[Paper Review]

Focal Loss for Dense Object Detection

ICCV 2017

2023.05 김다혜

Index

Introduction

❖ One-stage detector vs Two-stage detector

Object detector는 one-stage와 two-stage로 분류할 수 있다.

- **"One-stage" object detector**

  - Localization과 Classification 동시에 수행한다.

  ex) YOLO, SSD...

  - 기존 one-stage model은 빠르지만 two-stage에 비해 낮은 성능을 보인다.

- **"Two-stage" object detector**

  - First stage : 후보 object 위치를 제안하는 단계 (Region proposal)

  - Second stage : 각각의 후보 위치에 대한 classify하는 단계

  ex) FPN, Mask R-CNN, Faster R-CNN ...

  => 많은 Two- stage 모델 SOTA 달성 (on COCO benchmark)

Introduction

❖ **One-stage**에서의 Class Imbalance 문제

: Foreground-background class imbalance

Feature Map의 grid 마다 anchor box를 적용하기 때문에 후보 object locations의 개수가 많다.

=> 학습 중 배경에 대한 box를 출력하면 오류라고 학습이 되지만, 그 빈도수가 너무 많아 학습에 방해가 됨

❖ **Two-stage**에서 Class imbalance 덜 민감한 이유

- **First stage (= Proposal stage)**

  : Selective search, EdgeBoxes, DeepMask, RPN과 같은 region proposal 알고리즘 사용

  => "background samples 필터링"

- **Second stage (= Classification stage)**

  : fixed foreground-to-background ratio (1:3) 또는 Online Hard Example Mining (OHEM) 사용

  => "foreground - background balance 유지"

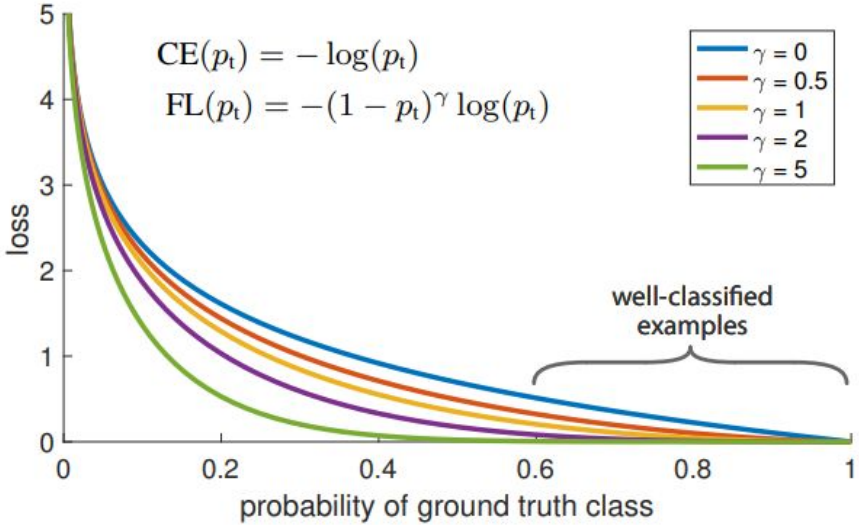
➔ One-stage에서 class imbalance 해결할 방법 ?

Introduction

❖ Focal Loss

본 논문은 새로운 loss function인 Focal Loss 을 정의하여 class imbalance 로 인한 문제를 해결하고자 했고,

One-stage detector만으로도 two-stage를 능가하는 성능을 달성했다.



