Day 27

深度學習與電腦視覺

學習馬拉松

cupoy 陪跑專家：楊哲寧

深度學習理論與實作

Object Detection-BBOX Regression

重要知識點

CUPOY

- 如何設計 Bounding Box Loss Function
- 了解 Bounding Box Regression 的原理

BBOX Regression

CUPOY

有 **Anchors**_的 **BBOX Regression** (FPN、One Stage object detection)

BBOX Regression

這一類的 BBOX regression 主要是 anchors 與標註好的 BBox 在做運算。

標註框(Ground Truth)

Anchors

Bounding Box Loss

再來看一下公式是如何設計的

g(x,y,w,h)代表 Ground Truth

P(x,y,w,h)代表 anchors

$$t_x^j = (g_x^j - p_x^i)/p_w^i$$

$$t_y^j = (g_y^j - p_y^i)/p_h^i$$

$$t_w^j = \log(g_w^j/p_w^i)$$

$$t_h^j = \log(g_h^j/p_h^i)$$

Source

參考來源： Fast Detetcion Models

Bounding Box Loss

上一步算出來的 **t** (x,y,w,h) 再與 **預測值**做回歸。

上一步算出的值

$$\mathcal{L}_{\text{loc}} = \sum_{i,j} \sum_{m \in \{x,y,w,h\}} \mathbb{1}_{ij}^{\text{match}} L_1^{\text{smooth}}(d_m^i - t_m^j)^2$$

我們並不是直接預測 BBOX 的值，而是其偏移量與縮放比例。

參考來源： [Fast Detetcion Models]

Bounding Box Loss

CUPOY

還有一點比較特別的是，這裡既不是用 L1 也不是用 L2 Loss 而是採用『Smooth L1 Loss』

$$L_1^{\text{smooth}}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

參考來源： Fast Detetcion Models

Bounding Box Loss

CUPOY

接下來我們就來了解每一步的原理

Bounding Box Loss

首先計算 **Anchor** 與 **Ground Truth** 的偏移量，然而為什麼不直接相減就好還要除以 **Anchor** 的**寬**、**高** 呢？

