[논문리뷰] SSD

Introduction

당시에 SOTA는 Faster-rcnn이었습니다. 정확도도 높고 deep 한 network입니다. 하지만 계산량이 너무 많으며 좋은 하드웨어를 사용해도 실시간으로는 사용하기 어렵다. 제일 빨라봤자 7 fps 밖에 나오지 않는다. 따라서 속도를 올리려고 노력했지만 속도가 올라간 만큼 저확도가 감소하였다.

본 논문은 object proposal를 사용하지 않고 object detection을 할 수 있으며 VOC2007 test를 기준으로 Faster R-CNN의 경우 7 fps, 73.2%이며 YOLO의 경우 45 fps, 63.4%이다. 본 논문에서 소개하는 SSD의 경우 59 fps, 74.3%의 결과를 가져왔다고 합니다. 즉, 속도와 정확도 두 마리의 토끼를 동시에 잡았습니다. 우선 proposal생성과 resampling 단계를 제거하여 속도를 증가시켰습니다. 또한 정확도를 위해 다른 scale과 aspect ratio를 가지는 default box를 사용하였습니다. (Faster-RCNN의 Anchor와 유사) 그 후에 다른 크기들의 feature map을 prediction에 사용하였습니다.

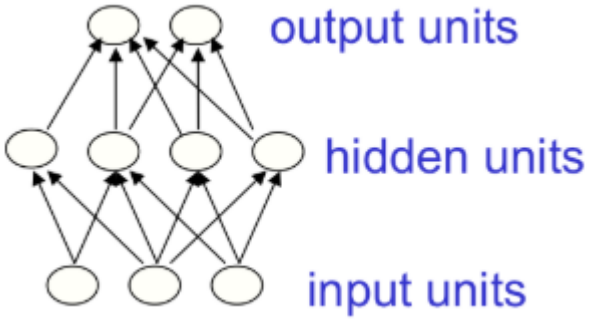
- YOLO 보다 빠르고 Faster-RCNN 보다 정확한 SSD를 소개합니다.
- SSD의 핵심은 작은 conv filter들을 사용한 default box들을 여러 feature map에 적용시켜 score와 box 좌표를 예측합니다.
- 정확도를 높이기 위해서 여러 크기의 다른 feature map들로부터 여러 크기의 predict를 수행하고 비율 또한 다르게 적용했습니다.
- end-to-end 학습을 할 수 있게 구축했으며 저해상도 이미지에서도 높은 정확도를 가집니다.
- 여러 대회(PASCAL VOC, COCO, ILSVRC)에서 실험을 진행한 것을 소개합니다.

The Single Shot Detector(SSD)

2.1은 SSD의 model을 설명하고 2.2는 학습 방법론에 대해 설명하겠습니다.

2.1 Model

SSD는 NMS를 거쳐 최종적으로 나오는 bbox와 score를 포함한 box들을 생각하는 feed-forward convolutional network(FFCNN)를 기반합니다. 밑의 그림은 단순한 FFCNN입니다. (원래 더 거대하다.)



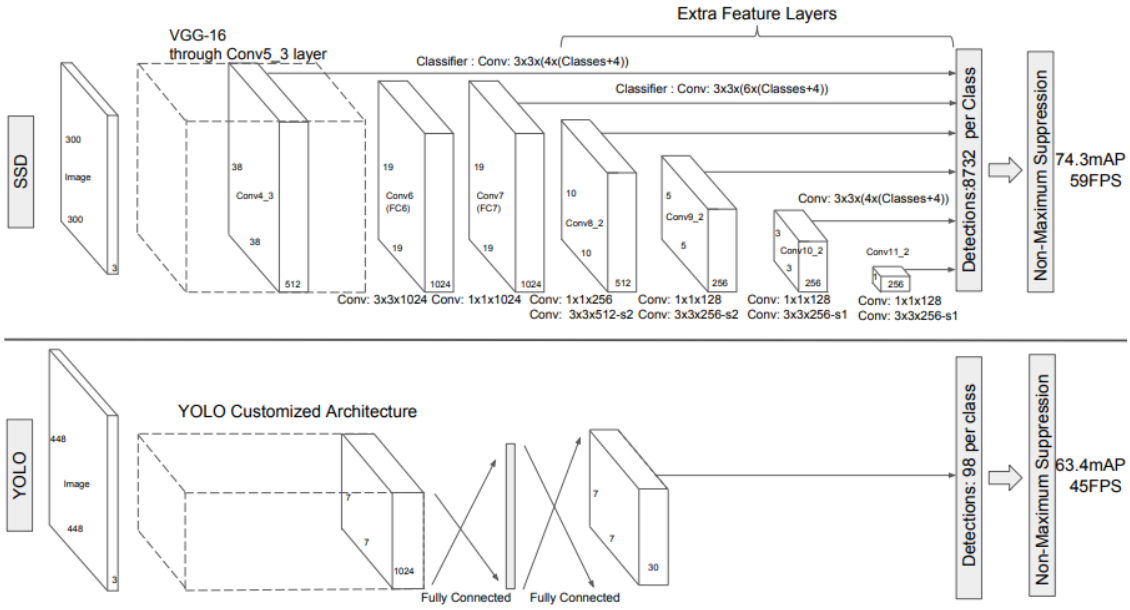
앞의 네트워크는 base network라고 불리는데 조금 수정한 pre-trained conv network이다. 그리고 detection을 위해 몇 가지 구조를 추가했는데 밑에서 설명하겠습니다.

