YOLO
(You Only Look Once)

YOLO(You Only Look Once)

https://pjreddie.com/

< YOLO 버전 별 요약 >

YOLOv1 : 2016년에 발표된 최초 버전으로, 실시간 객체 검출을 위한 딥러닝 기반의 네트워크
YOLOv2 : 2017년에 발표된 두 번째 버전으로, 성능을 개선하고 속도를 높인 것이 특징
YOLOv3 : 2018년에 발표된 세 번째 버전으로, 네트워크 구조와 학습 방법을 개선하여 객체 검출의 정확도와 속도를 모두 개선
YOLOv4 : 2020년 4월에 발표된 네 번째 버전으로, SPP와 PAN 등의 기술이 적용되어 더욱 정확한 객체 검출과 더 높은 속도를 제공
YOLOv5 : 2020년 6월에 발표된 버전으로 YOLOv4와 비교하여 객체 검출 정확도에서 10% 이상 향상되었으며, 더 빠른 속도와 더 작은 모델 크기를 가짐

https://velog.io/@qtly_u/n4ptcz54

< YOLO 버전 별 요약 >

YOLOv7 : 2022년 7월 발표된 버전으로, 훈련 과정의 최적화에 집중하여 훈련 cost를 강화화는 최적화된 모듈과 최적 기법인 trainable bag-of-freebies를 제안
YOLOv6 : 2022년 9월 발표된 버전으로, 여러 방법을 이용하여 알고리즘의 효율을 높이고, 특히 시스템에 탑재하기 위한 Quantization과 distillation 방식도 일부 도입하여 성능 향상
YOLOv8 : 2023년 1월 발표된 버전으로, YOLO 모델을 위한 완전히 새로운 리포지토리를 출시하여 개체 감지, 인스턴스 세분화 및 이미지 분류 모델을 train하기 위한 통합 프레임워크 로 구축됨

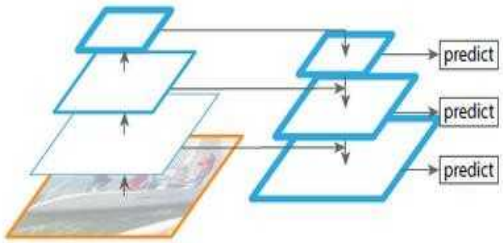
https://velog.io/@qtly_u/n4ptcz54

One Stage Detector

2015. 06	2015. 12	2016. 12		2017. 08	2018. 04	2019. 11		2020. 06
Yolo v1	SSD	Yolo v2		Retinanet	Yolo v3	EfficientDet		Yolo v5

Feature Pyramid Network

Feature Pyramid Network



Darknet 기반의 YOLO

YOLO V1 논문 저자

Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi

Darknet

C로 작성된 Deep Learning Framework

YOLO Version

Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	~ 1000 × 600
Fast YOLO	52.7	155	1	98	448 × 448
YOLO (VGG16)	66.4	21	1	98	448 × 448
SSD300	74.3	46	1	8732	300 × 300
SSD512	76.8	19	1	24564	512 × 512
SSD300	74.3	59	8	8732	300 × 300
SSD512	76.8	22	8	24564	512 × 512

V1

빠른 Detection 시간

그러나 낮은 정확도

V2

수행 시간과 성능 모두 개선

SSD에 비해 작은 Object 성능 저하

V3

V2 대비 수행 시간은 조금 느림

성능 대폭 개선

V4

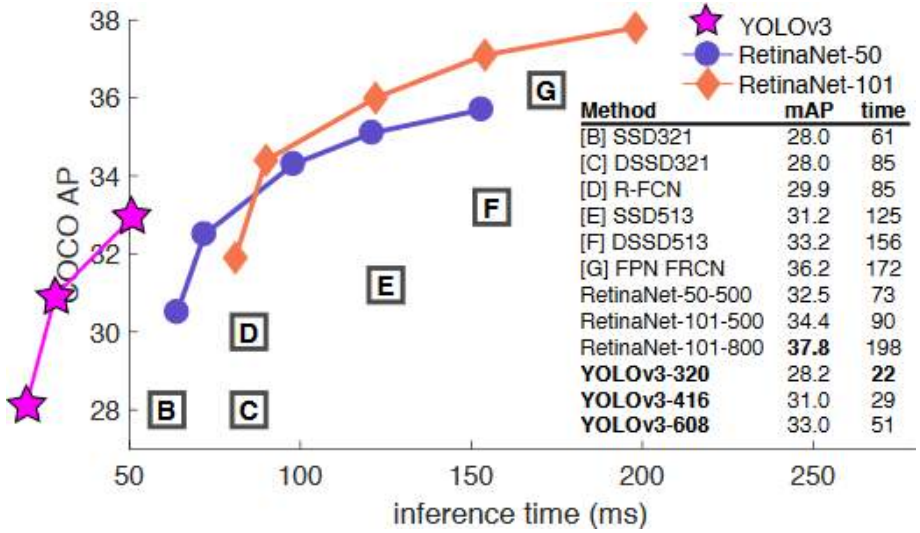
V3 대비 수행 시간 약간 향상

성능 대폭 개선

Yolo v3 성능

Abstract

We present some updates to YOLO! We made a bunch of little design changes to make it better. We also trained this new network that's pretty swell. It's a little bigger than last time but more accurate. It's still fast though, don't worry. At 320 × 320 YOLOv3 runs in 22 ms at 28.2 mAP, as accurate as SSD but three times faster. When we look at the old .5 IOU mAP detection metric YOLOv3 is quite good. It achieves 57.9 AP_{50} in 51 ms on a Titan X, compared to 57.5 AP_{50} in 198 ms by RetinaNet, similar performance but 3.8× faster. As always, all the code is online at https://pjreddie.com/yolo/.