$$t_x^j = (g_x^j - p_x^i)/\boxed{p_w^i}$$

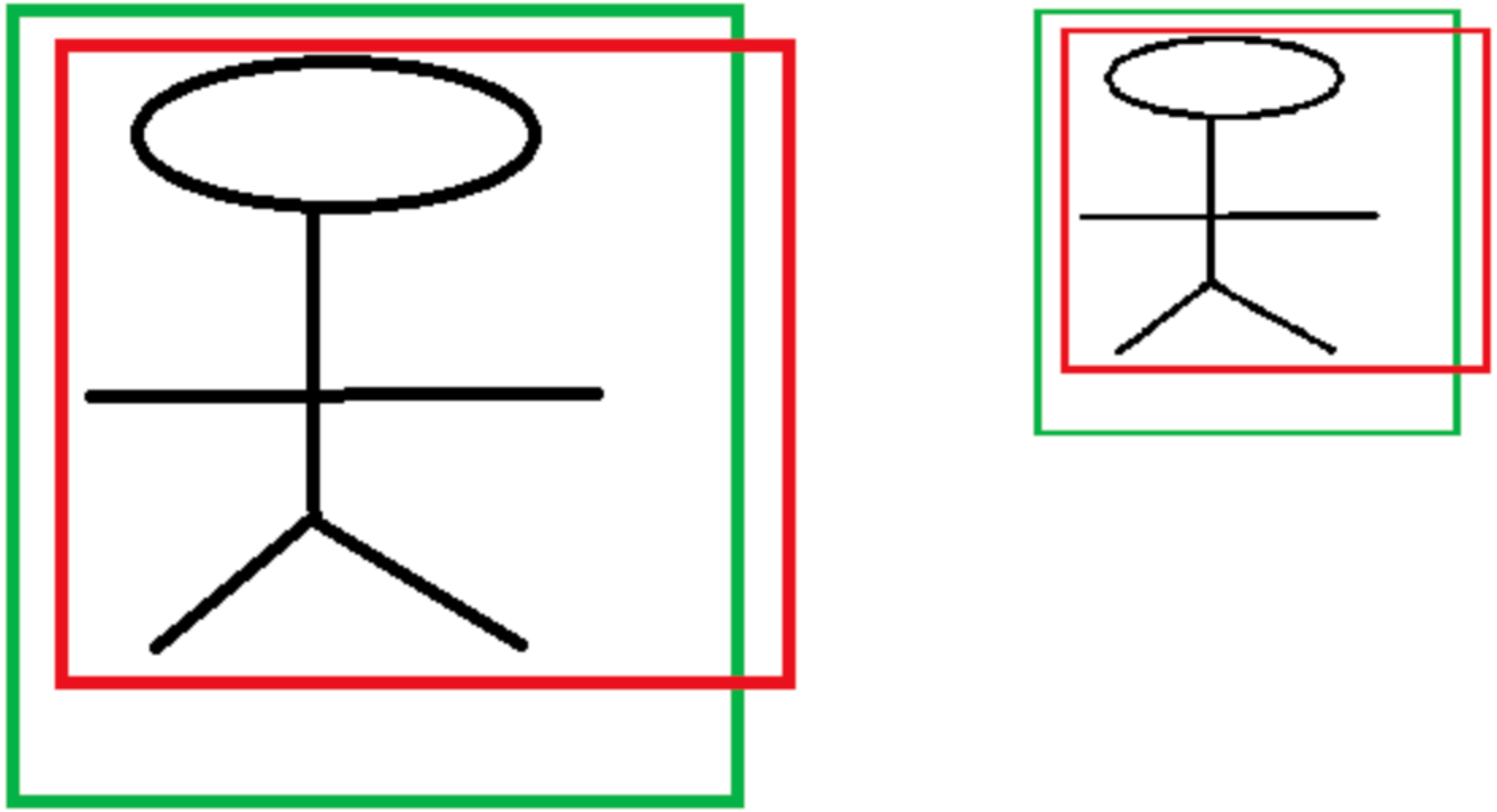
$$t_y^j = (g_y^j - p_y^i)/\boxed{p_h^i}$$

$$t_w^j = \log(g_w^j/p_w^i)$$

$$t_h^j = \log(g_h^j/p_h^i)$$

Bounding Box Loss

- 主要是因為要維持**尺度的不變性**，以下方兩圖為例，我們希望**框的偏移量不會因為物件的大小而受影響**，因此要是我們只是純粹相減兩個框的距離的話，很明顯物件越大其得到的值也會越大。



參考來源： 邊框回歸詳解

Bounding Box Loss

再來看到寬和高的部分，為什麼不能直接相除算比例而還要加入一個 **Log** 呢？

$$t_x^j = (g_x^j - p_x^i)/p_w^i$$

$$t_y^j = (g_y^j - p_y^i)/p_h^i$$

$$t_w^j = \boxed{\log}(g_w^j/p_w^i)$$

$$t_h^j = \boxed{\log}(g_h^j/p_h^i)$$

參考來源： [Fast Detetcion Models]

