You Only Look Once:
Unified, Real-Time Object Detection

Joseph Redmon (University of Washington),
Santosh Divvala
(University of Washington, Allen Institute for AI),
Ross Girshick(Facebook AI Research)
Ali Farhadi(University of Washington, Allen Institute for AI)

Index

Abstract

· YOLO가 나오기 전에 Detector(특히 Region Proposal 기반의 Detector인 R-CNN)는 Attention Mechanism을 적용했기 때문에 Unified되었다고 보기 힘들었다(Faster R-CNN도 RPN, Fast R-CNN Detector, Feature Extractor로 나뉘어짐).

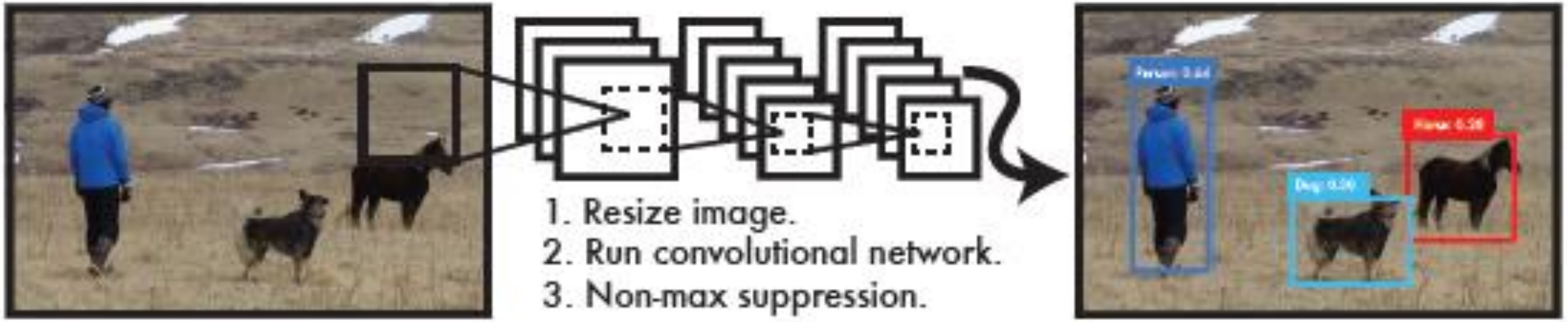
· YOLO는 전체 이미지를 여러 구역으로 나누어서 모두 검사하는 방식으로 탐지를 수행함.

=> Attention Mechanism에 필요한 구성 요소가 필요 없어지므로 말 그대로 이미지의 전체 특징을 보고 탐지가 가능함.

· Unified된 네트워크를 바탕으로 속도에 강점을 보임. 즉 한 장의 이미지를 여러 번 볼 필요가 없어짐.

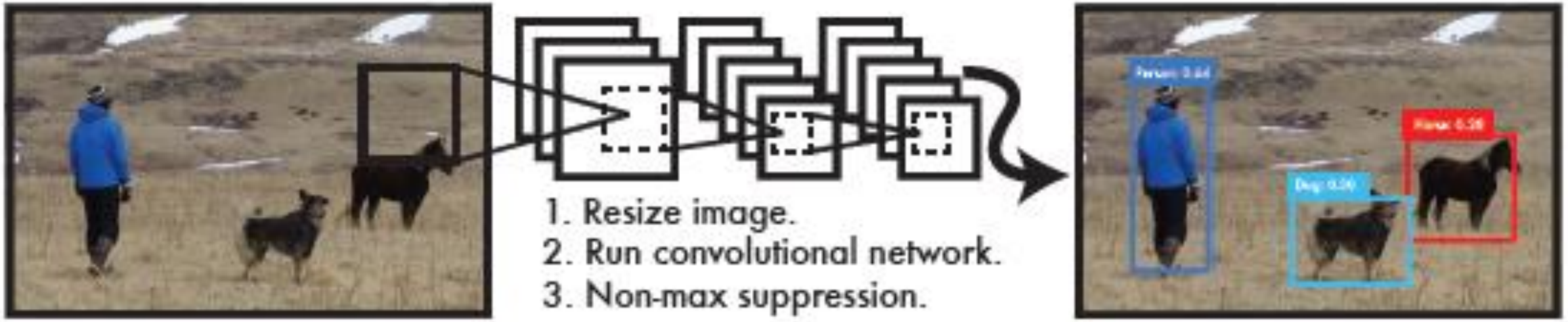
=> YOLO는 45 FPS, Fast YOLO는 155 FPS(크기가 좀 더 작은 모델)

Introduction



YOLO에서는 Unified된 네트워크를 구성하기 위해서 탐지 문제를 Attention Mechanism 대신 Bounding Box 좌표와 Class Probability에 대한 Single Regression Problem(하나의 통일된 역전파로 네트워크 업데이트)으로 봄.

