RetinaNet

Abstract

객체 인식에는 one-stage 방식과, two-stage방식이 있지만 대체로 R-CNN에 의해 대중화된 Two-stage 방식이 우월하였다. one-stage 방식은 단순하고 빨랐으며, two-stage방식은 정확도가 높았다.

이러한 이유에는 객체 인식 과정에서 사진 속 객체와 배경이 차지하는 비율에서 나온다. 따라서 RetinaNet은 이러한 class 불균형을 해결하고자 제안한다. 그리하면 RetinaNet은 two-stage방식보다 정확도가 높으며 one-stage 방식의 속도를 유지할 수 있다.

Introduction

Two-stage방식의 경우 첫 번째 단계로 객체의 후보 위치를 뽑아내고 각 후보를 CNN을 이용하여 클래스를 구분한다. 여기에는 R-CNN 계열의 모델 등이 있다. One-stage 계열의 모델은 YOLO, SSD같은 모델이 있으며 이 방법은 Two-stage의 경우보다 10-40% 정도 더 빠르다.

따라서 논문에서는 최초로 2단계 모델의 정확도를 가진 1단계 모델을 제시한다. Two-stage방식의 경우에는 객체 후보 지역 선택 과정 중 REGION PROPOSAL NETWORK, EDGEBOXES, DEEPMASK 등의 방식을 통해 배경 부분을 필터링 처리한다. 반대로 One-stage방식은 배경을 필터링하지 못하기 때문에 배경 부분의 영향을 줄여야 한다. 기존에 사용하는 크로스 엔트로피에서 수정된 focal loss를 제안한다.

One-stage방식의 모델은 이미지당 104-105개의 후보 위치를 평가하는데, 다만 객체는 일부 위치에만 포함되어 있다. 이러한 불균형은 Negative로 쉽게 판단할 수 있게 되어 모델의 학습에 좋지 못하다. 또한, 전부 Negative로 판단하며 모델이 퇴보할 수도 있다.

$$\mathrm{CE}(p, y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1-p) & \text{otherwise.} \end{cases} \qquad (1)$$

기존 크로스 엔트로피

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{otherwise,} \end{cases} \qquad (2)$$

위의 2번 그림에 –log를 씌우게 되면 크로스 엔트로피가 된다. 논문에서는 표기 편의를 위해 CE(p, y) = CE(Pt) = -log(Pt)라 쓴다.

Balanced Cross Entropy

$$\mathrm{CE}(p_t) = -\alpha_t \log(p_t). \qquad (3)$$

기존에 클래스 불균형을 해결하기 위해서는 class 1에 가중치 α를 주며 class 0에 가중치 (1 - α)를 사용한다. 우리는 여기서 유사한 확장된 방식을 사용한다.

가중 인자 α는 [0,1]를 사용하며 Pt와 작동 방식이 같다. 가중 인자 α는 positive/negative sample사이의 균형을 잡아준다. 훈련되는 변수로 처리될 수 있다.

Focal Loss Definition

$$\mathrm{FL}(p_t) = -(1 - p_t)^{\gamma} \log(p_t). \qquad (4)$$

손실의 비중은 쉽게 분류되는 부정이 손실의 대부분을 차지하였다. P=0.1일 경의 otherwise(배경)이라면 Pt=0.9가 됨. 따라서 Focal Loss를 통해서 이를 해결한다. Focal Loss의 두 가지 특성을 주목할 수 있는데

1. P=0.9일 때 otherwise(배경)이라면 Pt=0.1이 되고, 변조계수((1-Pt))는 1에 가까워 기존 손실을 유지할 수 있다.

2. P=0.9일 때 Y=1이라면 Pt=0.9가 되고, 변조계수((1-Pt))는 0에 가까워 쉽게 예측한 부분에 대한 가중치를 조정할 수 있다. 이는 P=0.1일 때 otherwise(배경)인 경우에도 동일하게 적용된다. 변조 인자 r의 경우에는 변조계수에 효과를 조정하며 실험에서 r=2일 경우 가장 잘 작동하는 것을 밝혀냈다.

실제로 r=0.2, Pt=0.9일 때, CE크로스 엔트로피보다 100배 적은 LOSS를 가지고, Pt=0.968일 때는 1000배 적은 loss를 가진다.

$$\mathrm{FL}(p_t) = -\alpha_t(1 - p_t)^{\gamma} \log(p_t). \qquad (5)$$

최종적으로는 위와 같은 손실 함수를 채택하였다. 이는 α-balanced focal loss라 하며 기존 focal loss보다 약간의 성능 개선을 확인한다. Focal loss의 형태는 크게 중요하지는 않다.

