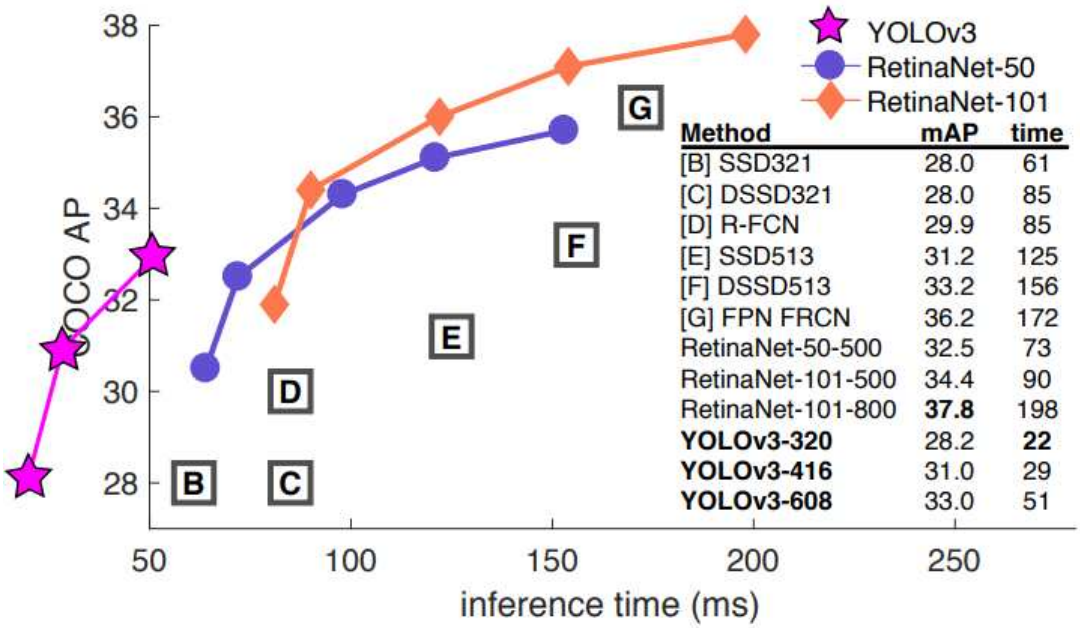
YOLOv3: An Incremental Improvement

■ Abstract

- 이전 YOLOv2 모형보다 조금 더 큰 새로운 모형입니다.
- 이전 모형보다 성능이 향상되었고 빠른 속도를 가지고 있습니다.



Method	mAP	time
[B] SSD321	28.0	61
[C] DSSD321	28.0	85
[D] R-FCN	29.9	85
[E] SSD513	31.2	125
[F] DSSD513	33.2	156
[G] FPN FRCN	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	**37.8**	198
YOLOv3-320	28.2	**22**
YOLOv3-416	31.0	29
YOLOv3-608	33.0	51

Figure 1. We adapt this figure from the Focal Loss paper [9]. YOLOv3 runs significantly faster than other detection methods with comparable performance. Times from either an M40 or Titan X, they are basically the same GPU.

- 예시)

1) (320,320)에서 yolov3는 28.2mAP에서 22ms로 실행되었습니다.
   SSD model만큼 정확하지만 3배 더 빠릅니다.
2) Titan X에서 51ms동안 57.9 AP50을 달성하였습니다.
   RetinaNet에서 198ms동안 57.5 AP50 달성하였습니다.
   성능은 비슷하지만 속도면에서 3.8배 더 빠릅니다.

■ 1. Introduction

- YOLOv2를 조금 변화를 줘서 실험한 게 YOLOv3입니다.
- paper(논문)의 종류로는 Accepted paper, Camera-ready paper가 있는데 YOLOv3는 이 중 Camera-ready paper입니다. 이때 source가 없어서 TECH

REPORT가 되었습니다.

- TECH REPROT section은 크게 다음과 같습니다.