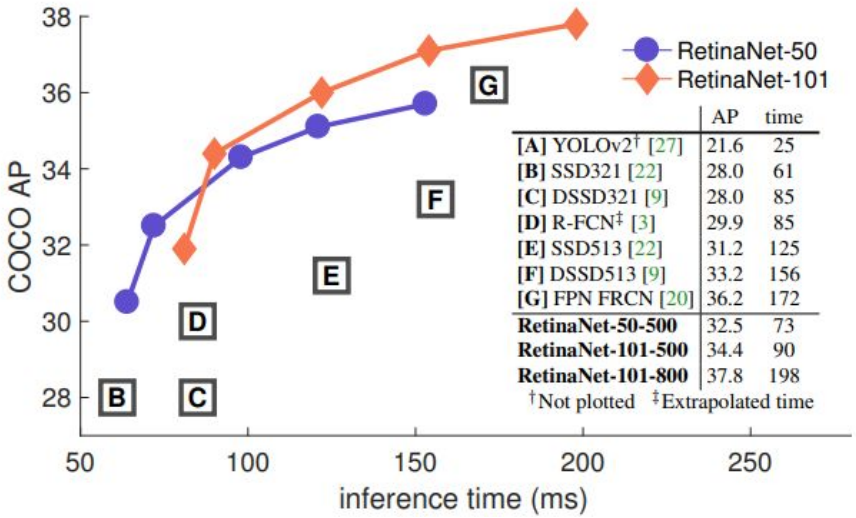
➔ Easy negative (background로 쉽게 분류되는 샘플)에 대한 weight를 낮추어 개수가 많더라도 모델을 압도하지 못하도록 함

Introduction

❖ RetinaNet

Focal Loss의 성과를 효과적으로 보여주기 위해, 새로운 one-stage detector인 RetinaNet 을 소개한다.

RetinaNet는 anchor box를 사용한 FPN 에 Focal Loss를 적용한 모델



	AP	time
[A] YOLOv2† [27]	21.6	25
[B] SSD321 [22]	28.0	61
[C] DSSD321 [9]	28.0	85
[D] R-FCN‡ [3]	29.9	85
[E] SSD513 [22]	31.2	125
[F] DSSD513 [9]	33.2	156
[G] FPN FRCN [20]	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	37.8	198

†Not plotted ‡Extrapolated time

다른 detector과의 성능 비교

➔ 본 논문에서 소개하는 Focal Loss를 적용한 RetinaNet 모델은 **기존 SOTA 모델보다 정확하면서도 빠른 성능**을 보인다.

Related work

❖ Two-stage Detectors

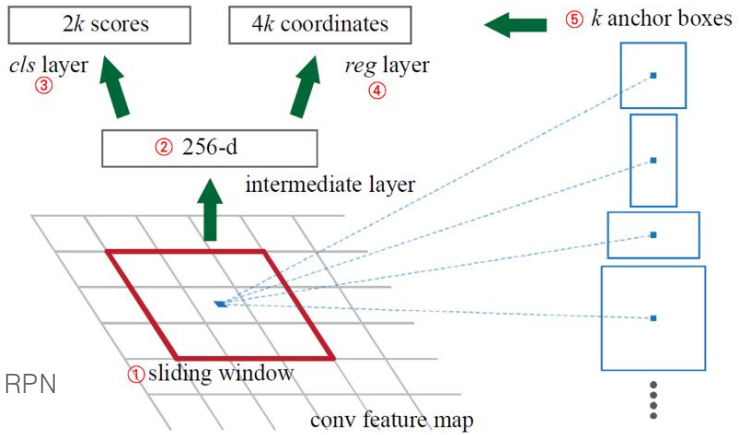
대표적인 모델 : **R-CNN, Faster R-CNN**

- 관심있는 지역(region proposal)을 추출하는 stage와, 물체를 분류 및 detect하는 stage로 분리
- **R-CNN** : 관심있는 지역을 추출하기 위해 Selective Search 알고리즘을 사용한다.

  * Selective Search = Over segmentation된 이미지를 여러번 반복하여 유사한 Region끼리 그룹핑을 진행한 뒤, bounding box를 생성하는 알고리즘

- **Faster R-CNN** : 관심있는 지역 추출 위해 Selective Search 대신 RPN(Region Proposal Network)을 사용한다.

  * RPN = 원본 이미지에서 나온 feature map에 sliding window를 통해 anchor box를 생성하여 region proposals 생성하는 네트워크



Related work

❖ One-stage Detectors

대표적인 모델 : **YOLO, SSD**

- **YOLO** : 이미지 전체에 대해서 하나의 신경망(a single neural network)이 한 번의 계산만으로 bounding box와 클래스 확률을 예측하는 end-to-end 방식이다.

- **SSD** : 여러 개의 feature map을 사용함으로써 bounding box 예측함으로써 작은 물체까지 detect 할 수 있지만, two-stage detector에 비해 낮은 AP를 보인다.

➔ RetinaNet 는 RPN에서 나온 Anchor box 개념과 SSD와 FPN에서 나온 Feature Pyramids 개념을 이용한다.