Class Imbalance and Model Initialization

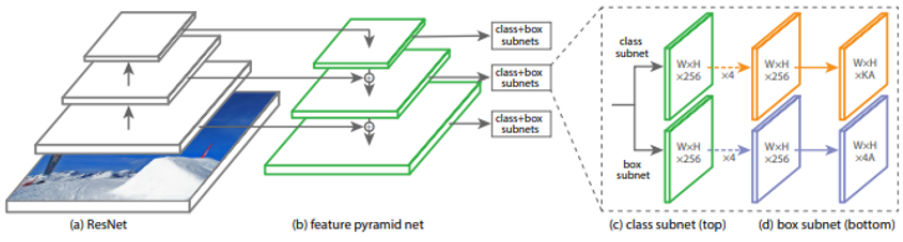
이진 분류 모델은 class에 대해 동일한 확률을 출력하게 초기화 되는데, 이러한 초기화는 imbalanced class에서 빈도가 높은 class가 우세해지기 때문에 prior개념을 통해 classification초기화를 마치 rare class의 확률로 초기화를 하도록 bias를 이용하여 0.01이 되게끔 초기화를 합니다. 이를 통해 훈련 안전성을 높인다.

https://csm-kr.tistory.com/5

Class Imbalance and Two-stage Detectors

Two-stage model은 one-stage를 이용해 positive negative sample의 수를 조정하는데 이는 가중 인자 α와 같다. 또한 이러한 메커니즘을 다루기 위해서 focal loss가 고안되었다.

RetinaNet Detector



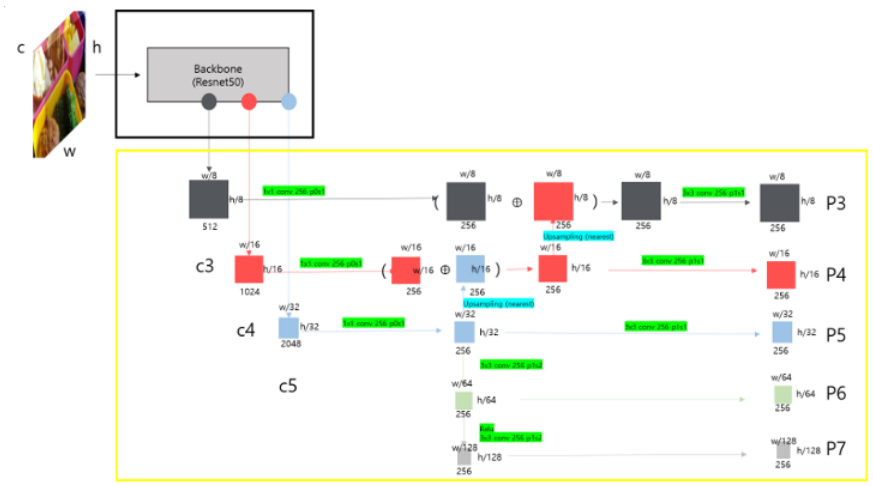
RetinaNet은 RESNET을 틀로 사용했으며, FPN network를 적용하였습니다.

그림(a)과정에서 P3~P7까지의 레벨로 피라미드를 구성한다. P3 -> 1/2^3수준으로 진행.

INPUT IMAGE가 BATCH, 3, 600, 600으로 들어갈 시 (RESNET50기준)

P3~P5는 (512, 75, 75), (1024, 37, 37), (2048, 18, 18)로 완성이 된다.

완성된 P5를 통해 채널을 256으로 줄이는 Pointwise Convolution을 진행하고 이를 이용해 Depthwise Convolution을 진행하여 (256, 9, 9)를 만들고 다시 이를 활용해 (256, 4, 4)를 생성한다.



P3는 다시 upsampling을 진행하며 기존 것과 합치며 output을 만들게 된다.

이렇게 만든 output을 통해 Anchors를 뽑아내는데 위의 모델의 크기를 고려하여 2^5^2~2^9^2을 넓이로 갖도록 제안한다. Anchors는 P3~P7의 각 셀 하나 당 aspect ratios {1:2, 1:1, 2:1}비율로 만들며 {2 0 , 2 1/3 , 2 2/3 }크기의 Anchors를 추가하도록 한다. 이는 각 셀 당 9개의 Anchors를 생성하며, P3의 경우 75*75*9의 output을 만든다. 이는 P7까지 더할 시 총 67995개의 Anchor를 사용한다. 각 Anchor에는 class subnet(본문에서는 class 80개)과 box subnet(x1, x2, y1, y2 좌표)을 통해 각각 classification target과 box의 target으로 활용된다. Anchor 중 ground truth와 IoU가 0.5이상일 시 ground truth로 사용하고, 0~0.4일 시 background로 할당이 되며, 0.4~0.5인 anchor는 무시된다.

Experiments

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [16]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [20]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [17]	Inception-ResNet-v2 [34]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [32]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [27]	DarkNet-19 [27]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [22, 9]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [9]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet (ours)	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet (ours)	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2