YOLO(You Only Look Once)

국민대 소프트웨어 학부
학부생 연구 기회 프로그램(UROP)
김대희

(2019.3.22)

INDEX

1. YOLO 이전의 Detection - Faster RCNN

- Region Proposal Network(RPN) 이후 ROI 풀링

=>object 가 있을 것 같은 공간을 300여개까지 따로 추출함

- 다뤄야 하는 공간의 수가 너무 많음

=>작은 크기에서 정확하게 잘 맞추지만, 속도가 느림(약 5fps)

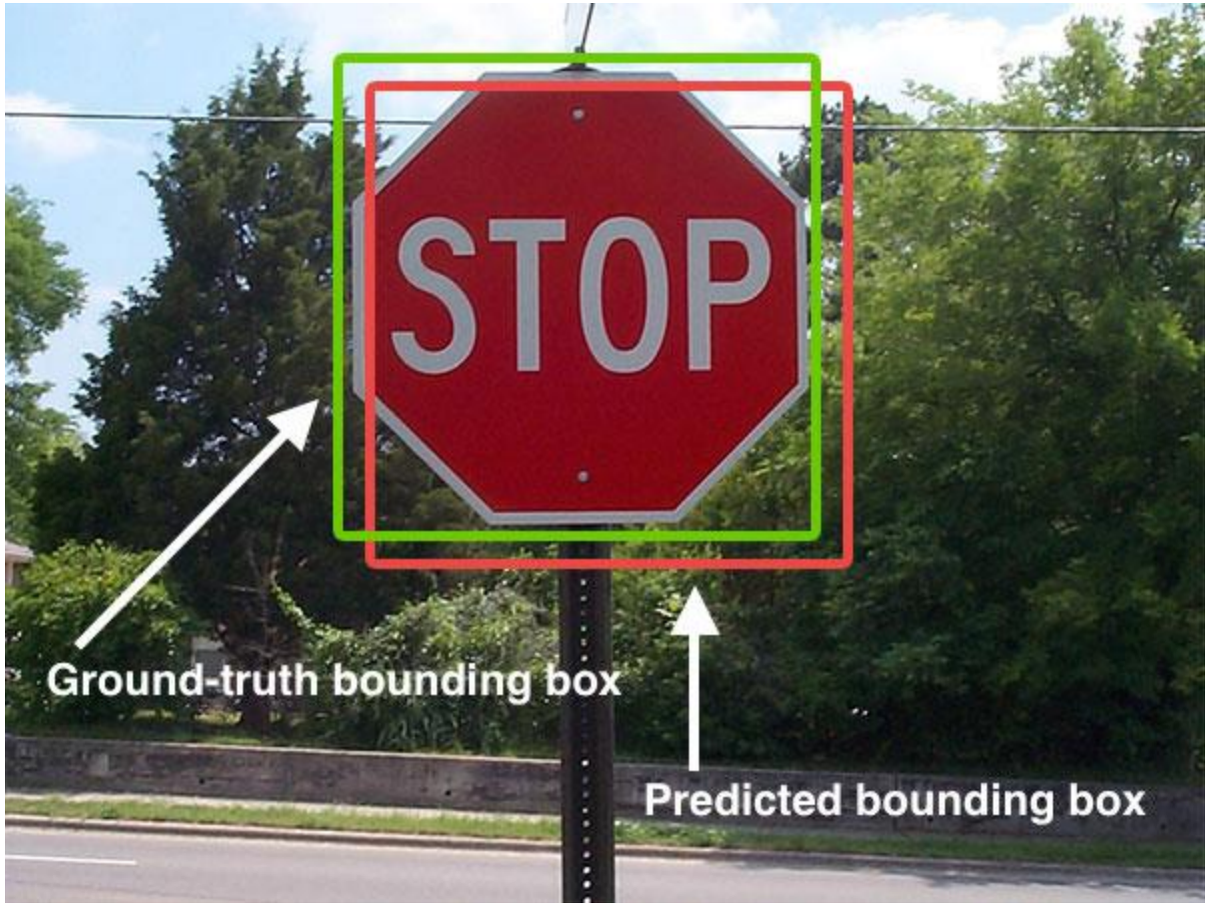
- 학습이 비교적 복잡함

2. YOLO's Unified Detection

- Single neural network로 전체 이미지를 봄
- Grid를 그어 한 Cell 마다 B개의 bounding box를 작성하고 (x,y,w,h)
  각 박스마다 Confidence score(objec가 있는지 신뢰도를 수치화) 를 산출
- Confidence score : Pr(object) * IOU(truth pred)

