

2021 인공지능 소수전공

36차시: YOLO

2021.07.29 21:30~22:15

Seokhwan Yang

YOLO (You Only Live Once)

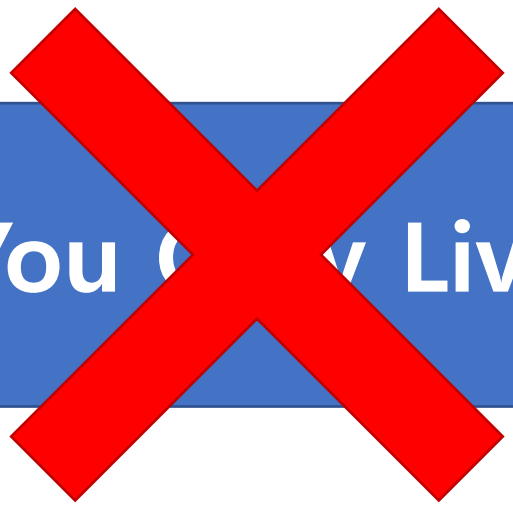


YOLO

YOLO (You Only Live Once)

A large, thick red 'X' is drawn over the text "YOLO (You Only Live Once)", indicating that this interpretation of the acronym is incorrect or discouraged.

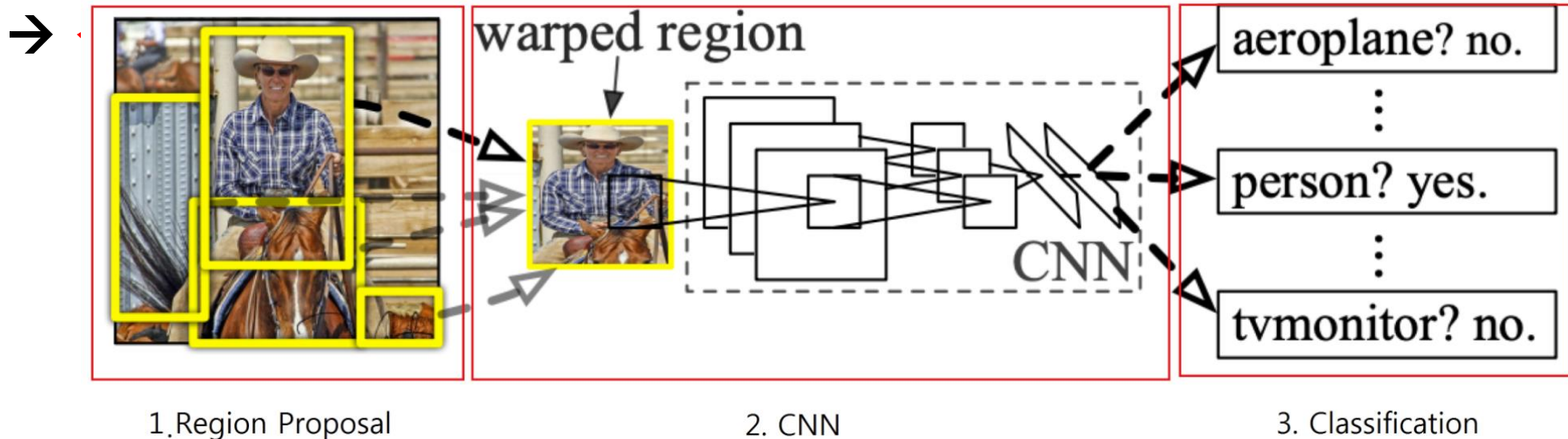
YOLO (You Only Live Once)

A large, thick red 'X' is drawn over the text "YOLO (You Only Live Once)", indicating that this definition is incorrect or being rejected.

YOLO (You Only **Look** Once)

YOLO (You Only Look Once)

- CNN과 R-CNN (Regions with CNN) :
 - CNN은 너무 많은 연산을 요구함 → 매우 느린 처리 속도
 - CNN 처리 이전에 인식하기 원하는 물체가 있을 가능성이 높은 후보 영역(Region)을 선택



YOLO (You Only Look Once)

- R-CNN의 프로세스

1. 이미지 입력
 2. 후보영역 추출
 3. CNN 특징 계산
 4. 영역 분류
- 다양한 방법이 제안되고 있음
 - 딥러닝의 영역이 아닌 데이터 정규화의 영역에 가까움 (CNN내부의 LCN과 비슷한 개념)
 - 후보영역 추출방법의 개선으로 Fast-RCNN / Faster-RCNN 등의 모델 등장

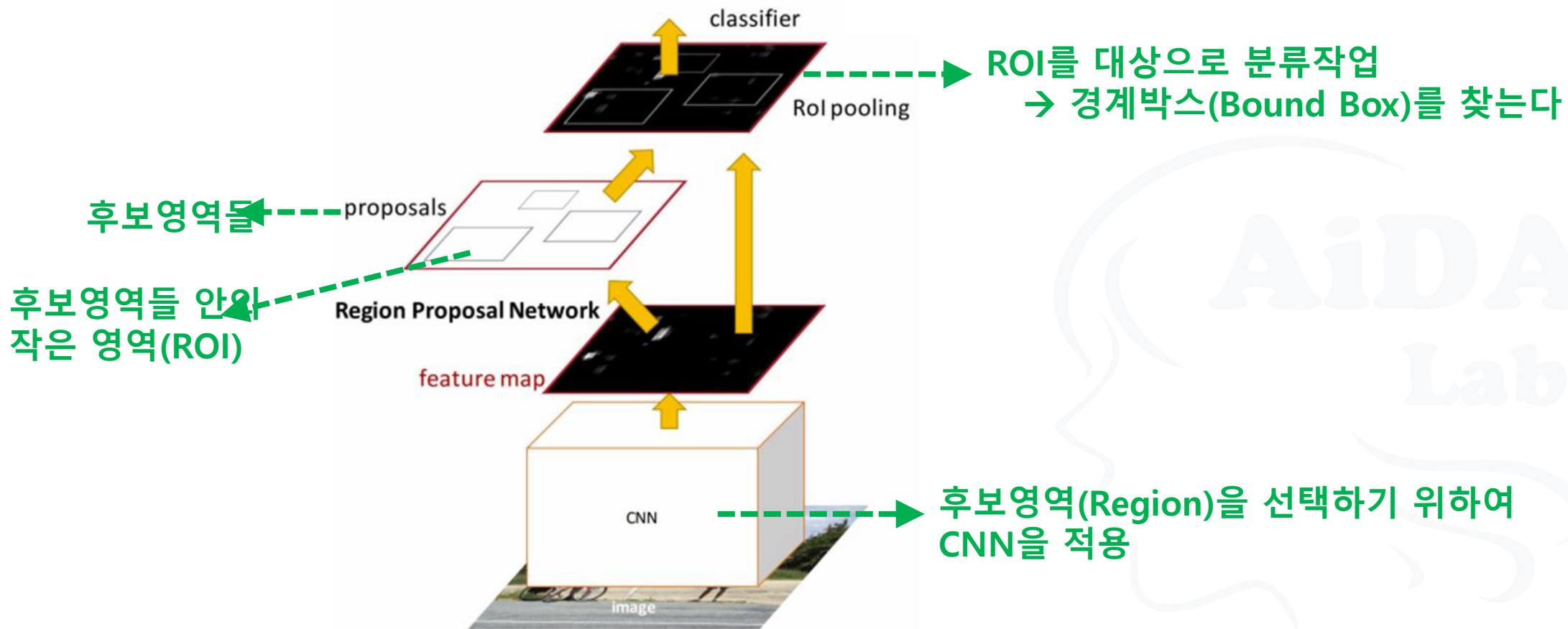
성능비교

CNN < R-CNN < Fast-RCNN < Faster-RCNN

그러나...
써먹기는 어려움.
모두 다 느려서...

