PIAI

Object Detection

Dahyun, Kim

FLOW

1. Intro

- What is Object Detection?
- Classification + Localization
- 1-Stage, 2-Stage Object Detection

2. YOLO

- YOLO v1 Architecture
- YOLO v2 Architecture
- YOLO v3 ~ v5

3. Object Detection Metrics

1. YOLO V5 Code Running

- Image Inference : Use detect.py
- Video Inference : Use detect.py
- Inference Image using pytorch load model function

PIAI
인공지능연구원
POSTECH Institute of Artificial Intelligence

가치창출대학
POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

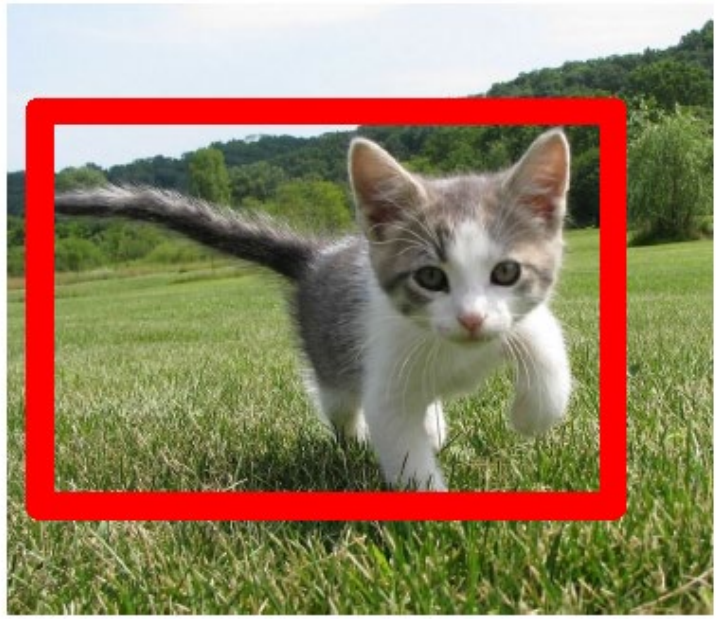
Intro

Dahyun, Kim

Object Detection

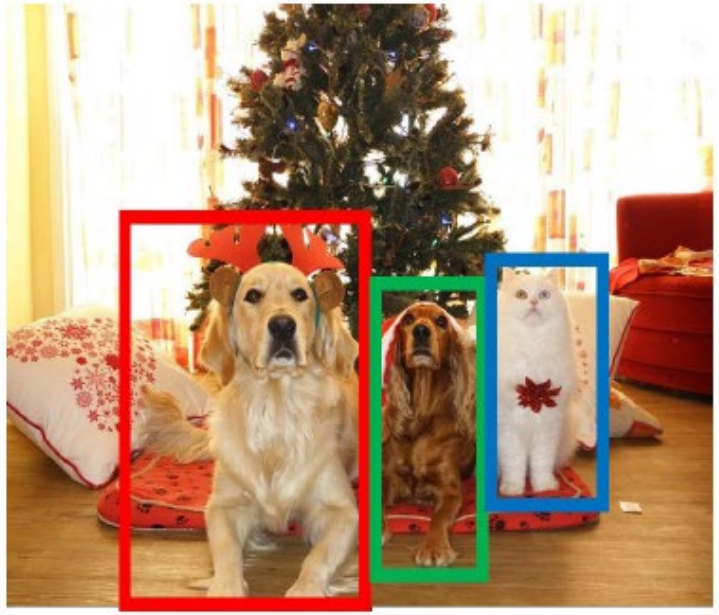
PIAI Research Department

Classification + Localization



CAT

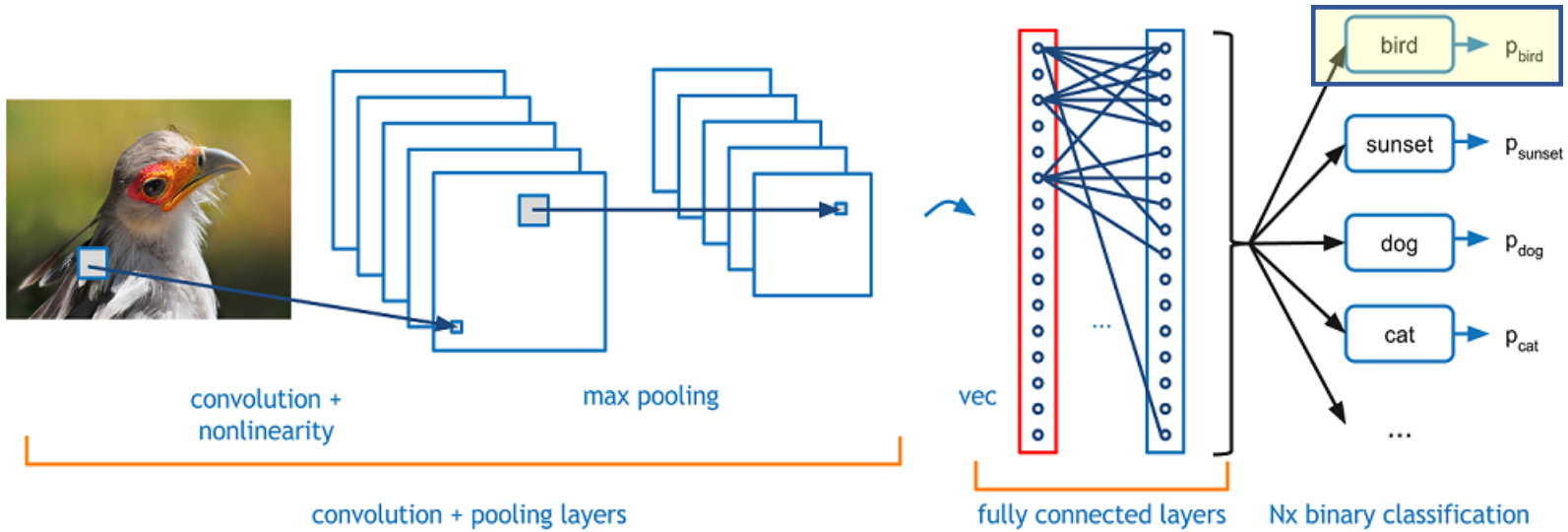
Object Detection



DOG, DOG, CAT

Image classification

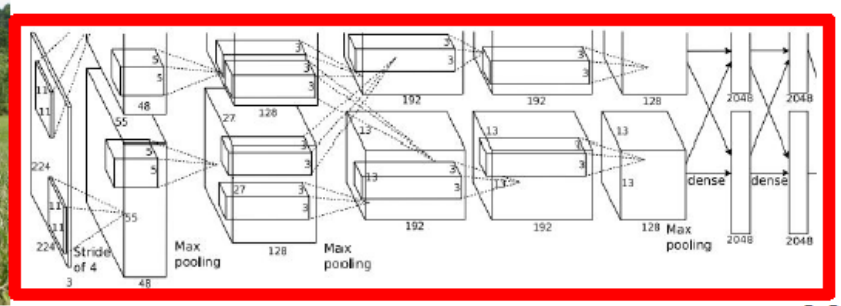
PIAI Research Department



Classification + Localization

PIAI Research Department

This image is CC0 public domain



Often pretrained on ImageNet
(Transfer learning)

Vector:
4096

Fully Connected:
4096 to 1000

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Softmax Loss

Correct label:
Cat

Regression B-box Head!!

Fully Connected:
4096 to 4

Box Coordinates
(x, y, w, h)

L2 Loss

+ → Loss

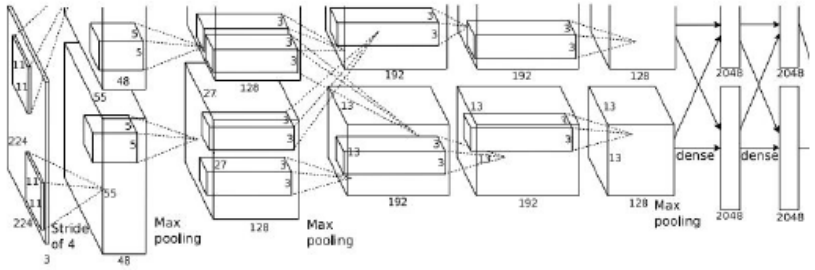
Correct box:
(x', y', w', h')

Treat localization as a regression problem!

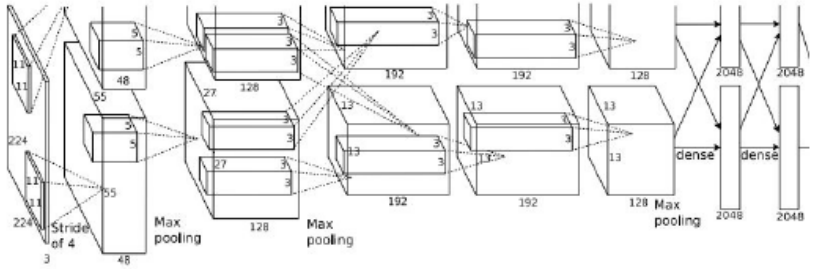
Object Detection as Regression ?

PIAI Research Department

Each image needs a different number of outputs!

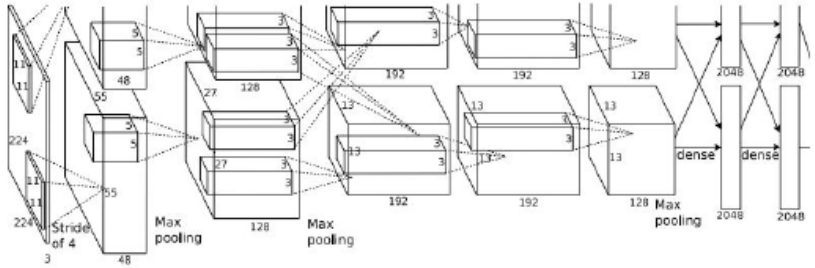


CAT: (x, y, w, h) — 4 numbers



DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)

12 numbers

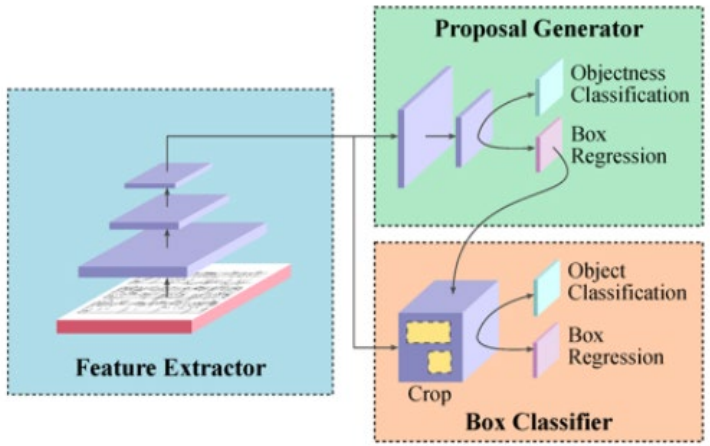
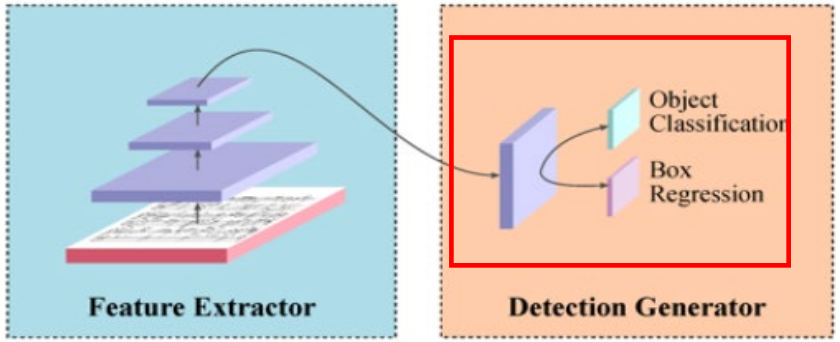


DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
….

Many numbers!

Object Detector (1-stage, 2-stage)

PIAI Research Department

2-Stage Object Detector	1-Stage Object Detector
	
- 1-Stage Object Detector에 비해 높은 검출 성능 - 1-Stage Object Detector에 비해 속도가 느림 - 정확성은 높으나 속도로 인해 실시간 객체 탐지가 어려움 - 대표 모델 : R-CNN / Fast R-CNN / Faster R-CNN	- 2-Stage Object Detector에 비해 낮은 검출 성능 - 2-Stage Object Detector에 비해 속도가 빠름. - 속도가 빠름으로 인해 실시간 객체 탐지가 가능 - 대표 모델 : RetinaNet, SSD, EfficientDet, YOLO 시리즈

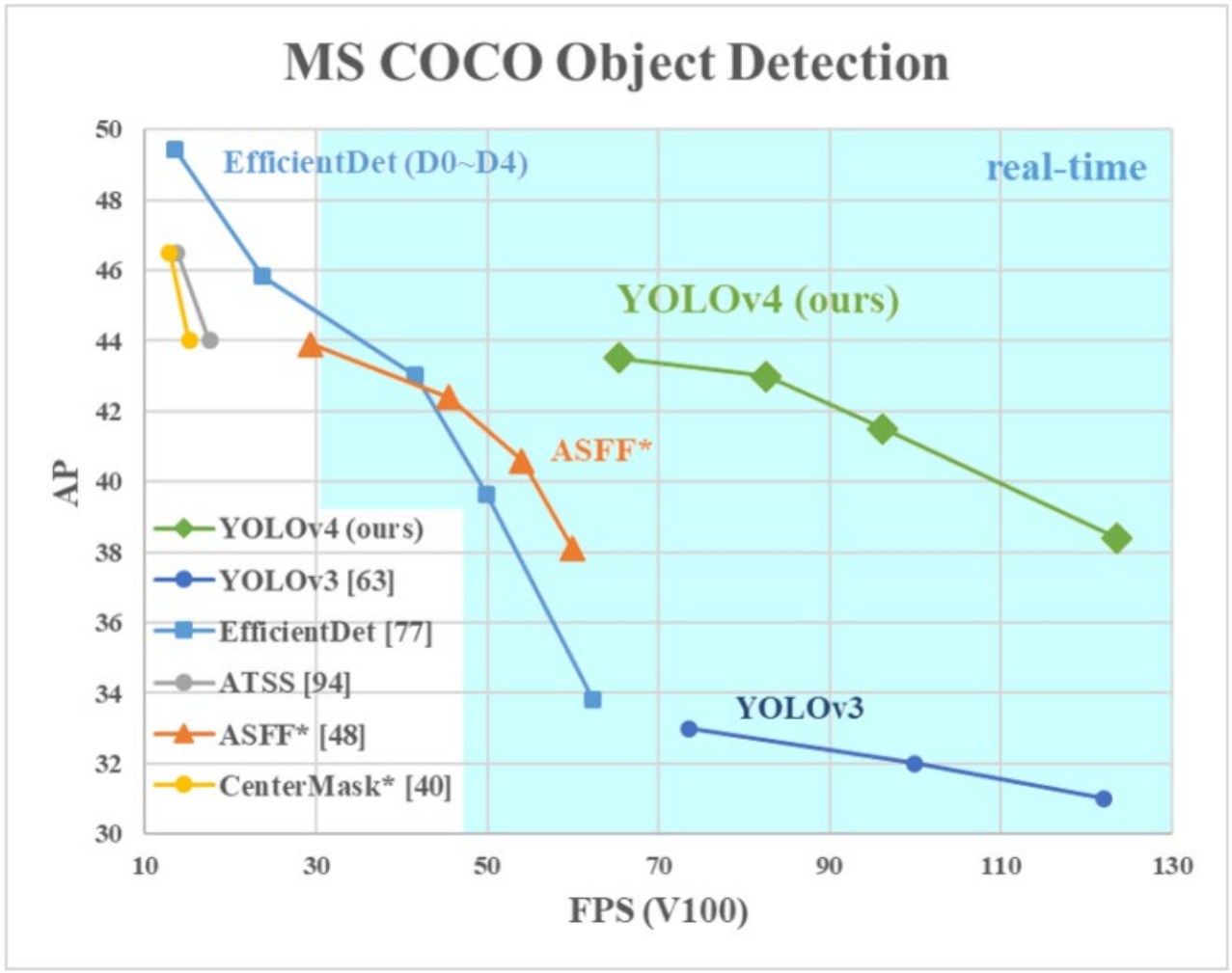
PIAI
인공지능연구원
POSTECH Institute of Artificial Intelligence

