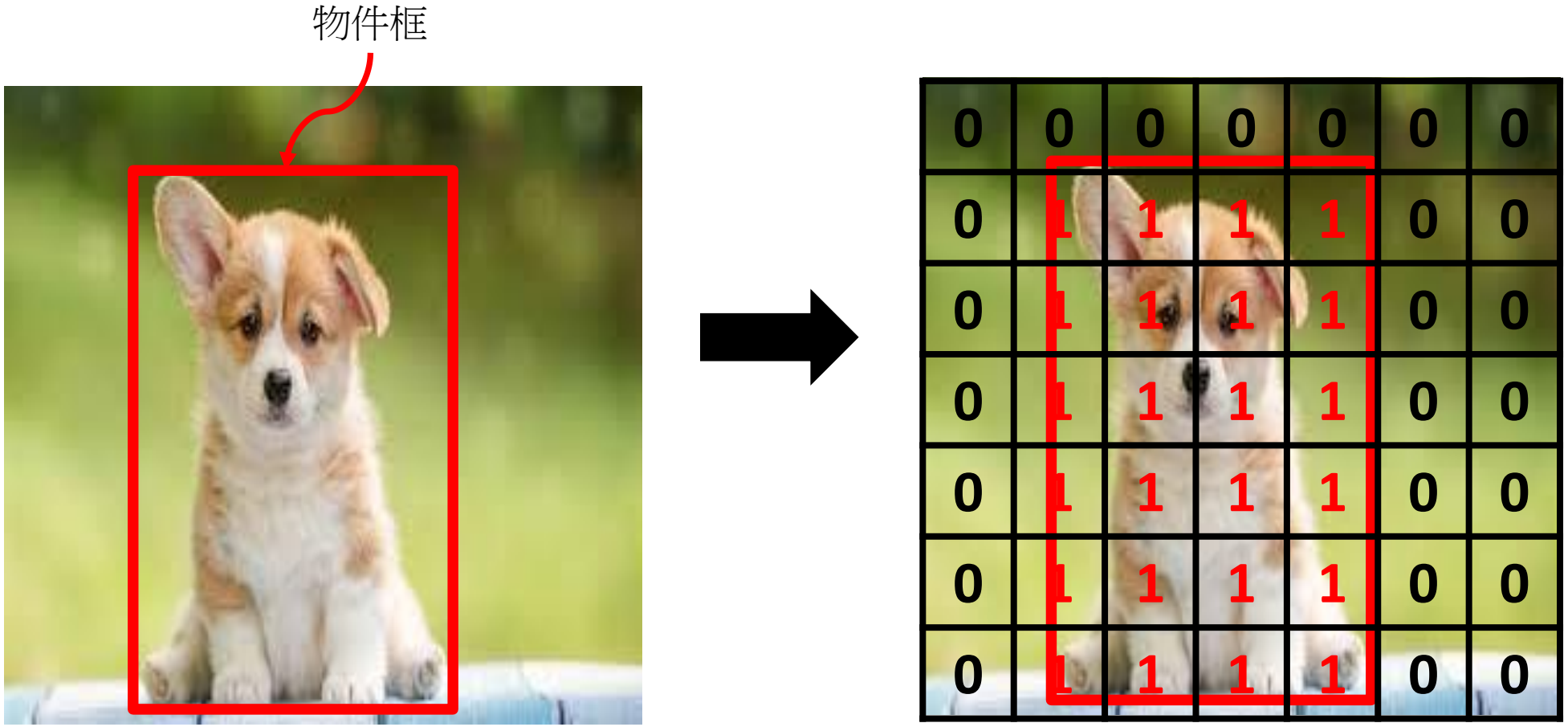
1^{obj}

0	0	0	0	0	0	0
0	1	1	1	0	0	0
0	0	1	1	1	0	0
0	0	1	1	0	0	0
0	0	1	1	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	0	0

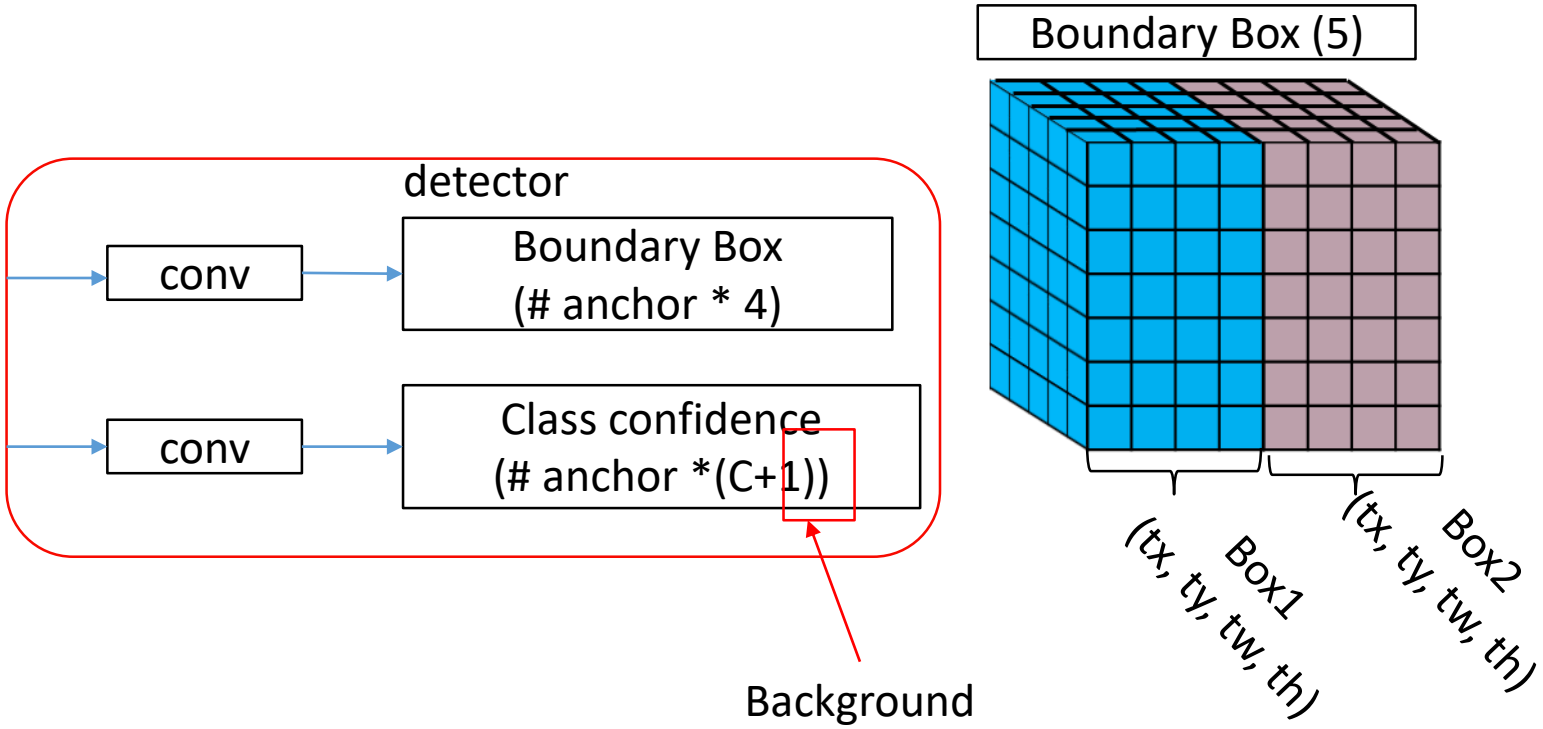
1^{noobj}

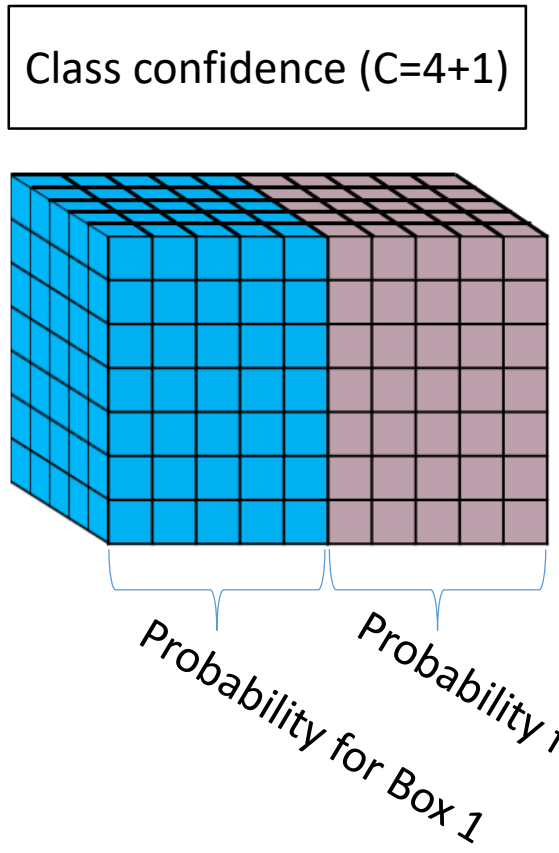
1	1	1	1	1	1	1
1	0	0	0	1	1	1
1	1	0	0	0	1	1
1	1	0	0	1	1	1
1	1	0	0	1	1	1
1	1	0	0	0	1	1
1	0	0	0	0	1	1

