Abstract

Object Detection은 이미지를 특정 class로 분류하는 Classification보다 더 큰 범주로, 시각자료에서 우리가 찾으려고 하는 object가 어느 위치에 있는지(localization), 어느 class인지(classification)에 대한 정보를 주는 것이다. YOLO가 쓰인 당시, Object Detection의 모델은 detection 속도와 정확도의 trade-off가 존재하는데, 정확도에 우위가 있는 것은 regional proposal과 classification이 순차적으로 이루어지는 파이프라인을 갖는 반면, 속도에 우위가 있는 것은 regional proposal과 classification이 동시에 이루어지는(end-to-end) model이다. 전자의 모델인 R-CNN와 비교하여 두 과정이 integration된(후자의 모델인) YOLO에 대해 자세히 알아본다.

1 Model before CNN

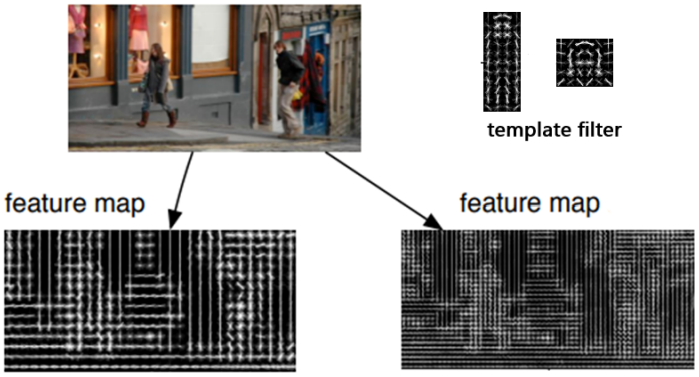

Figure 1: feature map template filter of DPM

CNN이 등장하기 이전 가장 성능이 좋았던 모델은 Deformable parts models(DPM)DPM [1]이다. DPM Sliding Windw 방식으로, 이미지에 일정 간격의 픽셀을 건너 뛰고 bounding box를 그린 후, 각 box영역을 여러 block(4×8 또는 8×8)으로 나누어 해당 block의 feature을 추출하게 된다. 이후 각 bounding box는 여러 template filter를 거치며 detection이 된다. block과 filter가 많이 존재하여 detection과정이 매우 복잡하며, 연산량이 매우 많았다. 이후 확인할 YOLO와의 결과 비교에서 정확도와 속도 모두 부족한 것을 확인할 수 있다.

2 Performance of detection models

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

Figure 2: mAP FPS of Real-Time Detectors for PASCAL VOC 2007

YOLO 이전의 R-CNN [2]과 같은 object detection은 selective search 알고리즘을 사용하여 이미지 내의 object를 localization하는 region proposal 방식을 사용하였다. 이는 network가 완전한 이미지가 아닌 이미지의 일부를 보는 것을 의미한다. 또한 이 방식은 single이 아닌 파이프라인을 독립적으로 튜닝해야 하기 때문에 속도가 매우 느렸다. 이와 달리 YOLO는 하나의 convolution network가 bounding box와 box에 대한 class probability를 동시에 예측한다. 이에 따라 object detection의 속도가 매우 빨라지는데, Figure2에서 다른 모델과 비교한 YOLO의 성능과 속도를 확인할 수 있다.
YOLO는 state-of-the-art에 못미치지만 FPS가 매우 높은 것을 확인할 수 있다. 즉, R-CNN모델들과 비교하였을 때, Object-detection에는 성능과 속도의 trade-off를 확인할 수 있다. 이제 YOLO의 design과 속도는 빠르지만 정확도가 조금 떨어지는 이유를 살펴본다.