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

Speed + Accuracy

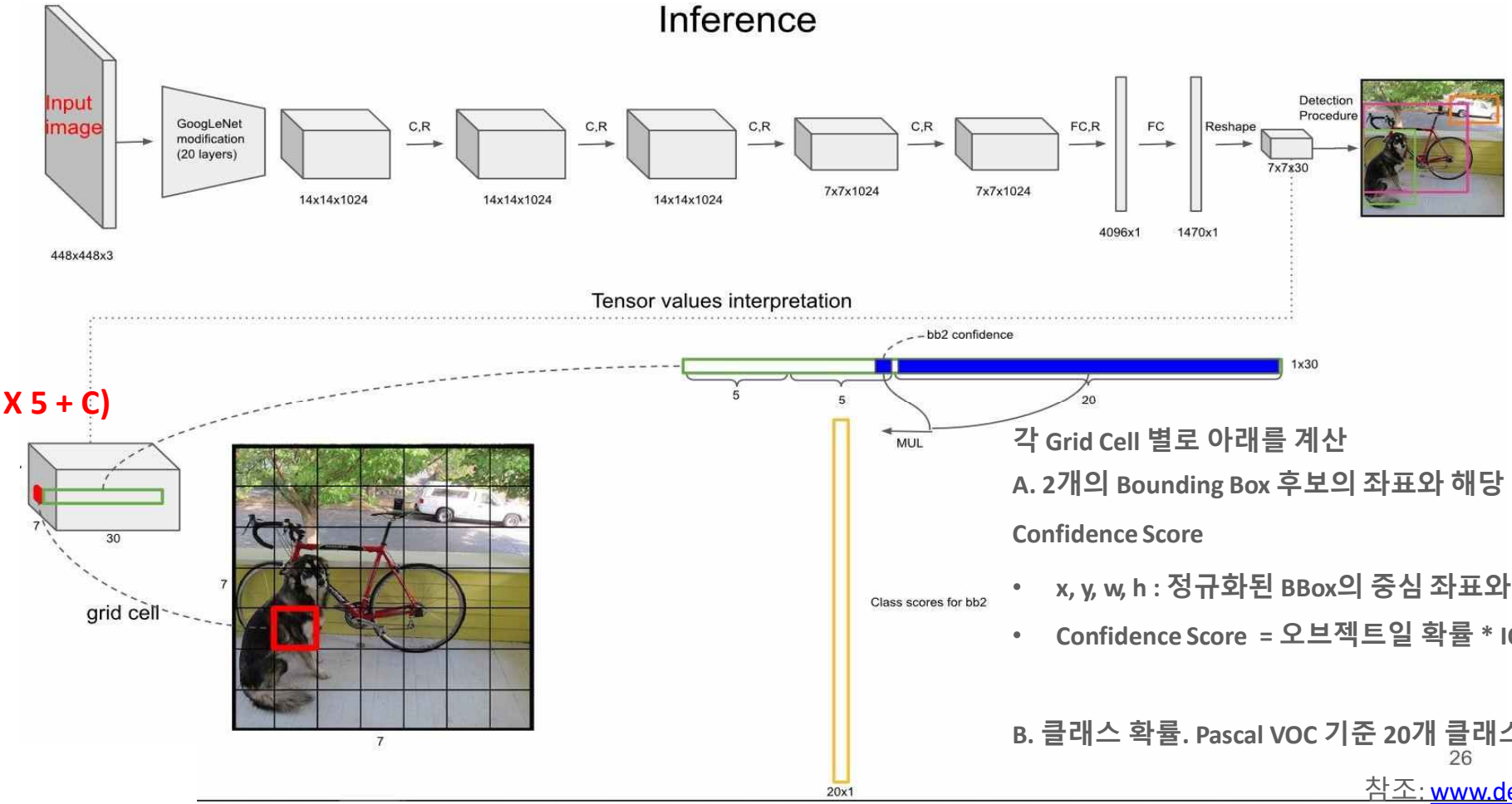
YOLO – V1

- Yolo V1 은 입력 이미지를 S X S Grid로 나누고 각 Grid의 Cell이 하나의 Object에 대한 Detection 수행
- 각 Grid Cell 이 2개의 Bounding Box 후보를 기반으로 Object의 Bounding Box 를 예측

448 X 448
Image

S x S Grid

YOLO-V1 네트워 및 Prediction 값



S X S X (B X 5 + C)

각 Grid Cell 별로 아래를 계산

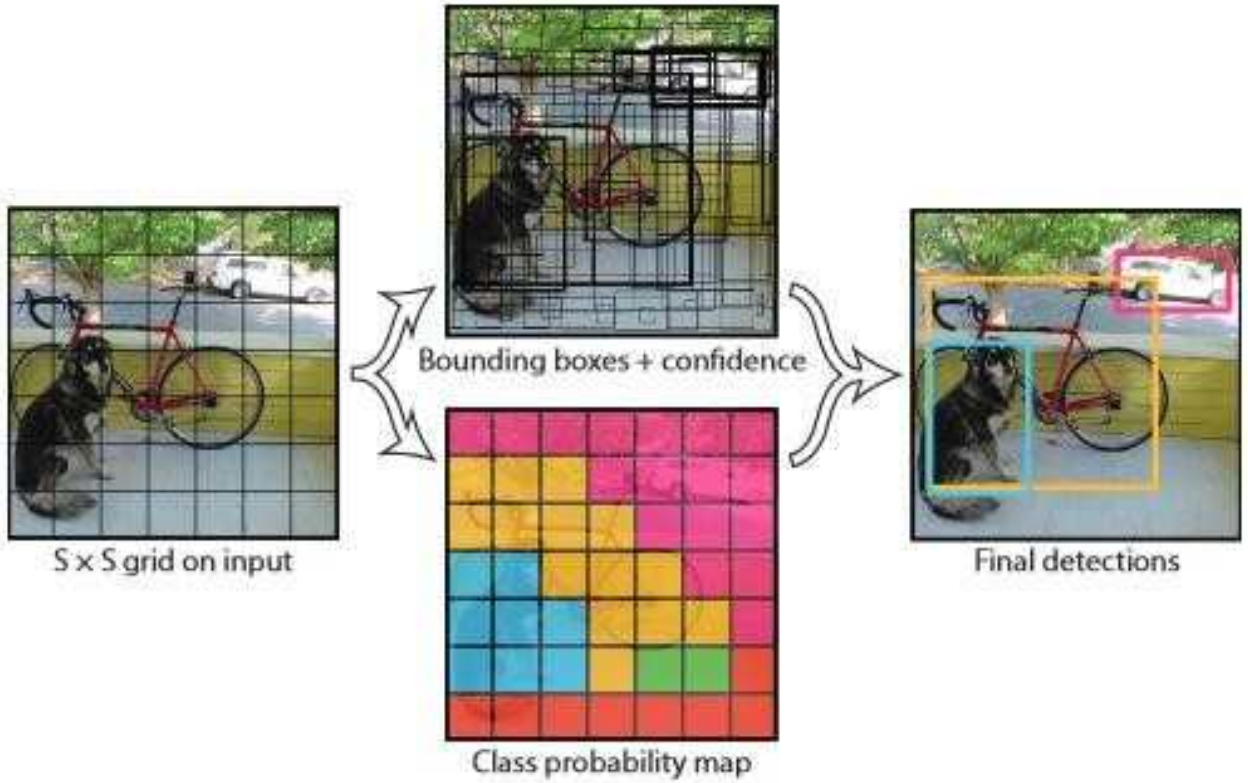
A. 2개의 Bounding Box 후보의 좌표와 해당 Box별 Confidence Score

