YOLO v1

↳ first real-time (over 30fps) object-detector

Will Cover

1. What's improved? (or suggested?)

- object-detection as single-regression problem
- three benefits over traditional models

2. Architecture & Computation flow

- network desingn
- how raw-image pass through model (checking in/out of each layer)

3. Train & Inference

- understanding each term of SSE
- using λ_{coord}, λ_{noobj} parameters

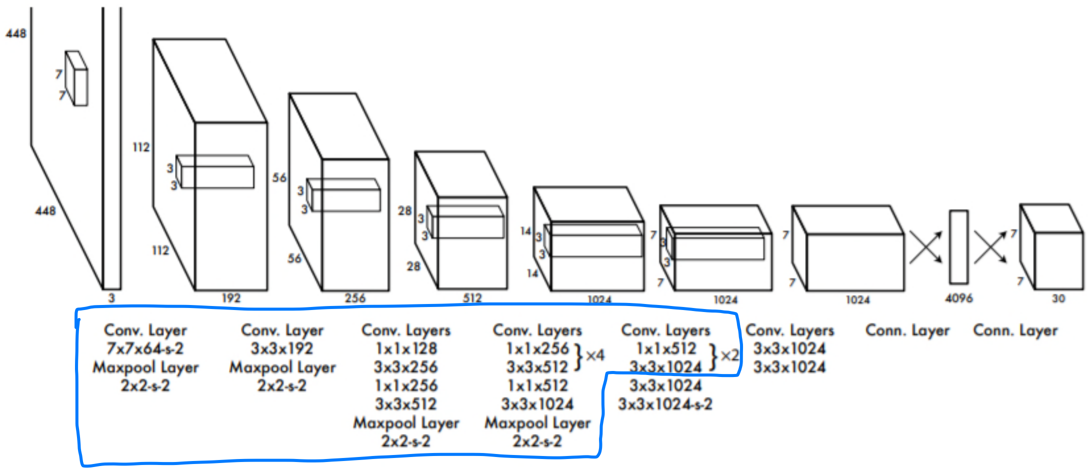
4. Limits & Comparison

- limits : spatial-constraint, small-object prob, coarse feature, loss balance
- comparison : DPM, Deep-Multibox, OverFeat, Multigrasp 저리

3. Train & Inference

학습을 위한 Ground-truth, dataset 정의 및 Loss function 에 대해 알아보기 전에 먼저 학습 전 선행 절차를 잠깐 보자면,

① GoogleNet 을 변형하여 만든 feature extractor 를 Imagenet data 를 통해 pre-train (only bottom 20 - conv. layers)



② 입력 사이즈를 증가, $224^2 \rightarrow 448^2$

검출 task 에서는 저해상도가 성능저하에 영향이 있을 확률이 높음

(= 증가시킨 input_shape 에 알맞은 FC 를 include 했다는 뜻)

해당 작업이 끝나면, 모델이 학습할 '정답' 에 맞게 Ground-truth 와 데이터셋 정의가 필요.

