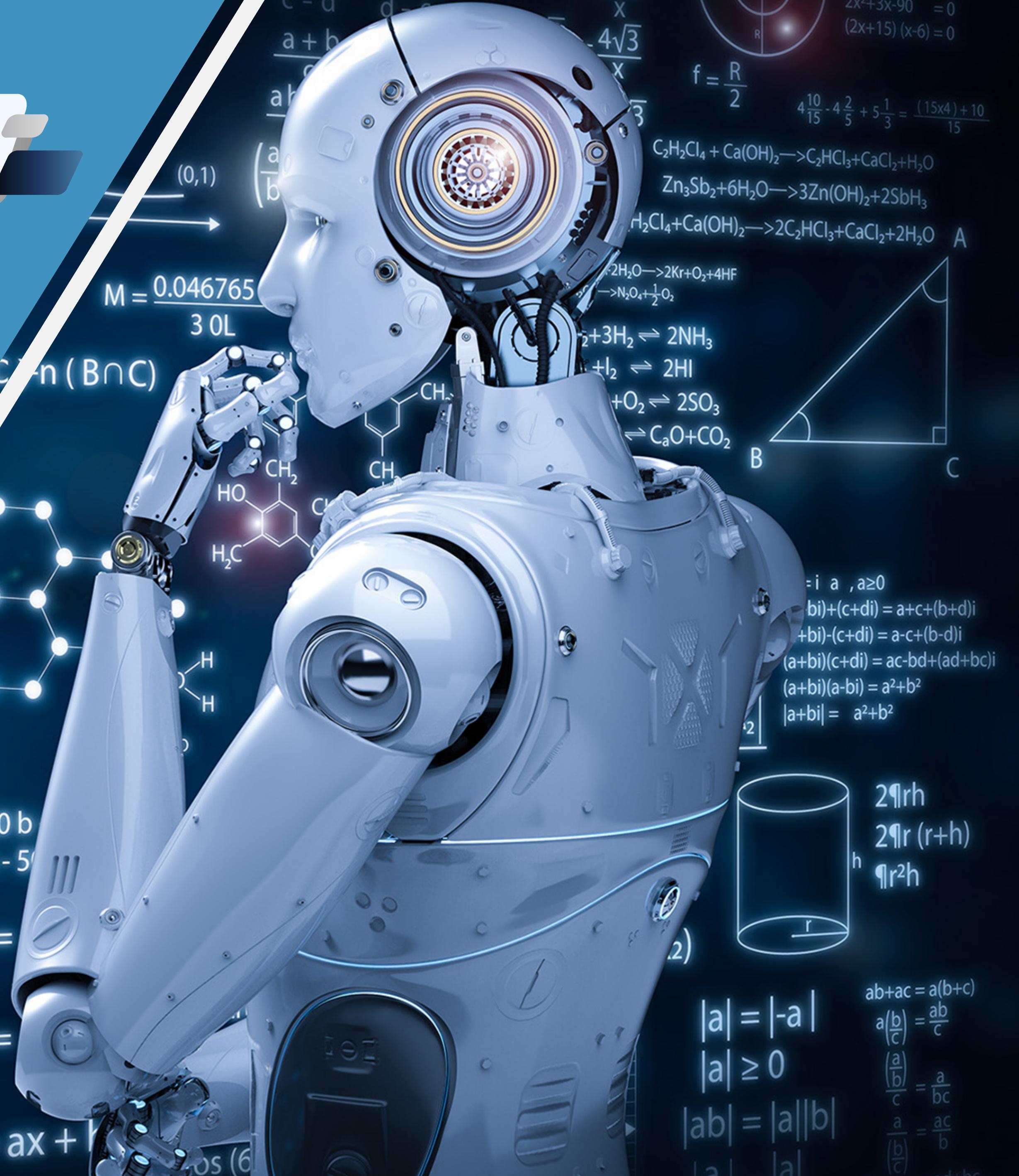


# Day 24

## 深度學習與電腦視覺 學習馬拉松

cupay 陪跑專家：楊哲寧





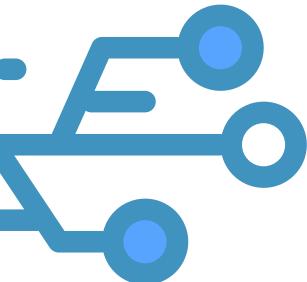
# 深度學習理論與實作

## Object Detection 演進



# 重要知識點

- 了解 Object Detection 的發展與各個模型的基本概念。
- 本章節不會詳細敘述技術部份，而是給學員們一個大方向的觀念，了解每個模型的優缺點。



# 演進

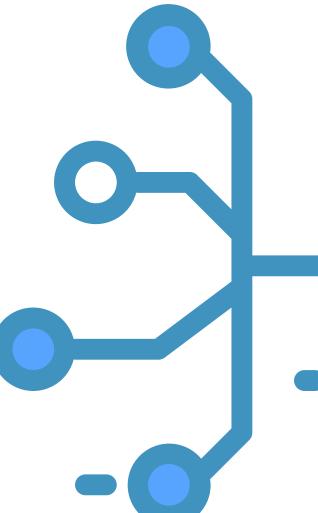


Object Detection 演算法主要有兩大分支，早期是以 Two Stage 系列為大宗，代表算法分別有

**R-CNN**

**Fast R-CNN**

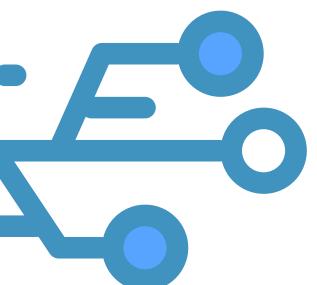
**Faster R-CNN**





**R-CNN**





# R-CNN



CUPOY

運用 Selective Search 提出

Region Proposal

Resize 到相同大小

透過 SVM 做分類，分類完再使用 Regressor 修正 BBOX

## R-CNN: *Regions with CNN features*

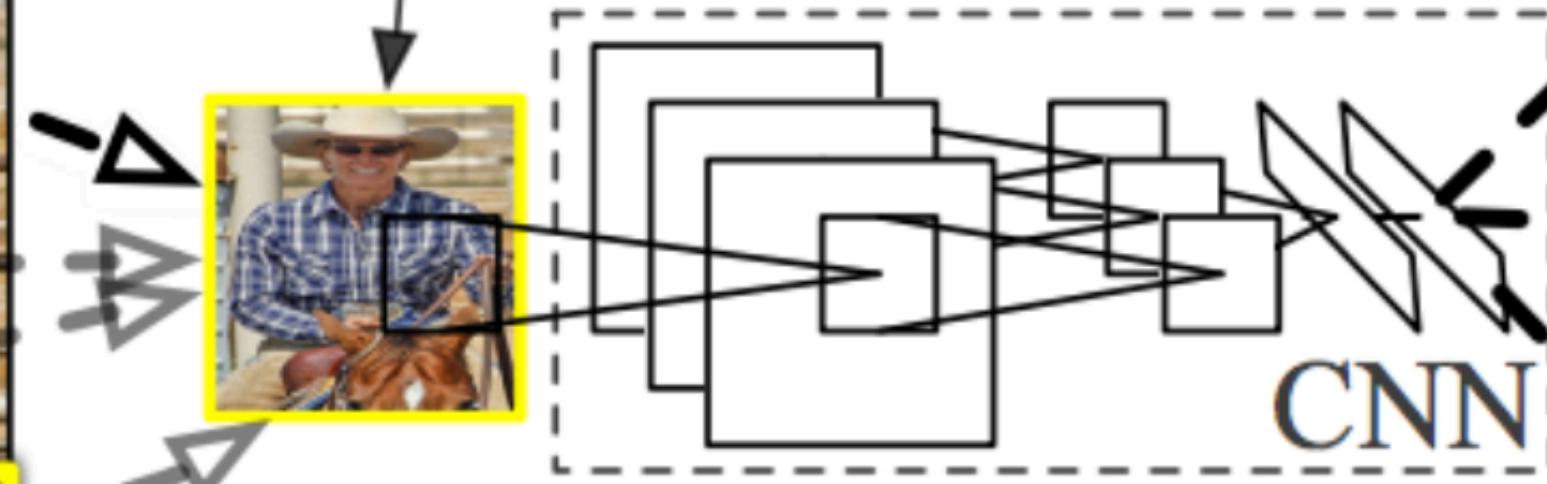


1. Input image



2. Extract region proposals (~2k)

warped region

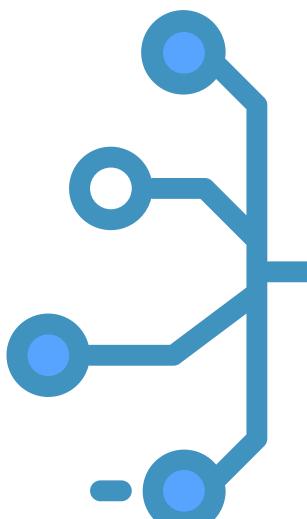


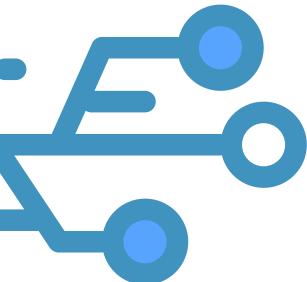
3. Compute CNN features

aeroplane? no.  
⋮  
person? yes.  
⋮  
tvmonitor? no.