- x, y, w, h : 정규화된 BBox의 중심 좌표와 너비/높이
- Confidence Score = 오브젝트일 확률 * IOU 값

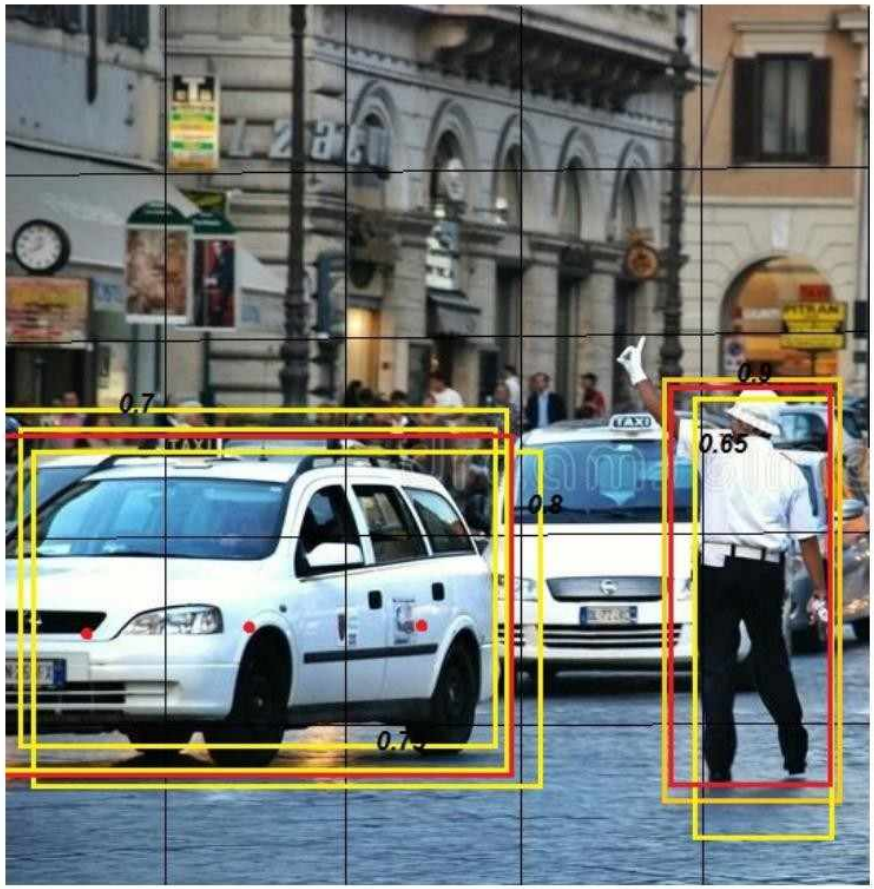
B. 클래스 확률. Pascal VOC 기준 20개 클래스의 확률

참조: www.deepsystems.io

YOLO-V1 Detection - NMS



NMS(Non Max Suppression)으로 최종 Bbox 예측



개별 Class 별 NMS 수행

1. 특정 Confidence 값 이하는 모두 제거
2. 가장 높은 Confidence값을 가진 순으로 Bbox 정렬
3. 가장 높은 Confidence를 가진 Bbox와 IOU와 겹치는 부분이 IOU Threshold 보다 큰 Bbox는 모두 제거
4. 남아 있는 Bbox에 대해 3번 Step을 반복

Object Confidence와 IOU Threshold로 Filtering 조절

YOLO-V1 이슈

Detection 시간은 빠르나 Detection 성능이 떨어짐. 특히 작은 Object에 대한 성능이 나쁨

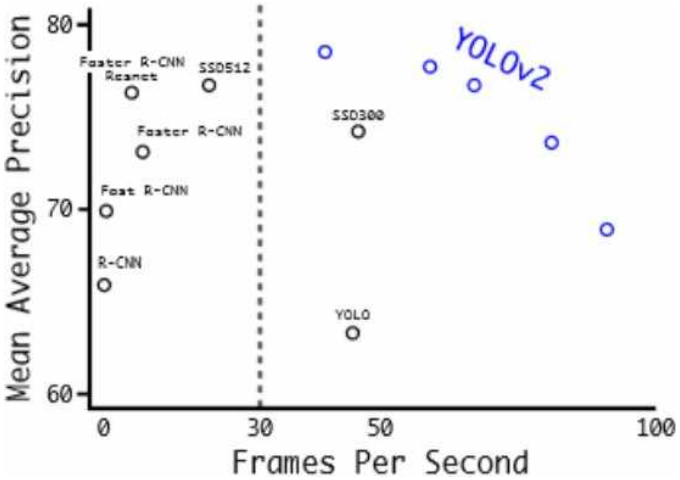
YOLO –v1, v2, v3 비교

anchor box 기반의 모델과 더 뛰어난 Backbone 구성, 다양한 성능 향상 테크닉을 적용하면서 발전됨.

항목	V1	V2	V3
원본 이미지 크기	446 X 446	416 X 416	416 X 416
Feature Extractor	Inception 변형	Darknet 19	Darknet 53
Grid당 Anchor Box 수	2개(anchor box는 아님)	5개	Output Feature Map당 3개 서로 다른 크기와 스케일로 총 9개
Anchor box 결정 방법		K-Means Clustering	K-Means Clustering
Output Feature Map 크기 (Depth 제외)	7 x 7	13 x 13	13 x13, 26 X 26, 52X52 3개의 Feature Map 사용
Feature Map Scaling 기법			FPN(Feature Pyramid Network)

YOLO-v2 Detection 시간 및 성능

PASCAL VOC 2007 Detection 시간



MS-COCO 기준 Detection 성능

		0.5:0.95	0.5	0.75	S	M	L	1	10	100	S	M	L
Fast R-CNN [5]	train	19.7	35.9	-	-	-	-	-	-	-	-	-	-
Fast R-CNN[1]	train	20.5	39.9	19.4	4.1	20.0	35.8	21.3	29.5	30.1	7.3	32.1	52.0
Faster R-CNN[15]	trainval	21.9	42.7	-	-	-	-	-	-	-	-	-	-
ION [1]	train	23.6	43.2	23.6	6.4	24.1	38.3	23.2	32.7	33.5	10.1	37.7	53.6
Faster R-CNN[10]	trainval	24.2	45.3	23.5	7.7	26.4	37.1	23.8	34.0	34.6	12.0	38.5	54.4
SSD300 [11]	trainval35k	23.2	41.2	23.4	5.3	23.2	39.6	22.5	33.2	35.3	9.6	37.6	56.5
SSD512 [11]	trainval35k	**26.8**	**46.5**	**27.8**	**9.0**	**28.9**	**41.9**	**24.8**	**37.5**	**39.8**	**14.0**	**43.5**	**59.0**
YOLOv2 [11]	trainval35k	21.6	44.0	19.2	5.0	22.4	35.5	20.7	31.6	33.3	9.8	36.5	54.4

