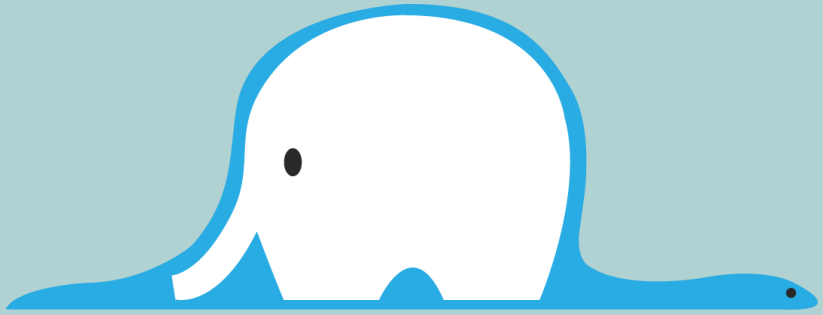
YOLO

Detection조 김지원

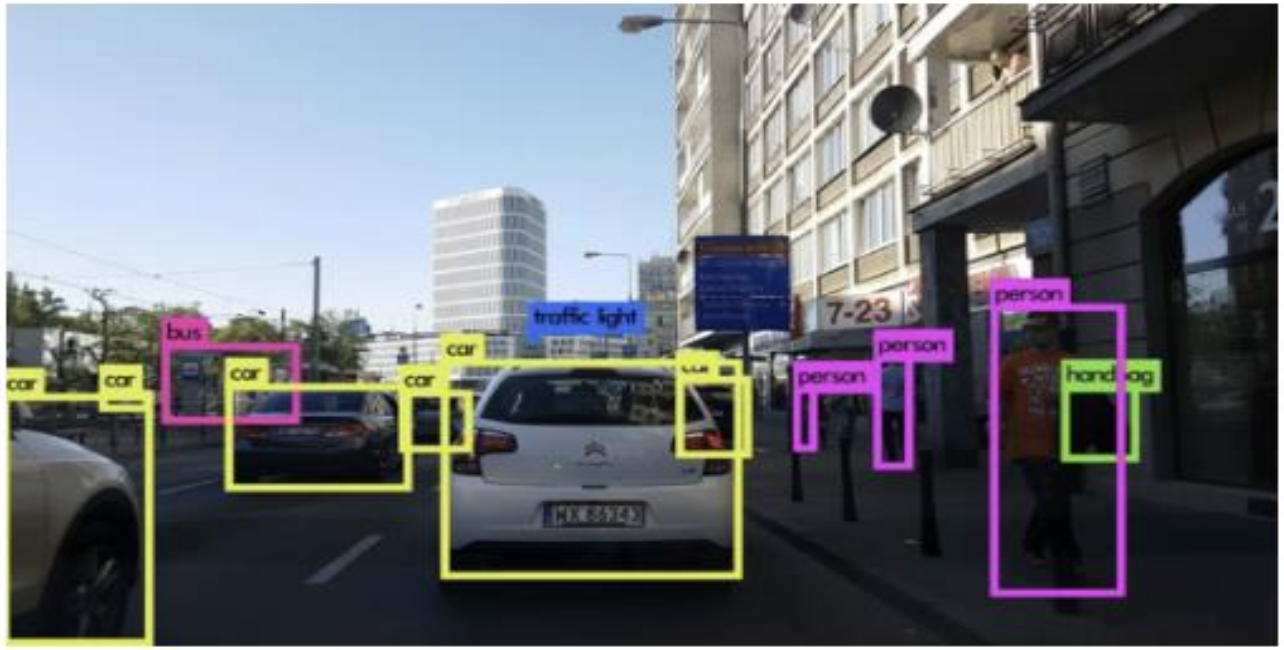


BOAZ

국내 최초 Big Data 연합동아리 BOAZ

목차

1. Introduction

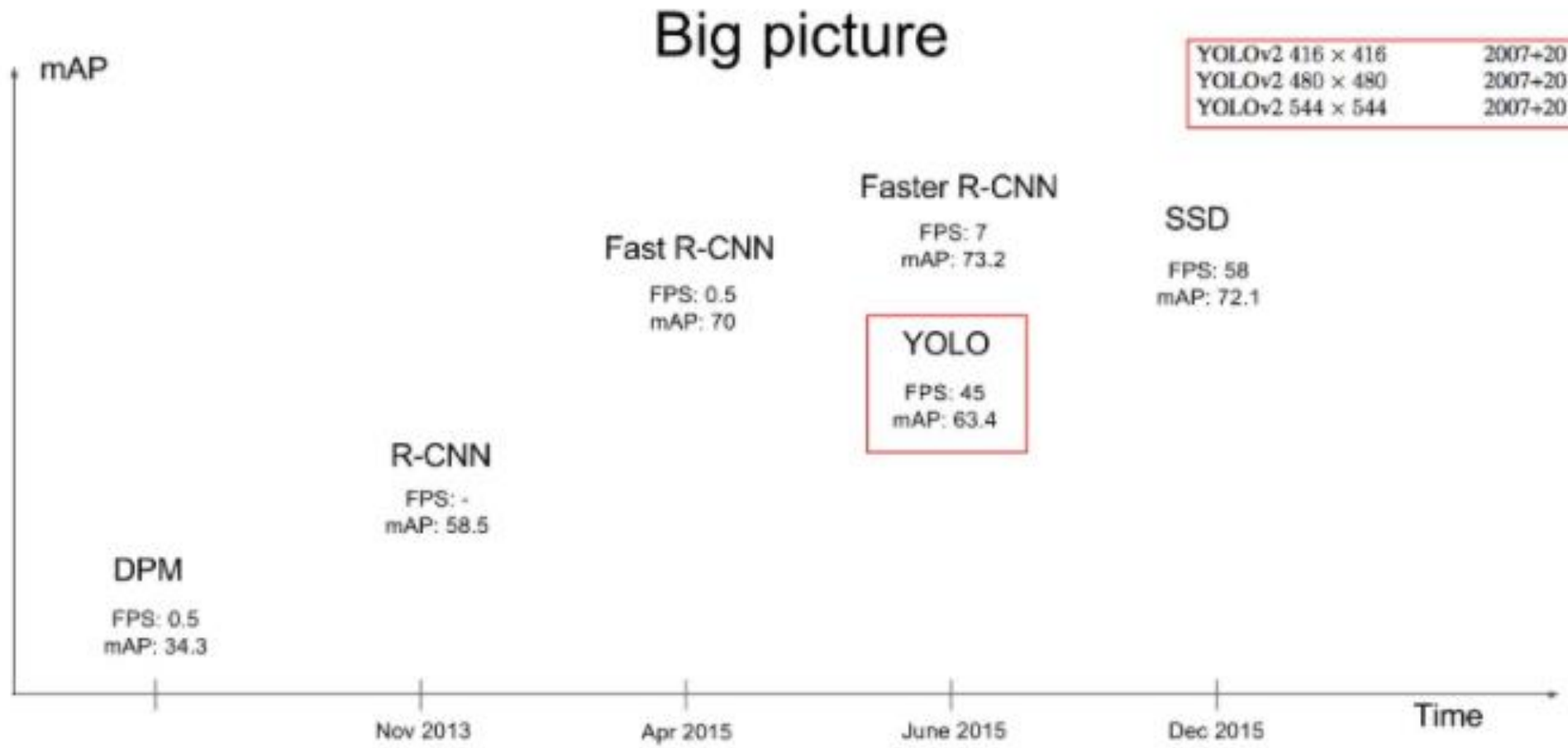


따라서 YOLO 저자들은 전체 이미지를 보고 이미지 안에 어떠한 객체들이 존재하구나 라고 즉각적으로 반응하기에는 부족하다라는 문제점을 인식하였다고 합니다.

- YOLO는 이미지 내 bounding box와 class probability를 하나의 regression 문제로 엮어, 이미지를 한 번 보는 것만으로 이미지 내에 위치한 객체의 종류와 위치 정보를 추측할 수 있는 unified detection을 이용한 모델 !
- 이것으로 인해 매우 빠른 속도를 기대할 수 있다.