가치창출대학
POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

YOLO

Dahyun, Kim

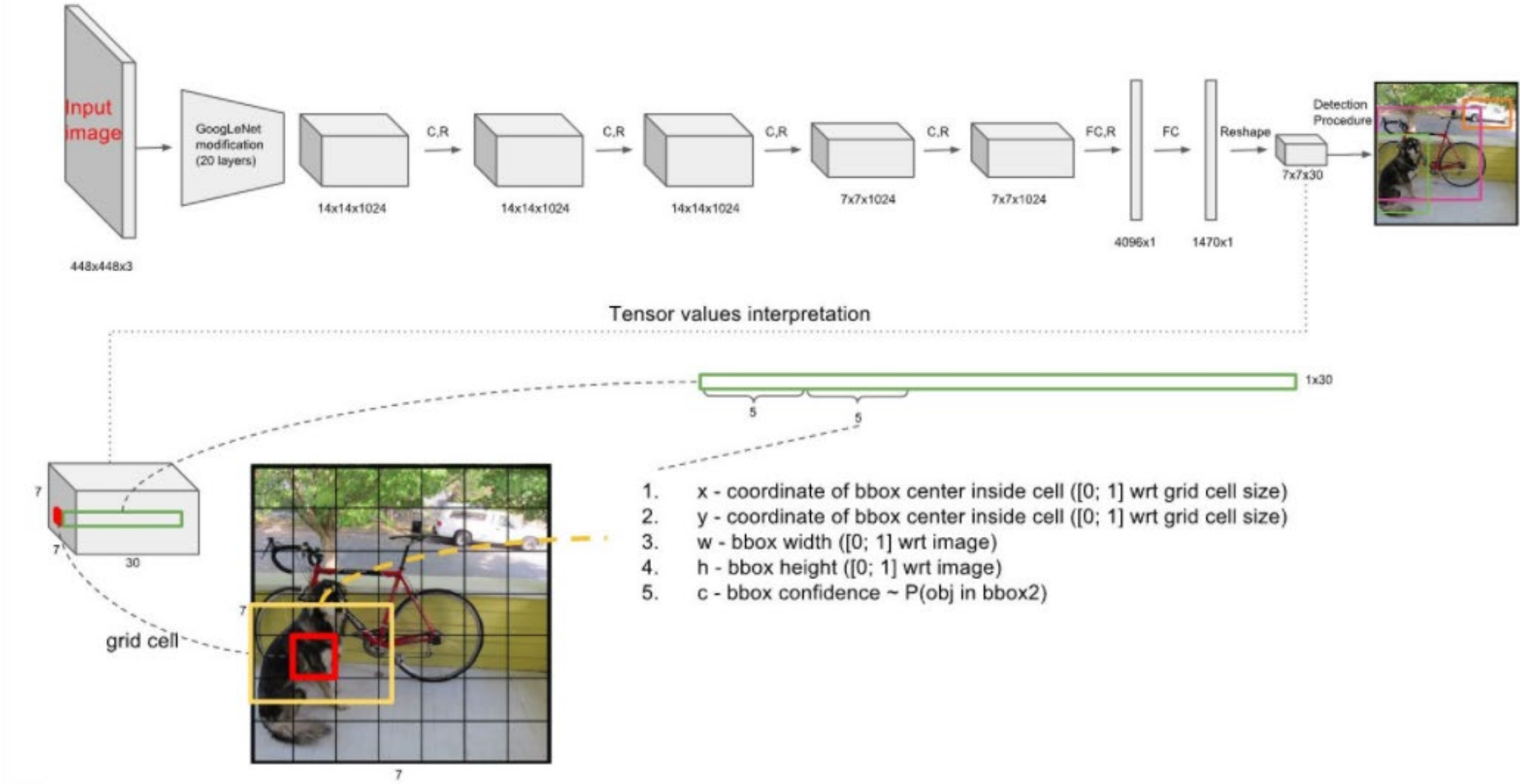
Why YOLO?



실시간 객체 탐지 가능

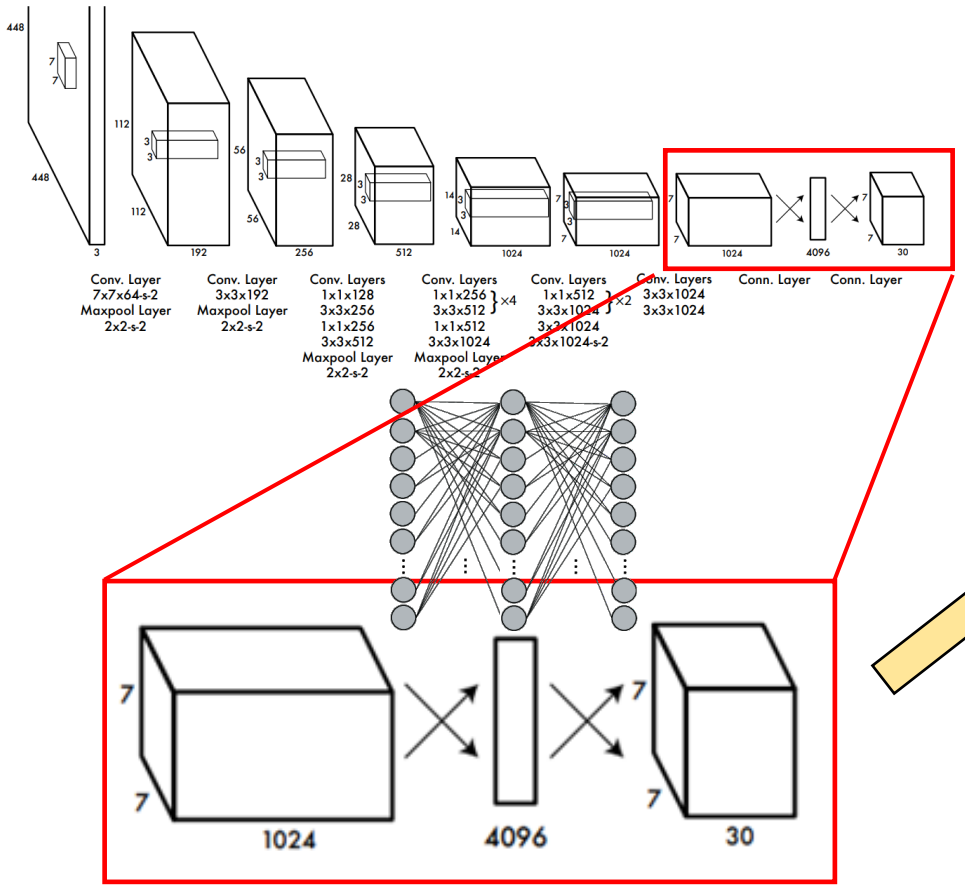
YOLO flow

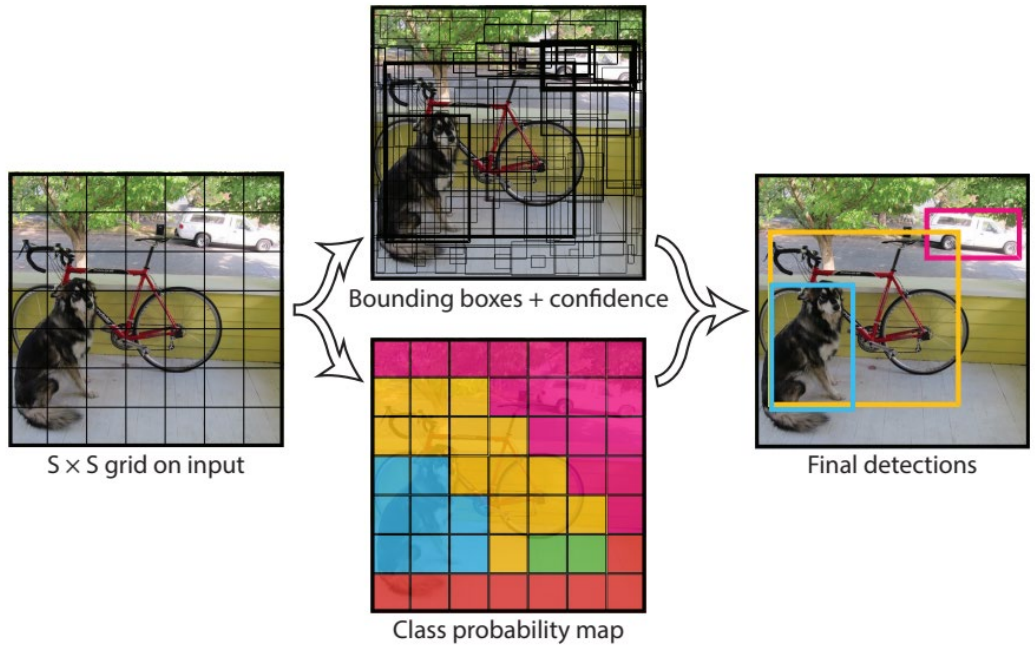
PIAI Research Department



YOLO network

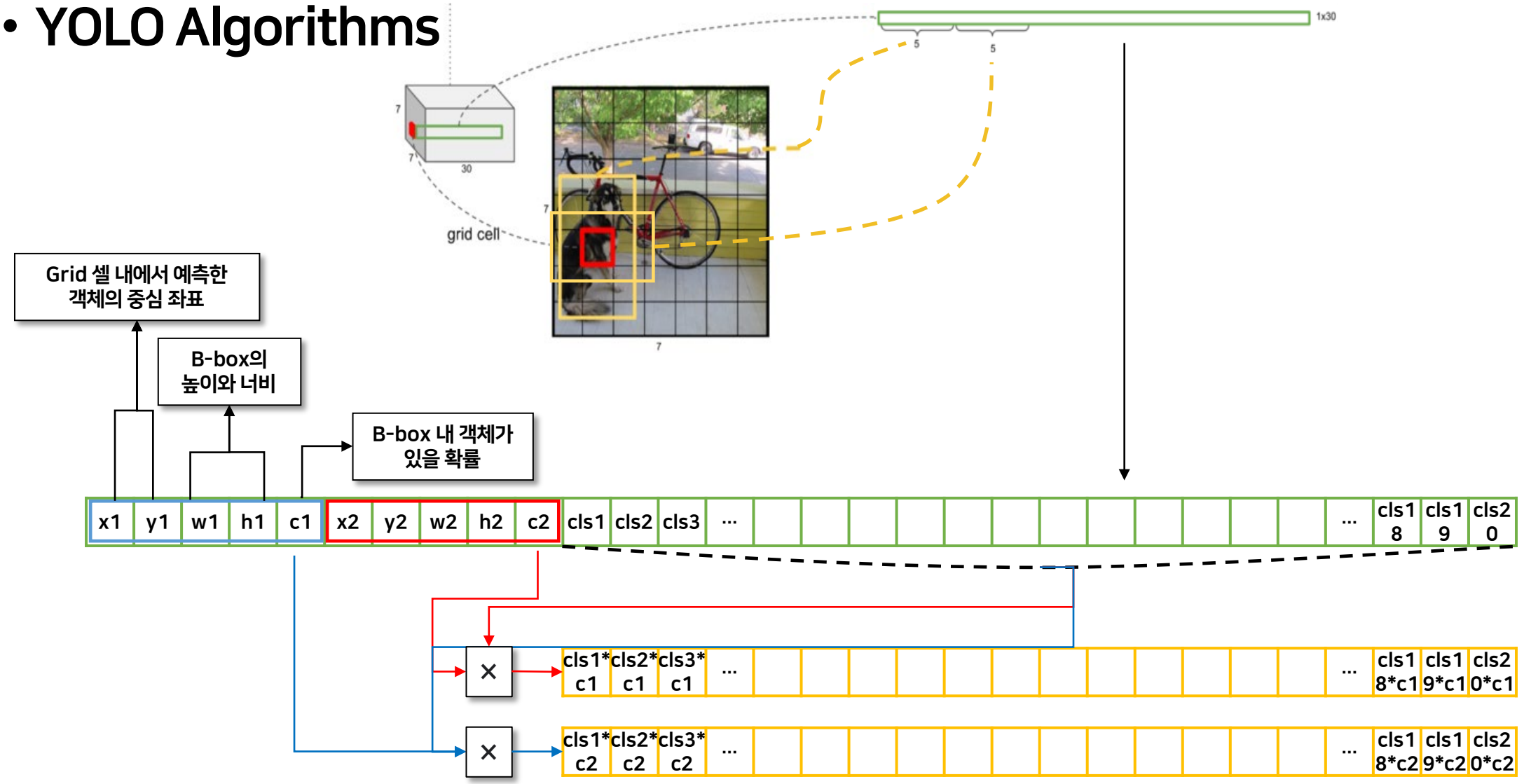
- YOLO Architecture



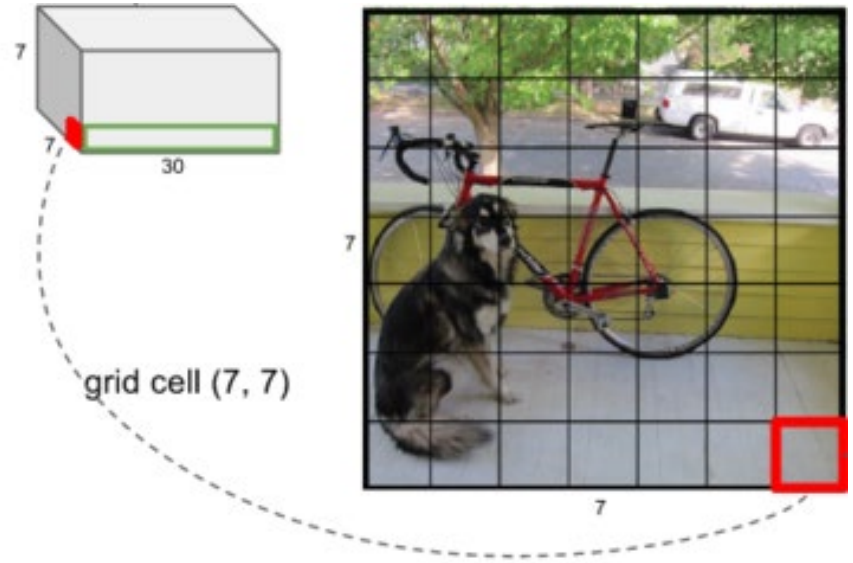


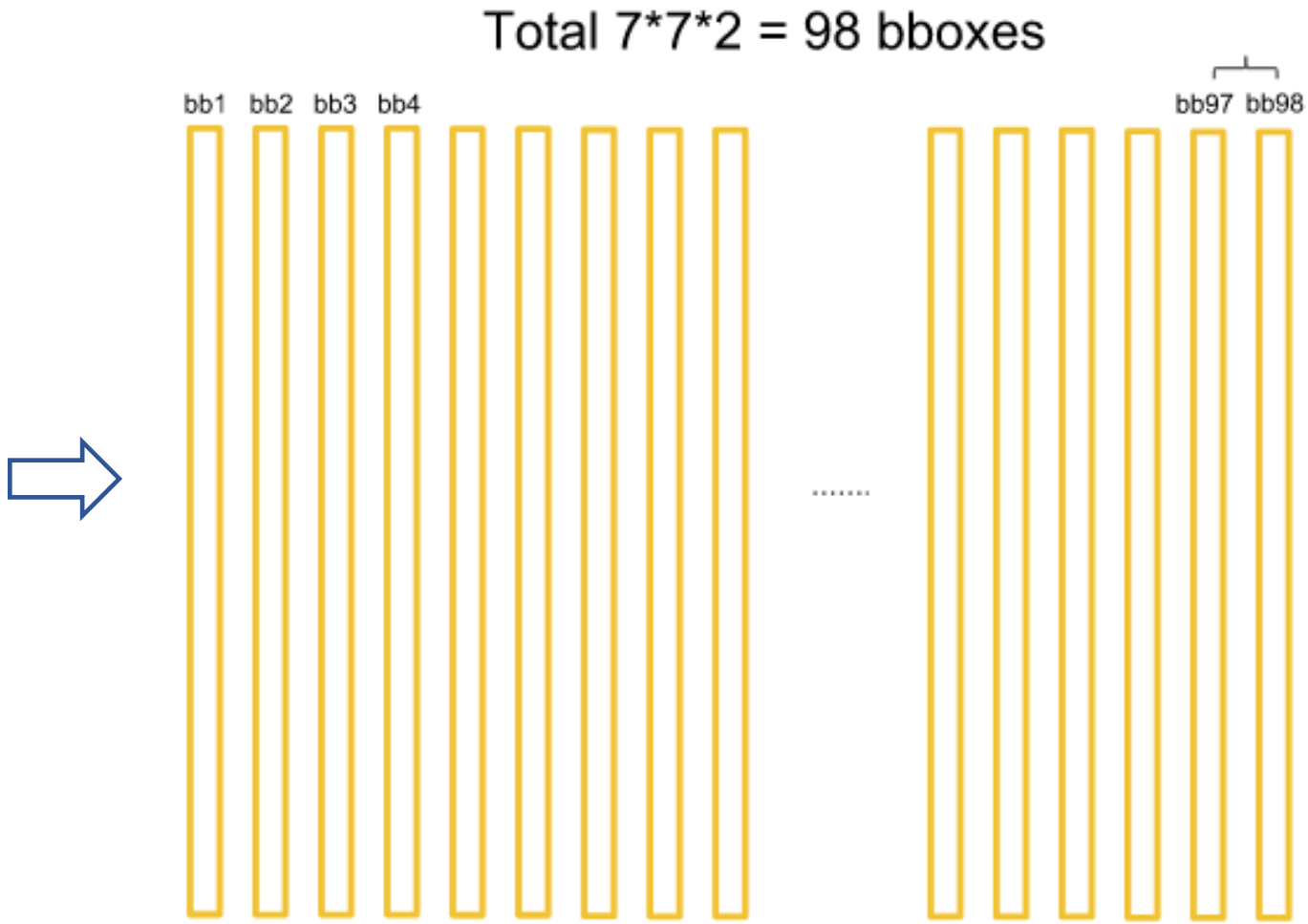
$$S \times S \times (B * 5 + C) = 7 \times 7 \times (2 * 5 + 20)$$