Yolo v2 의 특징

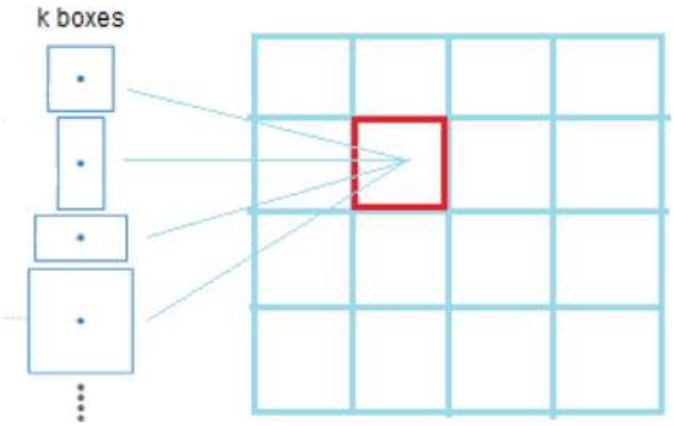
- Batch Normalization
- High Resolution Classifier : 네트웍의 Classifier 단을 보다 높은 resolution(448x448)로 fine tuning
- 13 x 13 feature map 기반에서 개별 Grid cell 별 5개의 Anchor box에서 Object Detection
  - anchor box의 크기와 ratio는 K-Means Clustering으로 설정.
- 예측 Bbox의 x,y 좌표가 중심 Cell 내에서 벗어나지 않도록 Direct Location Prediction 적용
- Darknet-19 Classification 모델 채택
- Classification layer를 fully Connected layer에서 Fully Convolution 으로 변경하고 서로 다른 크기의 image들로 네트웍 학습

Yolo v2 Anchor Box로 1 CeII에서 여러 개 Object Detection

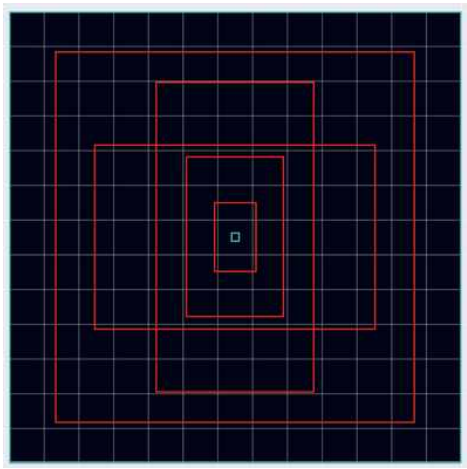
SSD와 마찬가지로 1개의 CeII에서 여러 개의 Anchor를 통해 개별 CeII에서 여러 개 Object Detection 가능

K-Means Clustering 을 통해 데이터 세트의 이미지 크기와 Shape Ratio 따른 5개의 군집화 분류를 하여 Anchor Box를 계산