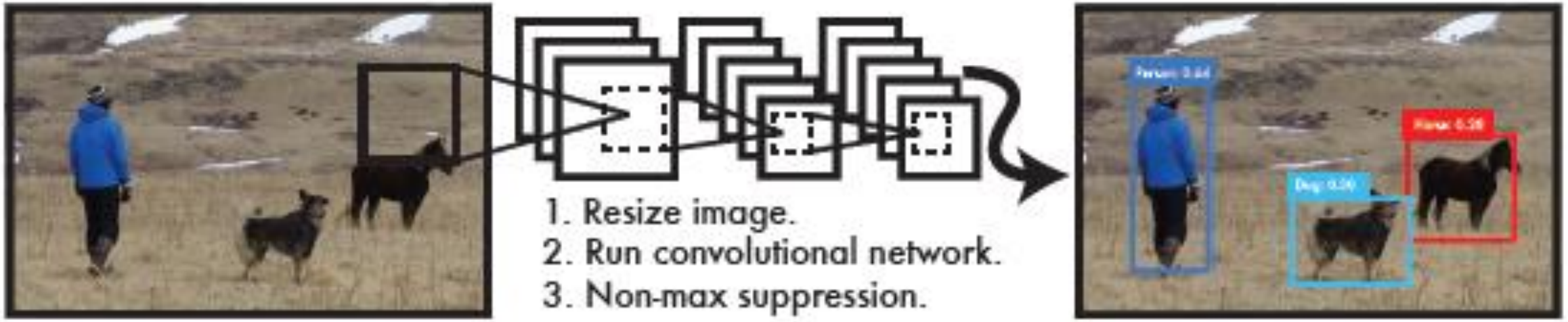
Introduction



YOLO 모델 훈련 과정

1. Resize Image – YOLOv1 기준으로 448x448x3으로 크기 재조정.
2. Run Convolutional Network – 입력 이미지 전체에 대해서 CNN을 통과 시켜서 Detection 목적에 맞게 Feature map 추출.
3. Non-max suppression – 1+2 과정에서 객체를 탐지하는데 책임이 있다고 판단된 Box들 중에서 가장 유효한 Box들만 남기고 제거.

Introduction



YOLO 모델 구조 장점

1. 여러 요소로 혼재되어 있지 않고 작업 파이프 라인이 통일되어 있어서 속도가 빠름.
2. 전체 이미지에 대해 추론하므로 Fast R-CNN과 같이 이미지 지역에 대해서 추론 하는 것보다 객체가 없는 배경에 대한 오류가 적다.
3. 객체의 일반적인 특징을 잘 학습한다.

Unified Detection

- 마지막으로 출력될 텐서를 S x S의 격자로 나누고 어떤 객체의 중심점이 격자 Cell 안에 들어 있다면 그 Cell이 객체를 탐지하는 책임을 진다.
- 각 Cell에서는 B개의 바운딩 박스에 대한 중심좌표, 넓이, 높이, Confidence Score를 예측한다. Confident Score는 Cell 안에 박스 안에 객체가 있을 법한 정도 * 얼마나 객체에 잘 맞춰져 있는가로 계산. 객체가 Cell 안에 없다면 Confident Score는 0

$$\Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}}$$

- 각 Cell에서는 다음과 같은 C개의 클래스에 대한 조건부 확률을 예측한다.