YOLO (You Only Look Once)

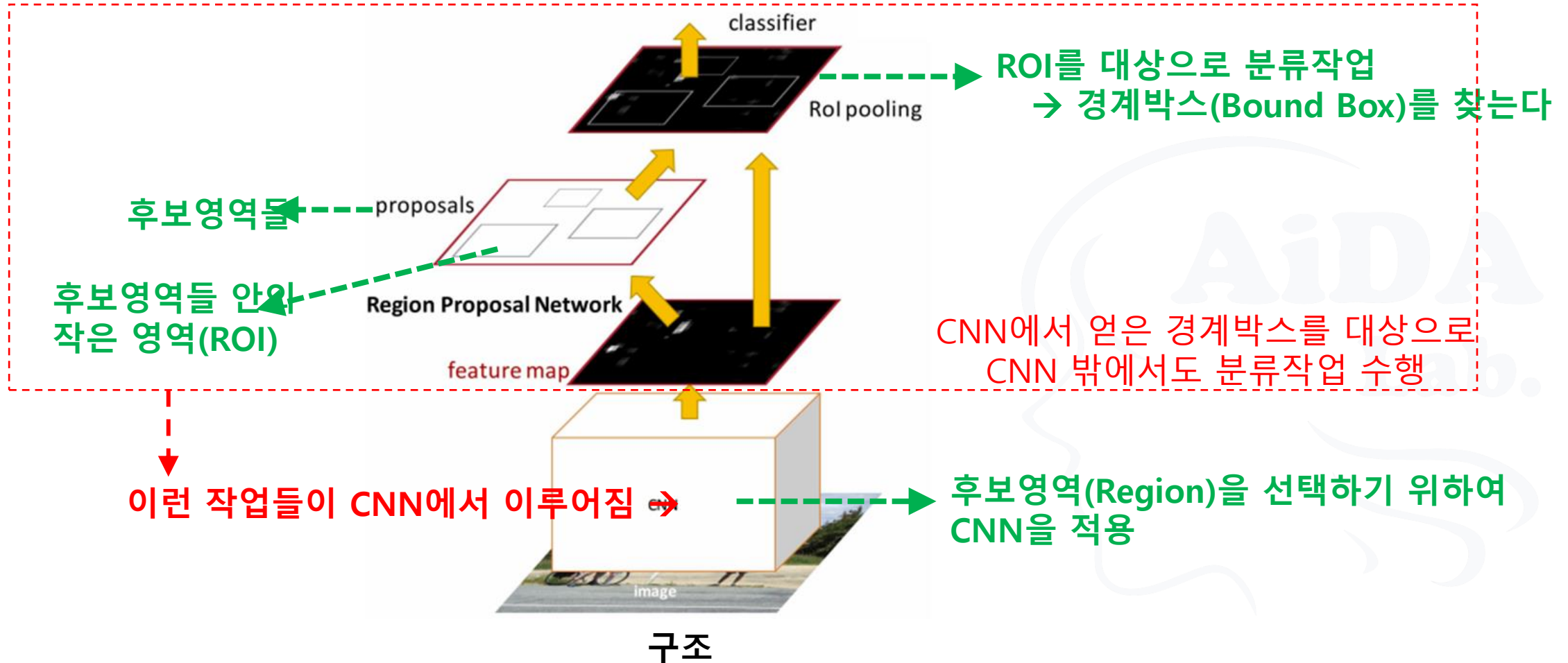
- Faster-RCNN



구조

YOLO (You Only Look Once)

- Faster-RCNN



YOLO (You Only Look Once)



- R-CNN 계열이 느린 이유
 - 제안하는 후보 영역의 수가 너무 많음
 - 후보 영역의 제안 과정에서도 부하가 큼

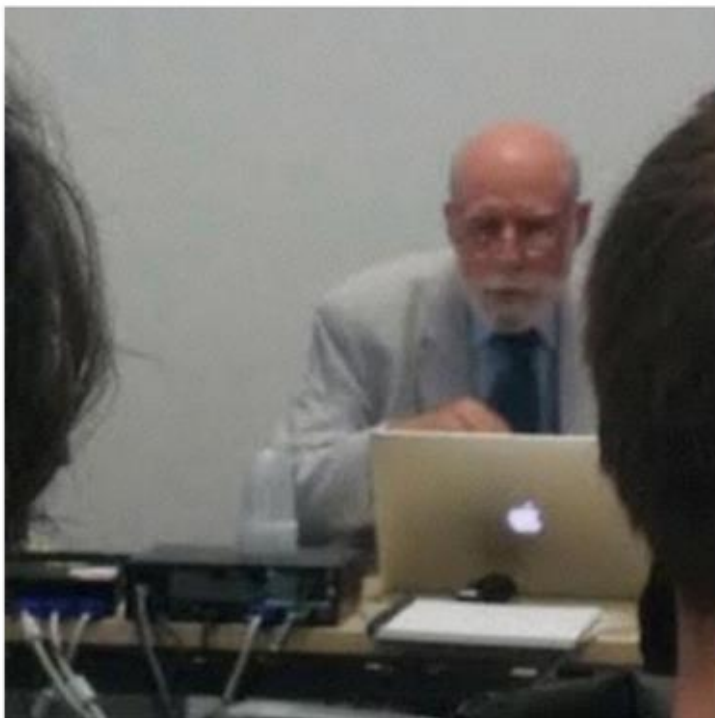


YOLO (You Only Look Once)

- 경계박스를 찾는 방법: Proposal 방식, Grid 방식

- Proposal 방식

Example: find the father of the internet

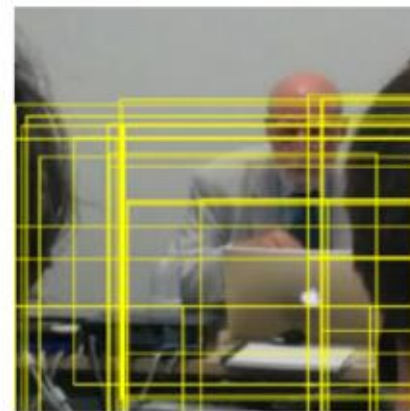


수많은
있을 것 같은 영역을
제안



Selective Search
2.24 seconds

Edge 정보를
바탕으로
Edge Box를 찾아서
영역 제안



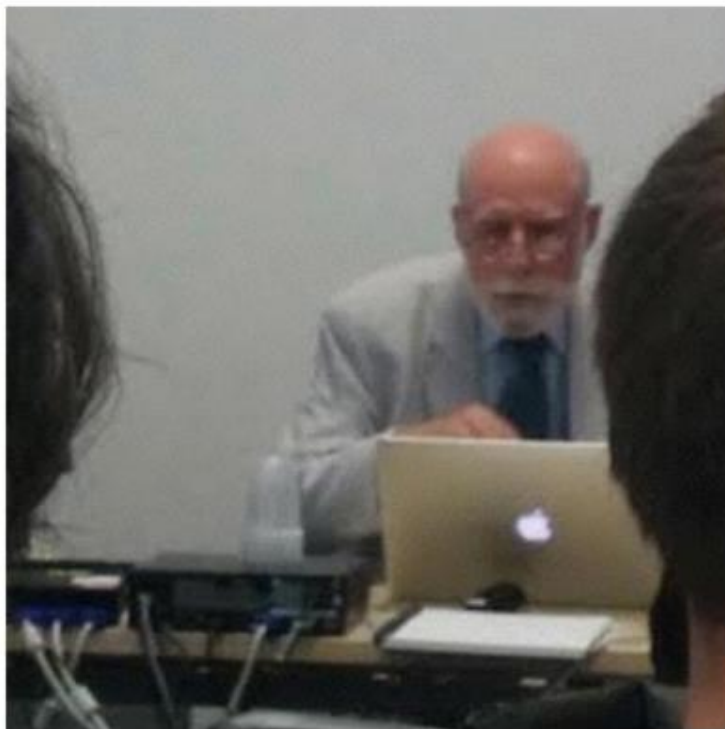
EdgeBoxes
0.38 seconds

훨씬 빠름

YOLO (You Only Look Once)

