이화여자대학교

You Only Look Once:

Unified, Real-Time Object Detection

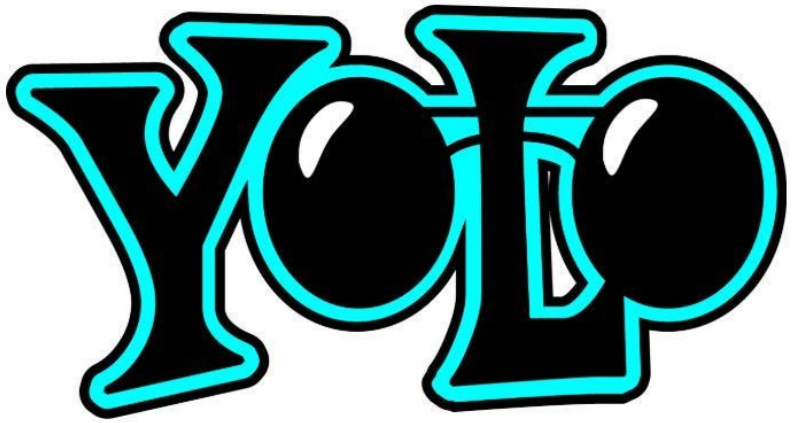
EURON 9th Research 추윤서

EWHA,
THE FUTURE
WE CREATE

목차

1. Introduction
2. Unified Detection
3. Comparison to Other Detection Systems
4. Experiments
5. Real-Time Detection in the Wild
6. Conclusion

Discussion



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

You Only Look Once

객체 탐지

- 기존 : 슬라이딩 윈도우 방식 + 복잡한 여러 단계 프로세스
- YOLO : 이미지를 한 번만 보기

Unified

객체 탐지 파이프라인의 여러 개별 구성 요소 → Single Neural Network

- DPM, R-CNN : 각 모듈을 따로 학습, 최적화
- YOLO : 모든 단계를 하나의 end-to-end 학습 가능한 신경망

Real-Time Object Detection

- 속도 강조

1. Introduction

1. Introduction

Object Detection

사람 : 이미지를 한 번 쳐다보고 파악

객체 탐지 : 다양한 분야에서 컴퓨터의 활용 가능성을 확장

DPM, R-CNN의 문제점

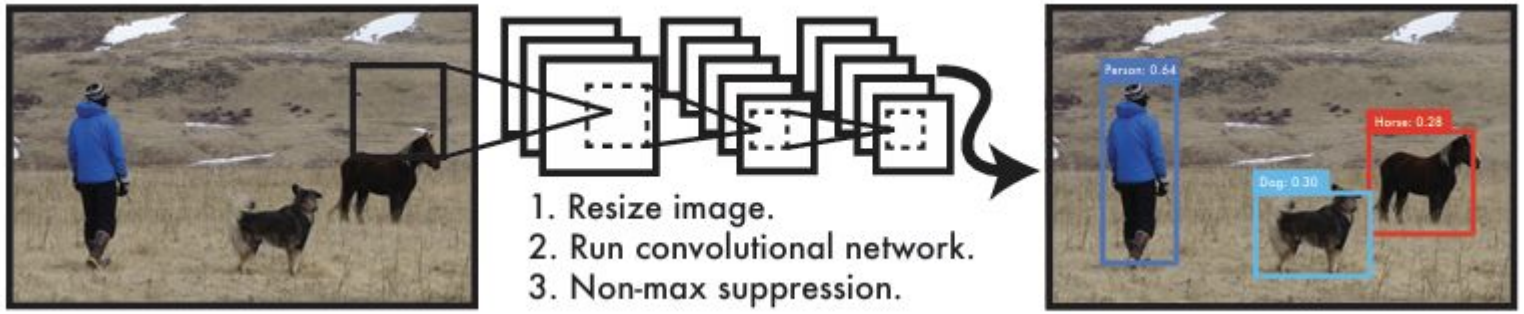
classifier 재활용 → 느린 속도, 최적화 어려움

- DPM : 슬라이딩 윈도우 접근법
- R-CNN : 후보 영역 생성한 후, classifier 실행

1. Introduction

YOLO Detection System

이미지를 한 번만 보고 어떤 객체가 어디에 있는지 예측



- Single Regression Problem

  이미지 픽셀 → multiple Bounding Boxes + Class Probabilites

- Single CNN
- End-to-End

1. Introduction

YOLO의 장점

- **Extremely Fast 빠른 속도**
  - 실시간 탐지에서 두 배 이상의 MAP
- **Global Reasoning 전역적 추론**
  - 전체 이미지 → 객체의 외관 + 클래스의 맥락적 정보 인코딩
  - 배경 오류 감소
- **Generalizable Representation 일반화 가능한 표현 학습**
  - 새로운 도메인이나 예상치 못한 입력에서 오류 발생 가능성이 적음