$$\Pr(\text{Class}_i|\text{Object})$$

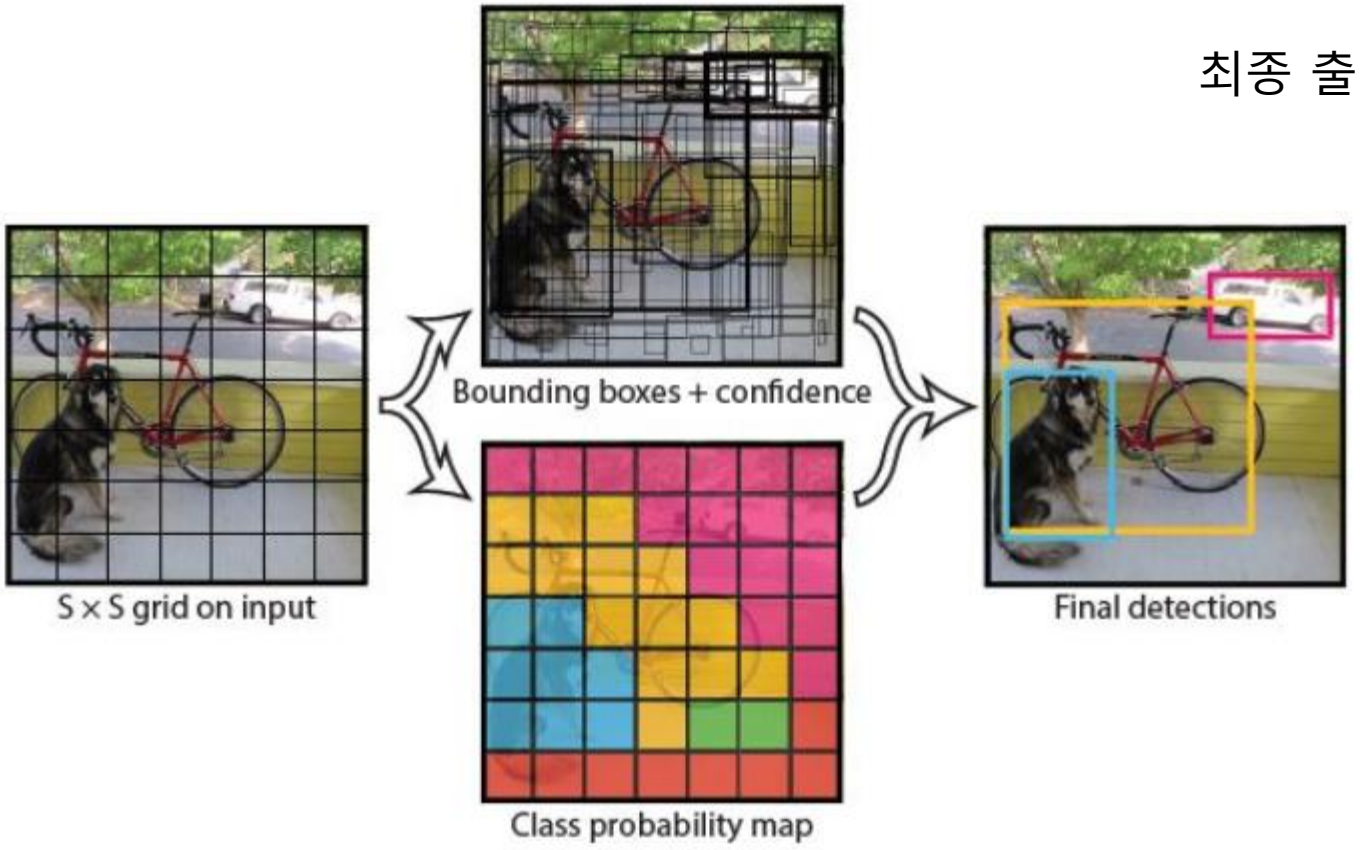
- 테스트 수행시에 클래스에 대한 조건부 확률과 Cell의 각 박스에 대한 Confident Score를 곱함.

$$\Pr(\text{Class}_i|\text{Object}) * \Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}} = \Pr(\text{Class}_i) * \text{IOU}_{\text{pred}}^{\text{truth}}$$

P(Object) is in {0, 1}, if the Predict_Box has zero IOU with Gt_Box, set P(Object) to 0, indicated background, otherwise, set it to 1.

