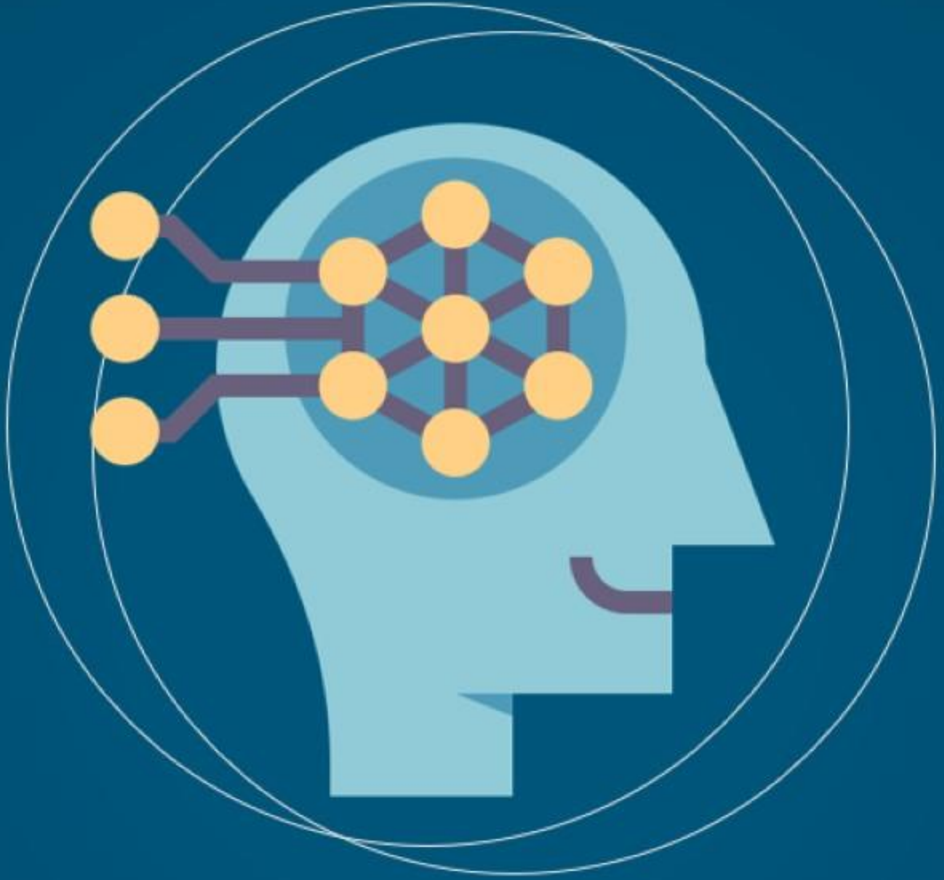
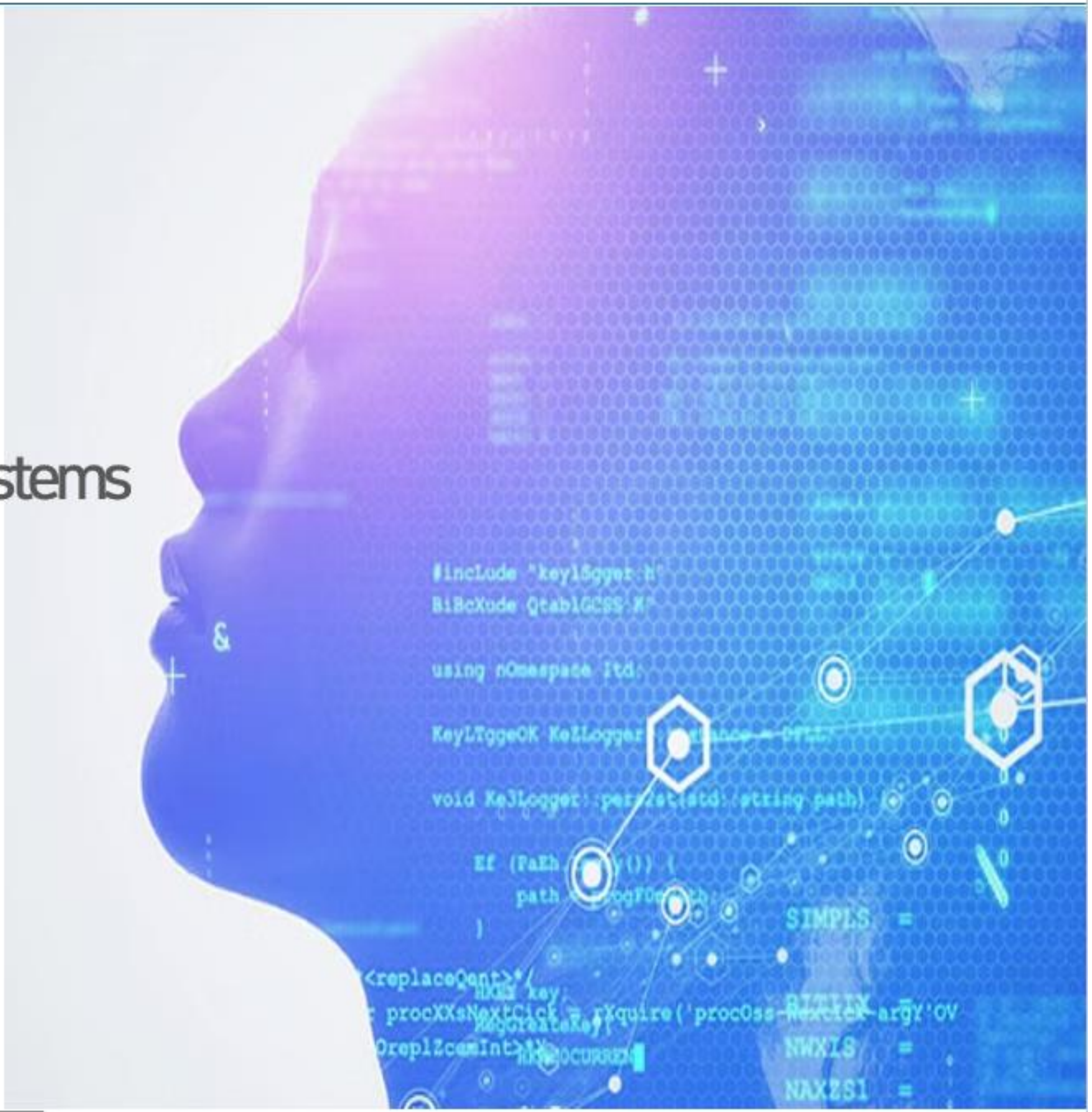


You Only Look Once: Unified, Real- Time Object Detection

이다희, 박유림

목차 Table of Contents



Chapter 1. Introduction

1 Introduction

You Look Only Once

이미지의 객체를 한번에 파악할 수 있는 객체 탐지 알고리즘

- Classification과 localization 단계를 단일화
- 복잡하고 다계층적인 R-CNN의 한계점 극복

R-CNN

ROI 생성 -> CNN 특징 추출 -> SVM 분류 -> Box regressor
Bounding box 세분화 및 학습 및 재평가로 인해 복잡한 파이프라인 필요, 시간 많이 소요되는 단점

1 Introduction

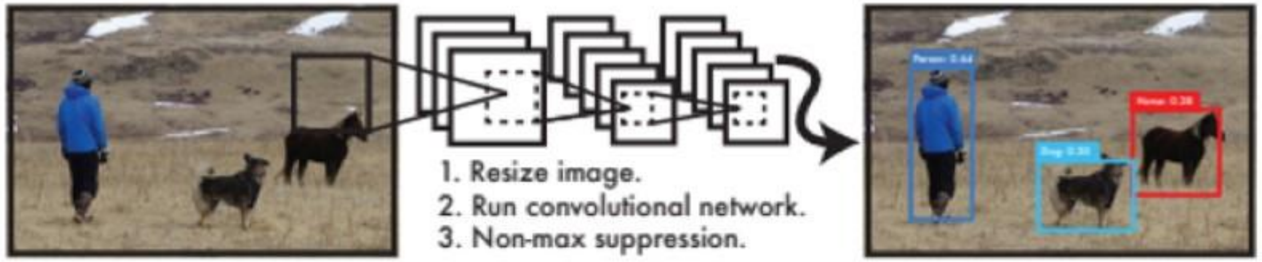


Figure 1: The YOLO Detection System. Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448×448, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.

