YOLO

(**Y**ou **O**nly **L**ook **O**nce)

Object Detection Review

Object Detection - YOLO

CNN 기반 사전 학습 모델	
Model	Key Feature
LeNet	CNN 초기 모델 심층 학습 모델의 기초 Convolutiona Layer + Pooling Layer
AlexNet	ReLU 활성화 함수 데이터 증강(Data Augmentation) Dropout 다중 GPU 활용
VGG	깊은 Network 구조 (512) 3×3 크기 필터 (작은 필터) -> 파라메터 수 감소
InceptionNet	Inception module Bottleneck Auxiliary classfier
ResNet	residual module skip connection Top5 오류율 3.57%
MobileNet	Depthwise Separable Convolution
DenseNet	Dense Connectivity (밀집 연결) Feature Reuse (특징 재사용)
EfficientNet	compound scaling method

Object Detection - YOLO

사전 학습 모델 *vs* 객체 인식 모델

(CNN 기반) 사전 학습 모델

- AlexNet, VGG, ResNet, InceptionNet, MobileNet
- 주로 이미지 분류 작업에 사용

  이미지로부터 다양한 특징을 학습, 새로운 이미지가 주어졌을 때 그 이미지가 어떤 카테고리에 속하는지 분류

1. 이미지 분류: 개별 이미지를 분석하고, 이미지에 표시된 객체의 종류 식별
2. 특징 추출: 복잡한 이미지로부터 유용한 정보를 추출, 다른 비전 작업에 사용될 수 있는 강력한 특징 벡터
3. 전이 학습: 사전 학습된 모델을 가져와서 관련 분야나 다른 종류의 비전 작업에 적용하는 기반으로 사용

Object Detection - YOLO

사전 학습 모델 *VS* 객체 인식 모델

객체 인식 모델

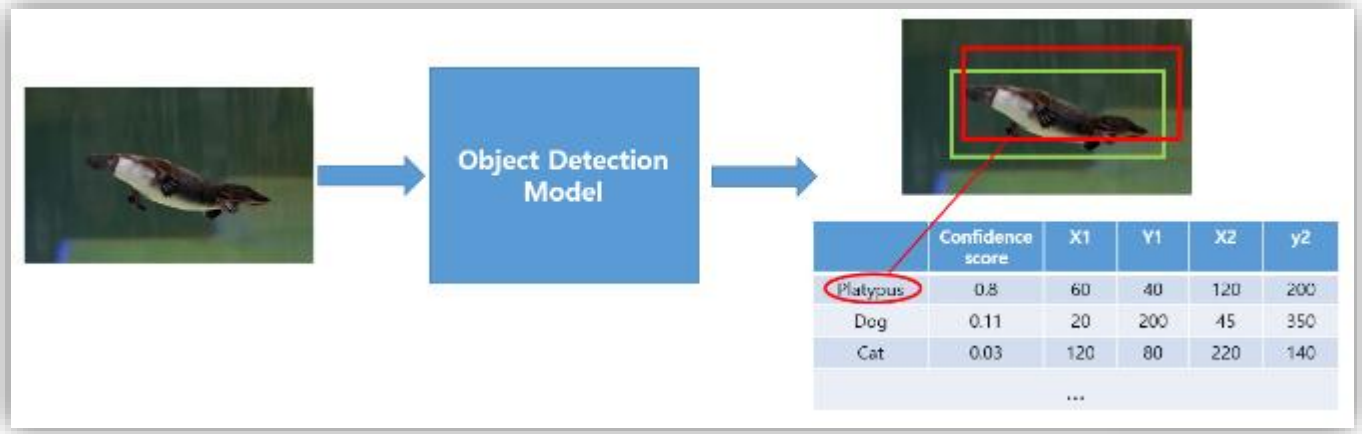
- SSD, YOLO, Faster R-CNN
- 이미지 내에서 하나 이상의 **객체를 식별**

1. 객체 감지: 이미지 내의 다양한 객체들을 식별하고 각 객체의 위치를 바운딩 박스로 표시
2. 인스턴스 분할: 객체 감지와 유사, 각 객체의 정확한 픽셀 단위의 윤곽을 식별.
3. 객체 추적: 비디오에서 객체들의 움직임을 추적하고, 시간에 따른 위치 변화를 식별
4. 자율 주행 차량: 도로 상의 차량, 보행자, 신호등 등을 식별하고 위치
5. 보안 감시: CCTV 영상 분석을 통해 사람이나 차량 등의 움직임 감지

Object Detection - YOLO

사전 학습 모델 *vs* 객체 인식 모델

객체 인식 모델




```
[7.6634e+01, 2.5491e+02, 9.1255e+01, 2.8626e+02],
[1.3929e+02, 2.3328e+02, 1.5933e+02, 2.5312e+02]]), 'scores': tensor([0.999
0.0272, 0.0270, 0.0267, 0.0264, 0.0262, 0.0258, 0.0249, 0.0244, 0.0234,
0.0229, 0.0221, 0.0220, 0.0220, 0.0215, 0.0213, 0.0211, 0.0199, 0.0190,
0.0190, 0.0185, 0.0176, 0.0175, 0.0173, 0.0173, 0.0172, 0.0172, 0.0171,
0.0167, 0.0164, 0.0162, 0.0161, 0.0159, 0.0158, 0.0155, 0.0150, 0.0147,
0.0147, 0.0146, 0.0140, 0.0140, 0.0140, 0.0137, 0.0132, 0.0131, 0.0131,
0.0131, 0.0129, 0.0128, 0.0128, 0.0127, 0.0127, 0.0125, 0.0124, 0.0123,
0.0123, 0.0122, 0.0121, 0.0121, 0.0120, 0.0119, 0.0118, 0.0117, 0.0116,
0.0116, 0.0115, 0.0114, 0.0113, 0.0109, 0.0109, 0.0109, 0.0108, 0.0108,
0.0108, 0.0108, 0.0108, 0.0107, 0.0105, 0.0105, 0.0104, 0.0104, 0.0104,
0.0104, 0.0103, 0.0102, 0.0102, 0.0102, 0.0102, 0.0101, 0.0101]), 'labels':
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, 35,  1,
35,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, 85, 85, 35,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, 35,  1,
 1,  1,  1,  1,  1,  1,  1,  1, 85,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1, 35,  1,  1, 85,  1, 35])}]
```

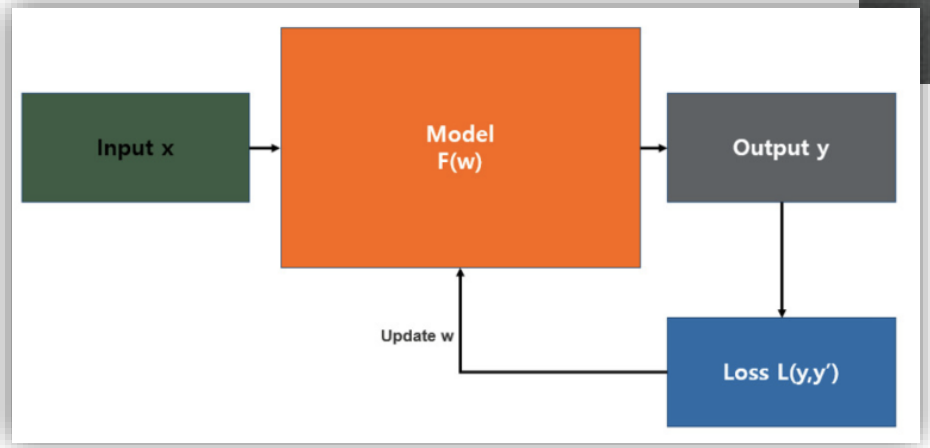
Object Detection - YOLO

Key Feature

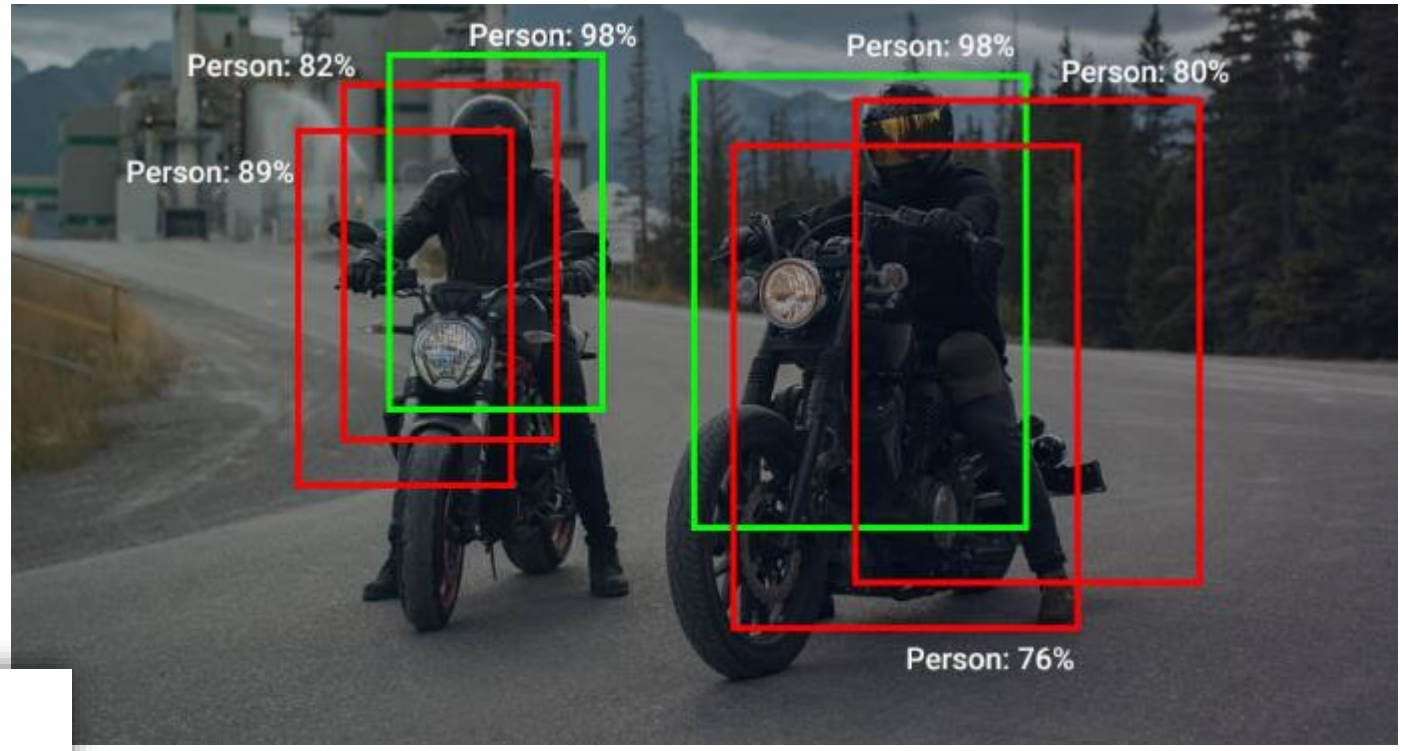
IoU(Intersection over Union)
:두 상자가 얼마나 많이 겹치는지를 나타내는 지표