是不是有物件

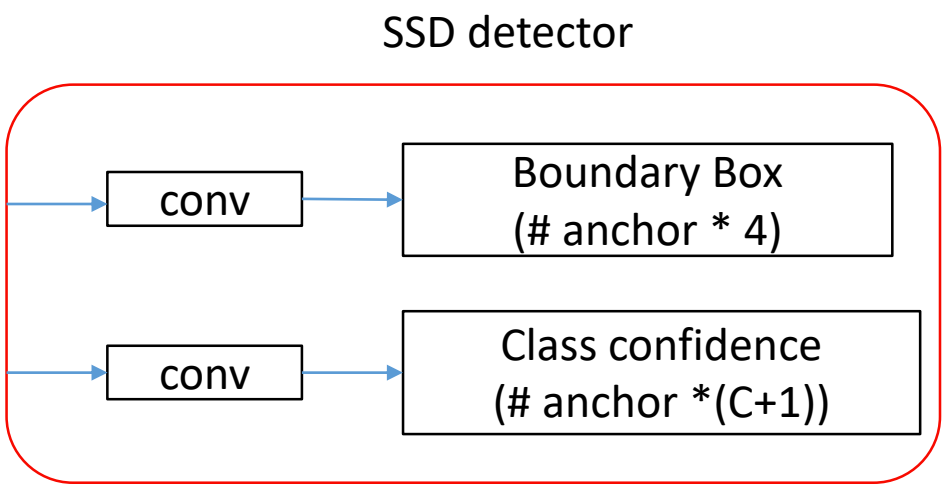

物件框
0 0 0 0 0 0 0
0 1 1 1 1 0 0
0 1 1 1 1 0 0
0 1 1 1 1 0 0
0 1 1 1 1 0 0
0 1 1 1 1 0 0
0 1 1 1 1 0 0

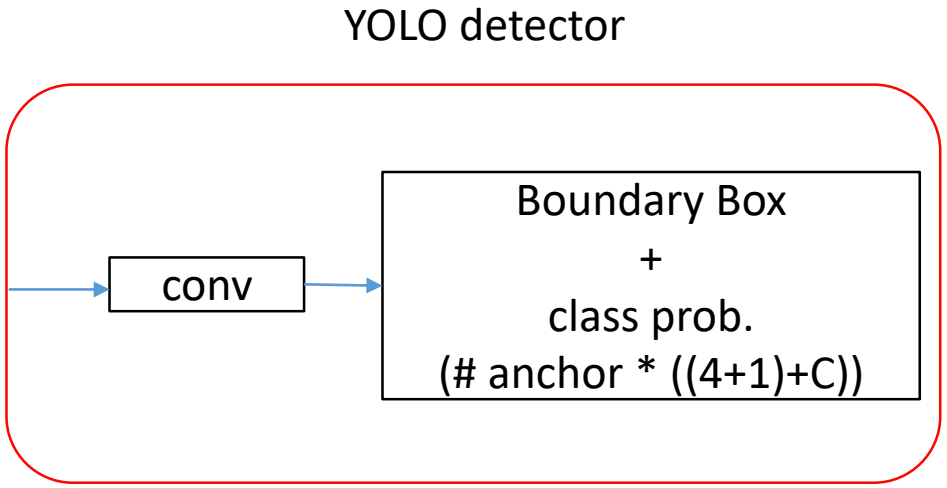
SSD detector





SSD vs YOLO

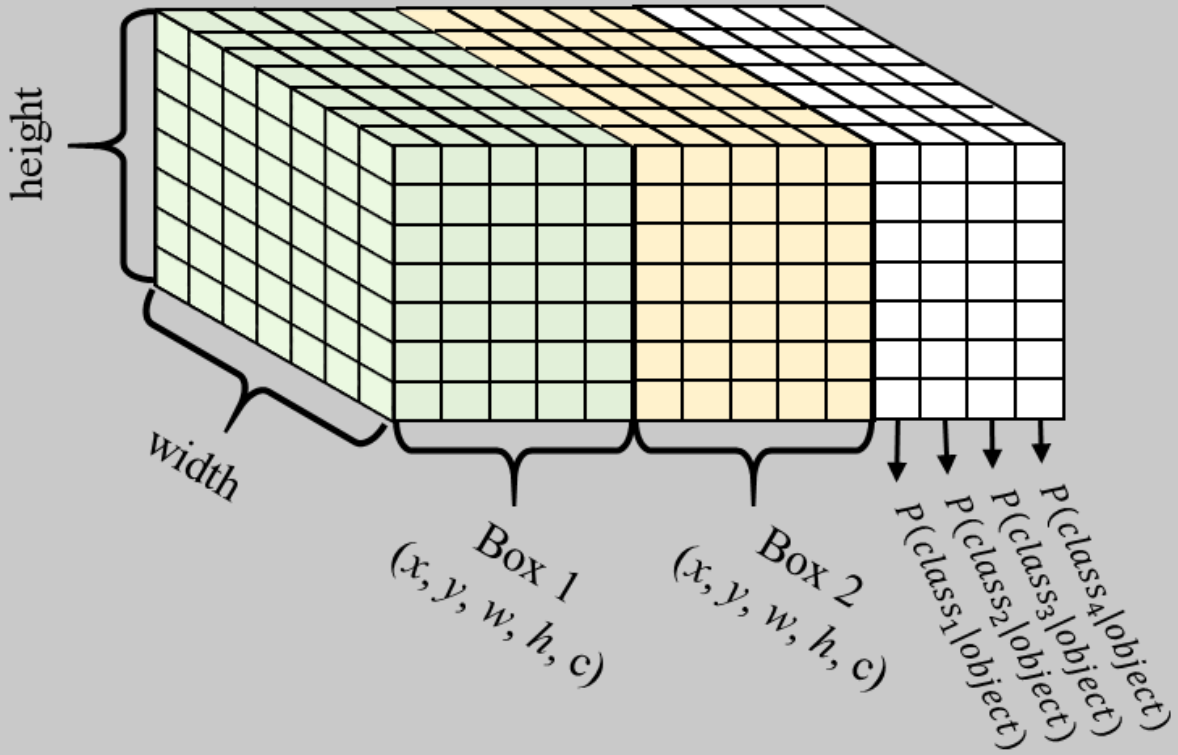

SSD detector
conv
Boundary Box
(# anchor * 4)
conv
Class confidence
(# anchor *(C+1))


YOLO detector
conv
Boundary Box
+
class prob.
(# anchor * ((4+1)+C))