4. Classify regions

透過 CNN 做圖像特徵萃取





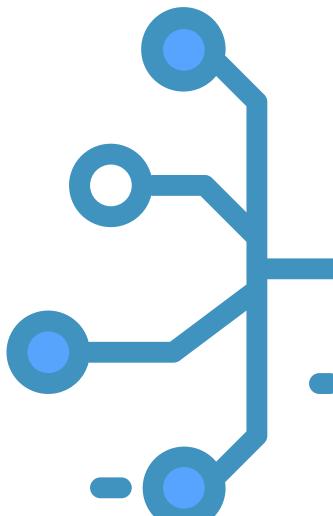
# R-CNN的問題



- 那 R-CNN 有什麼樣的問題呢？
- 簡而言之就是『速度慢』，其原因主要有以下兩點

經由 Selective Search 提出的 Region Proposal 都要獨自經過 CNN 做特徵提取，運算速度相當緩慢。

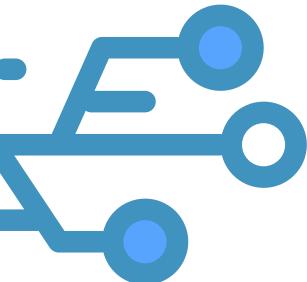
經過 CNN 得到 feature map，再用這些 feature 當成 SVM 的 input 當成訓練資料，因此並不是一個 end-to-end 模型 (SVM 的 LOSS 並不會改動到 feature map 的數值)





# Fast R-CNN





# Loss Function

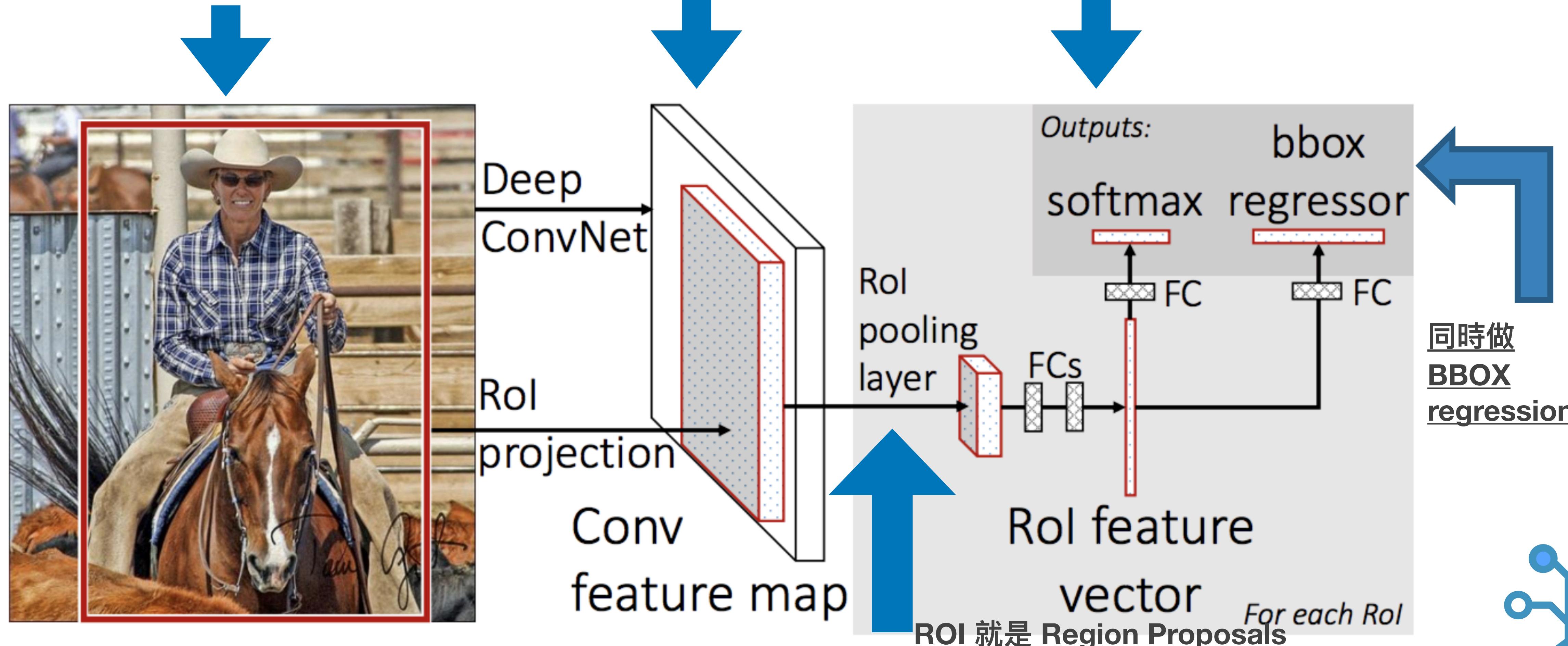


CUPOY

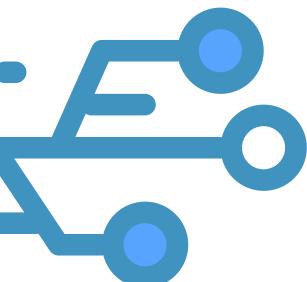
運用 Selective Search 提出  
Region Proposal

整張圖片送進去 CNN 提取特徵

Softmax 取代原本SVM



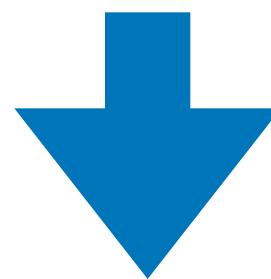
這裡先記得 ROI Pooling 是一種能將不同 Region proposals (候選框)，Pooling 成一樣大小的一個方法。



# Fast R-CNN v.s R-CNN



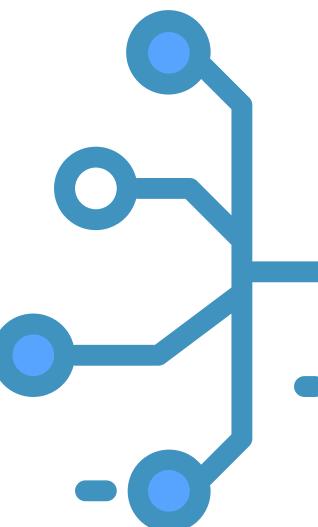
Winner!

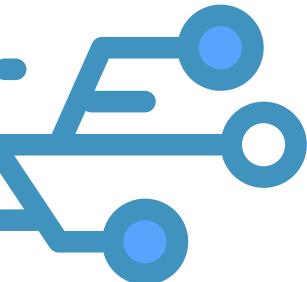


Faster!

FASTER!

	R-CNN	Fast R-CNN
Training Time:	84 hours	<b>9.5 hours</b>
(Speedup)	1x	<b>8.8x</b>
Test time per image	47 seconds	<b>0.32 seconds</b>
(Speedup)	1x	<b>146x</b>





# Fast R-CNN



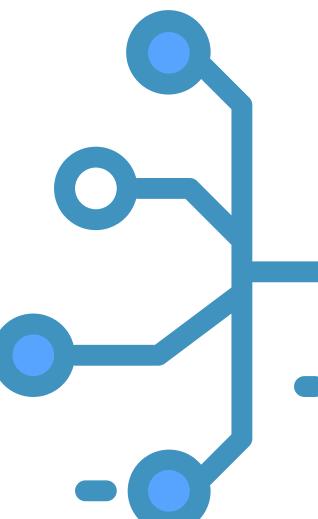
Fast R-CNN 解決了什麼問題？

原本每一個 Region Proposal 都要經過 CNN 提取特徵現在只需要將整張圖送入，速度上有很大的優勢。

運用 Softmax 取代 SVM，並同時加入了 BBOX regression 到模型中與分類一起收斂。

Fast R-CNN 有什麼問題？

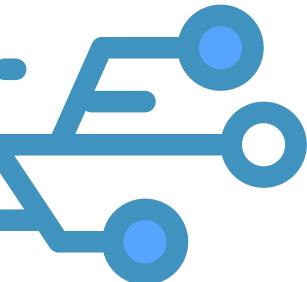
還是需要透過 Selective Search 提出 Region Proposal，沒辦法統整成 End-to-End model 訓練。