Fast, object detection을 회귀로 단순화, 실시간 처리 가능
Globally inferred, 이미지 전체를 통해 bounding box의 class 예측
Generalize, 객체의 일반화 가능한 표현 학습

Chapter 2. Unified Detection

2 Unified Detection

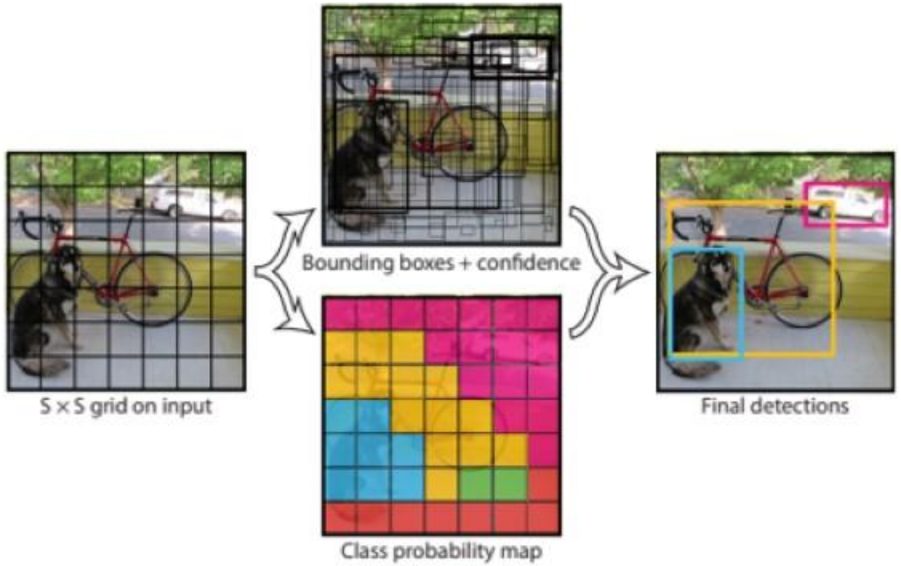
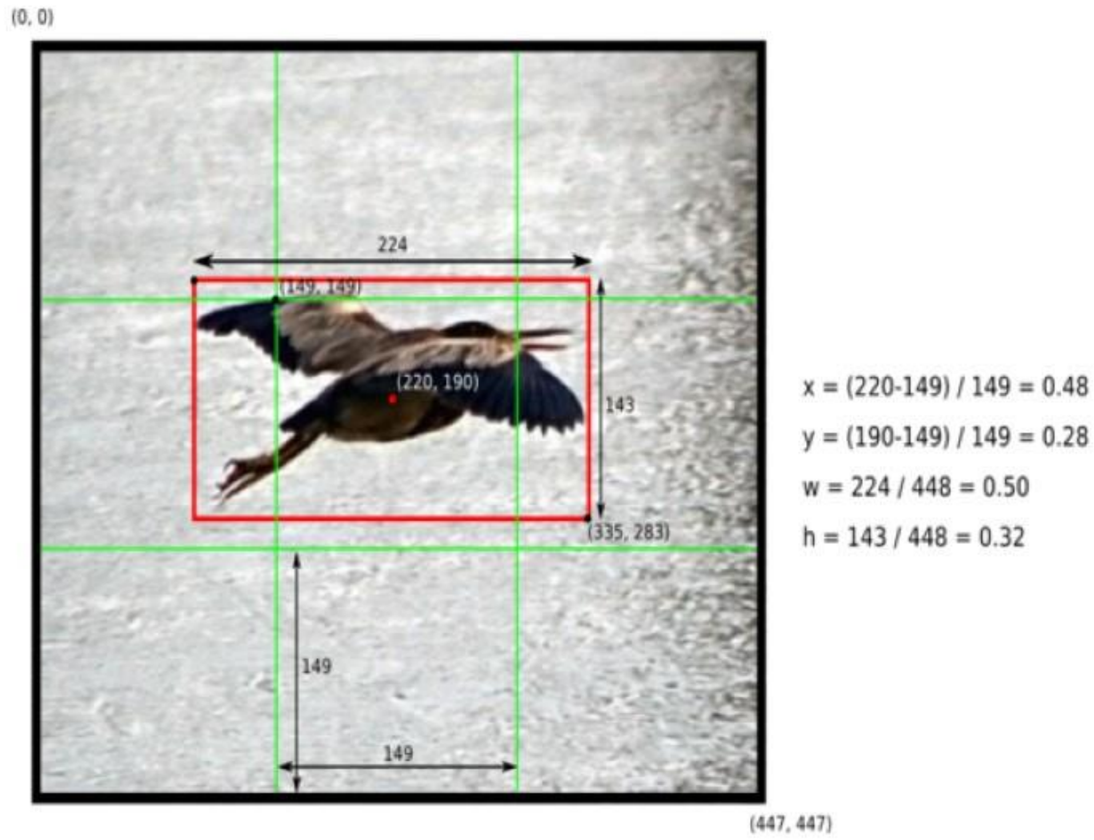


Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

Stage

Grid가 7개, b=2
1.49개의 셀 *2개 = 총 98개 bounding box 학습
2.각 bbx는 confidence score 가짐
Confidence score= Pr(object) *IOU(영역의 교집합/영역의 합집합)
3. 각 Grid cell- C개의 Conditional Class Probablity를 가짐
Conditional Class Probability = Pr(Classi | Object))
4. 최종 탐지

2 Unified Detection



출처:https://stwhat-does-the-coordinate-output-of-yolo-algorithm-representackoverflow.com/questions/52455429/

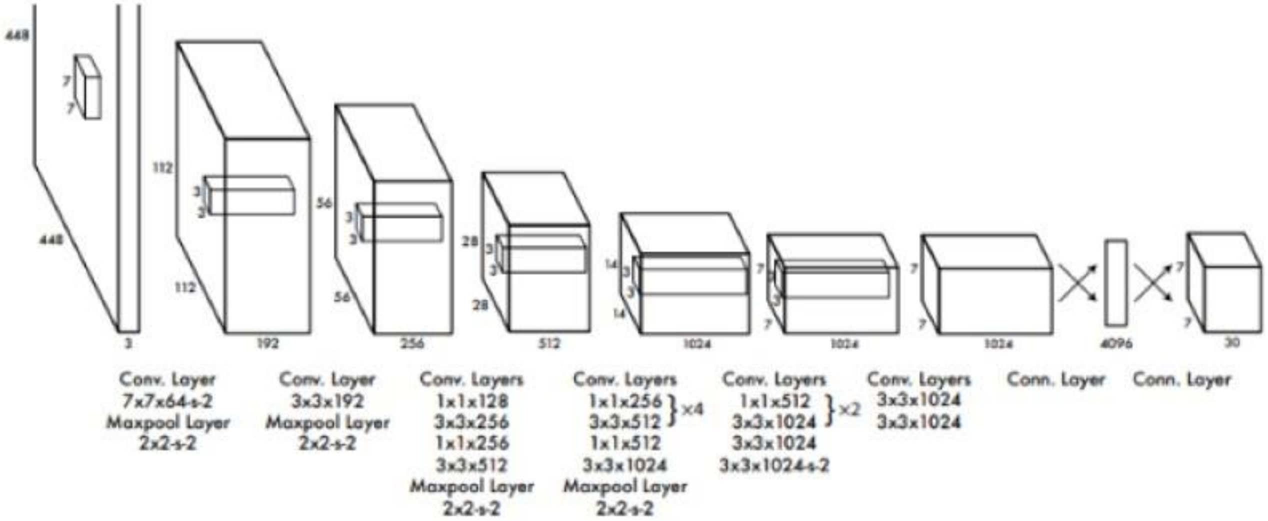
Output

1.x,y = bounding box의 중심표
2.w,h = 이미지 자체의 폭, 길이에 대한 상대값
Normalize 한 값으로 0과 1사이의 값
3. Class Specific Confidence Score
= Confidence score * Conditional Class Probability

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

2 Unified Detection

2.1 Network Design



Yolo v1은 GoogLeNet 모티브

24 Convolutional Layer + 2 fully connected layer

Fast Yolo- 9 Convolutional Layer + 2 fully connected layer

2 Unified Detection

2.2 Training Stage

IOU가 가장 높은 bounding box 한 개만 사용 – predictor로 선정, 오차 학습

loss function:

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

x,y의 loss 계산

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

w,h의 loss 계산

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

Object가 있는 곳의 SSE

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

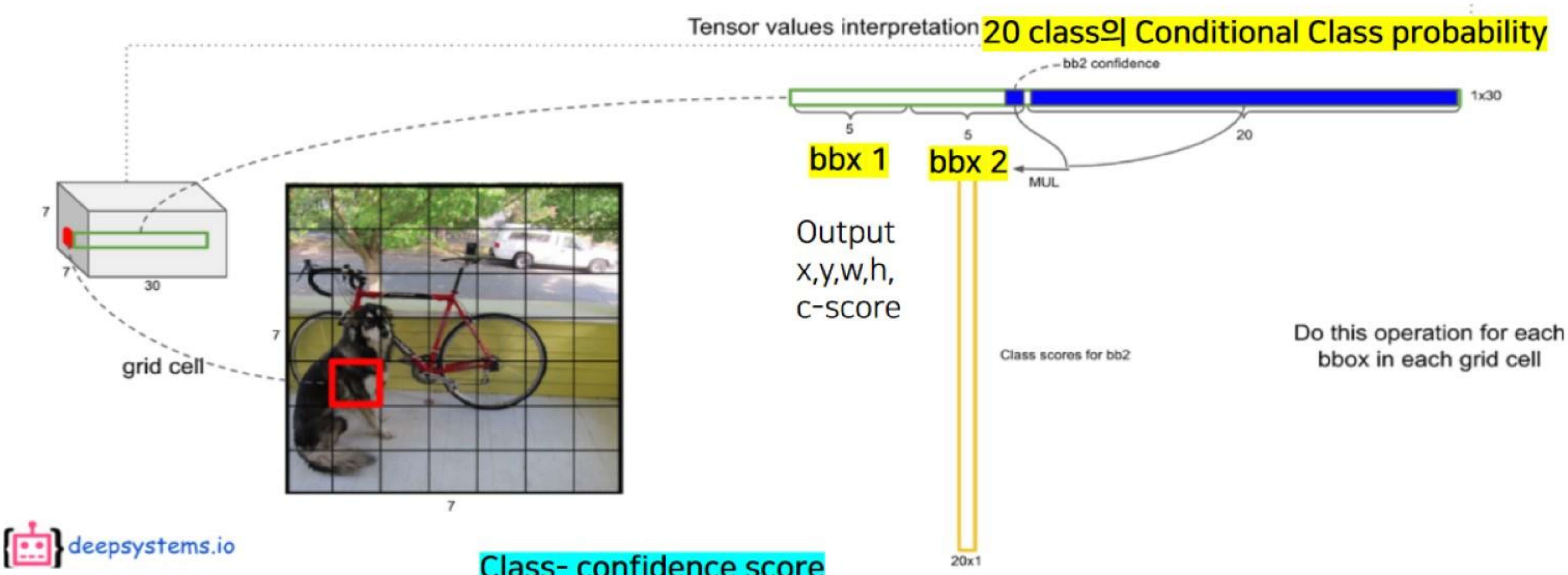
Object가 없는 곳의 SSE

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

각 class 별 SSE

Unified Detection

2.3 Inference



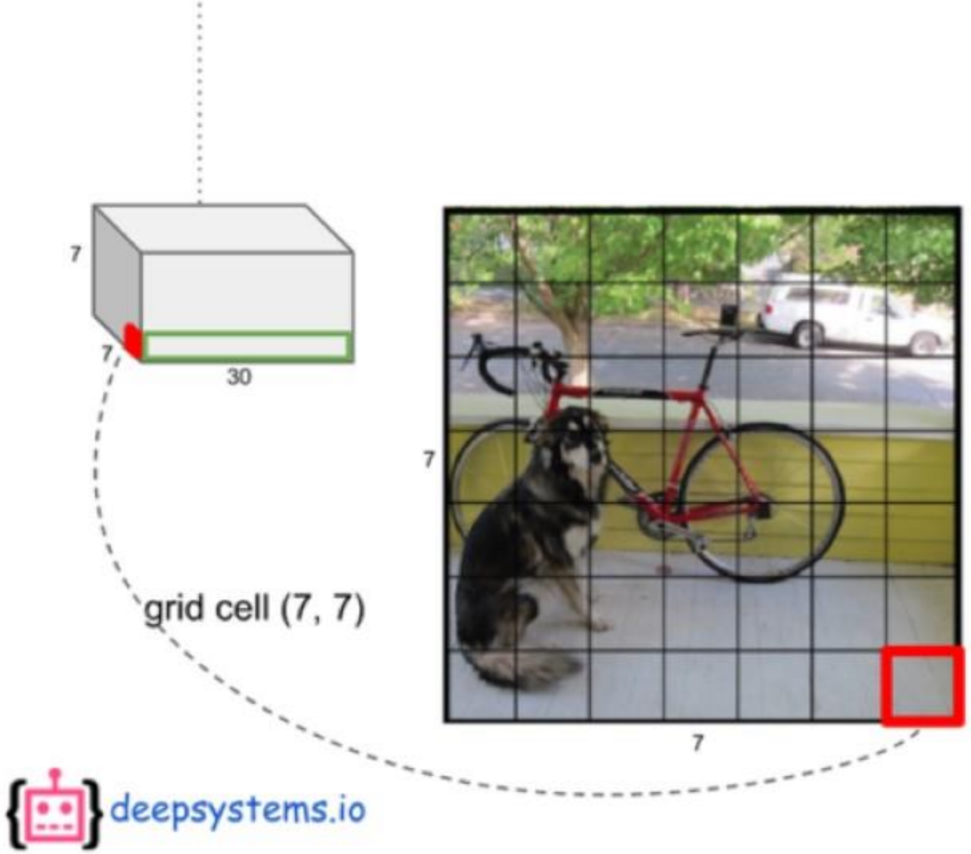
Class- confidence score

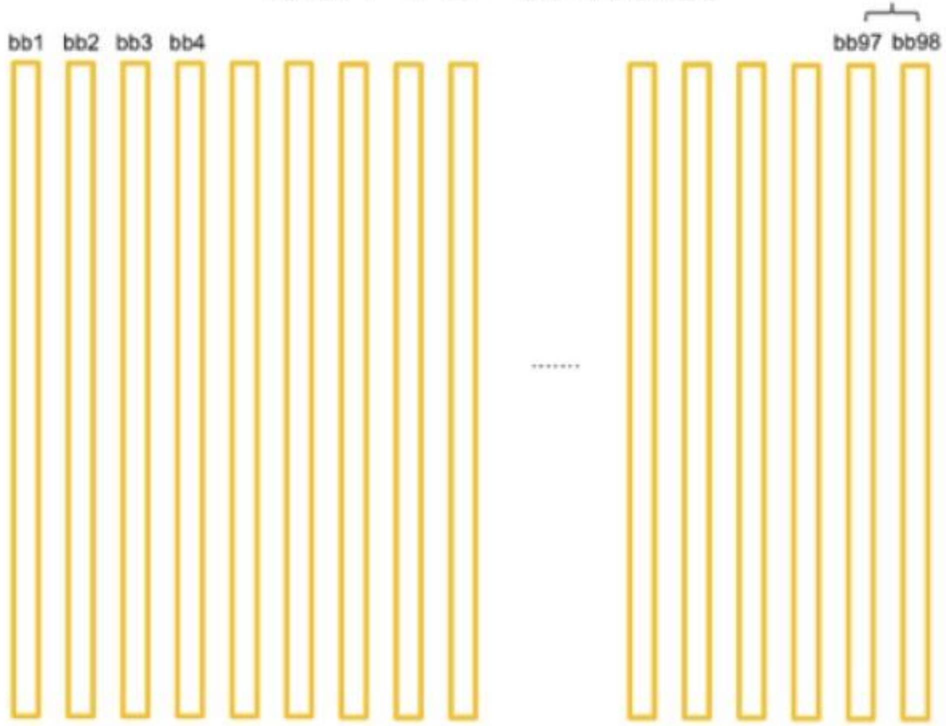
Step 1. bounding box의 confidence score x conditional class probability

2 Unified Detection

2.3 Inference

98개 박스 x 20 = 1440

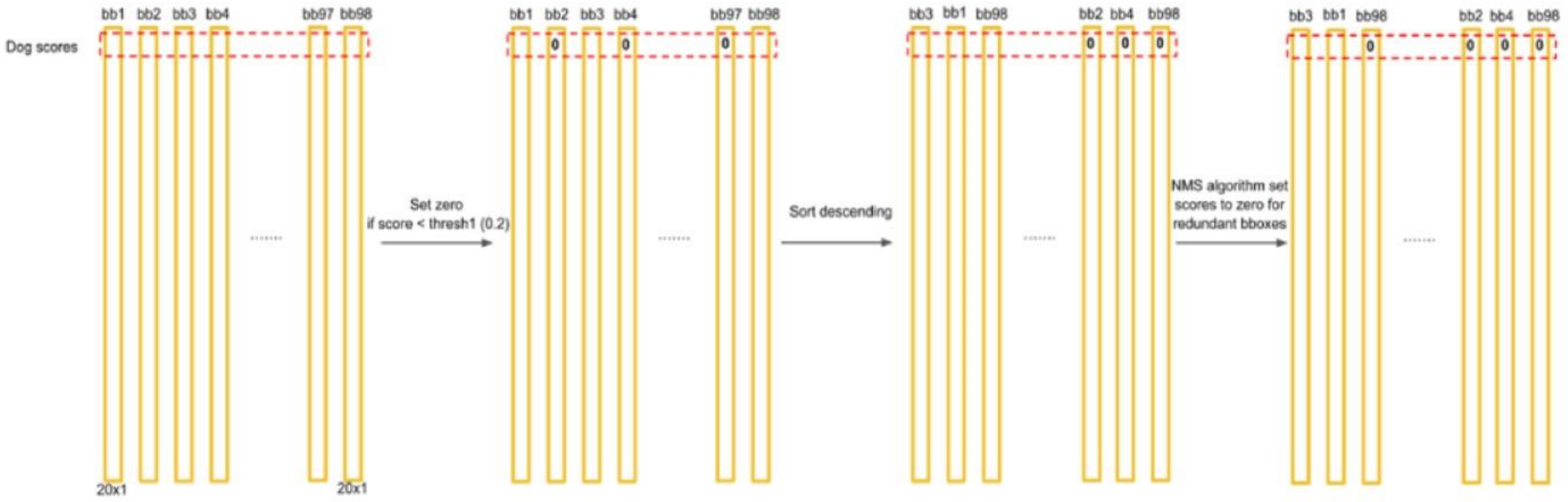




Step 2. 7 x 7 grid cell 2개 bounding box 예측 -> 총 98개의 class- specific confidence score(20x1)