1) The Deal(목표) : YOLOv3에 대한 전반적인 Architecture

2) How we do : Experiment (YOLO V3 결과 해석)

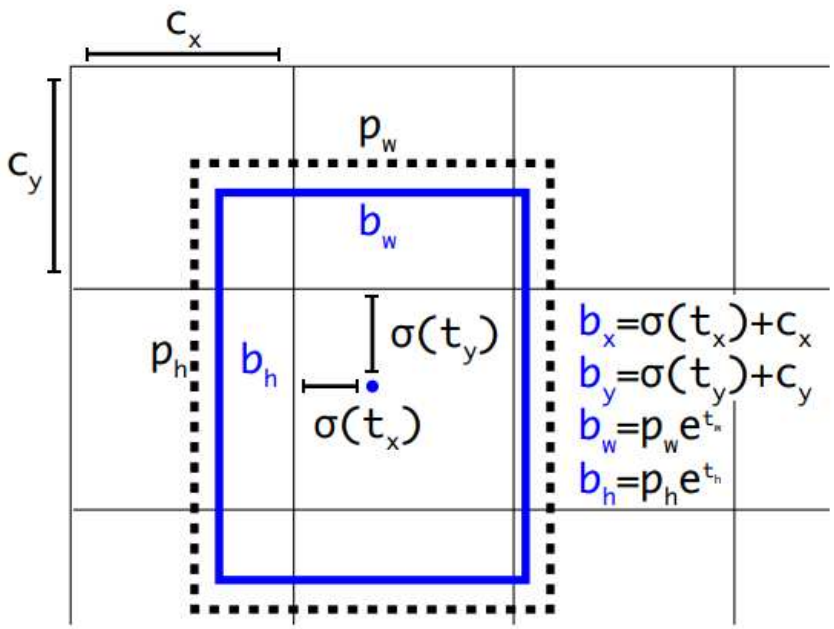
3) Things we tried that didn't work : Ablation Study(절제 연구) 개념

4) What this all means : Conclusion + 기존 evaluation 방식의 불만 제기

■ 2. The Deal

◆ 2.1. Bounding Box Prediction

- YOLO9000은 Anchor Box(YOLO에서는 prior box)로 dimention cluster를 사용하여 Bouding Box를 예측합니다.
- 각각의 Bounding Box로부터 4개의 좌표인 t_x, t_y, t_w, t_h를 예측합니다.



- 그림에서와 같이 각각의 좌표를 정의하자면 다음과 같습니다.

(c_x, c_y) : 각 grid cell의 좌상단 끝의 좌표

(P_w, P_h) : cluster로 예측한 Anchor Box의 width, height

(b_x, b_y) : GT(Ground Truth, 정답값)에 가까워지도록 학습되는 Anchor Box 중심

좌표

σ : logistic aactivate function

- 각 box의 중심점을 t_x, t_y에 sigmoid function을 적용하고 c_x,c_y를 표현함으로써 0에서 1 사이의 값으로 표현합니다.

$$b_x = \sigma(t_x) + c_x$$
$$b_y = \sigma(t_y) + c_y$$
$$b_w = p_w e^{t_w}$$
$$b_h = p_h e^{t_h}$$

- 학습하는 동안 우리는 SSE(Sun of Square Error) loss를 사용합니다.
- YOLOv3는 logistic regression을 사용하여 각각의 bounding box에 대해 objectness score를 예측합니다.

◆ 2.2. Class Prediction

- 각각 boxes는 class를 예측하는데 이때 여러 개를 분류하는 것을 multi-classification task라고 합니다.
- 일반적으로 여러 class를 classification하는 softmax를 사용하지 않고 independent logisitic classifier를 사용합니다.
- 그래서 학습시킬 때 BInary Cross Entropy를 사용합니다.

◆ 2.3. Predictions Across Scales

- YOLOv3는 3가지 다른 scales로 Boxes를 예측합니다.
- FPN(Feature Pytamid Networks)로 features를 추출합니다.
- 3개의 scale이 사용됨로 3개의 pyramid의 level에서 특징을 추출합니다.
- **3-d tensor encoding bounding box, objectness, class predictions**

COCO dataset으로 우리는 3가지 boxes를 예측합니다.
결과 tensor는 **N*N*[3*(4+1+80)]**인데 N은 grid 크기이고 boxes(4), object(1), class(80)입니다.

- Anchor Box를 생성할 때 **k-means clustering**을 사용한다. 3개의 scale에서 3개의 Box를 사용하므로 9개의 Boxes가 필요합니다.
- COCO dataset에서 k-meanas clustering을 적용하면 (10x13), (16x30), (33x23), (30x61), (62x45), (59x119), (116x90), (156x198), (373x326) Box가 생성됩니다.

◆ 2.4. Feature Extractor

- YOLOv2, Darknet-19 사이의 hybrid 접근법을 이용하여 새로운 network를 만들습니다.