Bounding Box Loss

主要是因為我們希望dw、dh
在換算回來時能恆為正數

反推

Dw是我們預測的tw

$$\mathrm{Exp}(d_w) = G\hat{}_w / p_w$$

$$t_w^j = \log(g_w^j / p_w^i)$$

我們推算的
Ground Truth

$$G\hat{}_w = \underline{\mathrm{Exp}(d_w)} * p_w$$

由於 log 返回來算就是 Exponential，
我們確保這項恆為正數

參考來源： Fast Detetcion Models

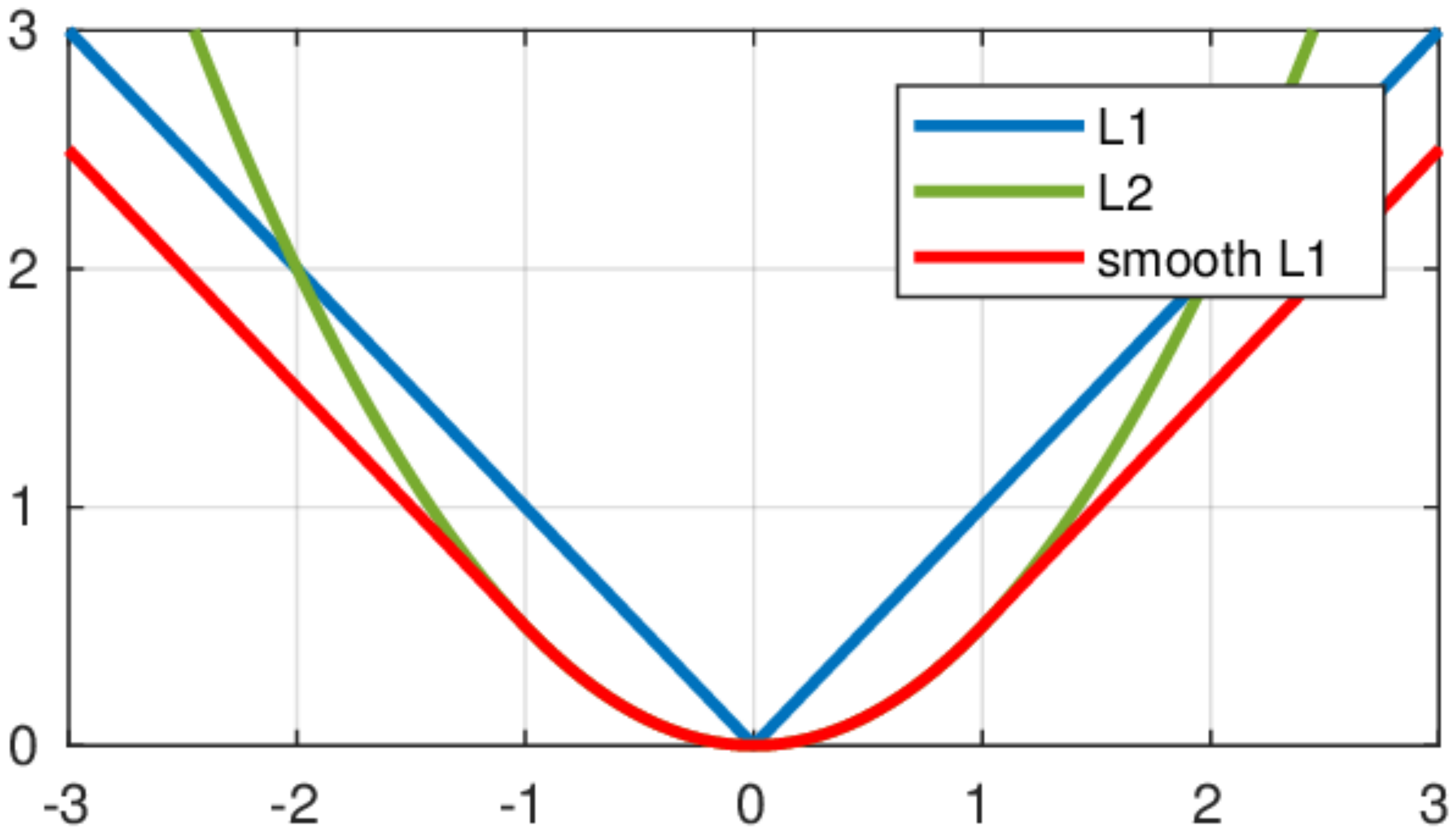
Bounding Box Loss

CUPOY

那為什麼要用 L1 smooth而不是 L2 或 L1?

