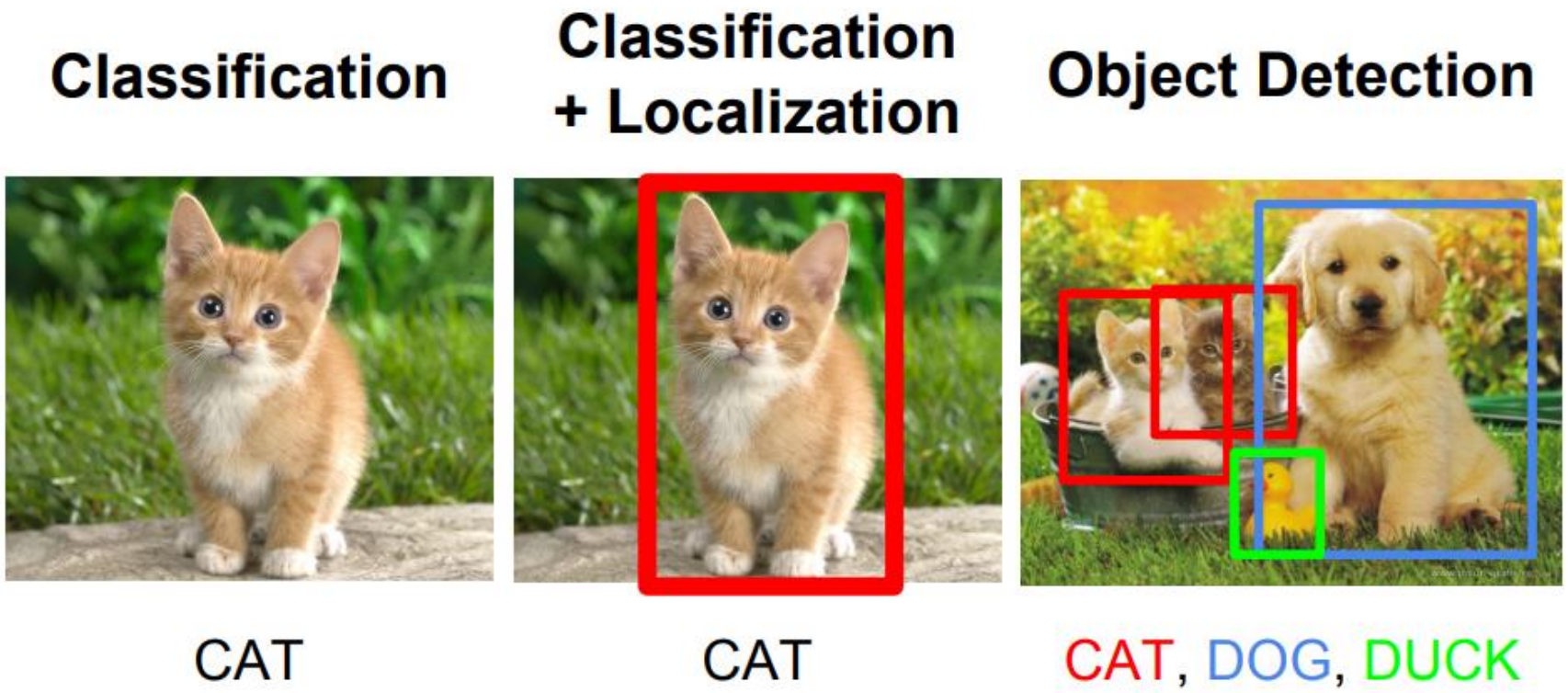
You Only Look Once

Unified, Real-Time Object Detection

발표자 한나경

Contents

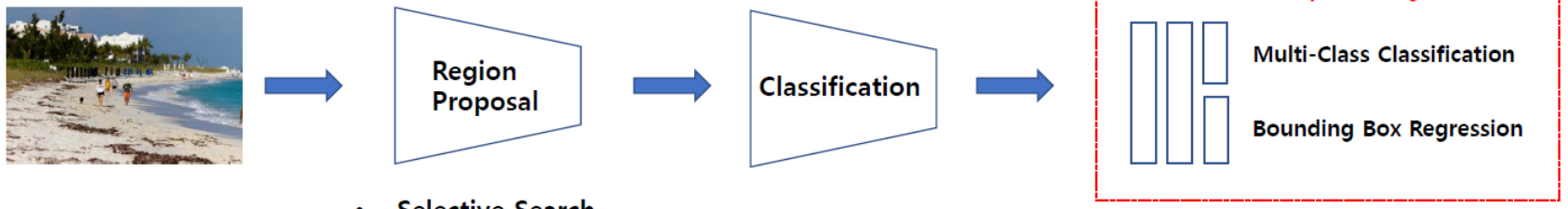
Object Detection



Object Detection

2-stage object detector

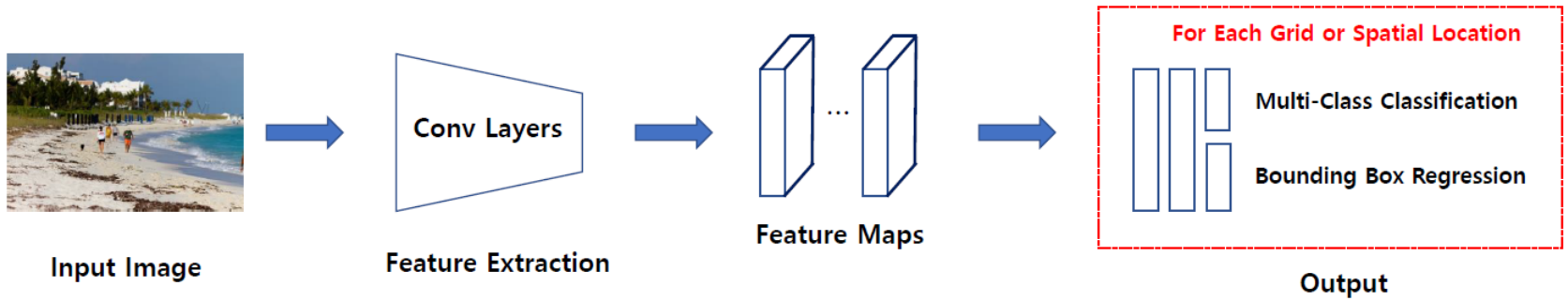
- Localization -> Classification 순차적으로 진행
- Object가 있을 만한 위치를 제안하는 Region Proposal 과정을 거침
- 정확도 높음, 속도 느림
- Fast R-CNN



- Selective Search
- Region Proposal Network
- etc.

1-stage object detector

- Localization, Classification 동시에 수행