YOLO의 한계

- 낮은 정확도
- 작은 개체를 정확하게 지역화하는 데 어려움

2. Unified Detection

2. Unified Detection

YOLO

- 모든 탐지 과정을 하나의 CNN에 통합

Detection Process

- input : 이미지 SxS grid
  - 어떤 객체의 center를 가진 cell이 해당 객체에 대한 예측을 담당함
- 각 grid cell이 B개의 Bounding Box를 예측
  - Bounding Box : (x,y), w,h, confidence
  - confidence score = $\Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}}$
- 각 grid cell이 C개의 조건부 클래스 확률 예측
  - 조건부 클래스 확률 = $\Pr(\text{Class}_i|\text{Object})$
- 클래스별 신뢰도 점수 $\Pr(\text{Class}_i|\text{Object}) * \Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}} = \Pr(\text{Class}_i) * \text{IOU}_{\text{pred}}^{\text{truth}}$

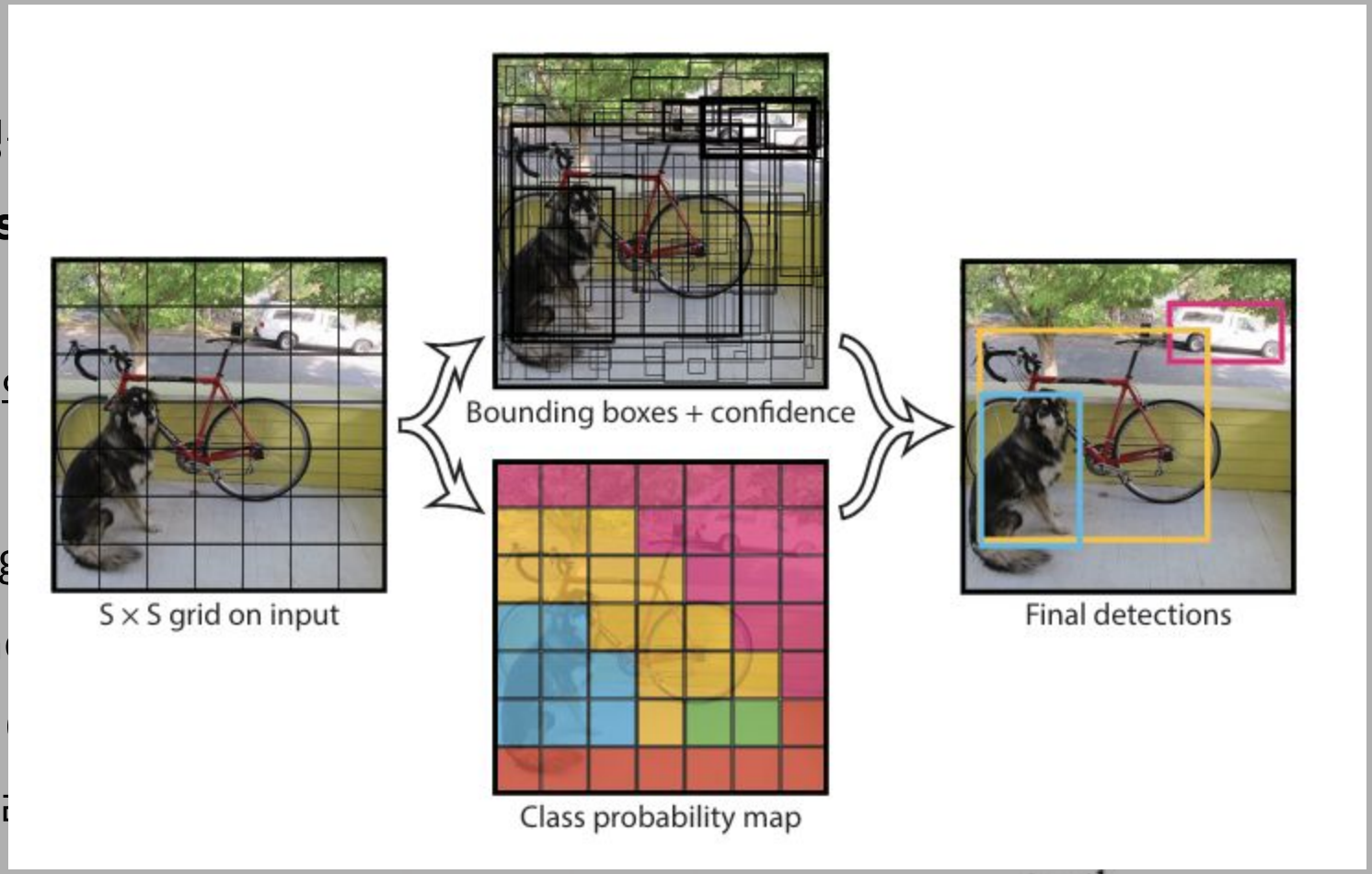
II. Unified Detection

YOLO

- 모든 탐지 과정

Detection Proces

- input : 이미지
  - 어떤 객체
- 각 grid cell이
  - Bounding
  - confiden
- 각 grid cell이
  - 조건부 클
- 클래스별 신뢰도 점수



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

2.1. Network Design

YOLO

CNN based Archiecture

- input : 이미지 전체
  - Convolutional Layers : 이미지에서 특징 추출
  - Fully Connected Layers : 객체의 위치 (Bounding Box)와 클래스 확률 예측
- GoogLeNet에서 영감 받아서 설계
  - 1x1 convolution + 3x3 convolution
  - S=7, B=2, C=20 → 최종 출력 : 7x7x30 tensor

Fast YOLO

속도 최적화 위한 경량 모델

- Convolution layers 9개 + 필터 수 줄이기

2.2. Training

Pretraining

일반적인 시각적 특징 학습시키기

- Dataset : ImageNet
- 구조 : 20 ConV + average-pooling + FC layer
- 성능 : Top-5 Accuracy 88%

Fine-tuning for Detection

- pretrained model + 4 ConV + FC 2 layer
- 입력 해상도 : 224x224 → 448x448

2.2. Training

예측 구조

최종 계층은 클래스 확률과 경계 상자 좌표를 모두 예측