Positive(객체 존재)
or
Negative 판단

$$IOU = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$



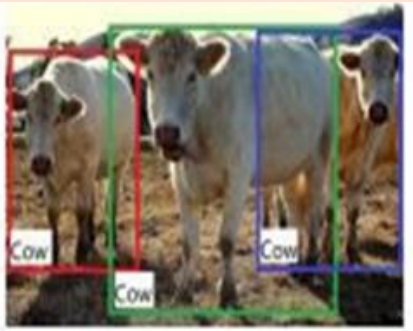
Non-Maximum Suppression(NMS)



선택된 BB(신뢰도가 높은)와 IoU가 임계값(Threshold) 이상인 다른 BB Suppression
- **중복된 BB 억제**
- **가장 신뢰도**(Confidence Score)**가 높은 BB만 남김**

GTBB
Ground Truth Bounding Box

Object Detection - YOLO



Object detection

객체 탐지(object Detection) – 훈련 Process

1. 학습 데이터 준비:
- 객체의 위치를 나타내는 **경계 상자**와 각 객체의 **클래스 레이블**이 포함 된 이미지 준비
- labeled data 데이터를 통해 객체의 위치와 클래스를 예측하는 방법 학습

2. 특징 추출:
- (ex, SSD) 특징 추출을 위해 **합성곱 신경망(Convolutional Neural Network, CNN) 사용**
- 이미지의 원시 픽셀 데이터를 받아 여러 계층을 통과시키면서 필요한 특징 학습, 추출

3. 경계 상자 예측:
- 여러 스케일에서 다양한 비율(aspect ratio)의 **경계 상자** 예측
- 각 피처 맵(feature map)의 각 위치에 대해 고정된 수의 상자(**앵커 박스**)를 사용하여 경계 상자 계산
- 이 앵커 박스를 기준으로 실제 객체의 위치와 가장 가까운 경계 상자를 조정하는 방법 학습

Object Detection - YOLO

객체 탐지(object Detection) – 훈련 Process

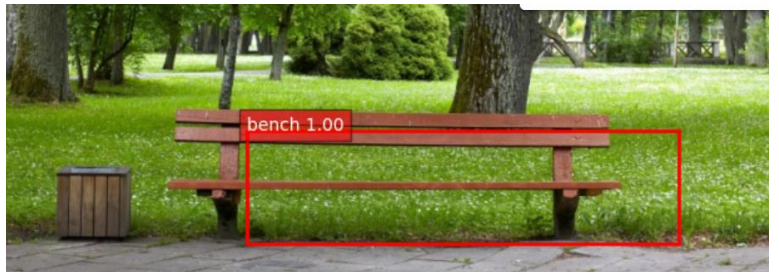
4. 클래스 예측:
- 각 경계 상자에 대해 모델은 **해당 상자 내에 객체가 포함될 가능성**과 **각 클래스에 속할 확률** 예측
- softmax 함수 사용, 각 클래스에 대한 확률 출력

5. Non-maximum Suppression(NMS):
- 모델이 여러 개의 (중복되는, 겹치는) 경계 상자를 생성할 수 있기 때문에, NMS를 통해 중복된 경계 상자 제거 -> 가장 높은 확률을 가진 경계 상자만을 남김
- 최종 객체 탐지 결과의 정확도 향상

입력 이미지 내의 각 객체의 위치와 클래스의 예측 수행

Object Detection - YOLO

객체 탐지(object Detection) – 예측 Process

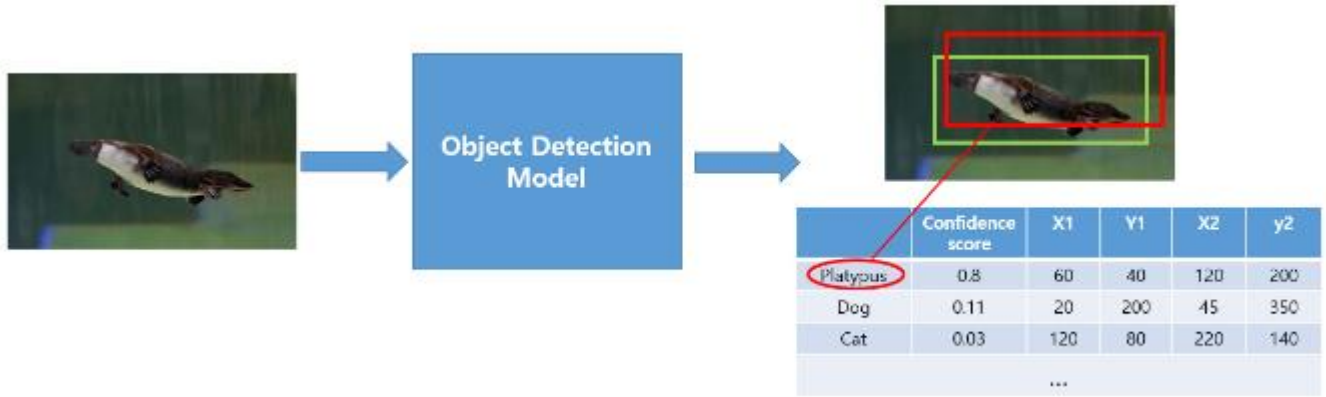


1. **이미지 전처리**:
   - 모델의 훈련 데이터셋과 동일한 형식으로 전처리
   - 크기 조정, 정규화, 색상 조정 등
2. **특징 추출**:
   - 합성곱 신경망 등 모델 內 layer를 통과하며 이미지의 다양한 특징 추출
   - 추출된 특징을 이용해 객체 인식, 위치 예측에 사용
3. **경계 상자 및 클래스 예측**:
   - 추출된 특징을 통해 이미지 전체에 걸쳐 다양한 위치와 크기의 경계 상자 예측
   - 각 경계 상자는 특정 클래스에 속할 확률과 함께 계산
4. **Non-Maximum Suppression(NMS)**:
   - 생성한 다수의 경계 상자 중 중복되거나 겹치는 상자 제거
   - 가장 높은 신뢰도를 가진 경계 상자 선택 -> 각 객체에 대한 하나의 경계 상자 결정
5. **결과 출력**:
   객체의 클래스 예측
   객체의 위치와 해당 객체의 클래스를 나타내는 결과 출력

Object Detection - YOLO

객체 탐지(object Detection) – 예측 Process

5. 결과 출력:
객체의 클래스 예측
객체의 위치와 해당 객체의 클래스를 나타내는 결과 출력
- **Bounding box의 좌표값**과 **Confidence score** 출력



- **Bounding box**(경계상자): 이미지 내 객체의 위치를 사각형 형태로 예측한 결과,
  사각형의 좌상단 좌표(x1, y1)과 우하단 좌표(x2, y2) 좌표 (좌표 형식은 모델마다 상이)
- **Confidence score:** 예측한 box 내 존재하는 객체의 클래스에 대한 예측 확률의 최대값
  ex, Confidence score 0.8 = bounding box 내의 객체가 오리너구리일 확률 80% (모델의 확신)

Object Detection - YOLO

Single shot detection(SSD) – Key Word

- **Single Shot Detector**

  : 'single shot' = 객체 위치 탐지와 분류를 동시에 수행

- **다양한 스케일의 피처 맵**

  : multi-scale approach - 다양한 규모(다양한 크기 및 해상도)의 특징 맵을 사용

  : 입력 이미지로부터 여러 레벨의 피처 맵 생성, 각 레벨에서 객체 탐지 수행

  : 각 피처 맵에서 서로 다른 크기의 디폴트 박스 사용함 -> 다양한 크기의 객체 감지

- **속도와 정확도**

  : 고속 처리 + 높은 정확도

  : 실시간 시스템(자율 주행 자동차 등) 활용

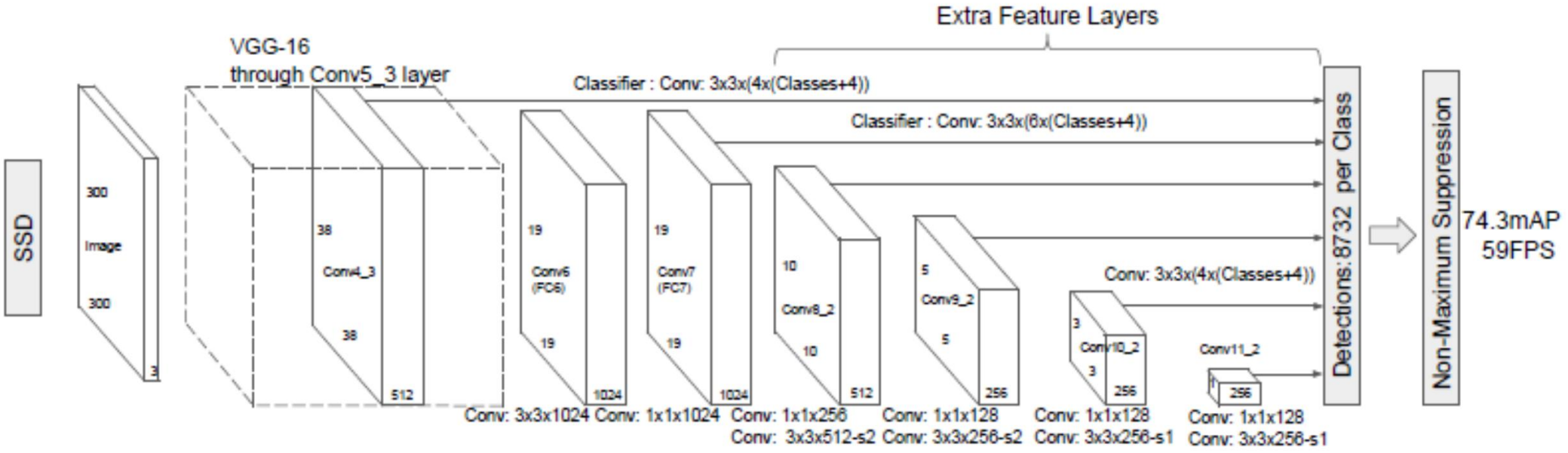
Object Detection - YOLO

- Single stage detector
- Backbone: VGG-Net
- 6개의 feature map 사용
  - Multi-scale detection
- Anchor box 사용
  - 각 셀마다 4개 또는 6개
  - 각 AB에 대해 4개의 좌표값과 21개의 클래스 확률값 출력 (백그라운드 클래스 포함)
- Hard negative sampling
  - Pos boxes : Neg boxes = 1:3
- 비용함수
  - 교차 엔트로피 + Smooth L1 loss
- Non-maximum suppression