2 Unified Detection

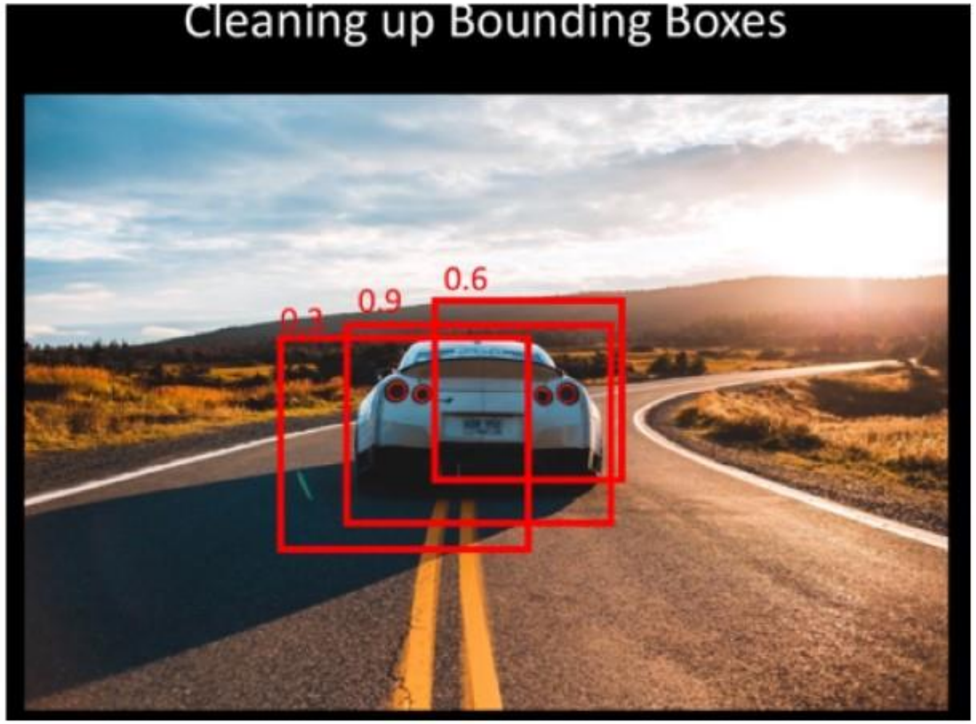
2.3 Inference



Step 3. Threshold 0.2보다 작은 값 -> 0으로 변환, 내림차 순 정렬 후 Non Max Suppression 적용

2 Unified Detection

2.3.Non Max Supression



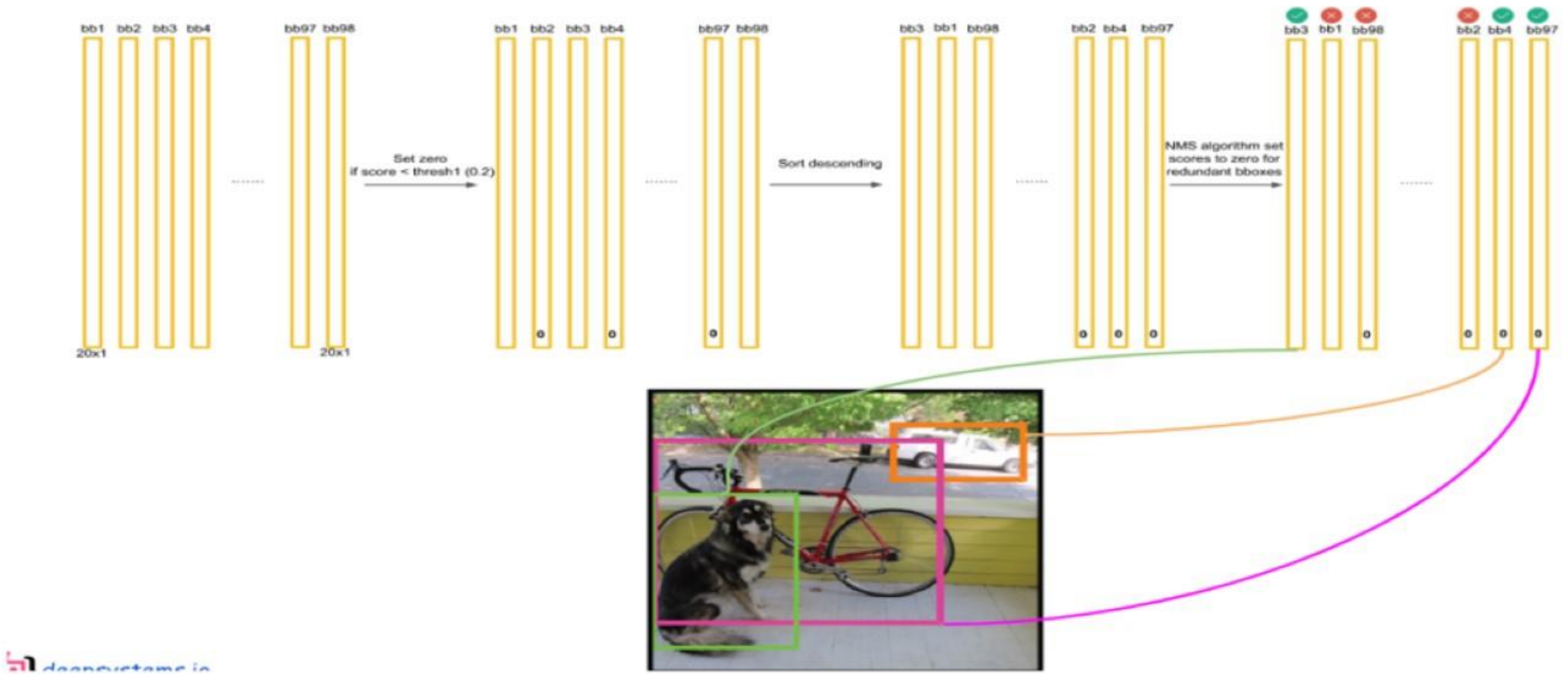
숫자는 물체를 맞추기 위한 확률

가장 높은 IOU값의 박스 고정, IOU값이 0.5 이상인 박스 제거

주변박스 제거를 방지하기 위함

Unified Detection

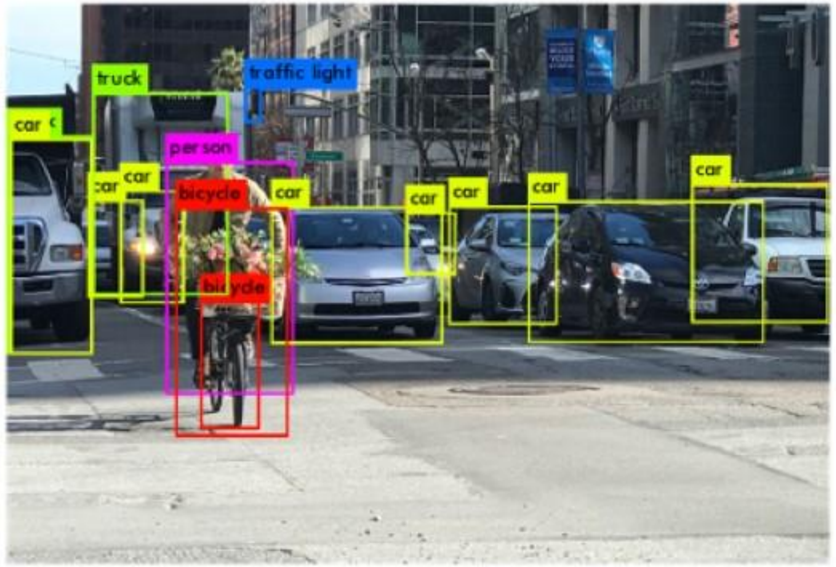
2.3 Inference



최종 output: bounding box가 크게 예측되면서 score >0인 값

2 Unified Detection

2.4 Limitations of YOLO



1. 각 grid cell 마다 b개의 bounding box 추측 -> 공간적 제약
밀집되어 있는 객체 탐지에 대한 정확성 낮아짐

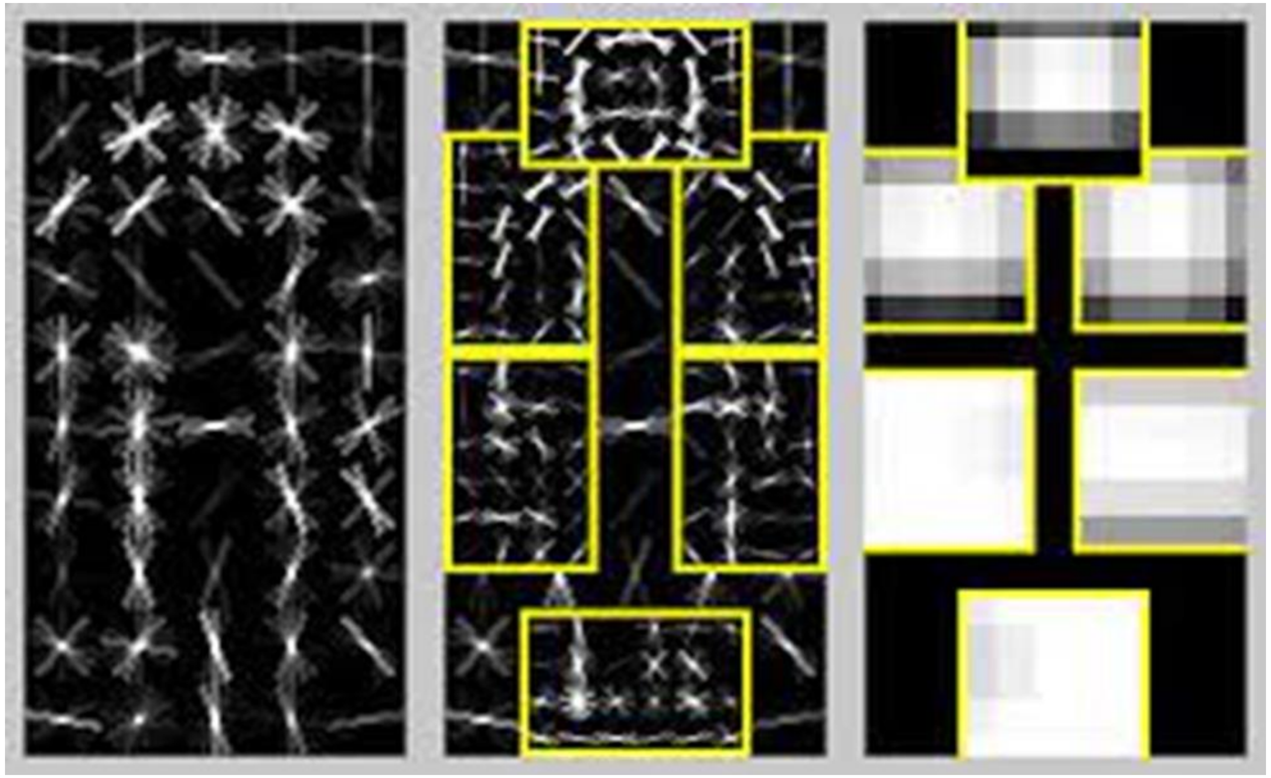
2. 다른 ratio에 대한 물체를 예측 하기 어려움 -> 일반화 어려움