Bounding Box Loss

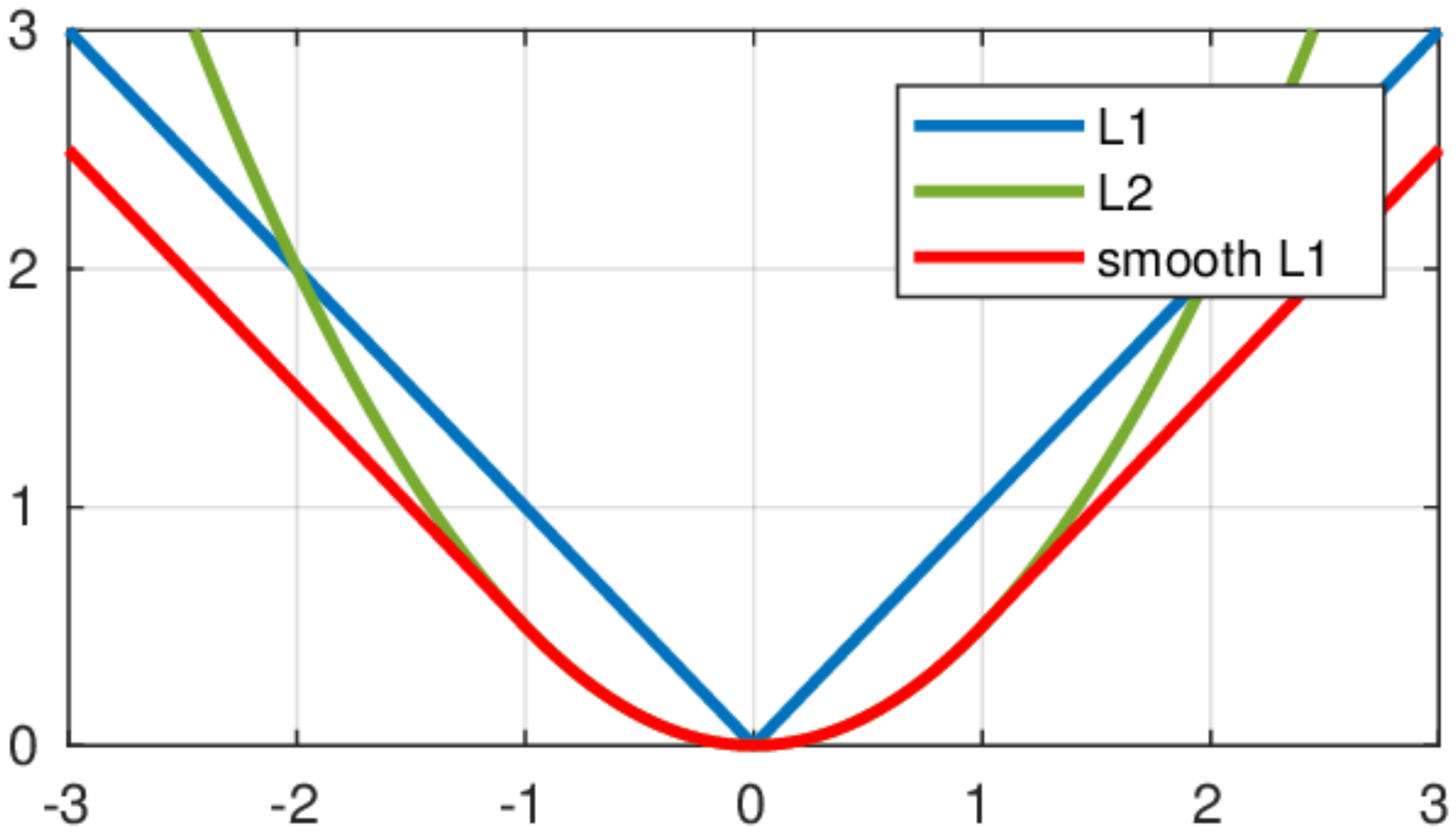
- L2 Loss 缺點是對離群值太敏感，L1 則是收斂太慢



參考來源： ResearchGate

Bounding Box Loss

Smooth L1 則是結合兩者優點



參考來源： ResearchGate

Bounding Box Loss

- 當 |y-y^| 不大時 (<1)，用L2 的方式
- 當 ly-y^l 相對大時 (等於是離群值)，採用 L1 距離的方式

$$L_1^{\text{smooth}}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

參考來源： [Fast Detetcion Models]