Object Detection - YOLO

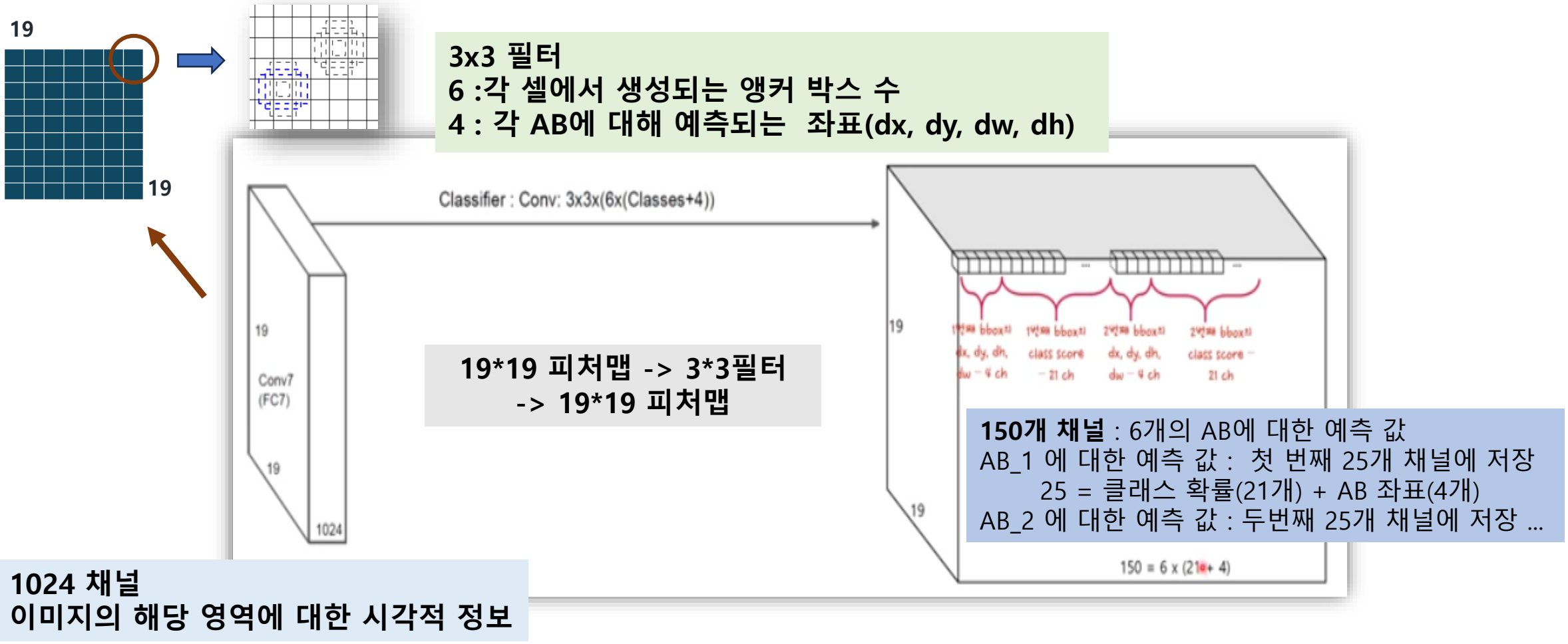
Single shot detection(SSD) - Paper

https://arxiv.org/abs/1512.02325



Object Detection - YOLO

Single shot detection(SSD) – 출력



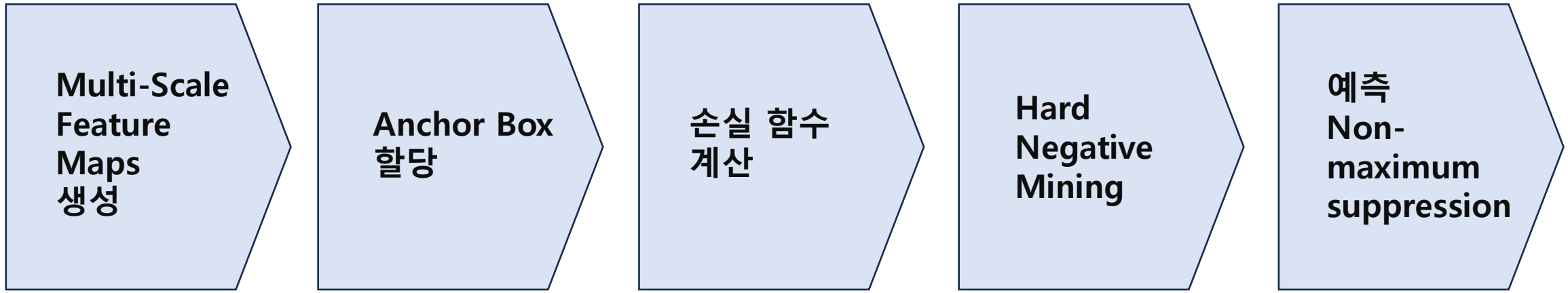
Object Detection - YOLO

Single shot detection(SSD) – 학습 프로세스

1. 다양한 크기의 **피처맵** 생성,
   -> 다양한 크기의 객체 탐지

3. 위치 손실 +
   클래스 신뢰도 **손실 계산**
   -> 모델 평가

5. 중복되는 객체 탐지 결과 중
   가장 신뢰도가 높은 하나를 선택
   최종 탐지 결과를 정제

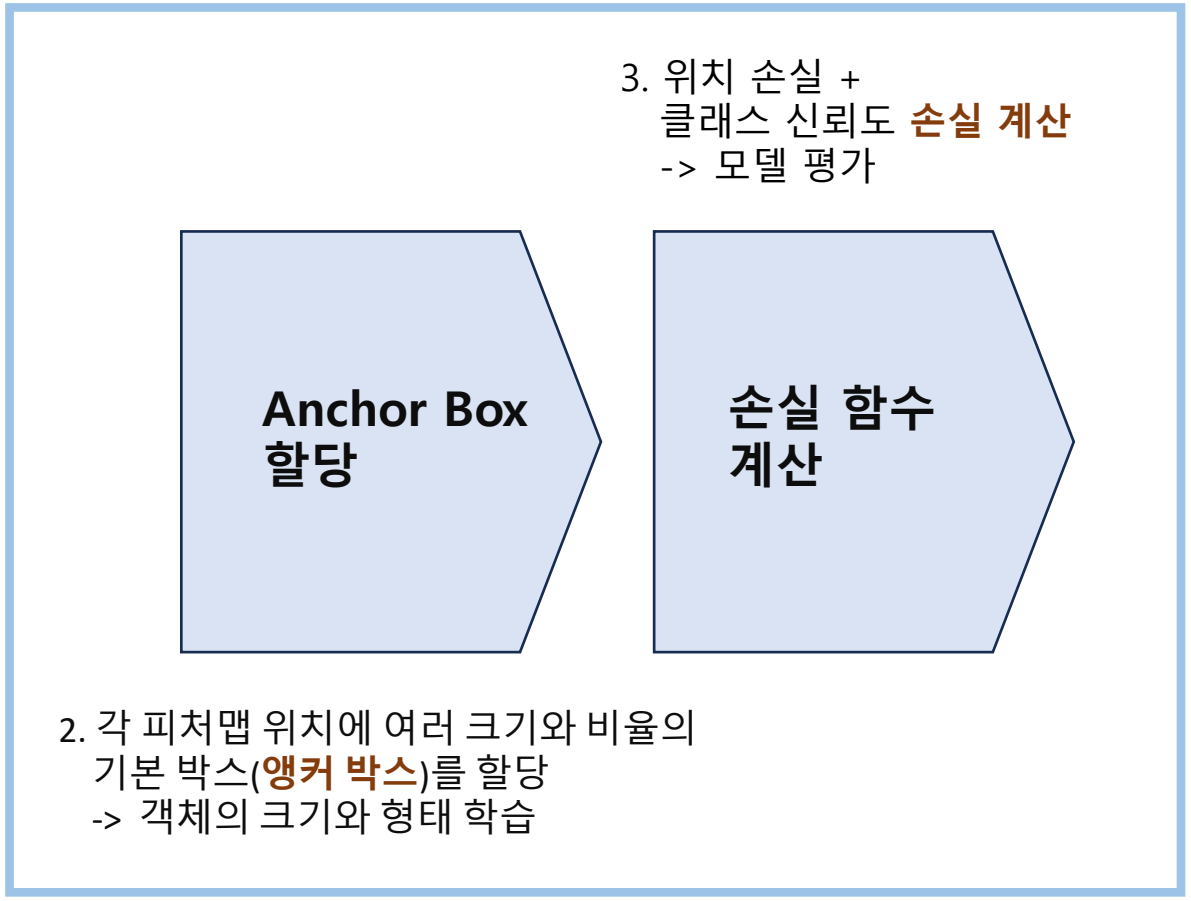


2. 각 피처맵 위치에 여러 크기와 비율의
   기본 박스(**앵커 박스**)를 할당
   -> 객체의 크기와 형태 학습

4. 배경으로 잘못 분류된 예측 중에서
   손실이 가장 큰 예측을 **선별적으로 학습**
   -> 모델의 성능 향상

Object Detection - YOLO

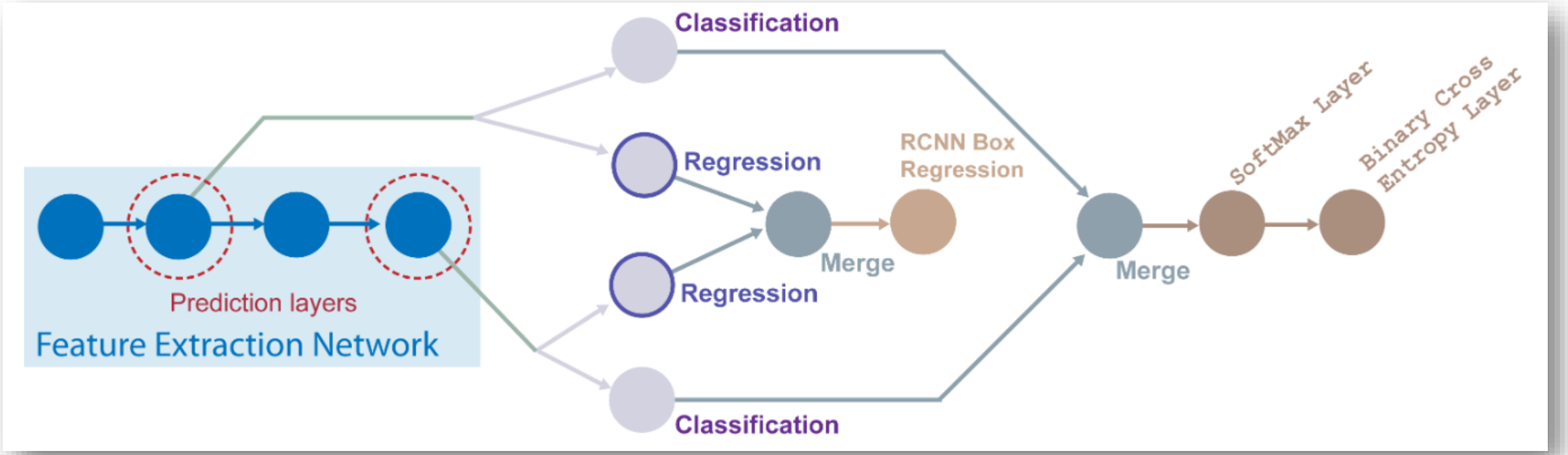
Single shot detection(SSD) – 학습 프로세스



Q. AnchorBox = 관측치 비용함수 ?

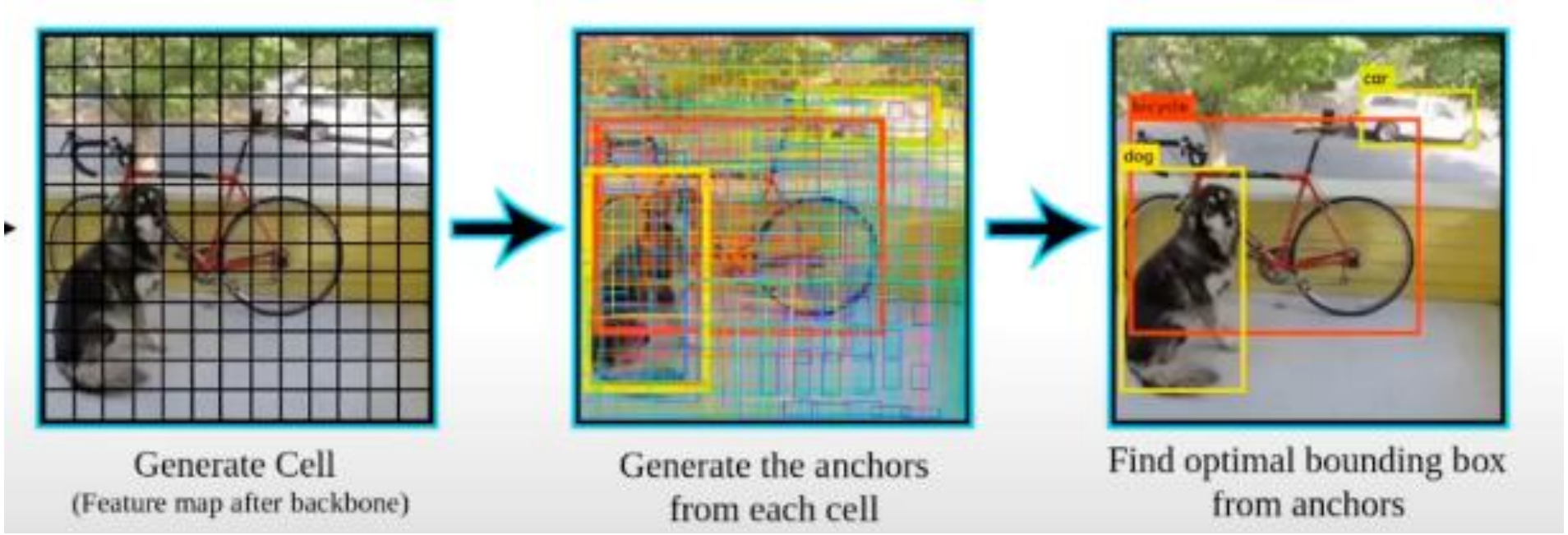