Multi-scale feature maps for detection

base network에서 끝이 생략된 conv feature layer를 추가했습니다. 이 layer들은 점진적으로 크기가 줄어들고 다양한 크기에서 prediction 하도록 했습니다. detection을 위한 convolutional model은 Overfeat 그리고 YOLO 같이 단일 크기의 feature map을 만드는 다른 feature layer입니다.

Convolutional predictors for detection

각각 더해진 feature layer은 prediction을 위해 conv filter가 사용된 세트를 만듭니다. 즉, 밑의 사진에서 SSD 네트워크 구조의 윗단을 가리킵니다.



m x n x p의 feature layer을 얻기 위해 각 conv layer로 부터 3 x 3 x p의 conv layer을 추가로 통과시킵니다. 즉, m x n 의 위치에 3 x 3 x p 의 kernel을 적용시킵니다. 이것을

detection에 사용합니다. bbox는 default box의 위치로 알 수 있습니다.

Default boxes and spect ratios

여러 개의 feature maps을 얻기 위해 각 feature map 당 default b box들을 적용했습니다. 위의 그림에서 각 con layer의 최상단에서 detections로 넘어갈 때 default box들에 적용했습니다. 각 feature map에 맞게 default box들 또한 바뀌어 적용합니다.(feature map cell이라고 불립니다.)

각 feature map cell은 classfier score 뿐만 아니라 box의 좌표 또한 예측할 수 있습니다. 즉, feature map cell에서 k개의 box를 가진 다고 하면, c개의 classifier score와 4개의 box 좌표를 계산해야 합니다. 즉 `(c+4) k의` 필터가 필요합니다. 여기서 만약 feature map이 m x n을 가진 다고 하면, `(c+4)kmn`의 파라미터를 가집니다.

default box의 예시는 첫 번째 그림에서 볼 수 있습니다. 이 default box는 faster RCNN의 anchor와 비슷합니다. 하지만, 각 default box를 여러 개의 feature map에 적용시켰습니다. 각각의 feature map 마다 default box를 가지는 것은 물체의 크기와 비율이 달라져도 효과적으로 대응할 수 있습니다.

2.2 Training

SSD와 여타 다른 detector과 크게 다른 점은 gt 정보가 고정된 detector 출력들에 특정한 출력으로 할당해야 하는 것이다. 이런 것은 VOLO, Faster RCNN, MultiBox에도 똑같이 필요하다. 할당되는 게 한번 정해지면, 손실 함수와 역전파는 end-to-end로 적용된다. 학습에는 default box들의 크기와 비율을 선택하는 것뿐만 아니라 negative mining과 data augmentation 방법들도 선택해야 한다.

Matching strategy

학습을 하는 동안, default box는 gt box와 대응하여 network가 학습된다. 따라서, default box들의 위치, 크기 그리고 비율은 다양하게 선택되어야 한다. MultiBox의 최고 jaccard overlap과 함께 gt box와 default box를 매칭 시켰다. MultiBox와는 다르게 threshold인 0.5보다 크게 잡았다. 이렇게 하면 좀 더 정확하게 겹쳐야 default box가 선택이 된다. 즉, 학습을 좀 더 단순화시킨다.

Training objective

SSD 학습은 MultiBox에서 파생되었지만, 더 많은 object 목록을 가진다.

$$x_{ij}^{p} = \{1, 0\}$$

가 있다고 하자. 여기서 i는 default box의 index, j는 gt box의 index, 그리고 p는 object의 index이다. 위의 _Matching Strategy_을 적용시키면

$$\sum_i x_{ij}^p \geq 1.$$

를 예측할 수 있다.

결국엔 전체 손실 함수는 localization loss (loc)와 confidence loss (conf)의 합으로 나타난다.

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g))$$

- N : gt box와 매칭 된 default box의 개수 (N이 '0'이면 loss 또한 '0'이다.)
- l : 예측한 box
- g : gt box
- c : confidence score(물체일 확률)
- loc는 l과 g을 파라미터로 가지는 Smooth L1 loss function이다.
- α : weight term이며, cross validataion에서 1로 설정된다.