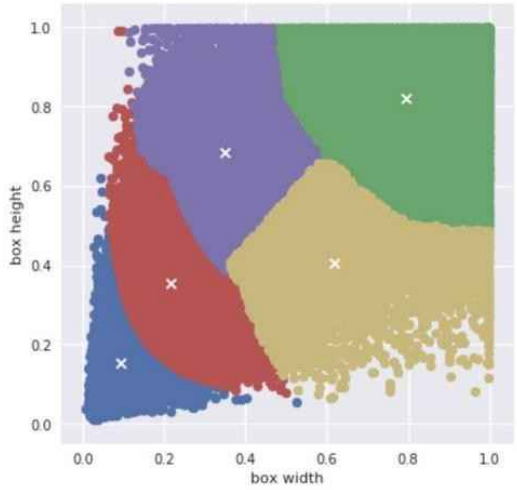
Convolutional With Anchor Boxes



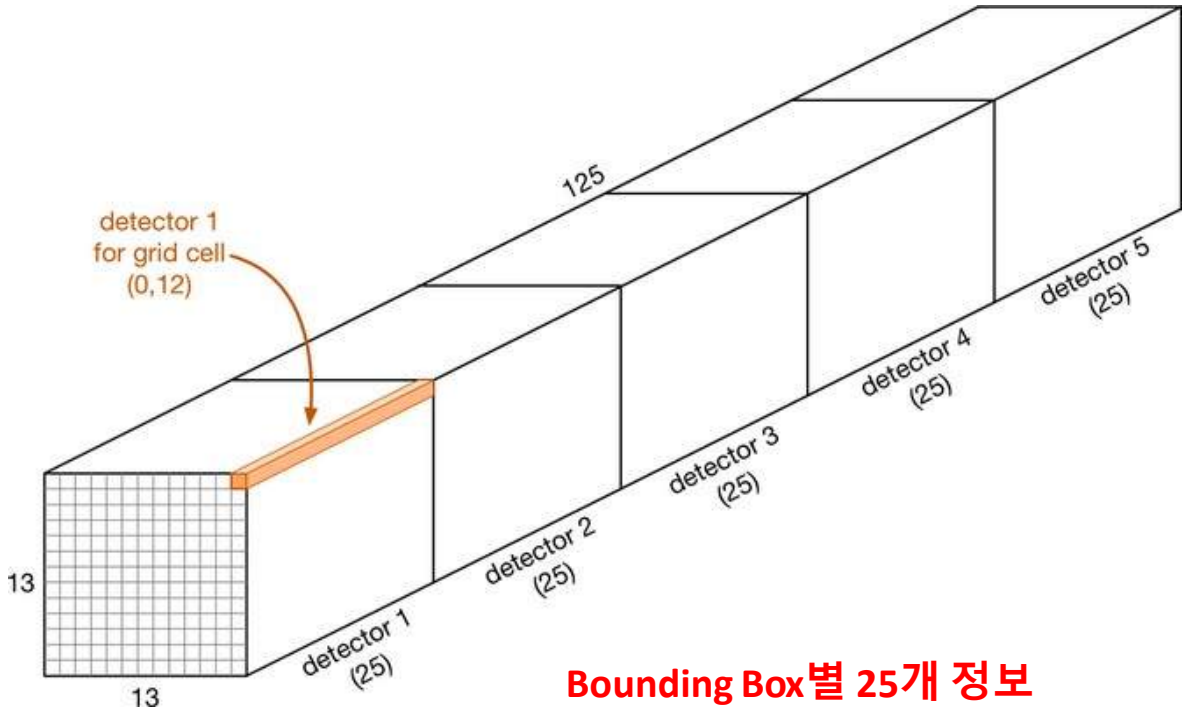
5개 Anchor Box

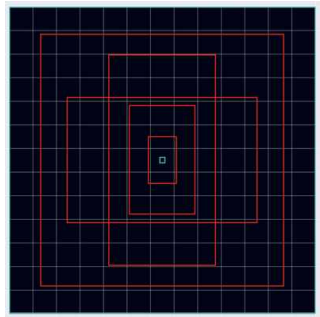


K-Means Clustering

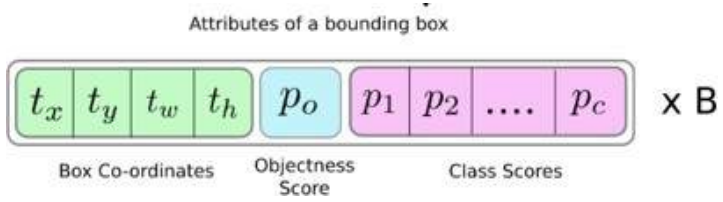


YOLO v2 Output Feature Map



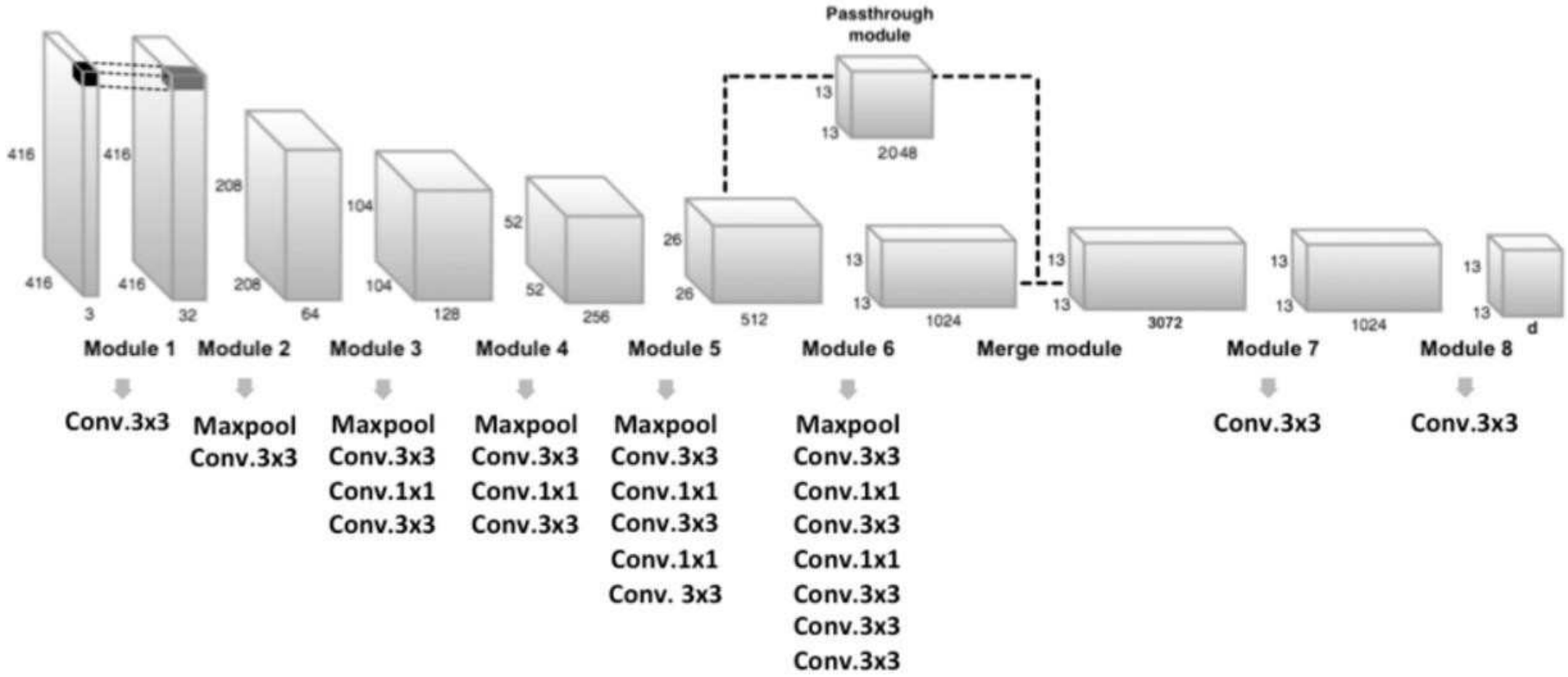


Bounding Box별 25개 정보



Passthrough module을 통한 fine grained feature

좀 더 작은 오브젝트를 Detect 하기 위해서 26x26x512 feature map의 특징을 유지한 채 13x13x2048 로 reshape한 뒤 13x13x1024에 추가하여 feature map 생성.



Multi-Scale Training

Classification layer가 Convolution layer로 생성하여 동적으로 입력 이미지 크기 변경 가능

학습 시 10 회 배치 시 마다 입력 이미지 크기를 모델에서 320 부터 608까지 동적으로 변경(32의 배수로 설정)

Detection Frameworks	Train	mAP	FPS
Fast R-CNN [5]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[15]	2007+2012	73.2	7
Faster R-CNN ResNet[6]	2007+2012	76.4	5
YOLO [14]	2007+2012	63.4	45
SSD300 [11]	2007+2012	74.3	46
SSD500 [11]	2007+2012	76.8	19
YOLOv2 288 × 288	2007+2012	69.0	91
YOLOv2 352 × 352	2007+2012	73.7	81
YOLOv2 416 × 416	2007+2012	76.8	67
YOLOv2 480 × 480	2007+2012	77.8	59
YOLOv2 544 × 544	2007+2012	**78.6**	40

Darknet 19 Backbone

Darknet 19

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

VGG-16: 30.69 BFLOPS, Top 5 Accuracy: 90%

Yolo v1: 8.52 BFlops , Top 5 Accuracy: 88%

Darknet-19: 19: 5.58 BFLOPS, Top 5 Accuracy: 91.2%

Classification layer에 Fully Connected layer를 제거하고 Conv layer를 적용.