Object Detection - YOLO

Single shot detection(SSD) – 학습 프로세스



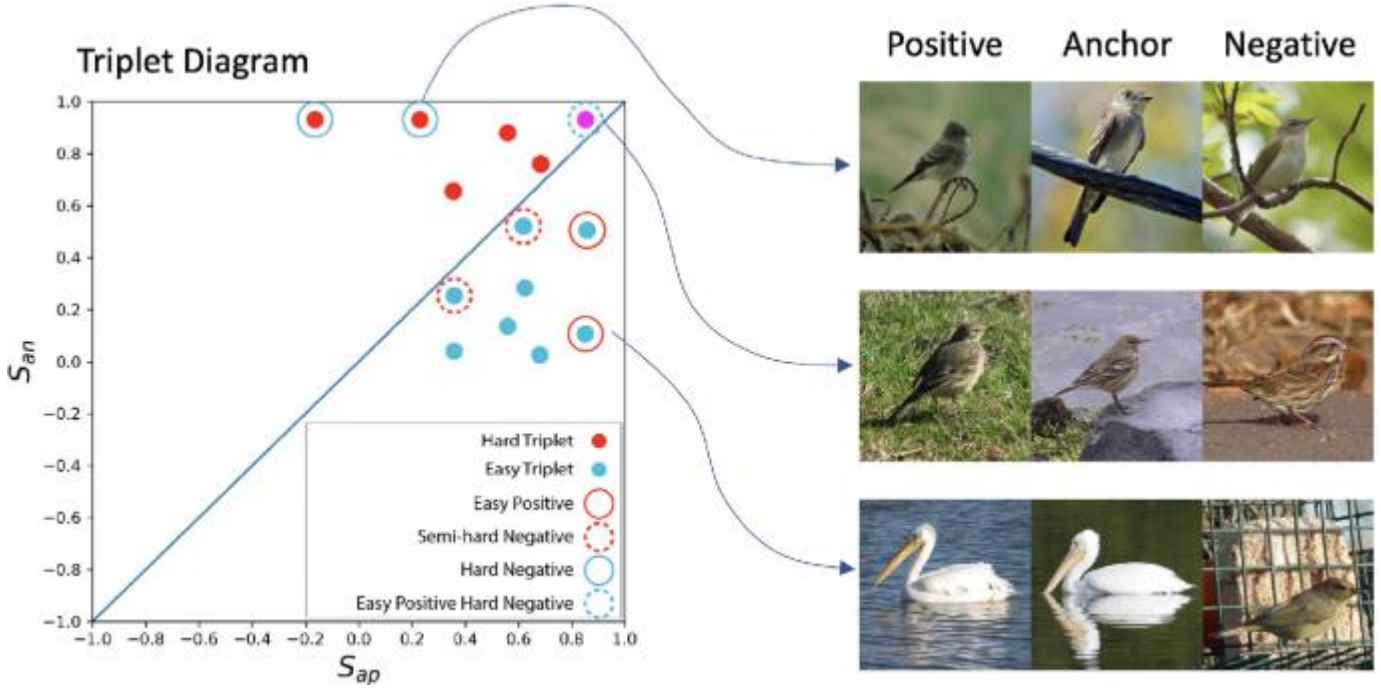
Object Detection - YOLO

Single shot detection(SSD) – cell, Anchors



Object Detection - YOLO

Hard Negative Sampling

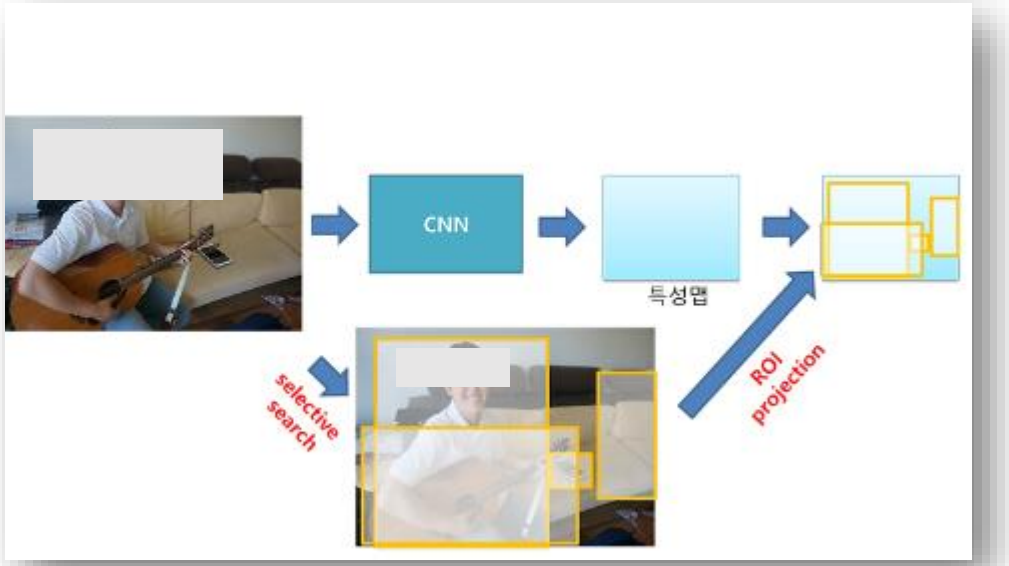


Object Detection - YOLO

R_CNN – Key Feature

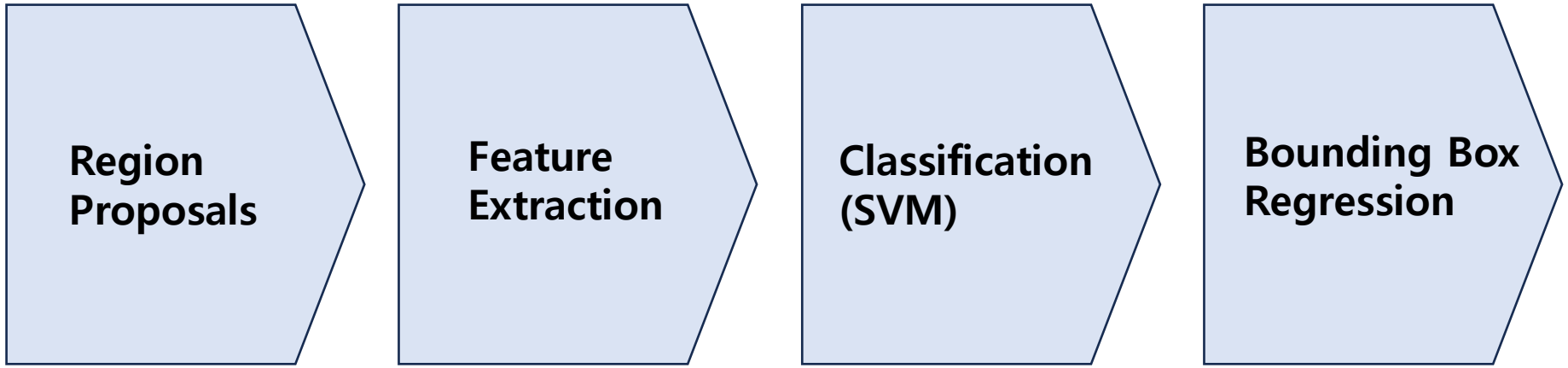
- **ROI**
  : 객체가 존재할 가능성이 있는 지역

- **Selective Search**
  : 유사한 segments를 합쳐서 물체가 있을만한 지역을 추천

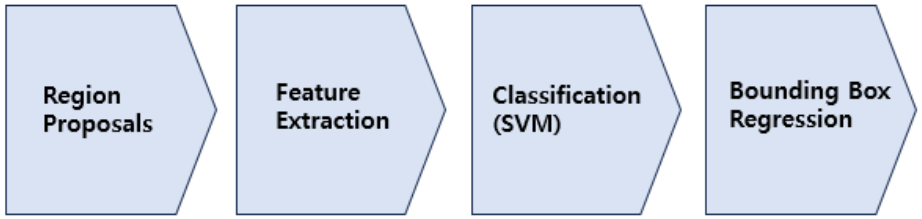


Object Detection - YOLO

R-CNN – 학습 프로세스



Object Detection - YOLO



R-CNN – 학습 프로세스

1.Region Proposals
: 이미지 내에서 잠재적인 객체를 포함할 것으로 예상되는 영역을 식별 <- "Selective Search"
: 2000 여개의 Region Proposals(후보 영역) 생성
(후보 영역: 다양한 크기와 비율을 가지며, 이미지 내의 다양한 위치에서 추출)
: 각 제안된 영역 신경망의 입력으로 사용