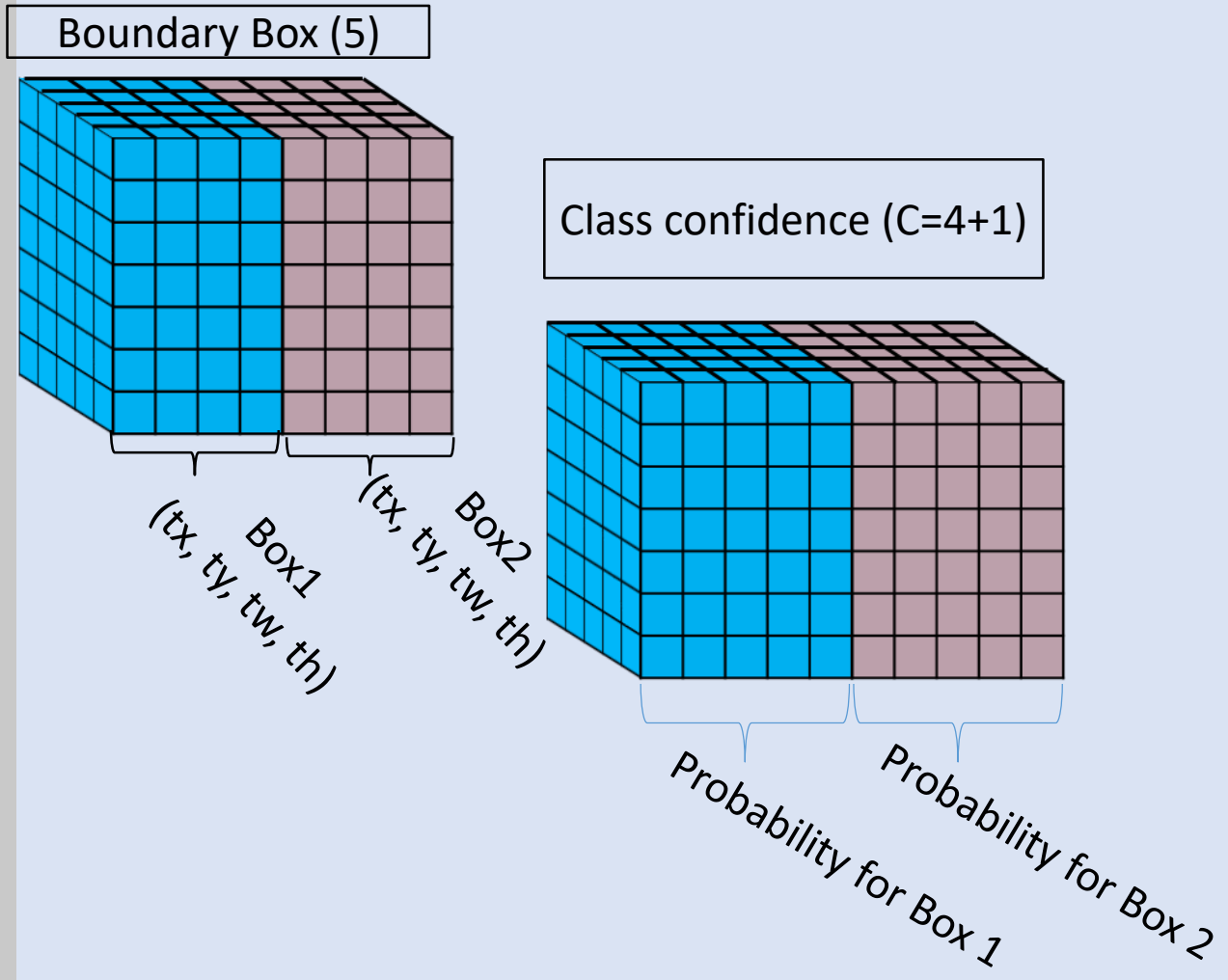
SSD vs YOLO

YOLOv1
 # class =4

- 最後一層是7*7*14



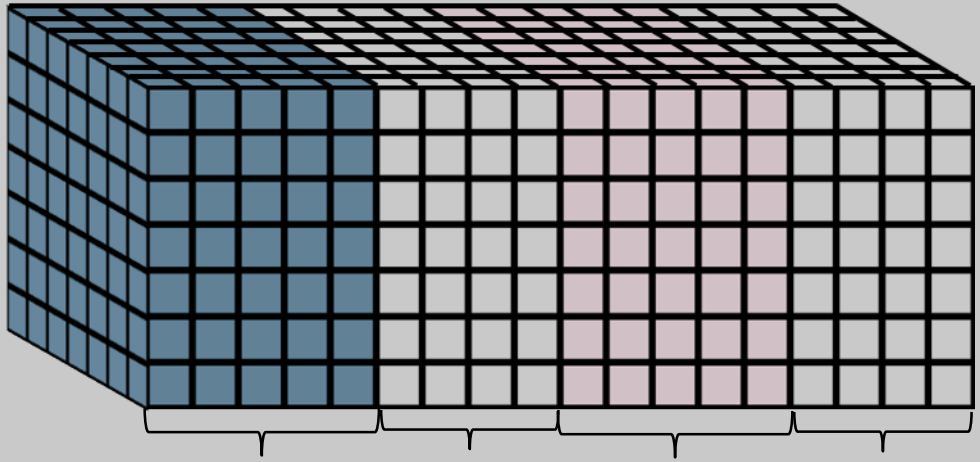
SSD
 # class =4



SSD vs YOLO

YOLOv2↑
 # class =4

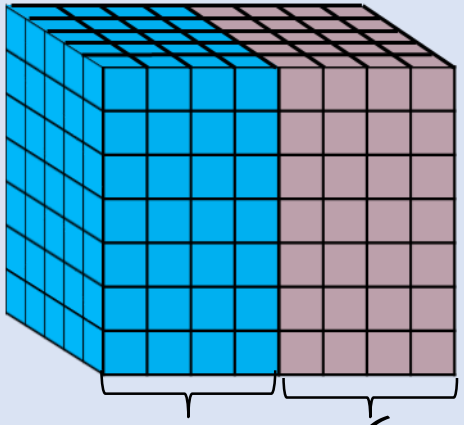
#Anchor * (5+4)



Box1 (Confidence, tx, ty, tw, th)
Box 1 Class Probabilities (c=4)
Box2 (Confidence, tx, ty, tw, th)
Box 2 Class Probabilities (c=4)

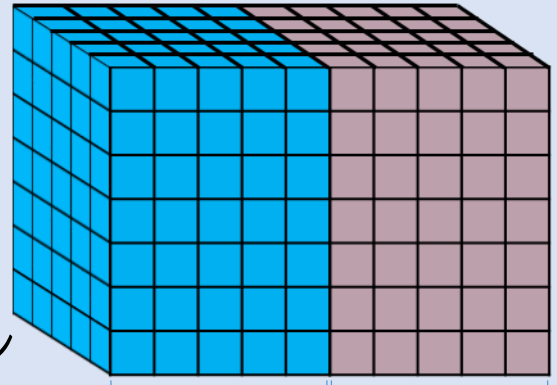
SSD
 # class =4

Boundary Box (5)



Box1 (tx, ty, tw, th)
Box2 (tx, ty, tw, th)

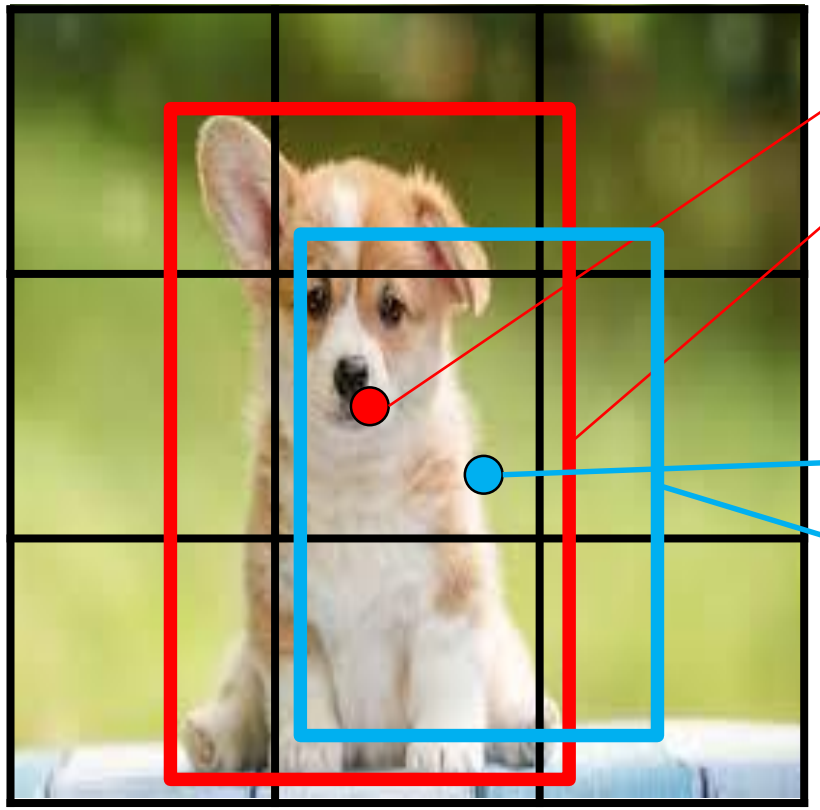
Class confidence (C=4+1)



Probability for Box 1
Probability for Box 2

YOLOv1和v2以上

- **YOLOv1:　(Confidence, x, y, w, h)**
- YOLOv2个: (Confidence, tx, ty, tw, th)



$(x, y) = (0.5, 0.5)$

$(w, h) = (0.4, 0.9)$

$(\hat{x}, \hat{y}) = (0.6, 0.7)$

$(\hat{w}, \hat{h}) = (0.3, 0.6)$

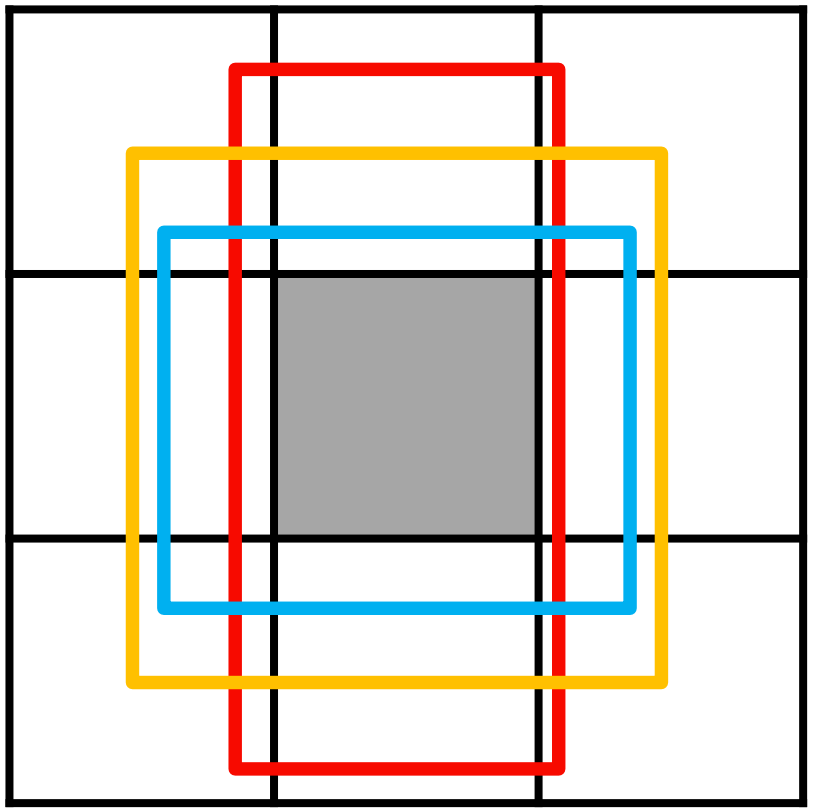
$$(x - \hat{x})^2 + (y - \hat{y})^2 = 0.05$$

$$\left(\sqrt{w} - \sqrt{\hat{w}}\right)^2 + \left(\sqrt{h} - \sqrt{\hat{h}}\right)^2 = 0.0375$$

YOLOv2後引入Anchor的想法

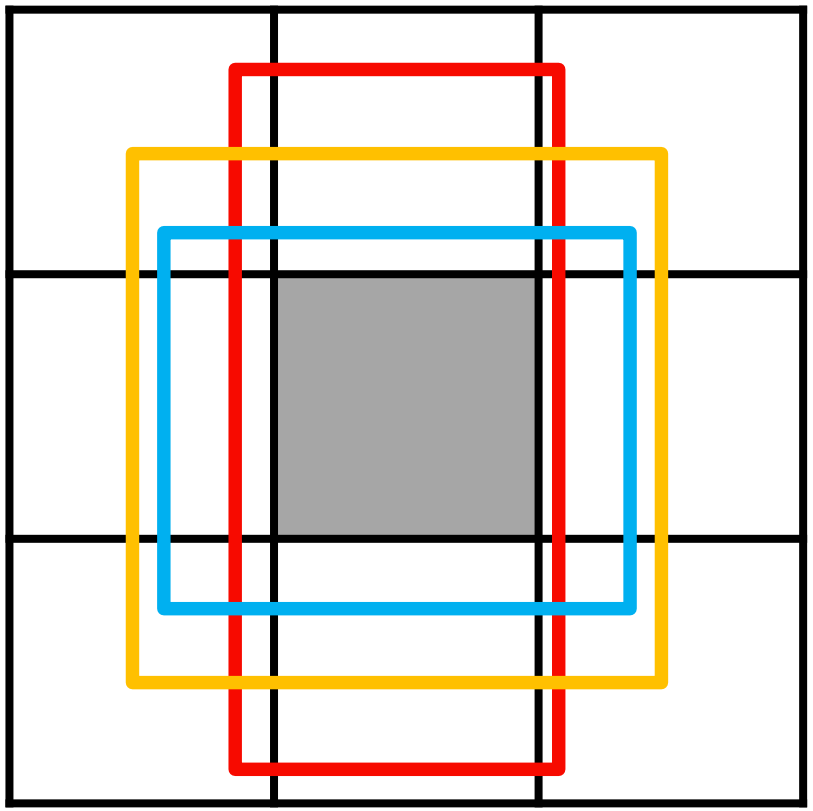
在最後預測的S*S 的每個grid cell都預設幾個Anchors。

範例為3*3的特徵圖，共有9個grid cells，每個cell內預設3個anchors (紅色、黃色和藍色框)