1. Introduction

Evaluation on VOC2007



YOLOv2 416 × 416	2007+2012	76.8	67
YOLOv2 480 × 480	2007+2012	77.8	59
YOLOv2 544 × 544	2007+2012	**78.6**	40

1. Introduction

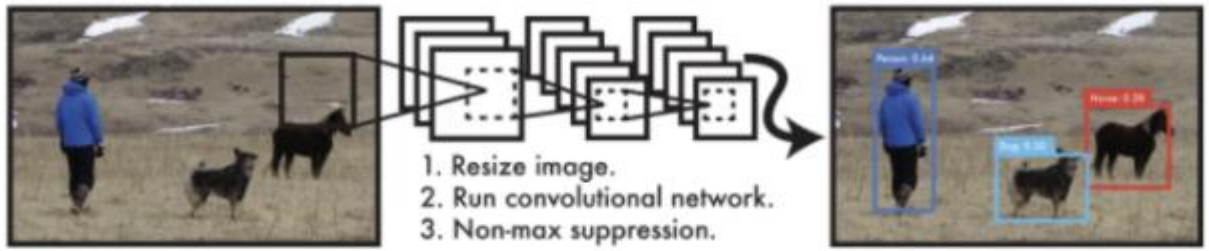


Figure 1: The YOLO Detection System. Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448 × 448, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.

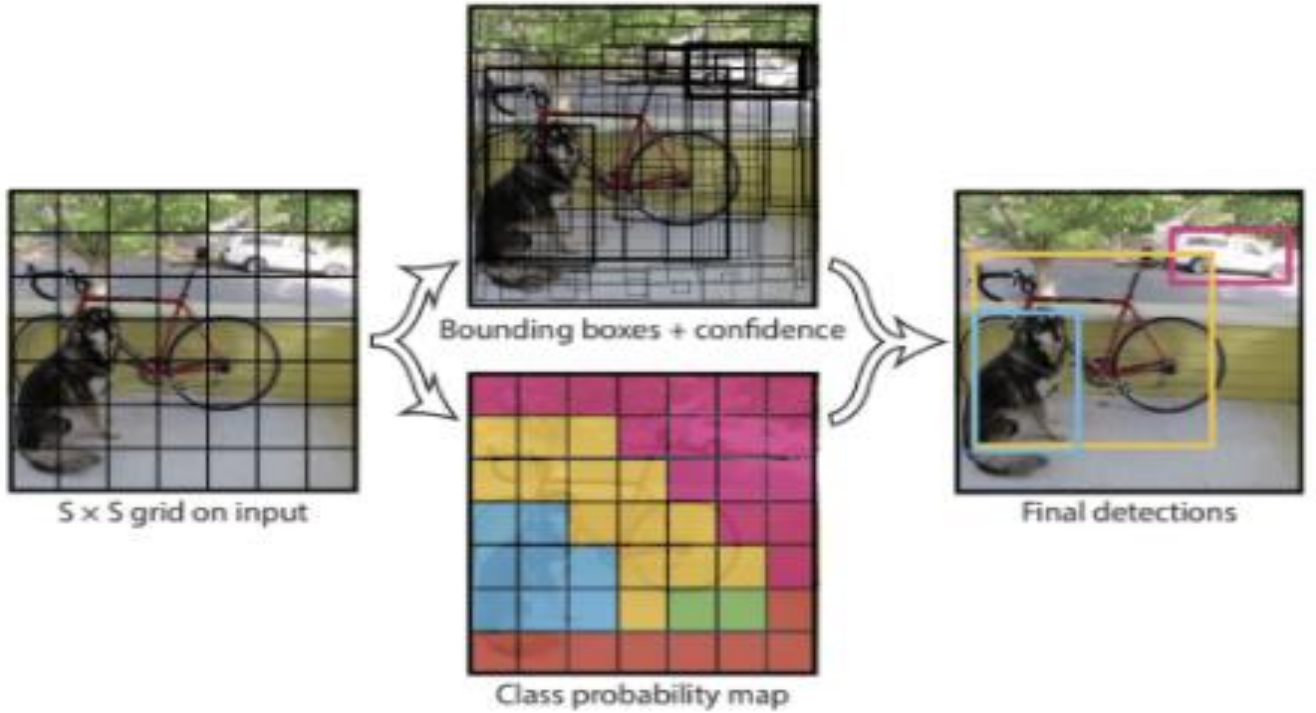
Pros:

- 간단한 처리과정으로 **빠른 속도** + 기존 Real-Time Detection 시스템들과 비교해 **2배** 가량 높은 **mAP**
- 이미지 전체를 한 번에 바라보기 때문에 Class에 대한 **맥락 이해**가 높고, 이 덕분에 **Background error가 낮음**
- 이미지 전체를 보며 객체의 **일반화된 특징**을 학습 -> "사진 속 의자의 모습을 학습한 후, 정물화 속 의자 추측 가능"

Cons: 다른 객체 인식 모델에 비해 상대적으로 낮은 정확도 (esp. 크기가 작은 객체 검출 시)

2. Unified Detection

2.1 Unified detection이란

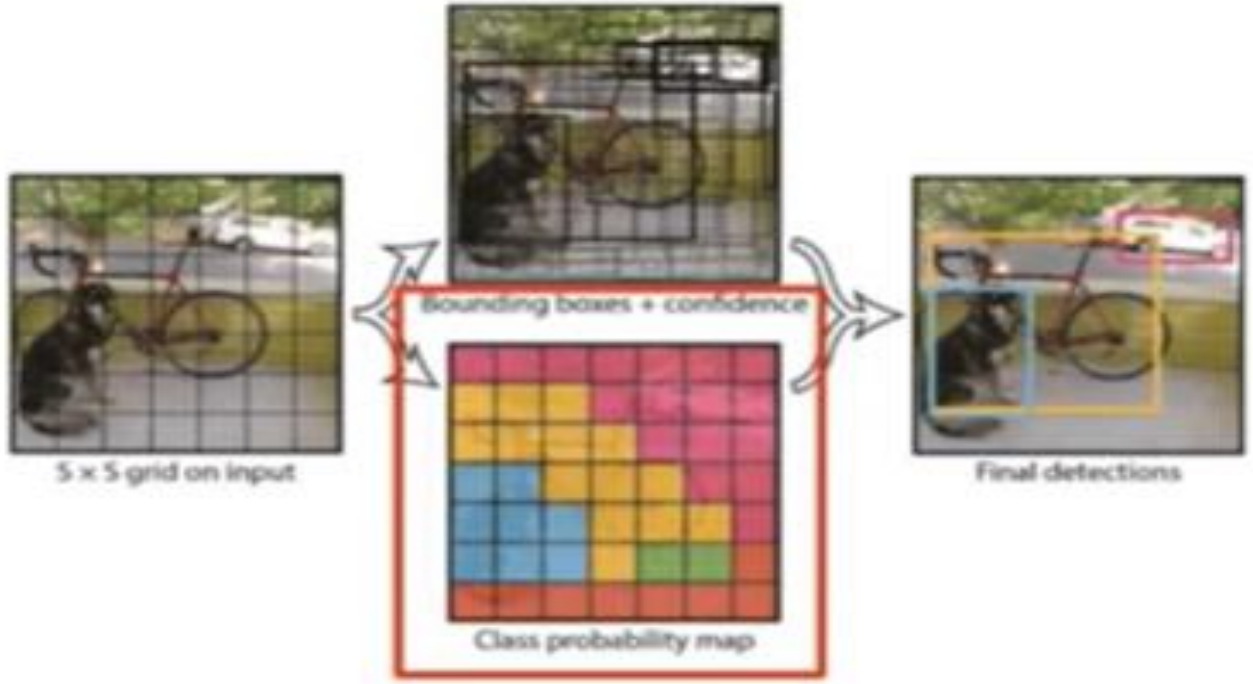


Input Image를 S X S개의 Grid cell로 나눠준다. (논문에서 S=7로 정의)
각각의 Grid cell은 B개의 Bounding box와 각 bounding box에 대한 confidence score를 가진다.
[Confidence Score: Pr(Object)*IOU truthpred] 이 논문에서는 B=2 설정

총 98개의 Bounding box가 생김

2. Unified Detection

2.1 Unified detection이란



각의 Grid cell은 C개의 Conditional Class Probability를 가짐

[**Conditional Class Probability**: Pr(*Class i* | *Object*)]