- YOLO Algorithms

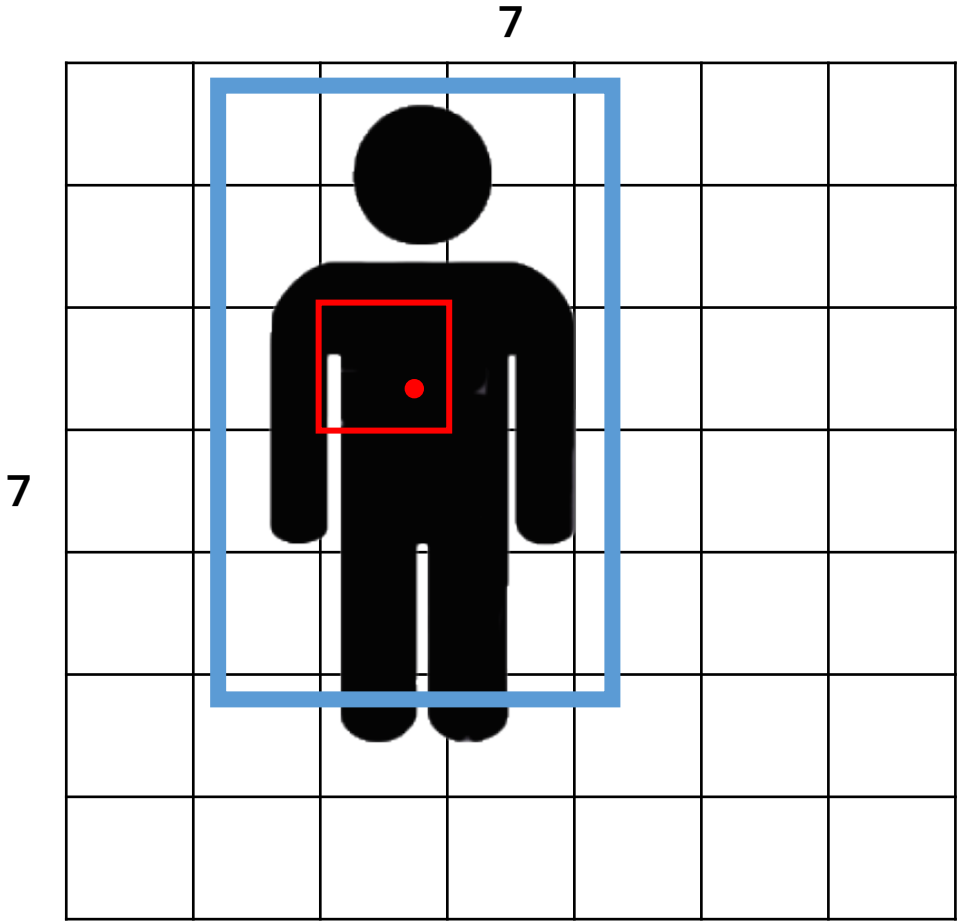


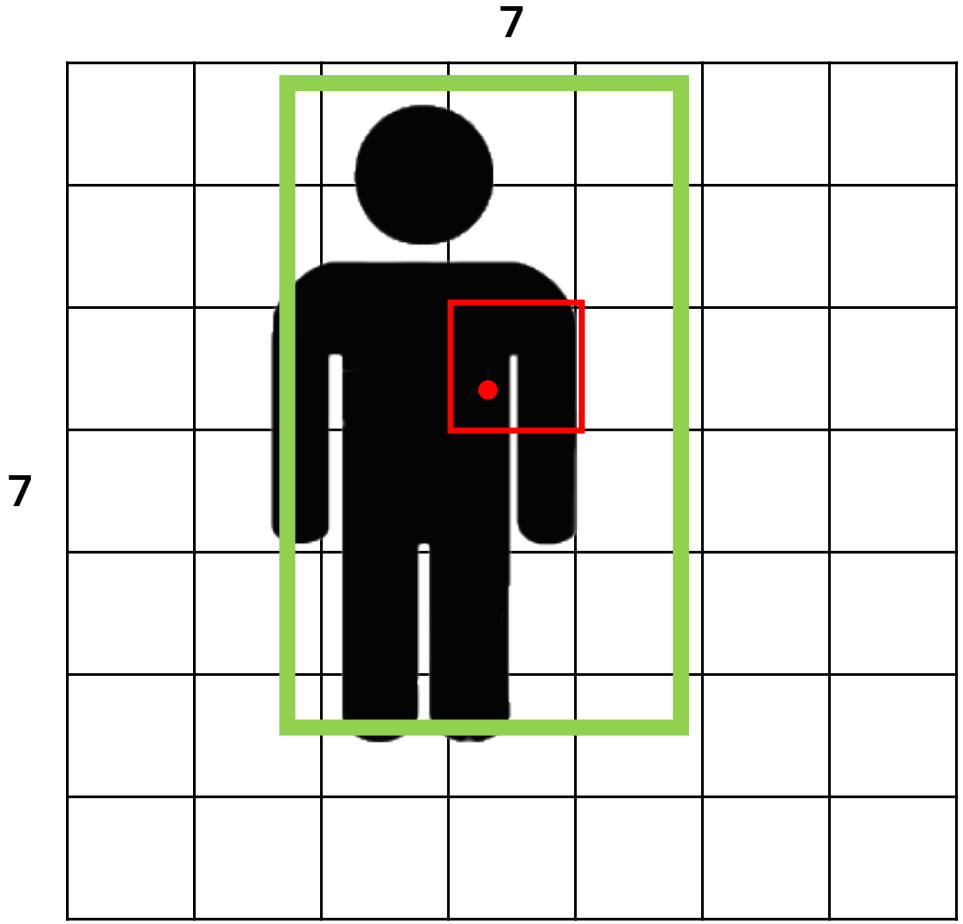
- YOLO Algorithms



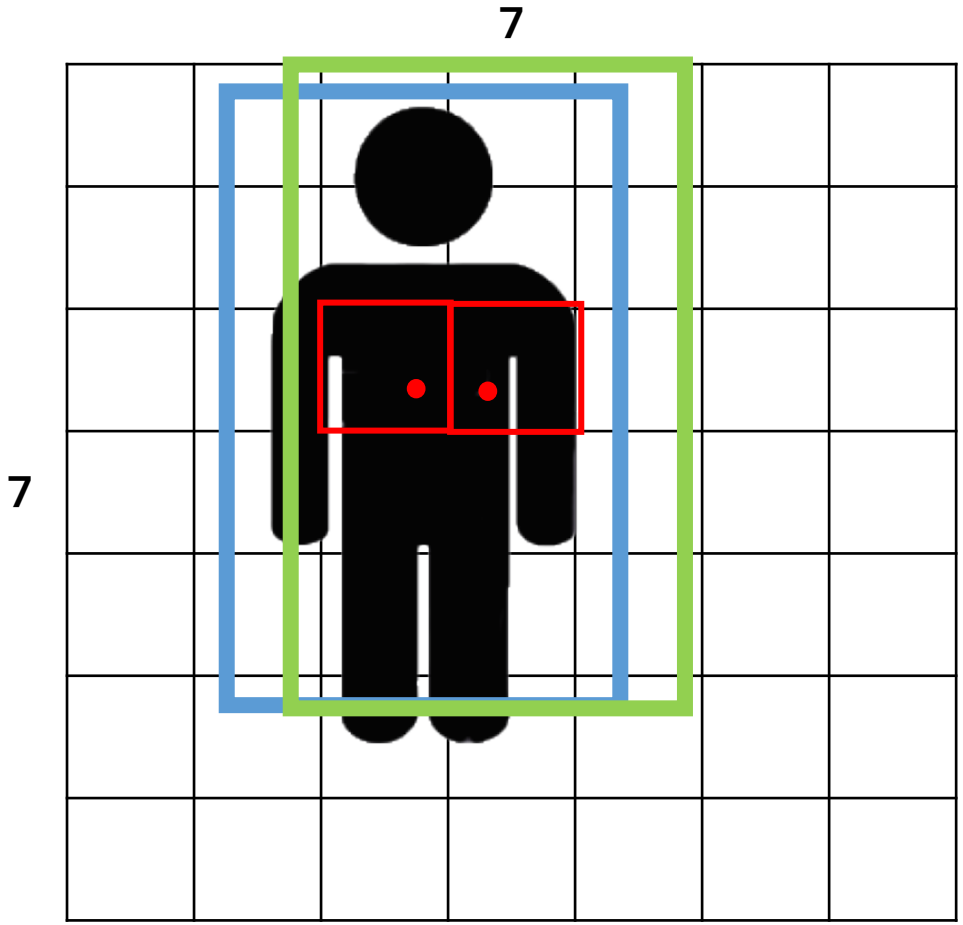


- Overlap Problem





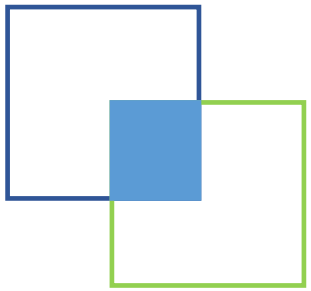
- Overlap Problem

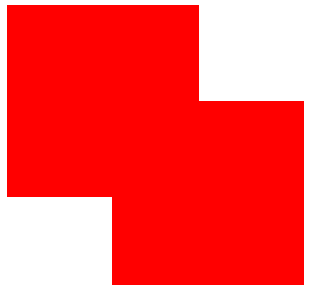


두개의 그리드 셀에서 생성된 각각의 B-box가 같은 객체 영역을 가리키고 있음 → 해결방법 : NMS

- **IoU(Intersection over Union)**

- B-box 간의 겹침 정도를 확인 할 수 있는 지표로서, 객체 검출에 대한 성능 지표로도 사용됨





$$IoU = \frac{\text{Overlapping Region}}{\text{Combined Region}} = \frac{Bbox_1\ Area \cap Bbox_2\ Area}{Bbox_1\ Area \cup Bbox_2\ Area}$$

YOLO - Non Maximum Suppression

- **NMS(Non Maximum Suppression)**

1. 목록에서 가장 높은 Confidence b-box 선택
2. 선택된 b-box와 다른 모든 b-box와 IoU 비교
3. 다른 b-box와의 IoU 값이 임계 값 이상이면 목록에서 제거
4. 반복

PIAI
인공지능연구원
POSTECH Institute of Artificial Intelligence

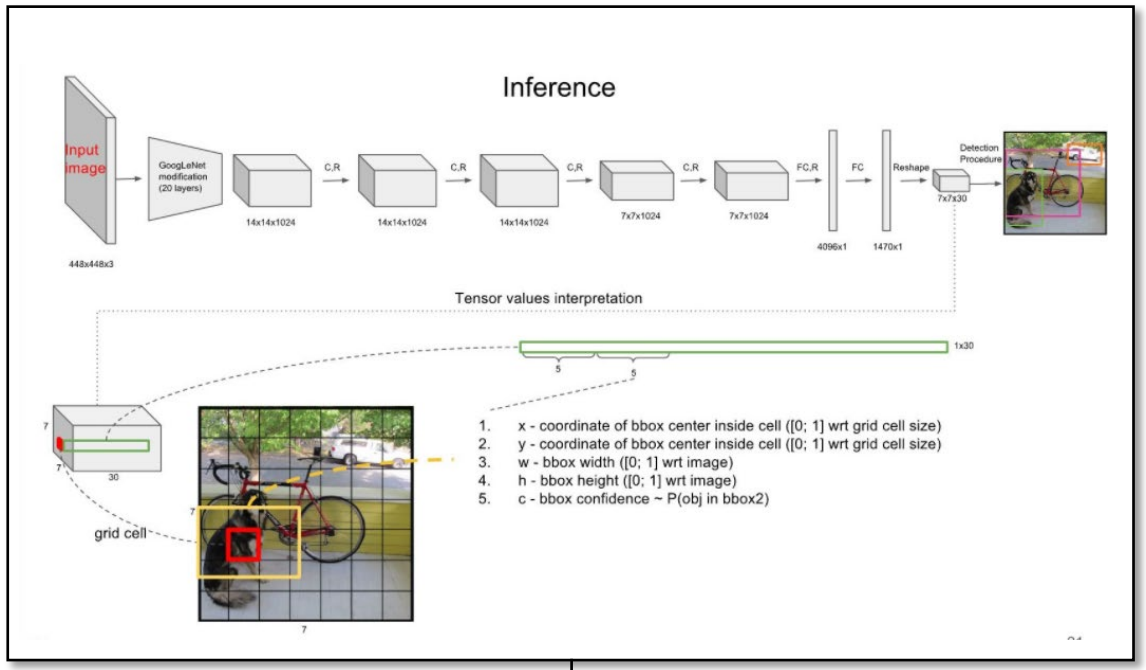
YOLO v2

Dahyun, Kim

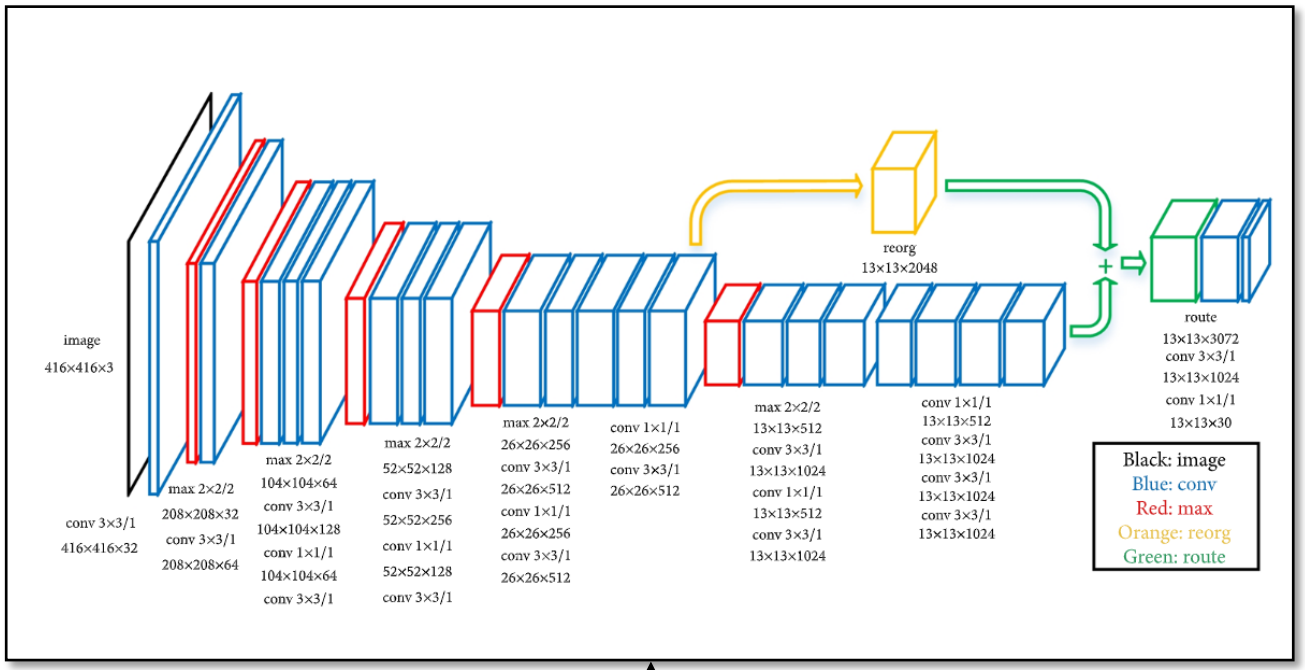
YOLOv2

- **YOLO v1 vs YOLO v2**

YOLO V1



YOLO V2

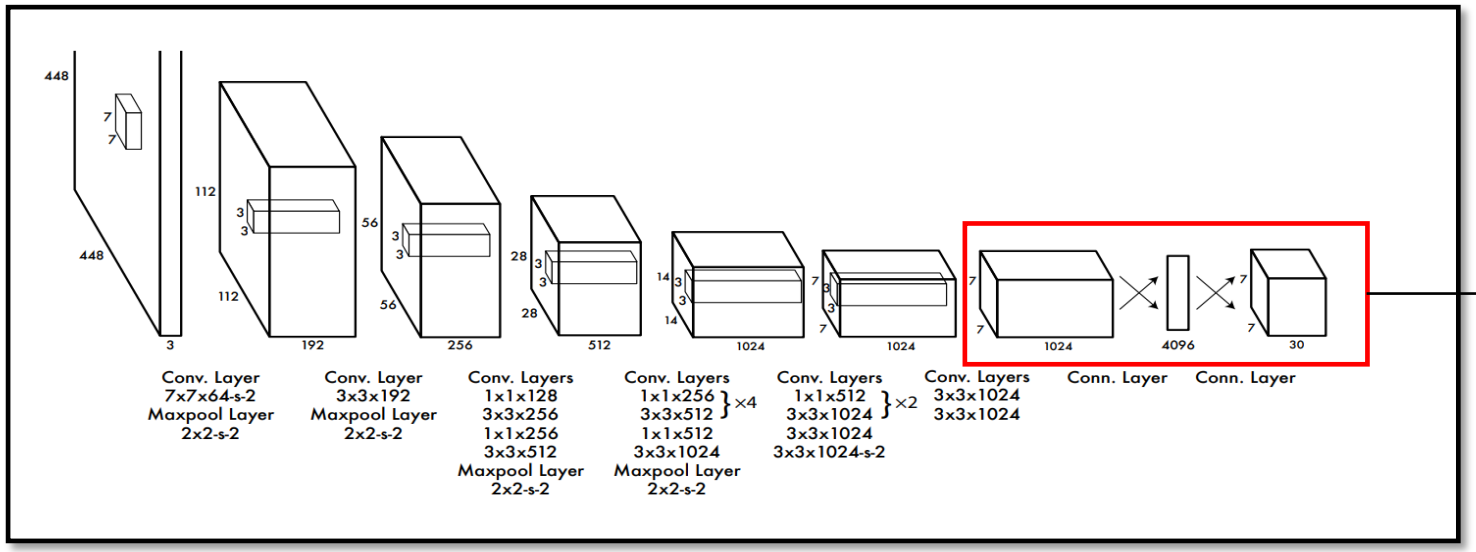


- GoogLeNet → Darknet-19
- Fully Connected Layer → 1×1 Convolution Layer
- Grid 당 2개 B-Box 좌표 예측 → Grid 당 5개의 Anchor Box를 찾음
- fine-grained features 적용