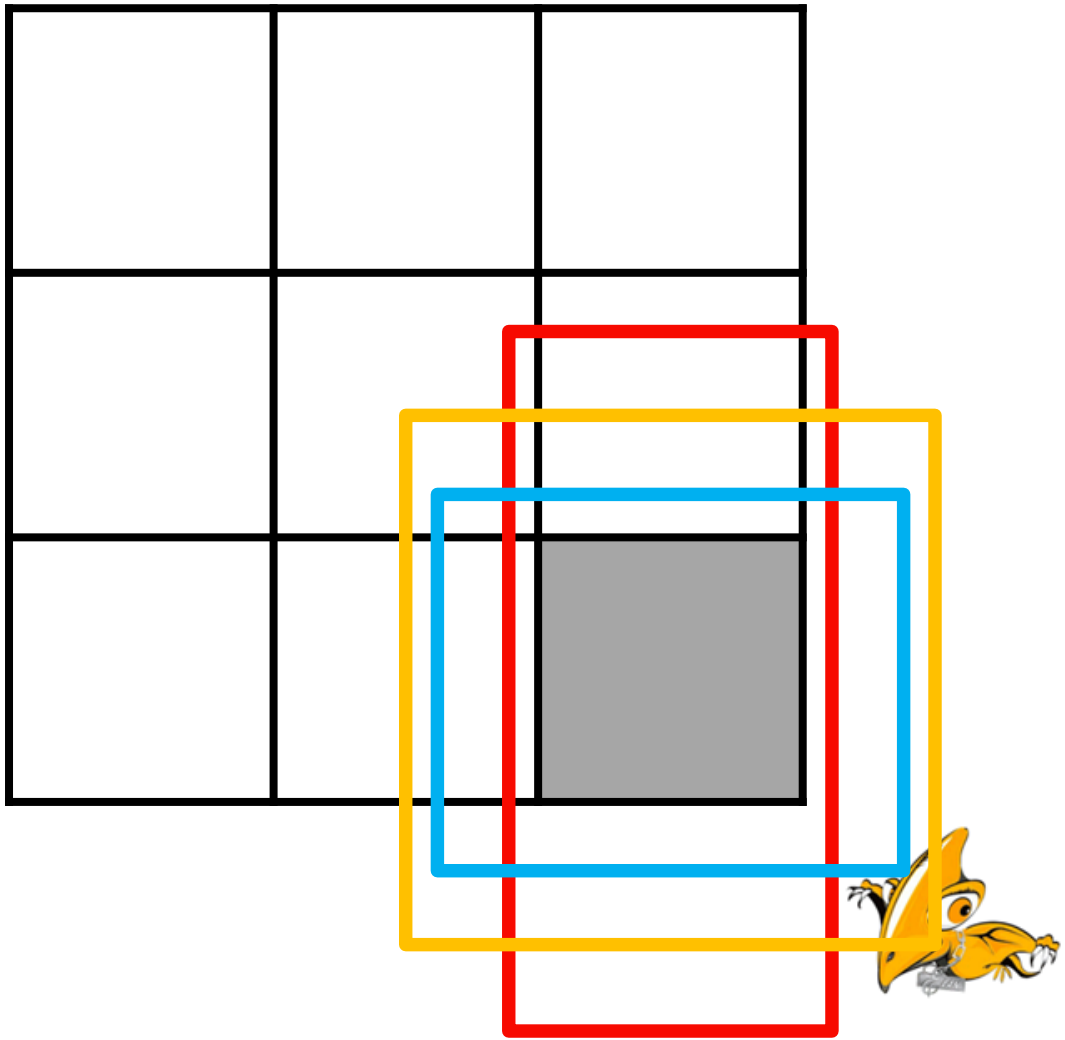


YOLOv2後引入Anchor的想法

在最後預測的S*S 的每個grid cell都預設幾個anchors



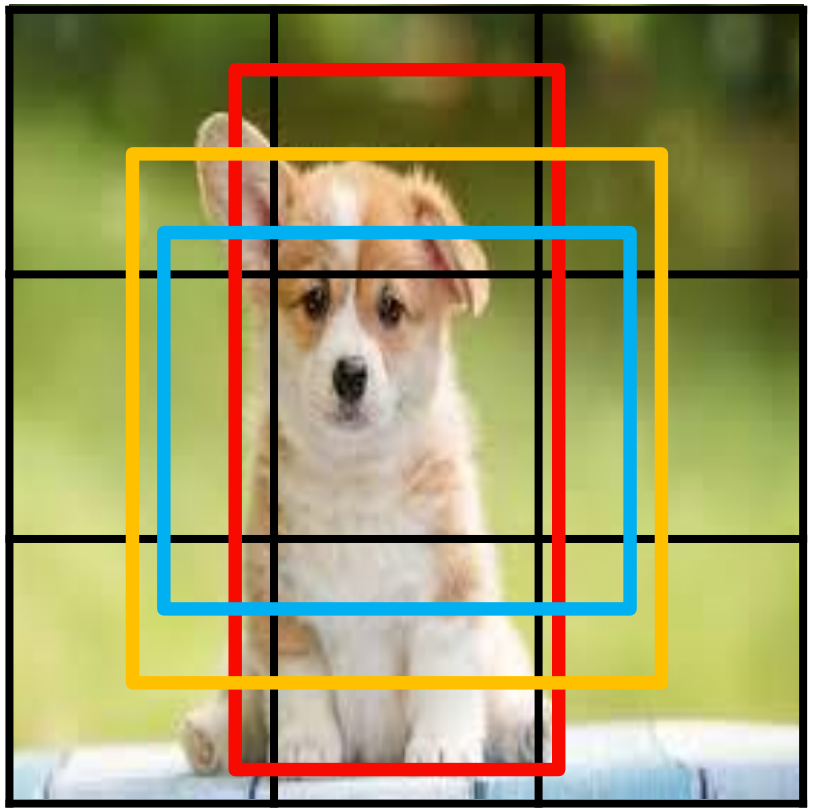
...