- 경계박스를 찾는 방법: Proposal 방식, Grid 방식
 - Grid 방식

Example: find the father of the internet



Cheaper Alternative: **grids**



Grid Cell의 수
=
제안 영역의 수

영역 제안을
위한
부하가 없음

YOLO (You Only Look Once)

- YOLO에서는
 - Grid 방식을 더욱 발전시켜서 사용함
 - Grid 방식 선택 이유
 - 후보 영역 제안 개수가 적으므로
 - 실시간성 확보가 용이하다



- **Proposal 방식, Grid 방식에서의 의문점**

- 찾고자 하는 물체(객체, Object)가 있을 것 같은 후보영역을 제안하는데...
- 저렇게 박스를 제안할 때, 무슨 방법으로 박스 내부에 물체가 있을 것이라고 판단, 선택하는가?
- 박스 내부에 물체가 있다고 어떻게 보장하는가?



- Proposal 방식, Grid 방식에서의 의문점

- 찾고자 하는 물체(객체, Object)가 있을 것 같은 후보영역을 제안하는데...
- 저렇게 박스를 제안할 때, 무슨 방법으로 박스 내부에 물체가 있을 것이라고 판단, 선택하는가?
- 박스 내부에 물체가 있다고 보장할 수 없다

보장 못함

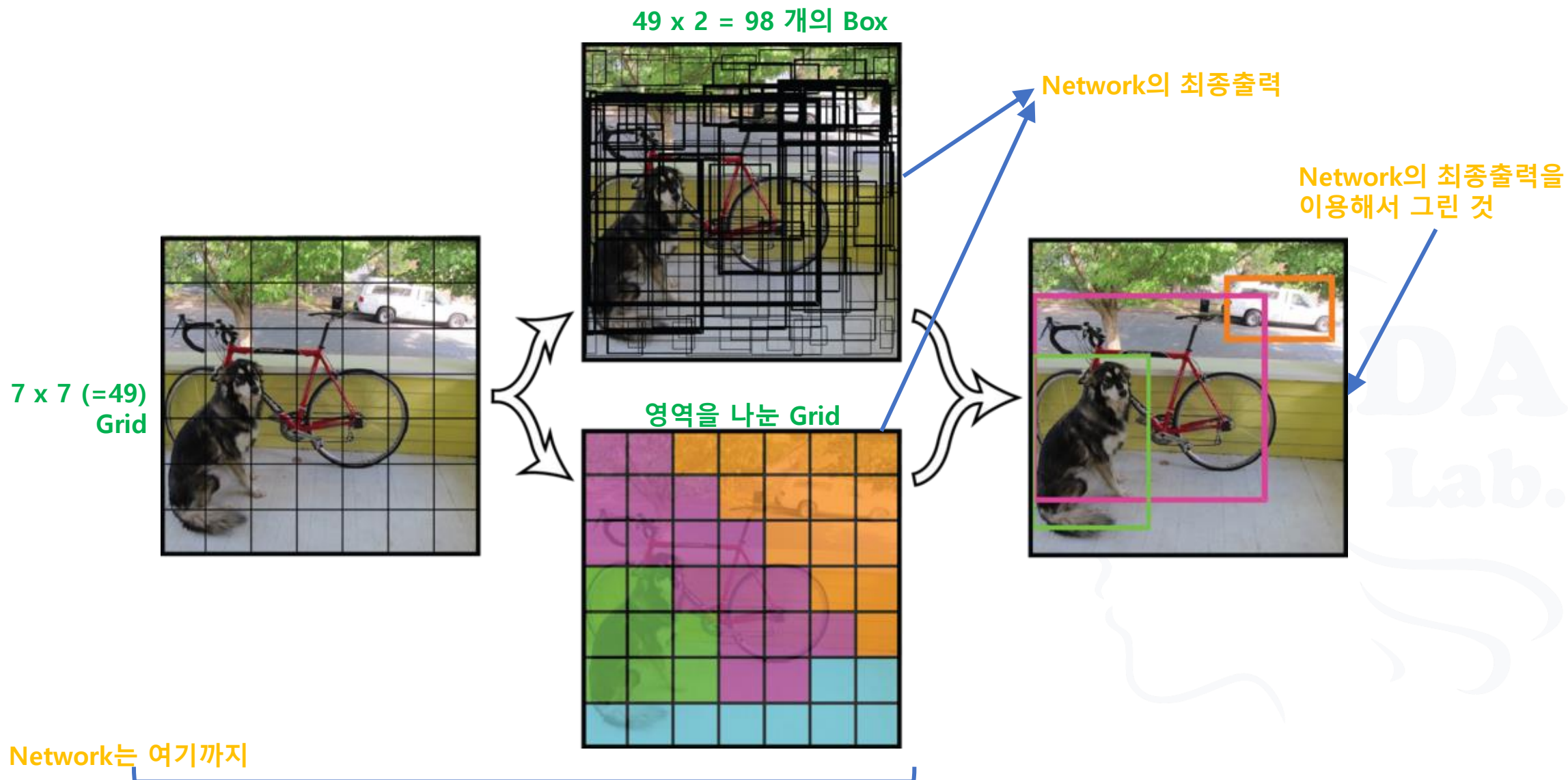
그냥 적당히 여러가지 정보, 기준을 잡고... 그냥 제안함
(영상의 Edge 정보, 임의의 박스 등등)

YOLO (You Only Look Once)



- YOLO 모델에서는
 - 말 그대로 영상을 1번만 읽음
 - 최종 출력단에서 경계박스 검색과 클래스 분류를 동시에 수행 → 속도가 빠르다
 - 즉, 하나의 네트워크가 → 동시에 → 특징 추출, 경계박스 만들기, 클래스 분류를 다 한다! (그래서 구조가 간단하고 빠르다)
 - 그런데 어떻게?