Bounding Box Loss

CUPOY

當 Anchors 變成 Proposal 時也是一樣的

Bounding Box Loss

CUPOY

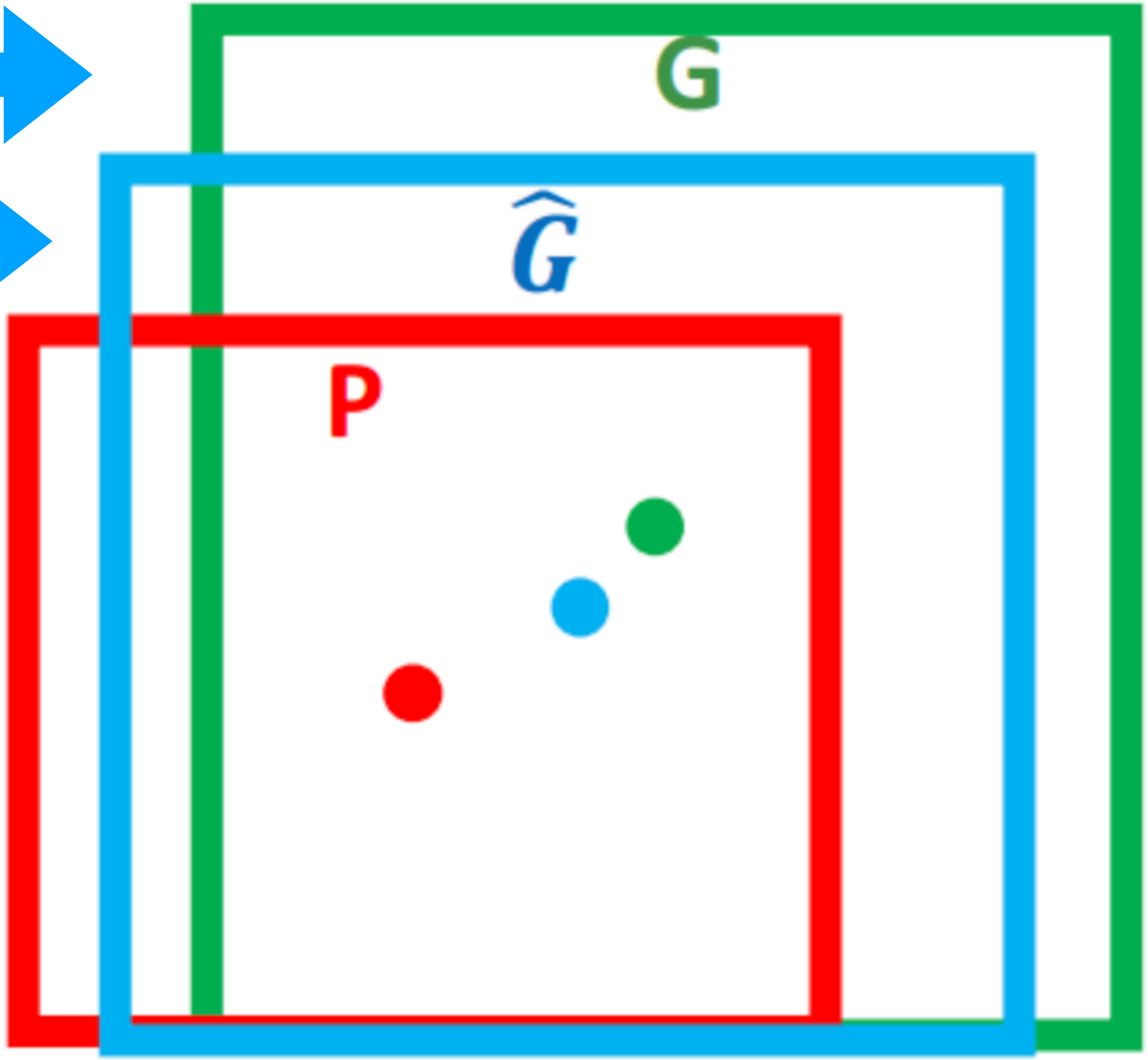
- 上一個介紹的是透過 **Anchor** 與 **Ground Truth Box** 回歸。
- 而在有 **Region Proposal** 的狀況下，其實也是一樣的。