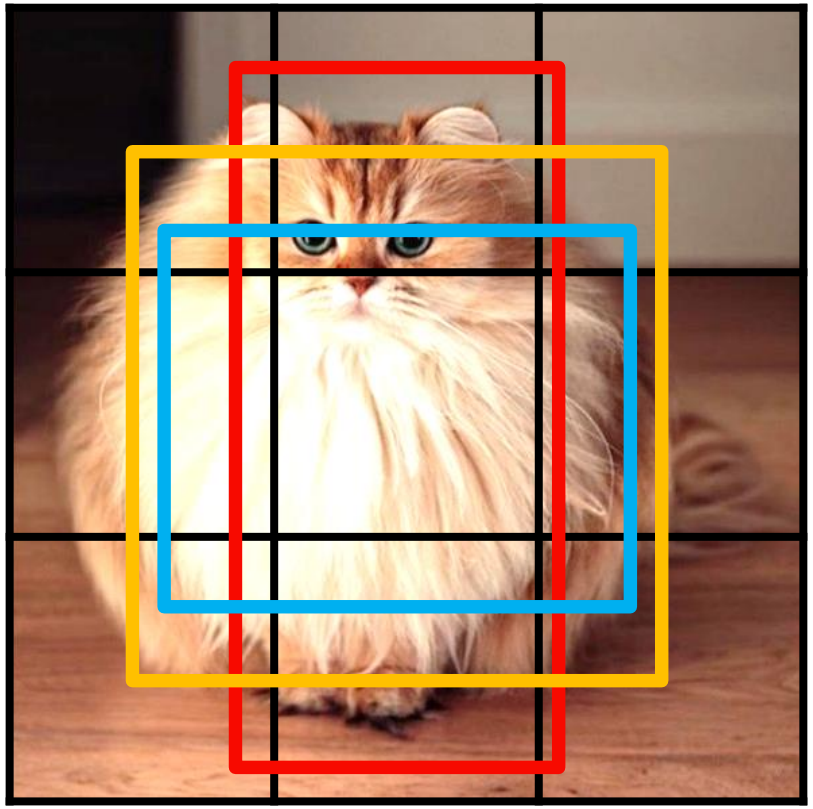
YOLOv2後引入Anchor的想法

不同的物件大小用不同anchors去Fit，所以我們在預測的時候只要去調整Anchor的大小來Fit物件就好。

紅色的Anchor比較適合狗



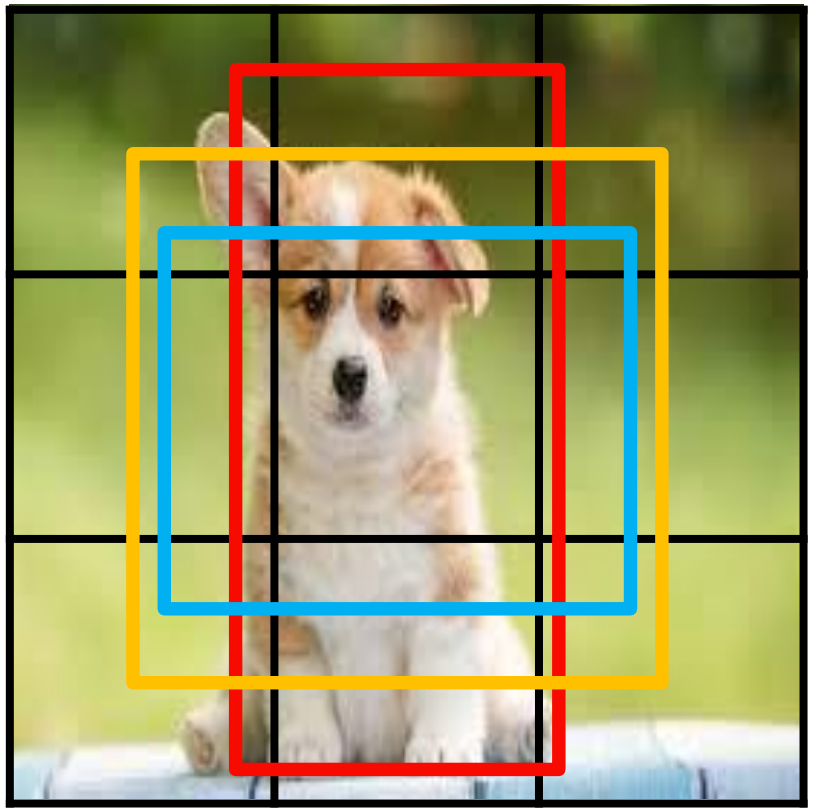
黃色的Anchor比較適合貓



YOLOv2後引入Anchor的想法

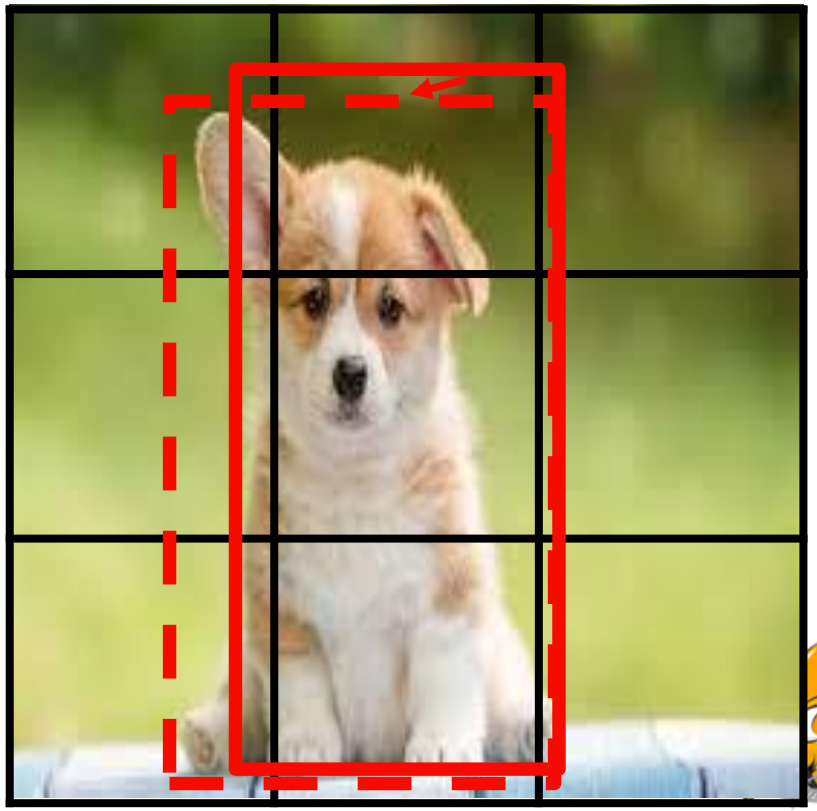
不同的物件大小用不同anchors去Fit，所以我們在預測的時候只要去調整Anchor的大小來Fit物件就好。

紅色的Anchor比較適合狗



神經網路學習
調整紅色的Anchor去fit狗

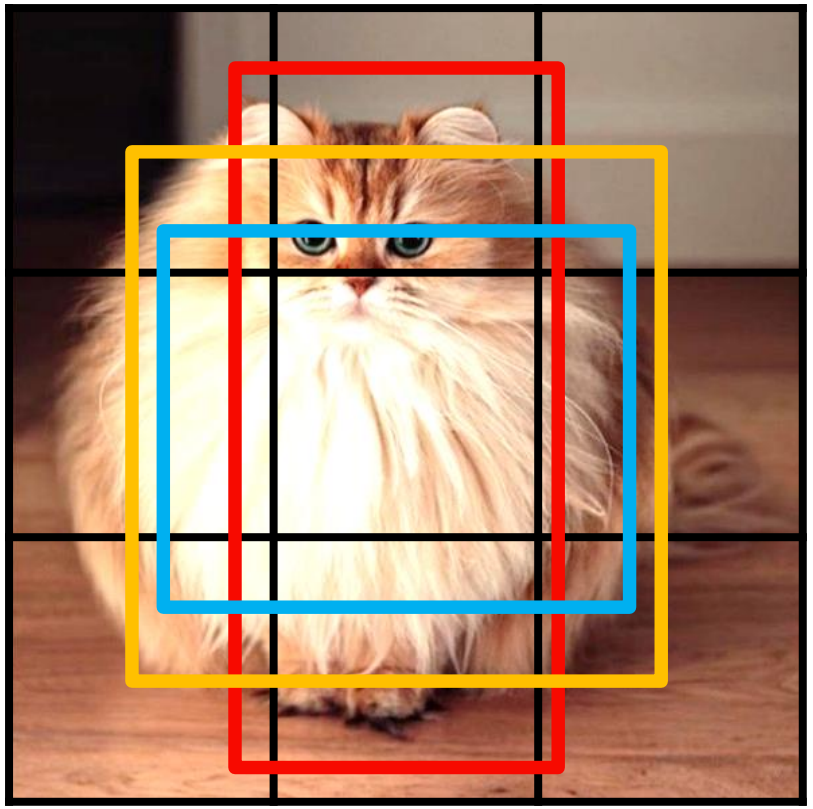
紅色anchor高縮小一點，往左下移動一點



YOLOv2後引入Anchor的想法

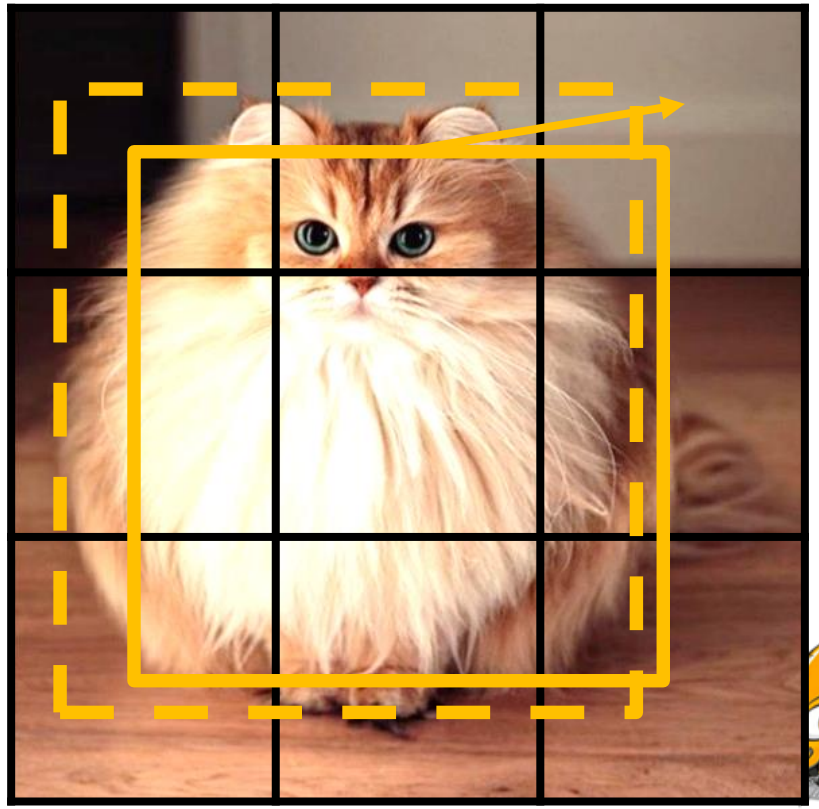
不同的物件大小用不同anchors去Fit，所以我們在預測的時候只要去調整Anchor的大小來Fit物件就好。

黃色的Anchor比較適合貓



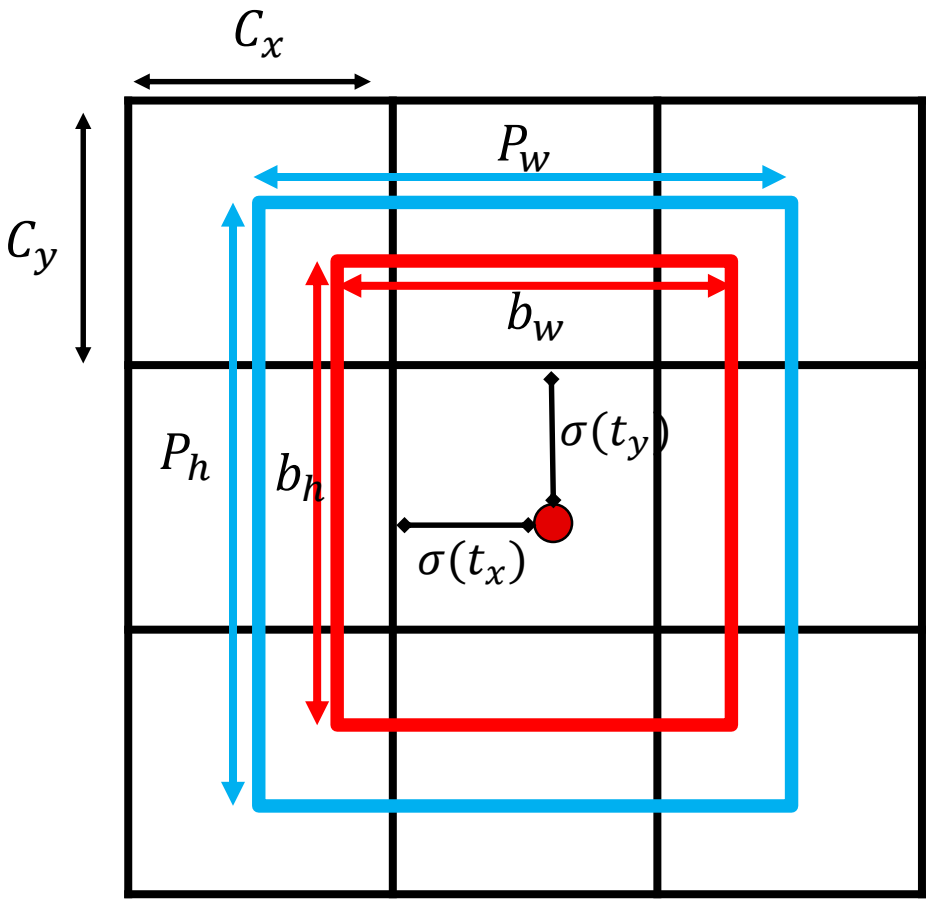
神經網路學習
調整黃色的Anchor去fit貓

黃色anchor長寬放大一點，中心稍移動



YOLO如何推算物件座標和大小

- 學習調整 Anchor (放大縮小和中心位移)
- **YOLOv2↑: (Confidence, tx, ty, tw, th)**



YOLO模型輸出(t_x, t_y, t_w, t_h)