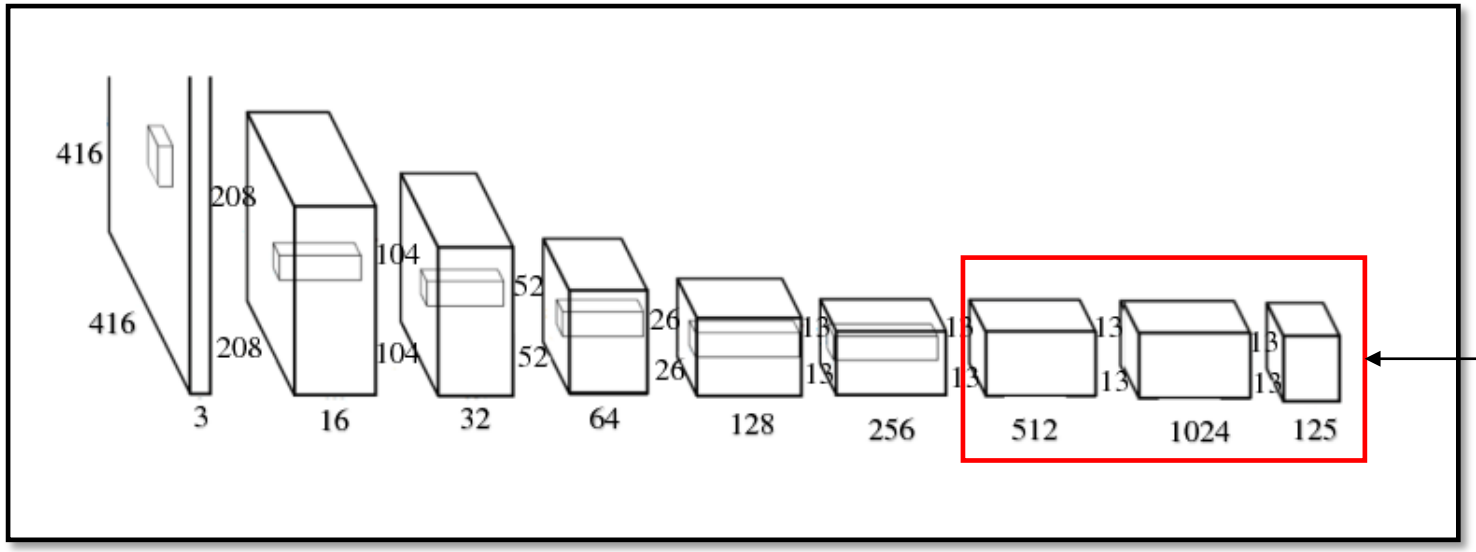
YOLOv2 network

PIAI Research Department

YOLO v1



YOLO v2



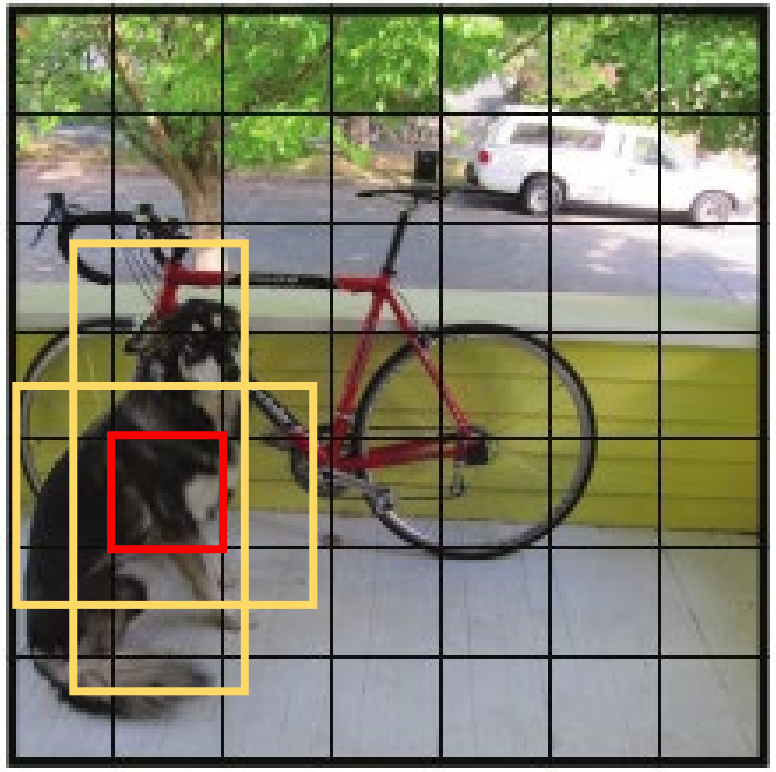
Fully Connected Layer

Convolution Network

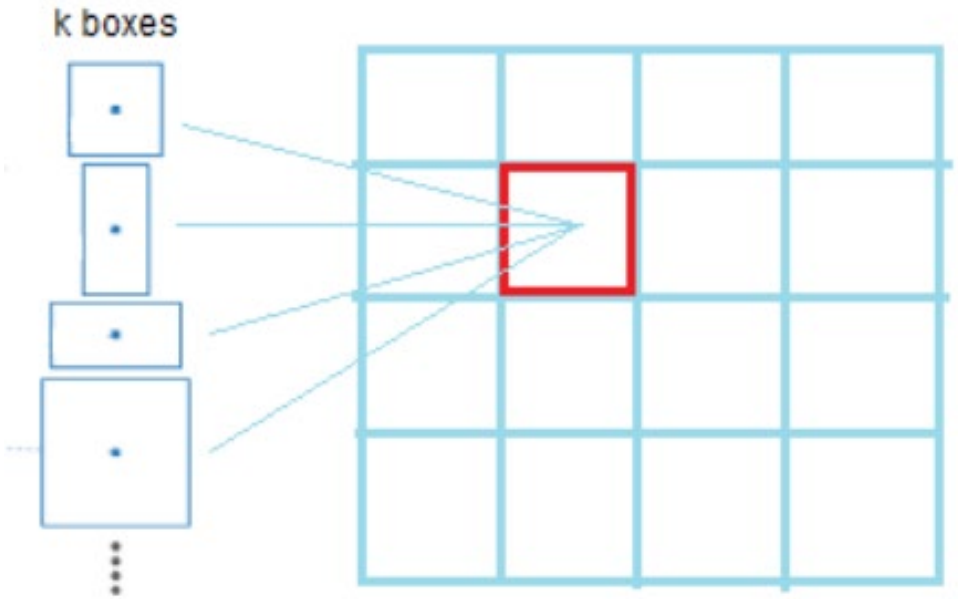
YOLOv2 network

- YOLO v2 B-box Prediction : Anchor Box

YOLO v1



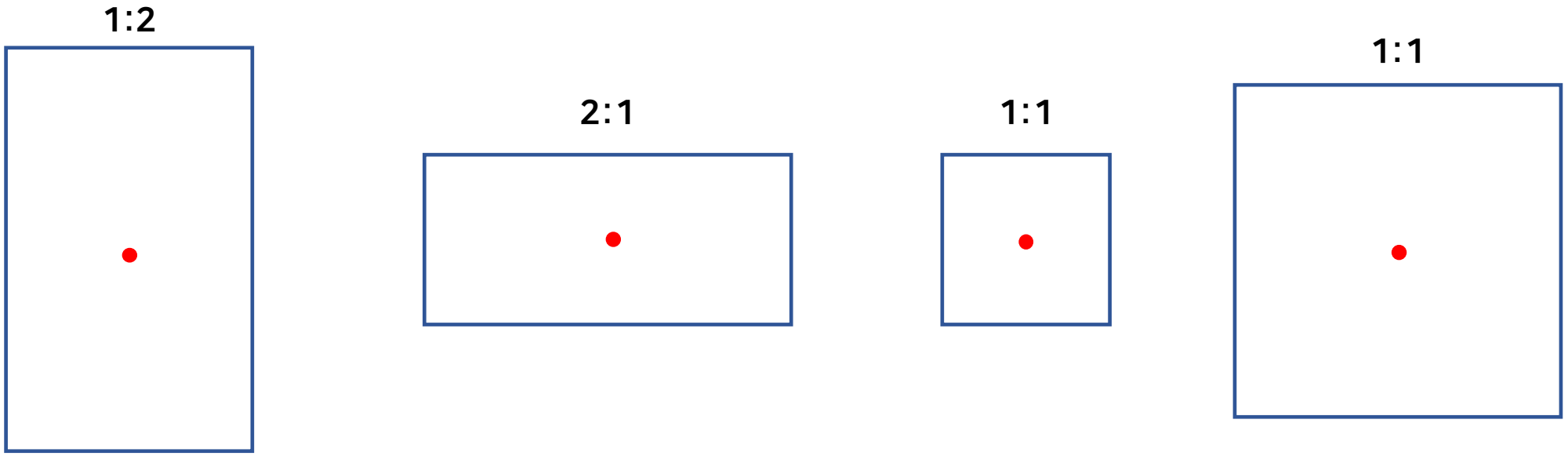
YOLO v2



YOLOv2 - Anchor box

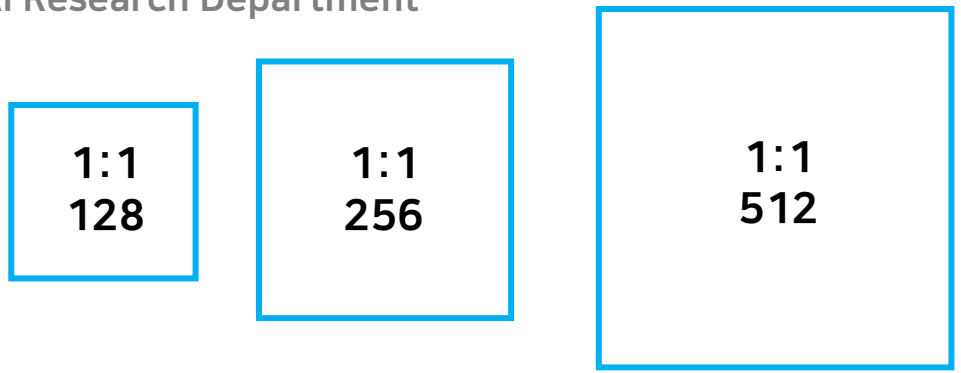
- **What is Anchor box?**

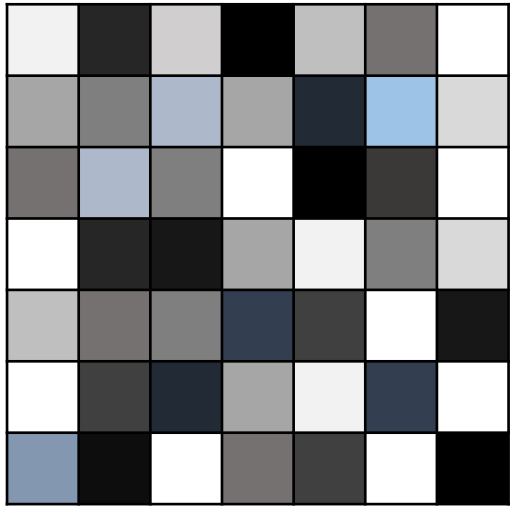
Anchor Box : 여러 개의 크기와 비율로 미리 정의된 형태를 가진 경계 박스(B-Box)로서 사용자가 개수와 형태를 임의로 지정 가능하다. YOLO B-box Regression은 B-box 좌표를 직접 예측하나, YOLO v2에서는 미리 지정된 Anchor box의 offset(size,ratio)을 예측하기 때문에 문제가 훨씬 간단해진다.