3 Architecture of YOLO

3.1 Bounding Box

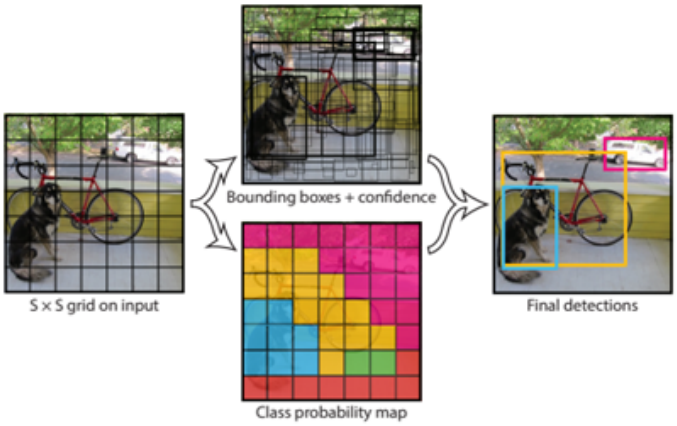

Figure 3: semantic segmentation task

region proposal 방식을 사용하지 않는 YOLO의 Unified Detection을 알아본다. 먼저 YOLO는 입력 이미지를 $S\times S$ grid로 나누며, 각 grid cell에서 B개의 bounding box를 예측한다. 그리드 셀 내부에 어떤 object의 center가 위치한다면 그 그리드 셀은 해당 object를 detecting해내야 한다. 그 prediction은 *confidence score*와 *conditional class probabilites*의 곱인 *class specific confidence score*으로 구할 수 있으며 그 값은 다음과 같이 정의된다. $classspecificconfidencescore = Pr(Class_i) * IOU^{truth}_{pred}$. $Pr(Class_i)$는 bounding box의 특정 class의 object가 나타날 확률이며 IOU^{truth}_{pred}는 실제 bounding box와 예측 bounding box의 겹치는 부분을 두 box를 합친 부분으로 나눈 것이다.

3.1.1 confidence score

$confidencescore = Pr(Object) * IOU^{truth}_{pred}$로, bounding box가 객체를 포함하는지와 얼마나 정확하게 예측하였는지를 나타낸다. $Pr(Object)$는 그리드 셀에 아무 object가 없다면 0, object가 있는 것이 확실하다면 1의 값을 갖는다. IOU에 대한 설명은 위와 동일하다. 이 confidence score를 사용하여 bounding box의 parameters를 representation할 수 있다. bounding box의 parameters는 (x,y), (w,h), confidence(=confidence score)로 구성되어 있다. 이때 x,y,w,h는 모두 정규화한다. (x,y) bounding box 중심의 그리드 셀 내 상대위치를 표현하며, (0.5, 0.5)일 때 그리드 셀의 중심에 위치해 있다. (w,h)는 전체 이미지 대비 bounding box의 width와 height를 의미한다.

3.1.2 conditional class probabilites

$conditional class probabilites = Pr(Class_i|Object)$로, 그리드 셀 내에 object가 존재할 때 그 object의 class가 무엇인지에 대한 Conditional prob 이다. 위 식에서 알 수 있듯이 YOLO의 grid cell은 Bounding box의 개수와 무관하게 하나의 class만 예측하는데 이는 YOLO의 limitation이라고 할 수 있다.

3.2 Detection Network

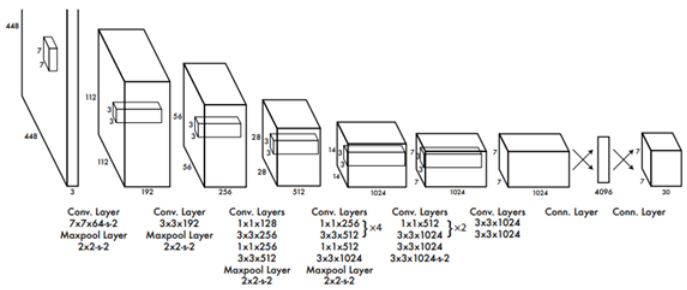

Figure 4: semantic segmentation task

YOLO는 feature extraction을 수행하는 24개의 convolution layer와 output 확률과 좌표를 예측하는 2개의 fully connected layer로 이루어져 있으며, 이는 GoogLeNet [3]을 따른다. GoogLeNet과 다른점은 (1,1), (3,3), (5,5)의 필터를 병렬로 사용하는 inception 구조 대신, 1×1의 reduction layer와 3×3의 convolution layer를 사용한다. 최종 output은 $S*S*(B*5+C), which C: of class$를 따른다
Fast YOLO는 좀 더 빠른 object detection을 위해 사용되며, 24개의 convolution layer를 9개로 축소시켜 사용한다.