模型輸出要轉換成為實際的物件中心座標(b_x, b_y)和寬高(b_w, b_h)

中心座標轉換:

$$b_x = \sigma(t_x) + C_x$$
$$b_y = \sigma(t_y) + C_y$$

寬高轉換由Anchor去進行縮放:

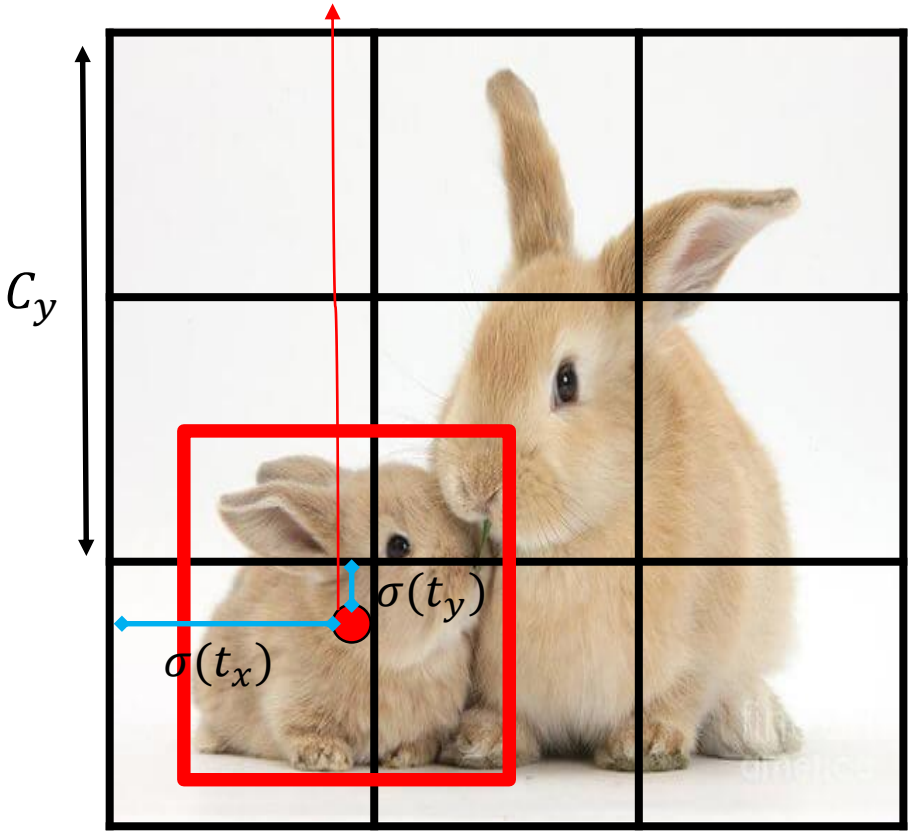
$$b_w = P_w e^{t_w}$$
$$b_h = P_h e^{t_h}$$

(P_w, P_h): 我們設定的Anchor寬高。

YOLO如何推算物件座標和大小

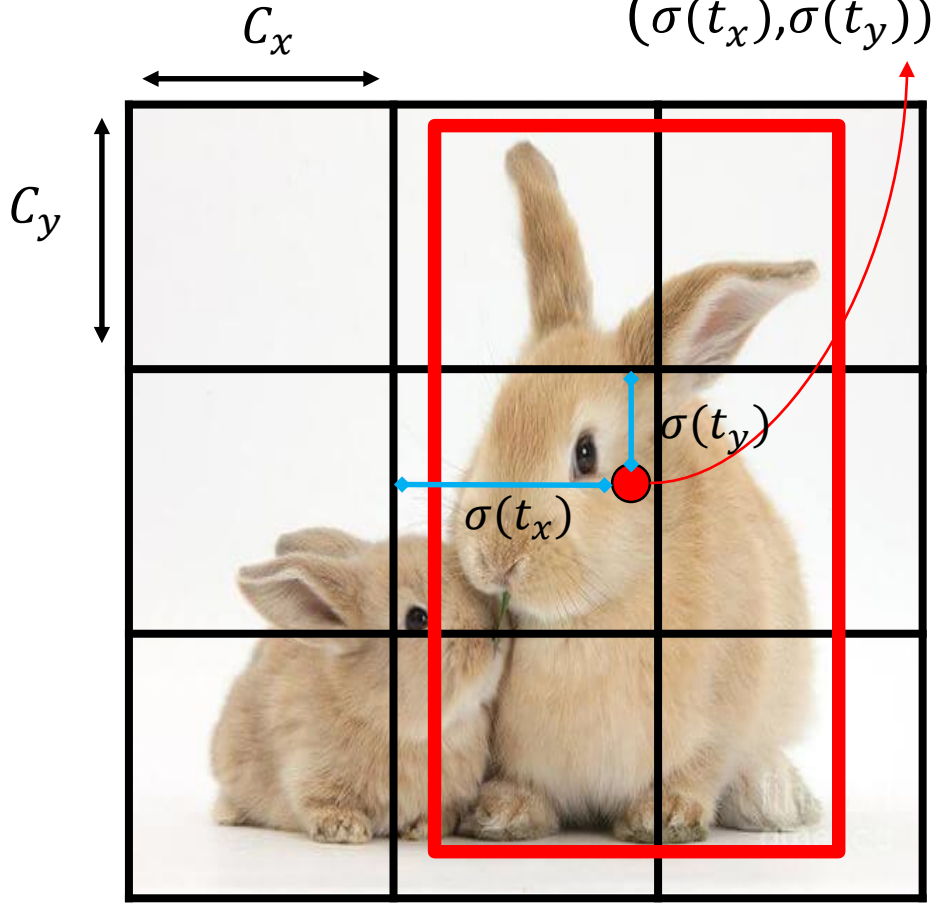
(C_x, C_y)=(0, 2)

$(\sigma(t_x),\sigma(t_y))$=(0.9, 0.5)



(C_x, C_y)=(1, 1)

$(\sigma(t_x),\sigma(t_y))$=(0.9, 0.5)

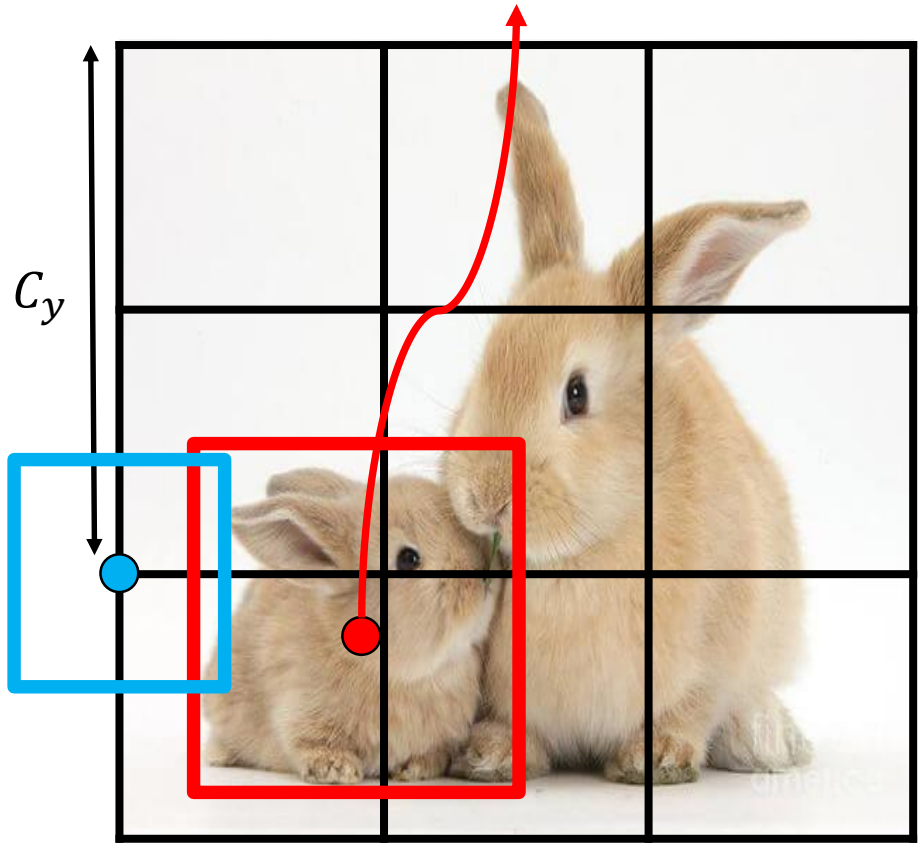


YOLO如何推算物件座標和大小

神經網路學習
調整藍色的Anchor去fit狗(兔子的紅框)

Anchor:(P_w, P_h)=(0.3, 0.3)

$$(x, y) = (0.3, 0.7\ ) \Longrightarrow (b_x, b_y) = (0.9, 2.5)$$
$$(w, h) = (0.4\ 0.4\ )$$



$$0.9 = b_x = \sigma(t_x) + C_x$$
$$2.5 = b_y = \sigma(t_y) + C_y$$
$$0.4 = b_w = P_w e^{t_w}$$
$$0.4 = b_h = P_h e^{t_h}$$

其中C_x=0, C_y=2

所以

$$\sigma(t_x) = b_x - C_x = 0.9$$
$$\sigma(t_y) = b_y - C_y = 0.5$$
$$e^{t_w} = \frac{b_w}{P_w} = \frac{0.4}{0.3} = 1.33 \Longrightarrow t_w = \log(1.33)$$
$$e^{t_h} = \frac{b_h}{P_h} = \frac{0.4}{0.3} = 1.33 \Longrightarrow t_h = \log(1.33)$$