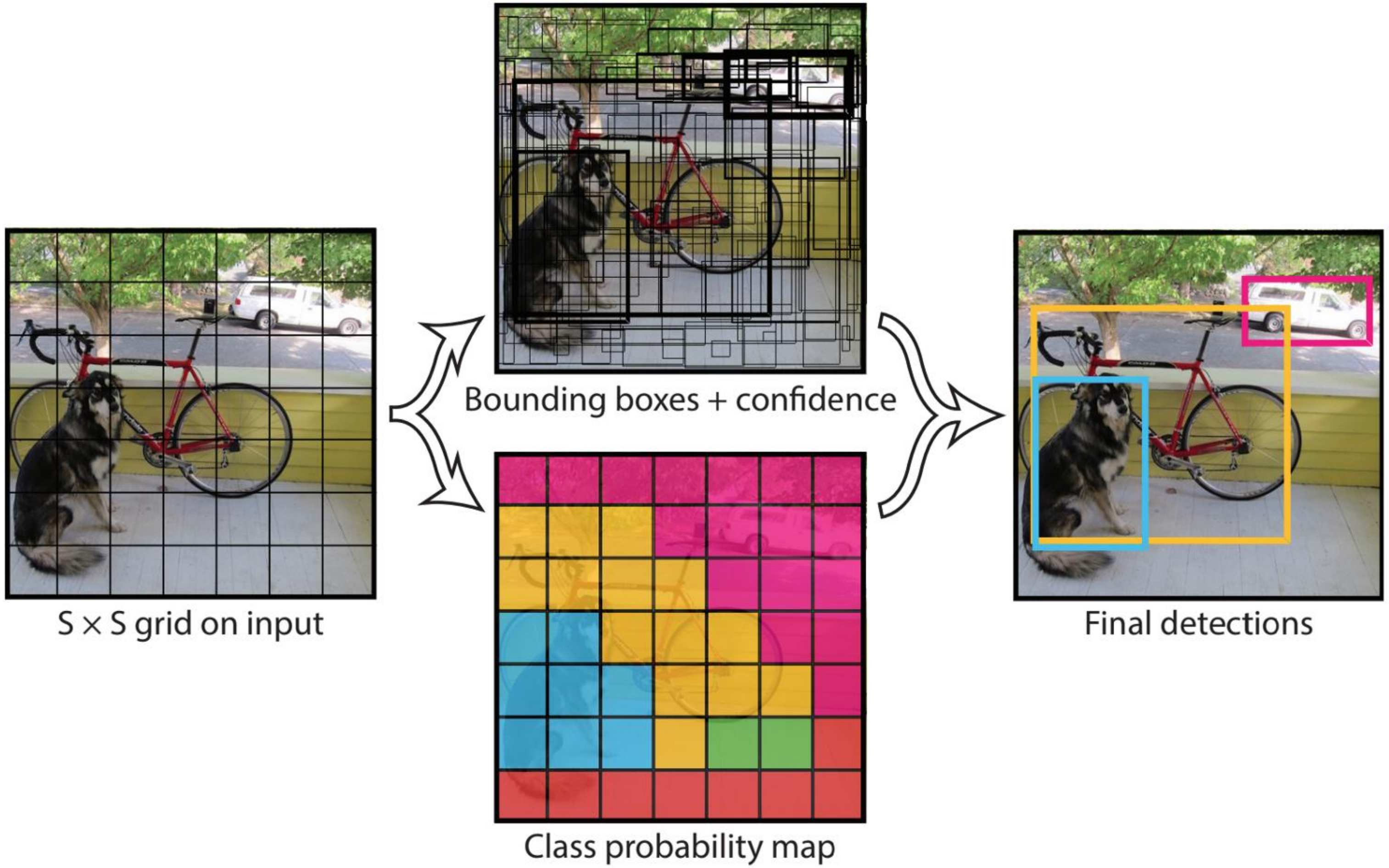
- 타이탄x기준 45fps, 정확도는 fast-RCNN 보다 약간 낮음
- Background error가 낮음 (이미지를 한번에 보기 때문에)
- A regression problem으로 간주, 학습이 편함