*cf) In the paper, C = 20 → totally 49 * 20 = 980 class probabilities*

2. Unified Detection

2.1 Unified detection이란

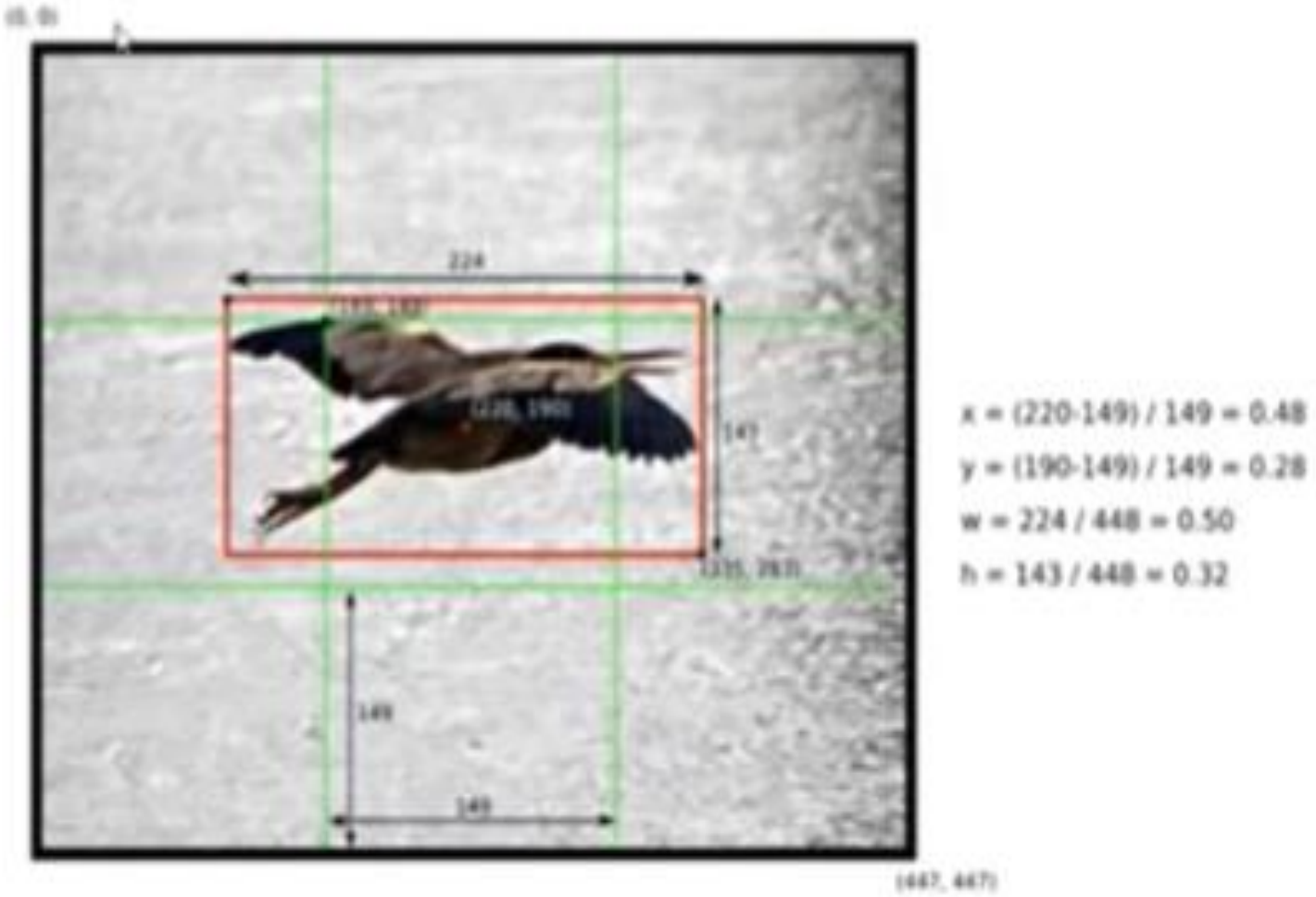
4. 각각의 Bounding box는 **x, y 좌표, w, h** 그리고 **confidence**를 지님

(x, y): Bounding box의 중심점을 의미하며, grid cell 범위에 대한 상대 값

e.g.) x가 grid cell 가장 왼쪽, y가 grid cell 중간에 있다면 x=0, y=0.5

(w, h): 전체 이미지의 width, height에 대한 상대 값

e.g.) Bounding box의 width가 전체 이미지 width의 절반이라면 w=0.5



$$class\ specific\ confidence\ score$$
$$= \Pr(Class_i|Object) * \Pr(Object) * IOU_{pred}^{truth}$$
$$= \Pr(Class_i) * IOU_{pred}^{truth}$$

앞서 봤던 confidence score와 Conditional Class Probability를 곱해서 Class-Specific Confidence Score가 되겠습니다.

2. Unified Detection

2.2 Network Design

-이제까지 다뤄온 Detection Model들과는 다르게 하나의 Convolutional Neural Network 구조
-Convolution 계층, FC layer로 클래스 확률과 Bounding box의 좌표(coordinates)를 예측
-모델의 최종 output은 7 x 7 x 30의 예측 텐서(prediction tensors)