3. 부정확한 localization 문제 -> 작은 bounding box와 큰 bounding box의 loss에 대해 동일한 가중치
크기가 작은 bounding box는 위치가 조금만 달라져도 성능에 큰 영향

Chapter 3. Comparison to Other Detection Systems

3 Comparison to Other Detection Systems

Deformable parts models



분리된 pipeline 이용, 특징 추출(feature extraction), 위치 파악(region classification), bounding box 예측(bounding box prediction) 등을 수행

3 Comparison to Other Detection Systems

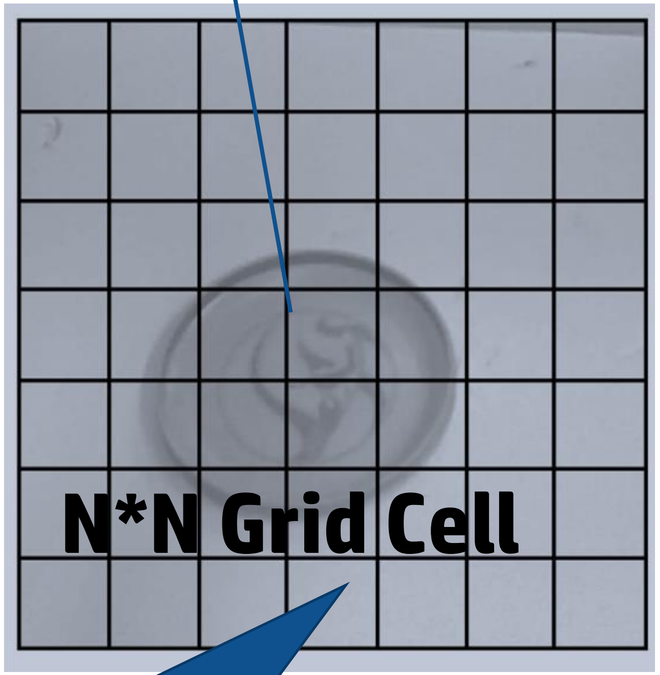
MultiGrasp

D. MultiGrasp Detection

Our third model is a generalization of the first model, we call it MultiGrasp. The preceeding models assume that there is only a single correct grasp per image and try to predict that grasp. MultiGrasp divides the image into an NxN grid and assumes that there is at most one grasp per grid cell. It predicts one grasp per cell and also the likelihood that the predicted grasp would be feasible on the object. For a cell to predict a grasp the center of that grasp must fall within the cell.

[27] J. Redmon and A. Angelova. Real-time grasp detection using convolutional neural networks. *CoRR*, abs/1412.3128, 2014. 5

One single correct grasp
(True grasp)



One Grasp per grid cell
하나의 cell 당 하나의 grasp 존재

Comparison to Other Detection Systems

MultiGrasp

True grasp 가정

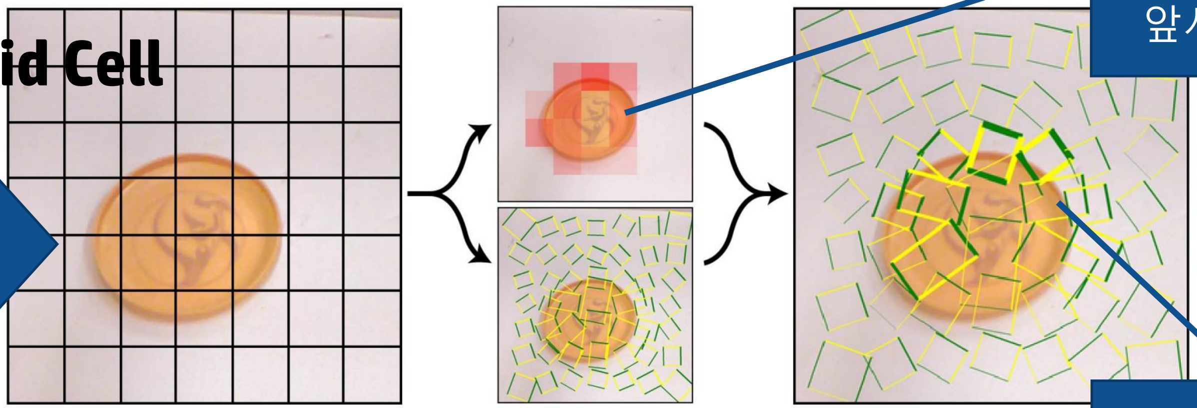


Fig. 4. A visualization of the MultiGrasp model running on a test image of a flying disc. The MultiGrasp model splits the image into For each cell in the grid, the model predicts a bounding box centered at that cell and a probability that this grasp is a true grasp for the image. The predicted bounding boxes are weighted by this probability. The model can predict multiple good grasps for an object, as in thi experiments on the Cornell dataset we pick the bounding box with the highest weight as the final prediction.

각 grid cell 에서 bounding box 예측,
앞서 가정한 **grasp가 true일 확률을 예측**

True grasp 와 관련되어,
부여한 **가중치는 두껍게** 표현됨

1. 매우 간단한 알고리즘

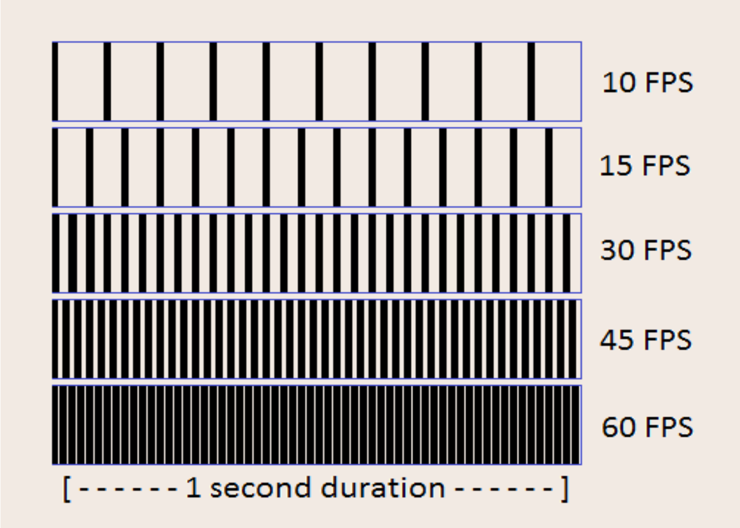
2. 객체 사이즈, 위치, 바운더리 측정 필요X (grasping 하기에 "적절한" 지역을 찾을 뿐)

3. YOLO 또한, 다중 class의 객체들을 탐지하기 위하여 bounding box 와 그 class의 확률을 예측한다.

Chapter 4. Experiments

4 Experiments

4.1. Comparison to Other Real-Time System



FPS
(Frame per Second)

$$\mathrm{mAP} = \frac{1}{N}\sum_{i=1}^{N}\mathrm{AP}_i$$

Mean Average Precision Formula

mAP
(Mean Average Precision)

- 알고리즘 or 모델이 얼마나 " 빠른지" 측정 [속도]
- Object Detection: 1초 당 detection 하는 비율
- 60fps는 초당 지속적으로 바뀌므로, 연속적
- **FPS가 높을 수록, 속도가 빠름**

- Object Detection Model 평가를 위한 지표.
- **각 category(class)의 AP 계산 후, 평균**

4 Experiments

4.1. Comparison to Other Real-Time System

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.

Comparison

[Fast YOLO]

-fps: 155 → Pascal Dataset에서 가장 속도가 빠른 성능을 보임

-mAP: 52.7%

[YOLO] : 실시간 Object Detection

-fps: 45

-mAP: 63.4%

[YOLO VGG-16]

