You Only Look Once (YOLOv1): Unified, Real-Time Object Detection

최성욱

목차

1. Abstract

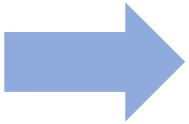
· Why do we need to use this proposed model?

1. Whole Detection Pipeline is a single Network.

2. Fast
    1. Real-Time에서 초당 45 frame 처리 가능
    2. Fast YOLO: 초당 155 frame 처리 가능

3. False Positive가 적다.

4. Generalized



✓ 빠르다.

✓ 간편하다.

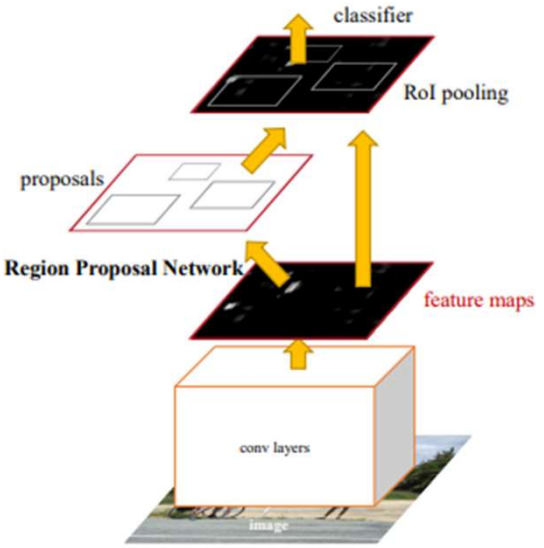
✓ 일반화가 잘된다.

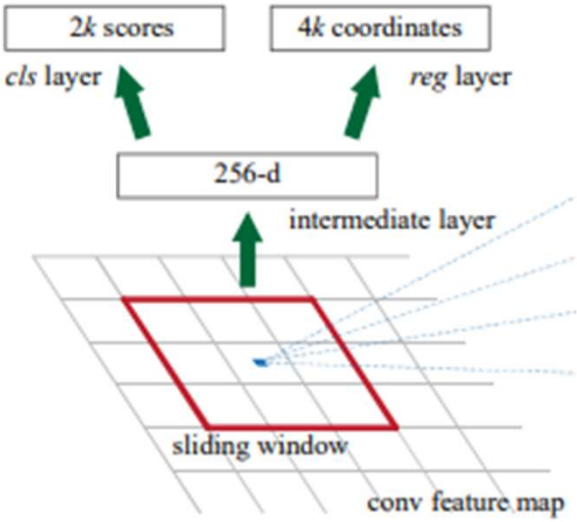
2. Introduction

· YOLO vs Traditional Detector

1. Traditional Detector (DPM, R-CNN)

   - RPN or Sliding Window를 활용하여 Potential Bbox
   - Potential Bbox를 바탕으로 분류 진행
   - Post-Processing (Bbox 미세 조정, NMS, Bbox rescore)