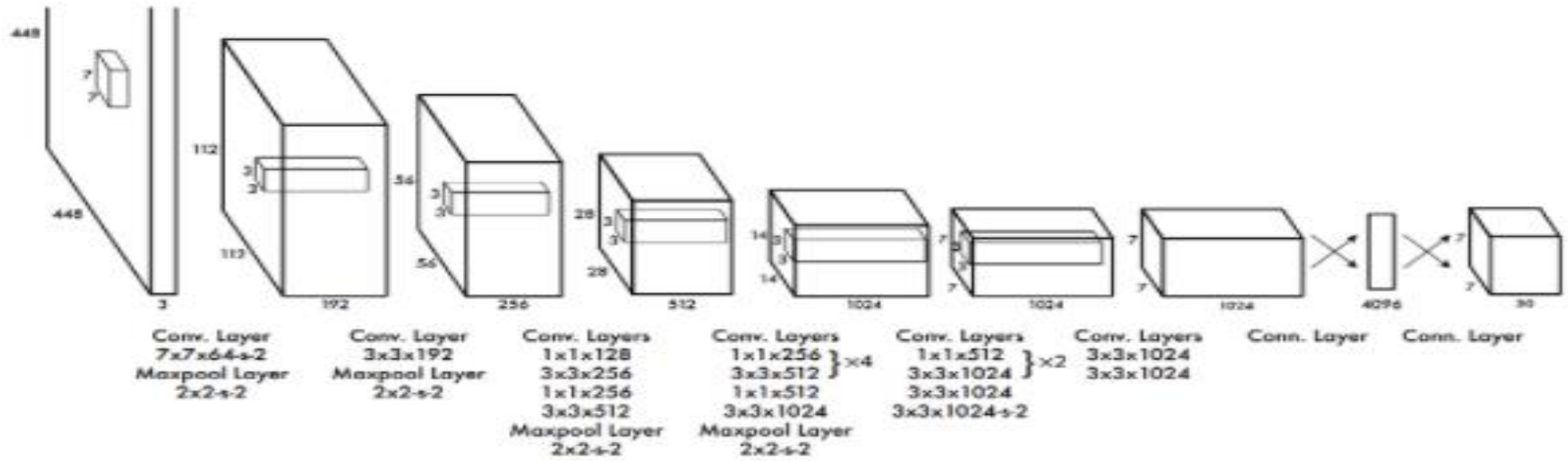
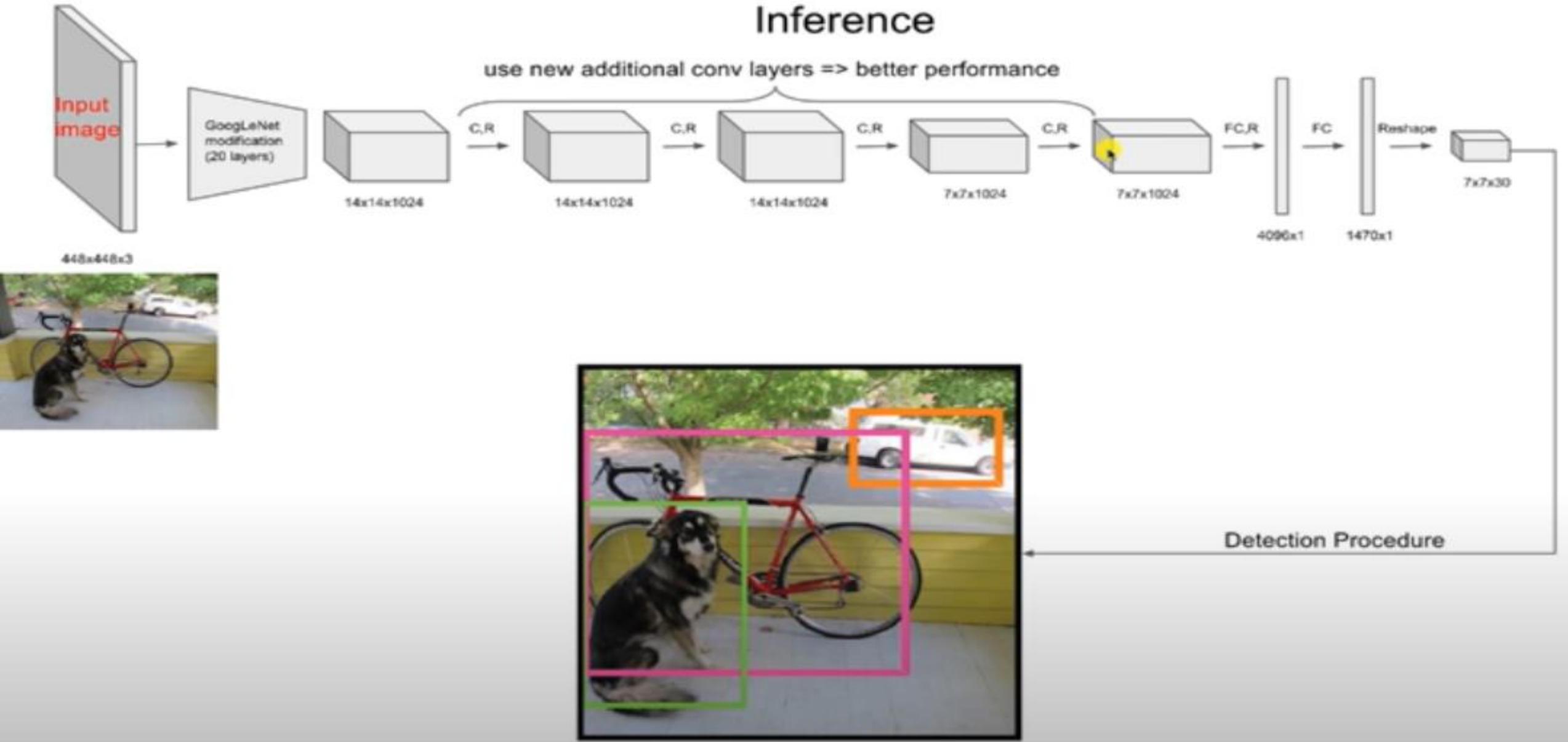


Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224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

- YOLO의 기본 Network Architecture는 **GoogLeNet**을 기반으로 함
- 24개의 Convolutional Layer + 2개의 Fully Connected Layer
- GoogLeNet의 인셉션 구조 대신 YOLO는 1 x 1 축소 계층(reduction layer)과 3 x 3 컨볼루션 계층의 결합을 사용
  - 좀 더 빠른 객체인식을 위해 Fast YOLO에서는 24개가 아닌 **9개의** Convolutional Layer 사용**(정확도 감소)**