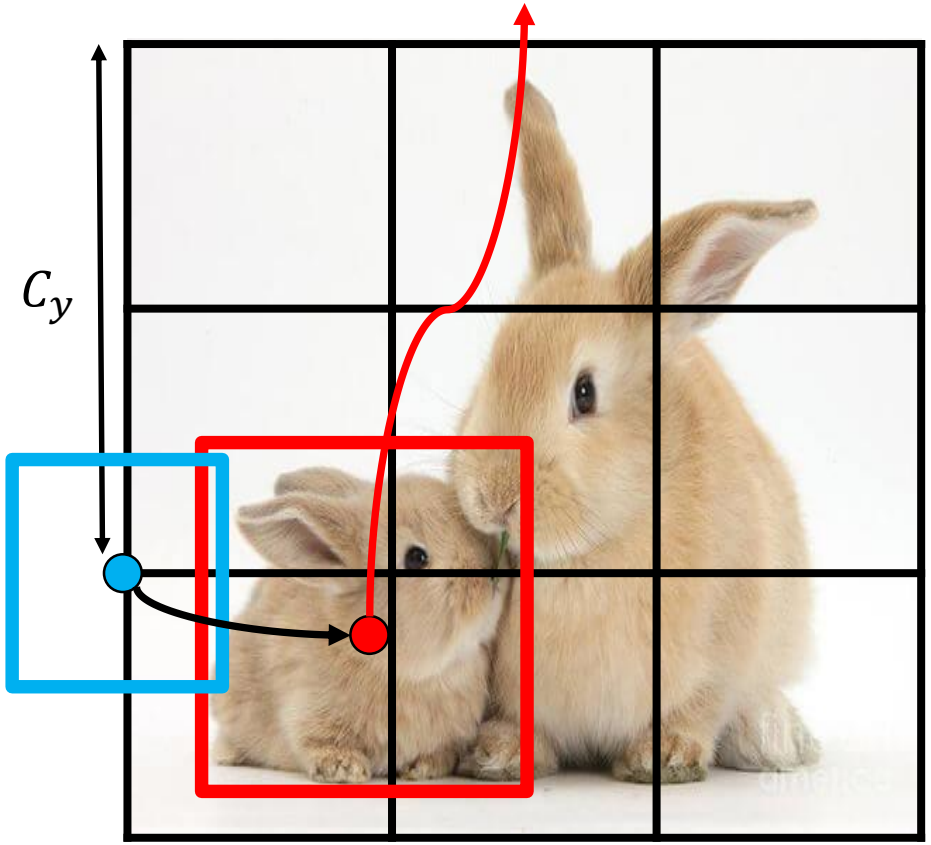
YOLO如何推算物件座標和大小

神經網路學習

調整藍色的Anchor去fit狗(兔子的紅框)

$$(x, y) = (0.3, 0.7\ ) \Longrightarrow (b_x, b_y) = (0.9, 2.5)$$

$$(w, h) = (0.4\ 0.4\ )$$



YOLO這個物件的答案是:

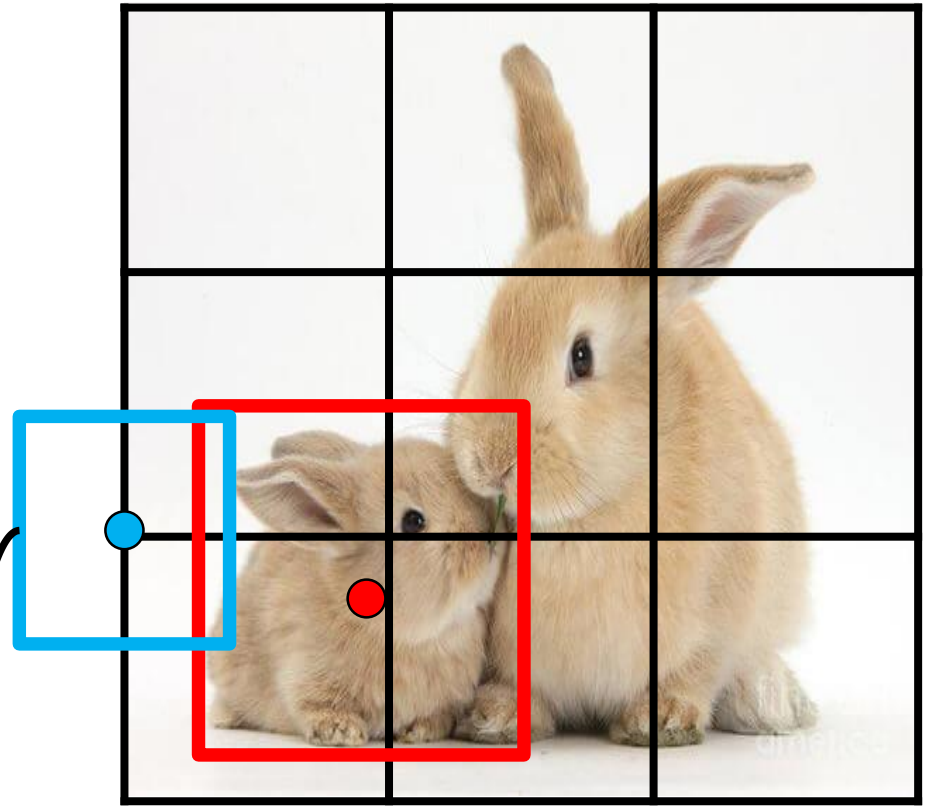
$$\sigma(t_x) = 0.9$$
$$\sigma(t_y) = 0.5$$
$$t_w = \log(1.33)$$
$$t_h = \log(1.33)$$

YOLO模型輸出$(\hat{t}_x, \hat{t}_y, \hat{t}_w, \hat{t}_h)$去逼近$(t_x, t_y, t_w, t_h)$

YOLOv1和V2以上差異

YOLOv1物件實際的框$(x, y, w, h) = (0.3, 0.7, 0.4, 0.4)$

YOLOv2↑物件實際的框$(b_x, b_y, b_w, b_h) = (0.9, 2.5, 0.4, 0.4)$



YOLOv2↑ 假設 Anchor:$(P_w, P_h)=(0.2, 0.2)$

$$b_x = \sigma(t_x) + C_x \Rightarrow \sigma(t_x) = 0.9 - 0 = 0.9$$

$$b_y = \sigma(t_y) + C_y \Rightarrow \sigma(t_y) = 2.5 - 2 = 0.5$$

$$0.4 = b_w = P_w e^{t_w} = 0.2e^{t_w} \Rightarrow e^{t_w} = 2 \Rightarrow t_w = \log(2)$$

$$0.4 = b_h = P_h e^{t_h} = 0.2e^{t_h} \Rightarrow e^{t_h} = 2 \Rightarrow t_h = \log(2)$$

$$(t_w - \hat{t}_w)^2 + (t_h - \hat{t}_h)^2 = 2 * (\log(2) - \log(1.5))^2 = 0.0312$$

$$loss_{v2} = 0.32 + 0.0312 = 0.3512$$

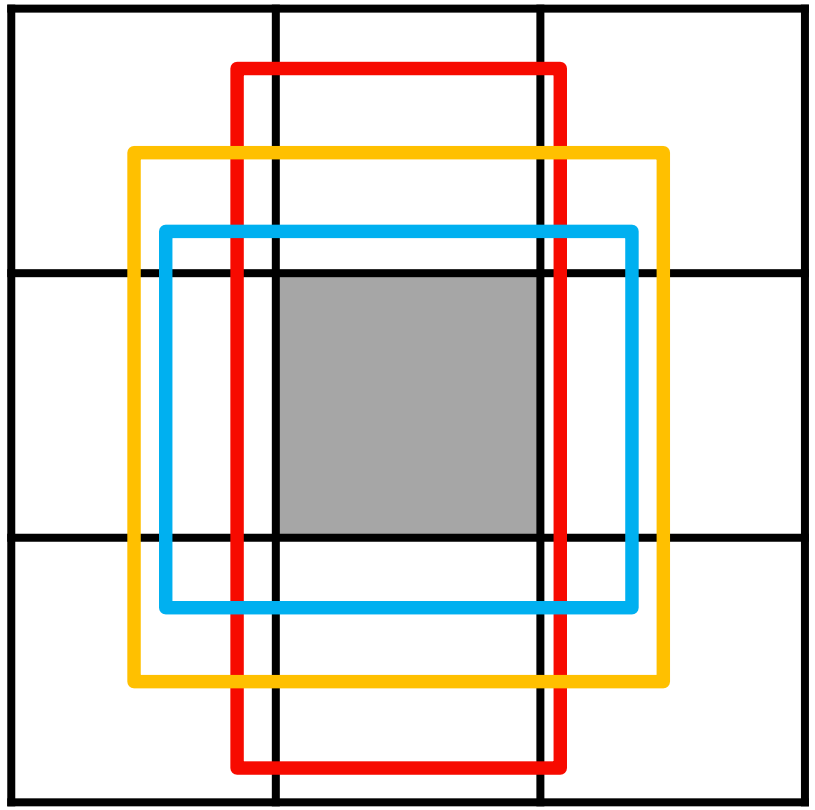
YOLOv1: 預測的框$(\hat{x}, \hat{y}, \hat{w}, \hat{h})$=(0.2,0.8, 0.3,0.3)

YOLOv2↑: 預測的框$(\sigma(\hat{t}_x), \sigma(\hat{t}_y), \hat{t}_w, \hat{t}_h)$=(0.5,0.5, log(1.5), log(1.5))