2+ IOU(Intersection Over Union)



$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

2. YOLO's Unified Detection



2. YOLO's Unified Detection

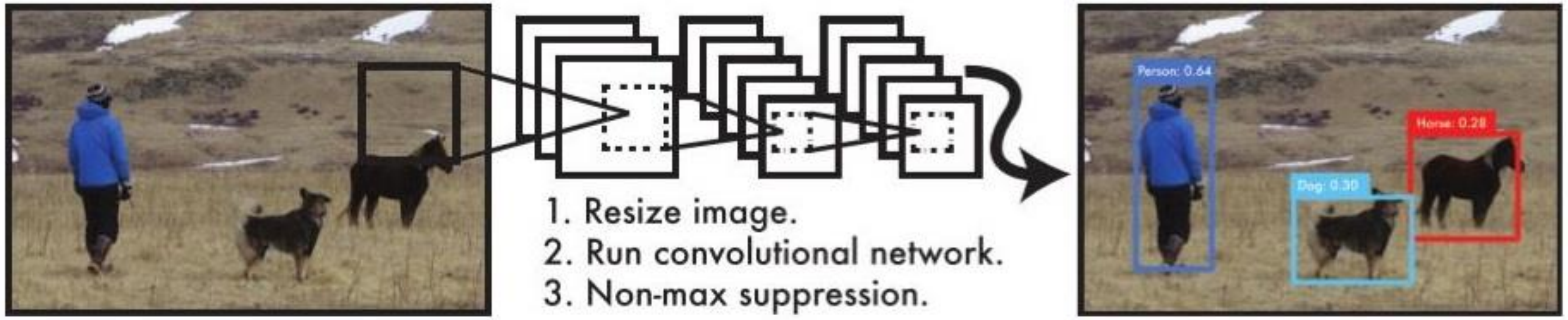


Figure 1: The YOLO Detection System. Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448×448, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.