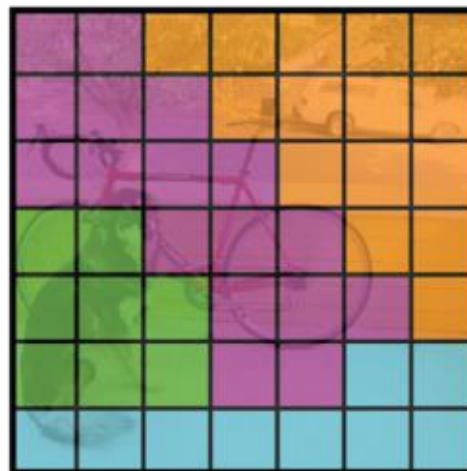
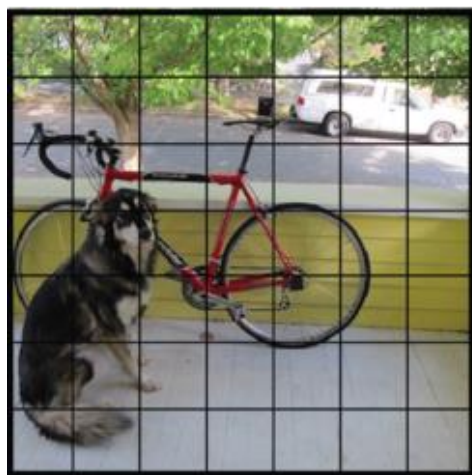
YOLO (You Only Look Once)



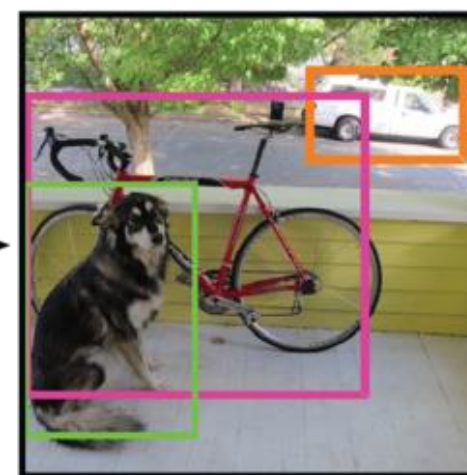
YOLO (You Only Look Once)

Grid의 각 Cell 당 2개씩의 임의의 Box를 그린다.
색깔 차이 등의 특징에 따라 Box내부에
뭔가 있어 보일수록 굵게 그린다.

특정 기준을 만족하는 정도를 측정,
임계값(0.5)에 미치지 못하면 삭제



굵은 Box를 제외하고 모든 Box를 지운다.
남은 Box 중에서 NMS 알고리즘을 통해서
Box를 선별하고 나머지 Box는 모두 지운다.



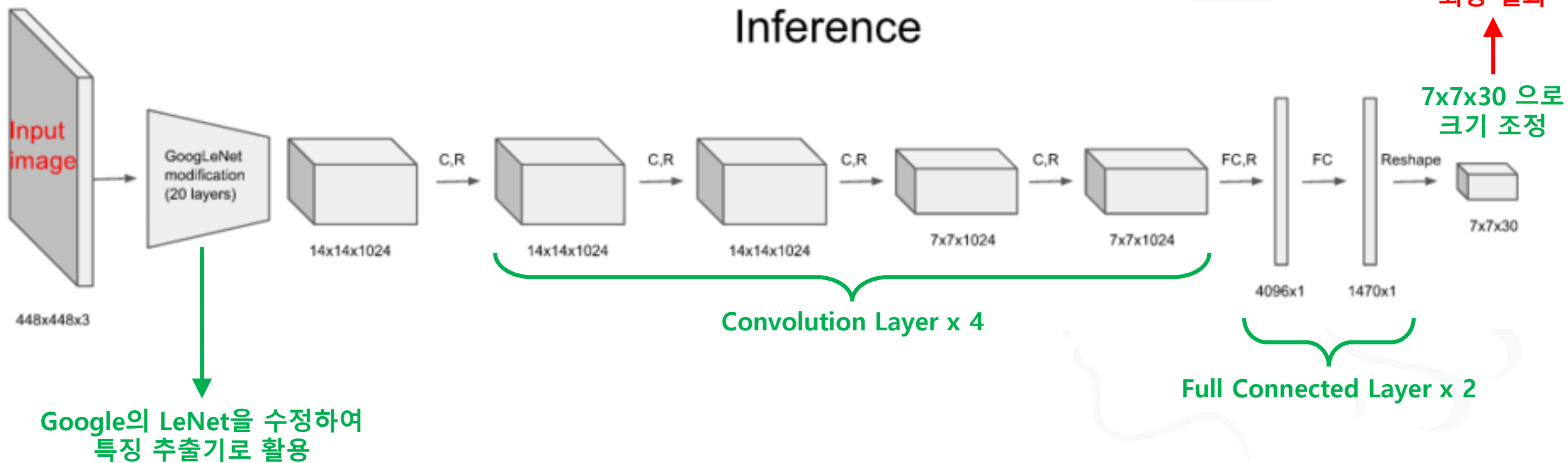
최종적으로
만들어낸
결과

남겨진 Box 내부의 이미지를 분류에
따라 색깔로 나누어 표시한다.

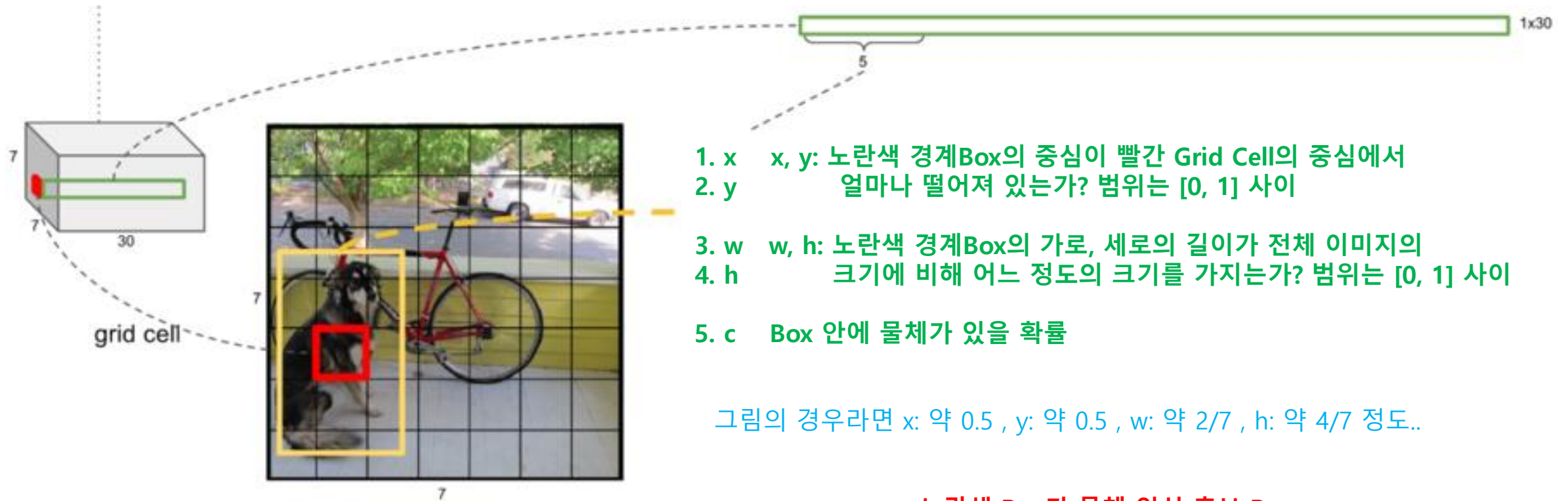
YOLO (You Only Look Once)