Unified Detection

네트워크의 마지막에
추출될 텐서



PASCAL VOC 데이터셋으로 테스트할 때는

7x7 격자에 각 격자마다 2개의 바운딩 박스에 대해

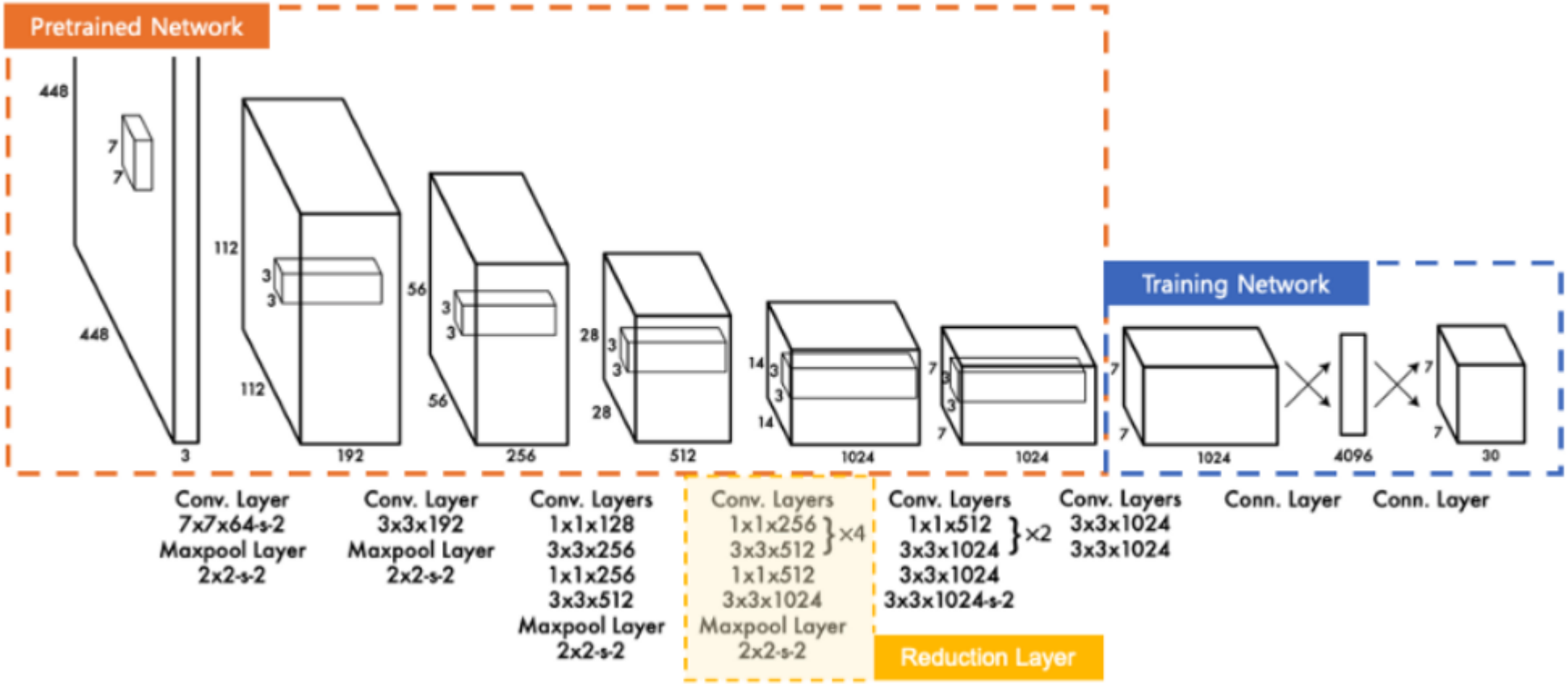
예측을 진행했고 20개의 클래스가 있었으므로

최종 출력은 7x7x(2x5 + 20)크기의 Tensor가 된다.

Unified Detection

출처: https://medium.com/curg/you-only-look-once-%EB%8B%A4-%EB%8B%A8%EC%A7%80-%ED%95%9C-%EB%B2%88%EB%A7%8C-%EB%B3%B4%EC%95%98%EC%9D%84-%EB%BF%90%EC%9D%B4%EB%9D%BC%EA%B5%AC-bddc8e6238e2

Network Design



주황색 테두리 – · GoogLeNet을 이용하여 ImageNet으로 Pre training한 네트워크를 Fine tuning함.