- 이미지 내의 모든 위치를 Object의 잠재 영역으로 봄
- 정확도 낮음, 속도 빠름
- YOLO v1



Abstract

전체 이미지를 보는 횟수가 1회

You Only Look Once

Unified, Real-Time Object Detection

1-stage object detector 방식 사용

Localization & Classification 단계 단일화

속도 측면에서 개선된 모델

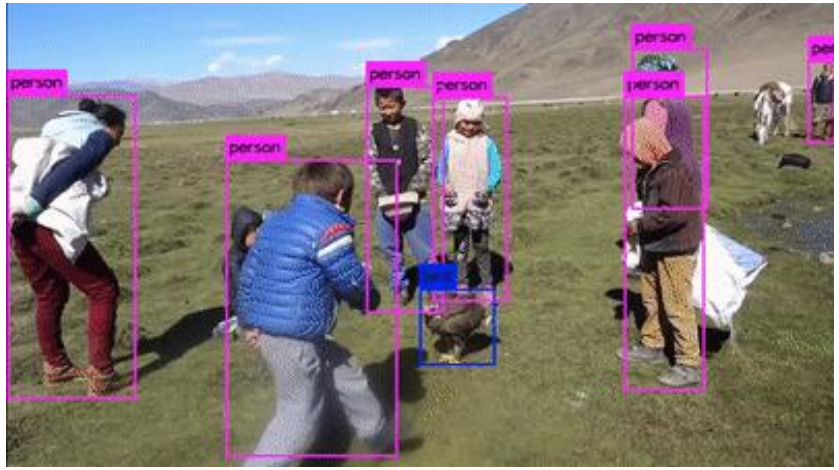
Introduction

Advantage

1. **속도 향상** : Object detection을 regression problem으로 관점 전환 -> 한번에 처리하는 end-to-end 구조
2. **Background error 감소** : 학습 과정에서 이미지 전체를 보고 예측하므로 Background error 감소
3. **다른 도메인에서도 Object detection 성능 우수**