- YOLO의 네트워크 구조

원래 입력 이미지는
448x448 크기



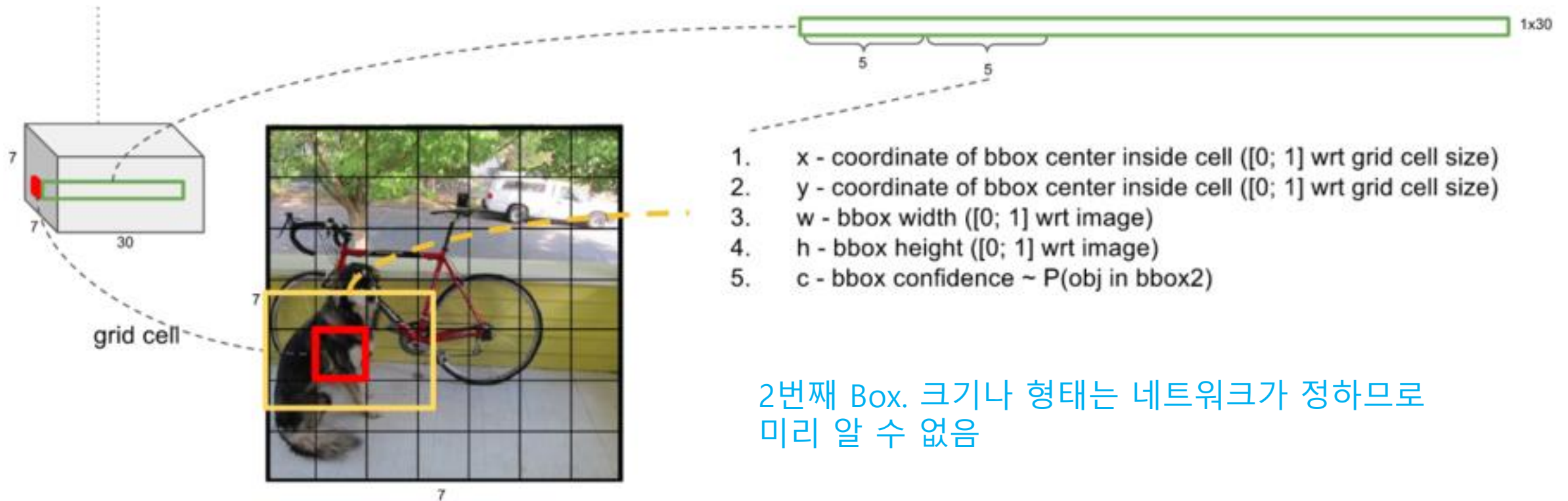
YOLO – 최종 결과에는 무엇이 들어있을까?



노란색 Box가 물체 인식 후보 Box

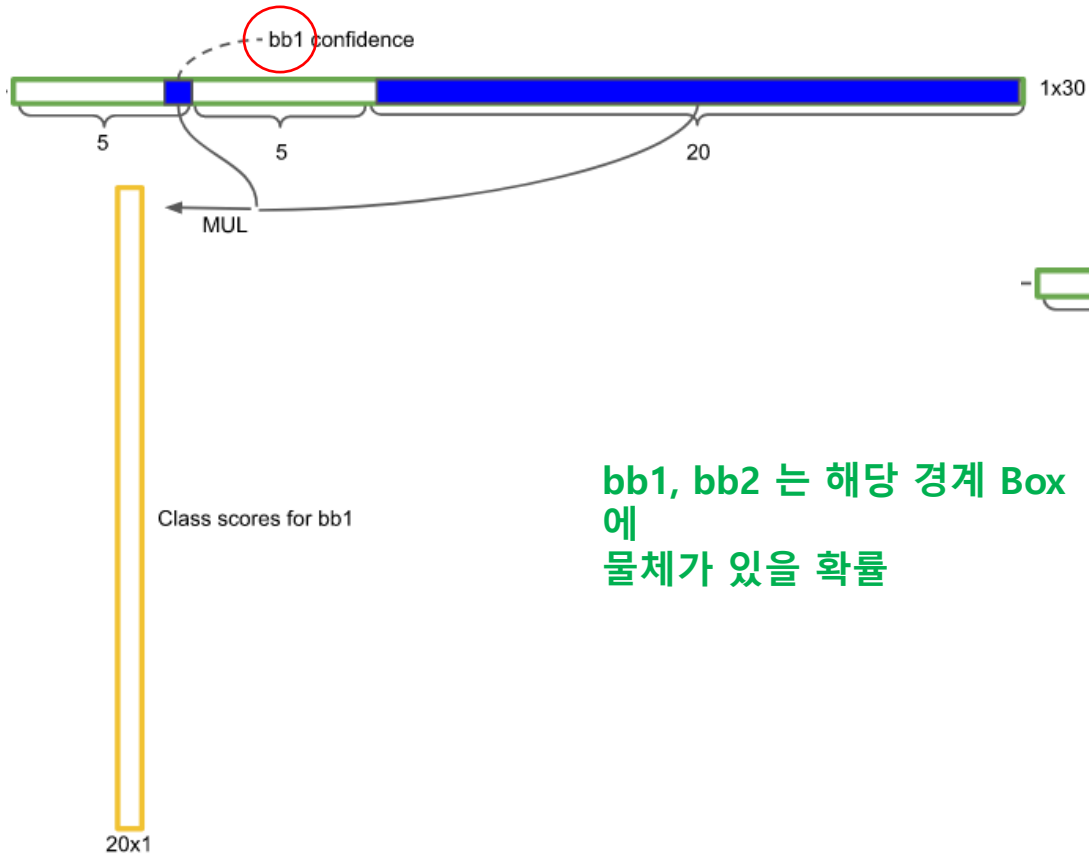
이런 Box를 2개씩 만든다.

YOLO – 최종 결과에는 무엇이 들어있을까?



2번째 Box. 크기나 형태는 네트워크가 정하므로
미리 알 수 없음

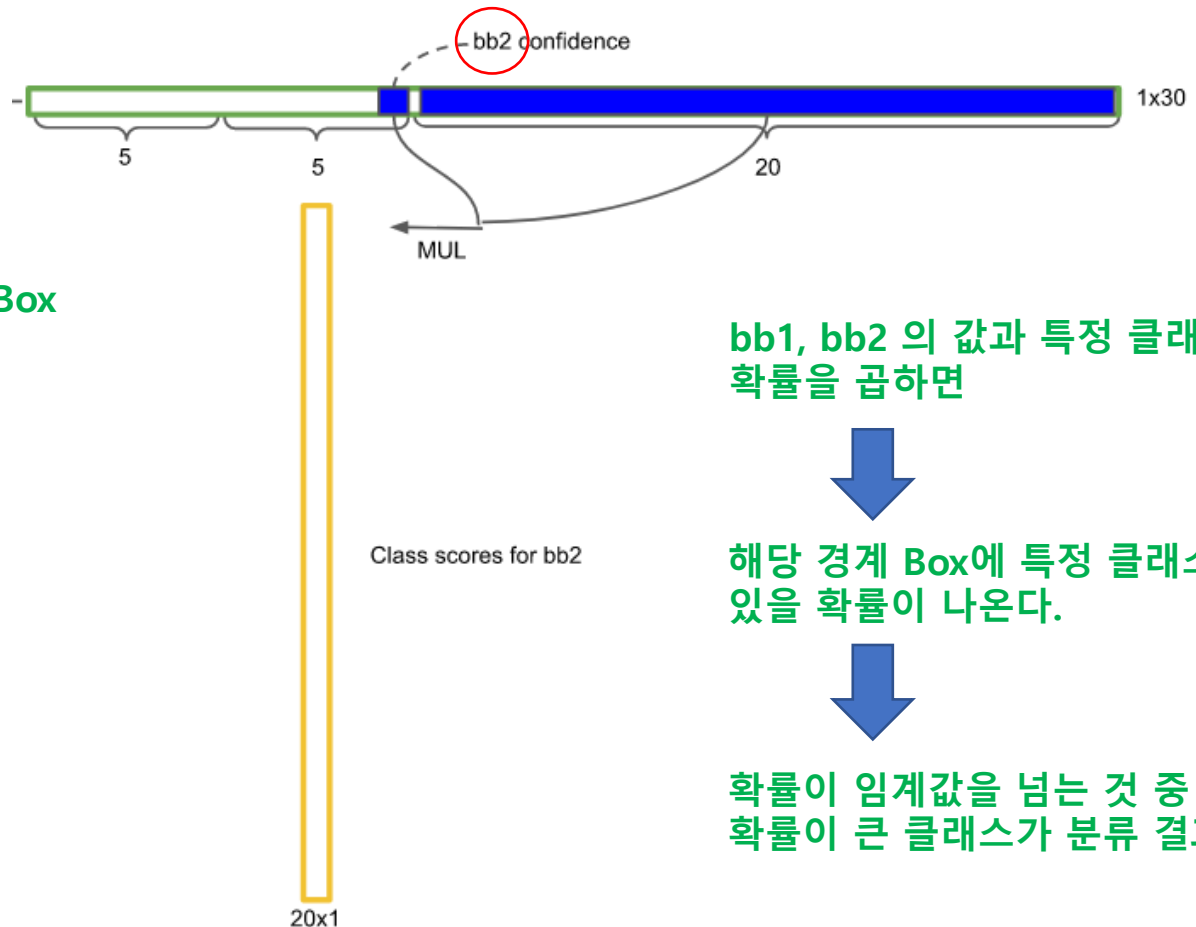
YOLO – 최종 결과에는 무엇이 들어있을까?