2. Introduction

· YOLO vs Traditional Detector

1. YOLO

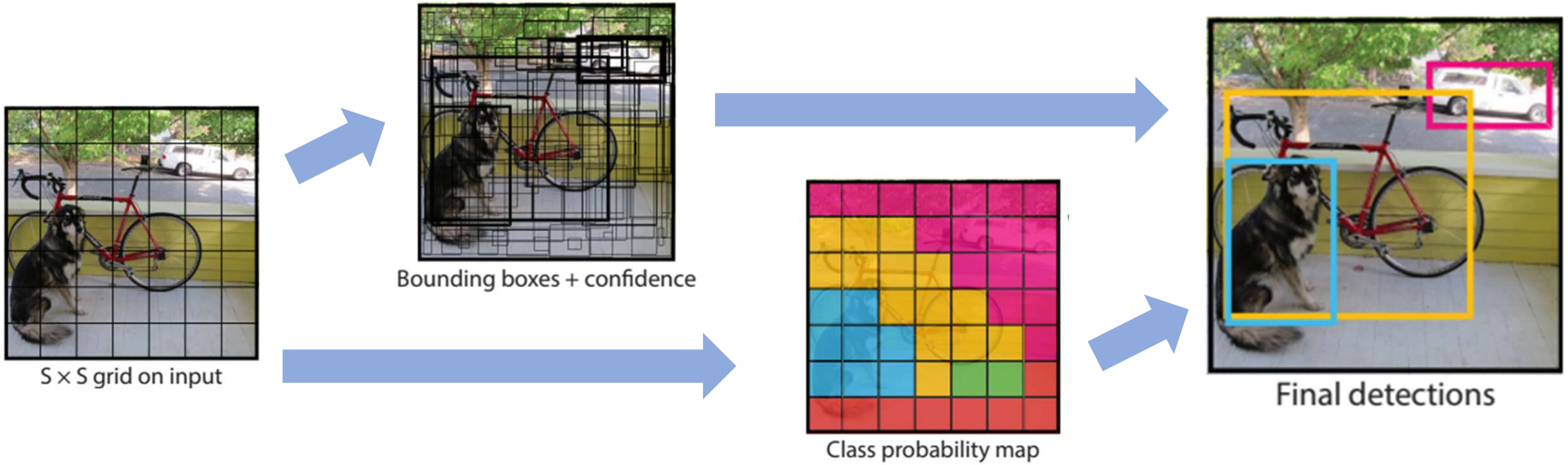
- Simple: Coordinate와 Class Classification을 동시에 Regression 문제로 해결
- Fast with Performance
  - Batch처리 없이 하나의 Titan X GPU에서 초당 45 frame을 처리
  - Real-Time system에서 가장 높은 mAP
- Reason Globally about the image
  - False Positive가 Fast R-CNN에 비해 2배 이상 적다.
  - Background Error가 낮다.
- Generalizable Representation
  - Natural Dataset으로부터 학습 -> artwork에서의 Test
  - DPM, R-CNN 보다 훨씬 성능이 좋다.

3. Technique

· Unified Detection

1. 전체 이미지에서 Bbox를 예측.

2. 각 Grid Cell에서 Class를 예측.