Disadvantage

1. **정확도 감소** : 빠르게 객체를 탐지할 수 있다는 장점이 있지만 정확성이 떨어짐, 특히 작은 물체에 대한 정확도 감소, 속도와 정확도 반비례



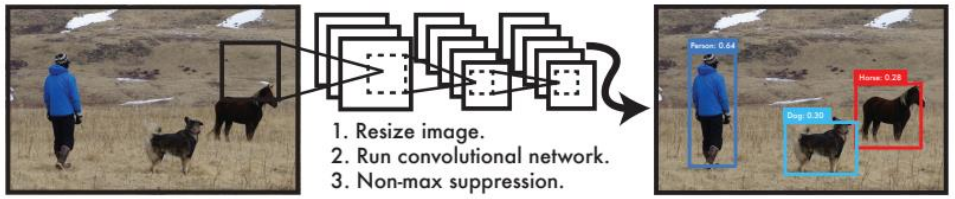


Figure 1: The YOLO Detection System. Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448 × 448, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.

Unified Detection

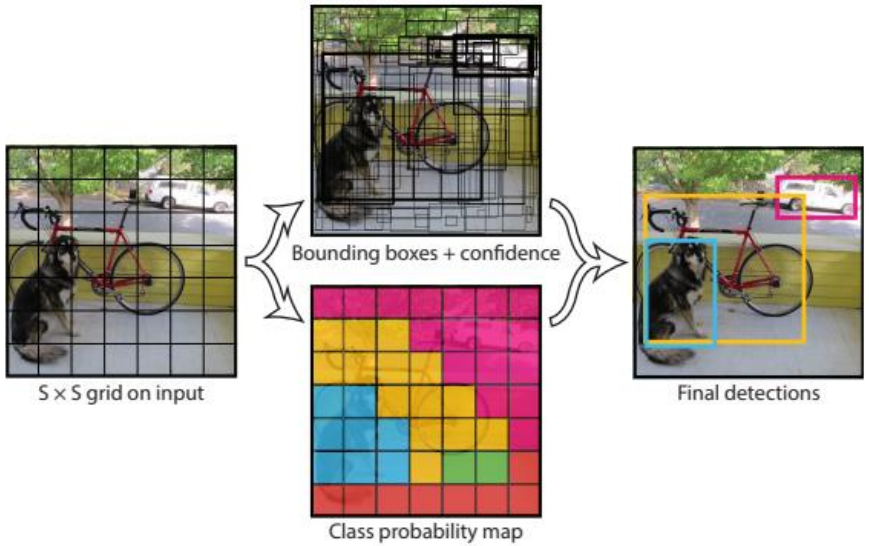


Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

For evaluating YOLO on PASCAL VOC, we use $S = 7$, $B = 2$. PASCAL VOC has 20 labelled classes so $C = 20$. Our final prediction is a $7 \times 7 \times 30$ tensor.

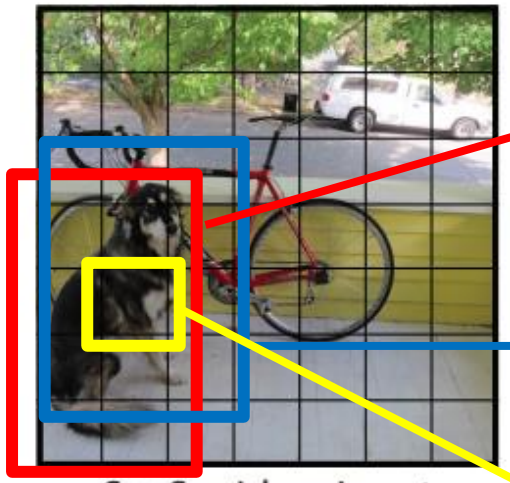
- **Region proposal, feature extraction, classification, bbox regression -> 1-stage detection로 통합**
- **이미지 전체로부터 얻은 feature를 활용하여 bbox 예측 & 모든 클래스에 대한 확률 계산**
- **SxS grid cell -> each grid cell, B bbox prediction + confidence & class probabilities -> S x S x (B * 5 + C)**