나머지 20채널에는 해당 Grid Cell에 물체가 있다면 어떤 클래스일까..에 대한 확률을 저장

bb1, bb2 는 해당 경계 Box 에 물체가 있을 확률

이런 작업이 모든 Cell에 적용됨



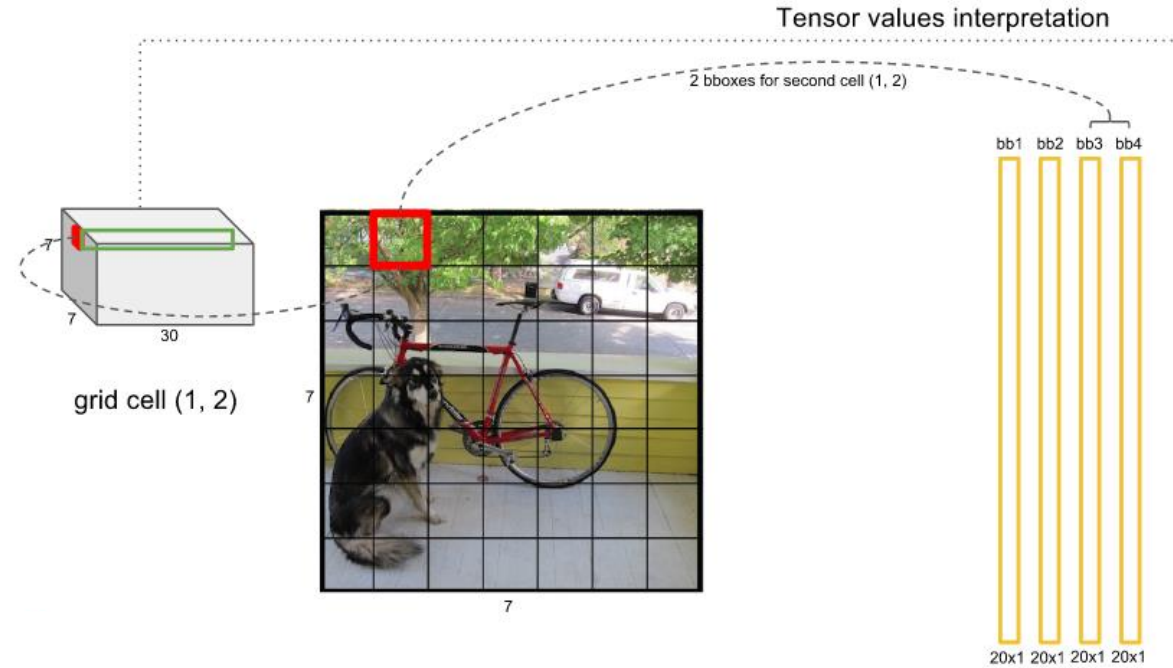
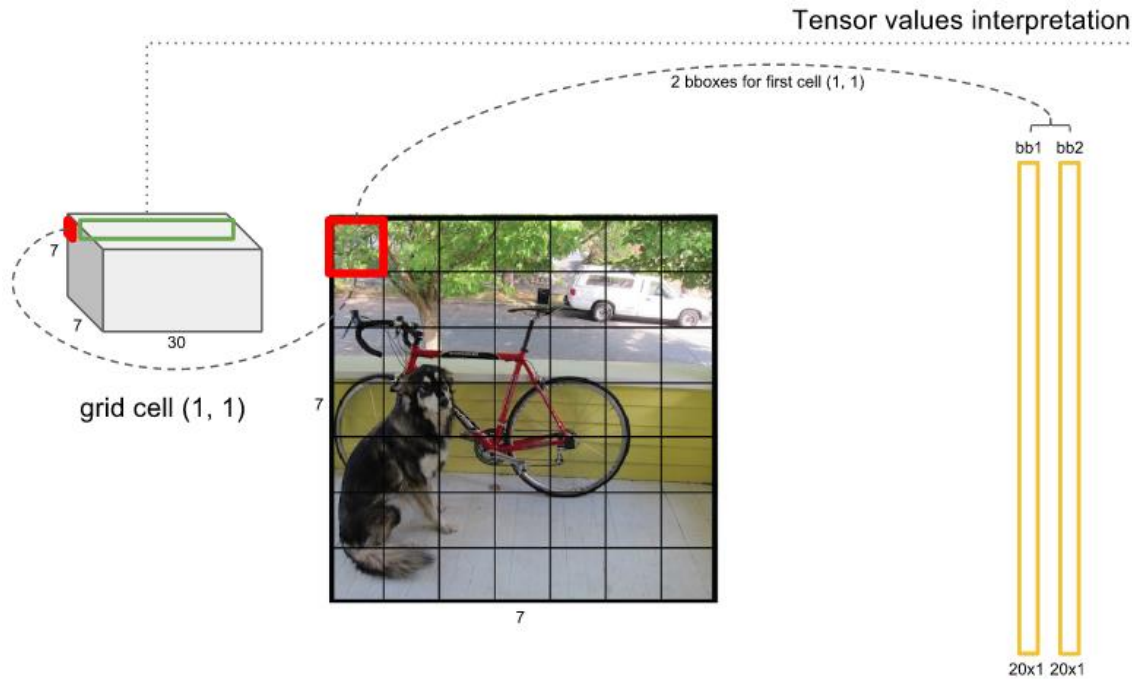
bb1, bb2 의 값과 특정 클래스일 확률을 곱하면

해당 경계 Box에 특정 클래스가 있을 확률이 나온다.