Related work

❖ Class Imbalance

- 기존 One-stage(ex. SSD..) 에서의 Class Imbalance

  1) 대부분의 easy negatives 는 **학습에 불필요한 배경**이기 때문에 **학습이 비효율적**이다. (inefficient)

  2) easy negative 가 압도적으로 많아 모델을 압도하는 문제가 발생 (degenerate models)

  -> hard negative mining / reweighting / OHEM 방법 으로 해결

* hard negative mining : 학습 도중에 hard negative examples을 sampling하여 학습에 다시 사용
* OHEM : Loss 값이 큰 Region(=Hard Example) 중 N개를 다시 학습에 사용

➔ Focal Loss를 통해 추가 sampling / reweighting 없이 효율적으로 Imbalance 문제를 해결할 수 있다.

Related work

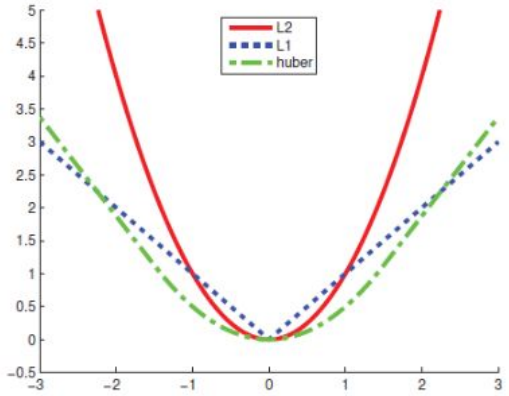
❖ Robust Estimation

- 대표적인 robust loss function : **Huber loss**

$$L_\delta(a) = \begin{cases} \frac{1}{2}a^2 & \text{for } |a| \le \delta, \\ \delta(|a| - \frac{1}{2}\delta), & \text{otherwise.} \end{cases}$$

에러가 클 때 outlier에 민감한 L2-error 대신 L1-error 적용한다.

-> hard examples의 loss를 반영을 줄임



L2, L1, Huber loss function

- **Focal Loss**
  - Huber Loss와 반대로, inliers의 민감도를 줄임으로써 class 불균형 문제 해결한다.
  - easy examples의 loss 반영을 줄임 -> 개수가 많더라도 total loss는 크지 않도록 함

➔ 즉, Focal Loss는 Hard Example 학습에 초점을 맞춘다.

Focal Loss

Focal Loss는 기존 Cross Entropy loss 에서 발전되었다.

❖ Cross Entropy loss

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

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

일 때,

해당 class가 존재할 확률

Cross Entropy는 다음과 같이 정의된다.

$$\mathrm{CE}(p, y) = \mathrm{CE}(p_t) = -\log(p_t).$$

이 때, P_t가 0.5보다 커도(= box에 물체가 존재할 확률이 50%가 넘어도) loss가 꽤 존재한다.

➔ easy examples가 많으면, 이러한 loss들이 쌓여서 물체를 제대로 검출하지 못하는 방향으로 학습이 될 수 있음

Focal Loss

❖ Balanced cross entropy loss

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

- Balanced cross entropy loss는 Class Imbalance를 해결하기 위해,
  Cross Entropy에 **weighting factor α ∈ [0, 1]** 를 추가했다.
  - 검출할 클래스 : α 값을 0~1 사이로 적용
  - 배경 : 1-α 를 적용

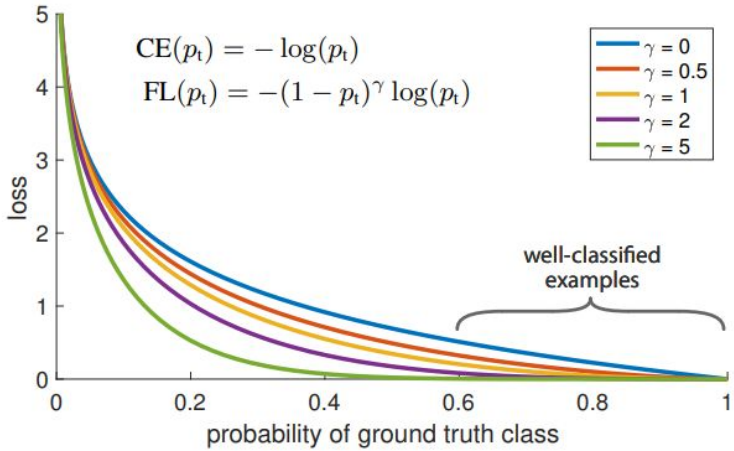
➔ Positive / Negative example은 balancing 하지만,
easy(p_t > 0.5) / hard example(p_t < 0.5)의 imbalance를 해결하지는 못함