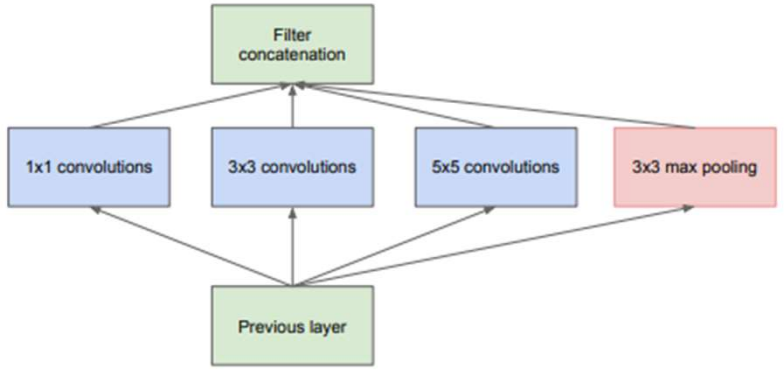


3. Technique

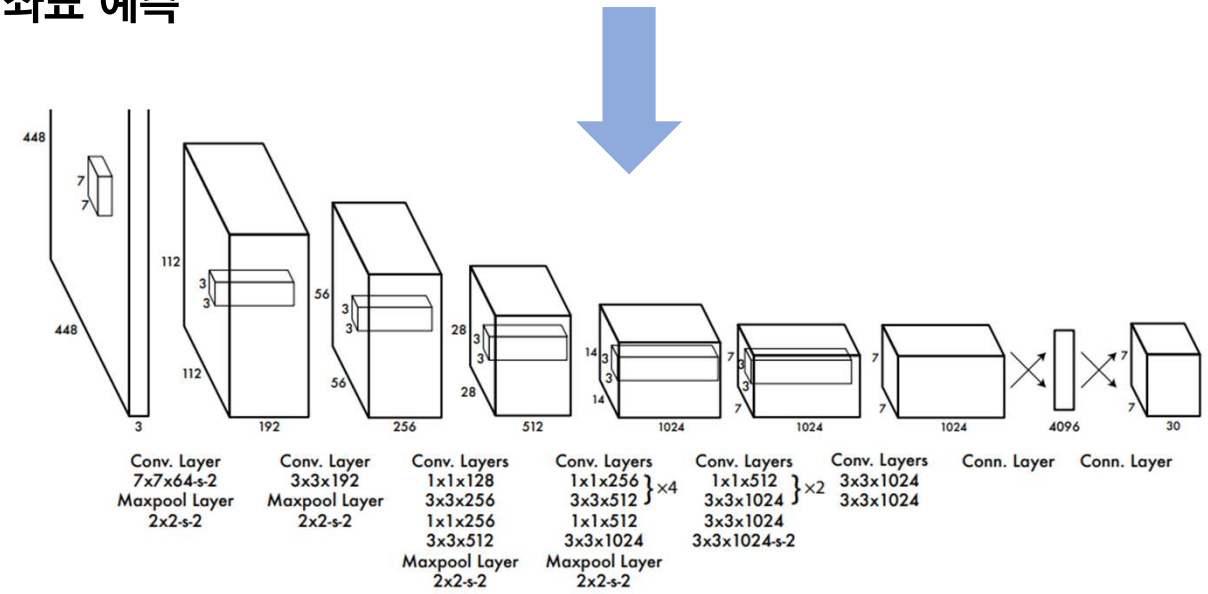
· Network Design

1. Convolutional Layers: 특성 추출
   - 24개의 Conv Layers
   - 1x1 Reduction Layer + 3x3 Conv Layer

2. Fully Connected Layers: class 확률과 좌표 예측



(a) Inception module, naïve version



3. Technique

· Training

1. Pre-Training
   - ImageNet Dataset을 활용하여, 20개의 Conv Layer에 대해서만 사전 학습을 진행

1. Additional Layer
   - 4개의 Conv Layer와 2개의 Fully Connected Layer 추가 –〉 성능의 개선 [4]

3. High Resolution (448 x 448)
   - 세밀한 Visual 정보를 요구 (Detection)

4. Normalize Bbox
   - Bounding Box의 width와 height를 image의 height, width에 대하여 정규화 하여 0과 1사이의 값으로 만든다.
   - x, y 역시 grid cell의 offset이 되도록 parameter화 하고, 0과 1사이의 값으로 만든다.

5. Activation Function
   - Leaky ReLU
   - Linear Activation Function at Final layer

3. Technique

· Loss Function

Notation
- 1_{ij}^{obj} : 객체가 탐지된 Bounding Box (즉, i번째 grid cell에서 j번째 객체를 포함한 Bounding Box)
- 1_{ij}^{noobj} : r객체가 탐지되지 않은 Bounding Box (즉, i번째 grid cell에서 j번째 객체를 포함하지 않은 Bounding Box)
- 1_{i}^{obj} : i번째 grid cell에서 객체를 포함하는지 여부.
- $\hat{C}_i$: i번째 grid cell에서의 Confidence 예측값. (Confidence = Pr(Object) $* IOU_{pred}^{truth}$)
- C_i : i번째 grid cell에서의 Confidence Ground Truth 값. (target값으로서 grid cell에 객체가 있으면 1, 없으면 0)
- x, y, w, h : 중심점의 좌표, width와 height. (C와 마찬가지로 예측값과 target을 표현)
- p(c) : class c일 확률

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

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

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

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

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

1. Bounding Box Loss
   - Object를 가지는 Bounding Box의 x, y에 대하여 SSE 연산
   - Object를 가지는 Bounding Box의 w, h에 대하여 SSE 연산
   - w, h의 경우, Large Box와 Small Box에서의 편차가 상대적으로 차이가 나므로 square root를 사용하여 그 편차를 줄여준다.
   - Penalty 부과