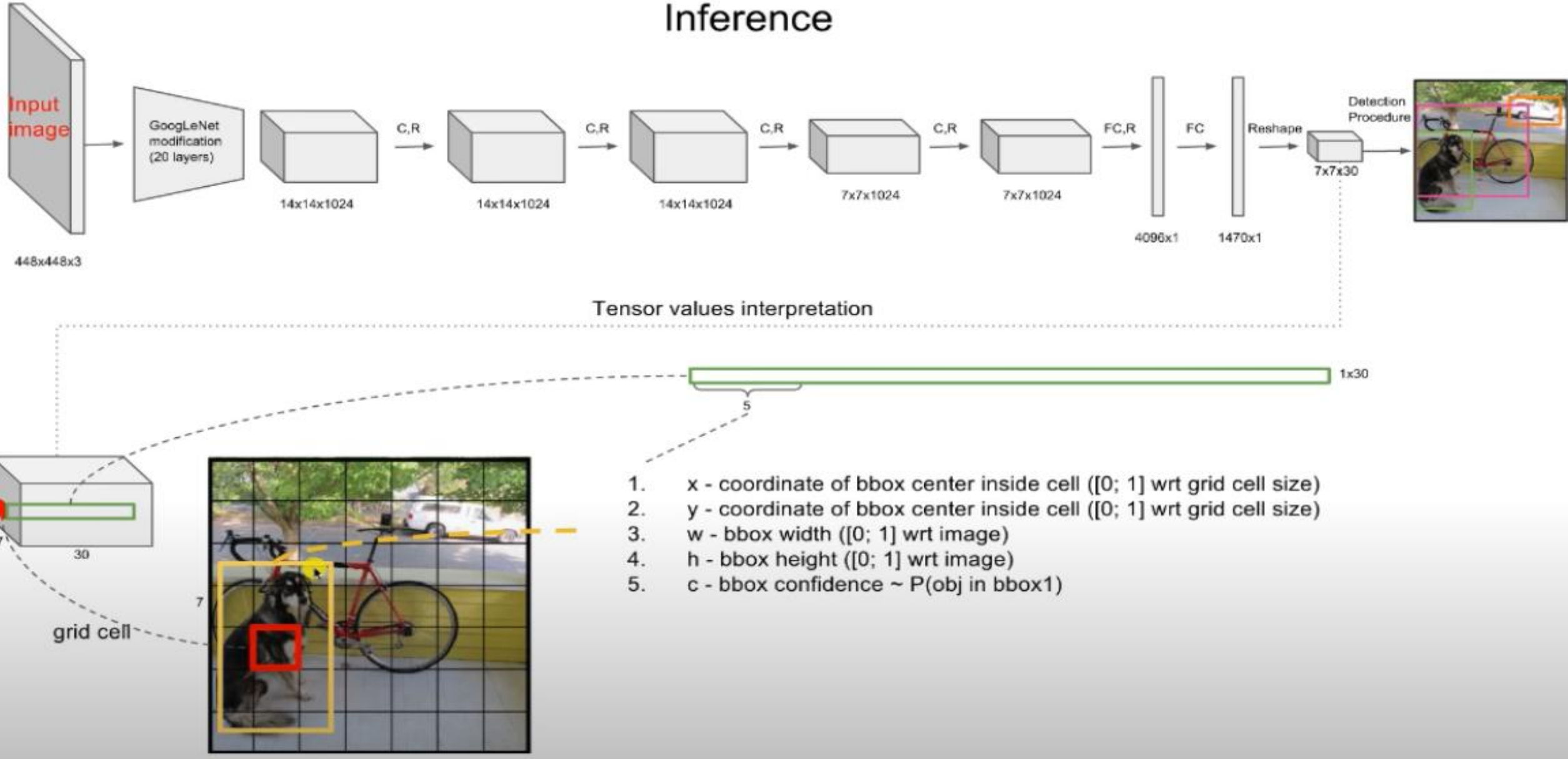
2. Unified Detection

2.2 Network Design



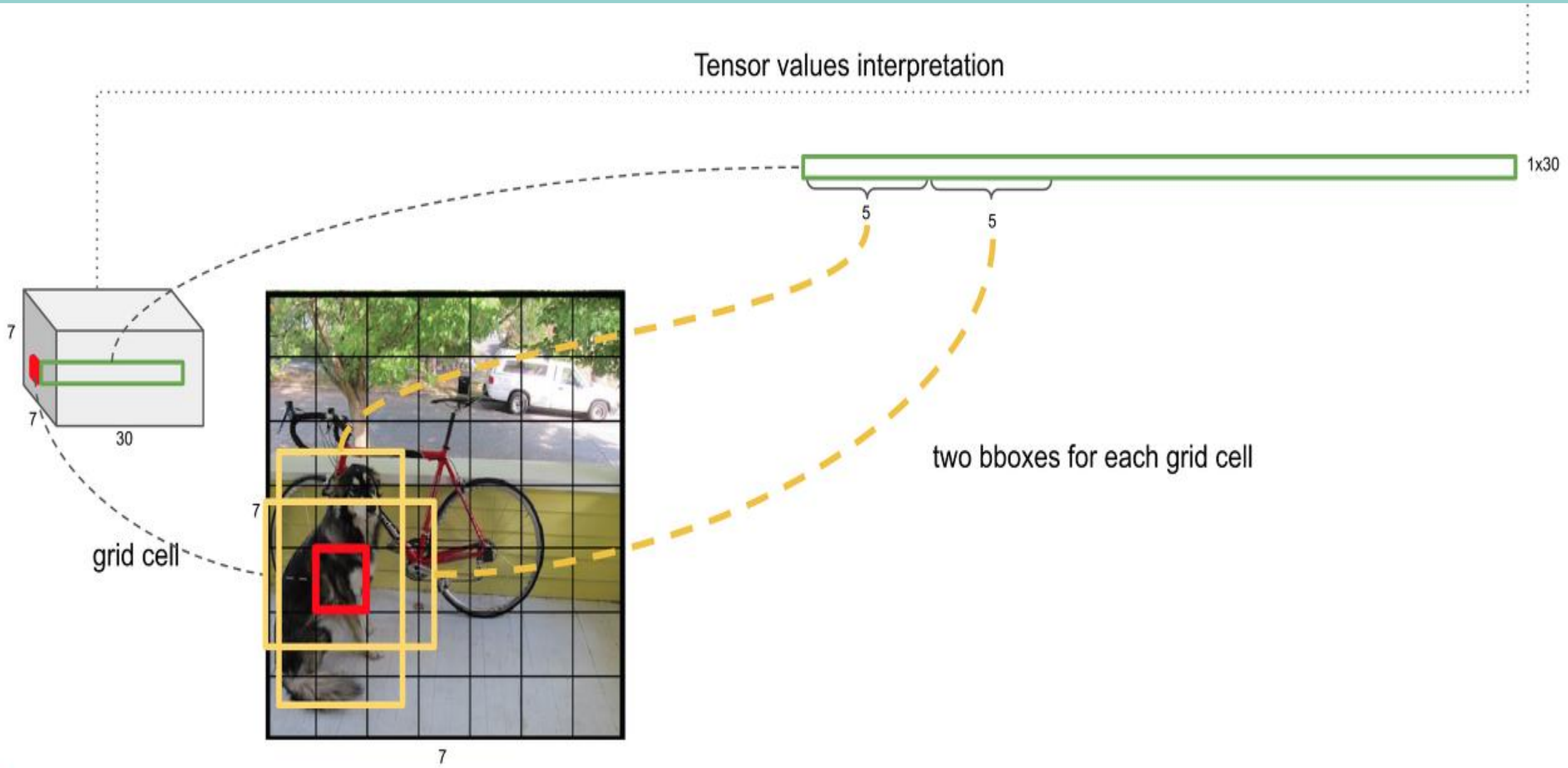
2. Unified Detection

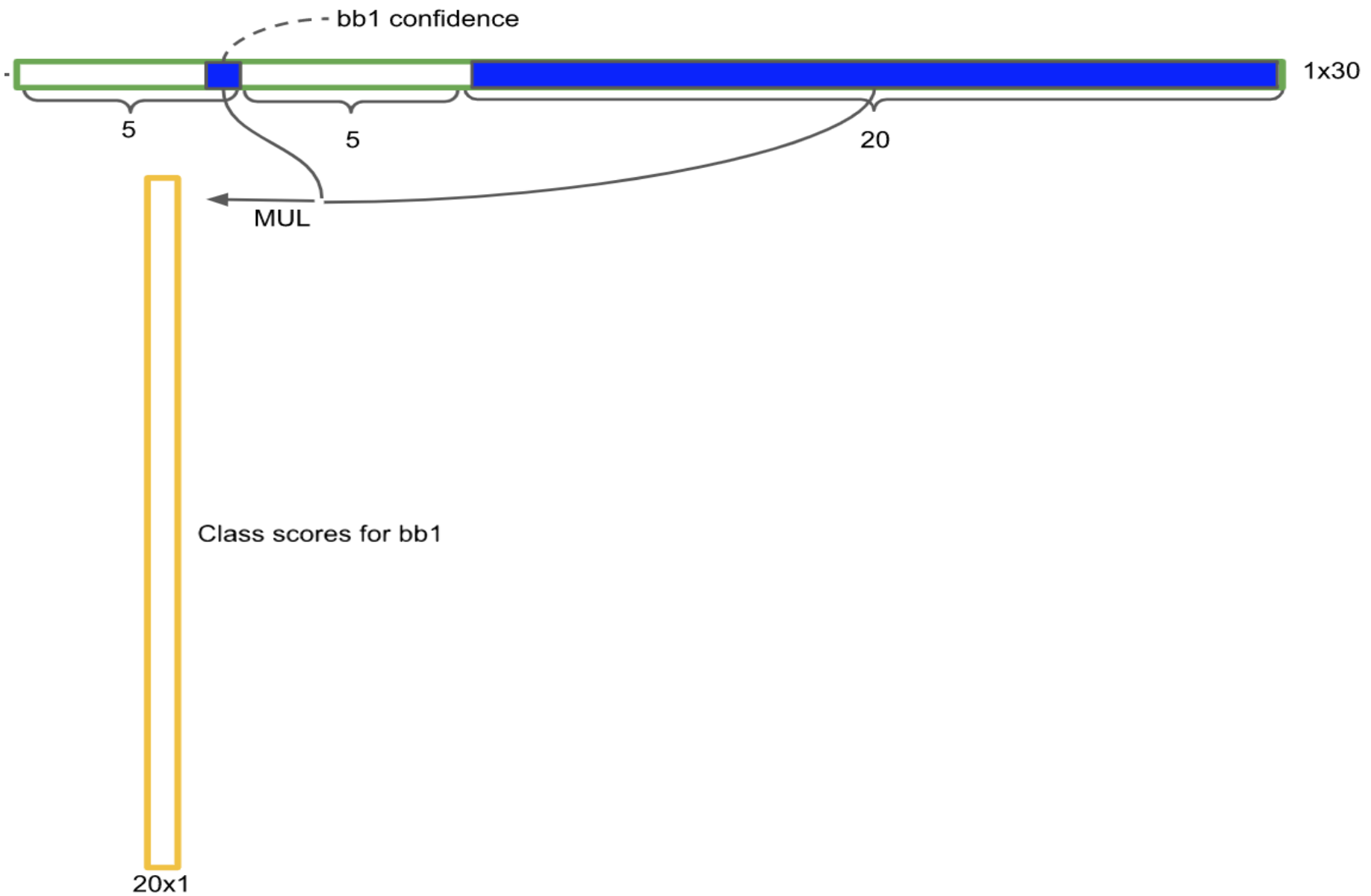
2.2 Network Design



2. Unified Detection

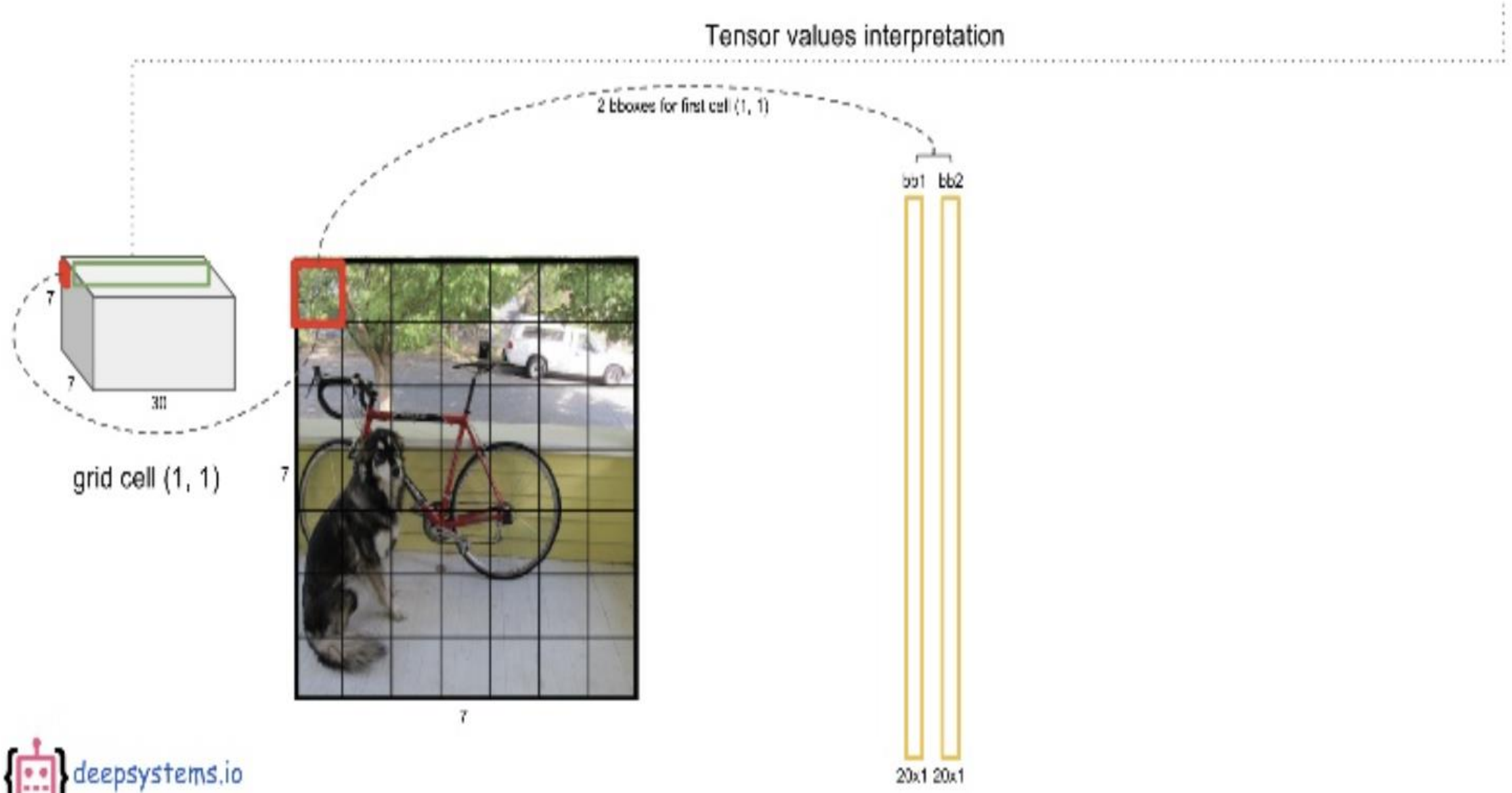
2.2 Network Design



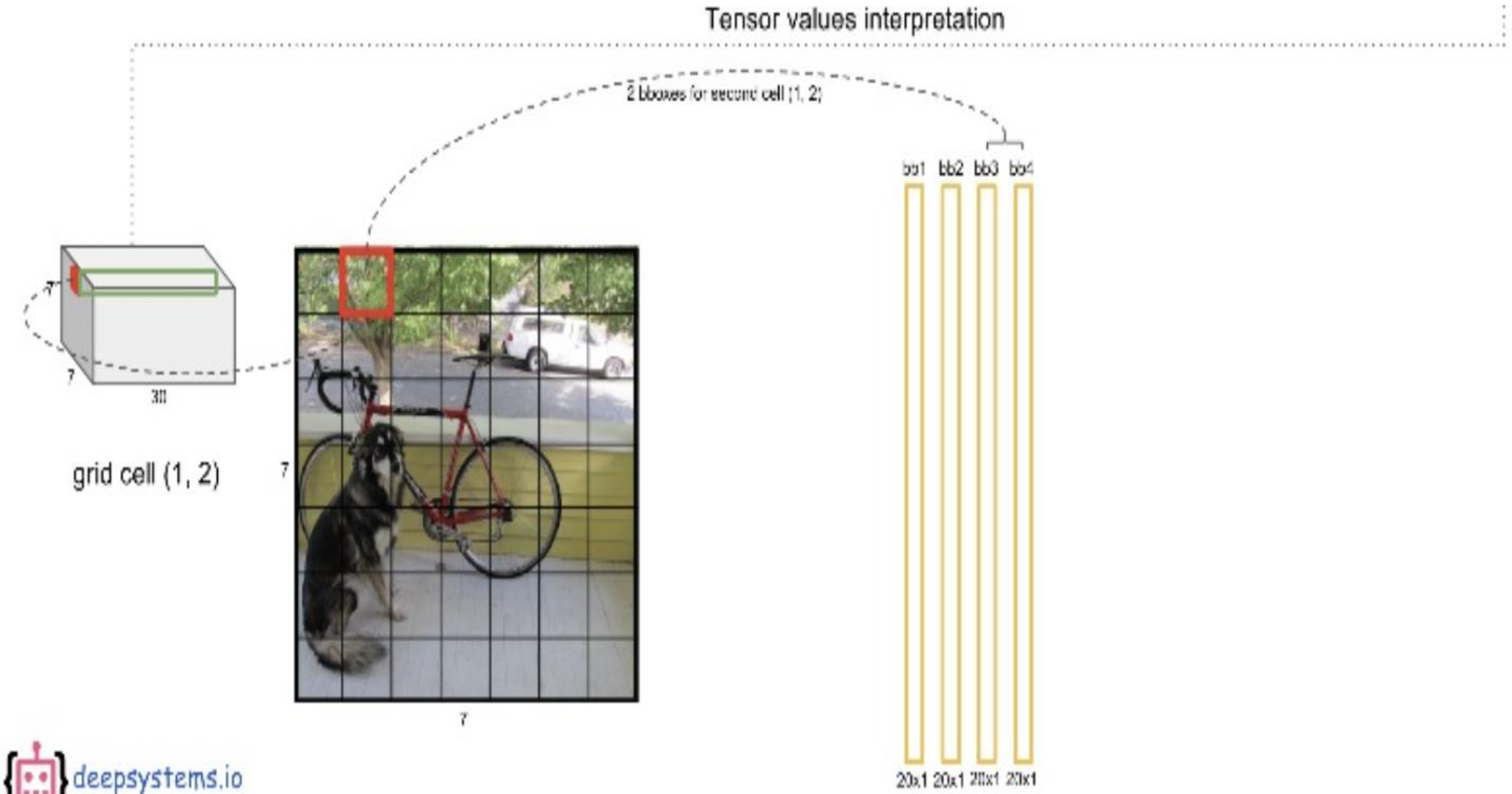

bb1 confidence
1x30
5
5
20
MUL
Class scores for bb1
20x1

2. Unified Detection

2.2 Network Design

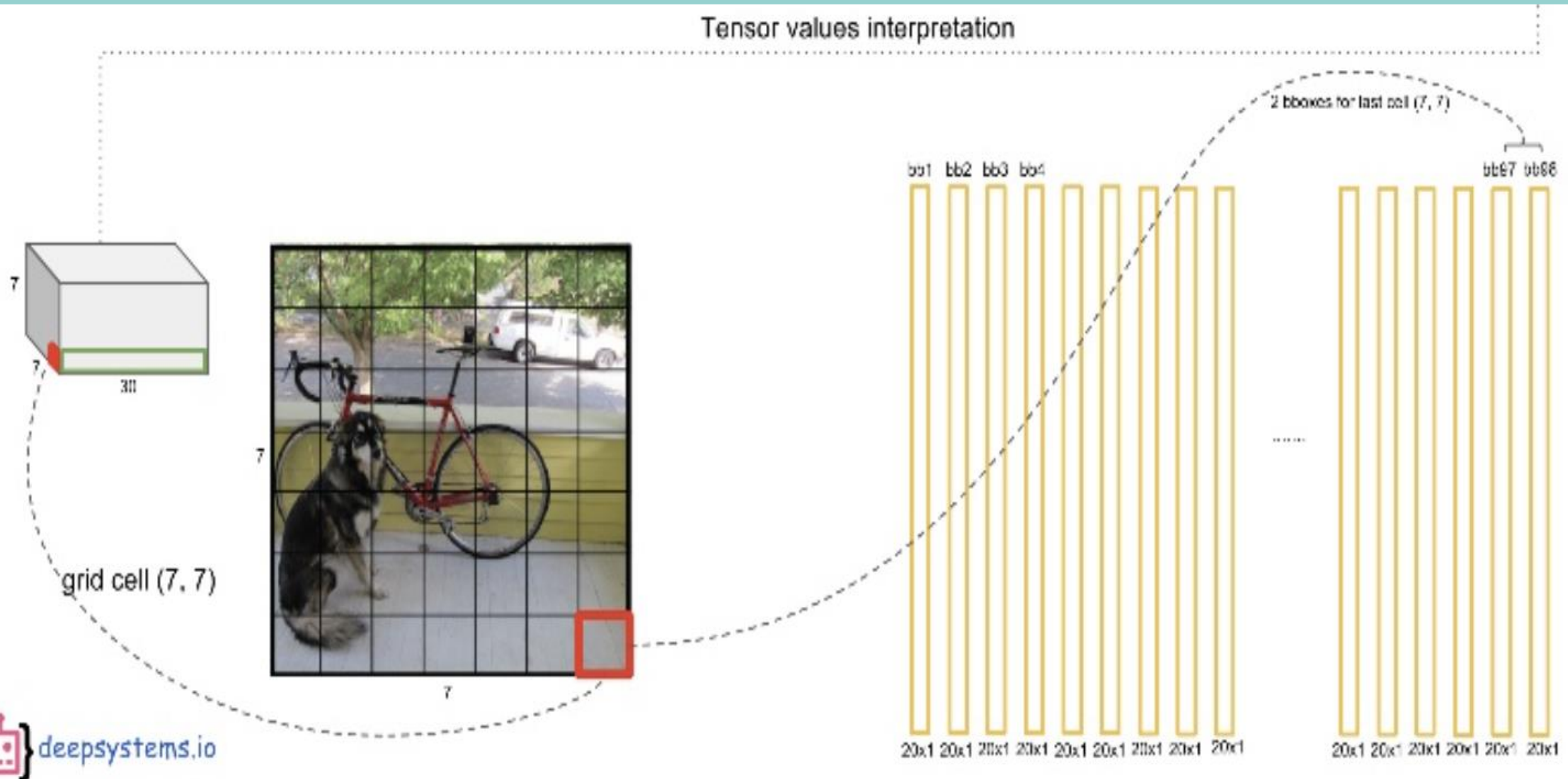


deepsystems.io