3. YOLO's Design

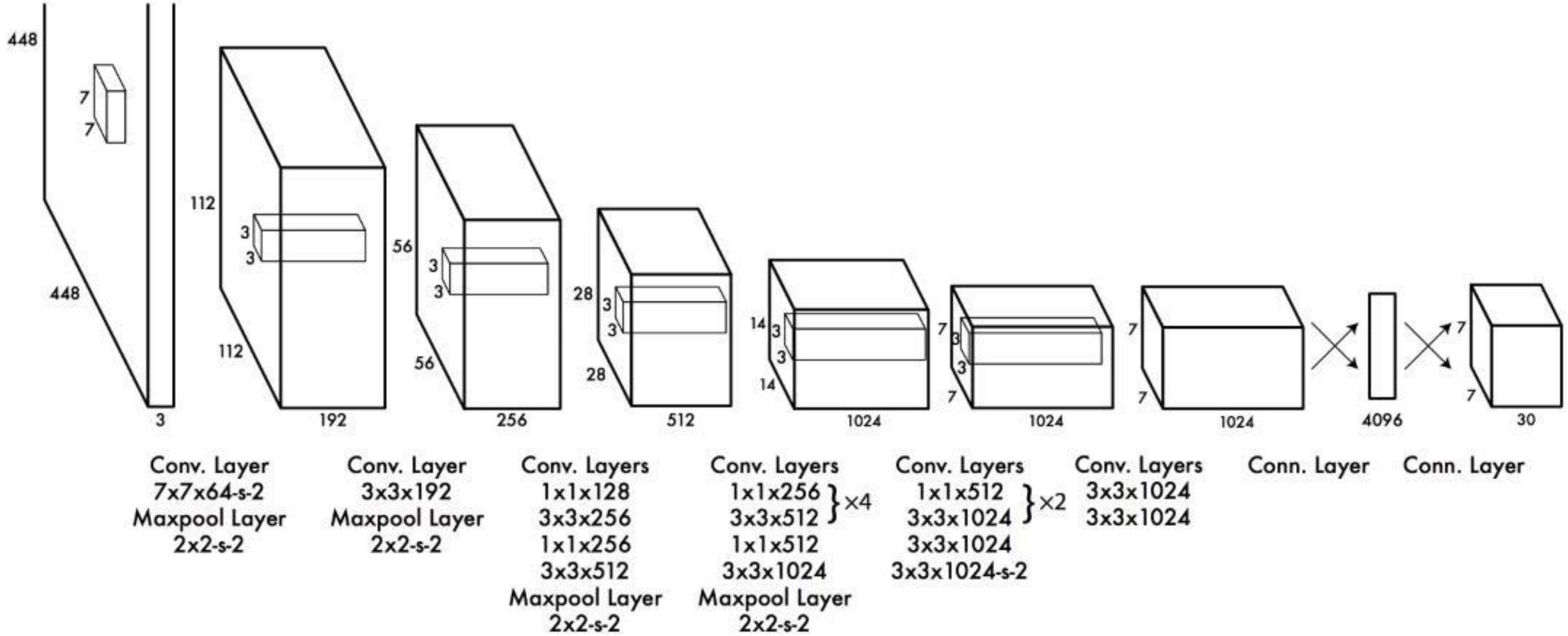
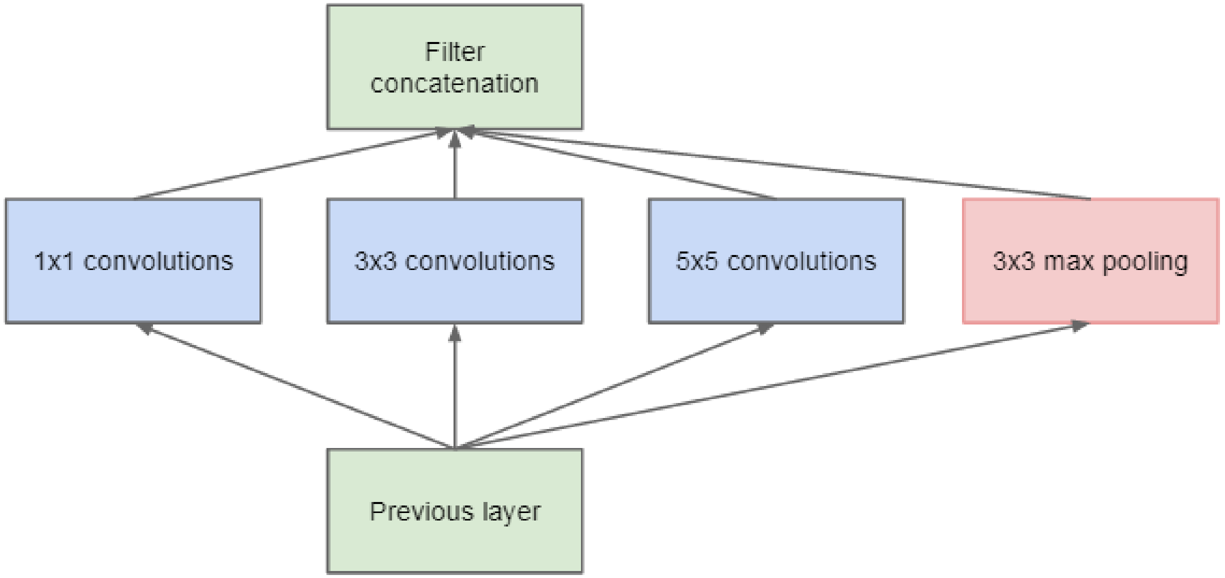
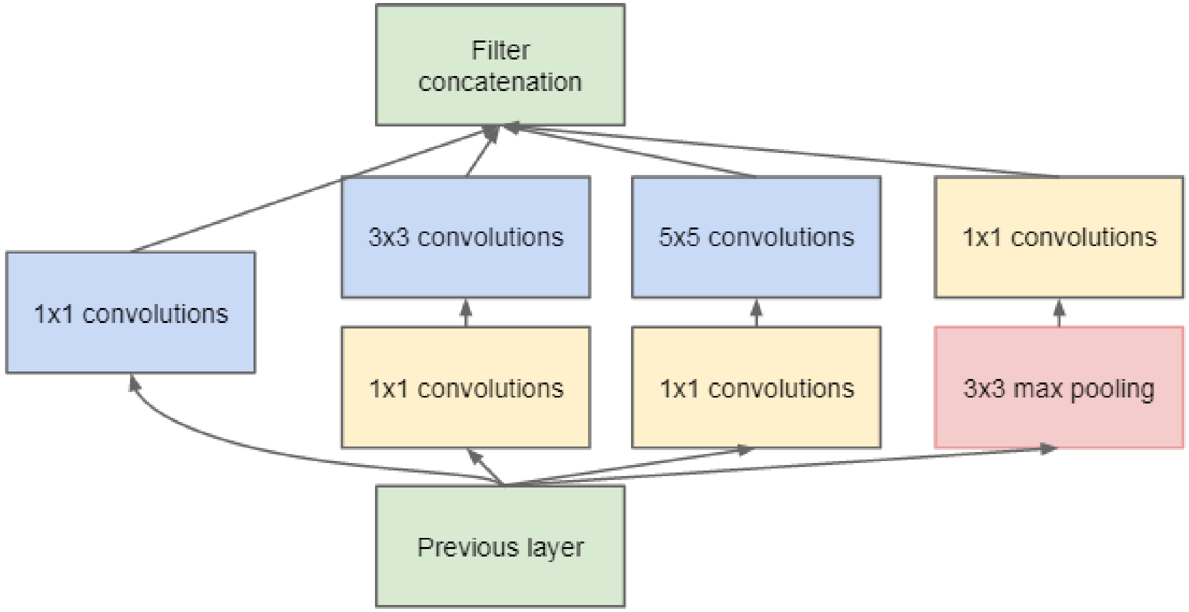


Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

3+ Inception module in googLeNet



(a) Inception module, naïve version



(b) Inception module with dimension reductions

YOLO에서는 단지 1*1 reduction layer 에 3*3 conv를 붙여 사용

3. YOLO's Design

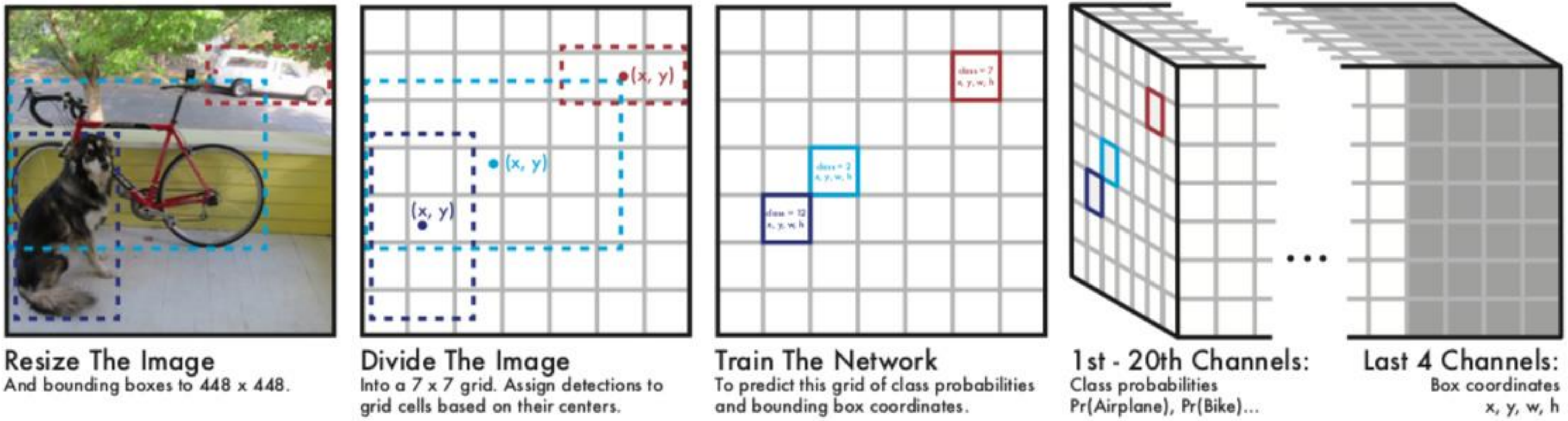
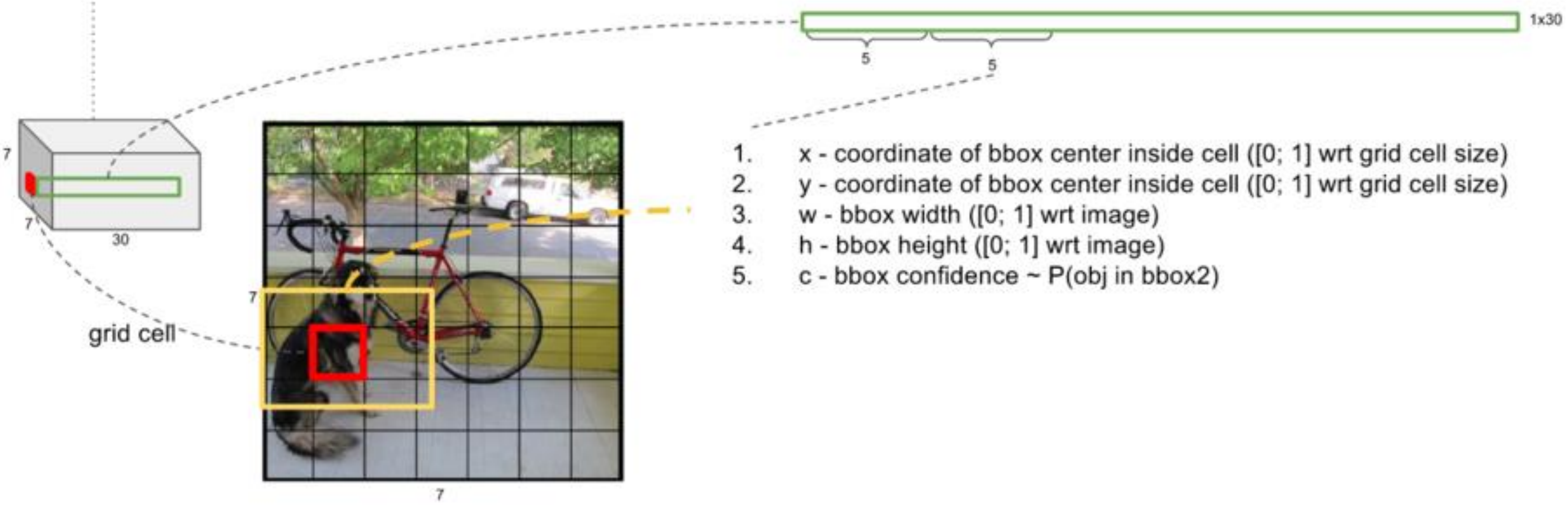
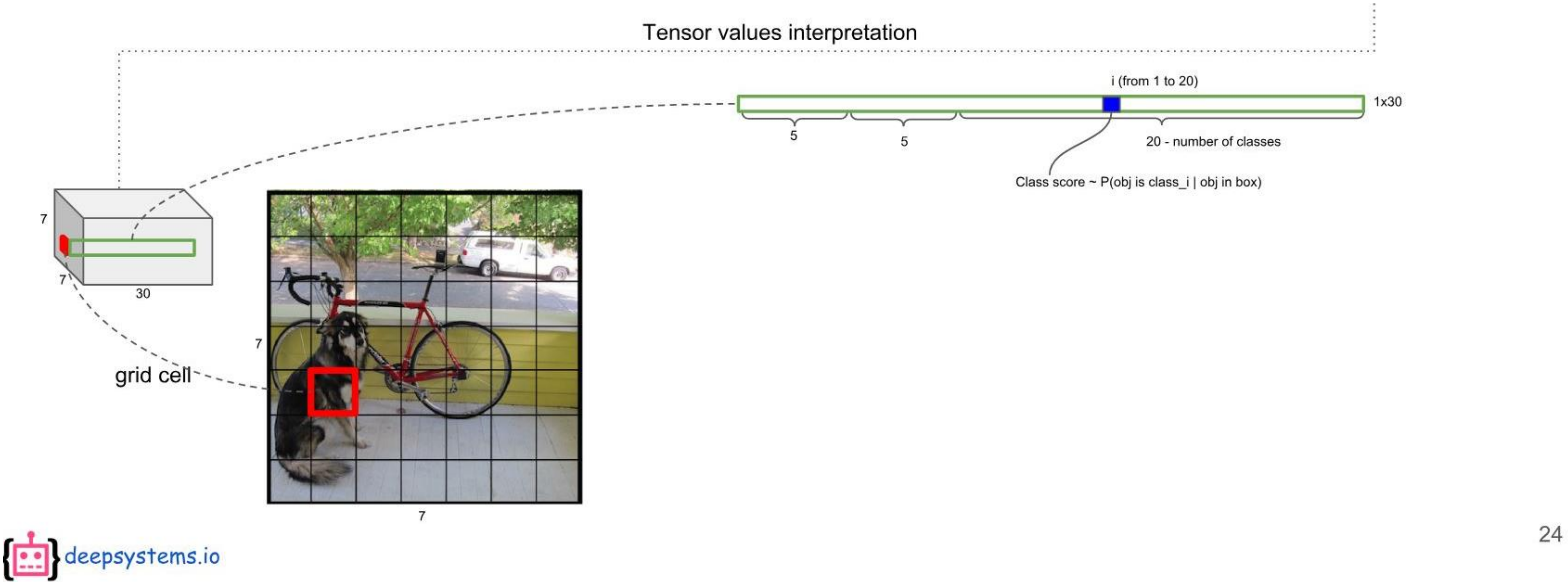