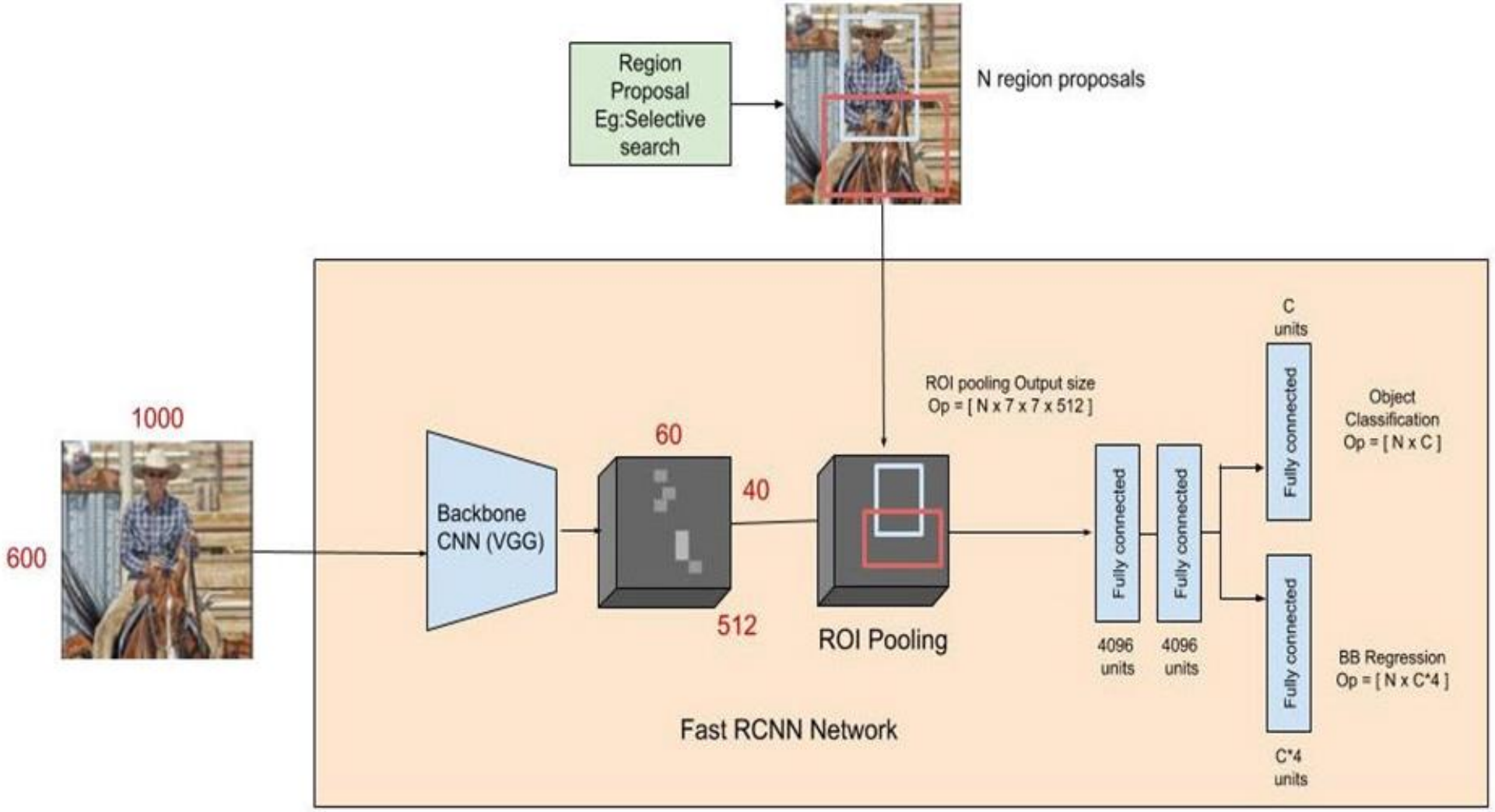
2. Feature Extraction (특징 추출)
: 생성된 Region Proposals은 고정된 크기(227x227) 변형
: CNN 층(ex, AlexNet) 을 통과, 특징 추출 -> 분류기에 입력으로 w전달

3. Classification (분류)
: 추출된 특징 벡터를 여러 개의 SVM (Support Vector Machine) 분류기에 입력
: SVM, 클래스로 분류 수행. + 각 후보 영역이 해당 클래스의 객체를 포함하는지 여부 판단

4. Bounding Box Regression
: 객체의 정확한 위치를 더욱 정확하게 조정 -> 객체의 위치를 보다 정확하게 파악

Object Detection - YOLO

- **Fast R-CNN**



Object Detection - YOLO

R-CNN

- Two stage detectors
- Region proposal: selective search
  - ✓ 2000 개
- Feature extraction: AlexNet
- Object detection
  - ✓ Classification: SVMs
  - ✓ Regression: Linear regression models

Fast R-CNN

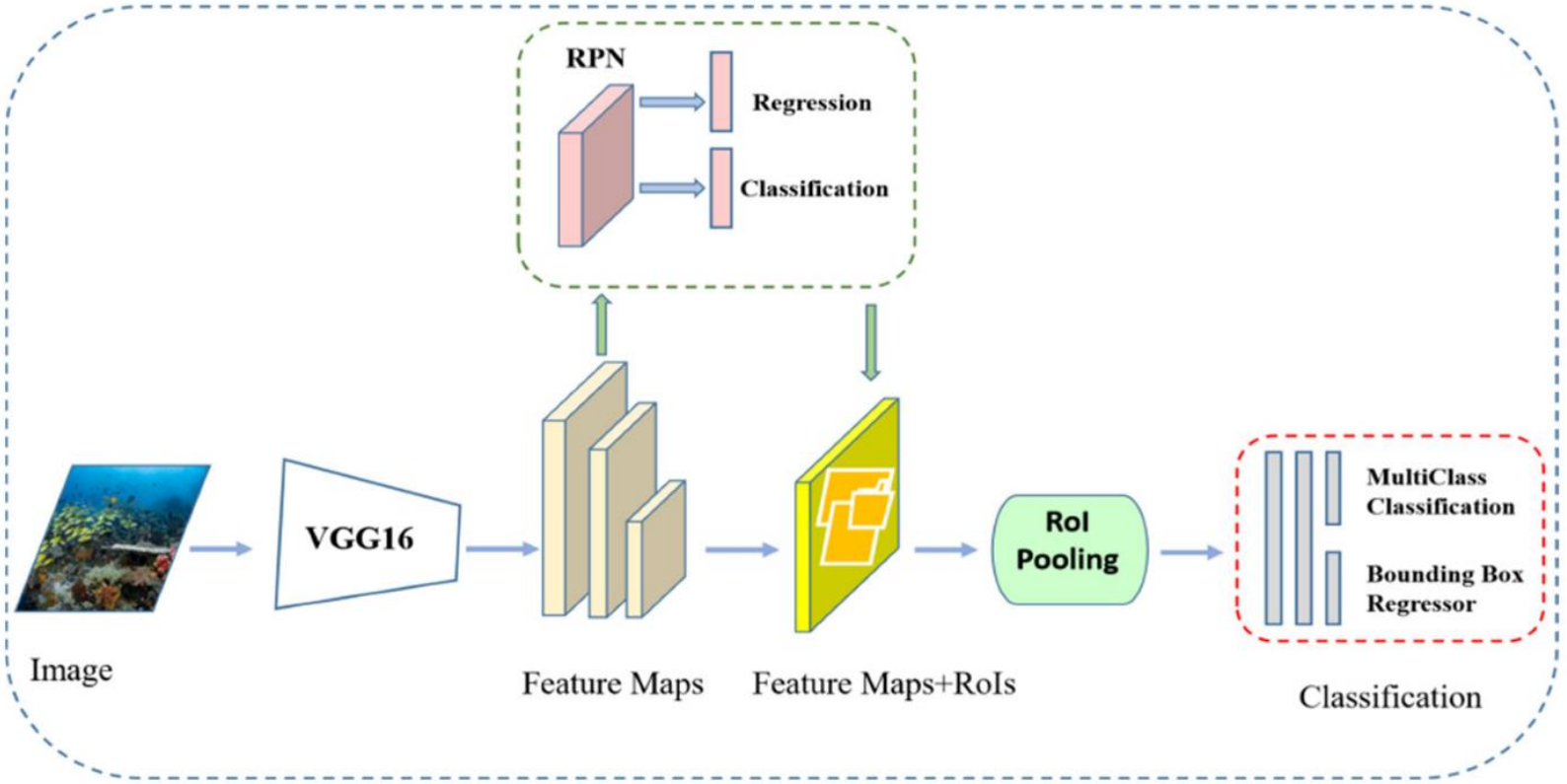
- Feature extraction을 한 번만 수행
- 이미지에 대해 selective search 적용 -> RoIs 추출 -> feature map 에 매핑
- RoI pooling 사용 -> 고정된 크기의 특성 벡터 추출
- Fully connected layer 사용 -> classification과 localization 수행

Object Detection - YOLO

- 모형의 구조
  - RPN (Region proposal networks) + Fast R-CNN
- 주요 단계
  - 단계1: Region proposal network ⇒ ROIs 추출
  - 단계2: 각 ROI에 대해서 Fast R-CNN을 이용해서 classification 과 localization 작업 수행