Tensor values interpretation
2 bboxes for second cell (1, 2)
bb1 bb2 bb3 bb4
grid cell (1, 2)
7
7
7
30
20x1 20x1 20x1 20x1

2. Unified Detection

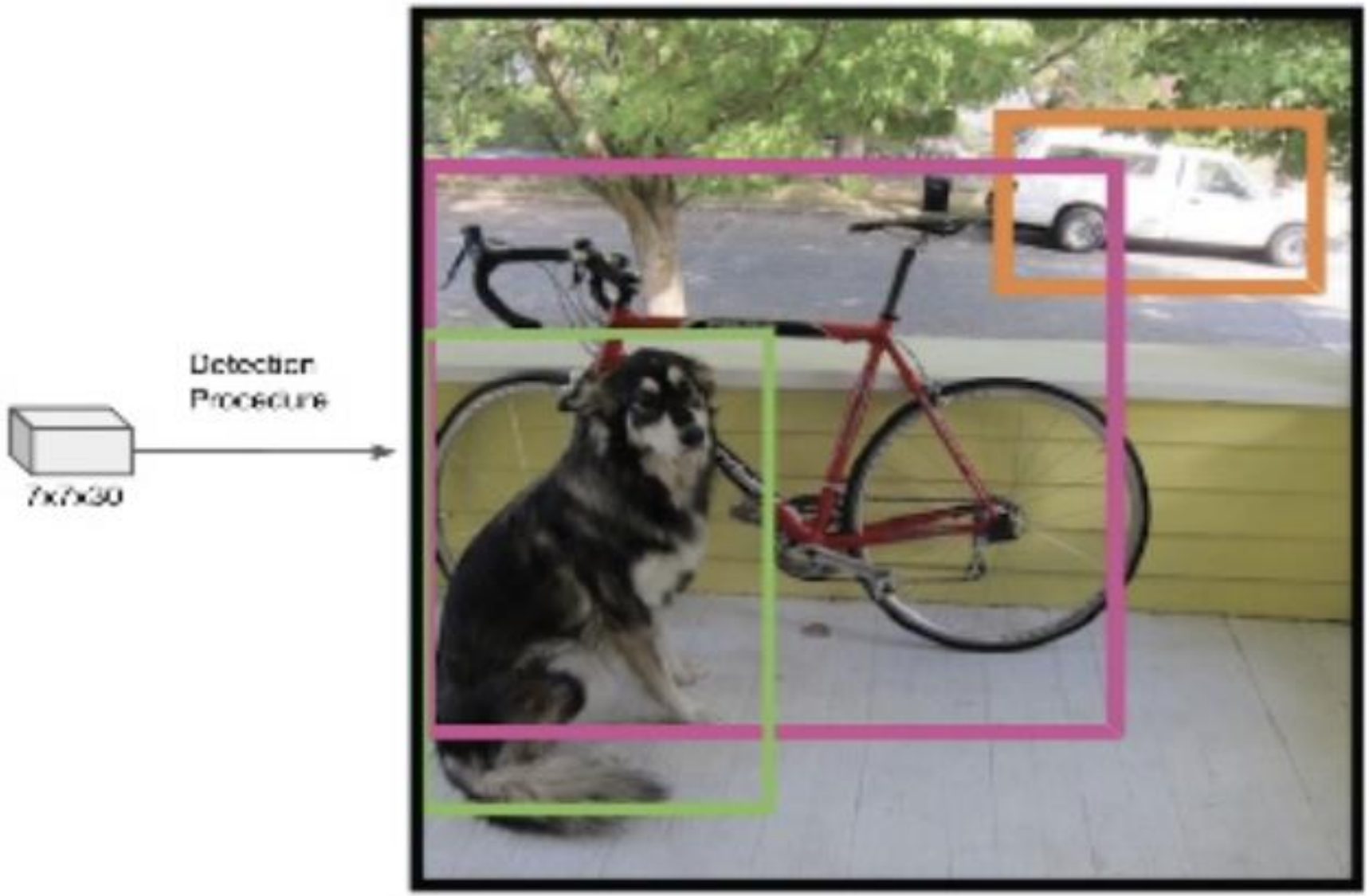
2.2 Network Design



deepsystems.io

2. Unified Detection

2.2 Network Design



1. Grid Cell의 여러 Bounding box들 중, Ground-truth box와의 IOU가 가장 높은

   Bounding box를 **predictor**로 설정

2. 1번의 기준에 따라 우측의 Notation들이 사용

   1) Object가 존재하는 Grid cell i의 predictor bounding box j $\mathbb{1}_{ij}^{\text{obj}}$ (1)

   2) Object가 존재하지 않는 Grid cell i의 bounding box j $\mathbb{1}_{ij}^{\text{noobj}}$ (2)

   3) Object가 존재하는 Grid cell I $\mathbb{1}_{i}^{\text{obj}}$ (3)

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

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

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

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

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

λ_{coord}: coordinates(x,y,w,h)에 대한 loss와 다른 loss들과의 균형을 위한 balancing parameter.