- w, h, x, y 모두 0~1 사이 값으로 정규화

활성화 함수

최종 layer : linear function, 나머지 layer : Leaky ReLU

손실 함수

sum-squared error 최적

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

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

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

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

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

2.2. Training

hyperparameter

- epoch : 135
- batch size : 64, momentum : 0.9, decay : 0.0005
- learning rate

과적합 방지

- Dropout : 50%
- Data Augmentation

2.3. Inference

단일 네트워크 평가

- test 시에도 CNN을 한 번만 실행

  → 속도가 매우 빠름 : 실시간 성능을 제공

예측 결과

PASCAL VOC 데이터셋

- S = 7, B = 2 → S x S x B = 98

공간적 다양성 및 NMS

- grid 기반 설계, 객체가 속해있는 cell이 명확함
- Non-Maximal Suppression
  - 중복 예측 제거 위해서
  - YOLO에 필수적이지는 않지만, mAP가 2-3% 증가함

2.4. Limitations of YOLO

Spatial Constraints

- 각 grid cell은 B개의 bounding box와 1개의 클래스만 예측 담당
  → 하나의 grid cell 안에 여러 객체가 존재하거나, 객체들이 서로 가깝게 붙어 있는 경우 탐지 어려움

Struggles to Generalize

- 훈련 과정에서 학습되지 않은 처음 보는 비율이나 새로운 위치 배치에 잘 반응 x
  → 제한적인 일반화 성능

Coarse Features 사용

- 여러 downsampling layers → 위치 정보 손실 위험
  → 작은 객체의 위치 파악이 어려움

Loss Function Imbalance

- sum-sqared error 이용하여 최적화 → mAP 최대화 목표와 불일치
- localizatioin error 와 classificatioin error 에 동일한 가중치 부여
  → 작은 박스에 대한 error도 큰 박스와 동일하게 처리됨

3. Comparison to Other Detection Systems

3. Comparison to Other Detection Systems

기존 Detection Systems 방식

- Feature 추출 → Classifier/Localizers → Sliding Window/Region Proposal

DPM (Deformable Parts Models)