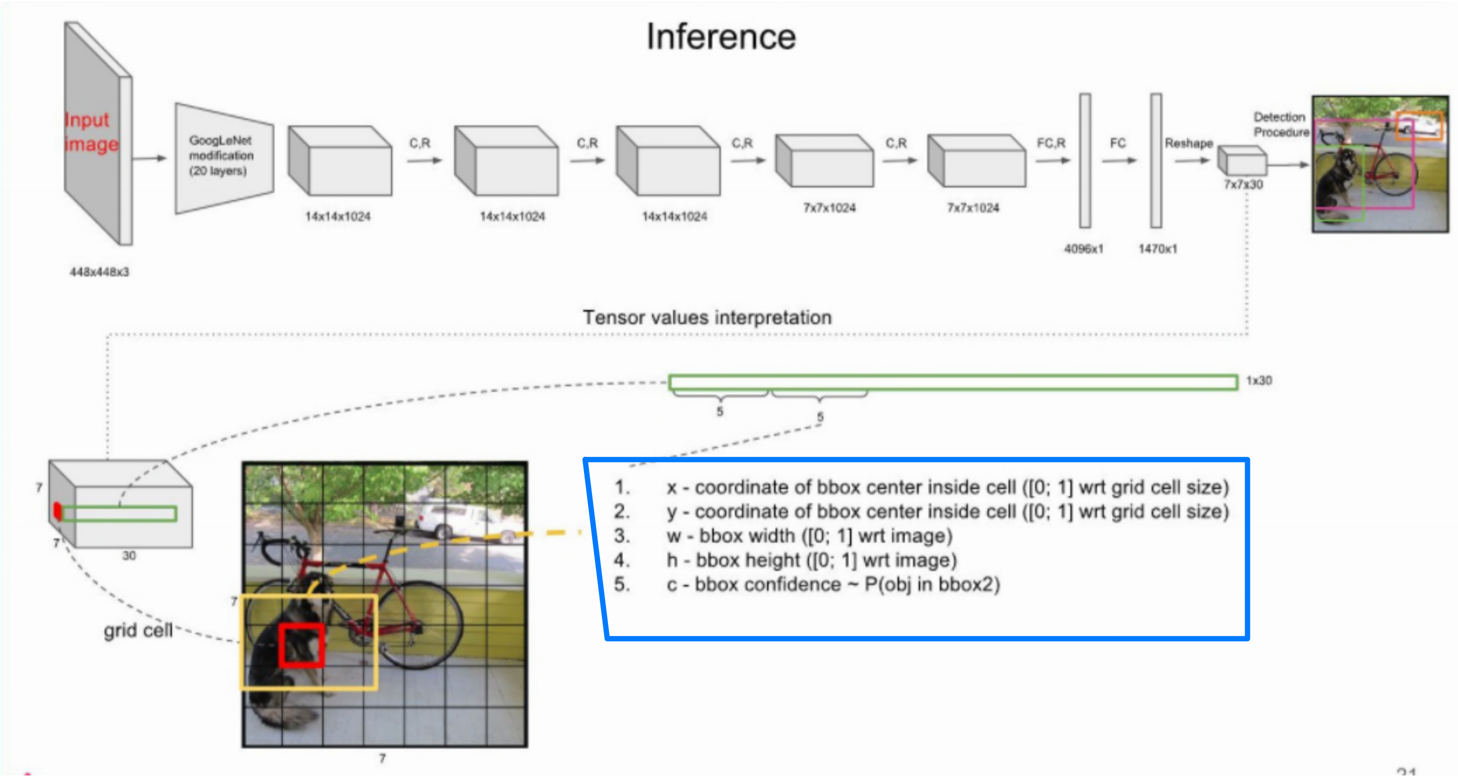
• **Ground-truth 정의.**

↳ YoLo 의 경우, 기존 2-stage 모델들과는 다른 parametizing 을 통해 Ground-truth 를 정의하는데,

Yolo 의 출력인 $7 \times 7 \times 30$ 에서 한 grid 에 대한
정보인 $1 \times 1 \times 30$ 데이터는 크게 2가지로 나눌 수 있다.