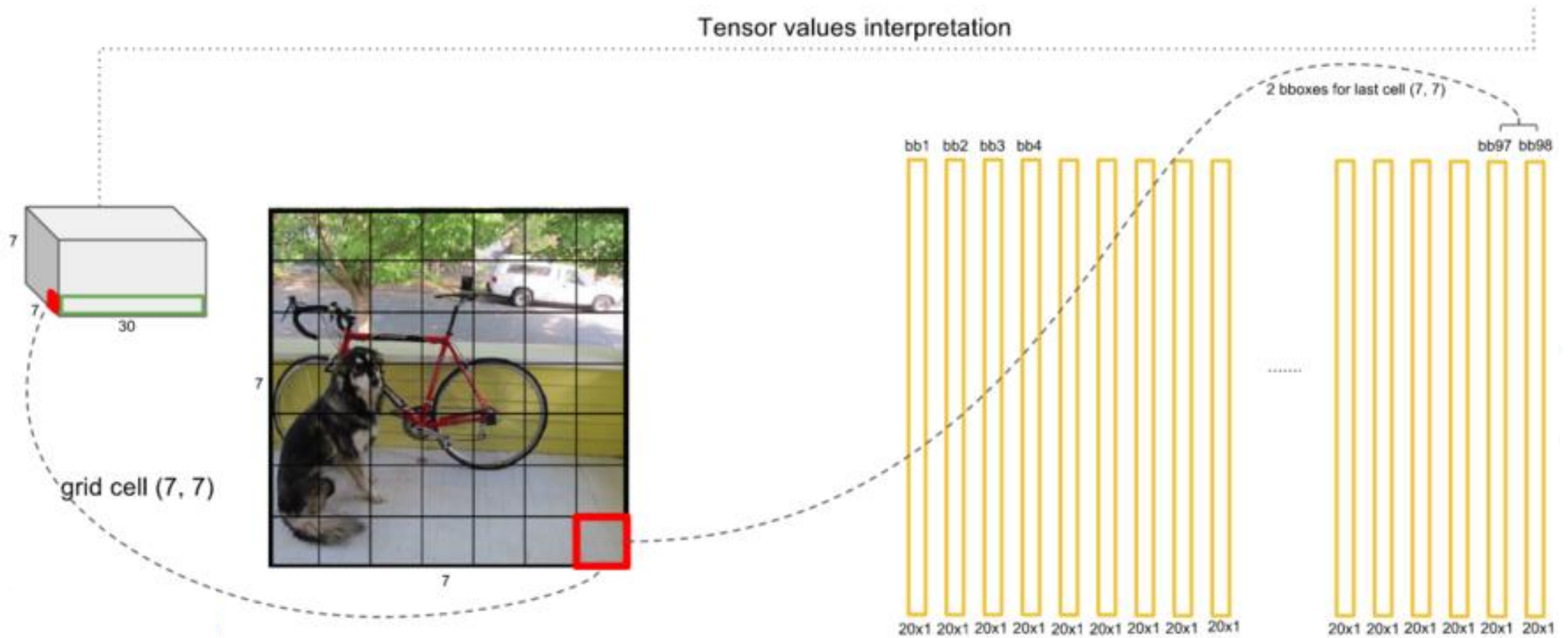
확률이 임계값을 넘는 것 중 가장 확률이 큰 클래스가 분류 결과

YOLO – 최종 결과에는 무엇이 들어있을까?

이런 작업이 모든 Cell에 적용하여 각각의 분류 점수를 획득

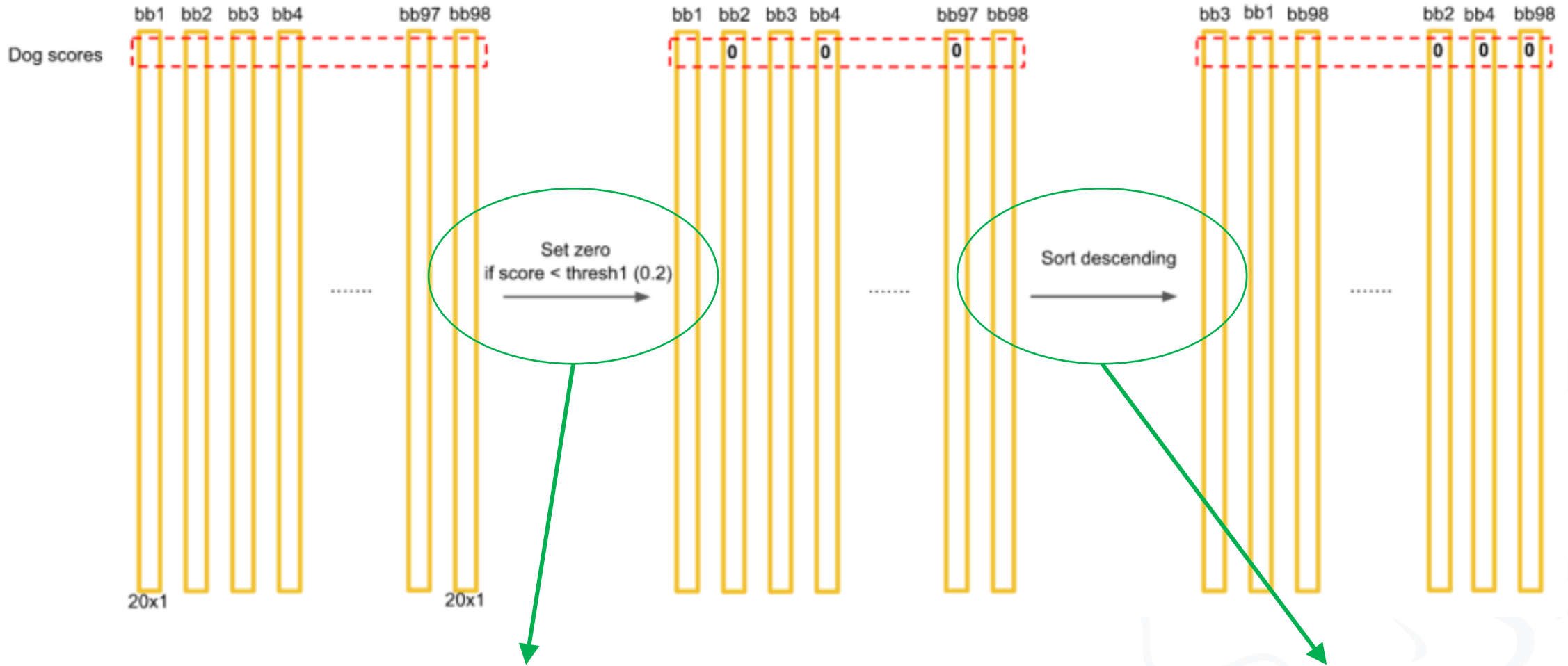


YOLO – 최종 결과에는 무엇이 들어있을까?