YOLO-v2 성능 향상

	YOLO								YOLOv2
batch norm?		✓	✓	✓	✓	✓	✓	✓	✓
hi-res classifier?			✓	✓	✓	✓	✓	✓	✓
convolutional?				✓	✓	✓	✓	✓	✓
anchor boxes?				✓	✓				
new network?					✓	✓	✓	✓	✓
dimension priors?						✓	✓	✓	✓
location prediction?						✓	✓	✓	✓
passthrough?							✓	✓	✓
multi-scale?								✓	✓
hi-res detector?									✓
VOC2007 mAP	63.4	65.8	69.5	69.2	69.6	74.4	75.4	76.8	**78.6**

Yolo v3 의 특징

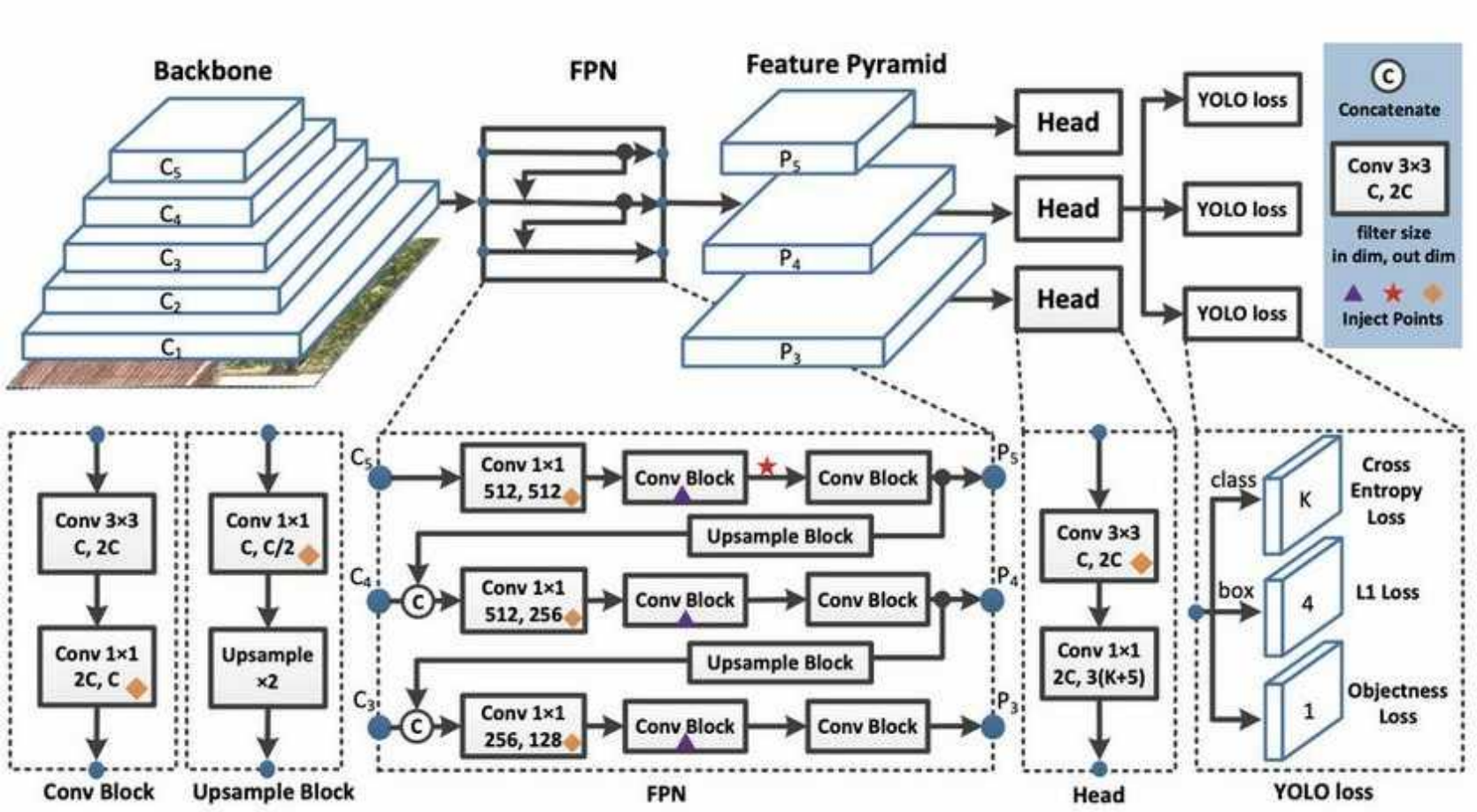
- Feature Pyramid Network 유사한 기법을 적용하여 3개의 Feature Map Output에서 각각 3개의 서로 다른 크기와 scale을 가진 anchor box 로 Detection
- Backbone 성능 향상 - Darknet-53
- Multi Labels 예측: Softmax가 아닌 Sigmoid 기반의 logistic classifier로 개별 Object의 Multi labels 예측

YOLO –v1, v2, v3 비교

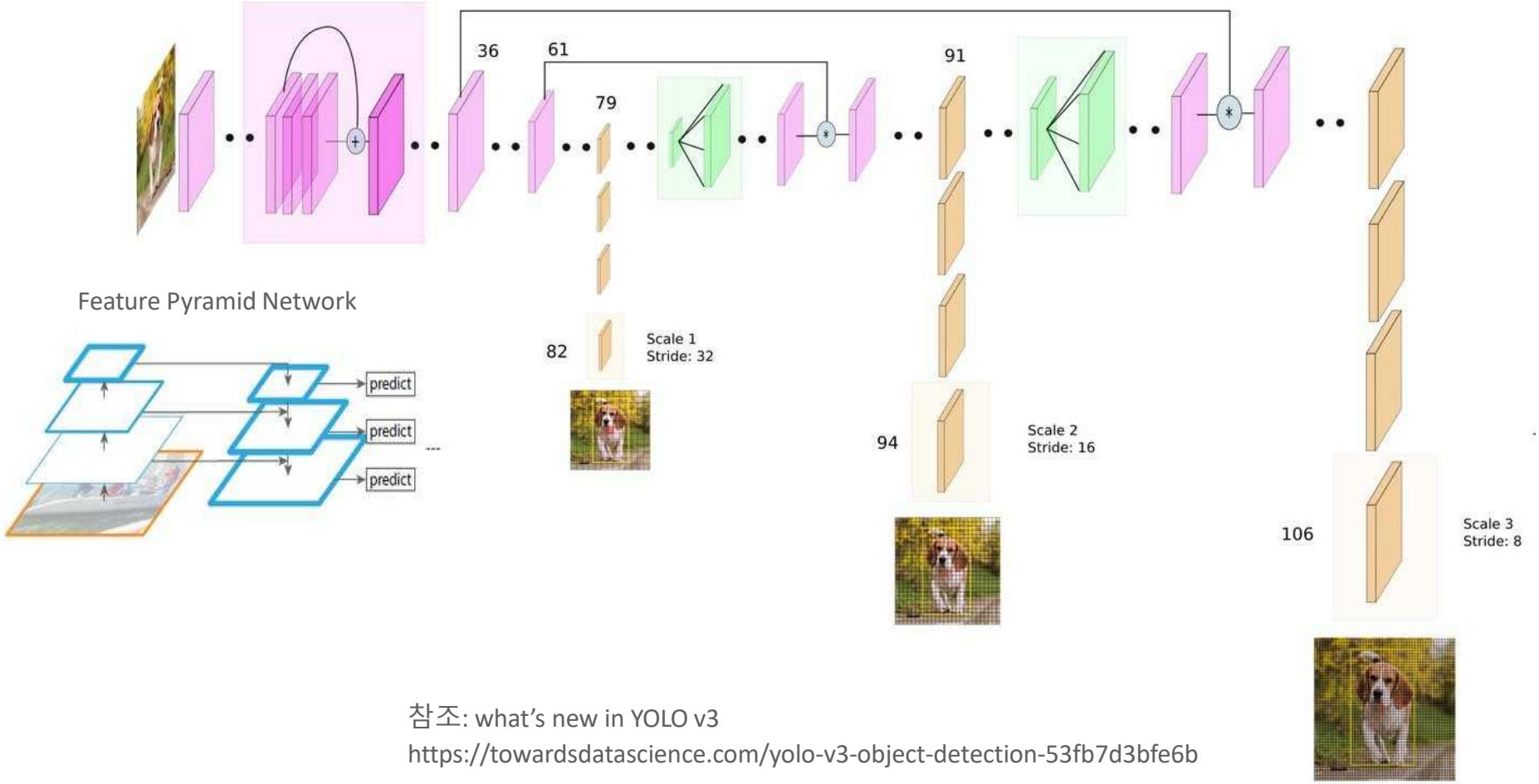
anchor box 기반의 모델과 더 뛰어난 Backbone 구성, 다양한 성능 향상 테크닉을 적용하면서 발전됨.

