Formula

2024年12月23日 15:34

Convolutional layer

Input $C_{in} \times H \times W$

hyp: Kernel size $K_H \times K_W$ 通常 $K_H = K_W$

#filters C_{out} 32, 64, 128, 256 2^n

Padding P 图像边界额外加 P 圈 0 → preserving more spatial info → imp performance

$P = \frac{K-1}{2}$

- $P = 0$, No (zero) Padding
- $P = 1$, One pixel (Zero) ~, Zero-padding
- $P = \lfloor \frac{K-1}{2} \rfloor$, Same ~ (Zero)
- $P = K-1$, FULL Zero ~

Stride S

- $S = 1$, Unit stride, highest resolution
- $>$ Strided conv, ↓
- $<$ Fractional stride, used in transposed conv. $Size_{out} > Size_{in}$

Weight matrix $C_{out} \times C_{in} \times K_H \times K_W$

#filters

Bias vector C_{out}

Output size (# output ele) $C_{out} \times H' \times W'$

$$H' = \frac{(H - K + 2P)}{S} + 1$$

$$W' = \frac{(W - K + 2P)}{S} + 1$$

无法整除则 Floor

Memory usage

For 32-bit floating point, bytes per ele : 4 (unit)

Weights m $= C_{in} \times C_{out} \times K_H \times K_W \times 4$ ①

biased m $= C_{out} \times 4$ ②

feature maps m:

input m $= H_{in} \times W_{in} \times C_{in} \times 4$ ③

out m $= H' \times W' \times C_{out} \times 4$ ④

Total mem $= \sum_{①}^{④}$

Full Form	Units	Bytes
1 Bit	Binary Digit (0/1)	
1 Nibble	4 bits	
1 Byte	8 bits	
1 kilobyte(KB)	1024 byte	2^{10} bytes
1 Megabyte(MB)	1024 KB	2^{20} bytes
1Gigabyte (GB)	1024 MB	2^{30} bytes
1 Terabyte(TB)	1024 GB	2^{40} bytes
1 Petabyte(PB)	1024 TB	2^{50} bytes
1 Exabyte(EB)	1024 PB	2^{60} bytes
1 Zettabyte(ZB)	1024 EB	2^{70} bytes
1 Yottabyte(YB)	1024 ZB	2^{80} bytes
1 Brontobyte	1024 YB	2^{90} bytes
1 Geopbyte	1024 Brontobyte	2^{100} bytes

#Para (#weight) = weight shape + bias shape

$= C_{in} \times C_{out} \times K_W \times K_H + C_{out}$

#FLOP = # Output size × Oper per out ele

floating pt. oper. $= C_{out} \times H' \times W' \times C_{in} \times K_H \times K_W$

Pooling

无可学习参数, $C_{in} = C_{out}$, 有 S. P. H'W' 计算同 conv.

FLOP $= C_{out} \times H' \times W' \times \begin{cases} (K \times K - 1) & \text{max-pooling} \\ K \times K & \text{avg-} \sim \end{cases}$

Flatten

无可学习参数

Output size $C_{in} \times H \times W$

FLOP 0

For each class: **confidence score** the probability that the prediction is correct.

e.g. person

IOU intersection-to-parallel ratio (Jaccard)

1° 给定坐标 → 画图 → 面积交集除以并集

2° 给定矩阵 → pred 标记 → $\frac{TP}{TP+FP+FN}$ ← 预测完area

TP、FP、FN 点 重叠 边角料

Dice $= \frac{2TP}{2TP+FP+FN}$

贴得更2次，都飞了

precision $= \frac{TP}{TP+FP}$ → 预测为 1 预测精度

recall $= \frac{TP}{TP+FN}$ → 真实1中哪些预测到，召回

Non-Maximum Suppression

removes **redundant** bounding boxes from a set of detected objects

① Bounding box generation

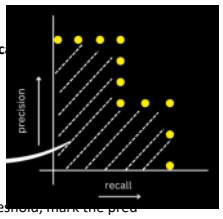
② Sorting by cs (conf score)

③ Suppression:

1) Select the box with the highest cs
2) Cal IoU with all other boxes
3) Suppress overlapping boxes (exceeds a predefined IoU threshold)
4) Repeat This continues until all boxes have been either selected as a detection or suppressed.

average precision (AP)

src:Mean Average Precision (mAP) | Explanation and Implementation for Object Detection

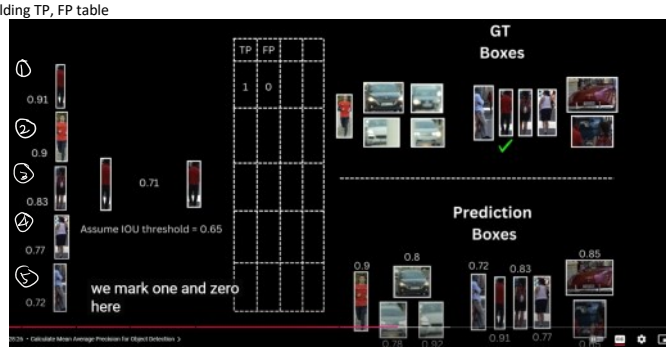


The average of precision **at different recall levels**. It is calculated as the **area under the precision-rec**
interpolation of precision at different recall points.
Improving one often comes at the cost of reducing the other.

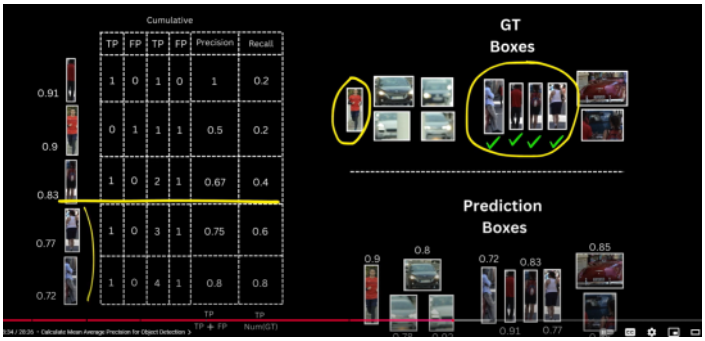
AP for person class:

a. take all the predicted boxes that have person as the pred class
b. sort them in the decreasing order of confidence scores
c. For each, find the best matching ground truth box for matching (based on overlap(Area of this GT box has not been matched before, and its IoU us greater than some predefined thre
as a TP, GT as matched.
d. building TP, FP table

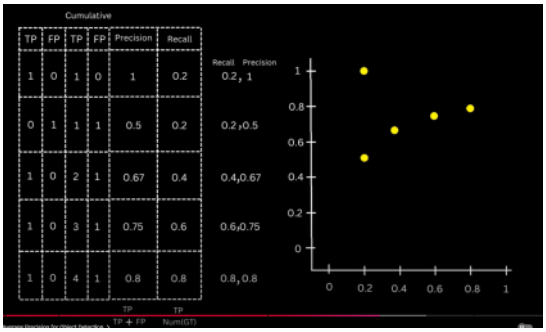
i.



ii.



We got the precision and recall values using different confidence threshold. e.g. 0.83. 2 detection below it will be ignored.? ?



link them -> precision recall curce -> Find the area under the curve (AUC-PR).
For object detection tasks, AP (Average Precision) is equivalent to the area under the Precision -Recall curve.

In VOC 2007, AP is calculated using the 11-point interpolation method, but modern object detection frameworks often use a finer, continuous approximation to compute AP.

In VOC2012, mAP = meanPrecision * meanRecall

Doing this among all classes and doing the mean of the APs, will get mAP.

For all classes
mean average precision