Unified Detection

YOLO on PASCAL VOC
S = 7, B = 2, C = 20

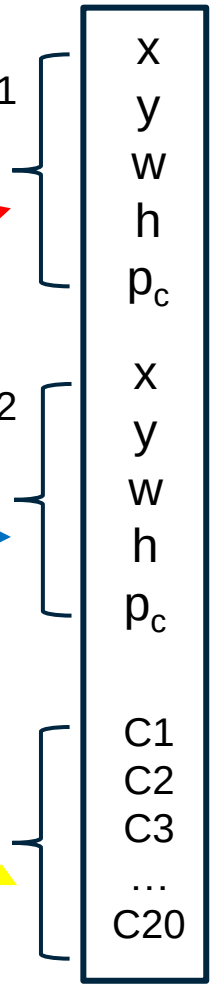
S × S grid on input

7 x 7 grid로 분할



S × S grid on input

Grid cell마다 bbox 2개씩 예측



Bbox의 중심좌표의 위치
(grid cell 기준)

Input image의 W, H로 normalize

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

Pr(Object) : object가 bbox에
존재하면 1, 존재하지 않으면 0

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

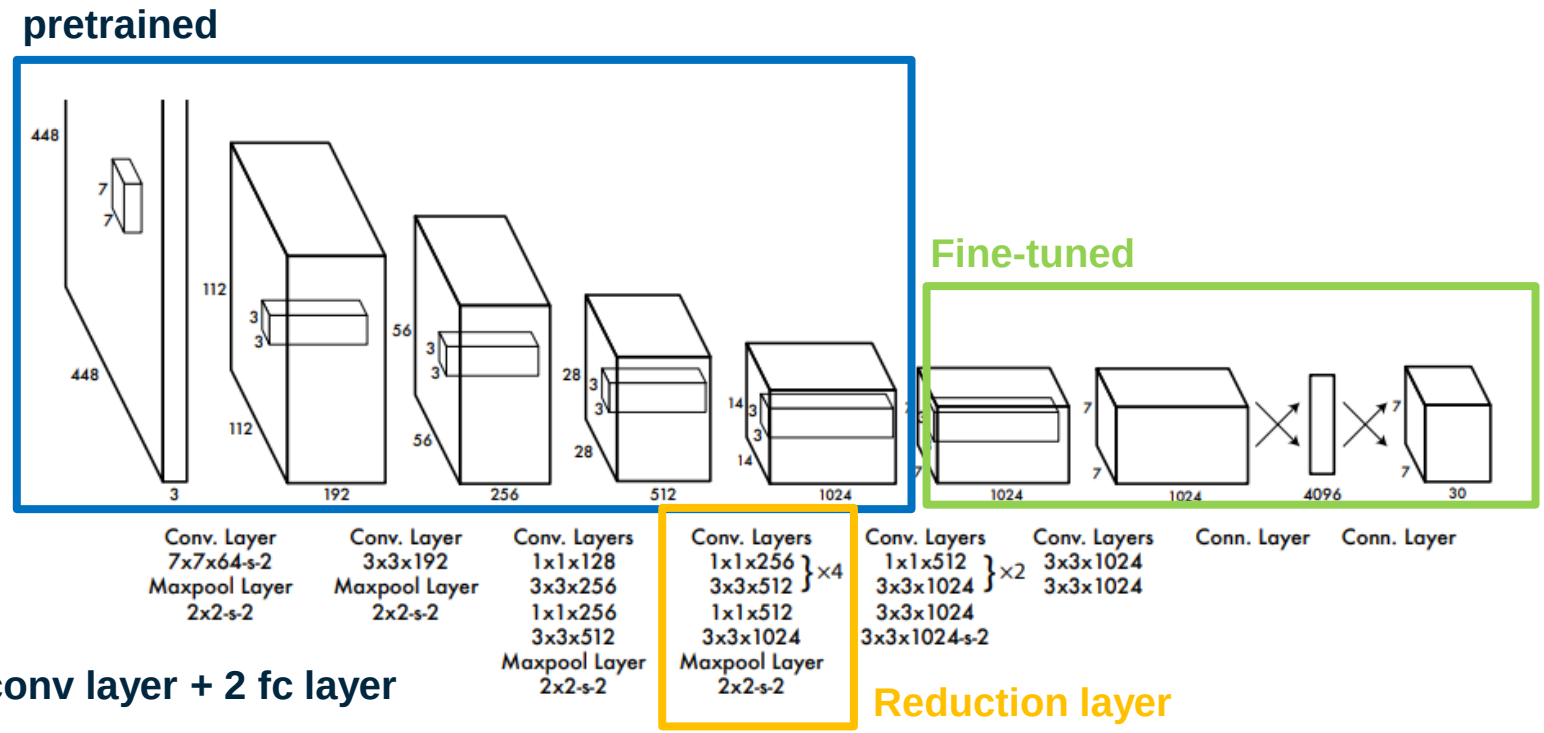
: 물체가 bbox 내에 있을 때,
grid cell에 있는 object가
i번째 class에 속할 확률

SxS grid cell -> S x S x (B * 5 + C)

Output tensor : 7 x 7 x (2 * 5 + 20)

* IoU : Intersection over Union의 약자로 실제 객체의 bbox와 예측한 bbox가 얼마나 일치하는지를 나타낸다.