3.3 Loss function

$$
\begin{aligned}
&\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2
\end{aligned}
$$

Figure 5: semantic segmentation task

3.3.1 Problem of loss function, SSE

YOLO는 sum-squard error(SSE)를 통해 loss를 계산한다. SSE를 최적화하는 방식은 여러 문제점이 존재하며 YOLO는 이에 따른 개선 방안을 제시하여 loss function을 재구성하였다. SSE는 localisation loss와 classification loss의 가중치를 동일하게 취급하여 학습을 진행한다. 이는 명확하게 좋은 방법이 아니며 YOLO의 최종 target인 mean Average Precision(mAP)를 높이는 것과 완벽하게 일치하지는 않게된다(이를 1번 문제라 명함). 또한 region proposal 방식을 따르지 않기 때문에 발생하는 문제로, 입력 이미지의 배경 영역이 object의 영역보다 넓어 모델의 불균형을 초래한다. 객체가 없는 cell의 confidence score는 0이 되는데, 이는 많은 grid cell의 confidence score가 0이 되는 것을 의미한다(이를 2번 문제라 명함). bounding box의 크기 차이에서도 문제가 발생한다. SSE에서는 bounding box의 크기에 따라 weight를 나누지 않고 동일하게 적용한다. 상대적으로 작은 bounding box일수록 작은 위치변화가 있으면 객체를 더 쉽게 벗어나게 된다(이를 3번 문제라 명함).

3.3.2 Remedy problems

1번과 2번 문제를 위해 객체가 존재하는 bounding box좌표에 대한 loss의 가중치를 5(λ_{coord})로 증가시키고, 객체가 존재하지 않는 것은 가중치를 0.5(λ_{noobj})로 감소시켰다. 이는 localization loss와 classification의 가중치를 다르게하는 것을 의미함과 동시에, 객체가 존재하는 grid cell의 confidence loss의 가중치를 증가시키는 효과를 얻게된다.
3번 문제의 해결을 위해 width아 height의 error계산에서는 square root를 취해주었다. 이로써 width와 height가 커지는 것에 따른 증가율을 감소시켜 loss에 대한 가중치를 감소시키는 효과가 있다.

4 Conclusion

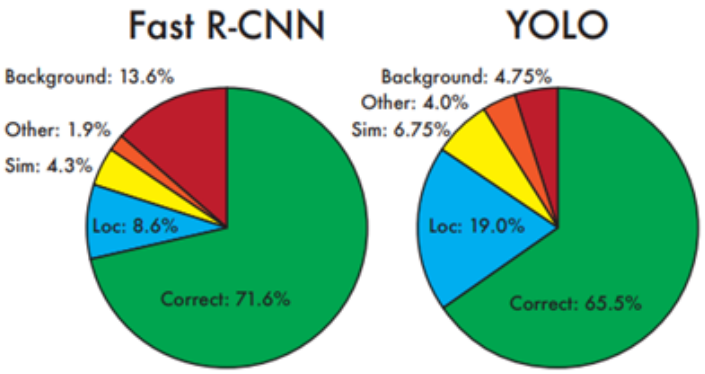

Figure 6: semantic segmentation task

YOLO는 fps에 강점이 있는 Object Detection model로 빠르게 물체를 detection하여 자율주행자동차 또는 드론을 이용한 건물내부 점검 등에 유용하게 쓰일 수 있다. YOLO 모델을 통해 많은 application에 적용이 가능할 것 같다.

4.0.1 Limitation

YOLO 모델의 한계는 3가지가 존재한다. YOLO는 grid cell에서 하나의 object만 detection하도록 설계되어 있어 grid cell 안에 여러 object가 있는 경우 detection하지 못하는 object가 존재하게 된다. 또한 bounding box를 예측하는 것을 학습하기 때문에 aspect ratio가 학습되며 새로운 종횡비를 test에서 잘 예측하지 못하게 된다. 마지막으로, 크기가 작은 bounding box는 IOU의 변화가 심하여 Incorrect localization문제가 발생한다.

4.0.2 Benefit

Figure2에서 확인할 수 있듯이, single neural network 구조이기 때문에 기존 모델에 비해 속도가 월등히 빠르고 정확도도 꽤 높은 수준이다. Figure6에서 정확도가 Fast R-CNN모델에 비해 조금 낮지만, background error가 작은 것을 확인할 수 있다. selective search를 통한 객체가 있는 부분만 학습하도록 제안하는 모델과 달리 전체 이미지를 학습하기 때문에 background를 찾아내는 것을 잘 수행한다고 볼 수 있다. 또한 전체이미지를 학습함으로써 training dataset과 다른 distribution을 가진 test dataset에 강인한 결과를 보여주었다.

[1] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009.

[2] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[3] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.