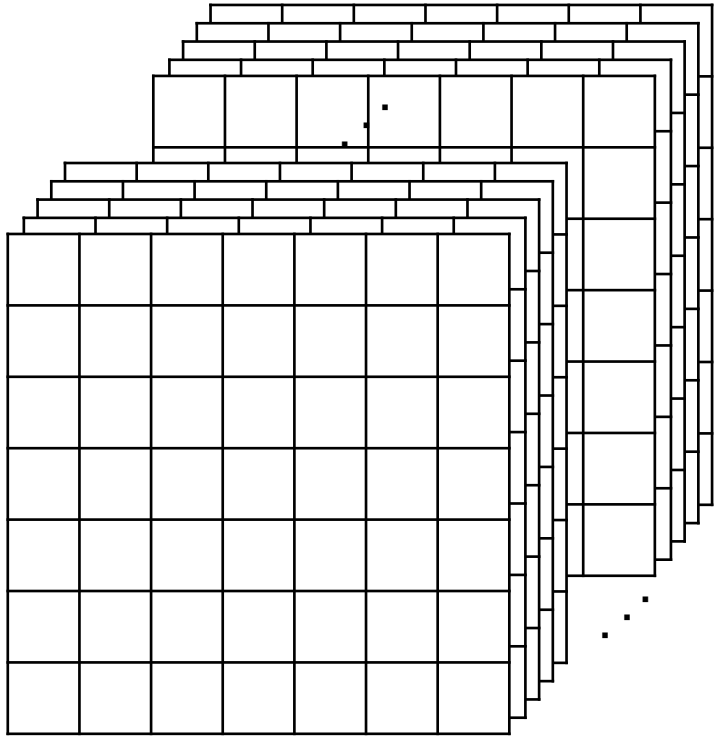
YOLO v1 & v2 prediction

PIAI Research Department



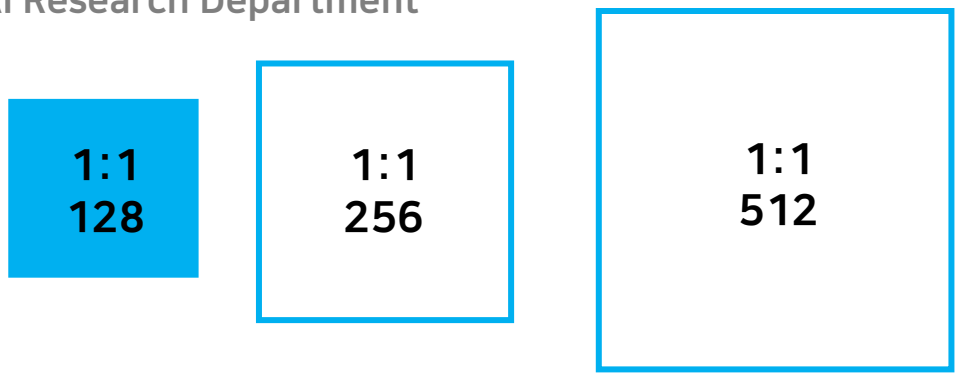


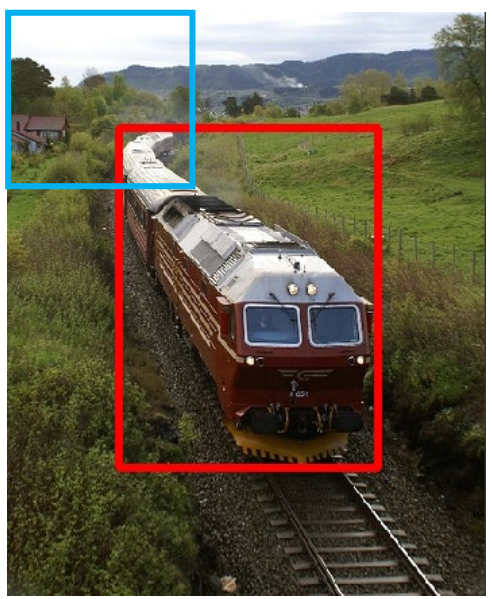
Feature Map

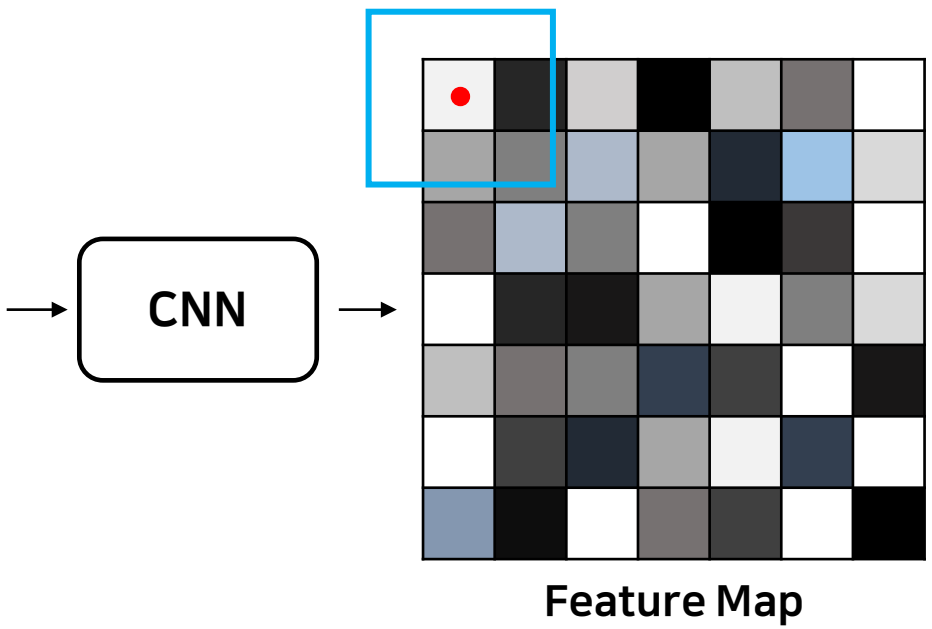


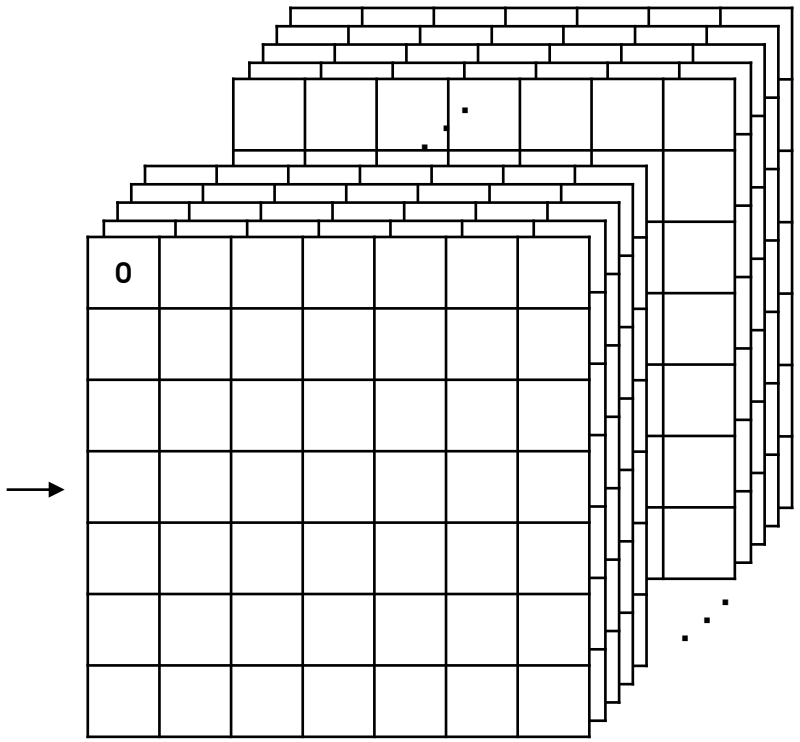
YOLO v1 & v2 prediction

PIAI Research Department



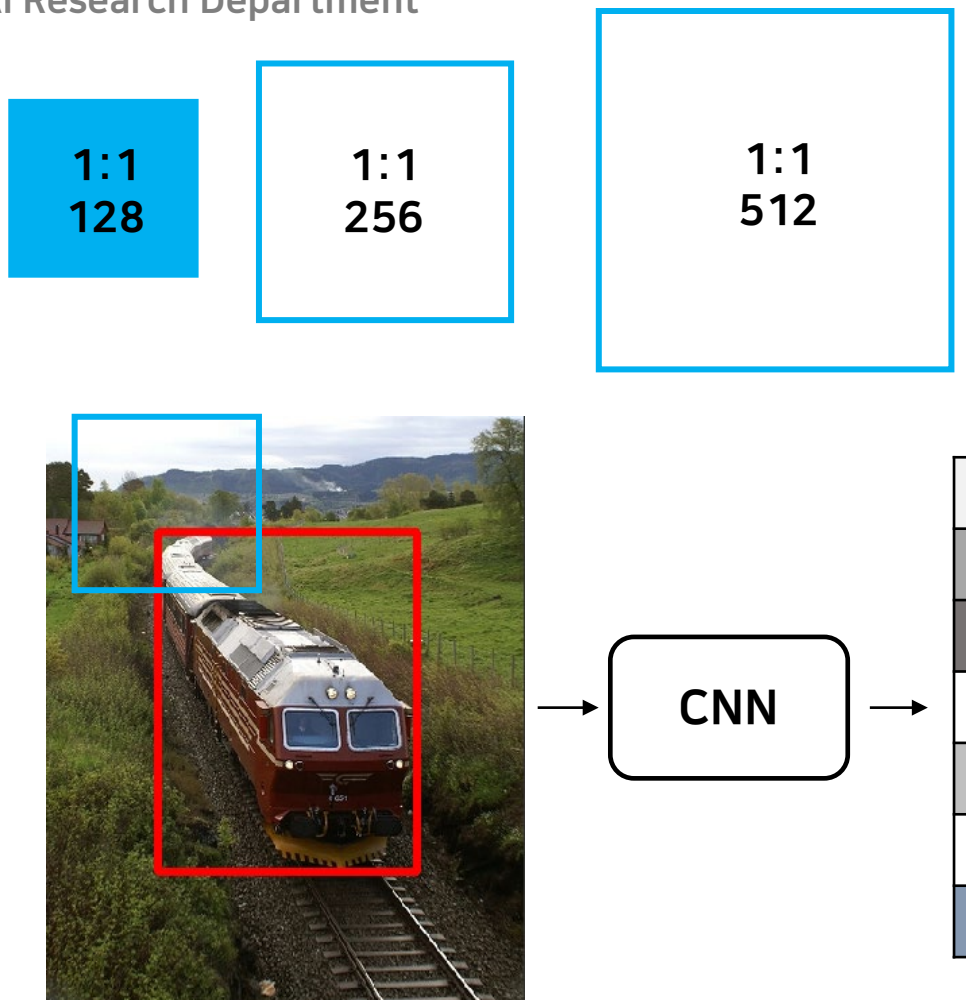






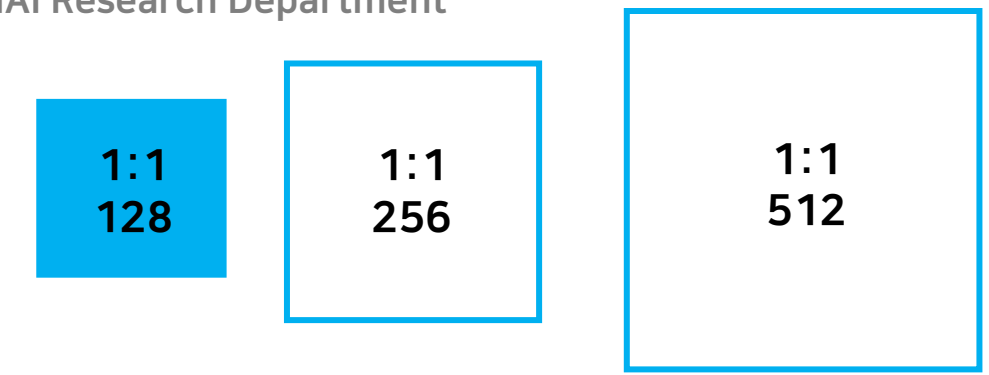
YOLO v1 & v2 prediction

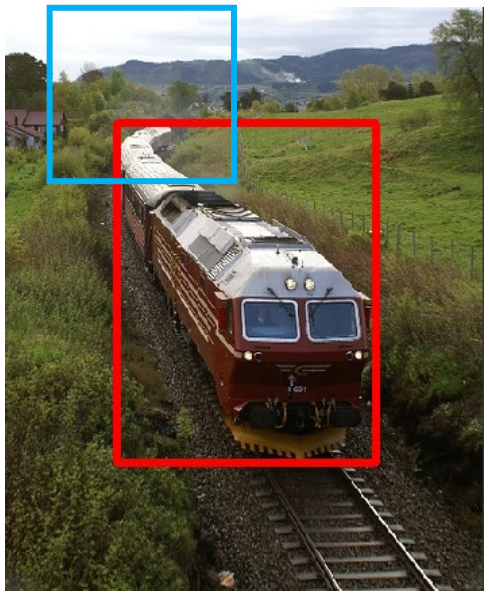
PIAI Research Department

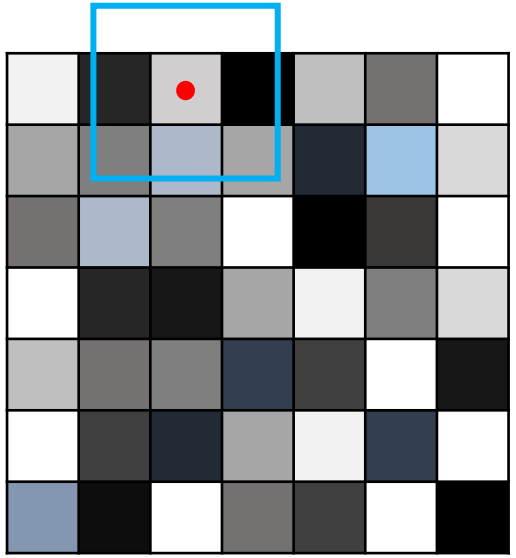


YOLO v1 & v2 prediction

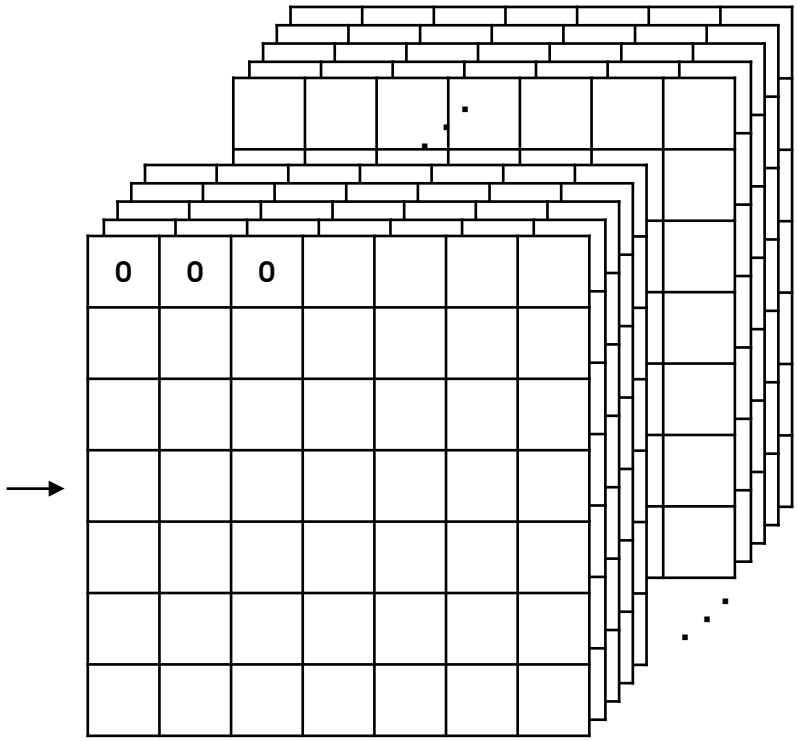
PIAI Research Department





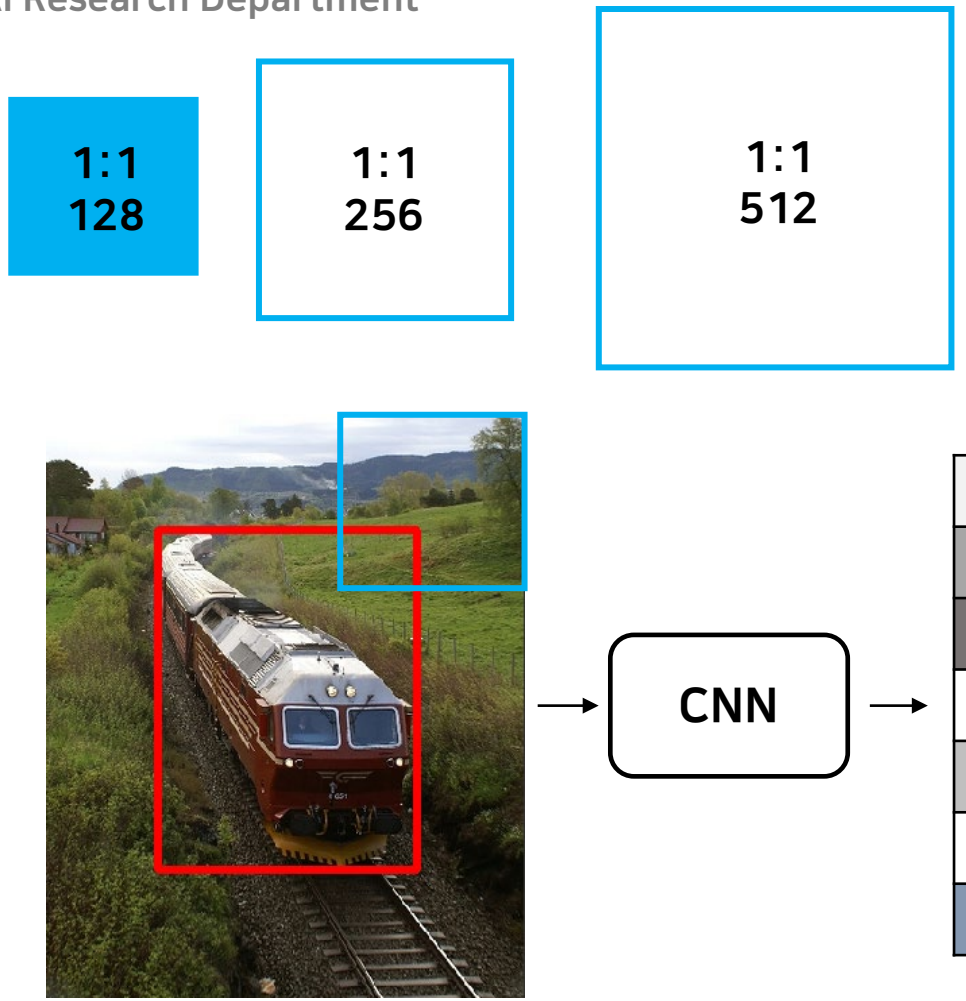


Feature Map



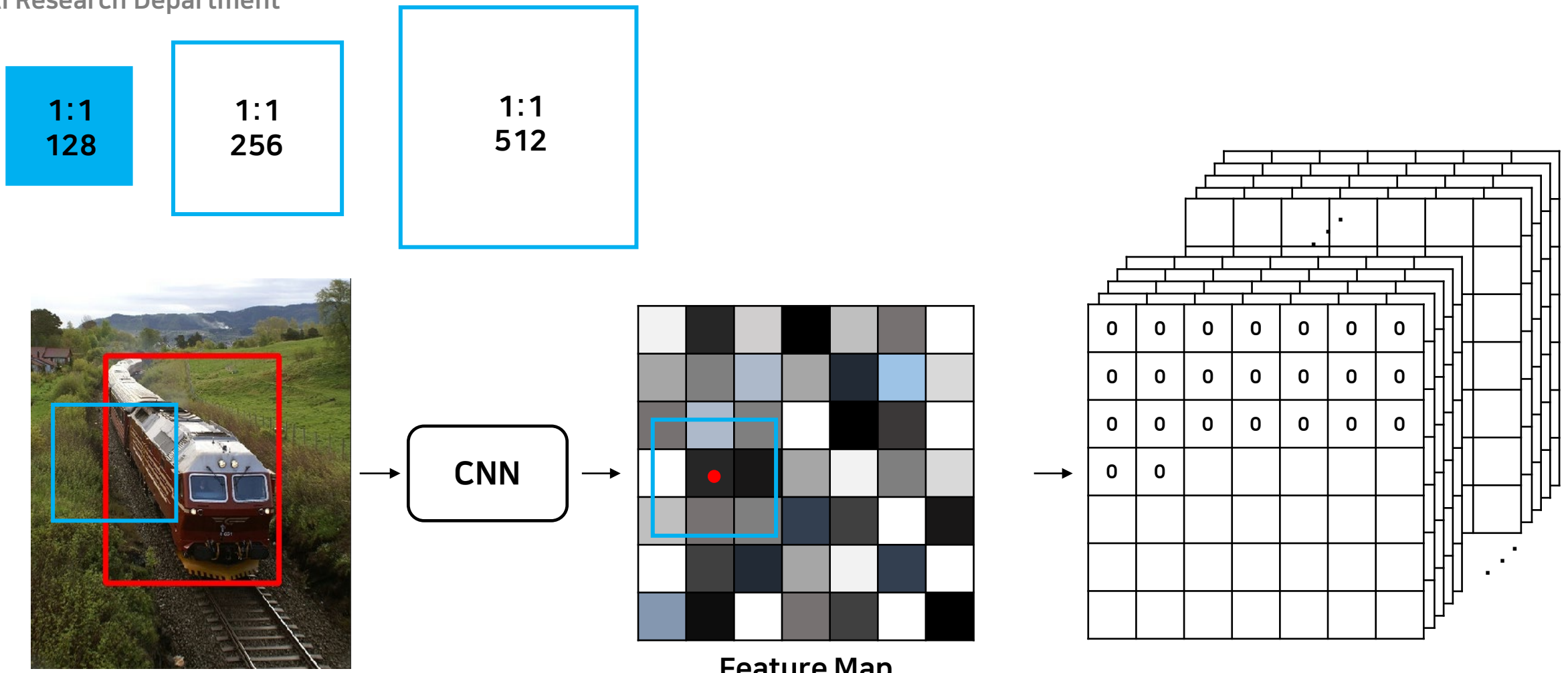
YOLO v1 & v2 prediction

PIAI Research Department



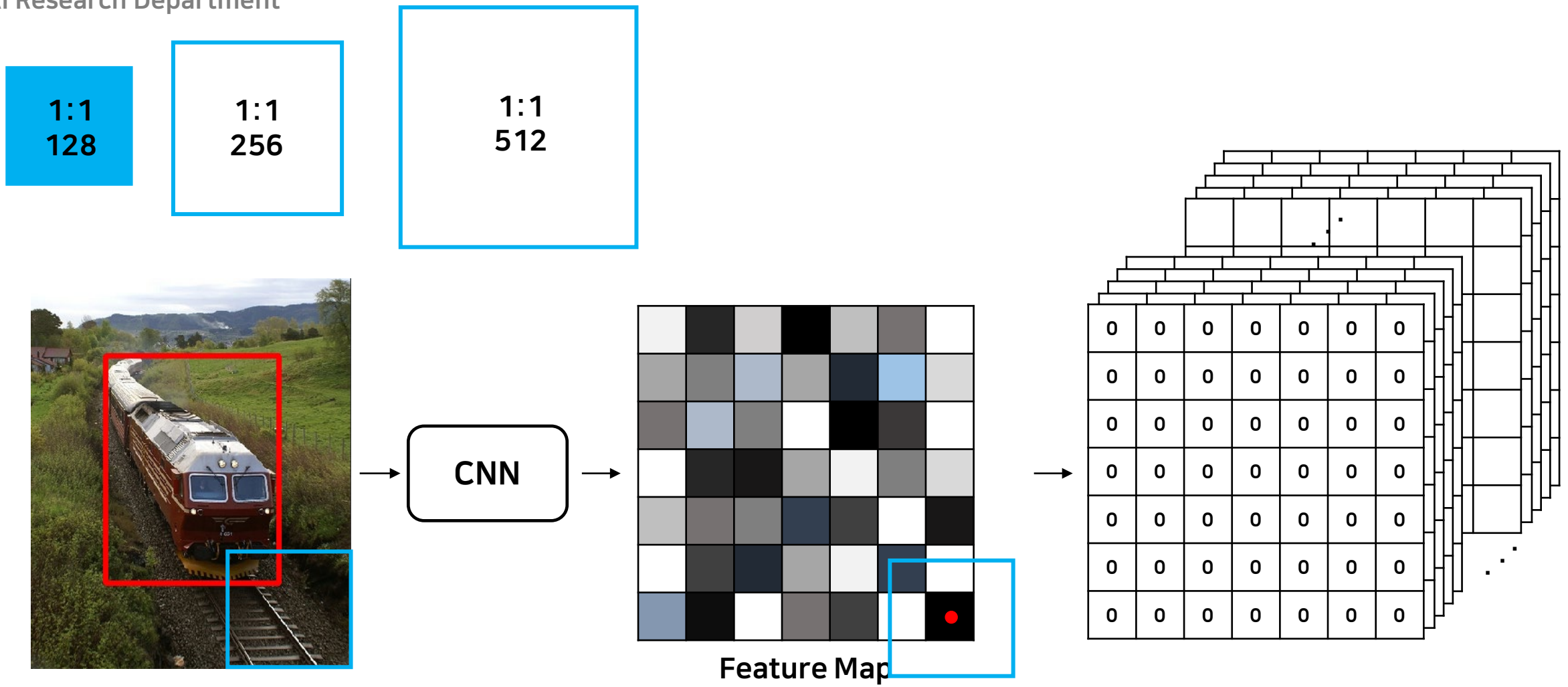
YOLO v1 & v2 prediction

PIAI Research Department



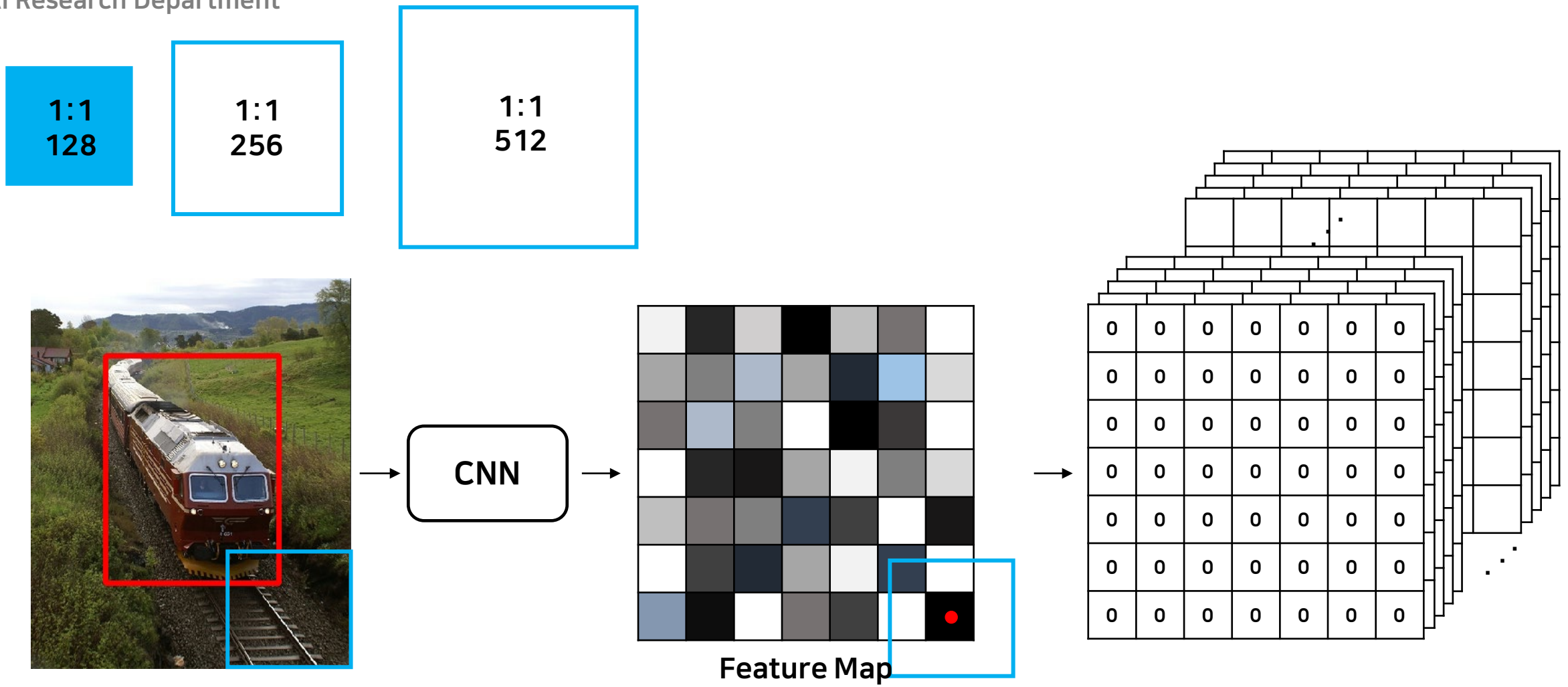
YOLO v1 & v2 prediction

PIAI Research Department



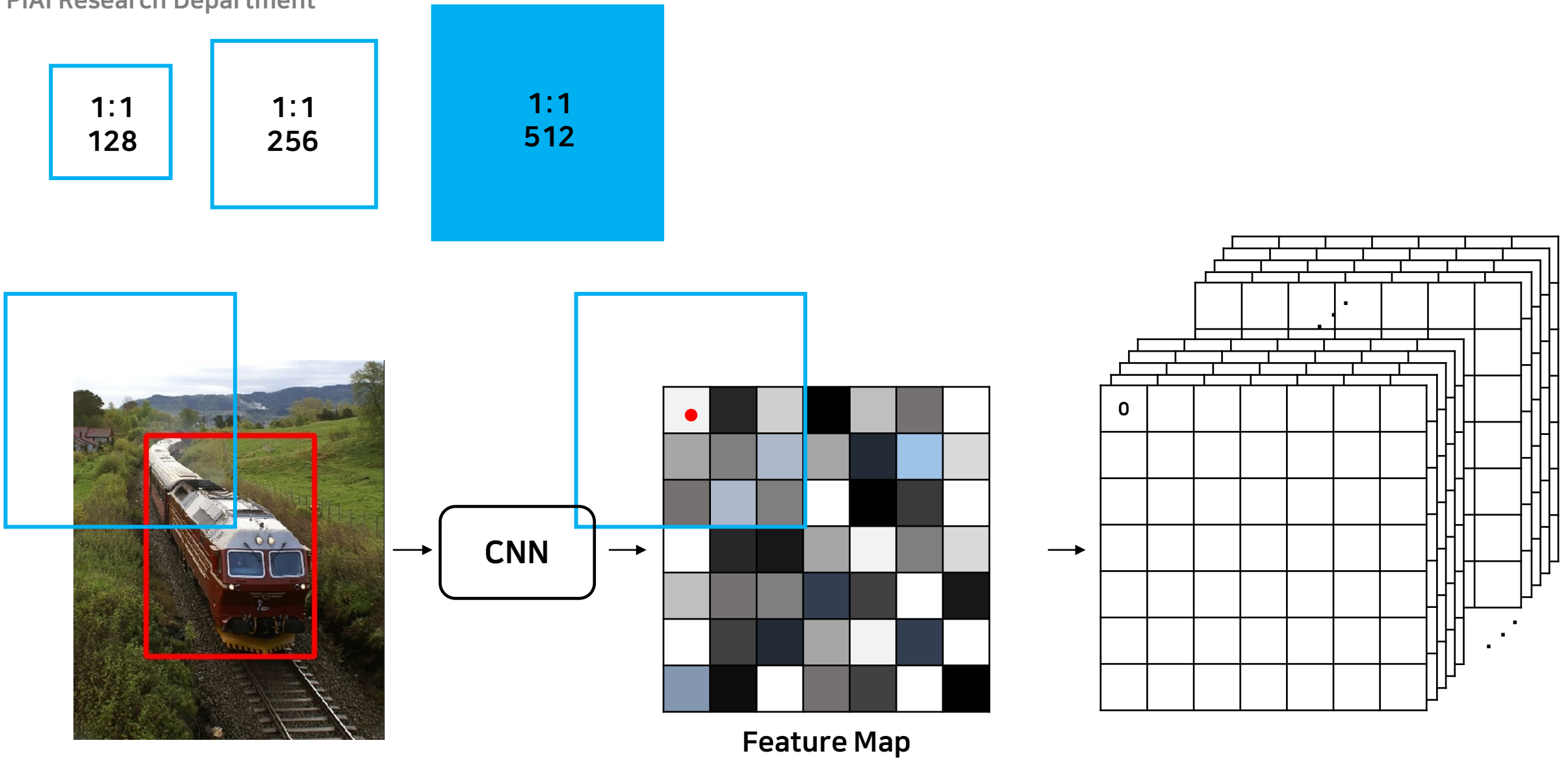
YOLO v1 & v2 prediction

PIAI Research Department



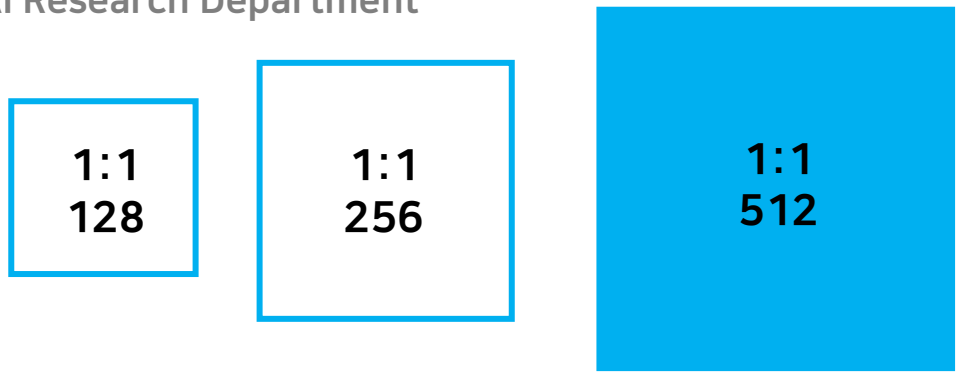
YOLO v1 & v2 prediction

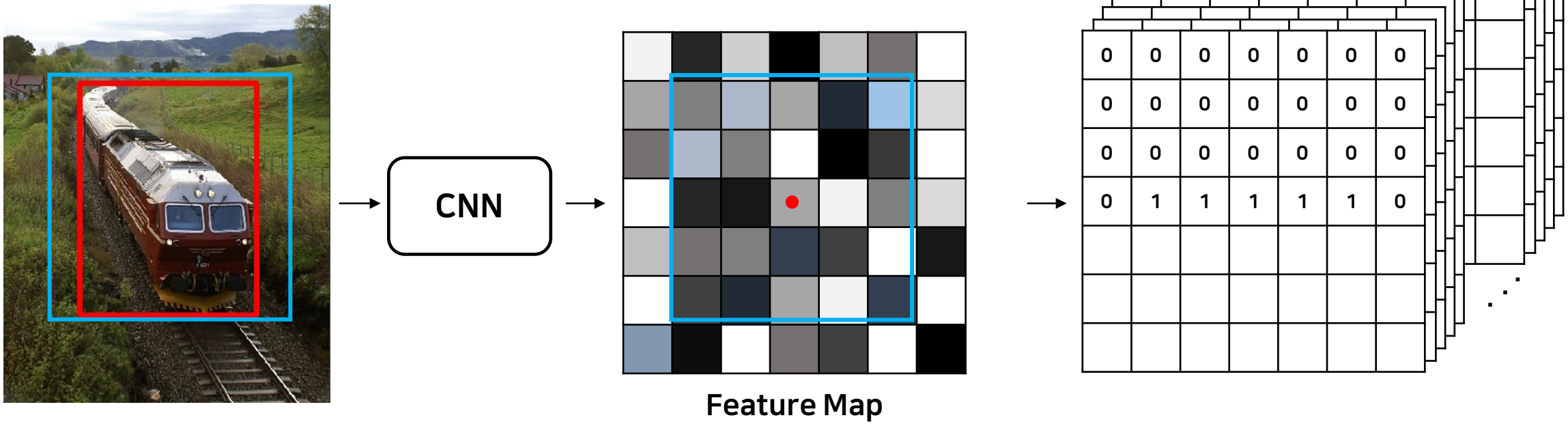
PIAI Research Department



YOLO v1 & v2 prediction

PIAI Research Department