Network Design



- 24 conv layer + 2 fc layer

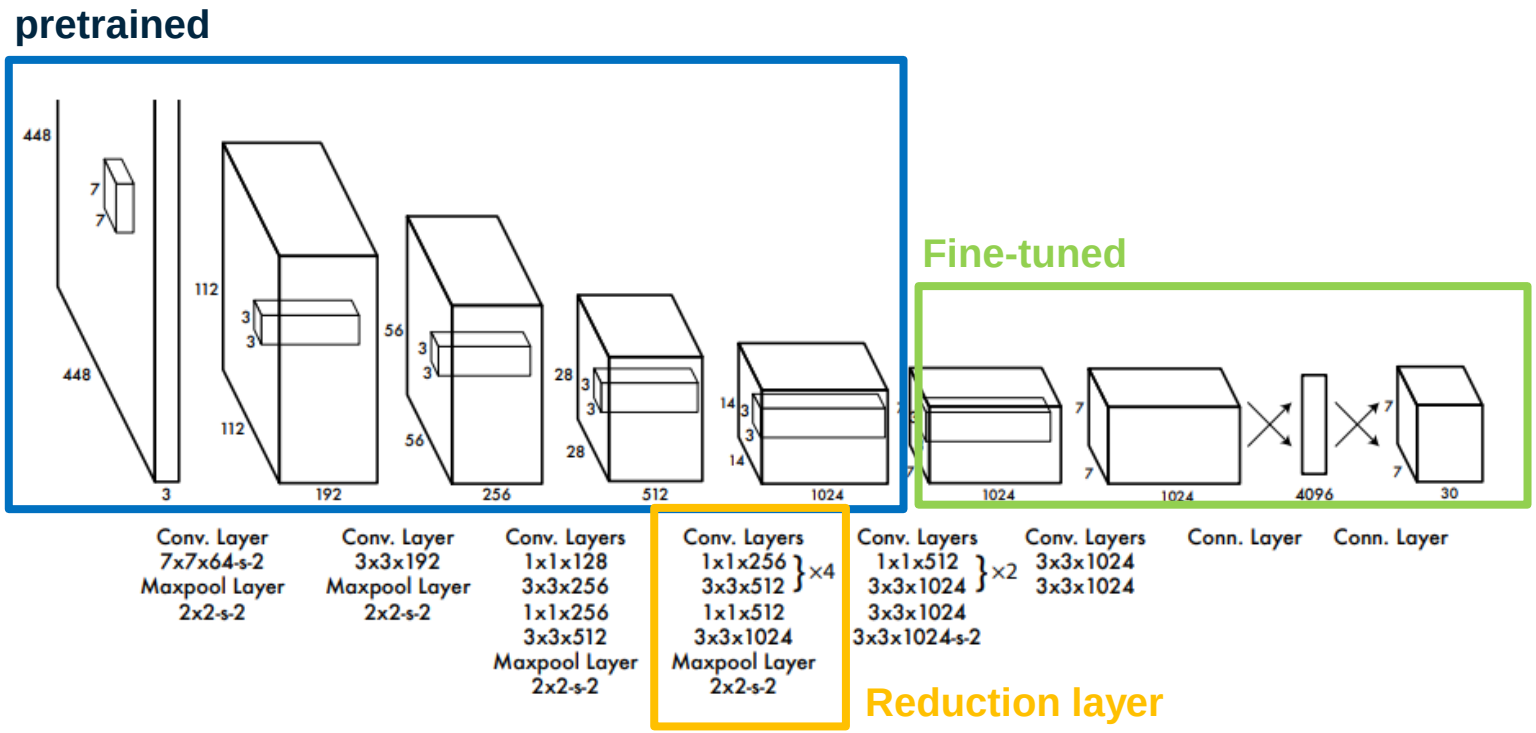
Reduction layer

- 20 conv layer : pretrained with 1000-class ImageNet (input image : 224 x 224)