$7 \times 7 \times 2 = 98$ 개의 분류점수 결과 획득

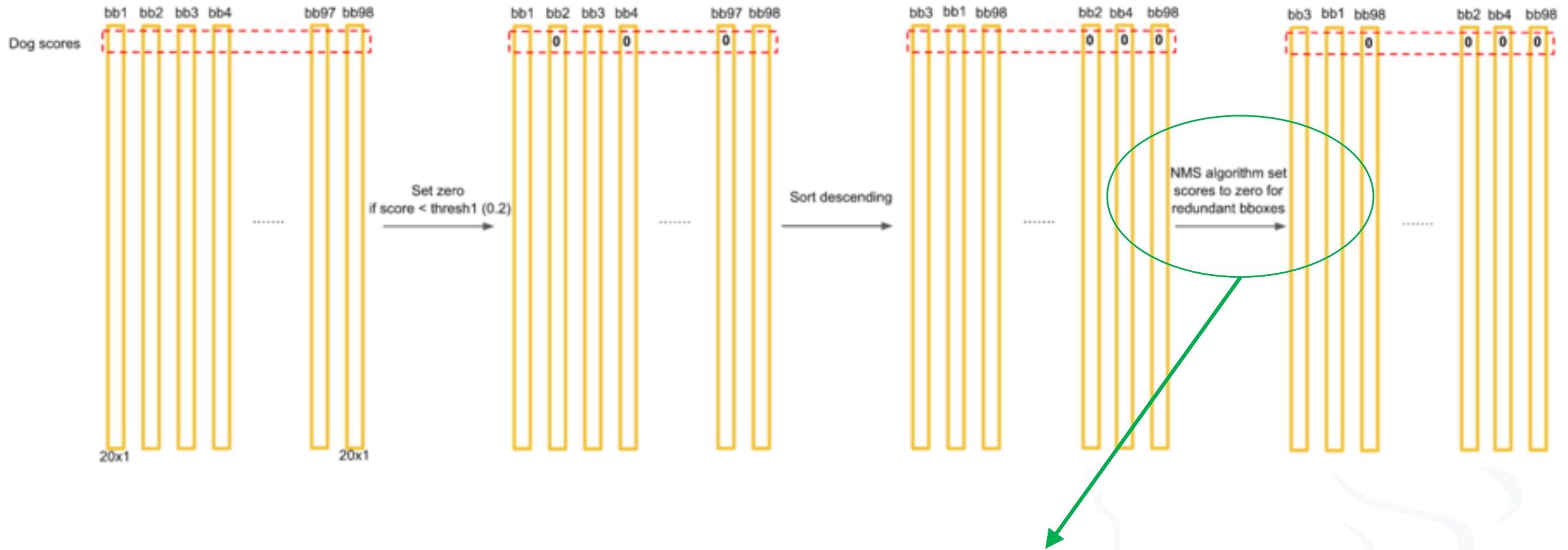
YOLO – 최종 결과에는 무엇이 들어있을까?



98개의 클래스 분류 점수 중에서 임계값(0.2) 보다 작은 것은 전부 0으로 채워준다.
(0.2보다 작으면 해당 클래스는 절대로 아닐 것이다!!! 라고 판단...)

각 클래스별로
크기 순으로 정렬한다

YOLO – 최종 결과에는 무엇이 들어있을까?



임계값 처리, 정렬이 끝나면 NMS 알고리즘을 통해서 최적의 값을 선출한다.
(하나의 물체에 대하여 경계Box가 겹치는 경우 신뢰도가 가장 높은 것을 남기고 나머지는 지움)

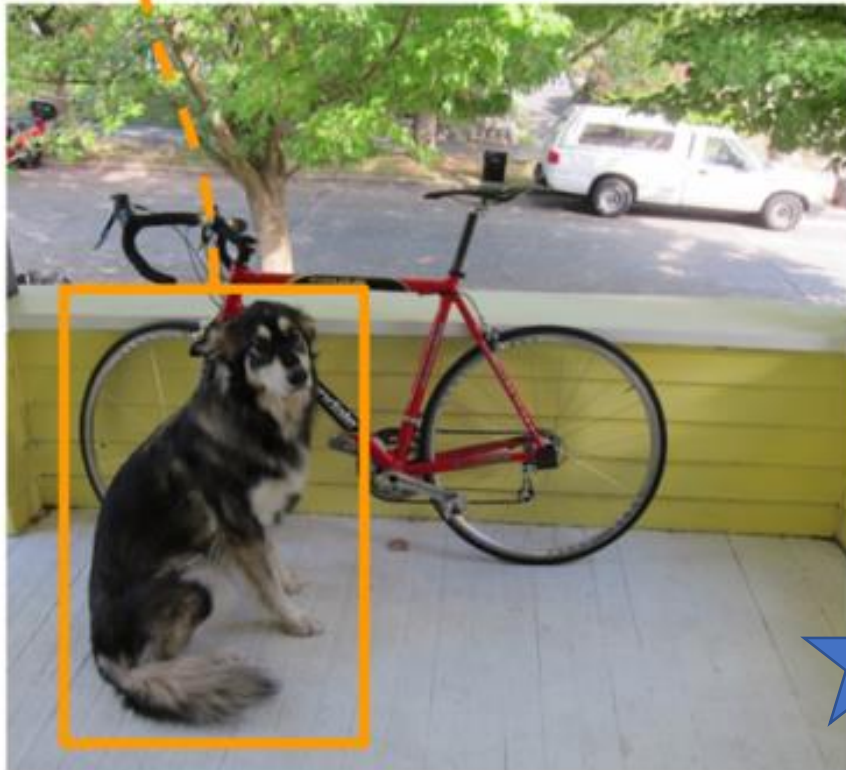
YOLO – 최종 결과에는 무엇이 들어있을까?

class (dog) scores for each bbox

| | bb47 | bb20 | bb15 | bb7 | | | | | | | | | | | | | | | | bb1 | bb4 | bb8 | bb98 |
|------------|------|------|------|-----|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|-----|-----|-----|------|
| class: dog | 0.5 | 0.3 | 0.2 | 0.1 | | | | | | | | | | | | | | | | 0 | 0 | 0 | 0 |

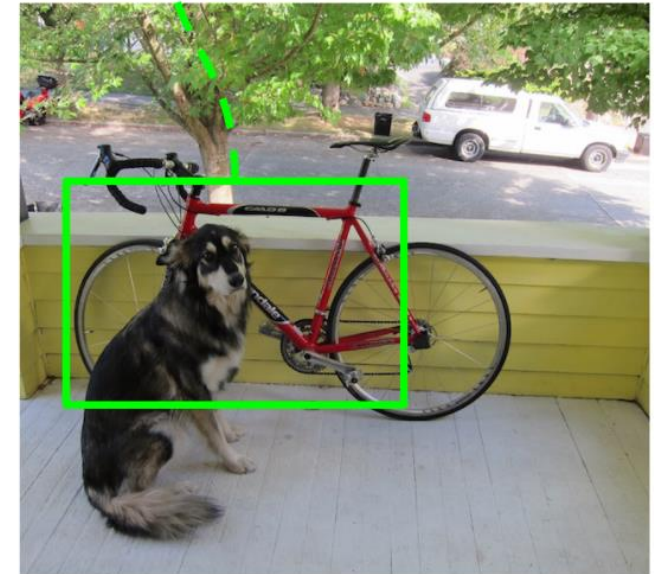
크기 순으로 정렬한 결과

확률이 제일 큰 Cell



| | bb47 | bb20 | bb15 | bb7 |
|------------|------|------|------|-----|
| class: dog | 0.5 | 0.3 | 0.2 | 0.1 |

확률이 두 번째로 큰 Cell



★ 제일 좋은 결과일
가능성이 높음

YOLO – 최종 결과에는 무엇이 들어있을까?



- 제일 가능성 높은 것만 남기고 지우는데...
 - 개가 2마리 있으면?
 - 위치가 겹치지 않고 동일한 클래스로 분류되면 큰 문제는 없음
 - 그러나 비슷한 위치에서 동일한 클래스들이 모여 있다면? .. 개떼
 - 2마리까지만 인식함 (경계Box가 2개씩이므로)
 - 단점!! → → → 점차 개선되어 가고 있음