# Faster R-CNN

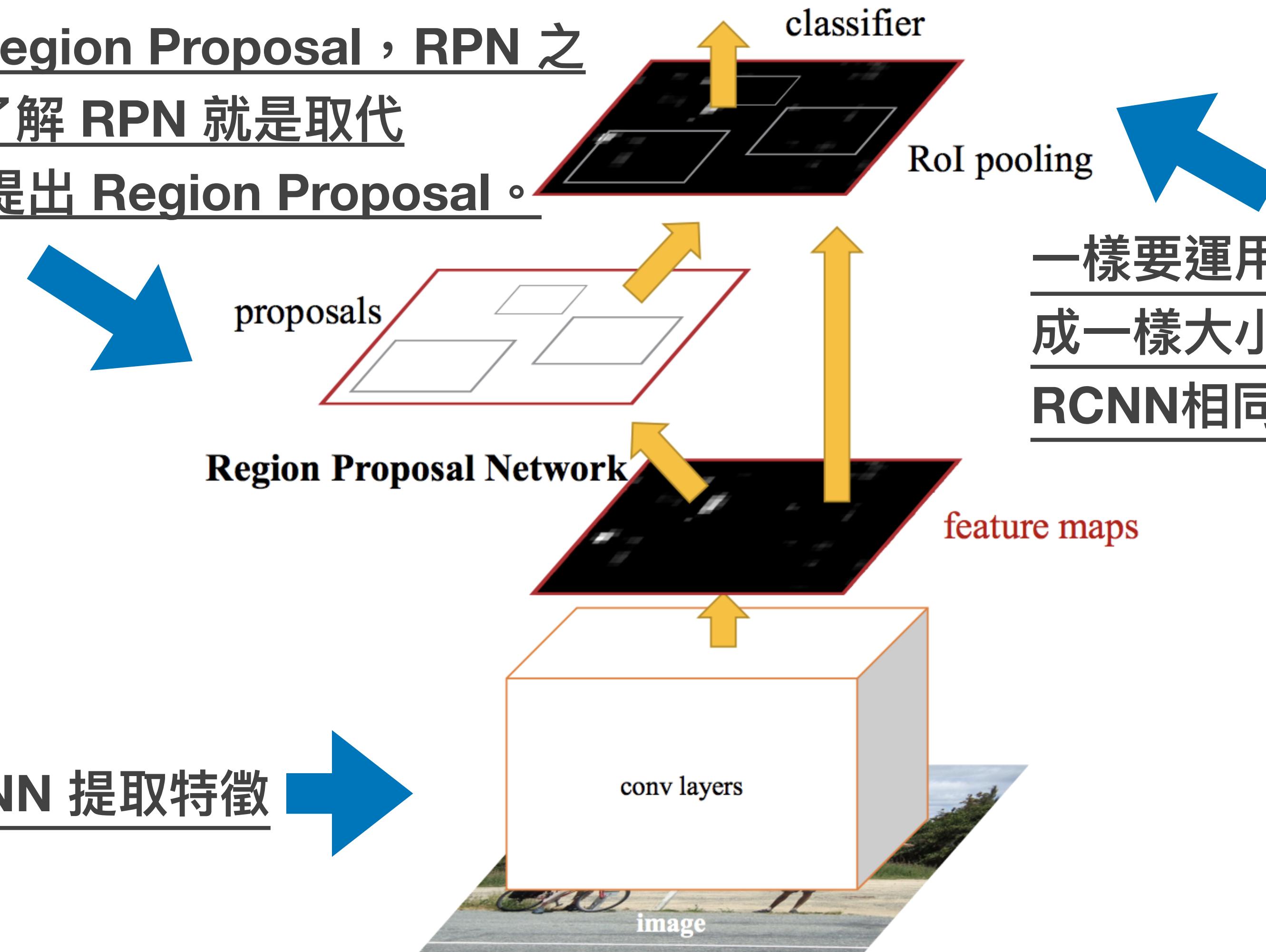




# Faster R-CNN

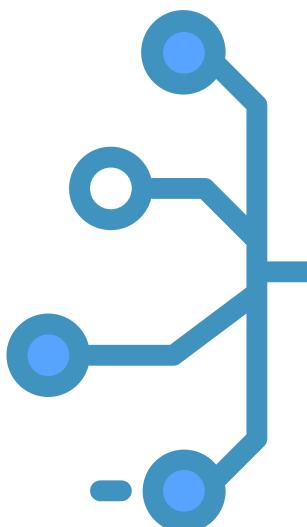


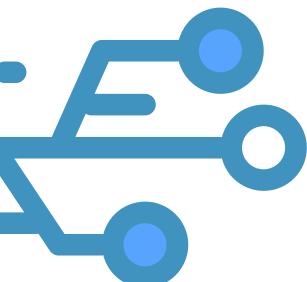
透過 RPN 結構提出 Region Proposal，RPN 之後會詳談目前可以先了解 RPN 就是取代 Selective Search 來提出 Region Proposal。



原圖直接通過 CNN 提取特徵

一樣要運用 ROI Pooling 將候選框變成一樣大小，後面結構則是跟 Fast-RCNN 相同



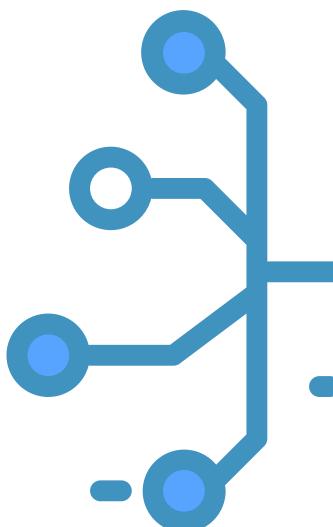
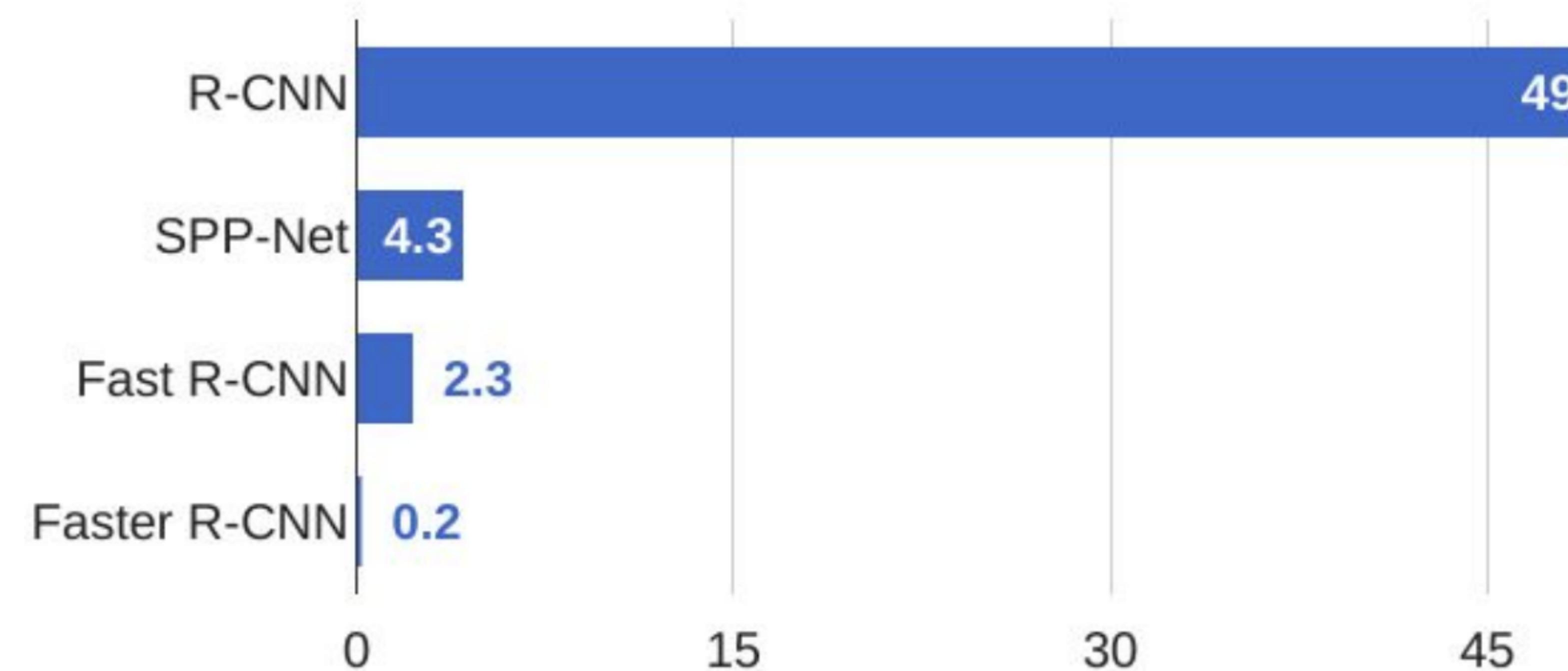


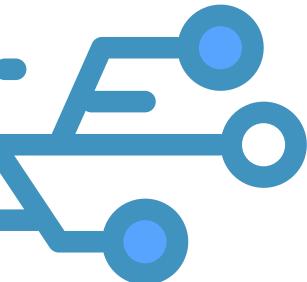
# Faster R-CNN



Faster R-CNN 在速度上完勝其他 Two-Stage 模型，對 SPP-NET 有興趣可看參考資料

## R-CNN Test-Time Speed





# Faster R-CNN

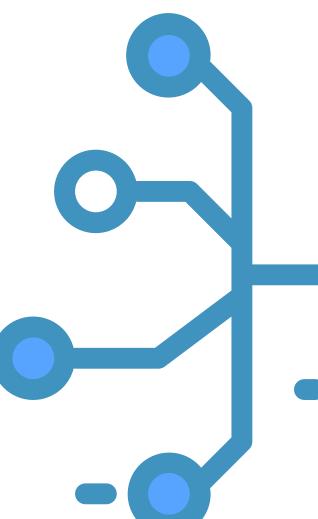


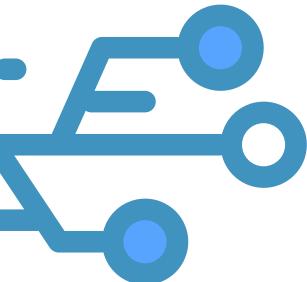
Faster R-CNN 解決了什麼問題？

不用 Selective Search，而改用 (Region Proposal Network) 來提取 proposals，達到真正的 End-to-End

Faster R-CNN 有什麼問題？

還是需要先提出 Region Proposal，整體而言速度受到一定的限制。





# One Stage

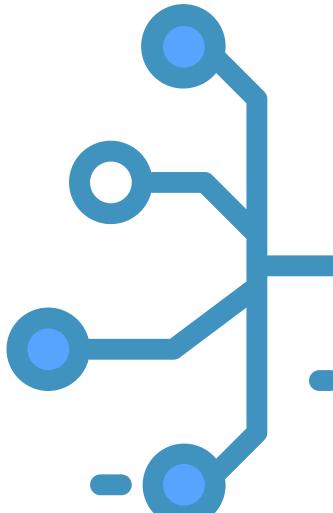


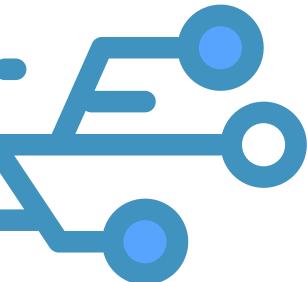
為了解決 Two-Stage 需要提出 proposals 導致速度變慢的問題，以追求速度為主要目的的 One-stage 模型開始出現。