- Sliding Window + 파이프라인 : 각 모듈을 개별 학습하여 느림
- **YOLO** : Single CNN으로 통합

R-CNN

- Region Proposal로 잠재적 bounding box 생성 → CNN → SVM으로 분류
  → 매우 느림
- **YOLO** : End-to-End, 98개 박스만 예측

3. Comparison to Other Detection Systems

Fast R-CNN, Faster R-CNN

- 연산 공유, 신경망 사용하여 영역을 제안 → 속도와 정확도 향상 / 실시간 성능x
- **YOLO** : 처음부터 높은 속도를 위해 설계 → 실시간 성능 달성

Deep MultiBox

- CNN 훈련 → 관심 영역 예측
- **YOLO** : 완전한 탐지 시스템

OverFeat

- CNN → localization → detection 적용
- **YOLO** : 전역적 추론

MultiGrasp

- bounding box 예측 위한 grid cell 방식은 YOLO와 유사
- **YOLO** : 다양한 클래스의 여러 객체에 대한 bounding box 예측

4. Experiments

4.1. Comparison to Other Real-Time Systems

YOLO 모델의 실시간 성능을 속도(FPS)와 정확도(mAP) 측면에서 비교

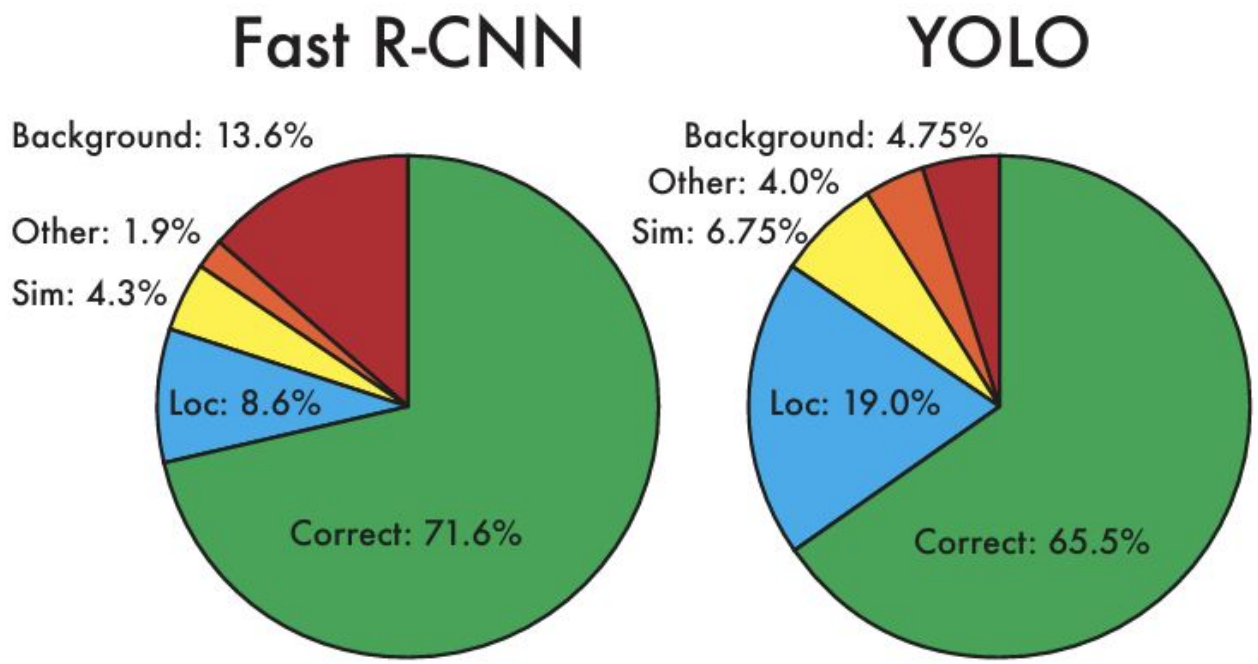
모델명	속도 (FPS)	mAP (%)
Fast YOLO	155	52.7
YOLO (기본 모델)	45	63.4
YOLO (VGG-16 사용)	21	66.4
Fastest DPM [38]	15	-
R-CNN Minus R [20]	6	-
Fast R-CNN [14]	0.5	70.0
Faster R-CNN (VGG-16)	7	73.2
Zeiler-Fergus 모델	18	62.1

