220908~ You Only Look Once : Unified, Real-Time Object Detection

https://arxiv.org/pdf/1506.02640.pdf

Abstract

- YOLO 란? : new approach to object detection
- YOLO 이전의 연구는 classifier를 detection 용도로 변환하여 object detection을 수행했으나,
- 해당 연구는 object detection을 bounding box와 class 확률에 대한 regression problem으로 정의했다
- YOLO는 :
    - 하나의 neural network가 bounding box와 class probability를 full image에서 한 번의 evaluation을 통해 예측해낸다
    - 이 unified architecture은 매우 빠르다 : 45frames/sec

- 다른 최신 알고리즘에 비해 localization error는 더 존재하지만 background의 false positive 수치가 낮다
- object의 일반적인 특징을 학습한다

1. Introduction

- 현재 detection system들은 object detection을 위해 classifier를 재정의하여 사용한다.
  - DPM(deformable parts model)의 경우 sliding window를 사용한다
    - sliding window란? classifier가 일정하게 나누어진 이미지 상에서 균일하게 이동하며 작동하는 방법
  - R-CNN 의 경우 potential bounding boxes의 생성을 위해 region proposal method를 사용한다 : bounding box를 생성한 뒤 해당 boxes들에 대해 classifier를 작동한다.
    - classification이후 bounding box를 refine하고, 중복된 검출을 지우고, 사진의 다른 object를 기반으로 box의 점수를 재 조정하는 등의 post-processing 작업이 필요하다.

  → 이러한 복잡한 pipeline은 느리며 최적화하기 어렵다. (각 component들이 따로 학습되어야 하므로)

- 반면 YOLO의 경우 다음과 같은 간단한 절차로 작동한다

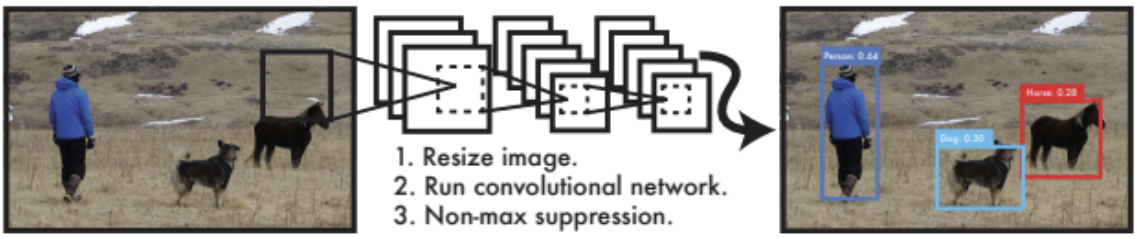


Figure 1: The YOLO Detection System. Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448 × 448, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.

- single convolution network가 다수의 bounding boxes와 class probabilities를 동시에 예측한다. 이러한 unified모델(즉, **YOLO)은 다음의 장점들을 가진다** :
  1. **YOLO는 매우 빠르다.** complex pipeline이 필요하지 않기 때문이다. streaming video에 대해 real-time으로 작업하는 것이 가능하다.
  2. 예측을 할 때 **이미지 전체를 통해 추론한다.**
     - sliding window나 region proposal-based techniques과 달리 YOLO는 training과 test 시 전체 이미지를 본다. → class에 대해서 contextual information을 학습하는 것이 가능하다.
     - R-CNN의 경우 larger context를 보지 못하기 때문에 배경의 작은 반점등을 처리하는데 약하다. YOLO는 R-CNN에 비해 background error를 반 이상 덜 만들어낸다.
  3. **YOLO는 object의 일반적인 특징에 대해 학습한다**
     - 자연 사진으로 학습을 하고 그림 사진을 이용하여 test를 할 때, YOLO는 다른 detection algorithm에 대해 훨씬 좋은 성능을 낸다. 따라서 new domain이나 예상치 못한 input에 대해 더 잘 대처한다.

- YOLO는 여전히 다른 state-of-the-art detection system에 비해 정확도가 약간 뒤처진다. 빠르게 object를 검출해야 할 때 특히 작은 object에 대해서는 판단을 어려워하는 경향이 있다.

2. Unified Detection

- YOLO는 input image를 S*S grid로 나눈다.

- 각 grid cell은 B개의 bounding boxes와 각 box에 대한 confidence score를 예측한다.
  - confidence score란?

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

    → bounding box가 object를 포함한다는 사실이 얼마나 믿을만 한지, 그리고 예측한 box가 얼마나 정확한지를 의미하는 score.