Object Detection - YOLO

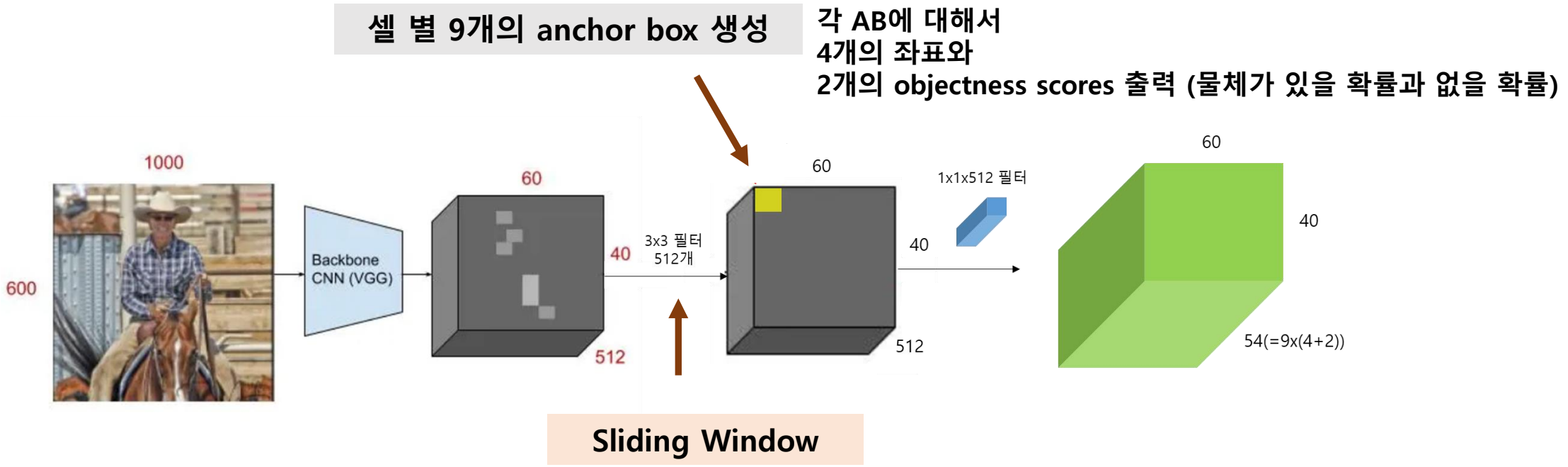
- **Faster R-CNN**
- **RPN (Region proposal networks)** + Fast R-CNN
- 주요 단계
  - 단계1: Region proposal network ⇒ ROIs 추출
  - 단계2: 각 ROI에 대해서 Fast R-CNN을 이용해서 classification 과 localization 작업 수행



Object Detection - YOLO

- **Faster R-CNN**

  RPN (Region proposal networks)

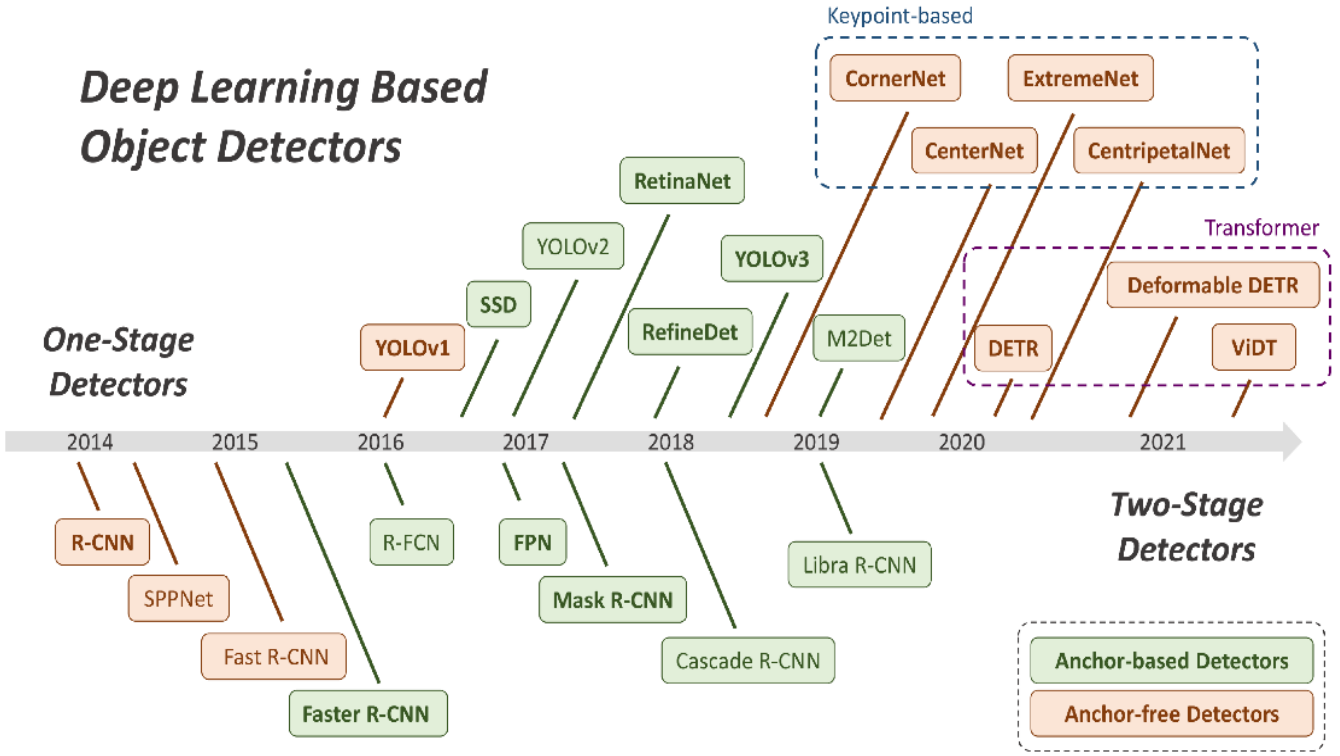


YOLO

(You Only Look Once)