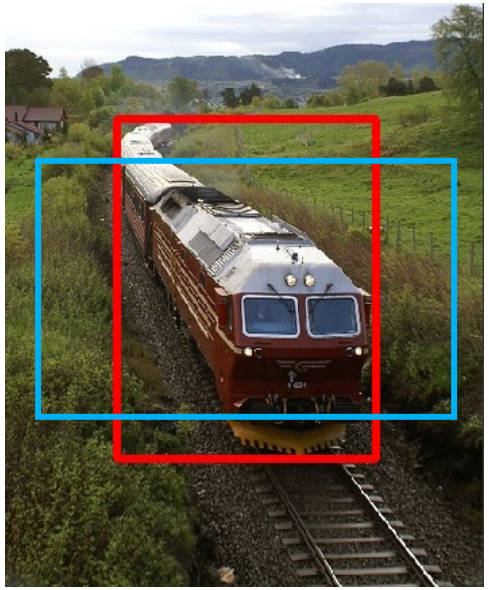
YOLO v1 & v2 prediction

PIAI Research Department

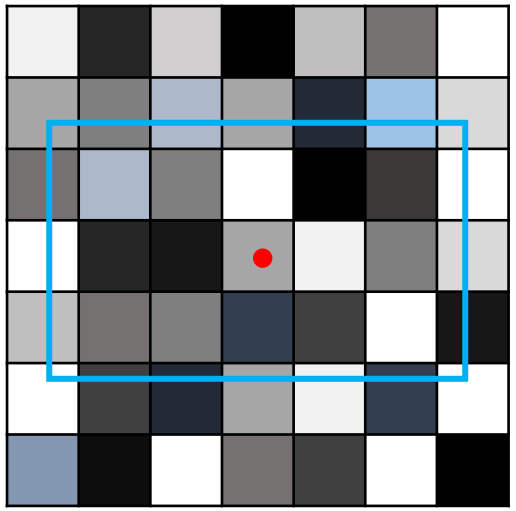
2:1
128

2:1
256

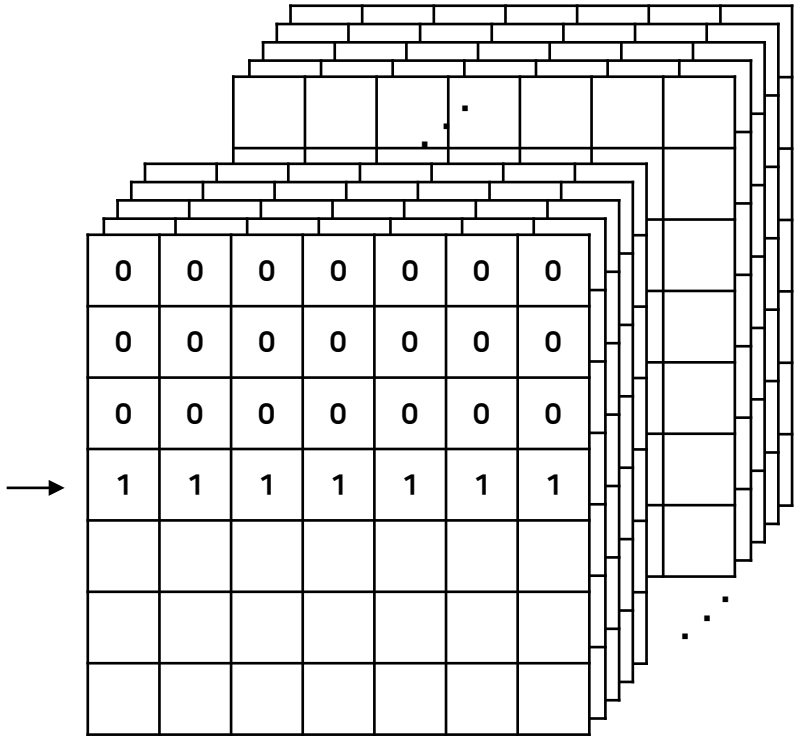
2:1
512



CNN

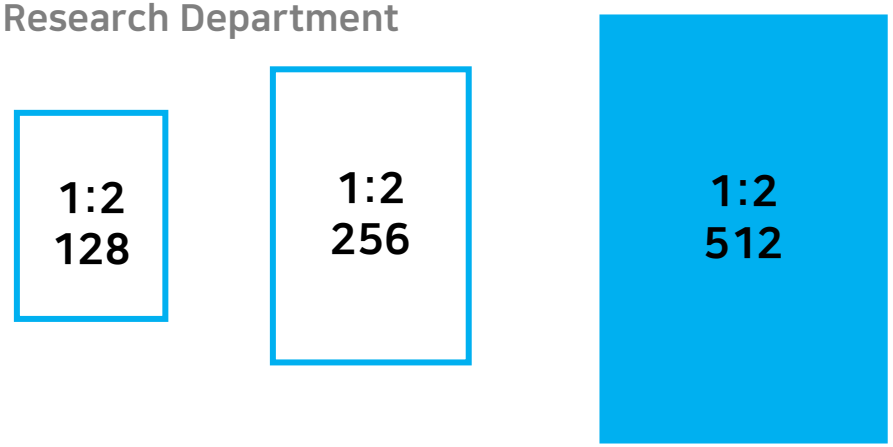


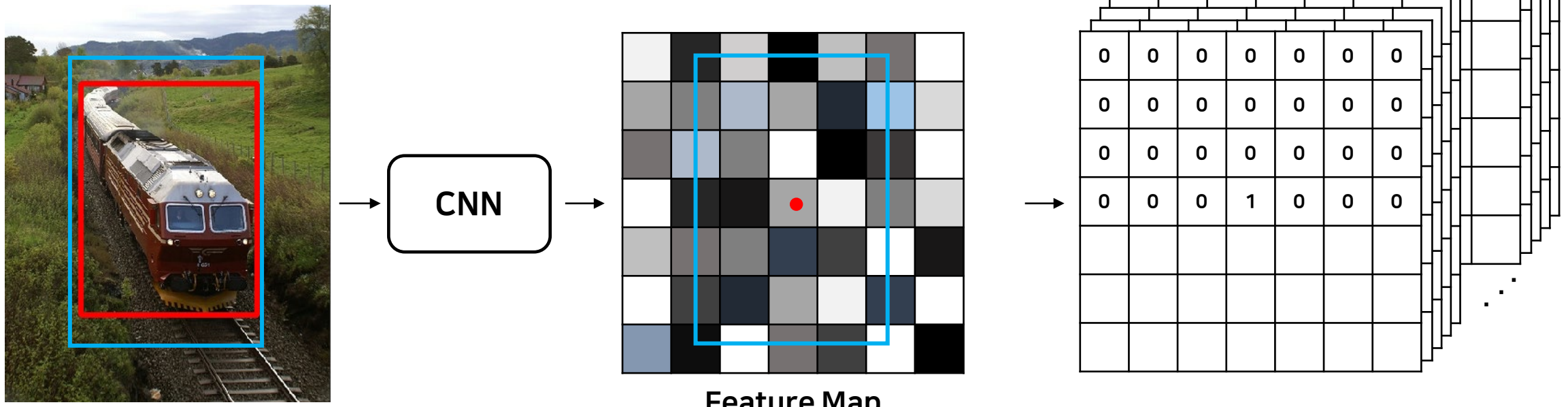
Feature Map



YOLO v1 & v2 prediction

PIAI Research Department





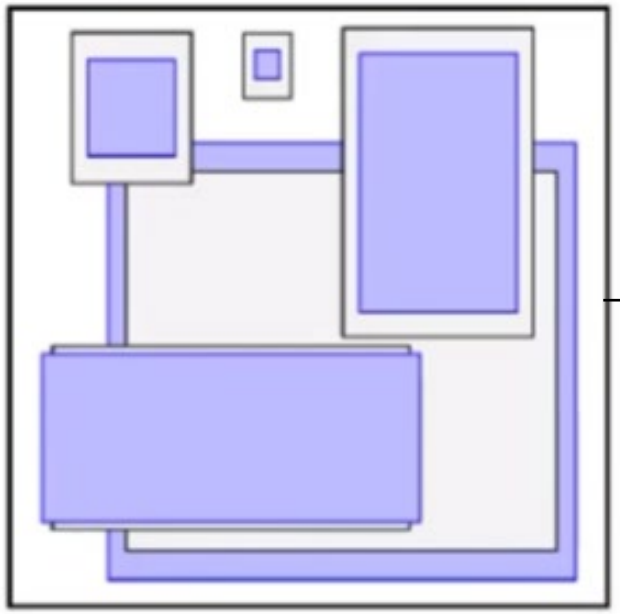
Feature Map

검출하고자 하는 사물의 크기 비율에 따라 적절한 Anchor Box를 설정하면 더 좋은 검출 성능을 이끌어낼 수 있음.

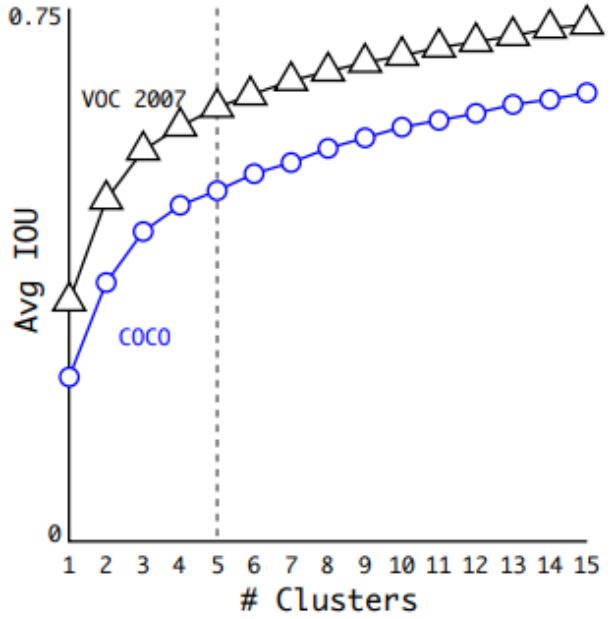
→ 내가 사용할 이미지 데이터 환경에 적합하게 앵커박스를 설정하는 방법은 없을까?

YOLOv2 - Anchor box

- Dynamic Anchor box



파란색 : COCO 데이터셋
회색 : Pascal VOC 데이터셋



- Faster R-CNN에서 Anchor Box의 사이즈와 비율을 미리 정하는 부분에 대한 문제를 제기
- YOLOv2 에서는 k-means Clustering을 통해 GT와 가장 유사한 Optimal Anchor box를 탐색함으로써 데이터셋 특성에 적합한 Anchor Box를 생성함.
- 실험 결과 Cluster 개수가 5개일 때 가장 적합했다고 알려짐

YOLOv2 - Anchor box

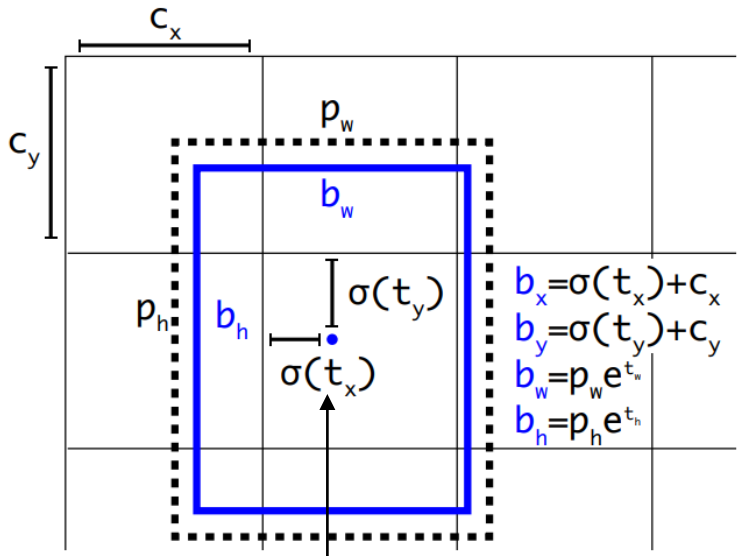
- **Constraint of Anchor box**

[YOLOv2 bbox]

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

[Faster R-CNN bbox]

$$\hat{G}_x = P_w d_x(P) + P_x$$
$$\hat{G}_y = P_h d_y(P) + P_y$$
$$\hat{G}_w = P_w \exp(d_w(P))$$
$$\hat{G}_h = P_h \exp(d_h(P)).$$



$Sigmoid(t_x)$
$Sigmoid(t_y)$

→ 0 ~ 1

YOLO v1 & v2 prediction

PIAI Research Department