- 4 conv layer + 2 fc layer : detection 수행 (input image : 448 x 448)

- 중간에 1 x 1 reduction layer로 연산량 감소
- 최종 Output : class의 확률 값, bbox의 위치 정보

Network Design



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

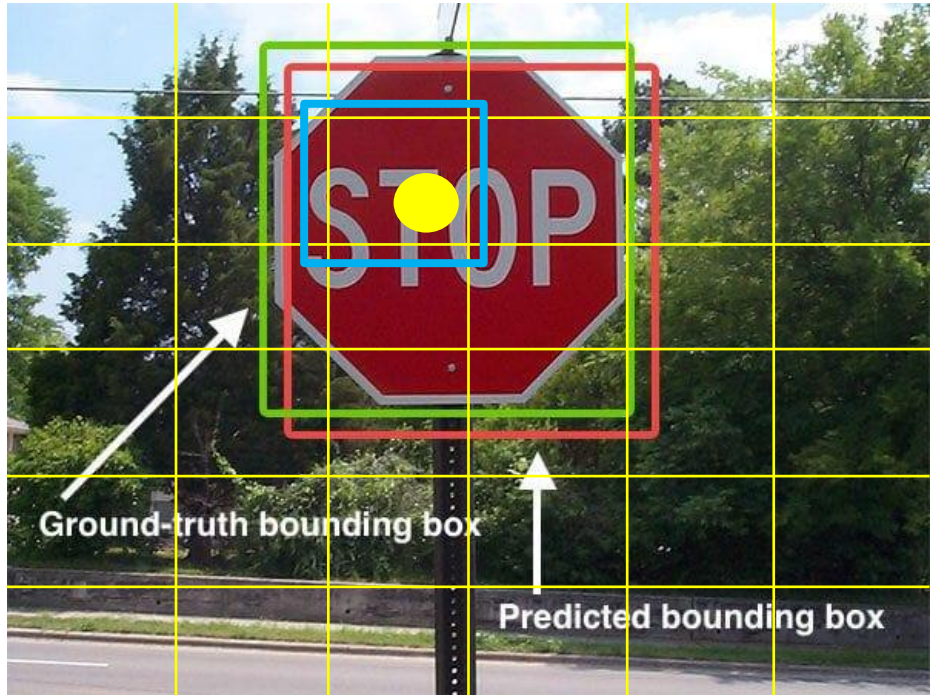
leaky ReLU

마지막 계층 제외 leaky ReLU

마지막 계층 : linear activation function h(x) = cx

Training

특정 object에 responsible한 cell i는 GT box의 중심이 위치하는 cell로 할당



Groundtruth

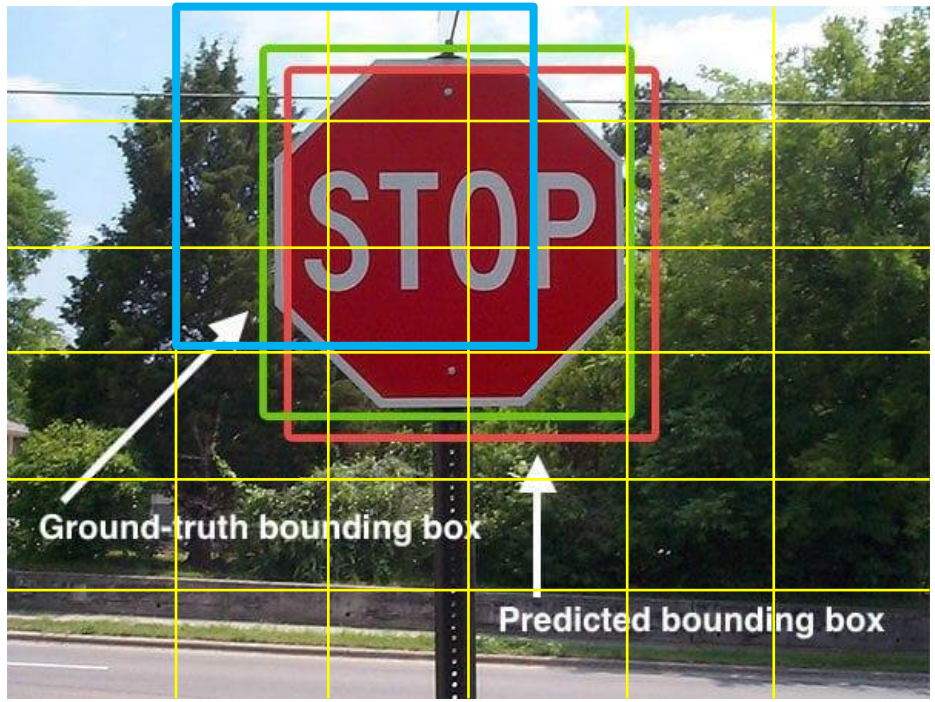
GT box내에 존재하는 Object에 responsible한 cell

Training

특정 object에 responsible한 cell i는 GT box의 중심이 위치하는 cell로 할당

YOLO는 여러 bbox를 예측하지만, 학습단계에서는 $\mathrm{IOU}_{\mathrm{pred}}^{\mathrm{truth}}$ 가장 높은 bbox 1개만 사용

-> $\mathbb{1}_{ij}^{\mathrm{obj}}$ 로 cell i에서 responsible한 j번째 bbox를 표시하여 loss function에 반영



Groundtruth

예측한 blue box

예측한 red box

Training

loss function:

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

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