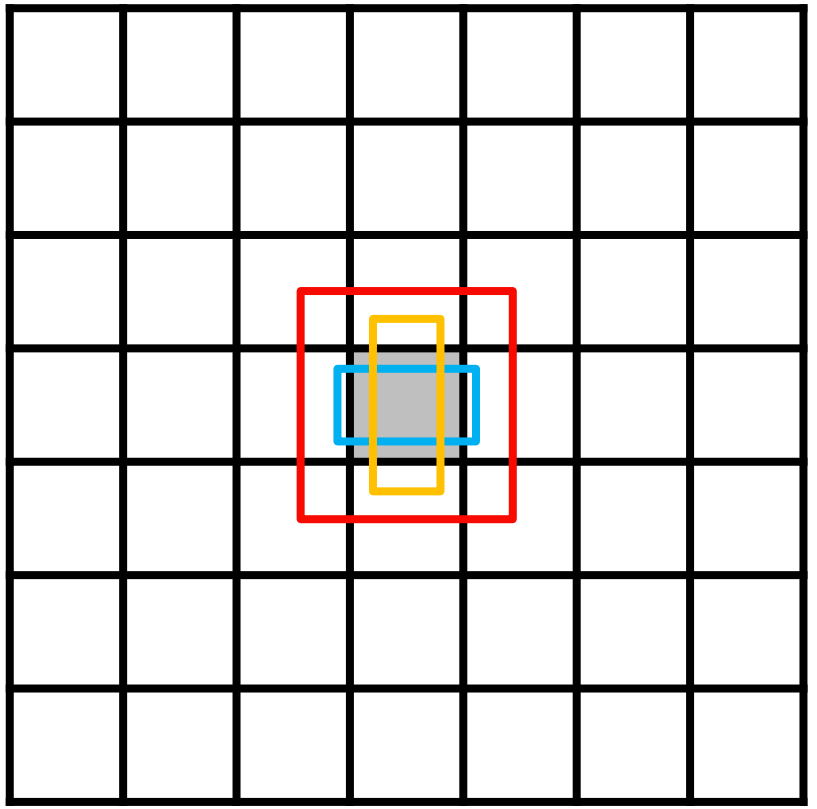
YOLOv3後引入Multi-scale Detection

在最後預測的S*S 的每個grid cell都預設幾個Anchors。

3*3的特徵圖



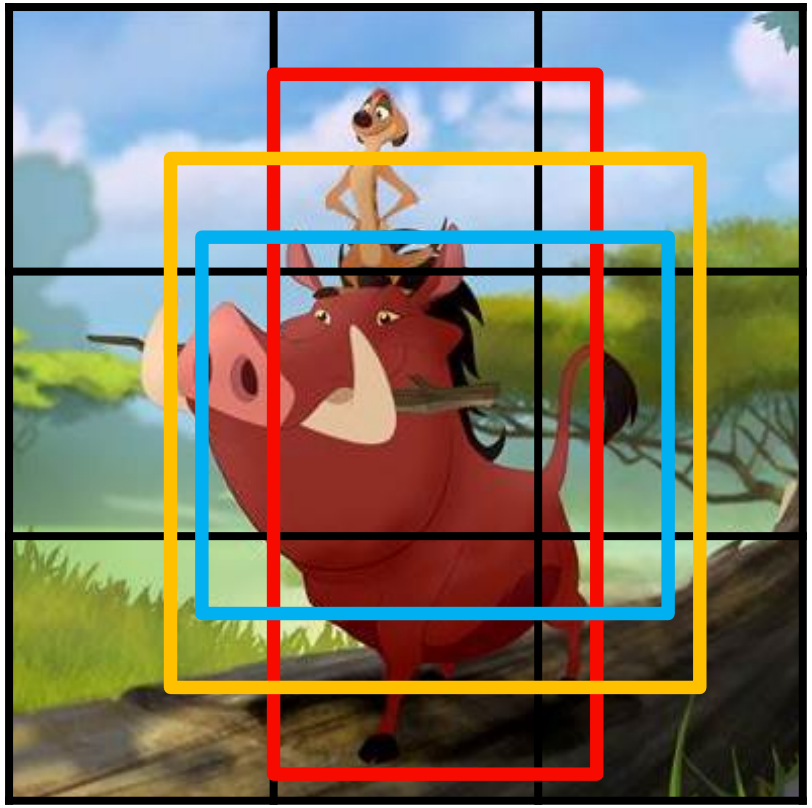
7*7的特徵圖



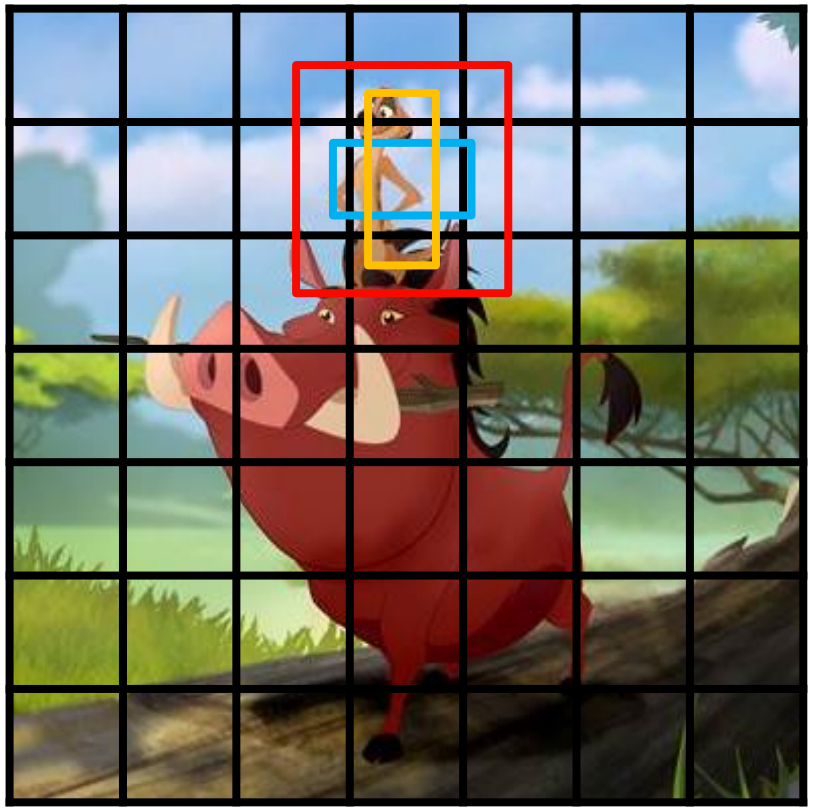
YOLOv3後引入Multi-scale Detection

在最後預測的S*S 的每個grid cell都預設幾個Anchors。

3*3的特徵圖



7*7的特徵圖



在小解析的特徵圖(3*3)適合最大物件偵測。

在大解析的特徵圖(7*7)適合最大物件偵測。

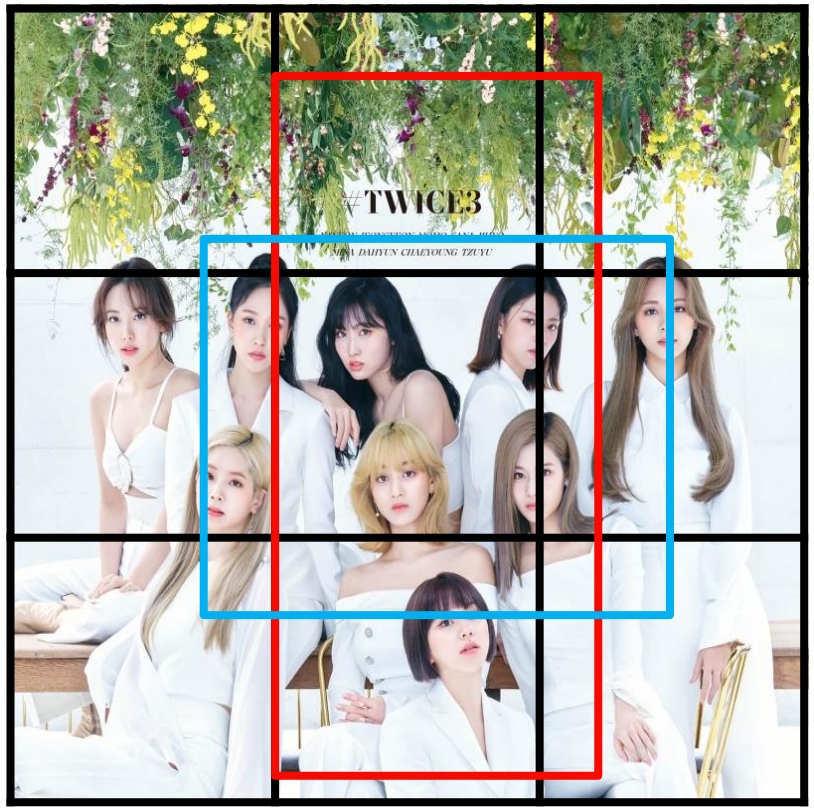
3*3特徵圖藍色的Anchor只需要微調就能fit彭彭

7*7特徵圖黃色的Anchor只需要微調就能fit丁滿

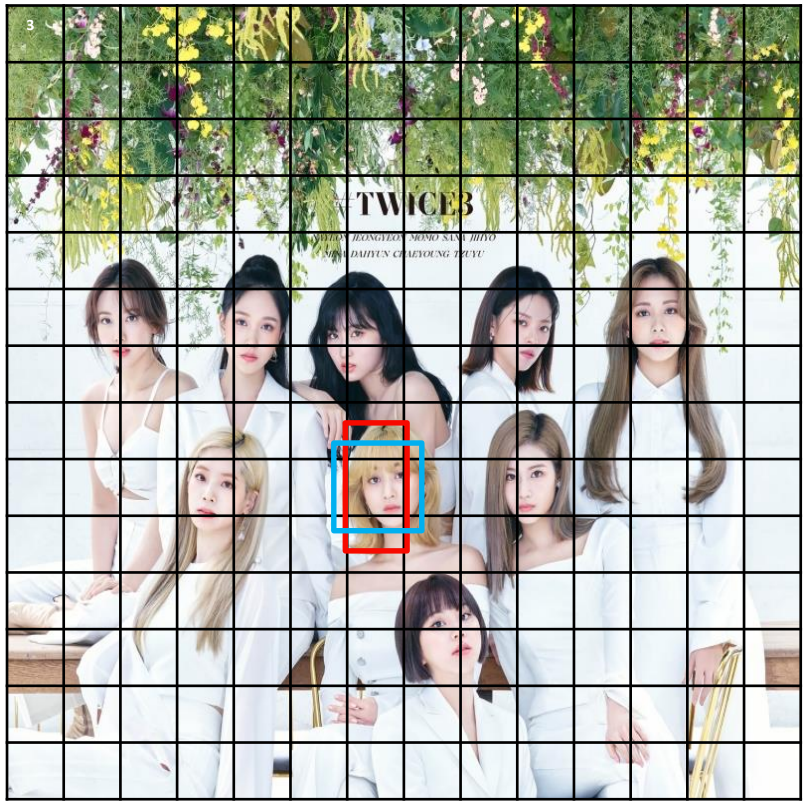
YOLOv3後引入Multi-scale Detection

人臉偵測

3*3的特徵圖



14*14的特徵圖



在3*3特徵圖設定的Anchor要縮小10倍，這樣的學習太難了