Focal Loss

❖ Focal Loss Definition

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

- α-balanced CE loss에 term을 추가하여, easy example에 대한 가중치를 줄이고, hard negative example의 학습에 초점을 맞출 수 있다.
- easy/hard example 뿐만 아니라 positive/negative example에 대한 영향도 반영할 수 있음

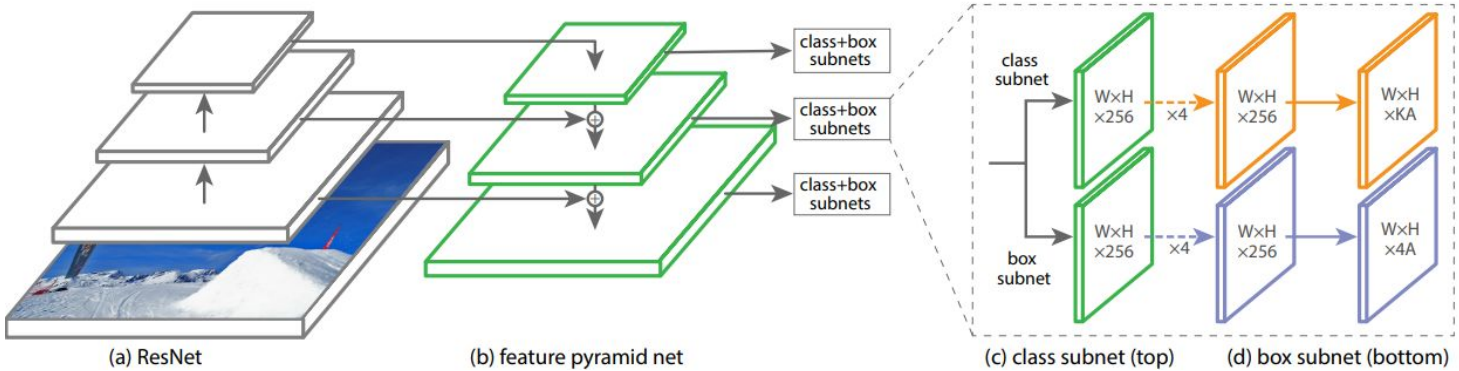


➔ ɣ 값이 커질 수록 **easy example의 loss 값은 더 작아진다.**

➔ ɣ= 0 일 때 -> Cross Entropy 와 같아짐

RetinaNet Detector

- RetinaNet는 FPN backbone과 두 개의 subnet(class & box regression)을 사용한다.



- **FPN backbone**
  - 피라미드의 각각의 level에서 각각 다른 scale로 object를 탐지한다.
  - RetinaNet은 larger object detection을 위해 기존 FPN 구조에서 level p7을 추가했다. (level P3~P7)
- **Subnet1 : Classification Subnet**
  - Anchor box 내의 object가 존재할 확률을 예측
- **Subnet2 : Box Regression Subnet**
  - Anchor box와 Ground Truth box를 비교하여 regression을 진행

RetinaNet Detector

❖ Feature Pyramid Network Backbone

Feature Pyramid Network 구조

FPN은 Single-scale 이미지를 convolutional network에 입력하여 다양한 scale의 feature map을 출력하는 네트워크이다.

(1) 각 level에 있는 feature map을 upsampling하고 channel 수를 동일하게 맞춰주는 **Top-down Pathway** 과정과,

(2) Upsampled feature map과 아래 level의 feature map의 **element-wise addition 연산**을 수행한다.

➔ Multi-scale feature map을 출력하기 때문에 높은 detection 성능을 보인다.