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

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

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

$\mathbb{1}_{ij}^{\text{obj}}$ Cell i의 j번째 bbox가 responsible (highest IoU), (1 or 0)

$\mathbb{1}_{i}^{\text{obj}}$ Cell i에 object 존재 여부 (1 or 0)

λ_{coord} 5, bbox coordinates loss

λ_{noobj} 0.5, 객체 없는 박스

1) Object 존재하는 cell i의 j번째 bbox predictor에 대해 x, y loss
2) Object 존재하는 cell i의 j번째 bbox predictor에 대해 w, h loss
3) Object 존재하는 cell i의 j번째 bbox predictor에 대해 confidence score의 loss
4) Object 존재하지 않는 cell i의 j번째 bbox predictor에 대해 confidence score의 loss
5) Object 존재하는 cell i의 conditional class probability loss

Part 3.3

Inference

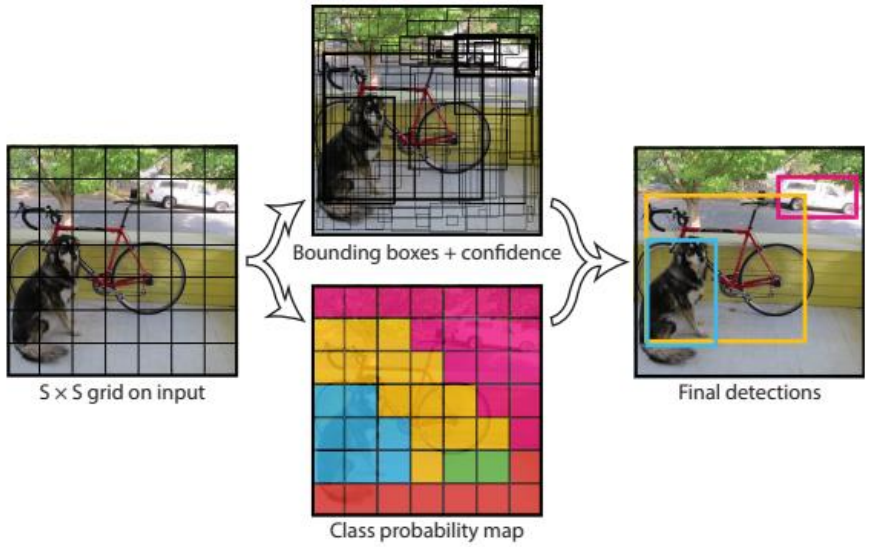


Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

For evaluating YOLO on PASCAL VOC, we use $S = 7$, $B = 2$. PASCAL VOC has 20 labelled classes so $C = 20$. Our final prediction is a $7 \times 7 \times 30$ tensor.