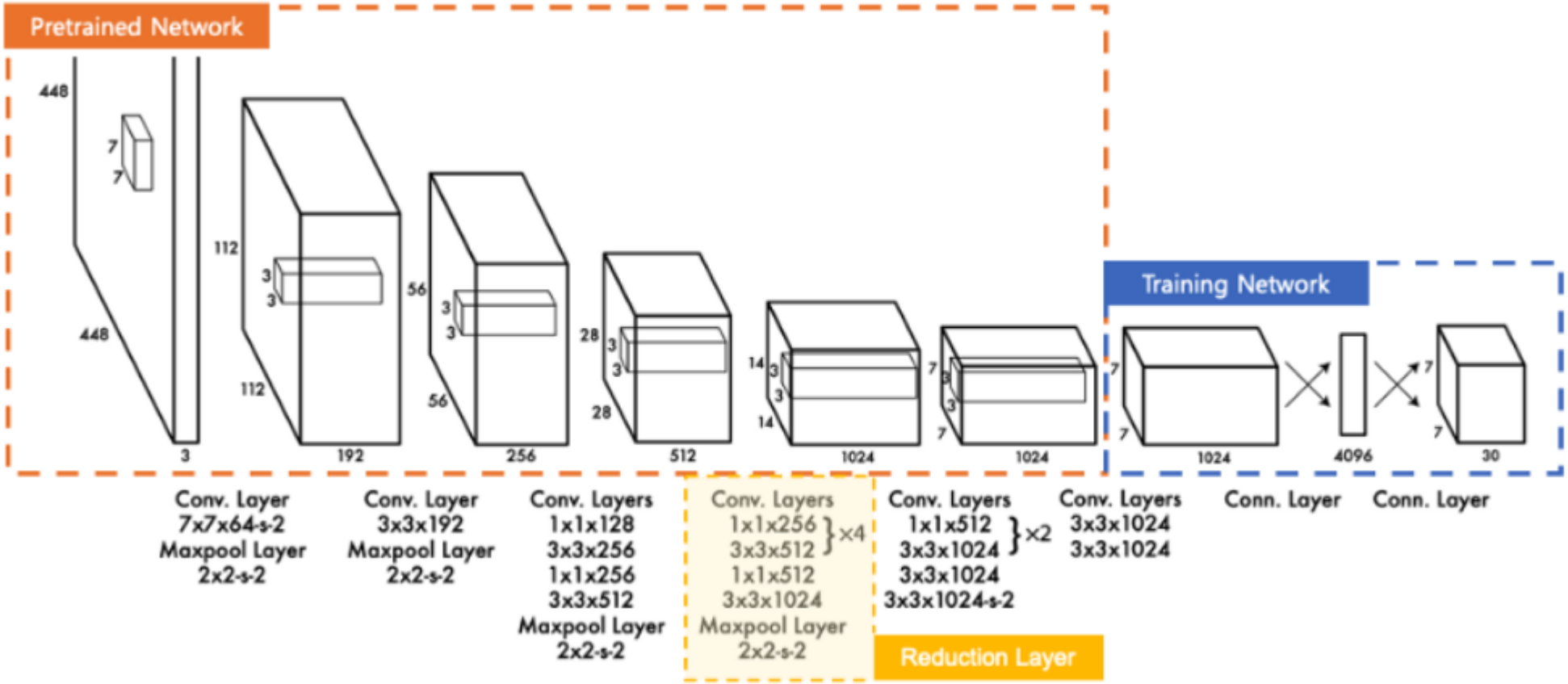
· 총 20개의 Conv Layer

· ImageNet은 224x224지만 여기서는 448x448로 이미지 크기를 키움.

Unified Detection

출처: https://medium.com/curg/you-only-look-once-%EB%8B%A4-%EB%8B%A8%EC%A7%80-%ED%95%9C-%EB%B2%88%EB%A7%8C-%EB%B3%B4%EC%95%98%EC%9D%84-%EB%BF%90%EC%9D%B4%EB%9D%BC%EA%B5%AC-bddc8e6238e2

Network Design

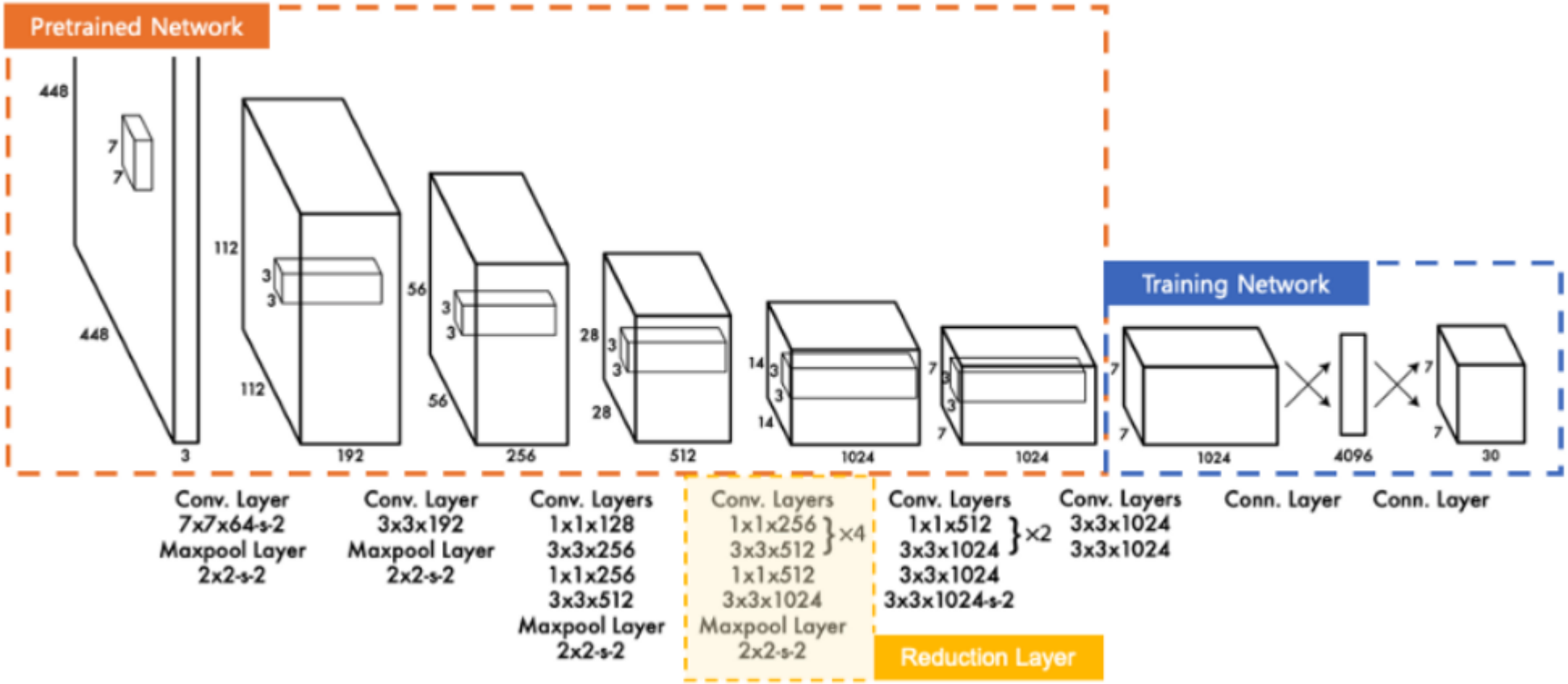


노란색 테두리 – 더 많은 이미지의 특징을 학습시키기 위해서 GoogLeNetD의 Inception 모듈 대신에 1x1 컨볼루션(Network In Network), 3x3 컨볼루션(VGG) 적용하여 연산 량을 감소 시키면서 층을 깊게 쌓음.