**YOLO系列V1-V3**

**SSD**

**RetinaNet**

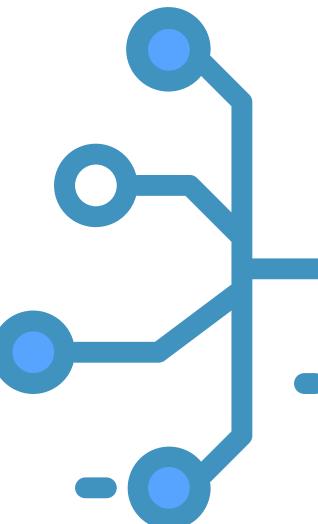




# One-Stage



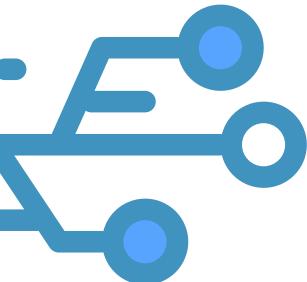
One Stage 核心觀念-不要先浪費時間提出 Region Proposal，而是以 Default Anchor Box 取代。





YOLO

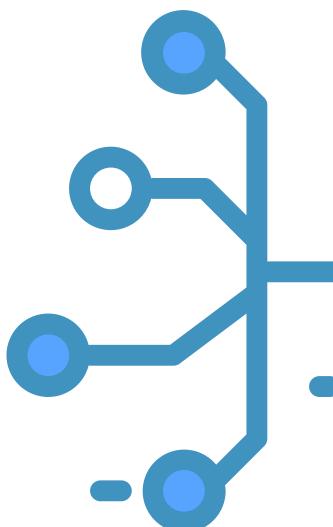
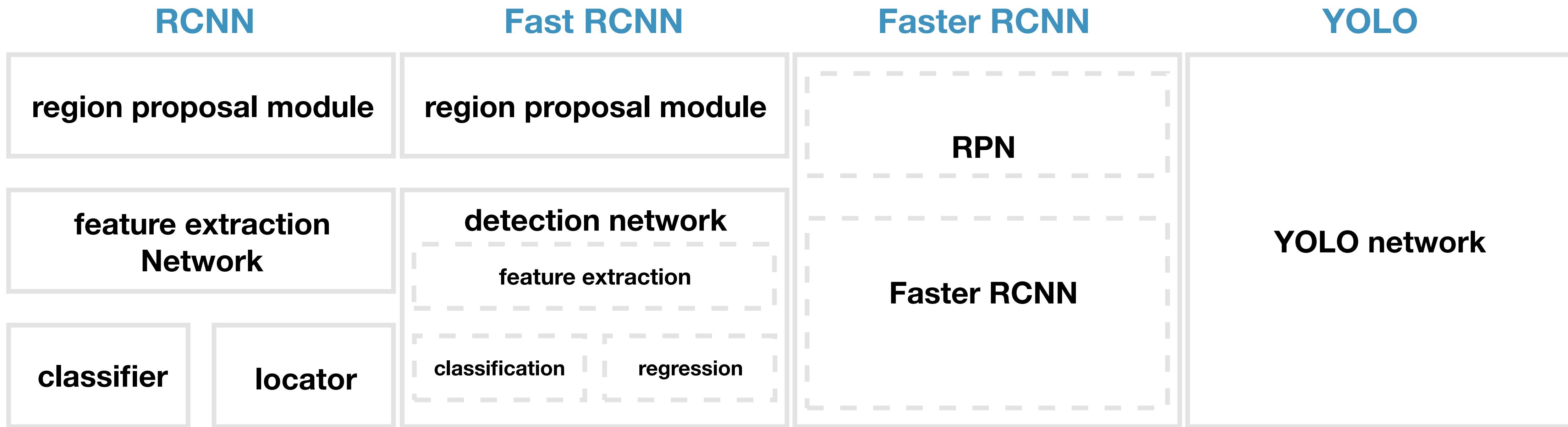


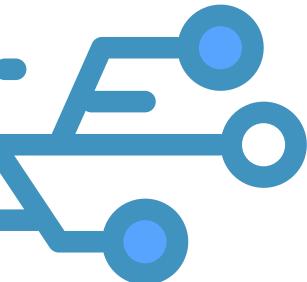


# Yolo V1- You Only Look Once



先來看一下 Yolo 與 RCNN 系列的比較

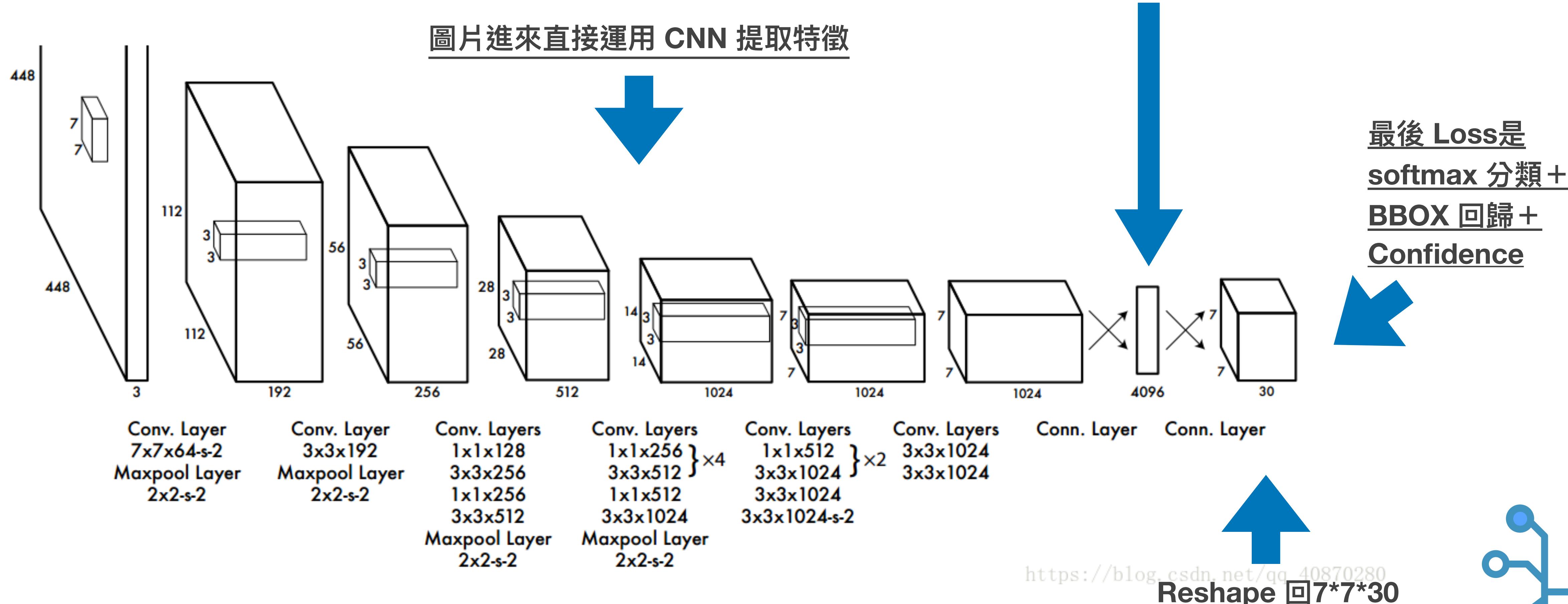


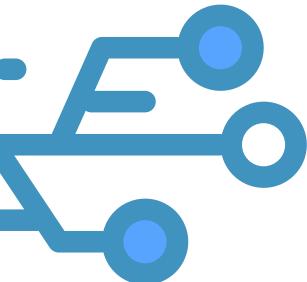


# Yolo V1- You Only Look Once



圖片進來直接運用 CNN 提取特徵



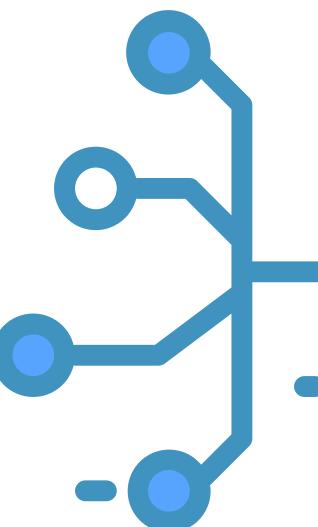


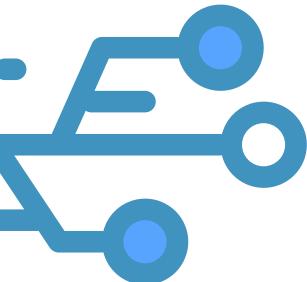
# Yolo V1- You Only Look Once



學員們可能會有一個疑問，沒有 Region Proposal 怎麼預測 BBOX？

YoloV1這裡提出一個新的想法，運用最後一層特徵圖中每一個像素中心點( Default Anchor Box )直接產生 Region Poposal，大家可以想像成直接以像素中心點來劃框框，而原文是每一個像素會有兩個預設 Bounding Box 。