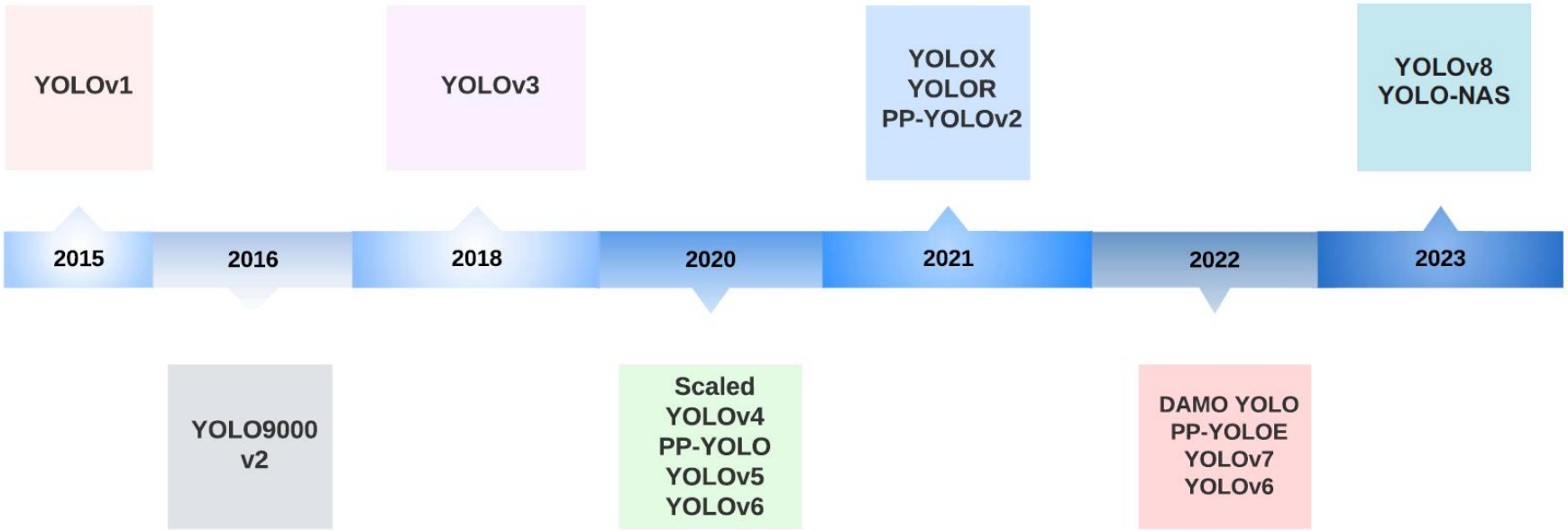
Object Detection - YOLO

객체 탐지(object Detection) - 모델 발전 타임라인



Object Detection - YOLO

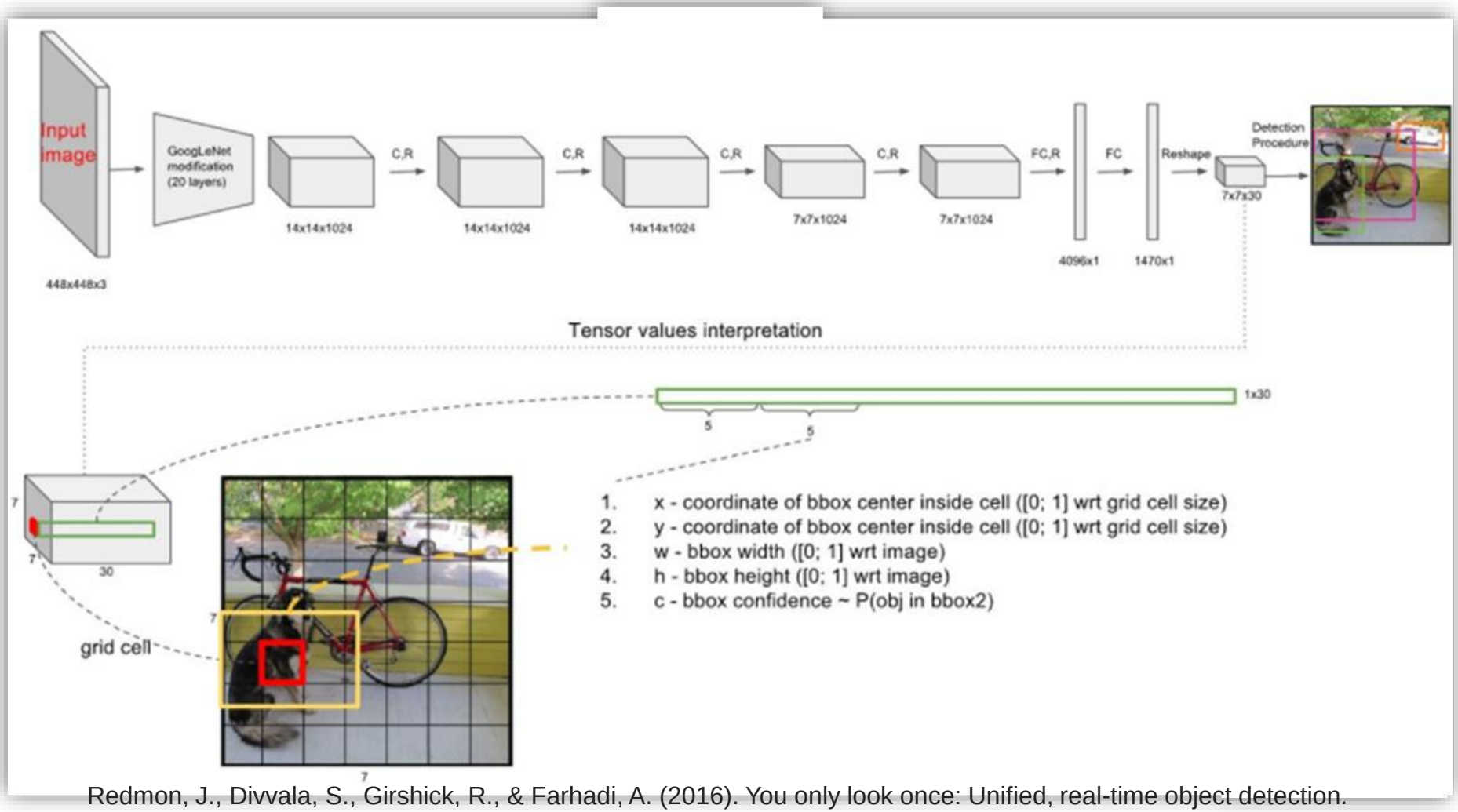
YOLO - 모델 발전 타임라인



<source: Terven, J., & Cordova-Esparza, D. A comprehensive review of YOLO: From YOLOv1 and beyond. arXiv 2023. *arXiv preprint arXiv:2304.00501*.>

Object Detection - YOLO

Model Architecture – V1



Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).

Object Detection - YOLO

Model Architecture V1 – 특징

- InceptionNet, 7x7x1024 형태의 feature map 생성 (7x7 셀)
- 하나의 셀이 하나의 객체 탐지
- 하나의 셀이 두 개의 후보 bounding box를 예측
  - 각 bounding box에 대해 좌표와 confidence score 계산
  - 이 중 confidence score가 높은 상자를 **책임 상자**로 간주
- 하나의 셀에 대해서 물체가 각 클래스에 속할 확률 계산
- PASCAL VOC 데이터셋 사용

Object Detection - YOLO

Model Architecture V1 – 성능

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

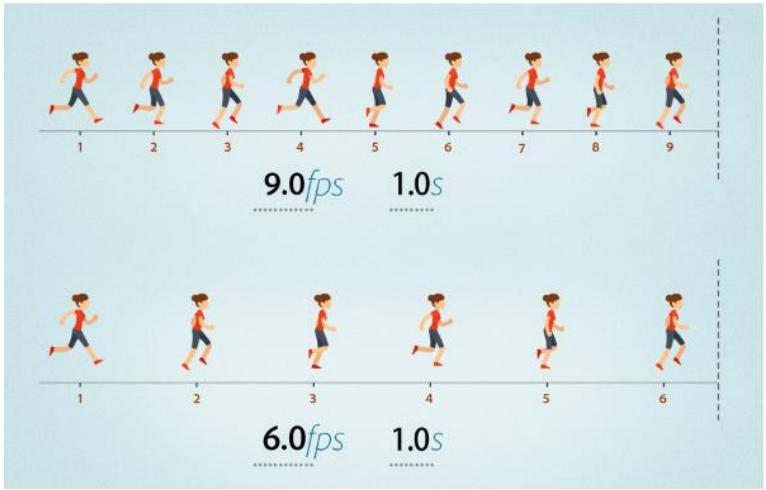
Object Detection - YOLO

Model Architecture V1 – 성능

mAP mean Average Precision

Ground Truth \ Predict Result	Positive	Negative
Positive	TP	FN
Negative	TN	FP

FPS Frame Per Second

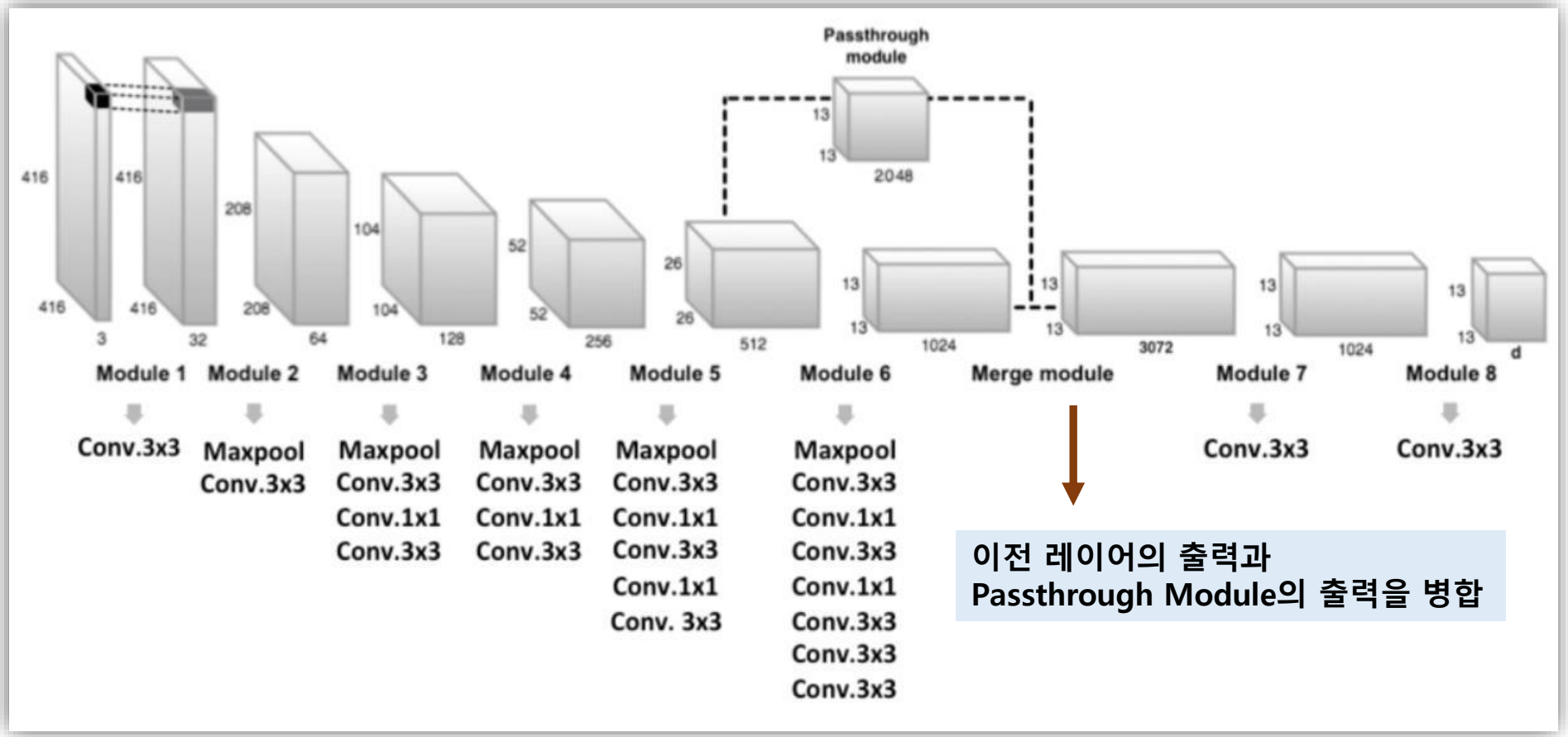


Object Detection - YOLO

Model Architecture – V2

Passthrough Module

이전 레이어의 저해상도 특징을
현재 레이어의 고해상도 특징과 결합



초기 레이어
저수준 특징(에지, 코너 등) 추출

Object Detection - YOLO

Model Architecture V2 – 특징

- Anchor box
  - 각 셀마다 5개 AB 사용
  - Clustering 방법 (K-Means) 사용 AB 형태 결정

- PassThrough module
  - cf) Skip connection
  - 작은 object 탐지

- Feature extractor
  - Darknet-19 사용

- Multi-Scale Training
  - 학습시 10 회 배치 마다 입력 이미지 크기를 320 부터 608 까지 동적으로 변경 (32 의 배수로 설정)