1. Bbox 정보 , 2. Class-probability 정보

각각에 대해서 Ground-truth 를 정의 해보자면,

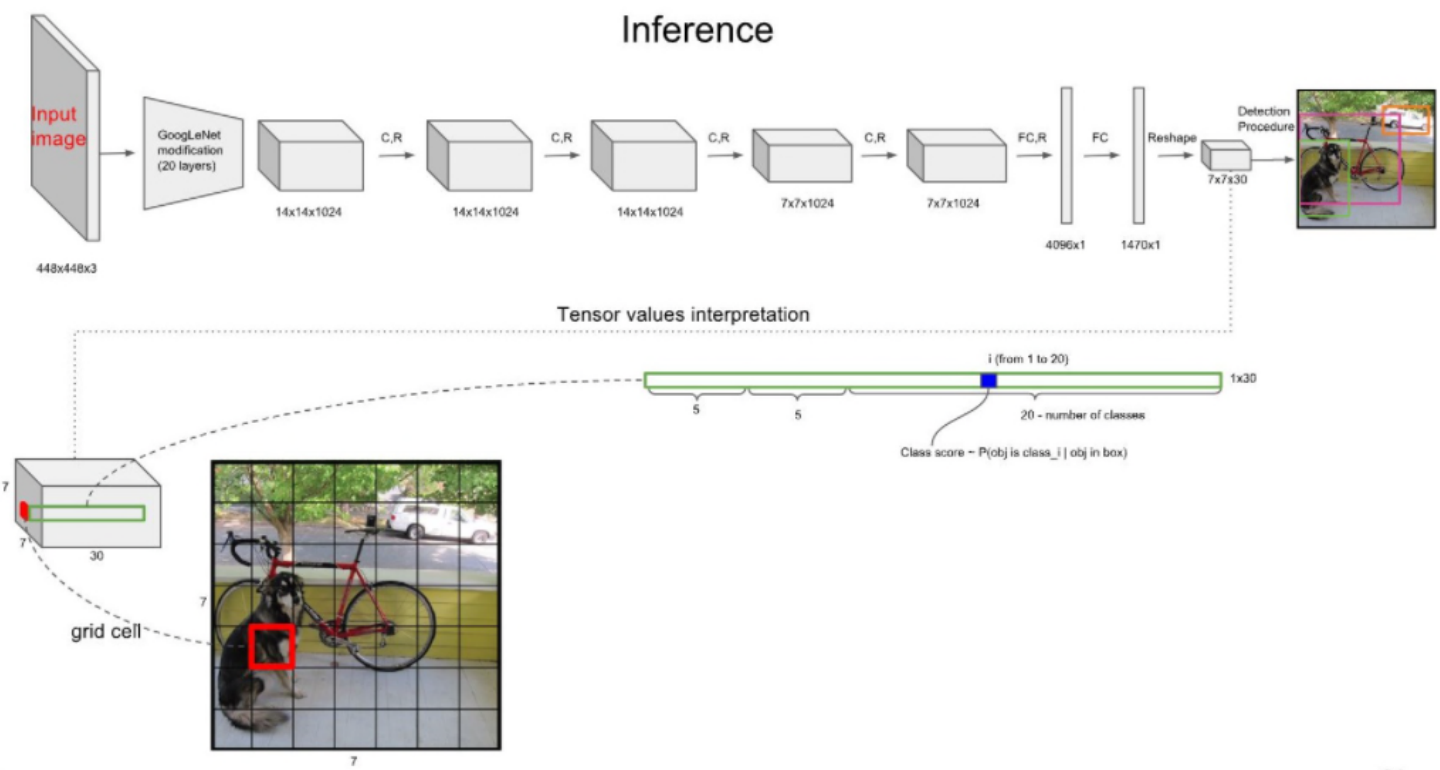


논문 기준 1 grid 당 2개의 Bbox 예측을 진행하고
각 Bbox 정보는 5개의 수로 구성

※ R-CNN family 처럼 정답의 x, y, w, h 와 Anchor (or ROI) 의 x, y, w, h 사이
파라미터화 된 t_x, t_y, t_w, t_h 가 회귀식의 target 이 아니다.
1-stage method 는 '후보영역' 의 개념 없이 바로 정답 x, y, w, h 를 회귀하는 것.