Figure 2: The Model. Our system models detection as a regression problem to a $7 \times 7 \times 24$ tensor. This tensor encodes bounding boxes and class probabilities for all objects in the image.

3. YOLO's Design - output



3. YOLO's Design - output



4. Non-Maximum suppression algorithm

-하나의 객체를 중복으로 찾지 않기 위해 거르는 과정

-https://docs.google.com/presentation/d/1aeRvtKG21KHdD5lg6Hgyhx5rPq_Z0sGjG5rJ1HP7BbA/pub?start=false&loop=false&delayms=3000&slide=id.p

5. Training - Loss Function

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

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

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

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

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

S = grid Size
B = bounding Box 개수(논문기준 2)

$\lambda_{coord} = 5$ $\lambda_{noobj} = 0.5$

Object가 존재할 때 가중치를 더 주고,
Classification보다 Localization에
가중치를 더 줘서
의도대로 학습이 잘 되도록 함

5. Training - Loss Function

S = grid Size (논문기준 7)
B = bounding Box 개수(논문기준 2)

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

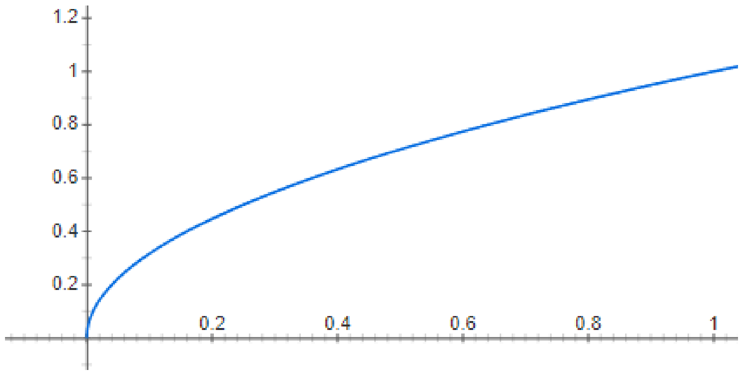
i번째 셀의
j번째 바운딩박스에
오브젝트가 존재할때

5. Training - Loss Function

S = grid Size
B = bounding Box 개수(논문기준 2)

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

바운딩 박스 크기가
클 때보다 작을 때
민감하게 작용하도록
제곱근을 씌워 줌



6. 한계점

-분류 능력 자체는 다른 모델(Faster R-CNN)보다 떨어짐

-빠르고 배경정보에 강하지만,
 군집한 작은 객체를 인식하는 데 애를 먹음

-바운딩 박스를 정확히 잡는 것에 약함(localization error)

끝. 감사합니다.

-참고

-https://pjreddie.com/media/files/papers/yolo.pdf

-http://christopher5106.github.io/object/detectors/2017/08/10/bounding-box-object-detectors-understanding-yolo.html

-https://docs.google.com/presentation/d/1aeRvtKG21KHdD5lg6Hgyhx5rPq_ZOsGjG5rJ1HP7BbA/pub?start=false&loop=false&delayms=3000&slide=id.p