-fps: 21 → YOLO 보다 현저히 낮은 속도

[Fast R-CNN] : 실시간 Object Detection에 부적합

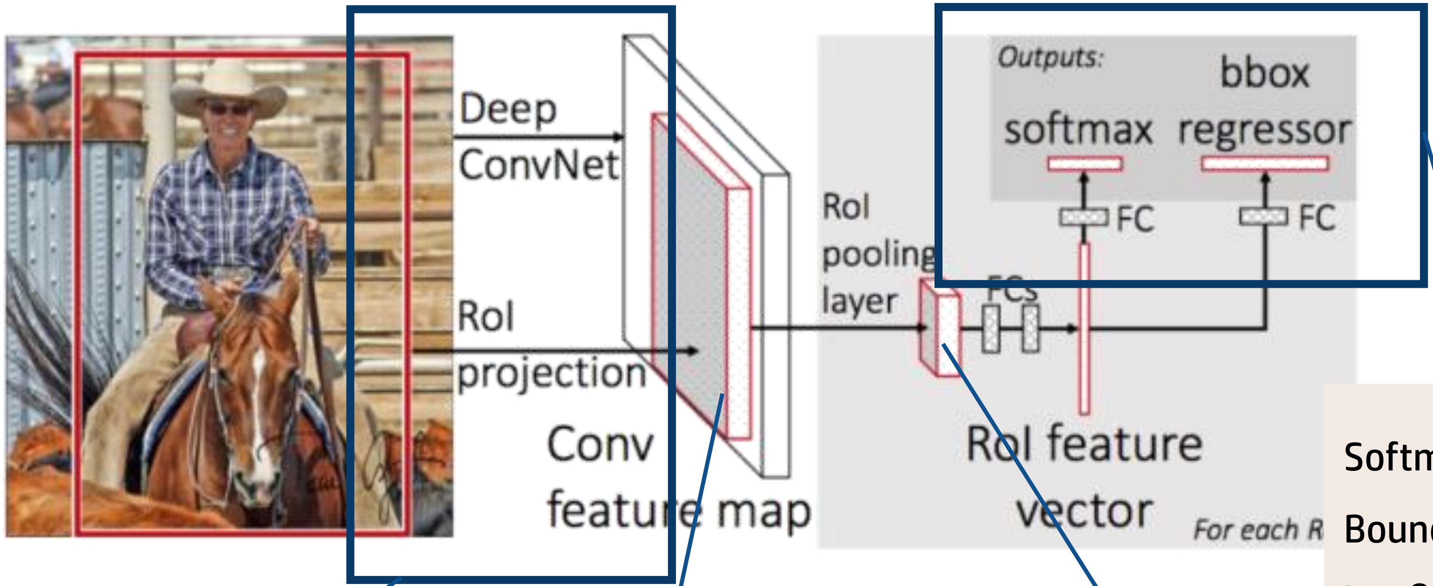
-fps: 0.5 [매우 낮은 속도]

-mAP: 70.0%

4 Experiments

4.1. Comparison to Other Real-Time System

What is Fast R-CNN? +) Fast R-CNN의 속도 성능이 저하되는 이유는?



Fast R-CNN architecture

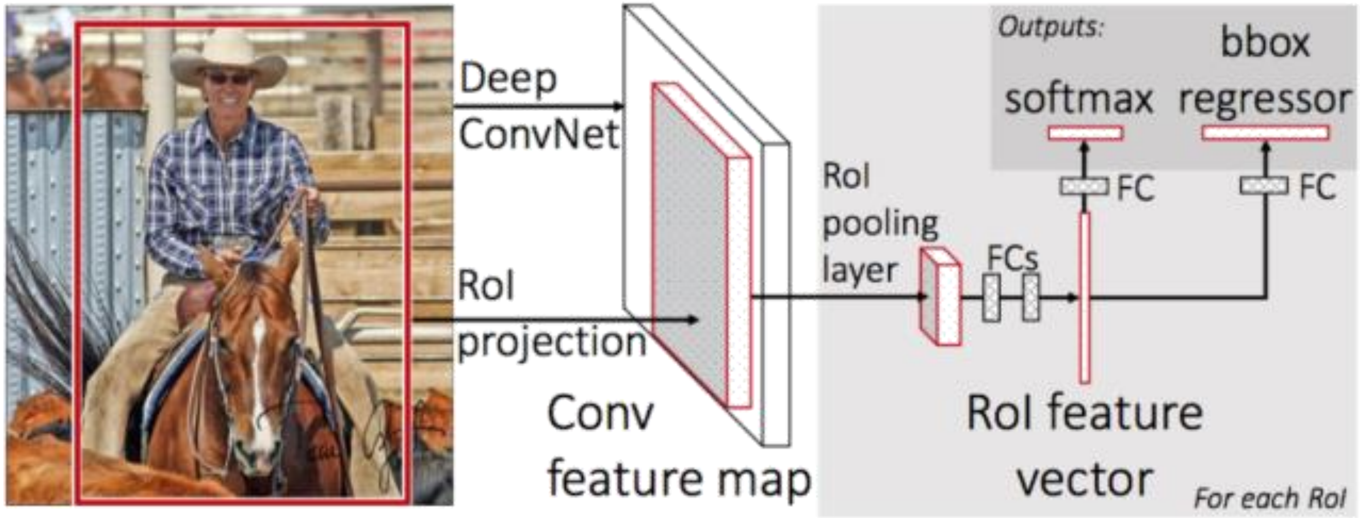
Softmax를 통하여 “분류”를,
Bounding box regression을 통해
box의 위치 조정

1. Selective Search Algo.: RoI 찾는다
2. 원본 이미지를 CNN 에 통과시켜
Feature map 추출

찾은 RoI를 Feature map 사이즈에 맞춰
투영(crop)한다 [Projection]

RoI Pooling을 통하여 고정된 크기의
RoI Feature vector 를 추출하고,

What is Fast R-CNN? +) Fast R-CNN의 속도 성능이 저하되는 이유는?



Fast R-CNN architecture

“Bounding Box를 생성하기 위해 Selective Search 알고리즘 (한 이미지 당 2초) 사용 >> 속도 저하의 원인

“CNN을 1회 사용하기 때문에, “분류” 단계의 처리 속도를 줄일 수 있다.

4 Experiments

4.1. Comparison to Other Real-Time System

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.

Comparison

{2 Versions of Faster R-CNN}

[Faster R-CNN VGG-16]

-fps: 7 (YOLO보다 6배 느리다)

-mAP: 73.2% (+10% than ZF)

[Faster R-CNN ZF(Zeller-Fergus)]

-fps: 18 (YOLO 보다 2.5배 느리다)

-mAP: 62.1% (inappropriate)

4 Experiments

4.2. VOC 2007 Error Analysis

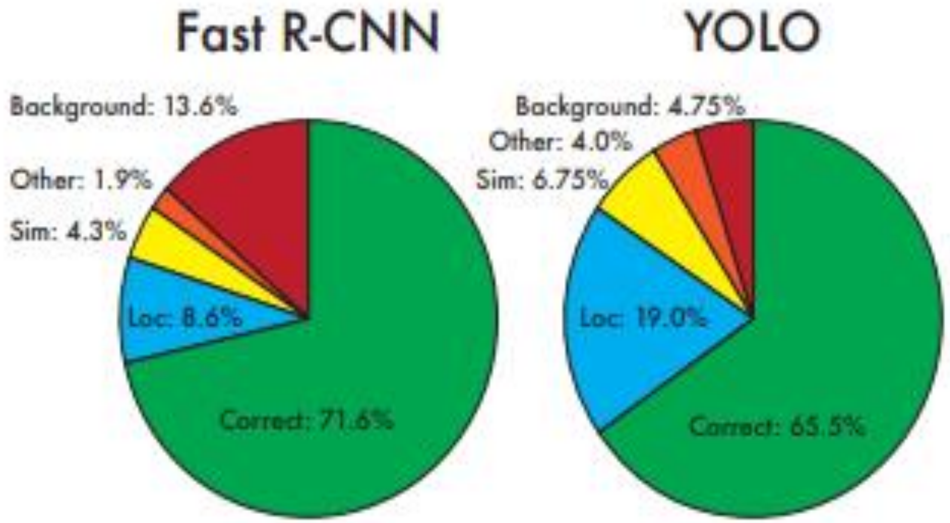


Figure 4: Error Analysis: Fast R-CNN vs. YOLO These charts show the percentage of localization and background errors in the top N detections for various categories (N = # objects in that category).

“

20개의 카테고리에 대하여 각 error type(background, Localization etc.) 들을 종합하여 평균을 낸 도표

4 Experiments

4.2. VOC 2007 Error Analysis

What is PASCAL Dataset?

- The VOC2012 Challenge
- The VOC2011 Challenge
- The VOC2010 Challenge
- The VOC2009 Challenge
- The VOC2008 Challenge
- The VOC2007 Challenge
- The VOC2006 Challenge
- The VOC2005 Challenge

PASCAL Visual Object Classes Challenge 2007

어떤 class에 속하는지(class recognition) 분류하기 위하여, 표준화된 이미지 데이터를 제공

[Object Detection] & [Semantic segmentation] & [Classification] 에 사용 → Widely Used