# Yolo V1- You Only Look Once



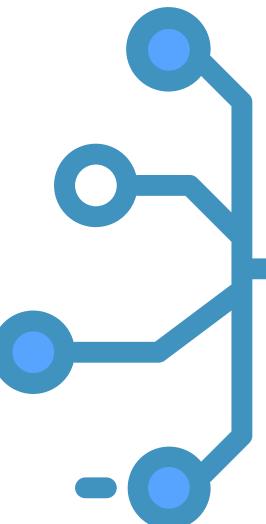
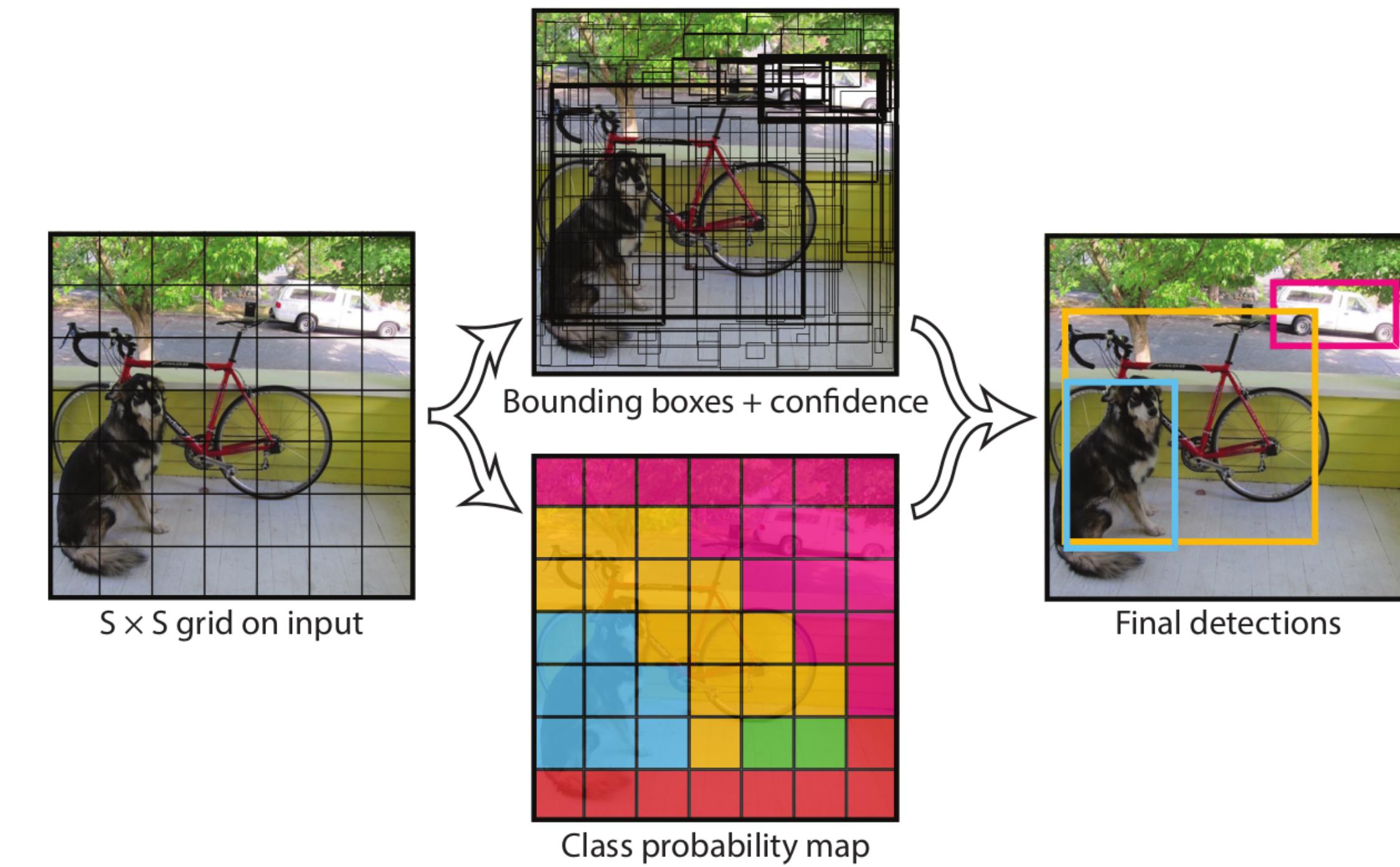
CUPOY

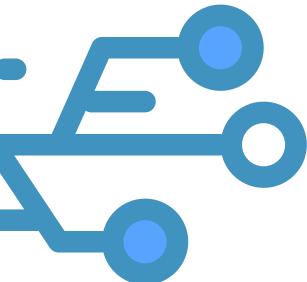
主要就是運用最後一層的 Feature Map 直接畫候選框，最後一層有  $7 \times 7 = 49$  個像素點，YoloV1 在每一像素點上設計兩個候選框，因此有 98 個候選框，每個框需要預測三個 Output，分別是：

屬於哪一類 (Classification)

校正BBOX (Regression)

Confidence (有物件的信心程度)





# Yolo V1- You Only Look Once



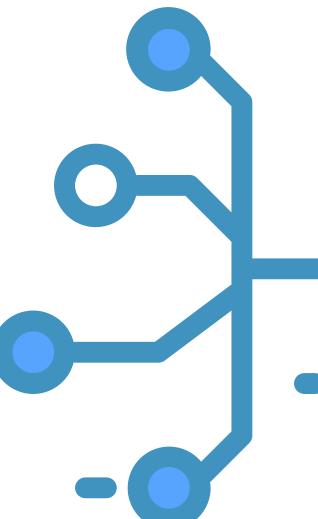
每個像素需要預測兩個 BBOX，每個 BBOX各要預測一個 confidence 以及偏移與縮放量，除此之外還要預測類別。

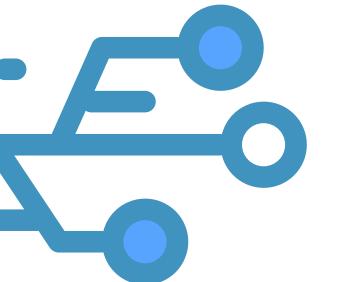
$$30 = 4*2 + 1*2 + 20$$

2個 BBOX 預測(x,y,w,h)的偏移與縮放

2個 BBOX 各有一個 confidence

20 個類別





# YoloV1



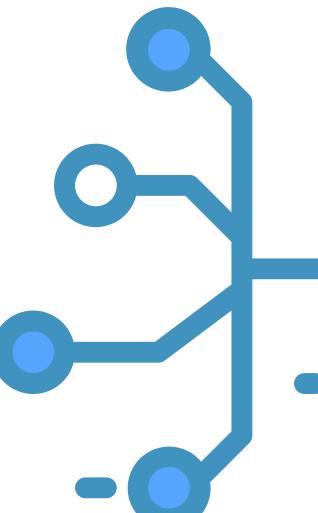
## YoloV1解決了什麼問題

不用花費時間提出 Region Proposal。

## Faster R-CNN 有什麼問題？

只透過最後一層預測，而由於最後一層的 Feature Map 尺寸只有  $7 \times 7$ ，已經喪失許多空間訊息，因此對小物件不敏感。

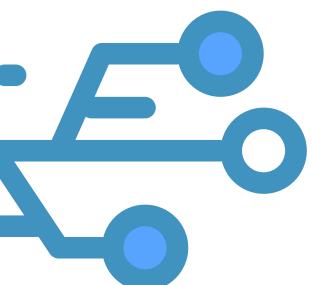
FC層造成空間訊息損失。





# Single Shot Multibox Dectector (SSD)



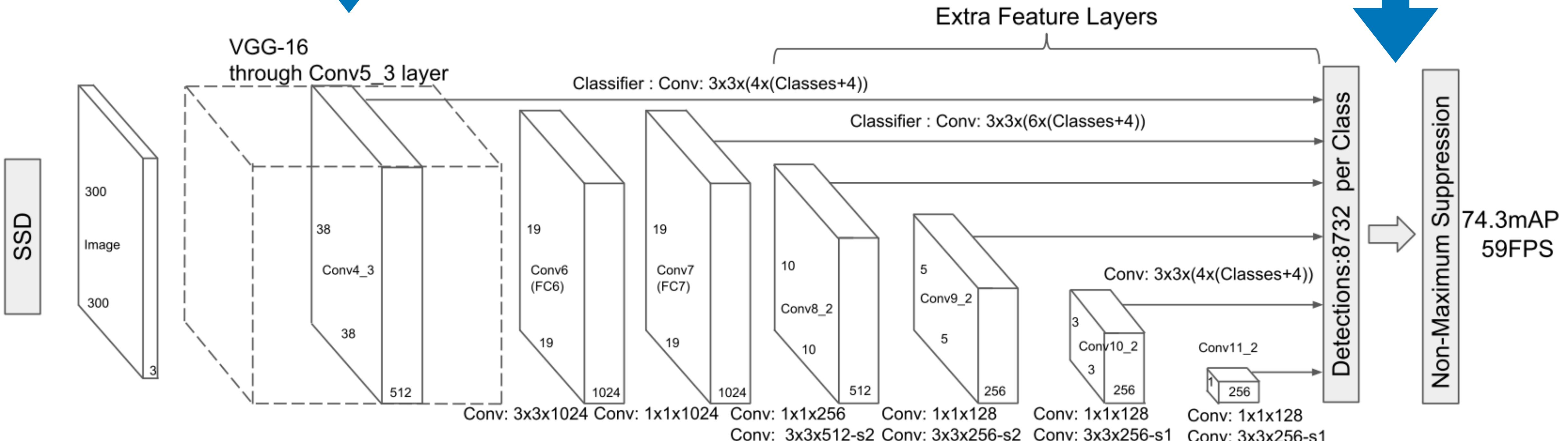
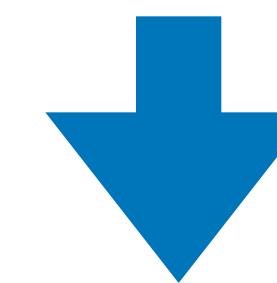