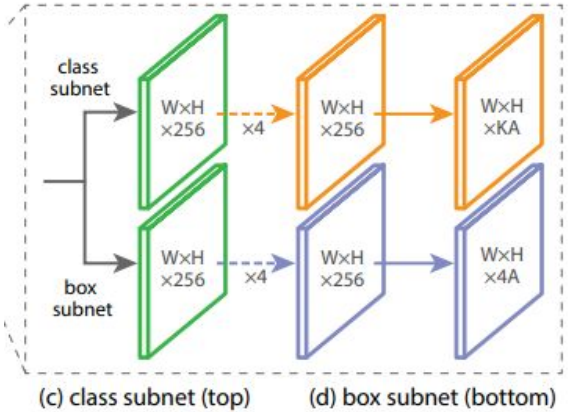
➔ 실제로 사전 실험에서 ResNet layer 만 사용했을 때 보다 나은 AP 성능을 보임

RetinaNet Detector

❖ Classification Subnet

- 각 FPN level에 붙어있는 Fully Convolution Network
- input feature map에 4개의 3*3 conv layers 적용 후, ReLU activations을 거친다.
- sigmoid 거친 KxA 개의 binary prediction을 구한다.

  (A = anchors 수, K = object classes)



❖ Box Regression Subnet

- Class Subnet과 마찬가지로 각 FPN level에 작은 FCN을 결합한다.
- 각 앵커 박스의 offset 4개(box_x_center, box_y_center, box_width, box_height)를 GT박스와 유사하게 regression 함
- 기존 연구와 달리, **'class-agnostic bounding box regressor'**를 사용한다.
  - Class 정보 없이 anchor box를 regression 하기 때문에, parameter 수가 적고, 효과적으로 성능 달성

RetinaNet Detector

❖ Inference and Training

- Inference
  - 빠른 추론을 위해, 각 FPN level에서 **가장 box prediction 점수가 높은 1,000개의 box만** result에 사용하였다.
  - 최종 detection에 NMS(non-maximum suppression)의 임계값을 0.5로 적용

- Training Settings
  - ImageNet으로 pre-train 된 **ResNet-50, ResNet-101** 모델 사용
  - FPN의 initialization은 FPN 논문과 똑같은 설정값으로 진행하였다.
  - 최적화 알고리즘으로 **SGD(Stochastic gradient descent)**를 사용
  - Class subnet의 output으로, **α 적용한 Focal Loss**를 사용 (실험 결과 $\alpha = 0.25$, $\gamma = 2$ 일때 최적의 성능)

Experiments

- 실험에서는 **(1) 다양한 loss functions을 적용한 결과**에 대해 분석하고, **(2) 기존 SOTA 모델과의 비교 분석**을 진행했다.
- 그 전에, prior 개념을 추가하여 Initialization을 진행했다.

❖ Network Initialization

- 기존의 classification model은 output이 1 혹은 -1로 고정됨
- 본 논문에서는 학습 안정성을 향상시키기 위해, object에 대한 모델이 추정한 확률 prior **(=π)**에 대한 개념을 추가하였다.
- **실험1)** proir 추가 없이 Cross entropy로 학습했을 때 -> 발산
- **실험2)** 모델의 마지막 레이어를 **π=0.01**로 초기화 하여 RetinaNet을 학습했을 때 -> COCO에서 AP 30.2의 성능을 보임

=> 실험 결과들이 π의 값에 민감하지 않았기 때문에, 뒤의 **모든 실험에서 π= 0.01로 설정하여 진행**하였다.

Experiments

❖ Balanced Cross Entropy & Focal Loss

α	AP	AP_{50}	AP_{75}
.10	0.0	0.0	0.0
.25	10.8	16.0	11.7
.50	30.2	46.7	32.8
.75	31.1	49.4	33.0
.90	30.8	49.7	32.3
.99	28.7	47.4	29.9
.999	25.1	41.7	26.1

(a) **Varying α for CE loss** ($\gamma = 0$)

γ	α	AP	AP_{50}	AP_{75}
0	.75	31.1	49.4	33.0
0.1	.75	31.4	49.9	33.1
0.2	.75	31.9	50.7	33.4
0.5	.50	32.9	51.7	35.2
1.0	.25	33.7	52.0	36.2
2.0	.25	**34.0**	**52.5**	**36.5**
5.0	.25	32.2	49.6	34.8

(b) **Varying γ for FL** (w. optimal α)

(a. α-CE loss) α-balanced CE를 통해 학습시킨 결과, α=0.75로 설정하였을 때 AP 0.9의 성능을 보였다.