4 Experiments

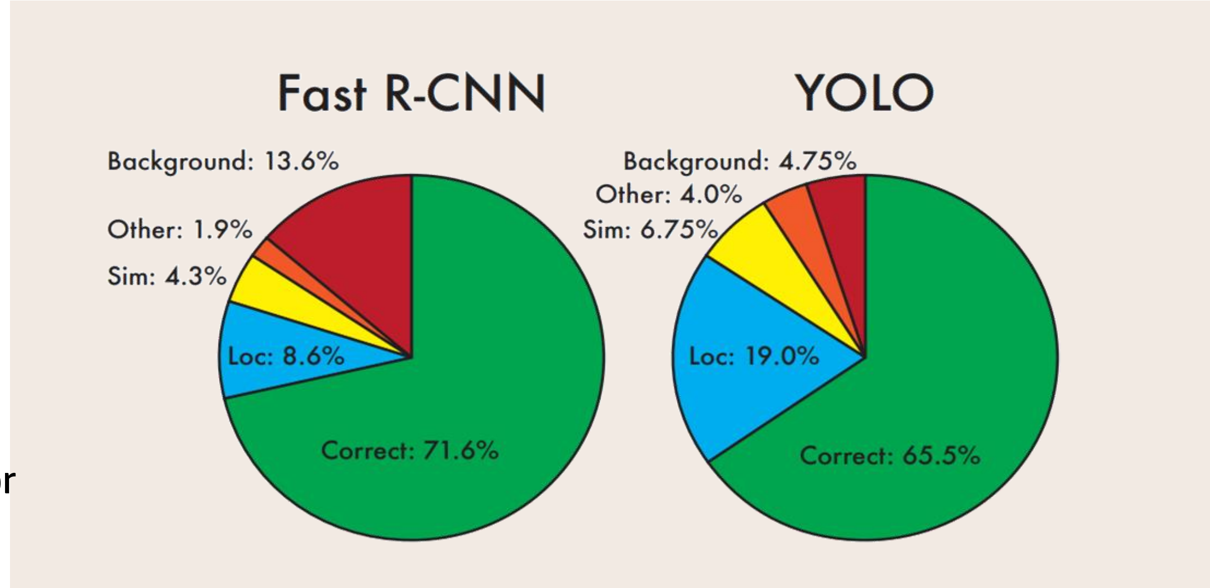
4.2. VOC 2007 Error Analysis

13.6%

Background Error

8.6%

Localization Error



4.75%

Background Error

19.0%

Localization Error

Fast R-CNN

YOLO

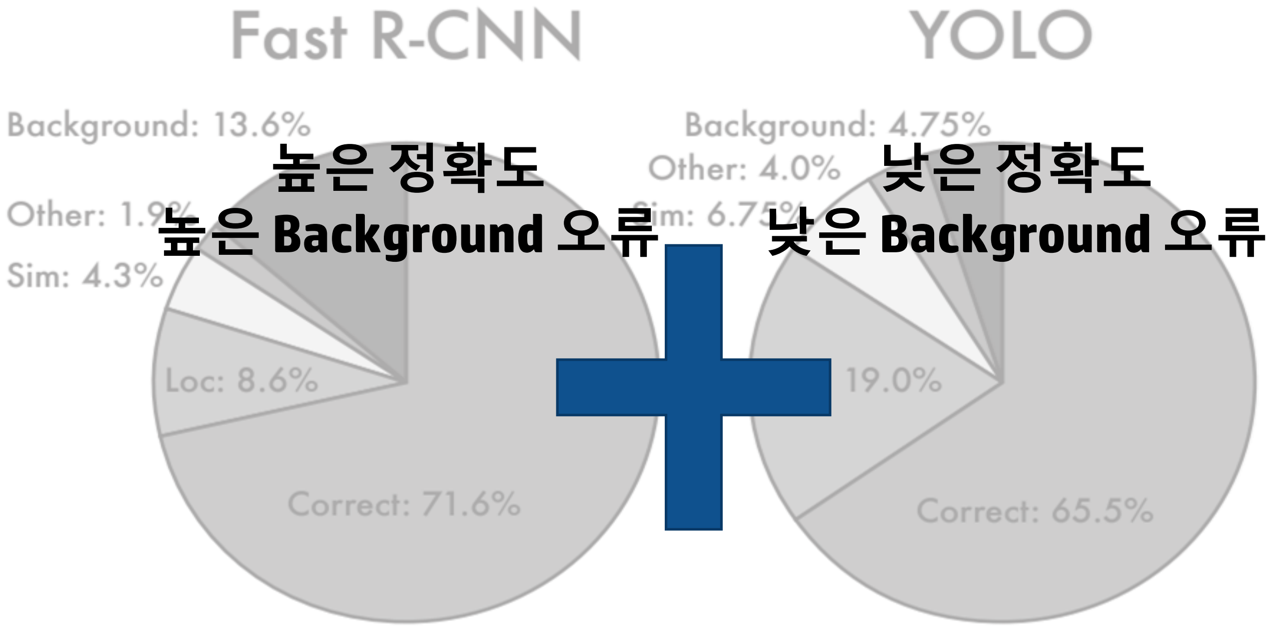
[거짓 양성]

Background를 올바르게 탐지하지만,

거짓된 정보(불필요한 정보)

YOLO보다 **3배** 더 잘 감지

4.3. Combining Fast R-CNN and YOLO



[성능 가속화(significant boost)]

높은 정확도
낮은 Background 오류

4 Experiments

4.3. Combining Fast R-CNN and YOLO

	mAP	Combined	Gain
Fast R-CNN	71.8	-	-
Fast R-CNN (2007 data)	**66.9**	72.4	.6
Fast R-CNN (VGG-M)	59.2	72.4	.6
Fast R-CNN (CaffeNet)	57.1	72.1	.3
YOLO	63.4	**75.0**	**3.2**

Table 2: Model combination experiments on VOC 2007. We examine the effect of combining various models with the best version of Fast R-CNN. Other versions of Fast R-CNN provide only a small benefit while YOLO provides a significant performance boost.

Fast R-CNN의 mAP: 71.8%

[Fast R-CNN + YOLO] 의 mAP: +3.2% → 75%

[Fast R-CNN+other version of Fast R-CNN] 의 mAP:

+0.3%, +0.6% → 약 72.3% (Just small benefit)

4 Experiments

4.4. VOC 2012 Results

- *Person:* person
- *Animal:* bird, cat, cow, dog, horse, sheep
- *Vehicle:* airplane, bicycle, boat, bus, car, motorbike, train
- *Indoor:* bottle, chair, dining table, potted plant, sofa, tv/monitor

4 Experiments

4.4. VOC 2012 Results

VOC 2012 test	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
MR_CNN_MORE_DATA [11]	**73.9**	**85.5**	**82.9**	**76.6**	**57.8**	**62.7**	**79.4**	77.2	86.6	**55.0**	**79.1**	**62.2**	87.0	**83.4**	**84.7**	78.9	45.3	73.4	65.8	80.3	74.0
HyperNet_VGG	71.4	84.2	78.5	73.6	55.6	53.7	78.7	**79.8**	87.7	49.6	74.9	52.1	86.0	81.7	83.3	**81.8**	**48.6**	**73.5**	59.4	79.9	65.7
HyperNet_SP	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO	70.7	83.4	78.5	73.5	55.8	43.4	79.1	73.1	**89.4**	49.4	75.5	57.0	**87.5**	80.9	81.0	74.7	41.8	71.5	68.5	**82.1**	67.2
MR_CNN_S_CNN [11]	70.7	85.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
N [28]	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
COCO	70.1	84.0	79.4	71.6	51.9	51.1	74.1	72.1	88.6	48.3	73.4	57.8	86.1	80.0	80.7	70.4	46.6	69.6	**68.8**	75.9	71.4
NoC [29]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
Fast R-CNN [14]	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	**87.5**	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
UMICH_FGS_STRUCT	66.4	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2
NUS_NIN_C2000 [7]	63.8	80.2	73.8	61.9	43.7	43.0	70.3	67.6	80.7	41.9	69.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3
BabyLearning [7]	63.2	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6
NUS_NIN	62.4	77.9	73.1	62.6	39.5	43.3	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7
R-CNN VGG BB [13]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3
R-CNN VGG [13]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7
YOLO	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
Feature Edit [33]	56.3	74.6	69.1	54.4	[illegible]	33.1	65.2	[illegible]	69.7	30.8	56.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8
R-CNN BB [13]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1
SDS [16]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7
R-CNN [13]	49.6	68.1	63.8	46.1	[illegible]	27.9	56.6	[illegible]	65.9	26.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6

2.3% 성능 향상
5순위 up

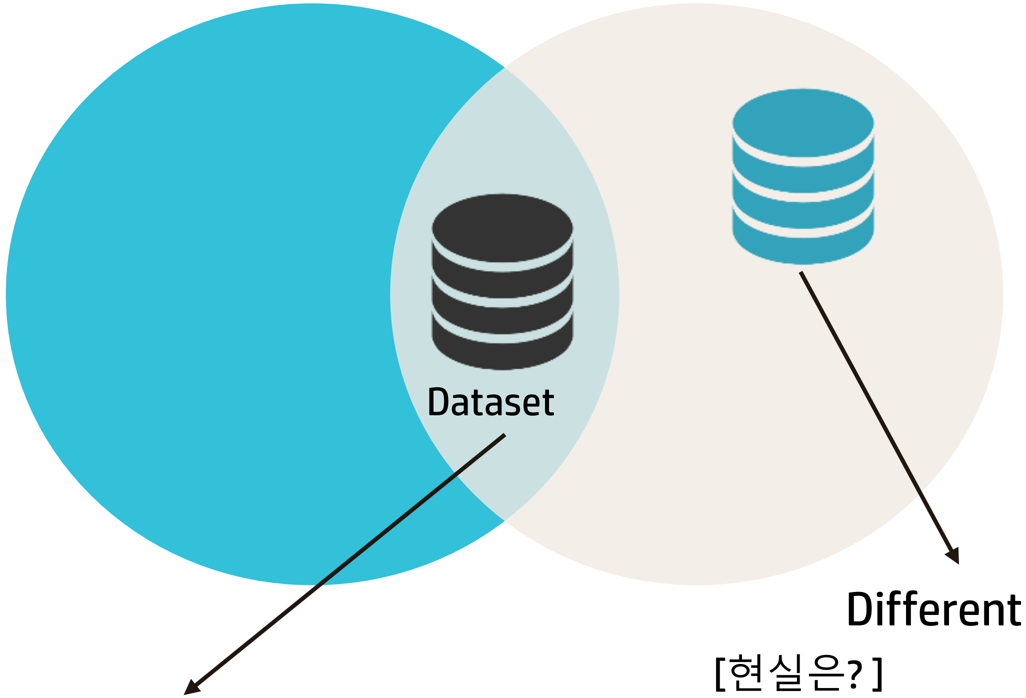
Low Performance (8-10%)

Small Objects

Big Objects

High Performance

4.5. Generalizability: Person Detection in Artwork



Different Distribution

[현실은?]