- **7 x 7 x 2개 bbox 예측, 각 bbox마다 class probability 계산**
- **Non-Maximum Suppression(NMS) 적용**

-> 다중검출 개선 : 하나의 **object**에 대해 여러 **cell**이 검출하는 문제 개선

- **mAP 2-3% 향상**

Experiments

4.1. Comparison to Other Real-Time Systems

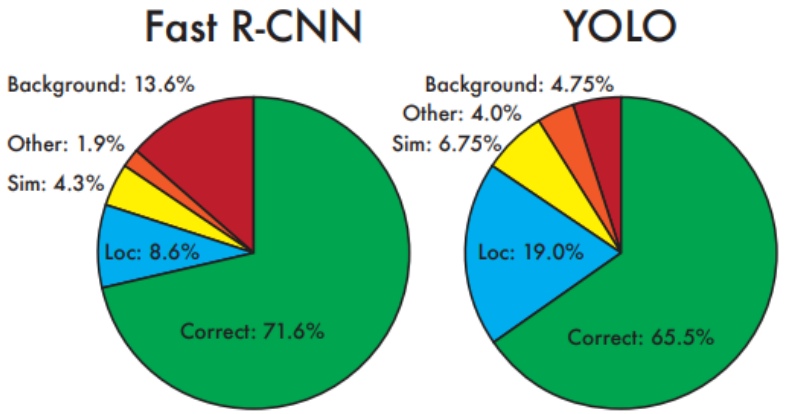
Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012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

mAP : mean average precision -> 정확도
FPS : Frame Per Second -> 속도

다른 Real-Time Detectors에 비해 정확도 높음, 속도 빠름

R-CNN 계열은 정확도 높음 but 속도 느림

4.2. VOC 2007 Error Analysis
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**

백그라운드 error의 감소

Fast R-CNN과 YOLO의 결합시 mAP이 3.2% 향상

Experiments

4.4. VOC 2012 Results

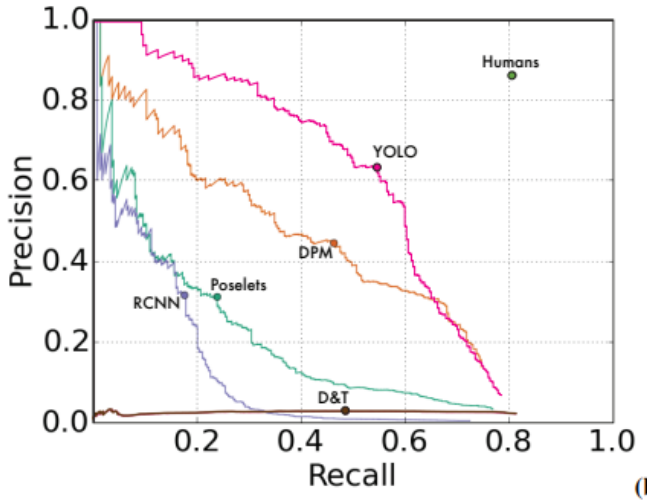
VOC 2012 test	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
MR_CNN_MORE_DATA [11]	**73.9**	**85.5**	**82.9**	**76.6**	**57.8**	**62.7**	**79.4**	77.2	86.6	**55.0**	**79.1**	**62.2**	87.0	**83.4**	**84.7**	78.9	45.3	73.4	65.8	80.3	74.0
HyperNet_VGG	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
Faster R-CNN [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
DEEP_ENS_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	39.1	33.1	65.2	62.7	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	29.4	27.9	56.6	57.0	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

YOLO 모델의 경우 57.9%의 mAP으로 정확도 낮은 편

But Fast R-CNN + YOLO의 경우에는 70.7%로 향상된 정확도를 볼 수 있음

Experiments

4.5. Generalizability: Person Detection in Artwork



(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.

학습 데이터셋과 다른 분포를 가진 Picasso Dateset과 People-Art Dataset 이용하여 테스트 진행

다른 모델들은 VOC2007에서의 정확도보다 현저히 떨어지지만, YOLO는 유사한 정확도를 가짐

-> YOLO는 객체의 일반적인 특징(feature)을 학습하기 때문

Limitations of YOLO

1. 작은 물체에 대한 검출 부정확 가능성 : object가 작을수록 bbox 간 IoU 값의 차이가 작아 object가 큰 박스보다 predictor가 작은 차이로 결정됨

2. 하나의 cell이 하나의 물체만 검출하므로 2개 이상의 객체가 겹쳐져 있다면 검출하기 어려움

3. 새로운 비율의 물체에 대해서는 검출이 어려움 : data에서 bounding box를 예측하는 것을 학습하기 때문

4. bbox의 크기와 상관없이 bbox의 loss에 동일한 가중치를 둠 : 크기가 작은 bbox의 경우 위치가 조금이라도 달라진다면 IoU 변화가 크므로 성능에 큰 영향을 줄 수 있음

감사합니다