2. Confidence Loss
   - Object를 가지는 Bounding Box가 있는 grid cell에 대하여 Confidence 연산
   - Object를 가지지 않는 Bounding Box가 있는 grid cell에 대하여 Confidence 연산
   - Object를 가지지 않는 Bounding Box가 있는 gride cell에 대하여, Penalty 부과

3. Classification Loss
   - Object가 있는 grid cell에 대하여, class 확률 예측

4. Penalty
   - Confidence Error에서 Object가 없는 Cell이 훨씬 많기 때문에 Confidence Error (특히, No Object)에 큰 power가 쏠린다.
   - 이를 보정하기 위하여, Bounding Box Loss에 Penalty를 추가하여 증가시킨다.
   - 더불어, No Object에 대한 Confidence Error에 Penalty를 추가하여 감소시킨다.

3. Technique

· Hyper Parameter

1. Pascal VOC 2007/2012 에 대하여 train / valid dataset을 활용하여 135 epoch 진행
   (VOC 2012에 대하여 test할 경우 -> VOC 2007도 함께 학습, 반대도 마찬가지로 진행)

2. Batch size: 64

3. Optimizer: SGD (Momentum: 0.9, weight_decay: 0.0005)

4. Learning Rate Scheduling: StepLR
   (첫 epoch에서는 10^{-3} -> 10^{-2}로 높임 / 75epochs: 10^{-2}, 30epochs: 10^{-3}, 30epochs: 10^{-4}

5. Dropout & Extensive Data Augmentation
   - Dropout Ratio: 0.5
   - 원본 image size의 20% 까지 random scaling and translation
   - HSV color space에서 image의 exposure & saturation을 최대 1.5배까지 랜덤 조정