Bounding Box Loss

Ground Truth

Proposal 轉換

Proposal



參考來源： 邊框回歸詳解

Bounding Box Loss

CUPOY

- 所以我們其實就是要學四個值，想辦法將 **Proposal** 轉換為 **G^**，並讓 G^ 越接近真實框越好。

$$f(Px,Py,Pw,Ph)=(Gx\hat{},Gy\hat{},Gw\hat{},Gh\hat{})$$

- 這四個值就是 **dx(p)**、 **dy(p)**、 **dw(p)** 、**dh(p)**

Bounding Box Loss

CUPOY

Proposal轉換 | **Proposal與標註框真實差值**

<u>di(p)要與Ti逼近</u>

$$\hat{G}x = Pw\,dx(P) + Px \quad\longleftrightarrow\quad tx = (Gx - Px)/Pw$$

$$\hat{G}y = Ph\,dy(P) + Py \quad\longleftrightarrow\quad ty = (Gy - Py)/Ph$$

$$\hat{G}w = Pw\exp(dw(P)) \quad\longleftrightarrow\quad tw = \log(Gw/Pw)$$

$$\hat{G}h = Ph\exp(dh(P)) \quad\longleftrightarrow\quad th = \log(Gh/Ph)$$

Bounding Box Loss

Loss是這樣寫的，由於這是較為早期的方式，沒有加入Smooth L1 Loss。

di(p)

$$Loss = \sum_{i}^{N}(t_*^i - \boxed{\hat{w}_*^T \phi_5(P^i)})^2$$

參考來源： Fast Detetcion Models

推薦延伸閱讀

Gradient Vanishing Problem — 以 ReLU / Maxout 取代 Sigmoid actvation function

tags: 李宏毅 `Maching Learning`

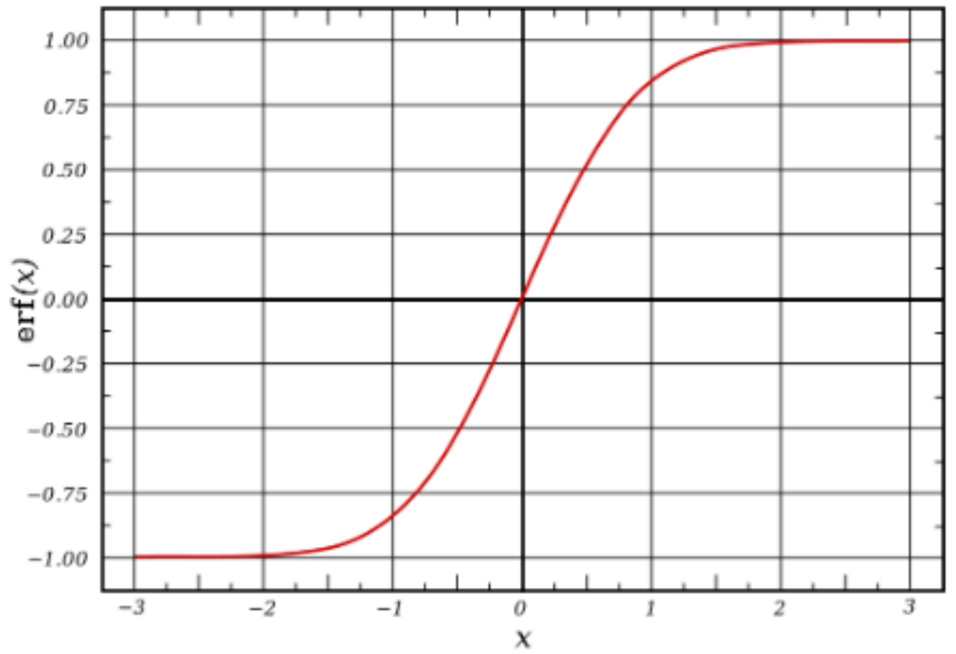
> - 本文內容參考自Hung-yi Lee , Machine Learning(2017) 課程內容 : Tips for Training DNN
> - 本文圖片部分來自於課程講義內容

梯度消失 Gradient Vanish

「類似」於 Sigmoid function 的激勵函數，普遍帶有梯度消失 (Gradient Vanish) 的隱憂，那究竟什麼是梯度消失?

$$Sigmoid\ function = \theta(s) = \frac{1}{1+e^{-s}}$$

此函數圖形為



(圖片取自： Wikipedia — Sigmoid function)

從圖上可知，其圖形切線斜率 (導數) 不會超過0.25，如此情況當我們在進行 Gradient Descent 的過程中，隨著迭代次數的增加，參數的更新會越來越緩慢 而整個 train 不起來。[1]

為何選用 Smooth L1

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