항목	V1	V2	V3
원본 이미지 크기	446 X 446	416 X 416	416 X 416
Feature Extractor	Inception 변형	Darknet 19	Darknet 53
Grid당 Anchor Box 수	2개(anchor box는 아님)	5개	Output Feature Map당 3개 서로 다른 크기와 스케일로 총 9개
Anchor box 결정 방법		K-Means Clustering	K-Means Clustering
Output Feature Map 크기 (Depth 제외)	7 x 7	13 x 13	13 x13, 26 X 26, 52X52 3개의 Feature Map 사용
Feature Map Scaling 기법			FPN(Feature Pyramid Network)

YOLO V3 모델 아키텍처

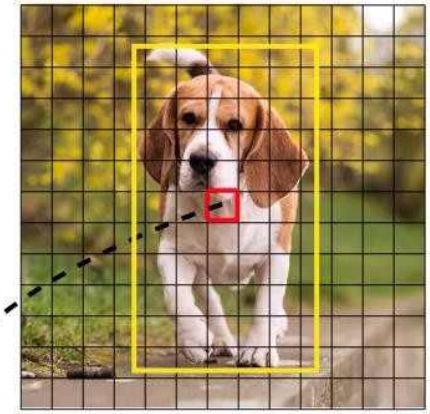


YOLO v3 Network 구조



참조: what's new in YOLO v3
https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b