YOLO V1

x1	y1	w1	h1	c1	x2	y2	w2	h2	c2	cls1	cls2	cls3	…											…	cls18	cls19	cls20
	bbox1						bbox2																				

$$S \times S \times (B * 5 + C) = 7 \times 7 \times (2 * 5 + 20)$$

YOLO V2

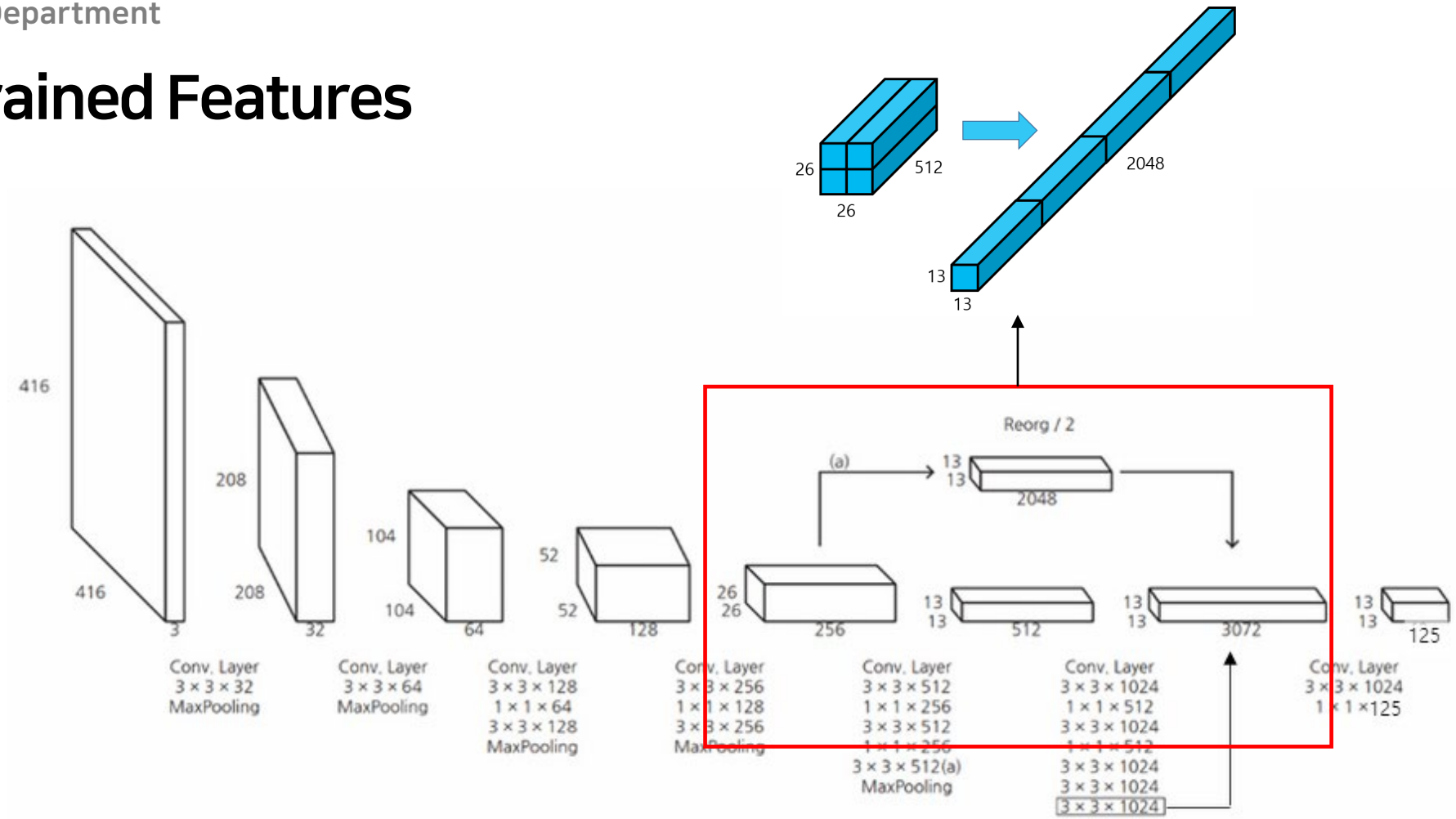
Anchor box1	x1	y1	w1	h1	c1	cls1	cls2	cls3	…												…	cls18	cls19	cls20
	x2	y2	w2	h2	c2	cls1	cls2	cls3	…												…	cls18	cls19	cls20
⋮	x3	y3	w3	h3	c3	cls1	cls2	cls3	…												…	cls18	cls19	cls20
	x4	y4	w4	h4	c4	cls1	cls2	cls3	…												…	cls18	cls19	cls20
Anchor box5	x5	y5	w5	h5	c5	cls1	cls2	cls3	…												…	cls18	cls19	cls20

$$S \times S \times (B * (5 + C)) = 13 \times 13 \times (5 * (5 + 20))$$

YOLO V2 – Fine grained Features

PIAI Research Department

- **Fine grained Features**

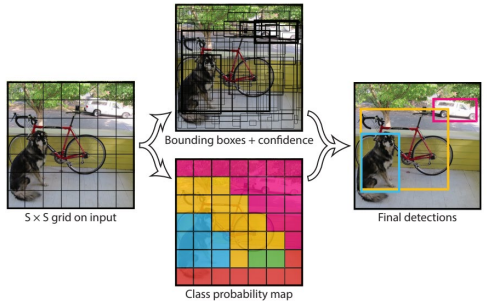


- 앞 Convolution Layer의 High Resolution Feature를 뒷 단의 Convolution Layer의 Low Resolution Feature에 추가
- High Resolution Feature는 작은 객체에 대한 정보를 함축하고 있는 것으로 알려짐 → 작은 객체 검출에 강인

YOLO Versions

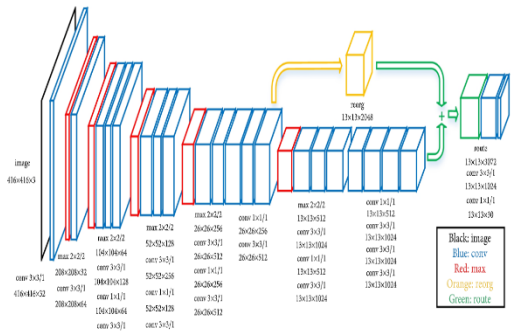
PIAI Research Department

YOLO V1



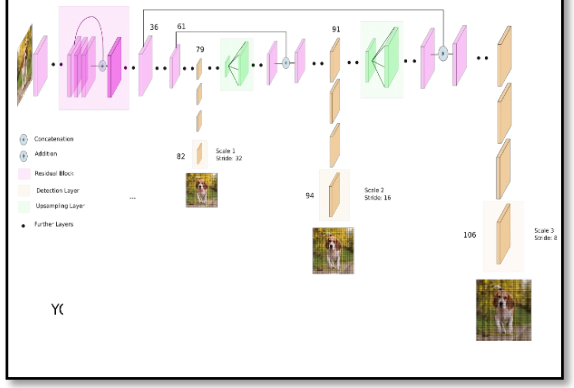
- Anchor Box
- Darknet19

YOLO V2



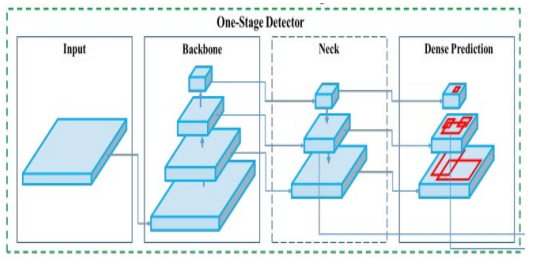
- Residual Block(Darknet53)
- Prediction across Scales
- Backbone–Neck–Head Architecture
- FPN(Neck)