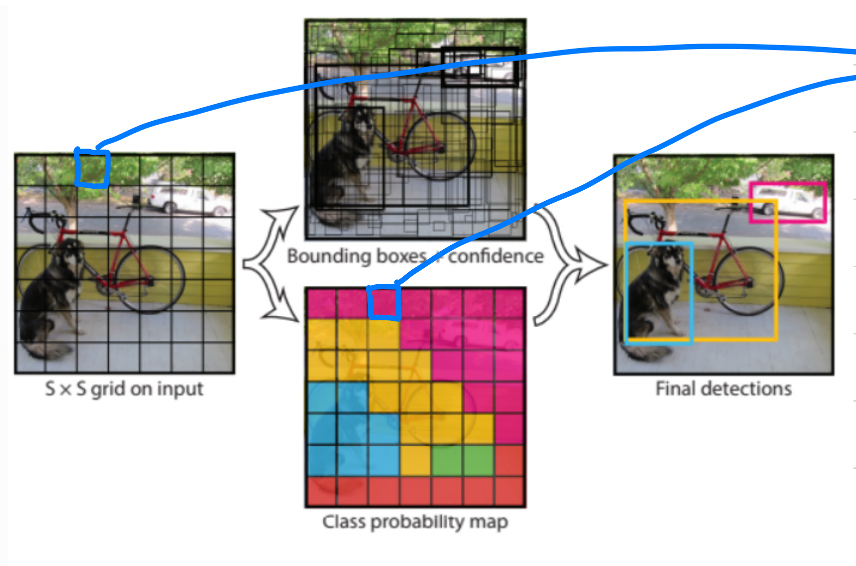
단 x, y, w, h 의 raw value 를 예측하는 것이 아닌, x, y 는 소속된 grid를 기준으로
[0,1] 사이로 normalize, w, h 는 이미지 전체 사이즈 대비 크기로 [0,1] normalize 한 값을 회귀한다.

Confidence Score 의 경우, $score = P_r(obj) \cdot IOU_{(any\ GT)}$ 이므로
해당 grid에 객체가 있으면 $P_r(obj) = 1$, 없으면 $P_r(obj) = 0$ 으로 GT 를
설정한다.



Class probability 의 경우, 해당 grid 에 '물체가 있다고 가정' 할 경우 어떤 객체 일지 클래스 별로 확률을 의미하며 YOLO 가 출력하는 특정 grid 의 Conditional class-probability 는 (0.01, 0.07, 0.89, ..., 0.004) 이런 식이 겠지만 우리가 GT로 설정하는 벡터는 해당 클래스만 1인 (0, 0, 1, ..., 0) 의 모습이다.

※ 한가지 결국 못찾은 내용은, 실제로 객체가 있는 grid의 경우 별 문제 없이 납득이 가나 객체가 없는 grid 의 경우 어떤 클래스가 들어 있는 grid 라고 봐야 하는 것인지에 대한 내용이 그 어디에도 없다.. 심지어 YOLO - annotation 형식을 따르는 Custom dataset 을 만들며, 저런 annotation 작업은 한 기억이 없는 걸로 봐서 내부에 이를 처리하는 로직이 있는 듯한데, 그 아무도 이걸 궁금해 하지 않는지 내용이 없다 ....



무슨 근거로

도대체 .... ?

• Loss Function 정의

↳ object detection 을 single-regression 으로 접근 한다는 것이 기존의 방식보다 단순하지만 그에 따른 리스크 또한 존재한다.

Yolo 에서는 최적화가 쉬운 SSE 손실을 사용하는데, 그렇게 되면 2가지 위험이 있다.

1. 같은 0.1 손실이라도 LOC 손실과 CLS 손실이 가지는 의미가 다르다. LOC 손실은 0~1 사이로 정규화된 상태로 계산된 차이 이므로 실제로는 더 큰 오차를 의미하고 두 손실의 가중을 다르게 설정하여야 함.

2. 전체 grid 에서 실제 객체가 들어있는 grid의 수는 상대적으로 소수이기에 confidence score 의 ground-truth 대부분의 값은 0 이다. 이러한 레이블 불균형은 실제로 모델이 0에 가까운 confidence score 를 예측하는 편향을 가지는 데 영향을 주기 때문에 객체가 있는 grid 에서 발생하는 손실과 없는 grid 에서 발생하는 손실의 가중을 다르게 해야한다.

이러한 이유로 Yolo v1 에서는 이러한 가중 차이를 위해 섬세한 손실 함수를 제안 한다.