# SSD



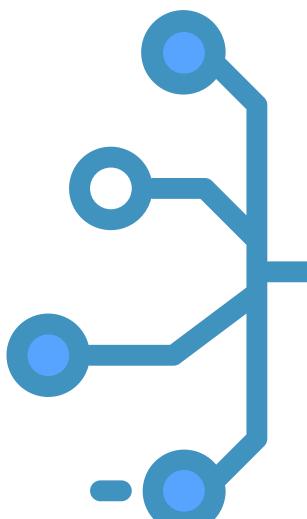
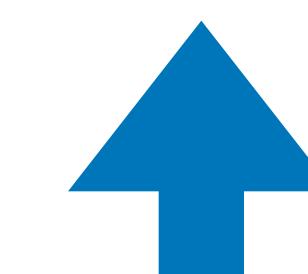
CUPOY

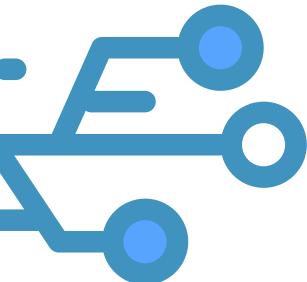
圖片進來直接運用 CNN 提取特徵



就是一個基本的 CNN 架構，不過運用多張 Feature Maps 來做預測

參考來源：[SSD 介紹](#)



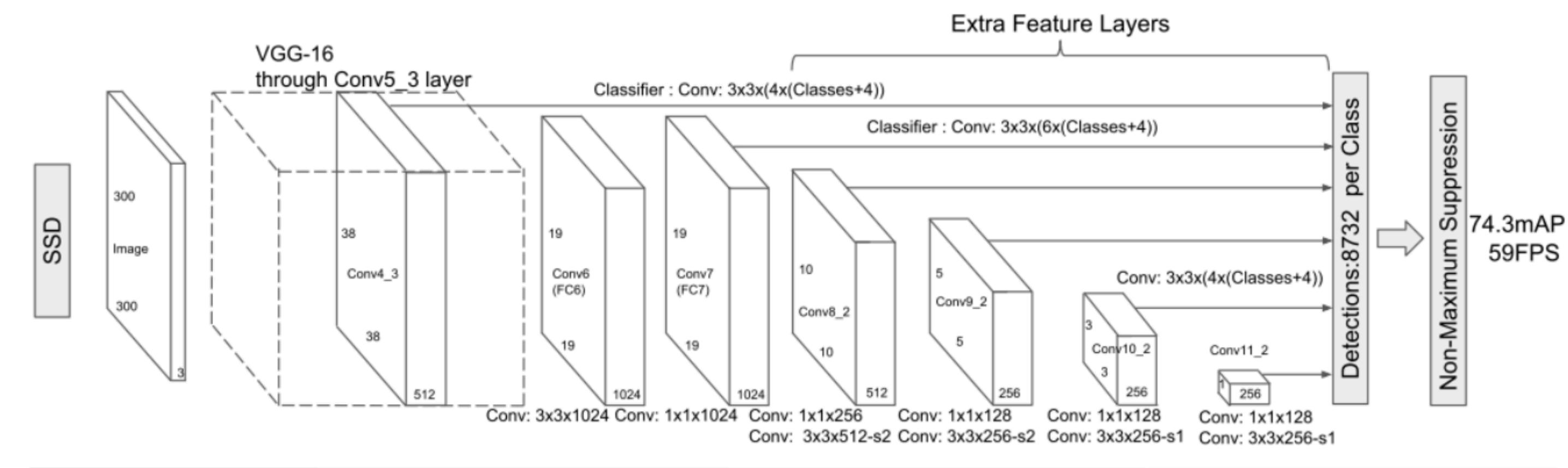


# SSD

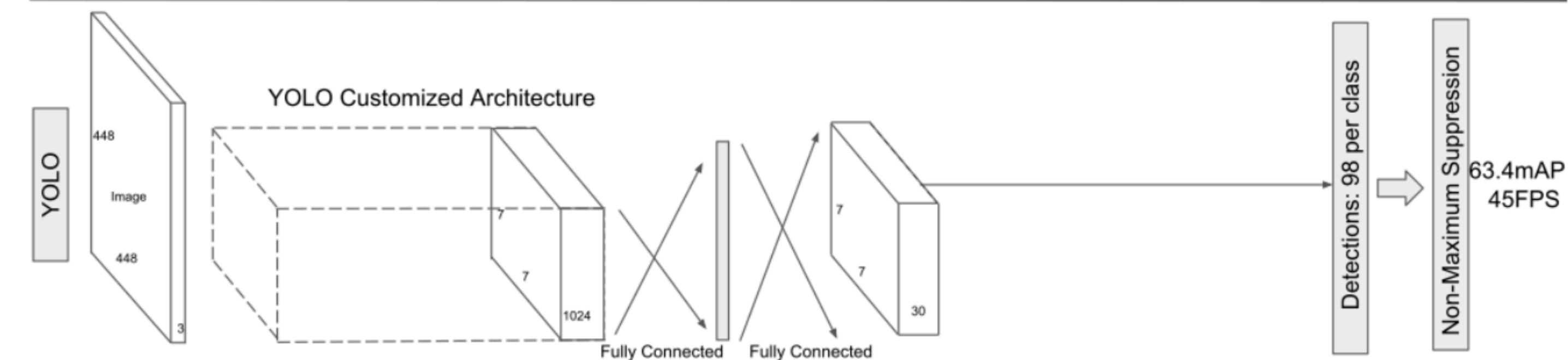


可以發現 SSD 運用多尺度 Feature Maps、不使用 FC 層，並且每一個像素上放置更多 default anchor BOX，全部一共有 8732 個( 相較於 YoloV1 只有 98個 )。

# SSD



# YOLO





SSD



CUPOY

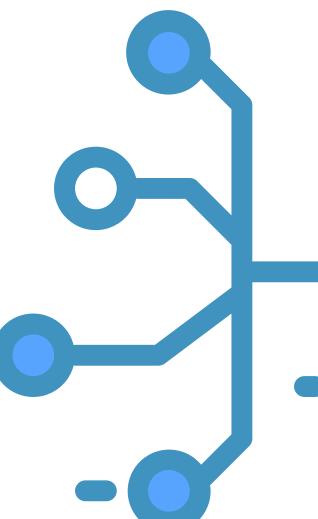
## SSD 解決了什麼問題

多尺度預測，對小物件更敏感。

## SSD 有什麼問題？

正負樣本不平均，太多背景，使用 Online hard example mining (OHEM) 使易分類樣品消失 (OHEM可看參考資料)。

運用淺層 Feature Map 偵測小物件，但特徵訊息不夠豐富對小物件敏感度仍比不上 Faster R-CNN。

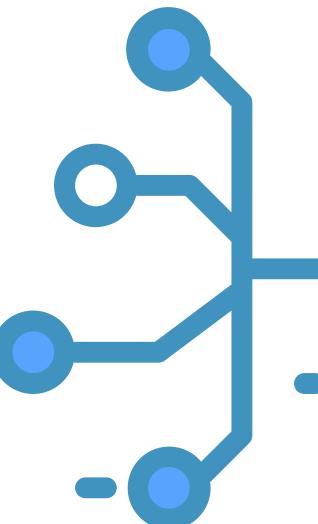




## Yolo V2、V3



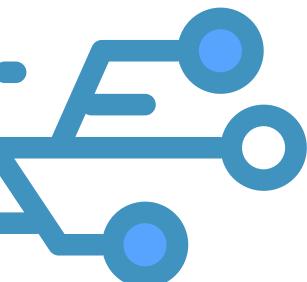
YoloV2、V3發表於 SSD 之後，也針對了 V1 的問題做改善，之後課程有更詳盡的介紹。





# Retina Net

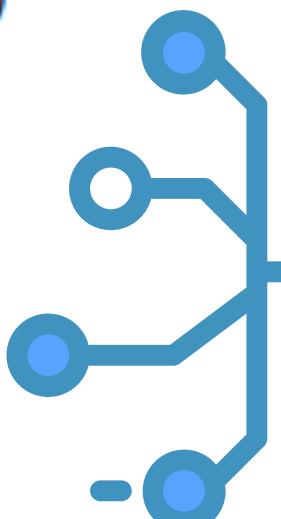
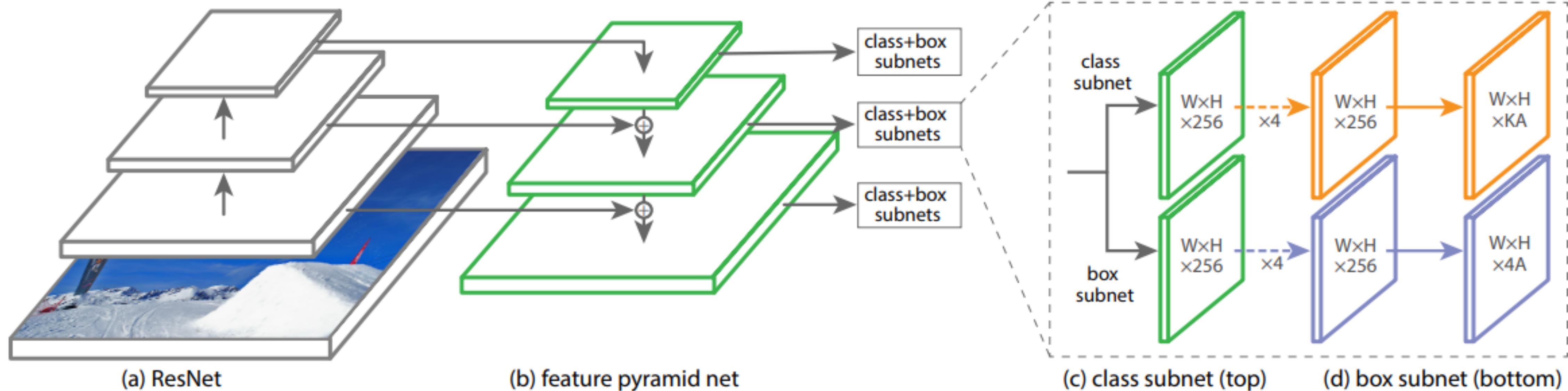