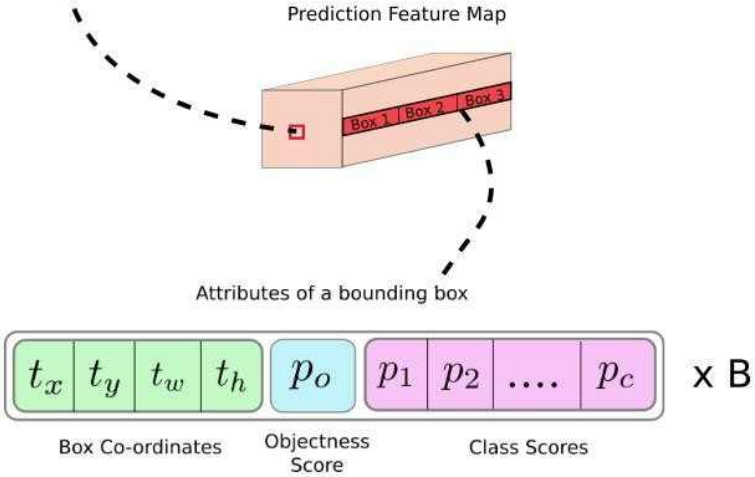
YOLO v3 Output Feature Map



13 x 13

26 x 26

52 x 52



https://blog.paperspace.com/how-to-implement-a-yolo-object-detector-in-pytorch/

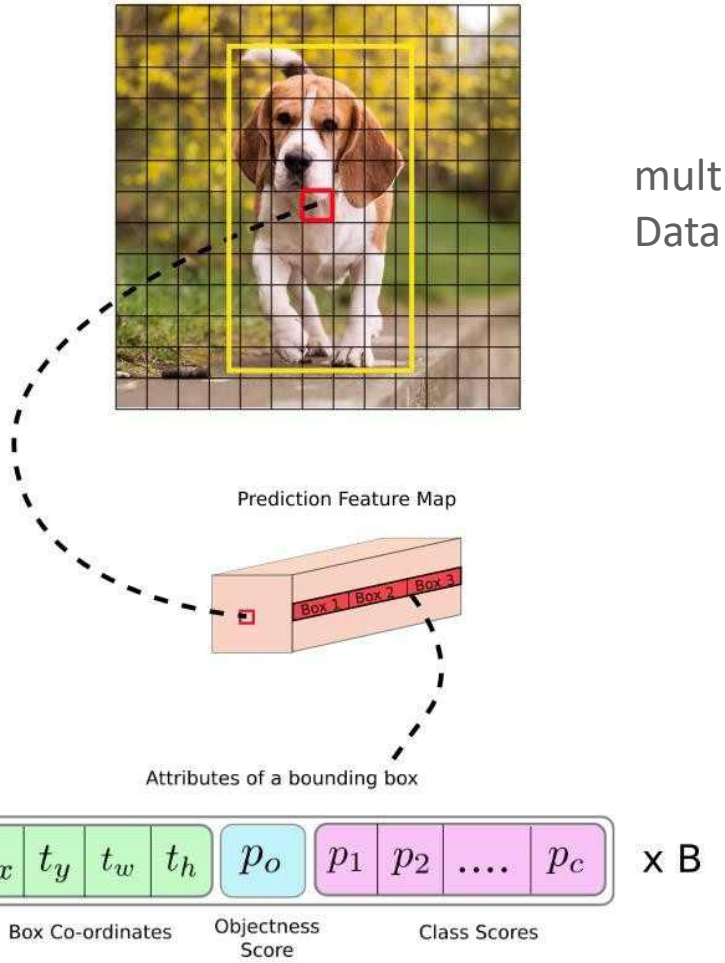
Darknet-53 특성

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

	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			

Training

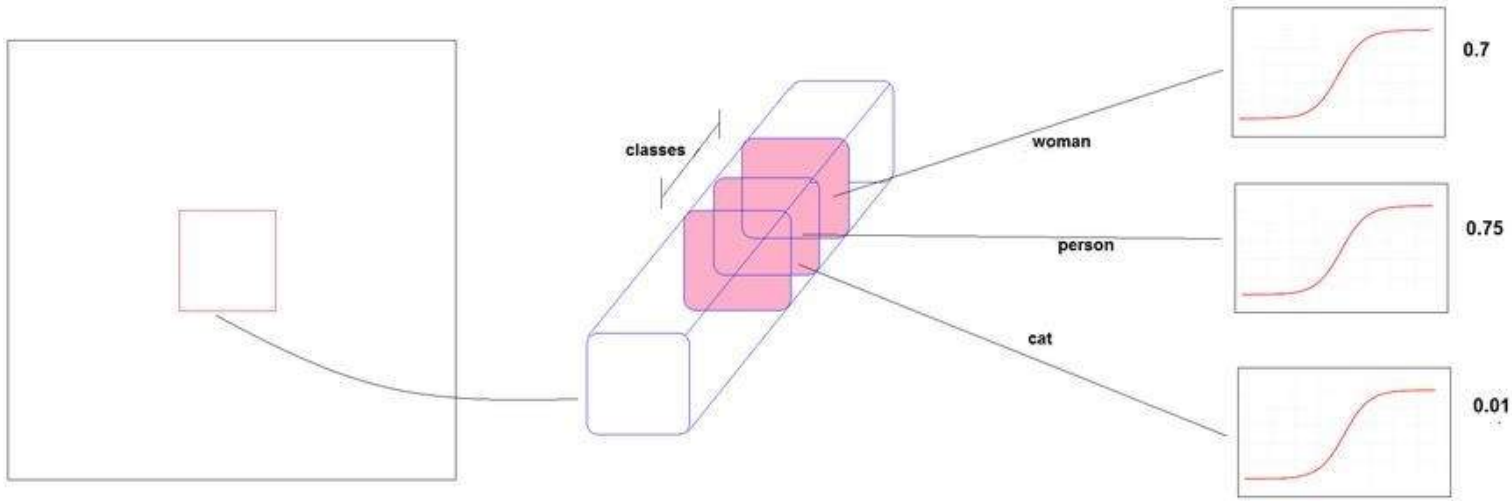
```
<annotation>
    <folder>VOC2007</folder>
    <filename>003585.jpg</filename>
    <size>
        <width>333</width>
        <height>500</height>
        <depth>3</depth>
    </size>
    <object>
        <name>person</name>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>138</xmin>
            <ymin>183</ymin>
            <xmax>259</xmax>
            <ymax>411</ymax>
        </bndbox>
    </object>
    <object>
        <name>motorbike</name>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>89</xmin>
            <ymin>244</ymin>
            <xmax>291</xmax>
            <ymax>425</ymax>
        </bndbox>
    </object>
```



multi-scale training,
Data augmentation

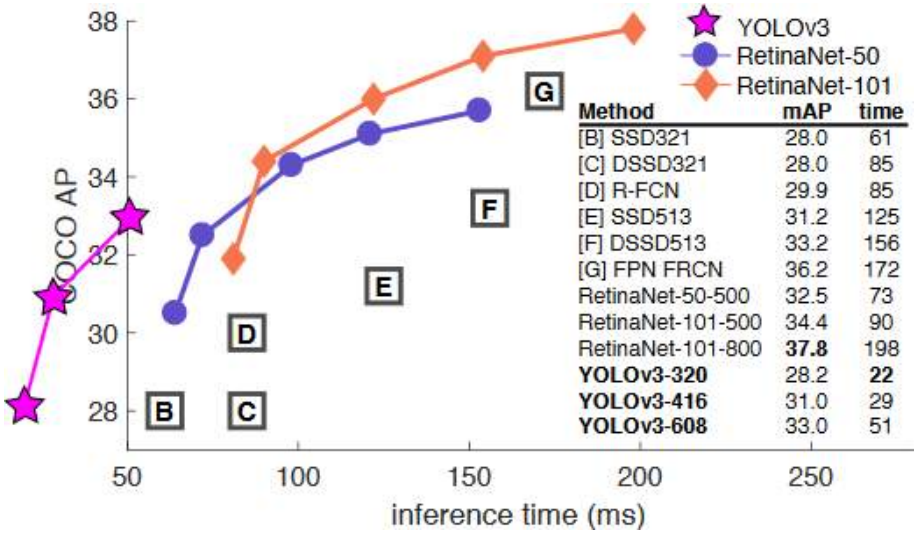
Multi Labels 예측

여러 개의 독립적인 *Logistic Classifier* 사용



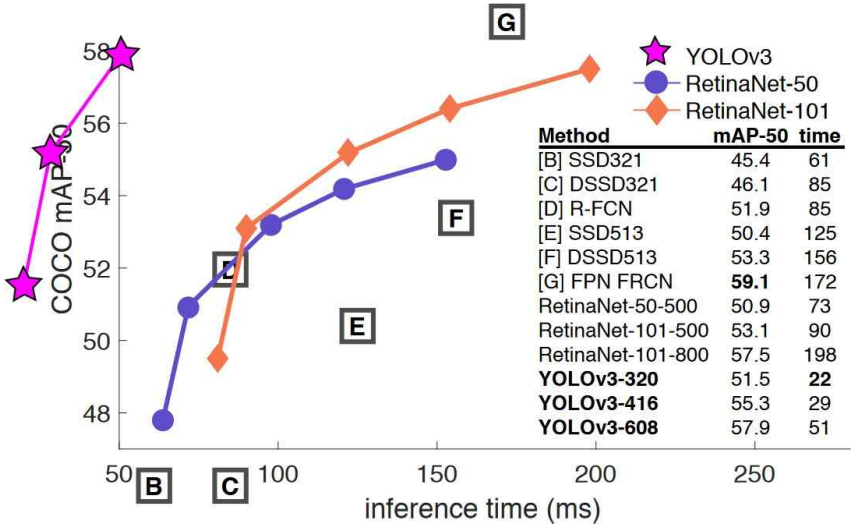
YOLO v3 성능 비교

COCO(IOU 0.5 ~ 0.95 기준)



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

COCO(IOU 0.5 기준)



Method	mAP-50	time
[B] SSD321	45.4	61
[C] DSSD321	46.1	85
[D] R-FCN	51.9	85
[E] SSD513	50.4	125
[F] DSSD513	53.3	156
[G] FPN FRCN	**59.1**	172
RetinaNet-50-500	50.9	73
RetinaNet-101-500	53.1	90
RetinaNet-101-800	57.5	198
YOLOv3-320	51.5	**22**
YOLOv3-416	55.3	29
YOLOv3-608	57.9	51