Object Detection - YOLO

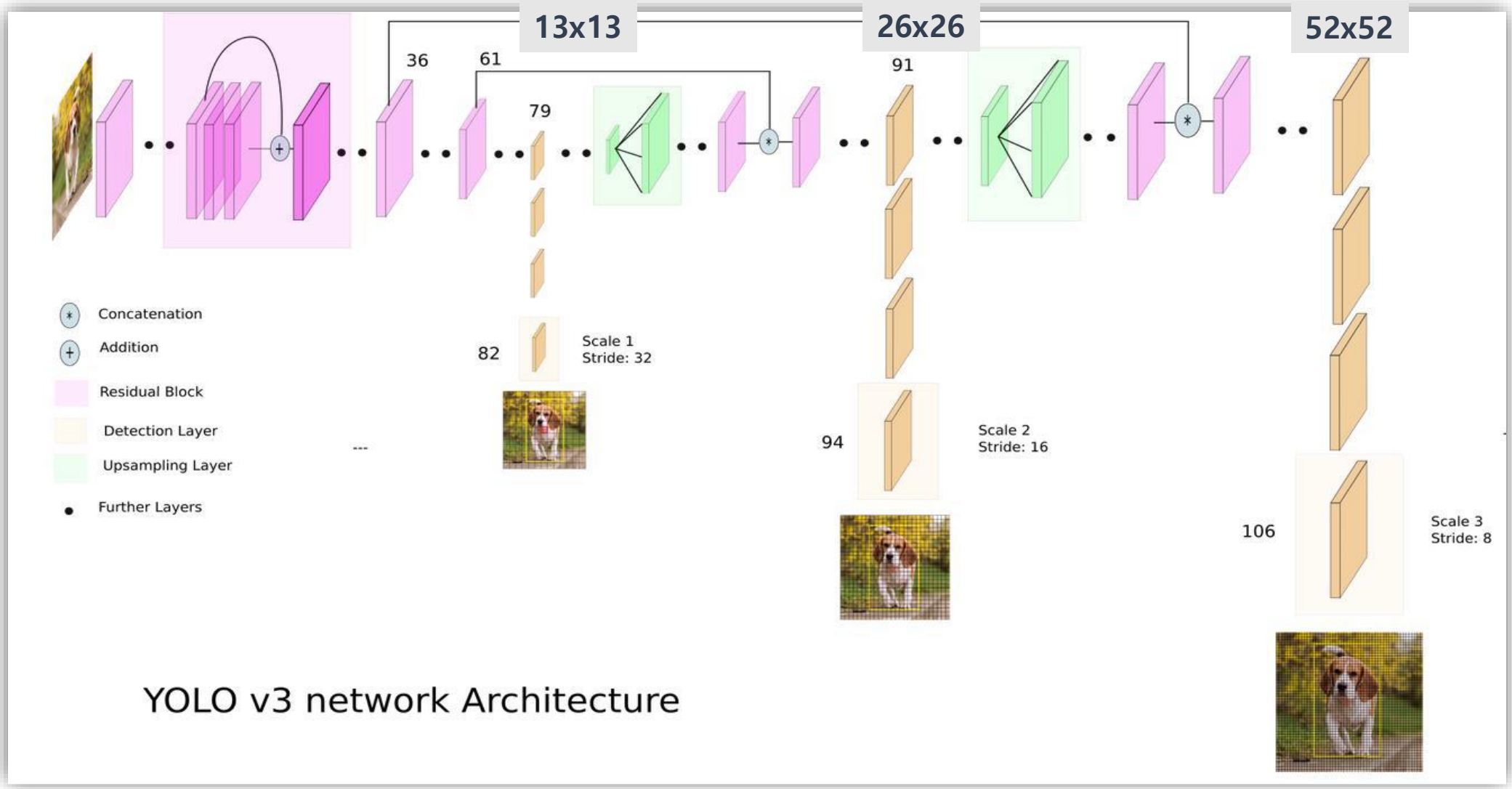
Model Architecture V2 – 성능

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

Object Detection - YOLO

Model Architecture – V3

Multi-scale Prediction



Object Detection - YOLO

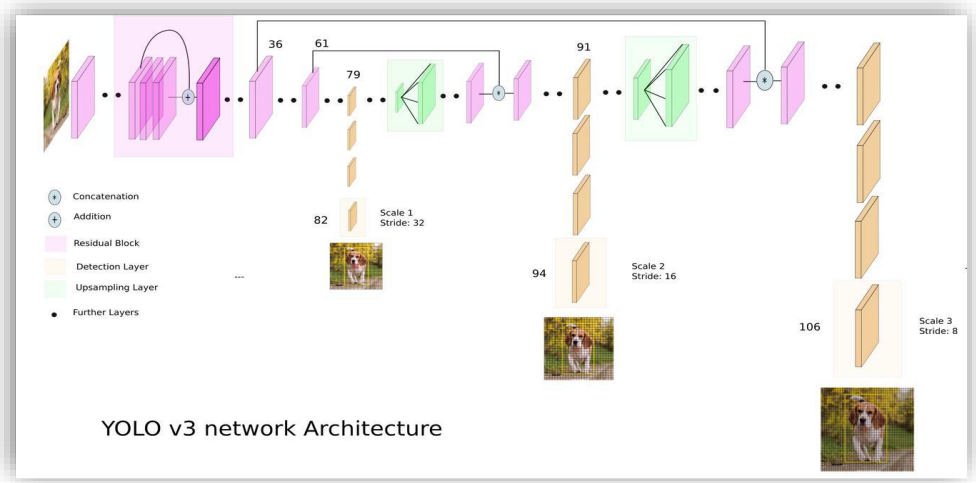
Model Architecture V3 – 특징

Fast and Efficient

Multi-scale Prediction
다양한 크기의 객체 탐지

Upsampling Layer
작은 객체 탐지

Residual Blocks
ResNet
입력추가
경사소실 대등



Object Detection - YOLO

Model Architecture V3 – 특징

- 3개의 feature map을 이용 -> detection
  - **Feature pyramid network**
  - 13x13, 26x26, 52x52 feature map

- Darknet-53 backbone
  - ResNet을 개선한 방법

Object Detection - YOLO

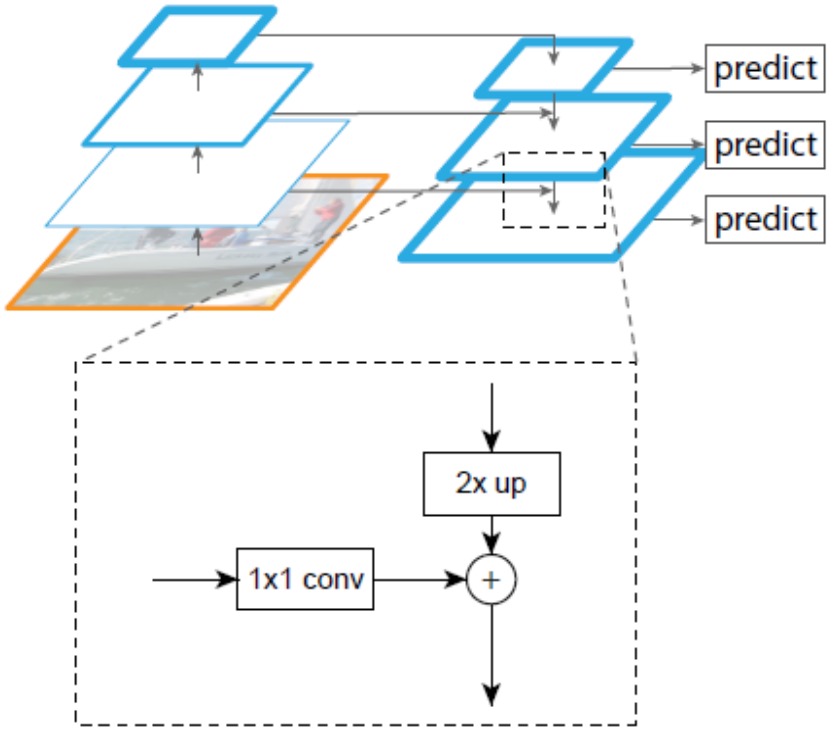
Model Architecture V3 – 특징

- **Feature pyramid network**

객체 탐지 및 분할 작업에서 다양한 크기의 객체를 효과적으로 처리하기 위해 사용하는 신경망 구조

일반적인 피라미드 구조를 활용하여 서로 다른 해상도의 특징 맵을 결합하고,
이를 통해 다중 스케일에서의 객체 탐지 성능을 향상시킵니다.

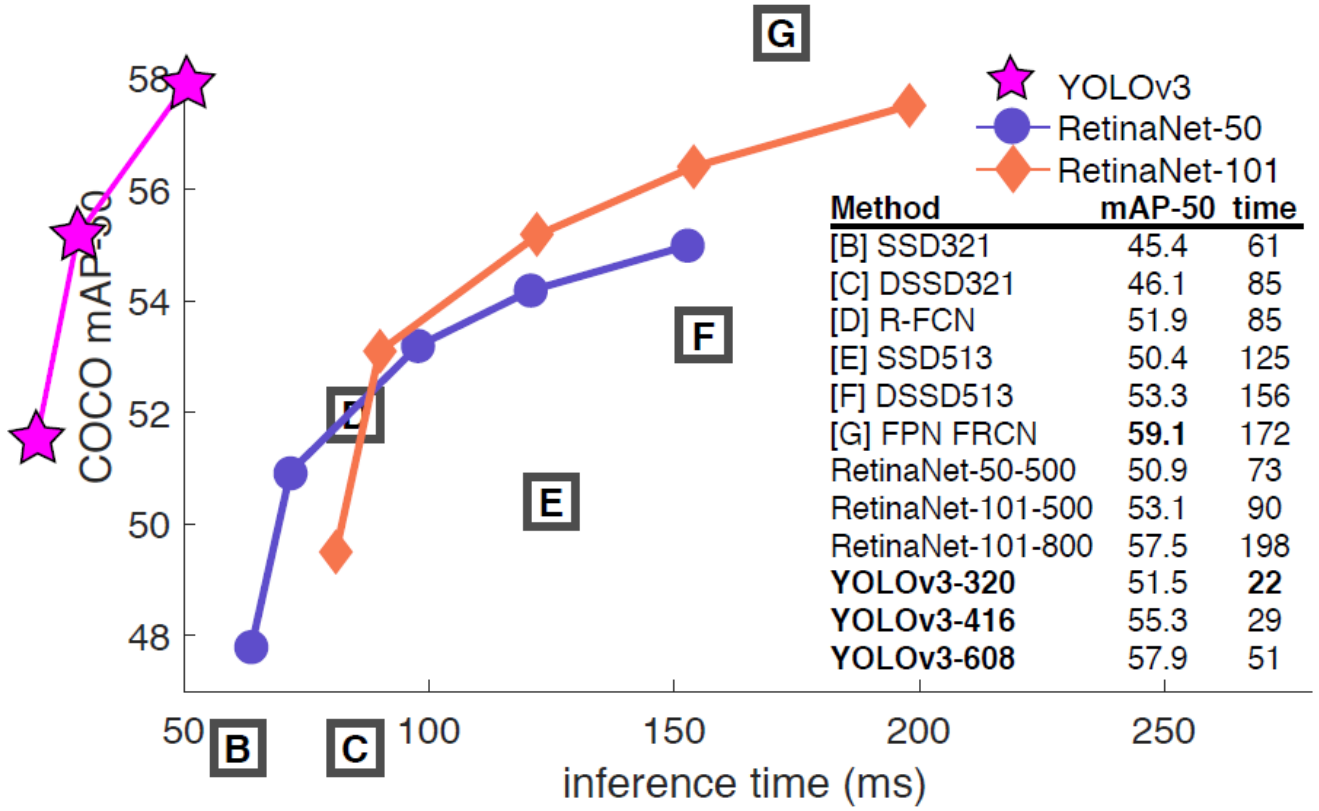
Top-down Pathway
Feature Map Fusion



Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2117-2125).

Object Detection - YOLO

Model Architecture V3 – 성능



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

Object Detection - YOLO

Model Summary

특징	YOLO v1	YOLO v2	YOLO v3
입력 이미지	446* 446	416*416	416*416
Feature Extraction	Custom CNN (24 Convlayers)	Darknet-19 (19 Conv layers)	Darknet-53 (53 Convl layers)
Anchor Box 결정	앵커 박스 없음 - 직접 바운딩 박스 예측	K-means 클러스터링	K-means 클러스터링성
출력 Feature Map 크기	7x7	13x13 (기본), 26x26 (Passthrough Layer)	13x13, 26x26, 52x52 (3개의 피쳐맵 사용)
Feature Map Scaling	단일 스케일 (7x7)	멀티 스케일 Passthrough Layer	멀티 스케일 FPN
mAP 성능	COCO mAP 48.1 (약 52ms)	COCO mAP 76.8 (약 25ms)	COCO mAP 57.9 (약 51ms)
기타 특성	한 번의 예측으로 클래스와 바운딩 박스 동시에 예측	배치 정규화 사용	잔차 연결

Object Detection - YOLO

- 실습 – Object Detection 사전학습 모델을 통한 객체 인식

7.05.OD.yolo_for_class.ipynb

THANK YOU