Unified Detection

출처: https://medium.com/curg/you-only-look-once-%EB%8B%A4-%EB%8B%A8%EC%A7%80-%ED%95%9C-%EB%B2%88%EB%A7%8C-%EB%B3%B4%EA%B8%B0-%EC%9D%84-%EB%BF%90%EC%9D%B4%EB%9D%BC%EA%B5%AC-bddc8e6238e2

Network Design

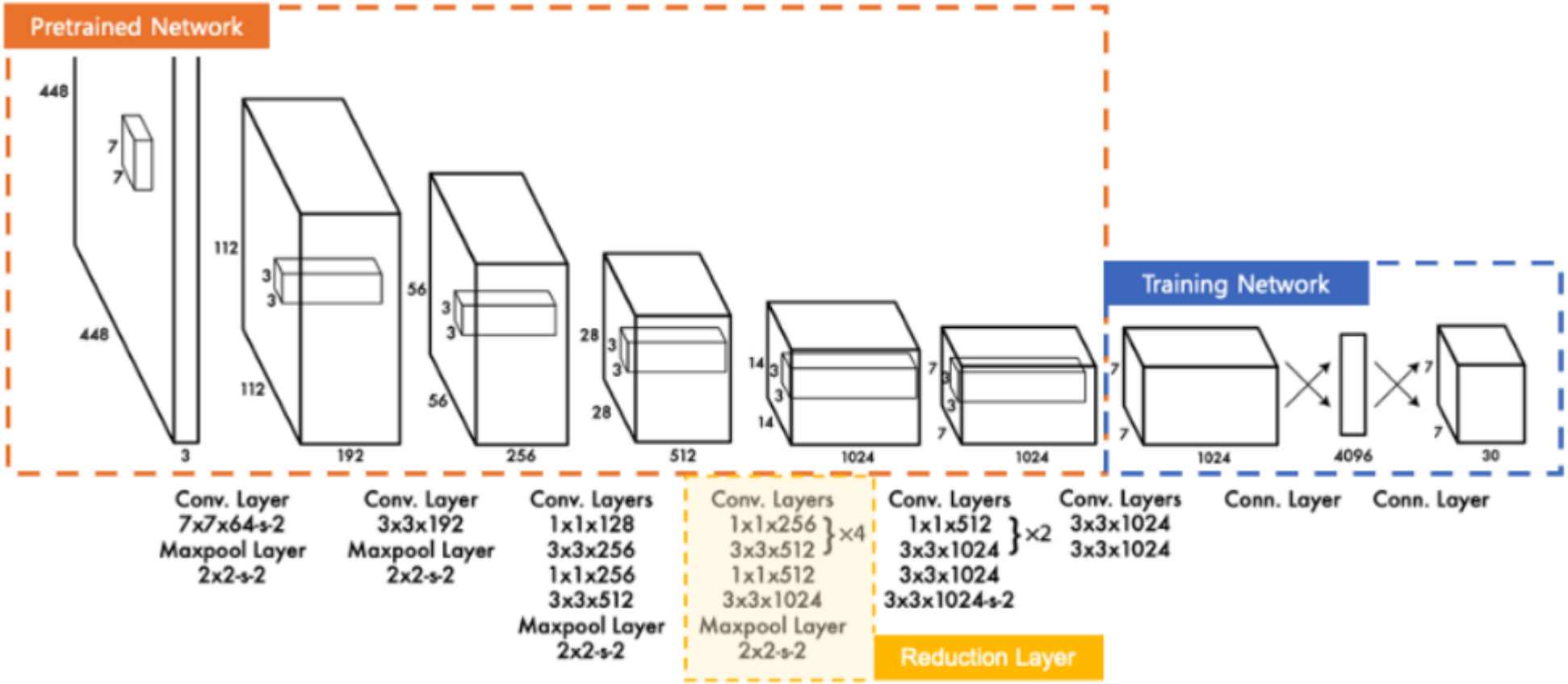


파란색 테두리 – 완전 연결 계층으로 Feature map을 통과시켜 이미지 정보가 들어 있는 모델 파라미터를 모두 전 결합하고(CNN에서는 주요 특징만 추출되므로) 앞에서 언급한 예측 값이 들어있는 형태로 출력 Tensor를 구성.