- residual connection을 이용하여 layer를 더 deep하게 쌓게 되었고 그 결과 Darknet-53을 만들 수 있게 되었습니다. Darknet-53의 Architecture는 다음과 같습니다.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

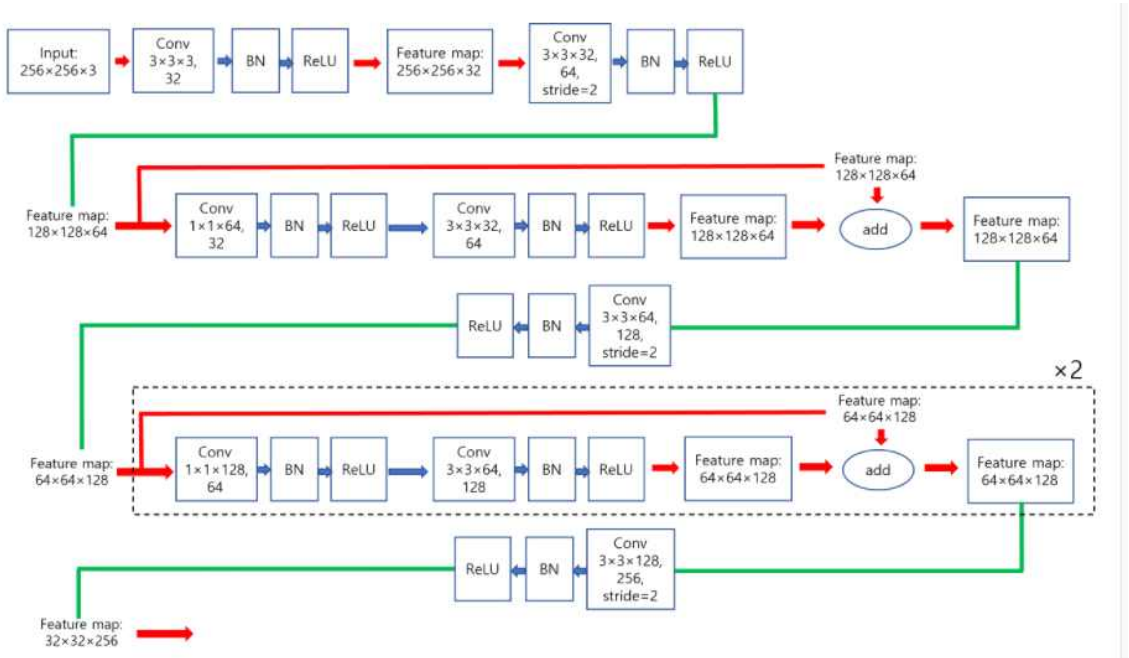
Table 1. **Darknet-53.**

Backbone	Top-1	Top-5	Bn Ops	BFLOP/s	FPS
Darknet-19 [15]	74.1	91.8	7.29	1246	**171**
ResNet-101[5]	77.1	93.7	19.7	1039	53
ResNet-152 [5]	**77.6**	**93.8**	29.4	1090	37
Darknet-53	77.2	**93.8**	18.7	**1457**	78

Table 2. **Comparison of backbones.** Accuracy, billions of operations, billion floating point operations per second, and FPS for various networks.

- 위 표를 살펴보면 Darknet-53이 Darknet-19보다 좋은 accuracy를 기록하고

있음을 알 수 있습니다.
-또한 Resnet-152보다 accuracy는 조금 떨어지지만 효율성 부분에서 Darknet-53이 더 좋다고 합니다. FPS, BFLOP/s값을 보고 효율성 부분에서는 더 뛰어나다고 생각했습니다.
- 이를 도식화하면 다음과 같습니다.