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

→ YOLO는 속도와 정확도 면에서 혁신적인 균형을 제공하였다

4.2. VOC 2007 Error Analysis

YOLO 모델과 Fast R-CNN 간의 성능 차이 파악을 위해 발생 오류를 분석



- **YOLO의 약점 : 낮은 위치 정확도**
- **YOLO의 강점 : 낮은 배경 오류**

4.3. Combining Fast R-CNN and YOLO

탐지 성능 향상을 위해 YOLO와 Fast R-CNN을 결합

- 배경 오류가 적은 YOLO의 강점을 살림
- 지역화 오류가 많은 YOLO의 약점을 보완함

→ YOLO를 Fast R-CNN의 배경 오류 탐지를 제거하는 필터, 신뢰도를 높이는 보조 수단으로 사용

	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**

→ YOLO + Fast R-CNN 은 좋은 성능을 낸다.

4.4. VOC 2012 Results

YOLO의 VOC 2012 성능

- 작은 객체 탐지에서 약점 : bottle, sheep, tv/monitor

  → 입력 이미지에 비해 거친 특성을 사용 + 각 grid cell이 예측할 수 있는 bounding box 수가 제한적
- 특정 객체 탐지에서 강점 : cat, train

  → 이미지 전체를 보고 context information을 활용

Fast R-CNN + YOLO

- 성능 향상을 보임

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

→ YOLO가 단독으로는 최고 정확도 달성은 못하지만,

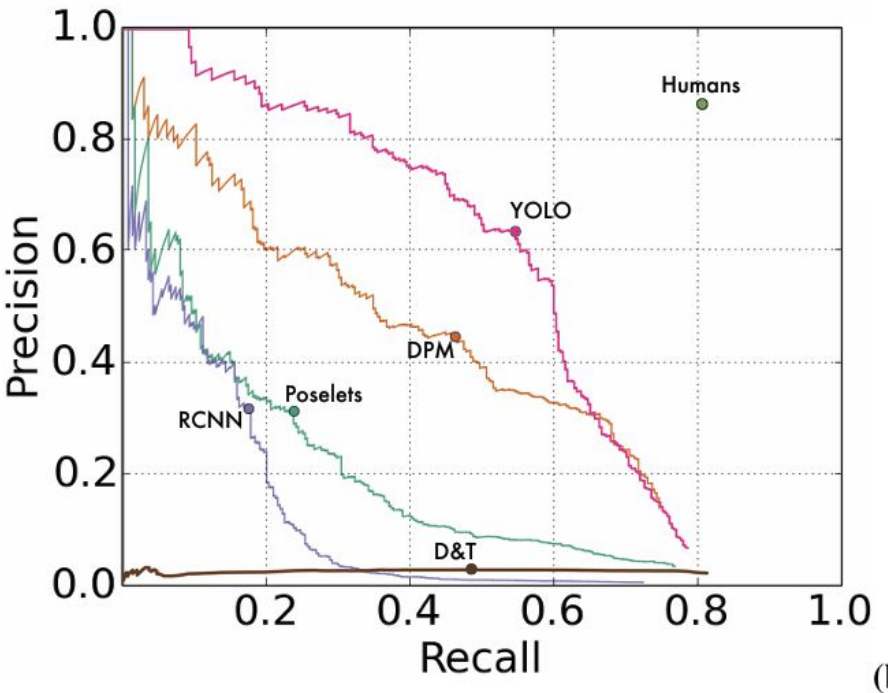
빠른 속도, 배경 오류 감소, 특정 객체에 대한 강점을 이용해 다른 고성능 모델의 약점을 보완할 수 있다.