(b. FL) 실험을 통해 **γ값에 따른 최적의 α값을 찾았다.**

- γ를 변화했을 때 성능 변화가 더 큰 것을 볼 수 있다.
- 실험 결과에 따라, γ=2.0, α=0.25로 설정하여 후속 실험을 진행했다. (α=0.5일때도 괜찮은 성능 보임)

Experiments

❖ Analysis of Focal Loss

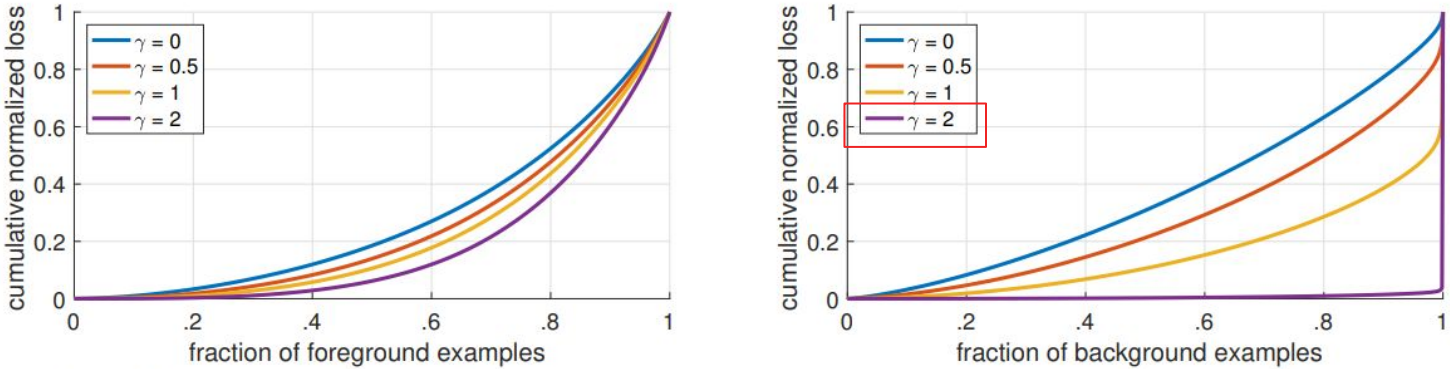


Figure 4. Cumulative distribution functions of the normalized loss for positive and negative samples for different values of γ for a *converged* model. The effect of changing γ on the distribution of the loss for positive examples is minor. For negatives, however, increasing γ heavily concentrates the loss on hard examples, focusing nearly all attention away from easy negatives. (ResNet-101 백본과 600픽셀 이미지로 학습)

학습 과정에서의 Focal Loss를 **Foreground와 Background로 나누어서** 누적분포함수를 그린 결과,

- Foreground에 대한 누적분포 함수를 보면(왼쪽), ɣ의값에 크게 영향을 받지 않는 것을 확인
- Background에 대한 결과(오른쪽)와 비교를 통해, Focal Loss가 Easy negatives의 영향을 낮추고, Hard negative example에 집중할 수 있도록 돕는 것을 확인

Experiments

❖ OHEM : Online Hard Example Mining vs Focal Loss

- **Focal Loss**는 easy samples의 영향력을 줄이는 것이지만 ,
- **OHEM**은 easy examples에 대해서는 계산을 제외하는 방식이다.

*OHEM : **모든 region proposals**를 forward pass한 후 loss를 계산하여,
높은 loss를 가지는 region proposals에 대해서만 backward pass를 수행하는 방법

method	batch size	nms thr	AP	AP_{50}	AP_{75}
OHEM	128	.7	31.1	47.2	33.2
OHEM	256	.7	31.8	48.8	33.9
OHEM	512	.7	30.6	47.0	32.6
OHEM	128	.5	32.8	50.3	35.1
OHEM	256	.5	31.0	47.4	33.0
OHEM	512	.5	27.6	42.0	29.2
OHEM 1:3	128	.5	31.1	47.2	33.2
OHEM 1:3	256	.5	28.3	42.4	30.3
OHEM 1:3	512	.5	24.0	35.5	25.8
FL	n/a	n/a	**36.0**	**54.9**	**38.7**

(d) **FL *vs.* OHEM** baselines (with ResNet-101-FPN)

- 두 methods를 비교한 결과,

  OHEM은 batch size=128, threshold=0.5 일 때 좋은 성능을 냈지만,

  Focal Loss 보다 좋은 성능을 내지는 못하였다.