Unified Detection

출처: https://medium.com/curg/you-only-look-once-%EB%8B%A4-%EB%8B%A8%EC%A7%80-%ED%95%9C-%EB%B2%88%EB%A7%8C-%EB%B3%B4%EC%95%98%EC%9D%84-%EB%BF%90%EC%9D%B4%EB%9D%BC%EA%B5%AC-bddc8e6238e2

Network Design

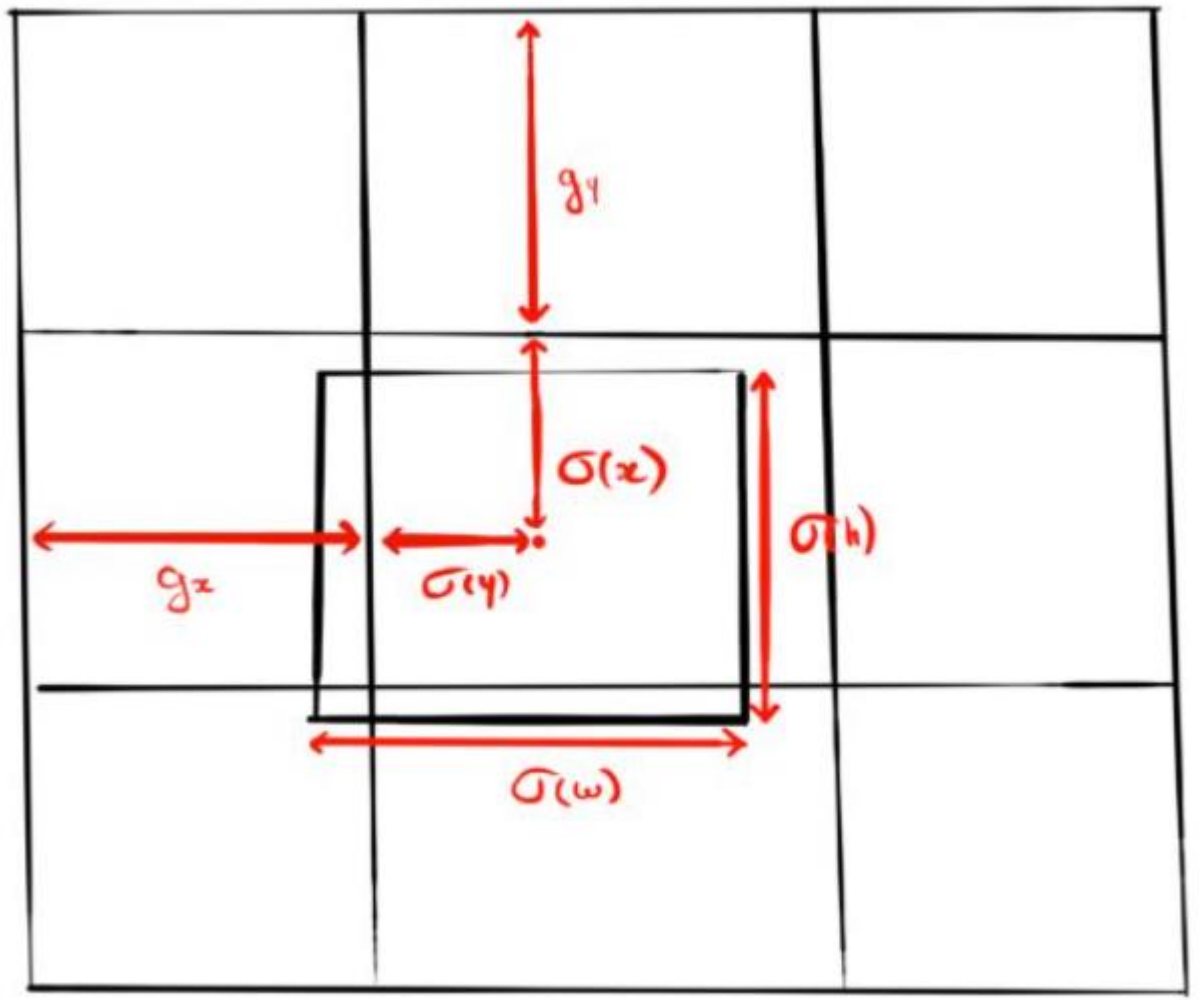


- 최종 출력 계층에는 선형 활성화 함수 적용, 다른 모든 계층에는 Leaky ReLU 적용

$$\phi(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.1x, & \text{otherwise} \end{cases}$$

Unified Detection

Network Design



- 최종 출력 계층에서는 바운딩 박스의 넓이와 높이를 이미지의 넓이와 높이로 정규화, 바운딩 박스의 중심 좌표 또한 Cell의 테두리로부터의 오프셋으로 만들어 0과 1 사이의 값이 되게 함.

Unified Detection

Training

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

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

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

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

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

- 손실 함수는 Sum of Squared Errors의 형태.
- 오른쪽 식의 1, 2번째 행은 좌표와 넓이, 높이에 관한 손실 값,
  나머지는 클래스 분류와 관련된 손실 값.
- 객체가 있는 박스보다 없는 박스가 훨씬 많아서 객체가 있는
  박스의 그래디언트를 억제 하므로 객체가 있는 경우의 손실의
  가중을 더 높이기 위해서 λcoord λnoobj를 각각 5, 0.5로 설정.
- 큰 박스의 넓이와 관련된 에러가 작은 박스의 넓이와 관련된 에러와 너무 크게 차이가 나지 않도록 제곱근으로
  손실 계산.