    → IOU : 실제 box와 예측 box의 교집합/실제 box와 예측 box의 합집합

  - confidence score와 IOU가 같은 것이 이상적이다

- 각각의 bounding box는 5개의 예측값을 가진다 : x, y, w, h, confidence score
  - (x, y) : grid cell 내에서 box의 중심에 대한 상대좌표(0~1)
  - (w, h) : 이미지 전체의 너비와 높이에 대한 bounding box의 너비와 높이(상대적인 값이므로 역시 0~1)

- 각각의 grid cell은 conditional class probability

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

  를 계산한다. object가 grid cell 내에 존재한다고 가정했을 때 그 class에 대한 확률을 뜻한다. 해당 확률 set은 grid cell당 한번만 구한다

- 테스트 단계에서는 conditional class probability와 confidence score를 곱한다

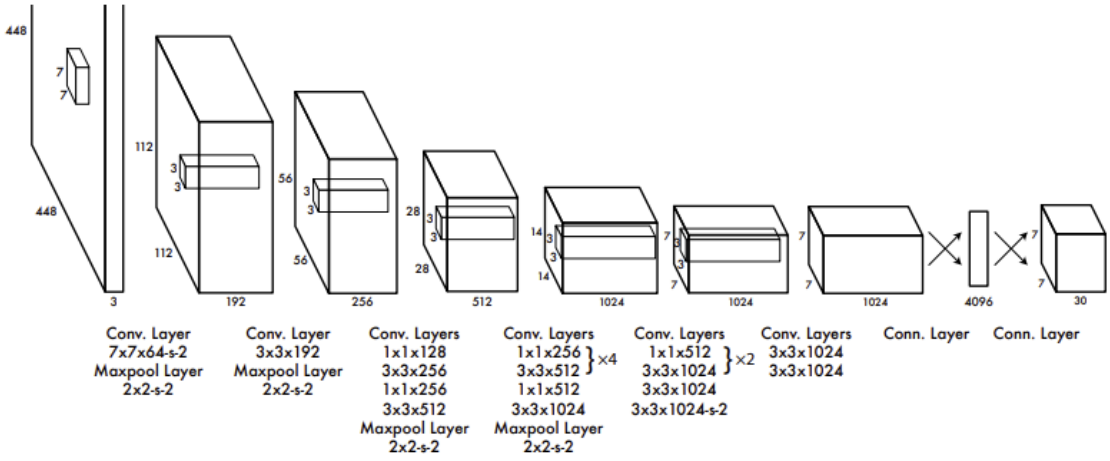
$$\Pr(\text{Class}_i|\text{Object}) * \Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}} = \Pr(\text{Class}_i) * \text{IOU}_{\text{pred}}^{\text{truth}}$$

  - 이는 박스 각각에 대해 class-specific confidence score를 구한 것이다.

- 해당 score는 1. box안에 그 class가 나타날 확률과 2. 예측한 box가 얼마나 object에 잘 들어맞는지에 대한 확률을 나타내고 있다.

2.1. Network Design

- YOLO의 모델은 CNN으로 구성이 되어있다
  - PASCAL VOC detection dataset으로 모델을 평가함
  - initial convolutional layers : image로부터 feature를 추출한다
  - fully connected layers : 확률과 좌표를 예측한다
- 모델의 architecture는 GoogLeNet model에서 영감을 받았다.



  - 24개의 convolution layer와 2개의 fc layer로 이루어져있다.
  - googLeNet의 inception module 대신 1*1 reduction layer와 3*3 convolution layer를 사용했다.
- final output은 7*7*30 크기의 tensor이다

2.2. Training

- convolutional layer의 pretrain을 위해 imageNet 1000-class competition dataset을 사용했다
  - 처음 20 layer, average-pooling layer와 fully connected layer를 사용
  - 7일동안 학습을 시켰으며 약 88%의 정확도를 얻었다

- 이후 detection을 위해 model을 변환하였다
  - 4개의 convolution layer와 2개의 fc layer를 weoght를 임의로 초기화하여 추가하였다
- 최종 layer는 class probability와 bounding box coordinate를 모두 predict
  - bounding box의 width와 height를 이미지의 크기에 대해 normalizing
- 최종 layer에는 linear activation function을 사용하였고, 다른 layer들에는 leakyRelu를 사용하였다
- loss function으로는 sum-squared error를 사용하였다. 최적화하기 쉽기 때문이다. 하지만 이는 평균 정확도를 높여야하는 목표와 정확히 일치하지는 않는다.
  - localization error와 classification error의 가중치를 동일하게 둔다(이상적인 방법은 아님)
- 또한, 많은 cell들은 object를 포함하지 않는데, 이는 confidence score를 0으로 만들어 object를 포함하는 cell들의 gradient를 가중시킨다. 이는 모델의 불균형을 초래한다.
  - 이를 해결하기 위해서, bounding box prediction의 loss를 증가시키고 object를 포함하지 않는 box에 대한 confidence prediction을 감소시켰다. 이는 다음의 두개의 파라미터로 구현하였다.

$$: \lambda_{\text{coord}} = 5 \text{ and } \lambda_{\text{noobj}} = .5.$$

- sum squared error는 bounding box의 크기에 상관없이 가중치가 동일하다. 하지만, 작은 bounding box의 경우 작은 위치 변화에 더욱 민감하다.
  - 이를 개선하기 위해 bounding box의 너비와 높이에 square root를 취해주었다. 너비와 높이가 커짐에 따라 증가율이 줄어들어 loss에 대한 가중치를 감소하는 효과가 있기 때문이다.
- YOLO는 하나의 cell당 여러개의 bounding box를 예측한다.
  - training time때 각 object에 대해 하나의 bounding box가 있는 것을 원하기 때문에 여러개의 bounding box중 하나를 선택해야한다.
  - 이를 위해 ground-truth bounding box와의 IOU가 가장 큰 bounding box를 택한다.
- 훈련 단계에서 사용하는 loss function은 다음과 같다

loss function:

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

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

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

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

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

  - 각 식의 의미는 다음과 같다 (출처 :[분석] YOLO (curt-park.github.io))

    (1) Object가 존재하는 그리드 셀 i의 bounding box predictor j에 대해, x와 y의 loss를 계산.