λ_{noobj}: obj가 있는 box와 없는 box간에 균형을 위한 balancing parameter. (일반적으로 image내에는 obj가 있는 cell보다는 obj가 없는 cell이 훨씬 많으므로)

2. Unified Detection
2.4 한계점

- **Limitations of YOLO**

1. 각 그리드는 두개의 bbox를 예측하고 하나의 class를 가질 수 있기 때문에 공간적 제약이 생길 수 밖에 없다. 때문에 작은 물체에 대한 탐지가 어려움이 생긴다.
2. bounding box의 형태가 학습 데이터를 통해서만 학습되기 때문에, 새로운 Input으로 이전에 학습되지 않은 형태의 Bounding box를 예측해야 할 경우, 제대로 된 예측을 할 수 없다.
3. 몇 단계의 Convolutional Layer를 거쳐 나온 Feature map을 대상으로 Bounding box를 예측하기 때문에 객체의 위치를 정확히 파악하는 Localization이 다소 부정확해지는 경우가 발생

3. Experiments

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

Table 1: Real-Time Systems on PASCAL VOC 2007. Comparing the performance and speed of fast detectors. Fast YOLO is the fastest detector on record for PASCAL VOC detection and is still twice as accurate as any other real-time detector. YOLO is 10 mAP more accurate than the fast version while still well above real-time in speed.

다른 Real-Time Detection 시스템보다 월등히 높은 mAP를 기록함
Real-Time에서는 확실히 강한 장점을 보이지만 Real-Time이 아닌 경우에는 조금 낮은 성능을 보인다.

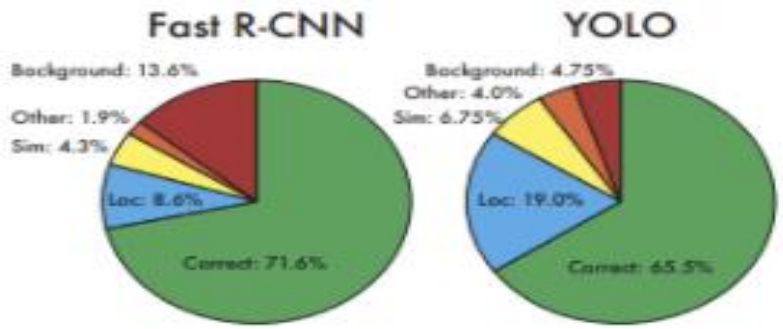
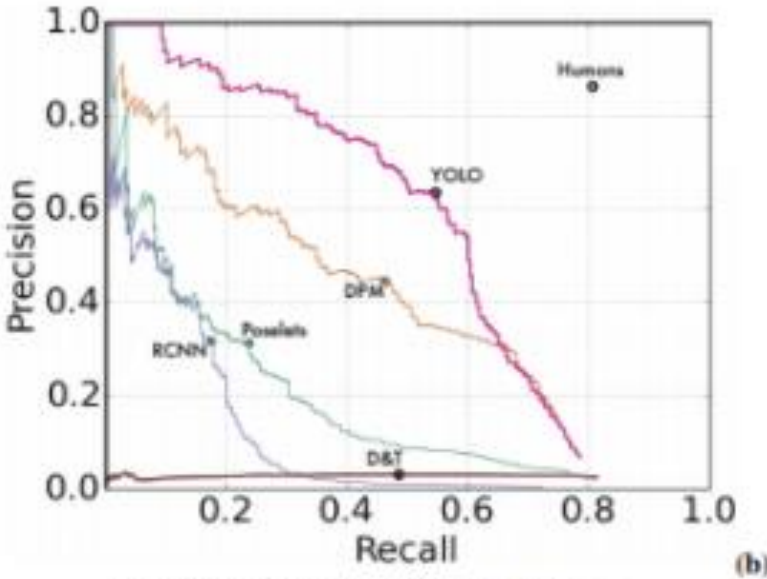


Figure 4: Error Analysis: Fast R-CNN vs. YOLO These charts show the percentage of localization and background errors in the top N detections for various categories (N = # objects in that category).

Fast R-CNN과 비교해보았을 때, 훨씬 낮은 Background error를 보이며, 이는 이미지 전체를 보고 학습하기 때문에 배경에 대한 학습이 잘 되었다는 반증이 되기도 한다. 하지만 Loc(위치) 에러가 Fast R-CNN에 비해 매우 크다.

3. Experiments



(a) Picasso Dataset precision-recall curves.

	VOC 2007	Picasso		People-Art
	AP	AP	Best F_1	AP
YOLO	**59.2**	**53.3**	**0.590**	**45**
R-CNN	54.2	10.4	0.226	26
DPM	43.2	37.8	0.458	32
Poselets [2]	36.5	17.8	0.271	
D&T [4]	-	1.9	0.051	

(b) Quantitative results on the VOC 2007, Picasso, and People-Art Datasets. The Picasso Dataset evaluates on both AP and best F_1 score.

Picasso Dataset을 이용해서 Pre-train 후, Object를 검출하게 한 결과, 다른 모델들 보다 훨씬 더 좋은 성능을 보였고, 이는 일반화가 아주 잘 되고 있다는 결과이다.

-> 사진에서 나타난 의자와 정물화해서 나타난 의자를 가지고 생각해볼 때 욜로는 둘다 잘 의자라고 분류하지만 R-CNN은 정물화된 의자는 잘 판단하지 못한다는 의미

4. Conclusion

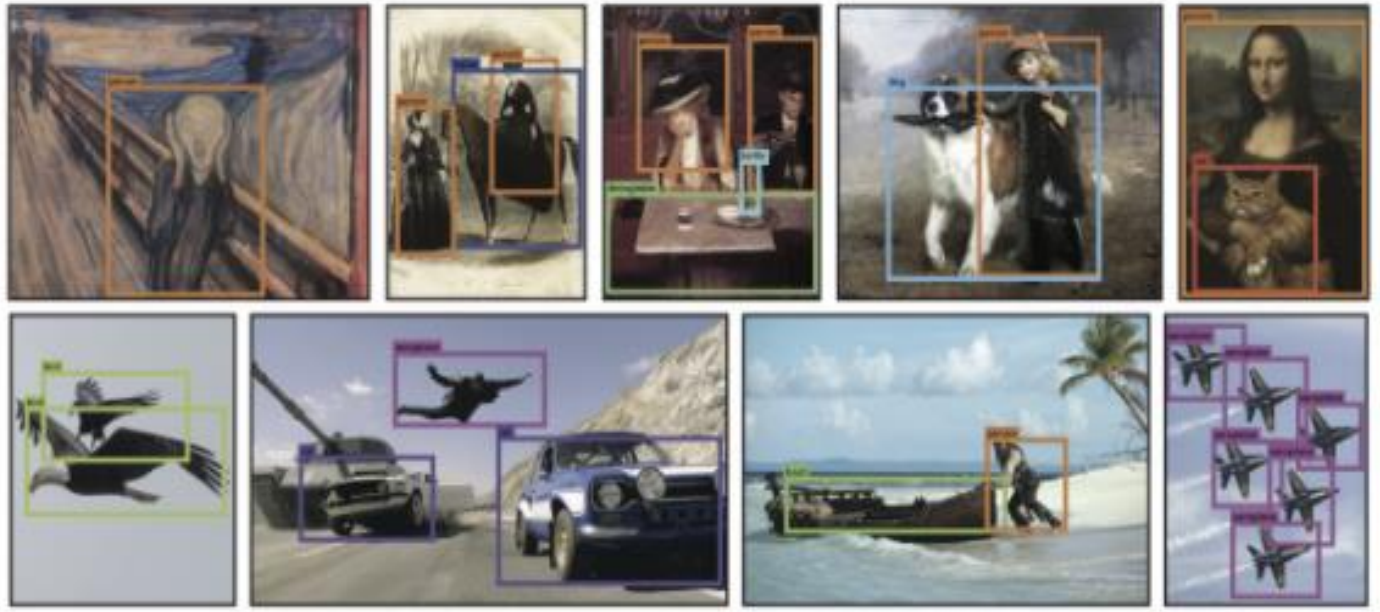


Figure 6: Qualitative Results. YOLO running on sample artwork and natural images from the internet. It is mostly accurate although it does think one person is an airplane.

- YOLO 모델은 간단하게 구현이 가능하며, 전체 이미지를 바로 학습시킬 수 있다
- Classified-based 접근 방식과는 다르게 Detection 성능과 직접적인 관계를 맺고 있는 loss function을 통해 학습
- YOLO는 Real-Time Object Detection의 성능 지표를 올렸으며, 일반화가 잘 되기 때문에 새로운 도메인에도 쉽게 적용 가능한 모델이다.

THANK YOU!