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx, cy, w, h\}} x_{ij}^k \text{smooth}_{\text{L1}}(l_i^m - \hat{g}_j^m)$$
$$\hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx})/d_i^w \qquad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^h$$
$$\hat{g}_j^w = \log\left(\frac{g_j^w}{d_i^w}\right) \qquad \hat{g}_j^h = \log\left(\frac{g_j^h}{d_i^h}\right)$$

- Faster-RCNN과 똑같이 box들의 좌표는 중심 좌표(cx, cy), 너비(w), 그리고 높이(h)를 가진다.
- g^은 gt box와 default box의 차이를 뜻한다.

$$L_{conf}(x, c) = -\sum_{i \in Pos}^{N} x_{ij}^p log(\hat{c}_i^p) - \sum_{i \in Neg} log(\hat{c}_i^0) \quad \text{where} \quad \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}$$

- conf의 경우는 c를 파라미터로 가지는 softmax loss function이다.
-

Choosing scales and aspect ratios for default boxes

다른 크기와 비율의 object를 인식하는 방법에는 여러 가지가 있다. 4, 9, 10, 그리고 11가 있다. 본 논문에서는 여러 가지의 크기와 비율을 가진 default box들을 이용한다. 다른 차원의 feature map들을 사용한 네트워크는 성능이 좋습니다. SSD 또한 다른 차원의 feature

map을 사용합니다. 위의 _Convolutional predictors for detection_의 그림에서와 같이 다른 차원의 feature map에서 default box를 가져와 인식에 사용합니다.

$$s_k = s_{\min} + \frac{s_{\max} - s_{\min}}{m - 1}(k - 1), \quad k \in [1, m]$$

- m : feature map의 개수.
- s : default box들의 크기
- min : 0.2 (가장 낮은 차원의 feature map 크기)
- max : 0.9 (가장 높은 차원의 feature map 크기)

  결과적으로 1 셀당 6개의 default box들이 생겨난다. 만약, 5x5 feature map이라고 하면 5x5x6x(c+1)의 박스가 생겨난다.

  더 큰 feature map에서는 작은 object를 더 작은 feature map에서는 큰 object를 인식할 수 있다. 그 이유는 셀이 많은수록 default box의 크기가 작아 object가 크다면 인식하지 못하기 때문이다.

Hard negative mining

matching 단계가 지난 후에는 대부분의 default box들은 negative 일 것입니다. 이것은 positive와 negative 학습 데이터의 언밸런스 문제로 이어집니다. 따라서 모든 negative 데이터를 사용하는 것이 아니라 negative : positive = 3:1 로 사용합니다. 본 논문을 제작한 분들이 이 비율이 제일 optimal 하다고 합니다.

data augmentation

input으로 사용할 object는 다양한 크기와 모양이 필요합니다. 따라서

- 0.1, 0.3, 0.5, 0.7, or 0.9의 jaccard overlap을 사용합니다.
- 무작위로 sample들을 사용합니다.

  각 sample을의 크기는 기존의 이미지의 0.1 에서 1배의 크기를 가지며 비율 또한 2배 또는 1/2배 입니다. 물론, 이미지의 중앙을 기준으로 적용시킵니다. 위의 샘플링 작업이 끝난 뒤, 원래의 크기로 resize 하고 50%의 확률로 뒤집어 적용시킵니다.마지막으로는 14처럼 왜곡을 적용합니다.

3. Experimental Results

여기에서는 실험 결과가 아니라 여러 실험을 통해 나온 팁을 적을 것입니다.

base network base network로는 VGG16을 적용시켰다고 합니다. 하지만 완벽하게 그대로 사용한 것이 아니라 layer를 조금 변형시켜 적용했다고 합니다.

3.2 Model analysis

Data augmentation이 중요합니다.

	SSD300				
more data augmentation?		✔	✔	✔	✔
include $\{\frac{1}{2}, 2\}$ box?	✔		✔	✔	✔
include $\{\frac{1}{3}, 3\}$ box?	✔			✔	✔
use atrous?	✔	✔	✔		✔
VOC2007 `test` mAP	65.5	71.6	73.7	74.2	**74.3**

data augmentation을 적용시켰을 때 8.8%의 성능이 향상되었다고 합니다.

변형한 VGG16이 빠릅니다.

SSD에서는 VGG16을 사용했는데 완전하게 쓴 것이 아니라 conv11_2를 제거하고 사용하였다. 그렇게 진행하니

Prediction source layers from:						mAP use boundary boxes?		# Boxes
conv4_3	conv7	conv8_2	conv9_2	conv10_2	conv11_2	Yes	No	
✔	✔	✔	✔	✔	✔	74.3	63.4	8732
✔	✔	✔	✔	✔		**74.6**	63.1	8764
✔	✔	✔	✔			73.8	68.4	8942
✔	✔	✔				70.7	69.2	9864
✔	✔					64.2	64.4	9025
	✔					62.4	64.0	8664

여러 feature map을 사용하는 것이 성능을 향상시킵니다.

위의 표에서 6개의 feature map을 사용하는 것이 아닌 마지막 feature map을 사용하지 않고 5개의 feature map을 사용하는 것이 더 좋은 성능이 나옵니다.

5. Conclusions

SSD는 multiple categories 1-stage detector이다. 제일 큰 특징은 여러 개의 featuremap으로부터 다양한 크기의 bbox를 가져오는 것이다. 이것이 매우 효과적으로 성능 향상에 도움이 되었다. 다른 obejcet detector(faster rcnn, yolo 등)와 비교하여 높은 인식률과 속도를 보여줍니다. 마지막으로 rnn이나 영상에서 object tracking에서도 잘 사용될 것이다.