출처. https://89douner.tistory.com/109

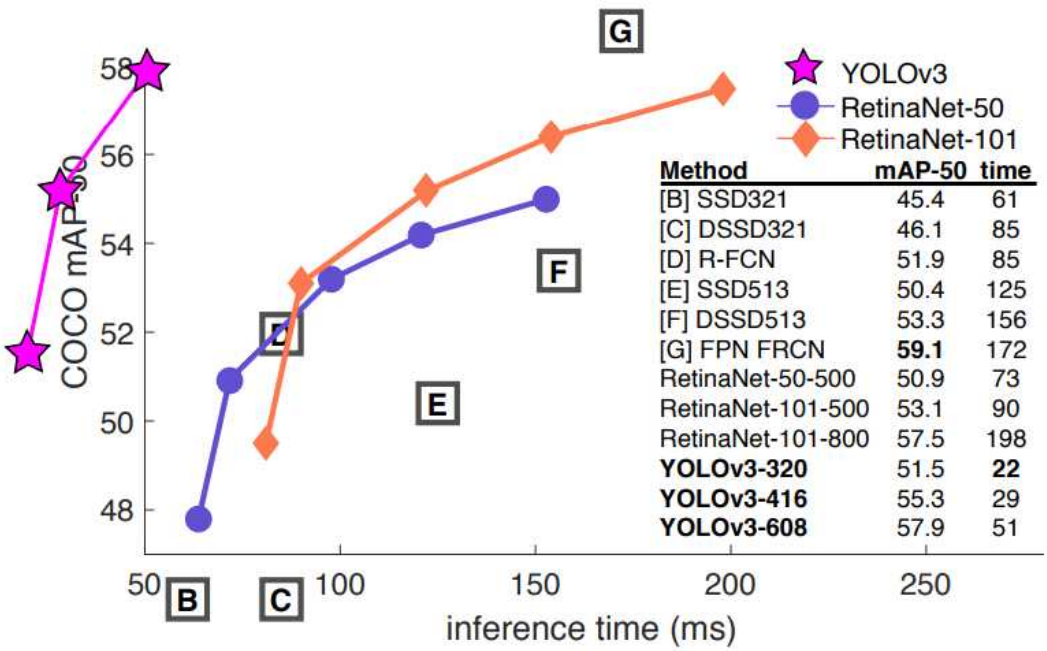
◆ 2.5. Training

- Full 이미지를 사용하여 학습을 합니다.
with no hard negative mining
or any of that stuff.
- 여러개의 scale와 많은 양의 데이터 Augmentation, Batch Normalization, 모든 평균적인 stuff를 이용하여 학습합니다.
- Darknet을 이용하여 학습과 검증에 사용합니다.

■ 3. How We Do

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [15]	DarkNet-19 [15]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [11, 3]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [3]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [9]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

- COCO dataset의 mAP 성능을 살펴보니 SSD와 동일한데 속도가 3배 빨랐습니다. 하지만, RetinaNet의 mAP 성능을 뛰어넘지는 못했습니다.
- IOU의 임계값을 올리면 AP 성능이 급격하게 떨어지는 것을 확인하였는데 이런 것으로 보아 YOLOv3 모형이 어떤 객체를 정확하게 detect하지는 못하는 것을 알 수 있습니다.
- 허나 YOLOv2와 YOLOv3의 AP_S를 살펴보면 small object에 대하여 많이 향상된 것을 확인할 수 있습니다.



- 성능 지표로 mAP가 아닌 mAP-50으로 보면 YOLOv3 성능이 좋습니다.

■ 4. Things We Tried That Didn't Work

시도는 했지만 실질적으로 도움이 안된 작업들을 소개합니다.

◆ 4.1. Anchor box x,y offset predictions

- 일반적으로 많이 사용하는 방법(R-CNN 계열 모형에서 쓰는 bounding box를 찾는 방법으로 box의 width나 height의 비율을 이용하여 값을 찾음)을 이용했는데 오히려 성능을 저하시키는 요인이 되었습니다.

◆ 4.2. Linear x.y predictions instead of logistic

- x,y 값을 직접적으로 뽑는 방법을 이용해보았다.

◆ 4.3. Focal loss

- Focal loss는 RetinaNet에서 썼던 loss값으로 class의 불균형을 해결하기 위한 loss인데 이거 역시 2 point정도 떨어트리는 안좋은 결과를 가져왔습니다.

◆ 4.4. Dual IOU thresholds and truth assignment

- Faster R-CNN에서 학습시킬 때 GT랑 pred를 보고 0.3 이하는 배경으로, 0.7 이상은 object로 판단합니다. 이때 0.3 ~ 0.7 사이는 network를 헷갈리게 한다고 해서 버리게 되는데 이런식으로 하는 방법을 Dual IOU라고 합니다. 이도 성능을 떨어트려 사용하지 않게 되었습니다.

■ 5. What This All Means

- YOLOv3는 좋은 detector입니다. 빠르고, 정확합니다.
- 하지만 COCO dataset을 기준으로 0.5에서 0.95까지 IOU를 키우면서 평가하는 방법에서는 좋지 않지만 전통적인 방법인 0.5 iou에서는 매우 좋게 나왔습니다.
- 사람들도 IOU가 0.3인거랑 0.5인거랑 구분을 시켰을 때 잘 못했습니다. 그렇게 나눠서 구별하는 것을 요구하는데 그게 정말 필요한지에 대한 내용입니다.

■ Reference

[1] https://arxiv.org/pdf/1804.02767.pdf

[2] https://89douner.tistory.com/109

[3] https://deep-learning-study.tistory.com/509

[4] https://www.youtube.com/watch?v=HMgcvgRrDcA&t=1741s