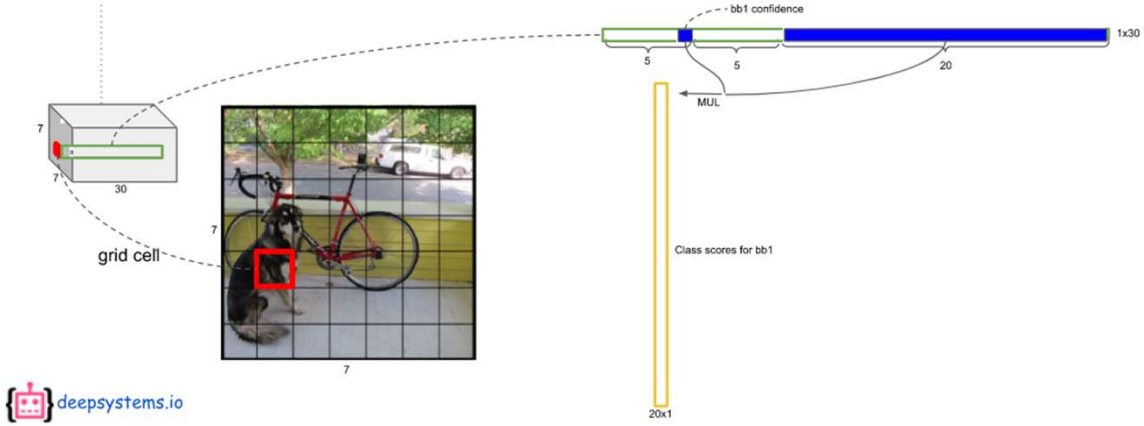
3. Technique

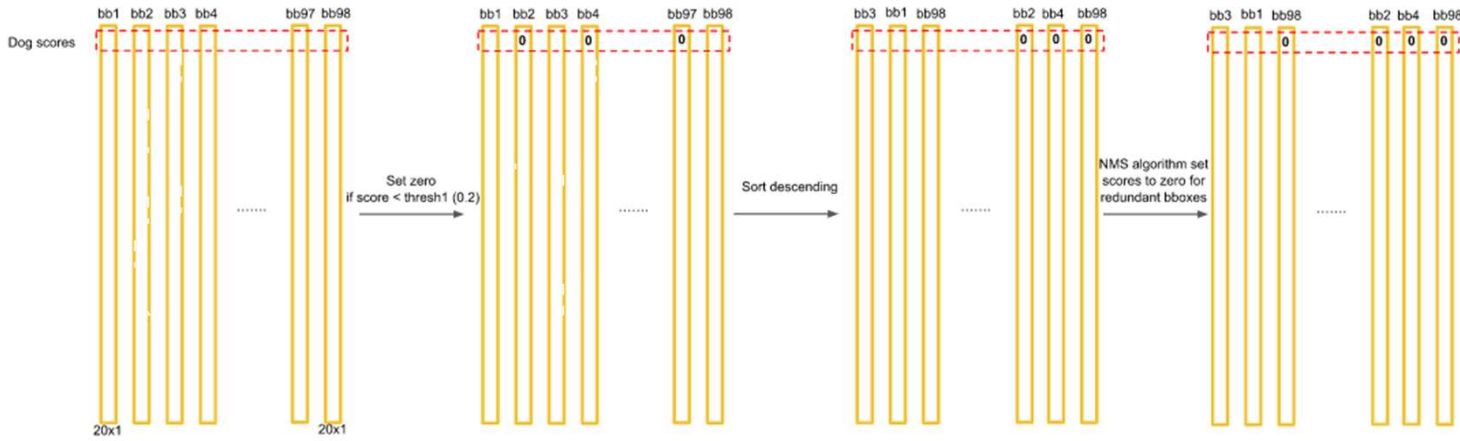
· Inference

1. 각 image당 98개의 Bounding Boxes 예측

2. 각 Bounding Box에서의 Class 확률 예측
   - 각 Bounding Box의 Confidence Score와 Class Score를 곱한다.

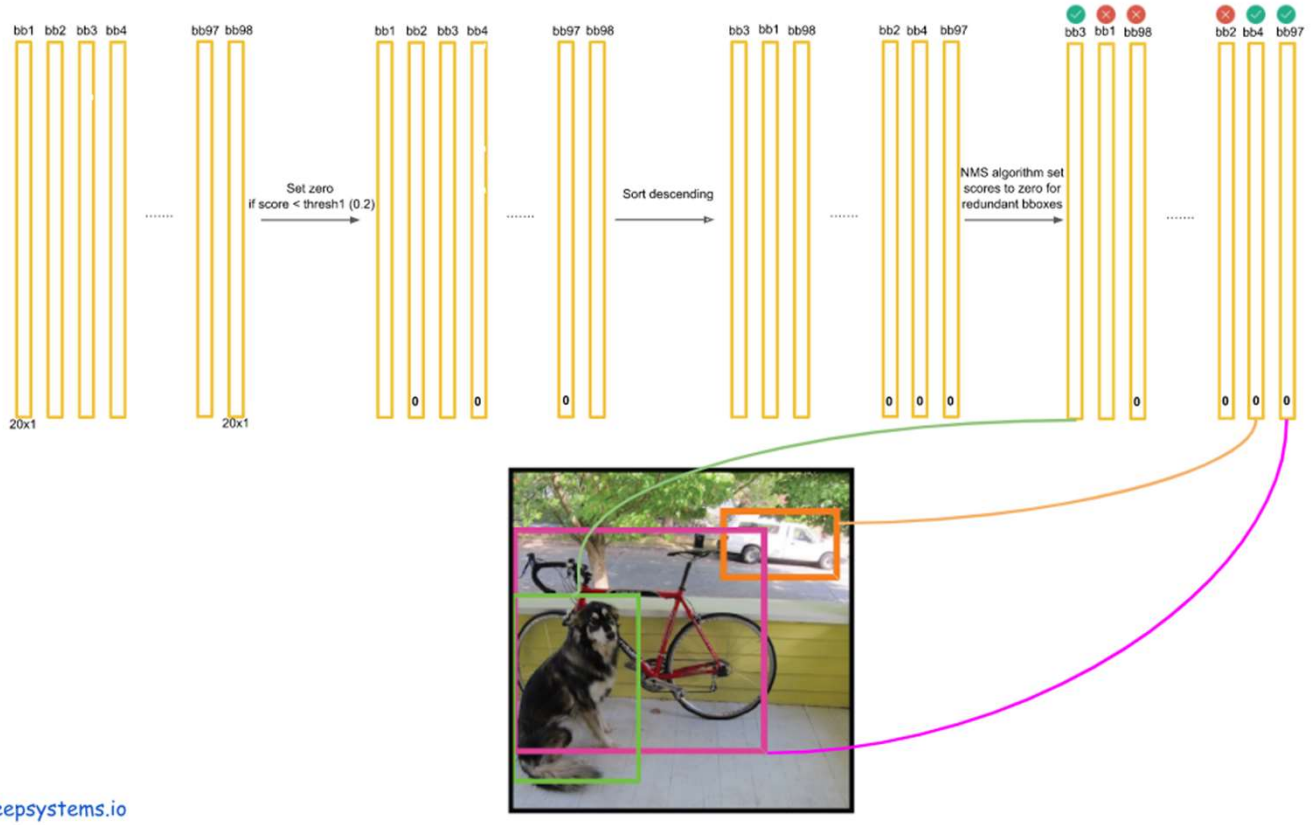
3. NMS (Non-maximal suppression)
   - 2~3%의 mAP 상승