4.5. Generalizability

YOLO의 우수한 성능

- 예술 작품에 적용했을 때 다른 방법보다 AP 감소가 적음

  → 객체의 크기와 모양 등 추상적인 특징 면에서 유사성이 있어 YOLO가 좋은 경계 상자와 탐지 예측 가능

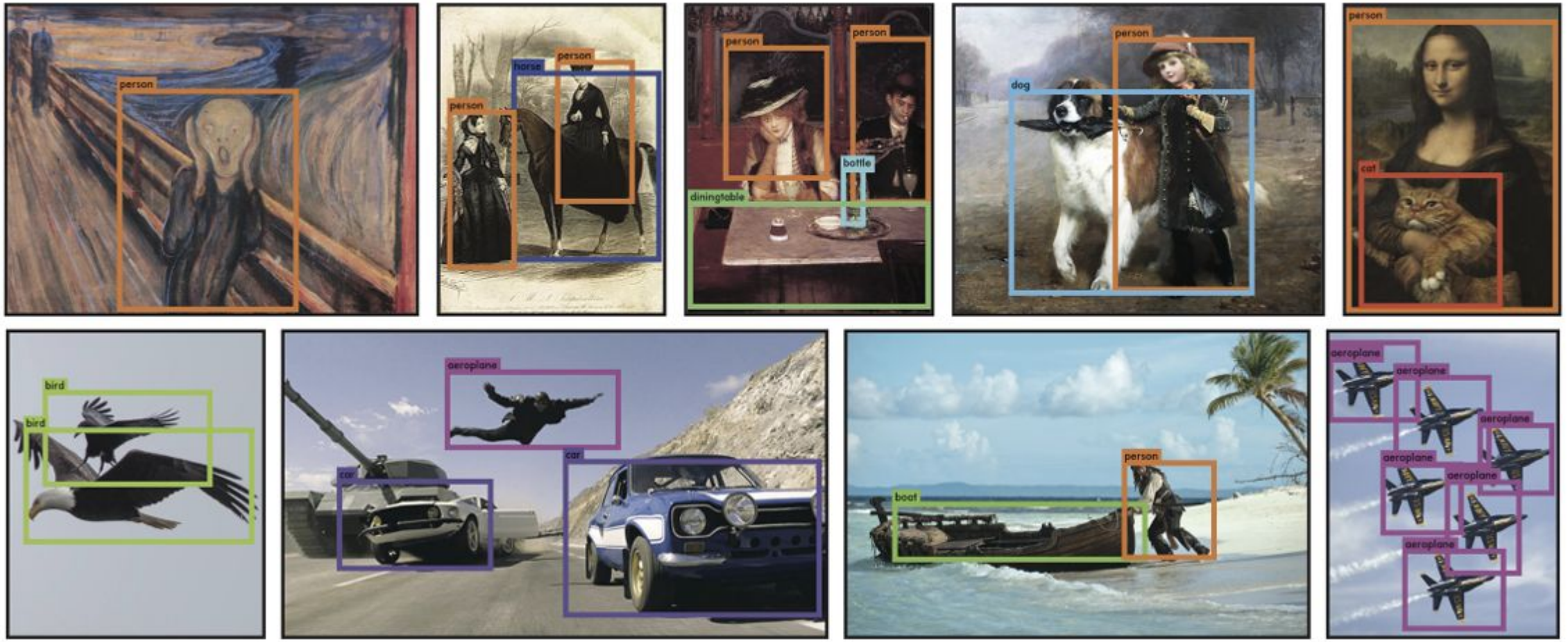


(a) Picasso Dataset precision-recall curves.

	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.

4.5. Generalizability



5. Real-Time Detection In The Wild

5. Real-Time Detection In The Wild

YOLO의 강점

- 빠르고 정확
- 다양한 컴퓨터 비전 응용에 적합

Webcam 연결 실험

- 실시간 성능 유지하는지 검증
- 객체가 움직이고 모습이 변하더라도 계속해서 객체를 탐지
- 추적 시스템처럼 기능 → 사용자와의 상호작용이 가능하여 몰입감 있는 경험을 제공

6. Conclusion

6. Conclusion

Contributions

- 객체 탐지를 위한 Unified 모델 제안
- 간단한 구조의 End-to-End 훈련 가능
- 탐지 성능에 직접적으로 관련된 loss function으로 훈련됨
- 새로운 도메인데 잘 일반화되어 다양한 응용 분야에 적합

Fast YOLO

- 가장 빠른 범용 객체 탐지기
- 실시간 객체 탐지 분야에서 SOTA 달성

Discussion

1. YOLO의 주요 오류 원인은 지역화 오류 (incorrect Localizations) 이다.
   grid cell 당 하나의 클래스만 예측할 수 있다는 공간적 제약 외에, 지역화 오류를 유발하는 다른 구조적 요인은 무엇이 있을까?

2. YOLO에서 NMS의 중요성이 낮은 이유는 무엇일까?

감사합니다.

EWHA,
THE FUTURE
WE CREATE