    (2) Object가 존재하는 그리드 셀 i의 bounding box predictor j에 대해, w와 h의 loss를 계산.

    (3) Object가 존재하는 그리드 셀 i의 bounding box predictor j에 대해, confidence score의 loss를 계산. (Ci = 1)

    (4) Object가 존재하지 않는 그리드 셀 i의 bounding box predictor j에 대해, confidence score의 loss를 계산. (Ci = 0)

    (5) Object가 존재하는 그리드 셀 i에 대해, conditional class probability의 loss를 계산.

- PASCAL VOC 2007, 2012 데이터를 사용하여 train 및 validation을 진행했으며, epoch는 135로 설정했다. batch size는 64, momentum 은 0.9, decay는 0.0005로 설정하였다.
- learning rate의 경우 처음에는 점점 증가시켰다가 다시 감소시켰다.

2.3. Inference

- PASCAL VOC 데이터 셋에 대해 한 이미지 당 98개의 bounding box를 예측하였으며, 각각의 class probability를 구해주었다.
- YOLO는 하나의 객체를 여러 그리드셀이 동시에 검출하는 경우가 있다. 이를 다중 검출 문제라고 한다. 이는 non-maximal suppression이라는 방법을 통해 개선하는 것이 가

능하다.

2.4. Limitations of YOLO

- 하나의 cell마다 두 개의 bounding box를 예측하고, 하나의 cell마다 하나의 객체만 검출할 수 있는데, 이는 spatial constraints를 야기한다.
    - 이는 하나의 cell에 두 개 이상의 객체가 있는 경우 이를 잘 검출하지 못하는 문제를 뜻한다.
- 또한 데이터로부터 bounding box 예측을 학습하기 때문에 새로운 aspect ratio에 경우 검출이 어렵다
- bounding box의 크기에 상관 없이 가중치가 동일하다. 크기가 작은 bounding box는 위치가 약간만 달라져도 성능에 큰 변화를 줄 수 있다.

4. Experiments

4.1. Comparison to Other Real-Time Systems

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [30]	2007	16.0	100
30Hz DPM [30]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [37]	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[27]	2007+2012	73.2	7
Faster R-CNN ZF [27]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

- 각종 객체 검출 모델 별 정확도와 속도를 보여주는 표이다.
- 정확도와 속도 모두가 높은 모델은 YOLO 계열 인 것을 확인할 수 있다

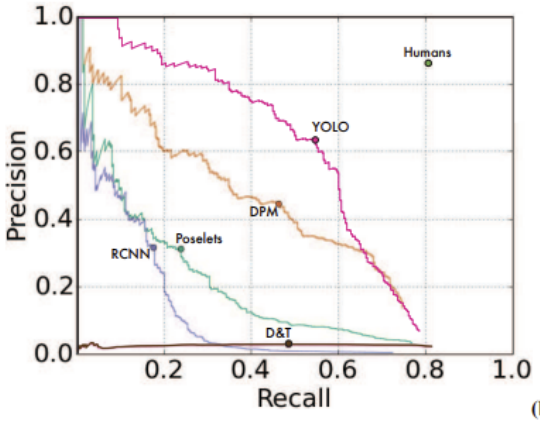
4.3. Combining Fast R-CNN and YOLO

- YOLO는 Fast R-CNN에 비해 background error가 훨씬 적다.
- PASCAL 2007 데이터 셋에 대해 Fast R-CNN과 YOLO를 결합했을 때 평균정확도는 75%이다.
- Fast R-CNN를 단독으로 사용하는 것과 앙상블 모델을 사용하는 것의 속도는 거의 유사하다.

	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**

4.5. Generalizability: Person Detection in Artwork

- 실제 이미지 데이터는 training과 test 데이터셋의 분포가 다를 수 있다.
- training 데이터셋과 다른 분포를 지닌 test 데이터셋을 활용하여 테스트를 해보았다. 피카소 데이터 셋과 일반 예술작품을 사용하였다.
- YOLO는 예술작품에 대해서 정확도가 크게 떨어지지 않았다.



(a) Picasso Dataset precision-recall curves.

	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	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.

Figure 5: Generalization results on Picasso and People-Art datasets.

6. Conclusion

- YOLO는 object detection을 위한 unified model이다
- 단순하며, full image를 직접 학습시킬 수 있다.
- classifier-based approach와 다르게 YOLO는 detection performance와 직접적으로 연관된 loss function을 최적화 시킨다.
- YOLO는 새로운 domain에서도 일반화 성능이 좋으며, 빠르고 robust한 object detection 모델이다.