3. Technique

- Inference



deepsystems.io

4. Limitations

1. Spatial Constraints
   - 하나의 Grid Cell에서 1개의 class 만을 예측한다.
   - 여러 작은 객체가 하나의 grid cell 내부에 있을 경우, detect 하기 어렵다.

2. Aspect ratios의 일반화의 어려움
   - Training Dataset에 의존하는 Bbox의 aspect ratios

3. Coarse Features
   - Upsampling Layer의 부재로 인한 Feature의 정보 부재

4. Bbox의 size에 따른 동일한 Loss 사용
   - Bbox의 크기가 크건, 작건 Loss가 동일
   - 큰 size의 Bbox에 비해, 작은 size의 Bbox에서 Loss에 더욱 민감해야한다.

5. Experiments

· Comparison

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

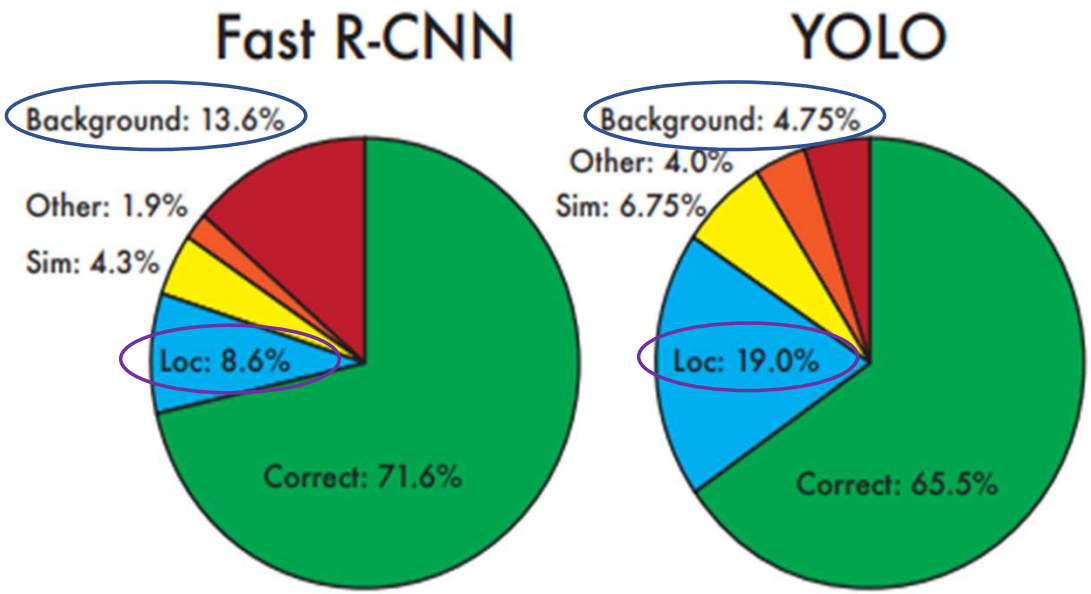
1. YOLO
   - 63.4%의 mAP
   - Maintaining Real-Time Performance
   - Fast YOLO의 경우 가장 빠른 Detector