# Retina Net

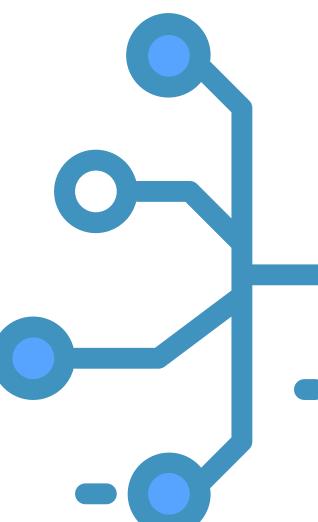


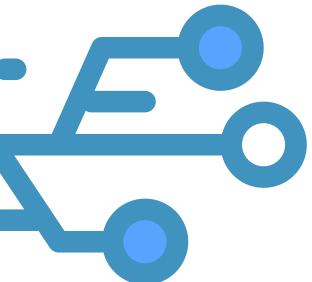
RetinaNet 的主架構與 SSD 很像，重點在於加入了 Feature Pyramid Network (FPN) 的結構，把淺層的 Feature Map 與深層的 Feature Map 疊加後做預測，確保淺層語義訊息也能夠豐富。



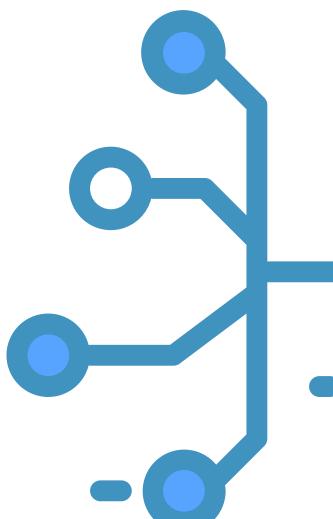
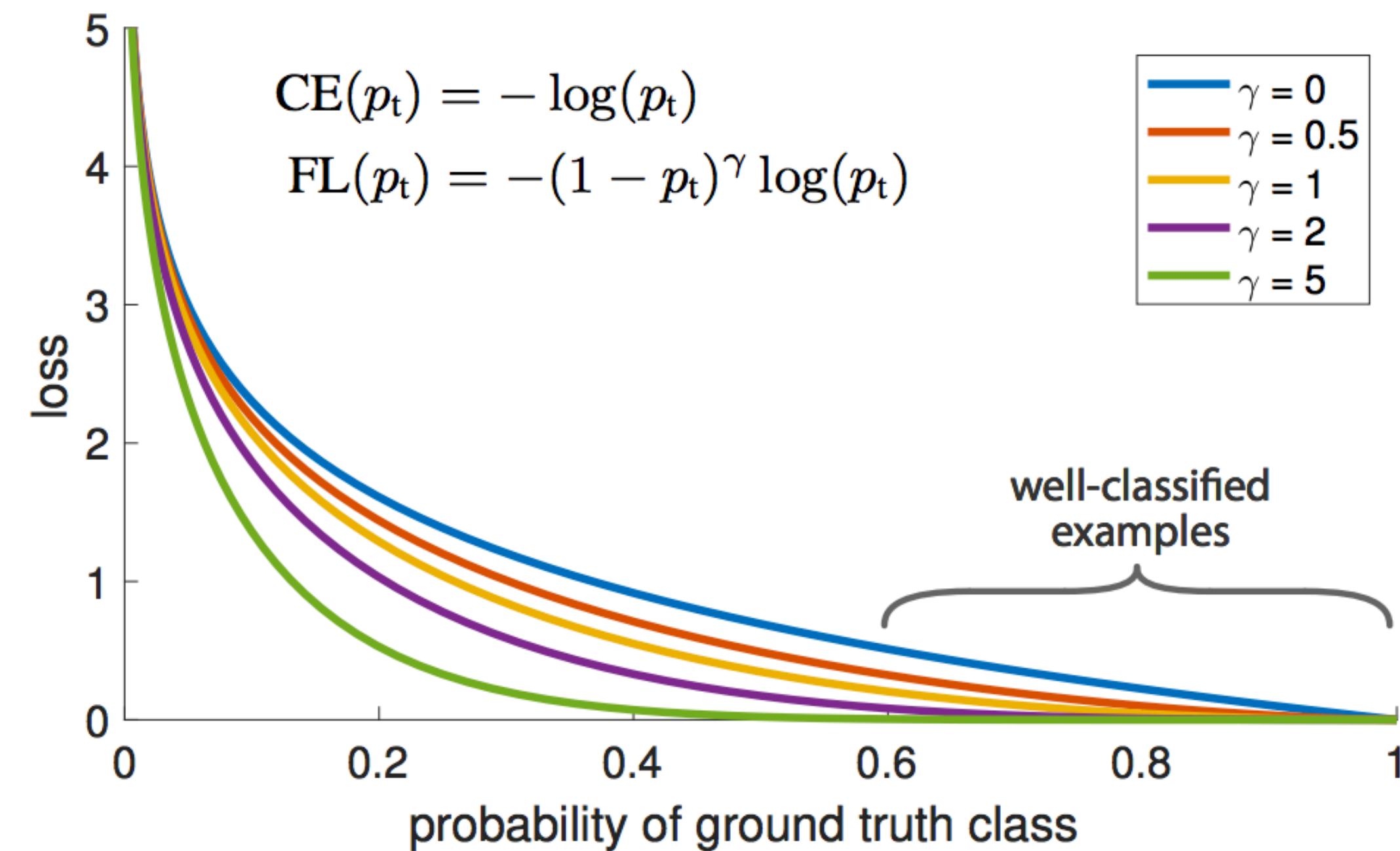


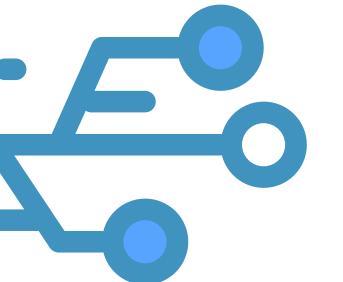
- 另一個重點就是設計 Focal Loss，用來解決背景預測框太多的狀況，並同時處理類別樣本不均的狀況。
- 在 RetinaNet 中，Default Anchor Box 有 10 萬個左右，其中真正有涵蓋物件的候選框少之又少(可能只有幾百個)，如果不經過任何處理，參數的更新主要都被負樣品影響 ( SSD中用OHEM解決 )。
- 除了正負樣品( 候選框 )不均外，類別不均也是常見的問題，如包含狗的照片很多，包含貓的照片很少。





Focal Loss 為 Cross Entropy 的變形，其中  $P_t$  是我們預測的機率值，將每個  $p_t$  賦予一個  $(1-p_t)^\gamma$  的權重，其中  $\gamma$  為超參數，當預測機率值越高( 易分類樣品 )，其所獲得的權重會越低，藉此抑制易分類的背景框，同時藉由 Alpha 賦予不同類別不同的權重( Alpha常見為一個List，ex. [1.0, 0.1] )。