Loss =

$$
\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]
+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]
+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2
+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2
+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \quad (3)
$$

한 번에 보기에는 다소 어려워 보이지만, 각 세부 내용을 보면

Notation이 복잡할 뿐, 수식 자체가 어렵진 않다.

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

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

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

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

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \qquad (3)$$

Bbox loss

cls loss

크게는 이렇게 두 부분으로 나누어 볼 수 있다. 먼저

Bbox loss 부터 살펴보자면,

???

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

x, y, w, h

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

???

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

confidence score

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

눈에 들어오는 부분들과 잘 모르겠는 부분이 존재한다.

현재 Bbox 예측 정보에 대한 SSE 계산 식 이므로 x, y, w, h, c 가 (무엇 - 무엇)^2 형식으로 계산되는 식에 들어 가있는 것은 눈에 들어온다.

$\sum_{i}^{S^2}\sum_{j}^{B}$ 이 부분은 현재 7×7 (s=7)로 grid를 나누었고 각 grid 당 2개(B)의 Bbox를 예측하도록 설계되어 있기 때문에 결국 모든 grid의 모든 Bbox에 대한 loss를 sum 하겠다는 표현이다.

눈에 잘 들어오지 않는 부분은 다음과 같은 4가지 이다.

$$
\begin{aligned}
&① \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} ② \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(③ \sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&④ + \lambda_{\textbf{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2
\end{aligned}
$$

① SSE 사용의 위험 얘기 중 첫 번째 해당하는 LOC loss를 더 가중하기 위한 balancing parameter 이다. 논문에서는 $\lambda_{coords} = 5$ 를 사용 하였다고 한다.
자세히 살펴보면 x, y, w, h 손실 계산 식에만 λ_{coords} 가 적용되는데 이는 confidence score ≠ LOC loss 이기 때문이다.
정확히는 Bbox loss = LOC loss + confidence loss 라고 할 수 있다.

② $\mathbb{1}_{ij}^{obj}$ 이 notation 은 현재 손실을 계산할 Bbox 가 어떤 Bbox 인가? 에 대한 표기이다.

여기서 i 는 몇번째 grid 인지, j 는 해당 grid 에서 몇번째 Bbox 인지, obj 는 해당 Bbox 에 객체가 있음을 의미한다.

즉, $\sum_{i}^{S^2}\sum_{j}^{B}\mathbb{1}_{ij}^{obj}(\quad)^2$ 의 의미는 결국 S^2 개의 grid 에 각각 존재하는 B 개의 Bbox, 총 $S^2 \cdot B$ 개 있는 Bbox 중 객체가 실제로 내부에 있는 Bbox 만 손실을 계산하여 sum 하겠다는 뜻!!!

③ 왜 x, y 는 $(x-\hat{x})^2$ 로 계산하고 w, h 는 $(\sqrt{w}-\sqrt{\hat{w}})^2$ 로 ??

앞서 SSE 손실 사용의 2가지 위험 이외에 사실 1가지 문제가 더 있는데, 바로 small / large Bbox 에 대해 동일한 가중이 적용된다는 것이다. 정답 $w=12$ 인 상황에서 $\hat{w}=11$ 로 예측하여 1 차이 가 나는 것과, $w=1.2$ 인 상황에서 $\hat{w}=0.2$ 로 1 차이 나는 것은 잘못 예측한 정도가 다르게 가중되어야 한다. 이에 YOLO 에서는 객체의 크기를 담당하는 w, h 는 root-scale 을 사용하여 이를 보완한다.

④ λ_{noobj} 와 $\mathbb{1}_{ij}^{noobj}$ 는 무엇일까?

이 부분은 SSE 를 손실로 쓰며 조심해야 할 2가지 위험 중 2번째 에 대한 대응 방법에 대한 내용이다.

λ_{coords} 와 비슷하게 λ_{noobj} 는 객체를 포함하지 않는 Bbox 의 손실 계산시 적용되는 balancing parameter 로 논문에서는 $\lambda_{noobj}=0.5$ 를 사용하였다고 한다.