Experiments

❖ Comparison to State of the Art

	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

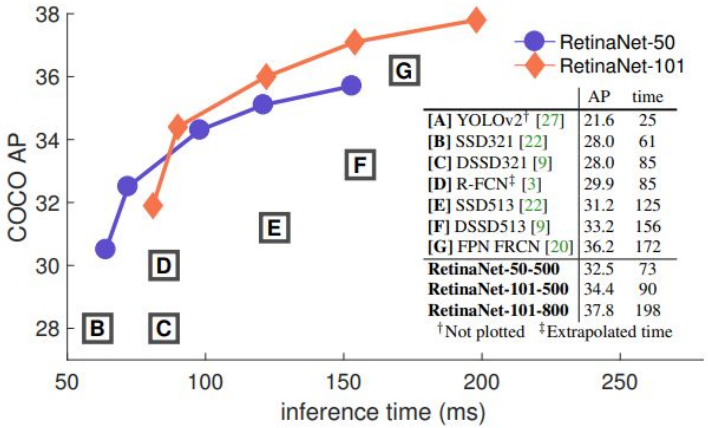
- COCO dataset으로 SOTA 모델 및 RetinaNet의 성능 평가 진행한 결과,
  - 기존의 모든 **one-stage methods 보다 최소 5.9AP 더 나은 성능**을 보임 (39.1 vs. 33.2)
  - **two-stage인 Faster R-CNN 계열 모델보다도 최소 2.3AP 나은 성능** 보임 (39.1 vs. 36.8)
  - ResNeXt32x8d-101-FPN을 backbone으로 적용했을 때, 성능이 더욱 향상됨 (40.8AP)
    (ResNext는 ResNet의 bottle neck에 Grouped convolution을 적용한 후속 모델)

Experiments

❖ Speed vs Accuracy

depth	scale	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	time
50	400	30.5	47.8	32.7	11.2	33.8	46.1	64
50	500	32.5	50.9	34.8	13.9	35.8	46.7	72
50	600	34.3	53.2	36.9	16.2	37.4	47.4	98
50	700	35.1	54.2	37.7	18.0	39.3	46.4	121
50	800	35.7	55.0	38.5	18.9	38.9	46.3	153
101	400	31.9	49.5	34.1	11.6	35.8	48.5	81
101	500	34.4	53.1	36.8	14.7	38.5	49.1	90
101	600	36.0	55.2	38.7	17.4	39.6	49.7	122
101	700	37.1	56.6	39.8	19.1	40.6	49.4	154
101	800	37.8	57.5	40.8	20.2	41.1	49.2	198

(e) **Accuracy/speed trade-off** RetinaNet (on `test-dev`)



- 일반적으로, backbone networks 클 수록 정확도는 높지만 추론 속도는 느려진다. **(trade-off)**
- 기존 methods와 비교했을 때, Focal Loss를 통해 학습한 RetinaNet이 정확도 대비 연산속도가 빠르다.
- Larger Scale에서도 RetinaNet이 기존의 모든 Two-stage Detector보다 속도와 정확도 모두 앞선다.

  -> Focal Loss를 통해 one-stage 구조임에도 높은 정확도 달성 가능

Conclusion

- Class imbalance 문제는 one-stage detector가 two-stage의 성능을 능가하지 못하는 주요 요인이다.
- 본 논문은 Focal Loss를 정의하여
  - ❖ Focal Loss $\mathrm{FL}(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t).$
    - ➔ modulating term을 둠으로써 hard negative examples에 집중할 수 있도록 함
    - ➔ 간단하면서도 효율적으로 Class imbalance 로 인한 성능 저해 문제를 해결하였다.

- Focal Loss의 성과를 효과적으로 보여주기 위해, 새로운 One-stage detector인 RetinaNet을 소개하였다.

  (1) Feature Pyramid Network을 활용하여 Multi-scale feature map을 추출

  (2) Focal Loss 적용

  - ➔ 높은 Detection 성능 보이고, 기존 SOTA 모델을 능가하는 AP를 달성하였다.