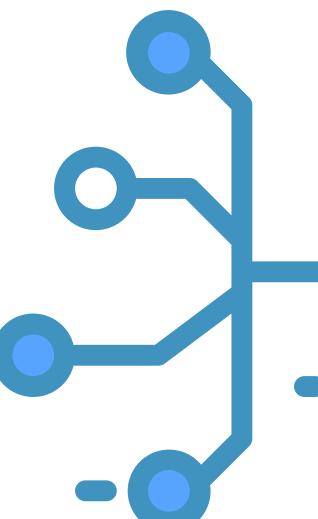


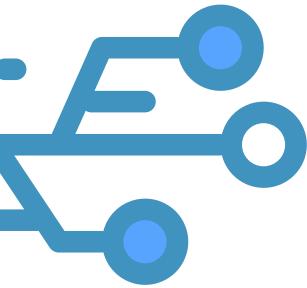
RetinaNet 解決了什麼問題？

**多尺度預測，對小物件更敏感。**

RetinaNet 有什麼問題？

**Focal Loss 中 Hyperparameters 的調整對結果影響大。**





# 推薦延伸閱讀



知乎

首发于  
晓雷机器学习笔记

关注专栏

写文章

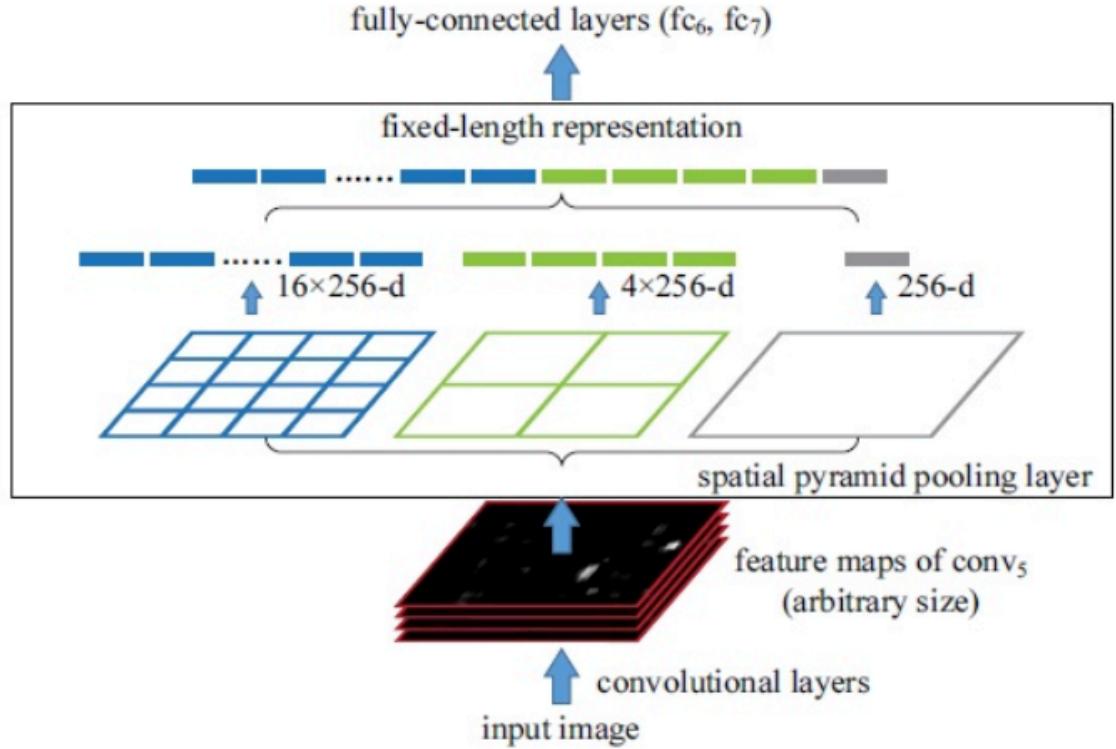


Figure 3: A network structure with a **spatial pyramid pooling layer**. Here 256 is the filter number of the conv<sub>5</sub> layer, and conv<sub>5</sub> is the last convolutional layer.

## SPPNet-引入空间金字塔池化改进RCNN

晓雷  
end2end for all sensors

130 人赞同了该文章

继续总结一下RCNN系列。上篇RCNN- 将CNN引入目标检测的开山之作 介绍了CNN用于目标检测的基本思想和流程。后续出现了SPPnet,Fast-RCNN ,Faster-RCNN等一些列改进。最终实现了端对端学习，同时带来速度与精度的提升。

## Object Detection SppNet

連結

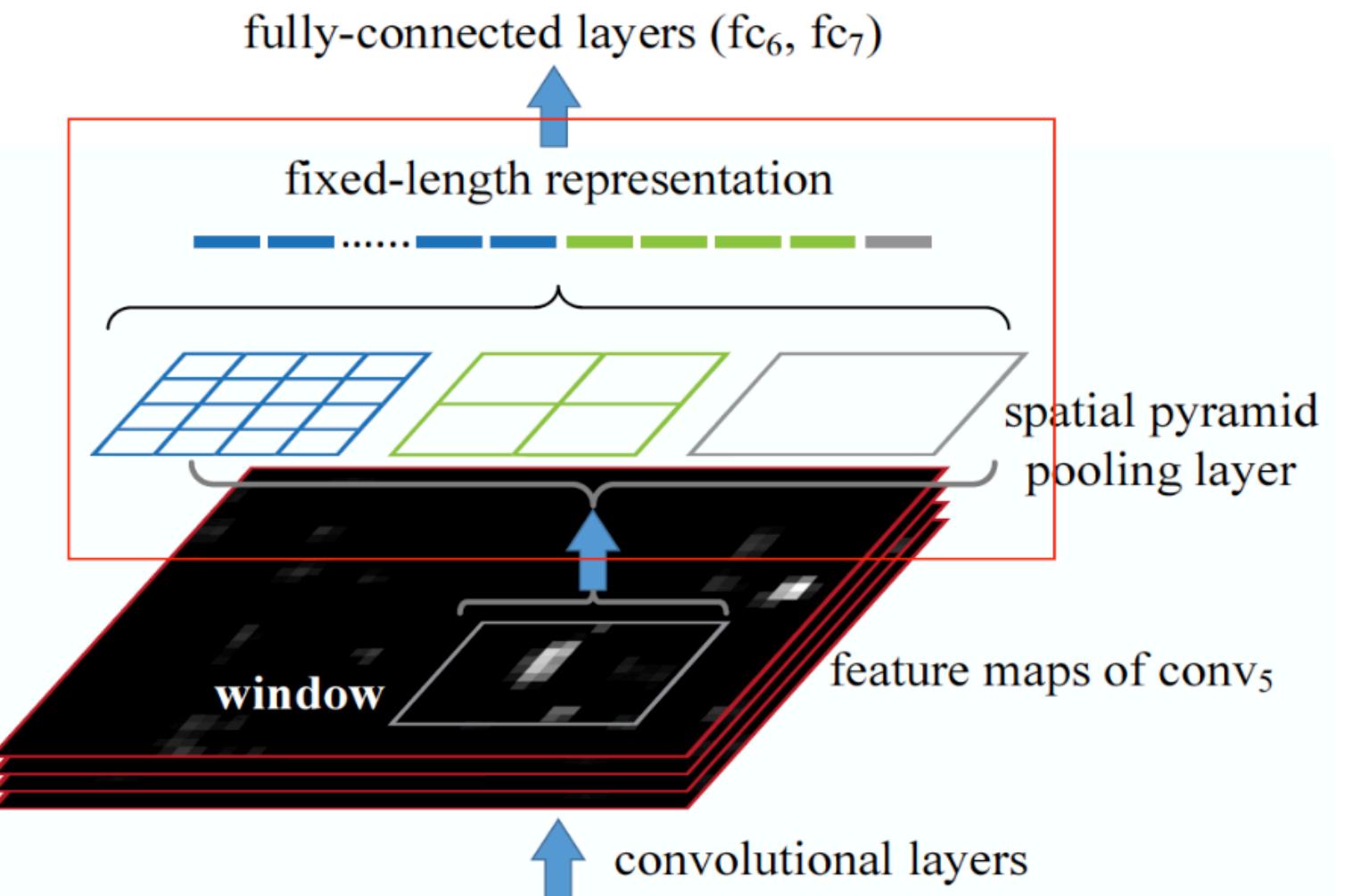
## 深度学习: 从 RoIPooling 到 RoIAvg

2017-12-16 21:25:02 JNingWei 阅读数 14588 更多

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。  
本文链接：<https://blog.csdn.net/JNingWei/article/details/78822159>

### SPP Layer

对RoI进行pooling的操作最早由SPPNet中的SPP layer提出：



## Object Detection ROIPooling

連結

## Focal Loss(RetinaNet) 與 OHEM

其他 · 發表 2018-11-30



和平公園

和平公園第一排 2-3房 | 近捷運莒光站 8952-1177

### Focal Loss for Dense Object Detection-RetinaNet

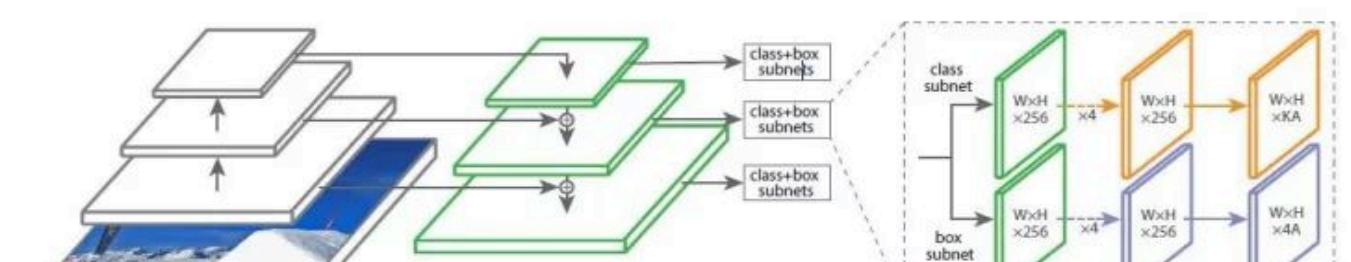
YOLO和SSD可以算one-stage演算法裡的佼佼者，加上R-CNN系列演算法，這幾種演算法可以說是目標檢測領域非常經典的演算法了。這幾種演算法在提出之後經過數次改進，都得到了很高的精確度，但是one-stage的演算法總是稍遜two-stage演算法一籌，於是就有了Focal Loss來找場子。

在Focal Loss這篇論文中，作者認為one-stage精確度不如two-stage是因為下面的原因：

① 正負樣本比例極度不平衡。由於one-stage detector沒有專門生成候選框的子網路，無法將候選框的數量減小到一個比較小的數量級（主流方法可以將候選框的數目減小到數千），導致了絕大多數候選框都是背景類，大大分散了放在非背景類上的精力；

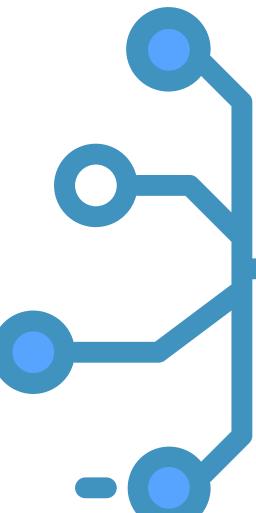
② 梯度被簡單負樣本主導。我們將背景類稱為負樣本。儘管單個負樣本造成的loss很小，但是由於它們的數量極其巨大，對loss的總體貢獻還是佔優的，而真正應該主導loss的正樣本由於數量較少，無法真正發揮作用。這樣就導致收斂不到一個好的結果。

既然負樣本數量眾多，one-stage detector又不能減小負樣本的數量，那麼很自然的，作者就想到減小負樣本所佔的權重，使正樣本佔據更多的權重，這樣就會使訓練集中在真正有意義的樣本上去，這也就是Focal Loss這個題目的由來。



## Object Detection OHEM與Focal Loss

連結



# 解題時間 Let's Crack It



請跳出 PDF 至官網 Sample Code & 作業開始解題