$\mathbb{1}_{ij}^{noobj}$ 는 $\mathbb{1}_{ij}^{obj}$ 와 반대로 객체를 포함하지 않는 i 번째 grid 의 j 번째 Bbox 를 지칭하는 notation 으로 눈여겨 볼 점은 x, y, w, h 의 경우 $\mathbb{1}_{ij}^{obj}$ 에 대해서만 손실을 계산하는 반면 c 의 경우 $\mathbb{1}_{ij}^{obj}$ 뿐만 아니라 $\mathbb{1}_{ij}^{noobj}$ 에 대해서도 계산한다는 점이다.

($\because$ 객체가 없는 Bbox 의 경우 $c=0$ 이라는 target 값이 있으므로)

자, 이를 종합적으로 Bbox loss 를 다시 살펴보며 전체를 말로 해석하면,

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

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

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

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

x, y, w, h, c 중 물체의 '위치' 에 대한 오차인 x, y 의 경우 $S^2 \times B$ 개의 Bbox 중 물체가 실제로 있는 Bbox 에 한해 raw-scale 로 SSE 를 계산하여 λ_{coords} 를 곱해 sum 하고,

x, y, w, h, c 중 물체의 '크기' 에 대한 오차인 w, h 의 경우 $S^2 \times B$ 개의 Bbox 중 물체가 실제로 있는 Bbox 에 한해 root-scale 로 SSE 를 계산하여 λ_{coords} 를 곱해 sum 하고,

x, y, w, h, c 중 물체의 '존재확률' 에 대한 오차인 C 의 경우 $S^2 \times B$ 개의 Bbox 중 물체가 실제로 있는 Bbox 는 raw-scale 로 SSE 를 계산하여 sum 하고,

x, y, w, h, c 중 물체의 '존재확률' 에 대한 오차인 C 의 경우 $S^2 \times B$ 개의 Bbox 중 물체가 실제로 없는 Bbox 의 경우 raw-scale 로 SSE 를 계산하여 λ_{noobj} 를 곱하여 sum 한 것이 최종적인 Bbox loss 이다.

$$\lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$
$$+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$
$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$
$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \quad (3)$$

Bbox loss

Cls loss

그 다음은 conditional class probability 에 대한 손실인 Cls loss 를 살펴보자면

$$\boxed{+ \sum_{i=0}^{S^2}} \boxed{\mathbb{1}_{i}^{\text{obj}}} \boxed{\sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2}$$

이렇게 3가지로 나누어 해석할 수 있는데,

먼저 1번의 경우 앞선 Bbox loss 와 다르게 $\sum_i^{S^2} \sum_j^{B}$ 가 아닌 $\sum_i^{S^2}$ 임을 통해 Conditional-class prob 예측은 각 Bbox 별로 하는 것이 아닌 grid 별로 이루어 짐을 다시 확인 할 수 있다.

2번의 경우 앞서 확인한 $\mathbb{1}_{ij}^{obj}$ 와 비슷한 의미로 실제 물체의 중심이 존재하는 "grid" 를 지칭하는 notation 이다.

3번의 경우 각 클래스에 대한 Conditional class probability SSE 를 뜻하며,
num classes = 3 인 경우를 가정하여 간단한 예시를 들자면,

$$p_i = (1, 0, 0) \qquad \hat{p}_i = (0.7, 0.1, 0.2) \Rightarrow \sum_{c}^{3} \left(p_i(c) - \hat{p}_i(c)\right)^2 = 0.14$$

같은 계산이 이루어 진다.

지금까지 Yolo v1 학습을 위한 사전 준비, GT 설정, Loss 설정을 알아보았다. YOLO v1 또한 여러 trade-off를 가지고 있지만 이후 버전에서 다수 해결되었으며 RP → Judge 의 2-stage 를 하나의 출력에 모두 담는 접근이라는 점은 object-detection 분야에 큰 기여임은 분명하다.