2. Fast R-CNN
   - Selective Search (2 sec per image)
   - 0.5 fps

3. Faster R-CNN
   - Selective Search -> Region Proposal Network
   - YOLO에 비해 높은 성능
   - 같은 Backbone인 VGG-16을 사용했을 때, YOLO에 비해 3배 느리다.

5. Experiments

· Error Analysis



- Correct: correct class and IOU > .5
- Localization: correct class, .1 < IOU < .5
- Similar: class is similar, IOU > .1
- Other: class is wrong, IOU > .1
- Background: IOU < .1 for any object

각 Category의 Top N Prediction 진행

1. BackGround Error
   - IU 〈 0.1 for any object
   - IoU가 낮은데 객체를 담고있을 경우
2. Localization Error
   - Correct Class
   - 0.1 〈 IoU 〈 0.5
   - 제대로 예측했으나 IoU가 작다.
3. YOLO
   - 객체 탐지는 잘하지만, 좌표를 제대로 찾지 못한다.
   - 이미지 전체를 보기 때문에, BackGround Error의 비중이 낮다.

5. Experiments

· Ensemble

1. YOLO + Fast R-CNN
   - Fast R-CNN의 Background Error를 낮추기 위하여 YOLO와 Ensemble
   - R-CNN이 예측한 모든 Bounding Box에 대하여, YOLO가 유사한 Box를 예측했는지 확인한다.
   - 예측된 Box들의 Overlap을 바탕으로 최종 예측 결정

2. 의의
   - "Fast R-CNN + Fast R-CNN" 에 비하여 더 높은 성능의 개선
   - 서로 다른 종류의 모델 Ensemble을 통한 높은 성능향상
   - Output에 대한 Soft-Ensemble 이기 때문에, 속도에서의 이득은 없다.
   - But, YOLO의 실행 속도는 미미하므로, Fast R-CNN 측면에서는 큰 속도의 손실 없이 성능 향상 가능

	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**

5. Experiments

· Generalizability

1. Generalizable
   - Real World에서는 모든 사례들을 예측할 수 없고, Test Data가 Train Data와 많이 상이할 수 있다.
   - Artwork (Picasso Dataset, People-Art Dataset) 에 대하여, 일반화 Test를 진행한다. (Train은 VOC 2007로 진행)

2. R-CNN
   - VOC 에서의 높은 AP 성능
   - Artwork에서의 큰 성능 감소
   - Selective Search (혹은 RPN)을 통한, Region탐색 -〉 좋은 RPN 성능을 필요로 한다.

3. YOLO
   - 다른 모델들에 비해, 더 낮은 성능 감소
   - Image 전체를 보며, 객체들 사이의 관계, 위치를 모델링

4. DPM
   - 객체의 모양과 Layout에 대한 강한 Spatial Model
   - 즉, 일반화가 어느 정도 잘 되는 Model로 알려져 있다.
   - But, 기존의 낮은 AP로 인해 성능이 좋지 않다.

	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	

References

- [1] You Only Look Once: Unified, Real-Time Object Detection
  https://arxiv.org/pdf/1506.02640.pdf

- [2] Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks:
  https://arxiv.org/pdf/1506.01497.pdf

- [3] GoogLeNet: Going deeper with convolutions
  https://arxiv.org/pdf/1409.4842.pdf

- [4] Object detection networks on convolutional feature maps
  https://arxiv.org/pdf/1504.06066.pdf

- [5] deepsystem.io (YOLO)
  https://docs.google.com/presentation/d/1aeRvtKG21KHdD5lg6Hgyhx5rPq_ZOsGjG5rJ1HP7BbA/pub?start=false&loop=false&delayms=3000&slide=id.p