⇒ AI 모델이 모든 case(사건) 를 예측하기란 매우 어려운 일

⇒ Test data 가 처음 보는 data일 수도, 혹은 기계가 전혀 예상하지 못했던 data일 수도 있기 때문

Same Distribution

성능이 잘 나올 수 밖에 없는 이유

Reality; 실생활, 실무

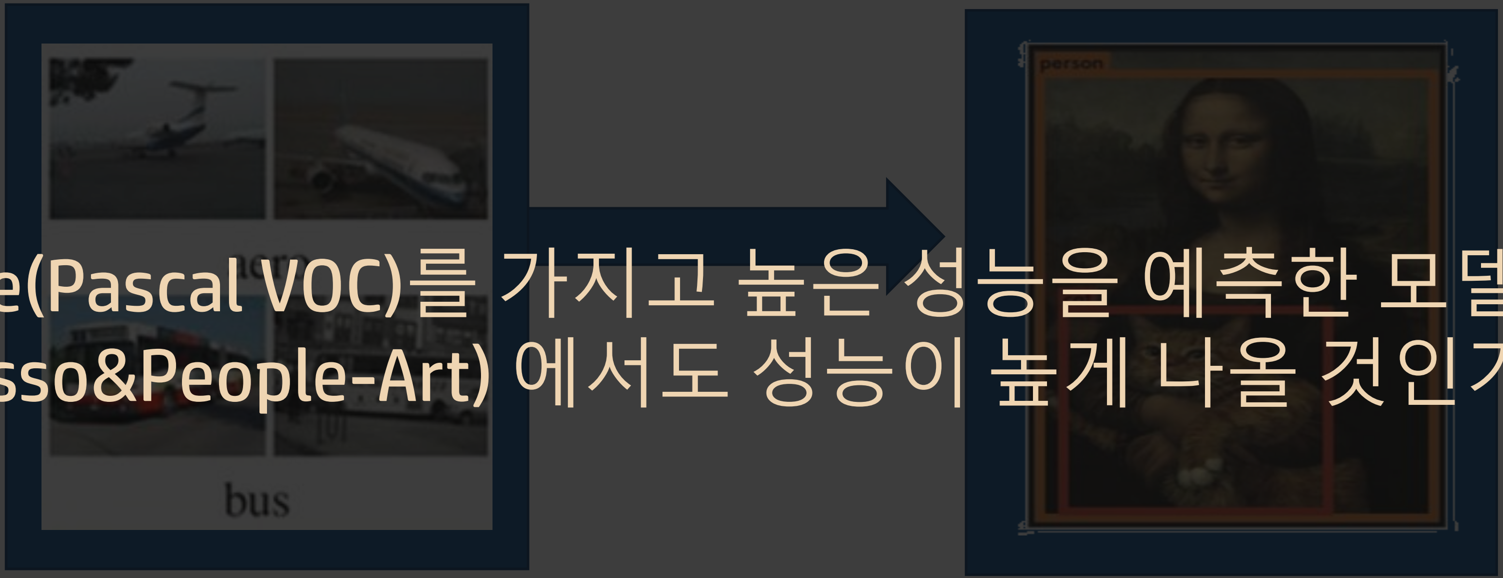
VS

Academic; 이론

VOC 2007의 결과를 Artwork에 적용해도 적합한가

4.5. Generalizability: Person Detection in Artwork

VOC 2007 Dataset 을 이용해 추출한 결과를 Artwork에 적용해도 적합한가



Natural image(Pascal VOC)를 가지고 높은 성능을 예측한 모델이, 과연
Artwork(Picasso&People-Art) 에서도 성능이 높게 나올 것인가 (일반화 문제)

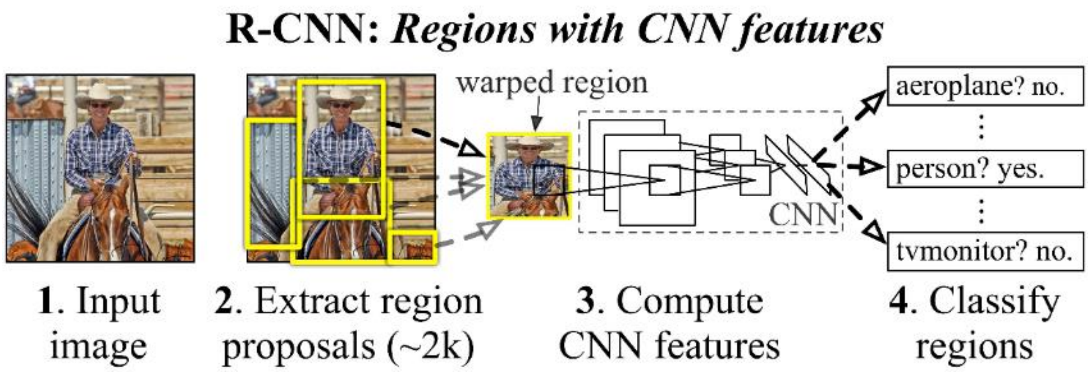
‘Natural Image’
Pascal VOC 2007 Dataset

‘Artwork’
Picasso&People-Art
Dataset

4 Experiments

4.5. Generalizability: Person Detection in Artwork

	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	



R-CNN

Pascal VOC 2007 Data(Natural Image): 54.2% AP

Picasso&People-Art Data(Artwork): 10.4%>26% AP

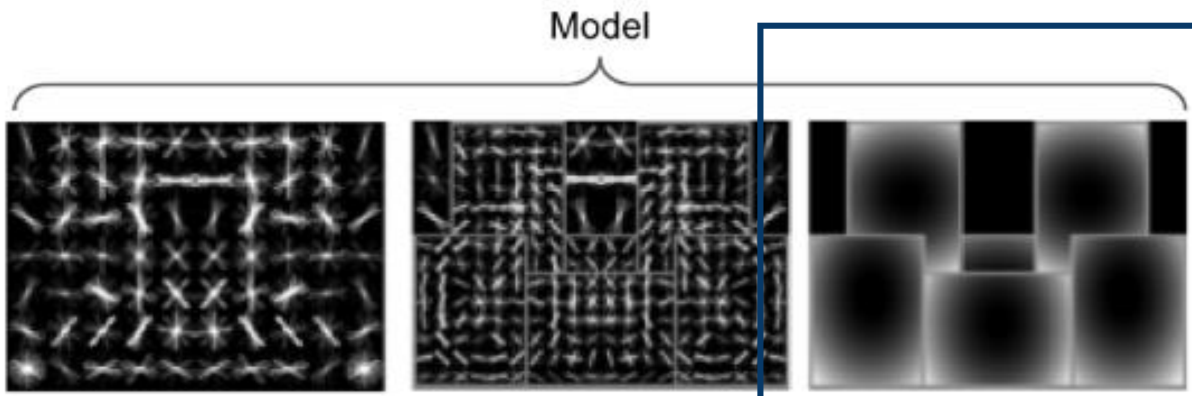
Selective Search 알고리즘으로 Bounding Box 추출

→ Natural Image에 튜닝된 알고리즘

4 Experiments

4.5. Generalizability: Person Detection in Artwork

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



(a) Root filter

(b) Part filters in higher resolution

(c) A spatial model for part locations

DPM

Pascal VOC 2007 Data(Natural Image): 43.2% AP

Picasso&People-Art Data(Artwork): 37.8%>32% AP

[Spatial Model (공간 모델)]

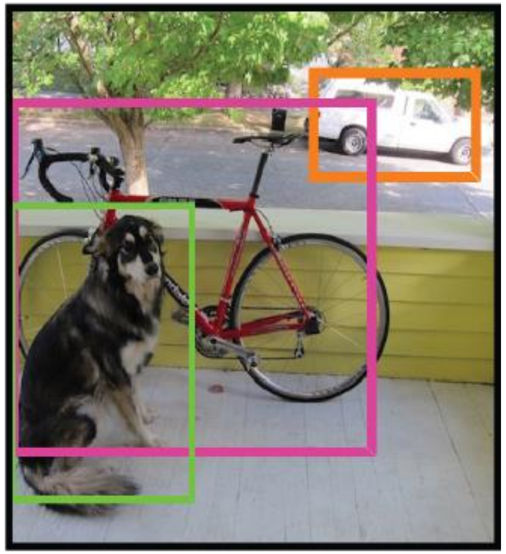
객체의 레이아웃, 모양 (layout and shapes)을 캐치하는데 강하기 때문 (공간의 관계성에 핵심)

→ Natural Image, Artwork 에 큰 영향을 미치지는 않음

4 Experiments

4.5. Generalizability: Person Detection in Artwork

	VOC 2007	Picasso		People-Art
	AP	AP	Best F_1	AP
YOLO	$\boxed{\textbf{59.2}}$	**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	



YOLO

Pascal VOC 2007 Data(Natural Image): 59.2% AP

Picasso&People-Art Data(Artwork): 53.5%>45% AP

객체의 레이아웃, 모양 (layout and shapes)을 모형화

→ 객체 간 관계성과 공통적으로 나타나는 지역 고려

Chapter 5. Real-Time Detection In The Wild

5 Real-Time Detection In The Wild

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: 빠르고 정확한 Object Detection (객체 탐지)
- 웹캠과 YOLO 모델을 연결하여, 실시간으로 객체 탐지 수행하는 기능이 사용

  → [실시간?]: 카메라를 통하여 사물을 잡아내고, 감지하는 시간 포함.
- 웹캠과 연결될 경우: Tracking System 과 같은 기능을 한다 [카메라의 이동으로, 탐지할 객체가 지속적으로 달라짐]

THANK YOU