Unified Detection

Training

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

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

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

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

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

- $\mathbb{1}_{ij}^{\text{obj}}$ 는 Cell i의 j번째 박스가 객체에 대한 책임을 질 경우 1,

  아니면 0. 여기서 책임이라는 것은 Cell 안에 객체의 중심 좌표가

  있으며 그 중에서도 객체와 IoU가 가장 높은 바운딩 박스를 의미함.

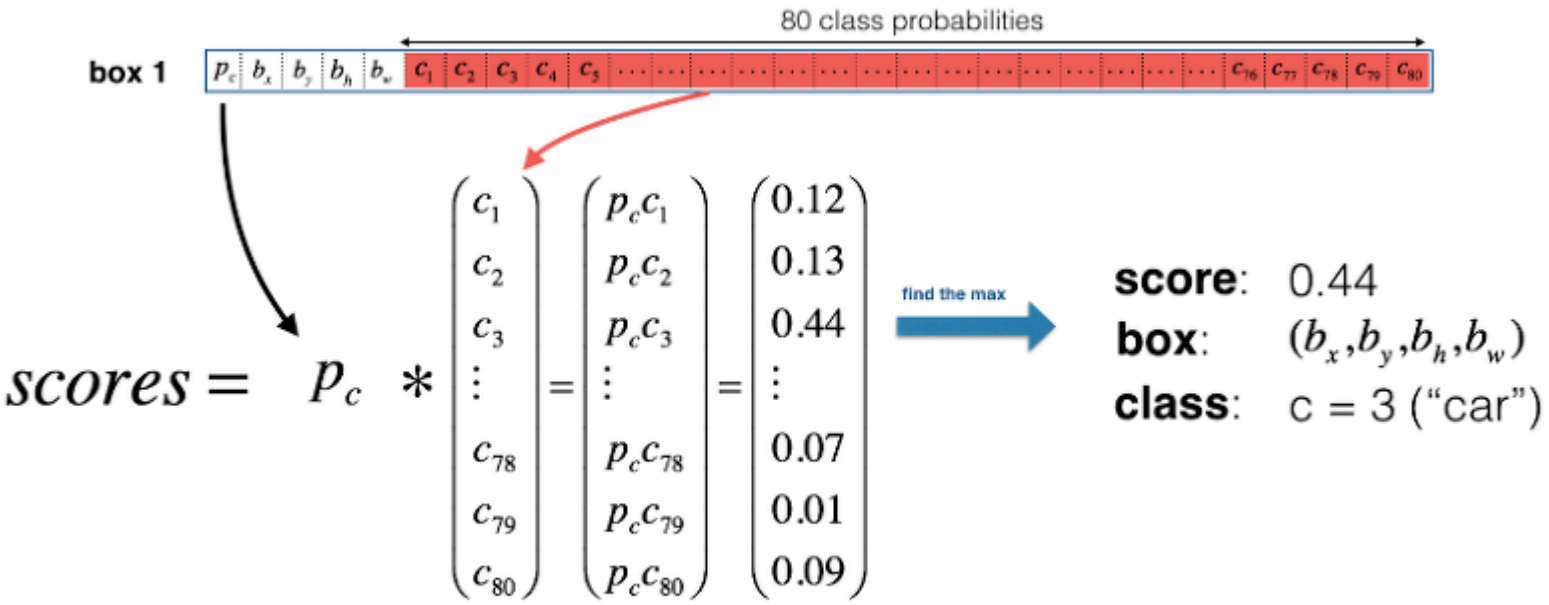
  $\mathbb{1}_{ij}^{\text{noobj}}$ 는 객체가 없는(즉, 배경)과 관련된 클래스 손실.

  $\mathbb{1}_{i}^{\text{obj}}$ 는 객체가 Cell i에 있을 때와 관련된 클래스 손실.

Unified Detection

Inference

Non-max Suppression을 통해서 Threshold보다 낮은 Class Confidence Score의 박스를 제거.



the box (b_x, b_y, b_h, b_w) has detected c = 3 ("car") with probability score: 0.44

출처: https://arclab.tistory.com/167

Unified Detection

Limitations of YOLO

- B개 이상의 객체가 Cell에 몰려 있을 경우 탐지 성능이 제한됨.
- 객체가 특이한 모양(종횡비)을 갖을 경우 바운딩 박스 정보에 대한 예측에 어려움을 겪음.
- Finer한 Information을 이용하지 않기 때문에 정확한 위치 추정이 어려울 수 있음.
- 큰 박스와 작은 박스의 에러를 동등하게 취급하므로 작은 박스의 에러는 박스와 객체의 IoU에 큰 영향을 줌.