YOLO V3



- Backbone : CSP
- Neck : SPP + PAN
- Head : SAM
- DropBlock
- Mish Activation

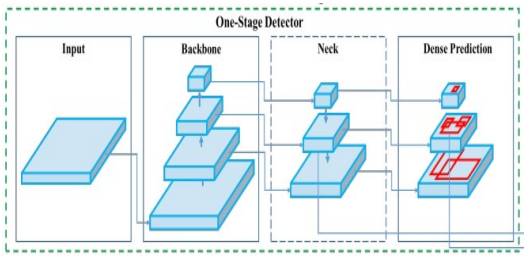
YOLO V4



- Pytorch Implementation

YOLO V5

PyTorch



PIAI
인공지능연구원
POSTECH Institute of Artificial Intelligence

가치창출대학
POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Object Detection Metrics

Dahyun, Kim

Confusion Matrix

PIAI Research Department

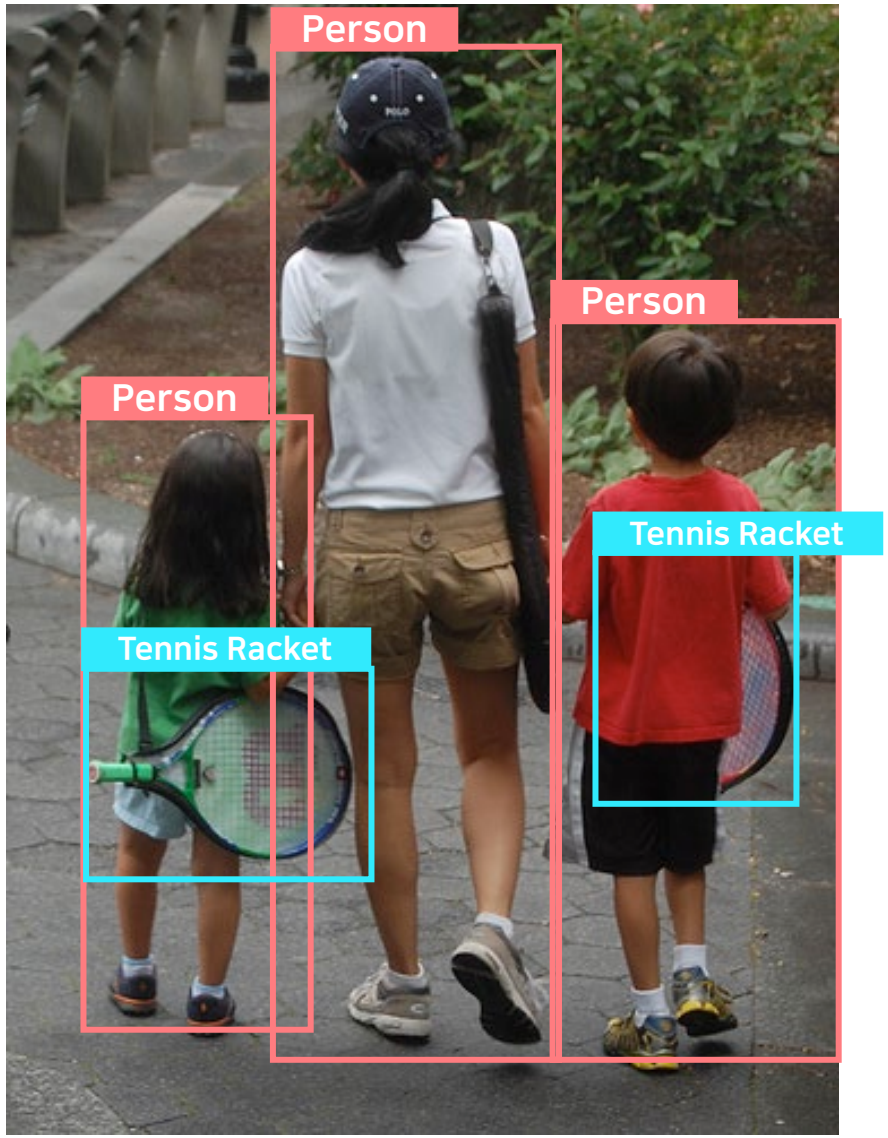
		Predicted Value	
		Positives	Negatives
Actual Value	Positives	TP (True positive)	FN (False Negative)
	Negatives	FP (False Positive)	TN (True Negative)

항목	실제	예측	설명	의미	판단 기준
TP	양성	양성	검출되어야 할 영역에 잘 검출됨	올바른 탐지	IoU >= Threshold
FP	음성	음성	객체가 없는 영역에 검출됨	오탐지	IoU <= Threshold
FN	양성	음성	객체가 있는 영역에 검출이 되지 않음	미탐지	GT 에는 Bbox 존재하나 미탐지
TN	음성	음성	객체가 없는 영역에 검출이 되지 않음	사용 X	

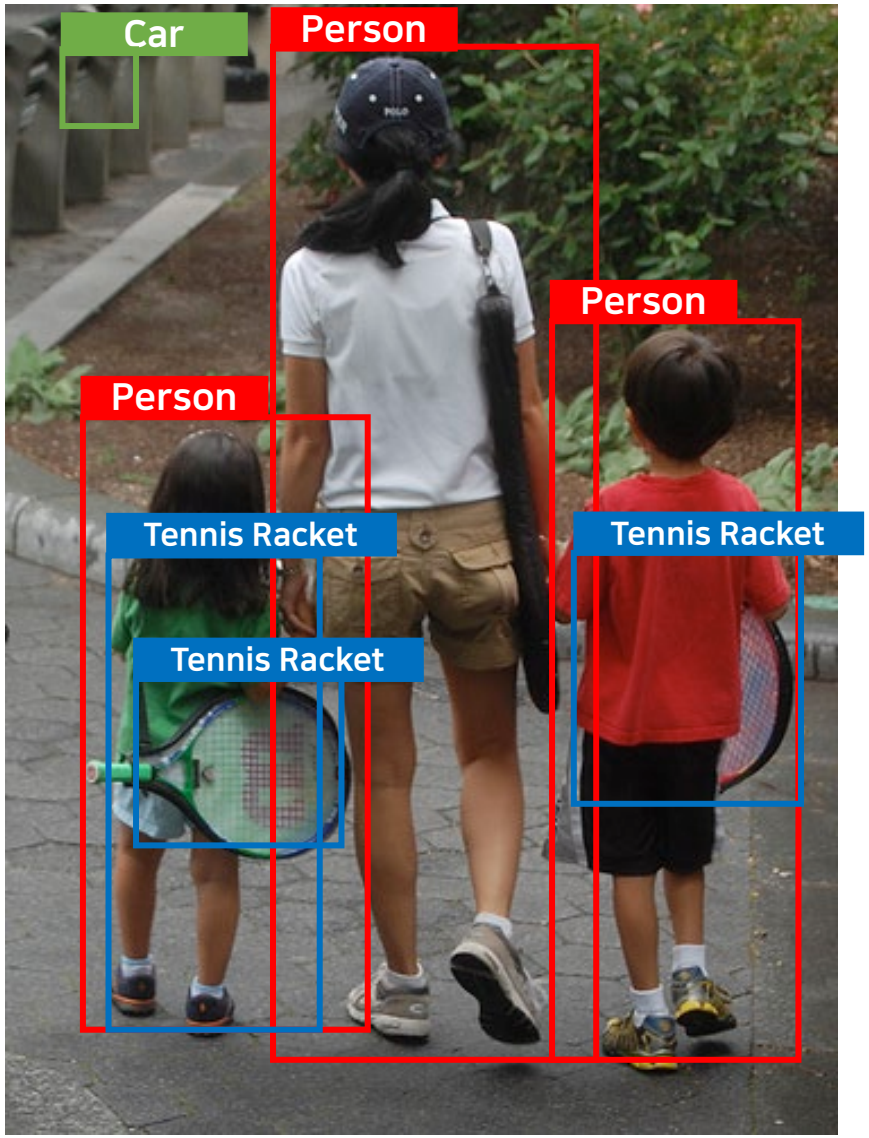
OD Metrics – Ground Truth & Predict

PIAI Research Department

Ground Truth



Predict

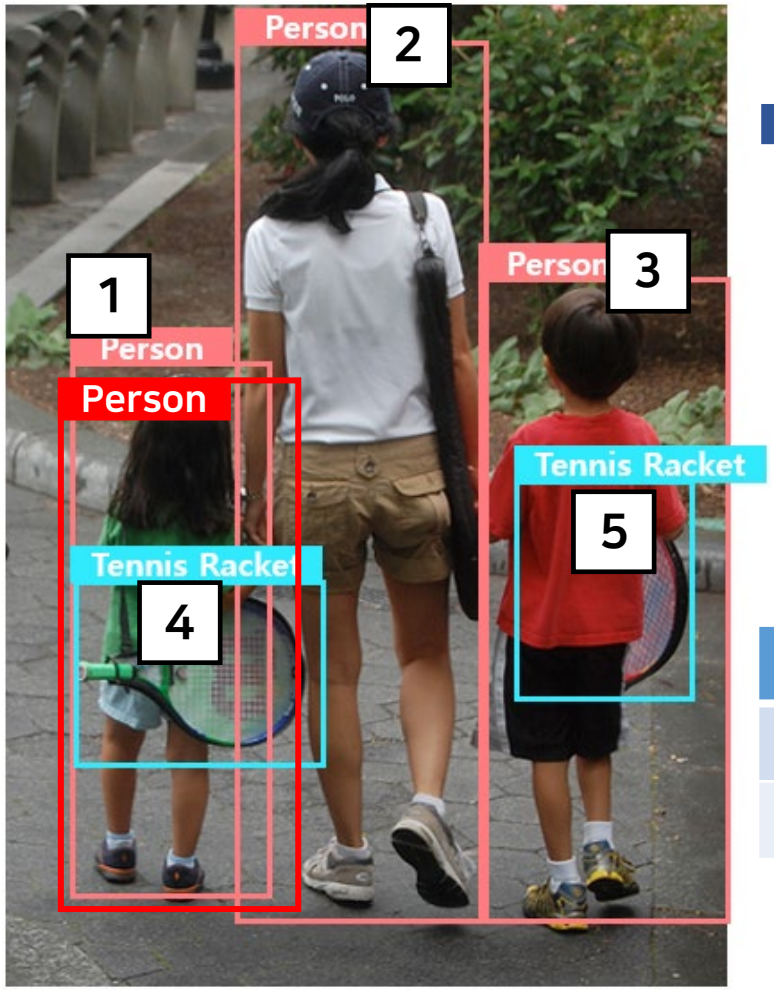


Counting TP/FP per Object (Person Class)

PIAI Research Department

GT Person — Predict Person

GT Tennis Racket — Predict Tennis Racket



모든 GT Bbox와 IoU 계산

No	GT Class	Pd Class	IoU
1	Person		0.89
2	Person	Person	0.15
3	Person		0
4	Tennis Racket		0.3
5	Tennis Racket		0

IoU>=0.5

GT Class == Pd Class

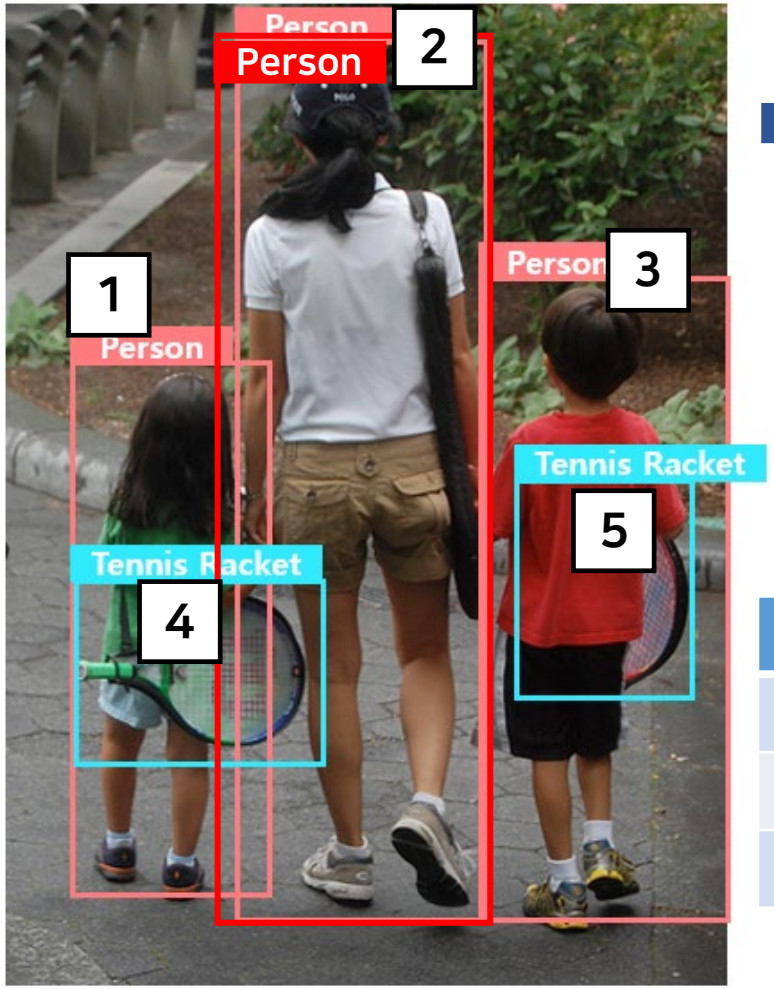
Person Class						
No	Predicted Class	Confidence	IoU	TP/FP	Acc TP	ACC FP
1	Person	0.89	0.89	TP	1	0

Counting TP/FP per Object (Person Class)

PIAI Research Department

GT Person — Predict Person

GT Tennis Racket — Predict Tennis Racket



모든 GT Bbox와 IoU 계산

No	GT Class	Pd Class	IoU
1	Person		0.23
2	Person		0.85
3	Person	Person	0.05
4	Tennis Racket		0.21
5	Tennis Racket		0

IoU>=0.5

GT Class == Pd Class

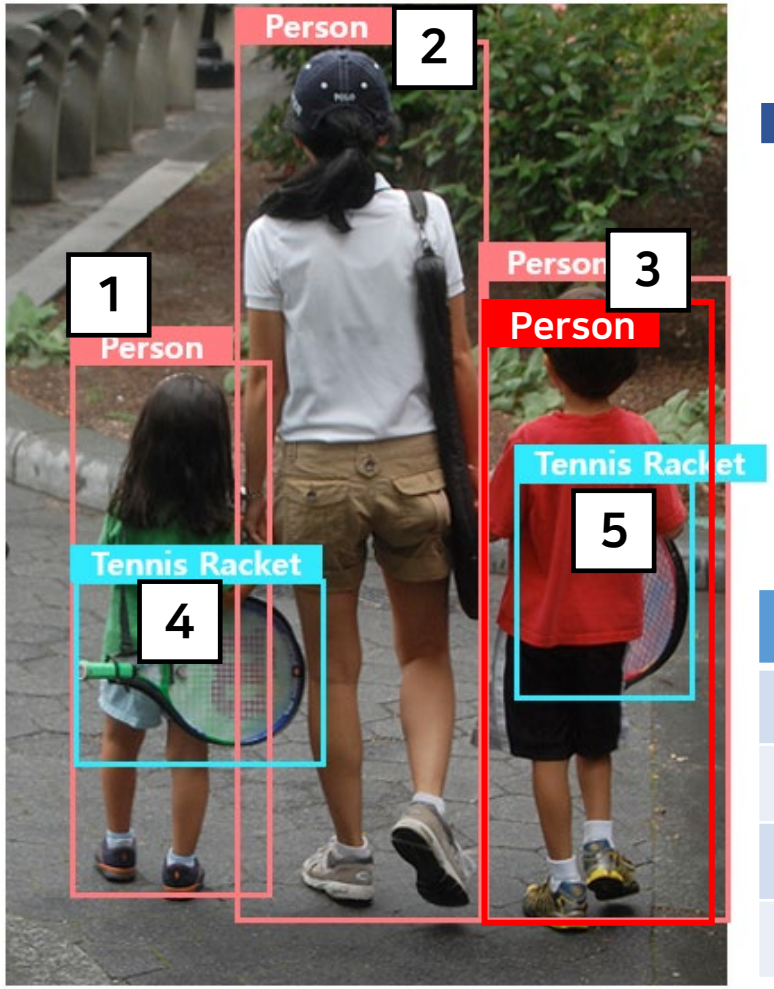
Person Class						
No	Predicted Class	Confidence	IoU	TP/FP	Acc TP	ACC FP
1	Person	0.89	0.89	TP	1	0
2	Person	0.85	0.89	TP	2	0

Counting TP/FP per Object (Person Class)

PIAI Research Department

GT Person — Predict Person

GT Tennis Racket — Predict Tennis Racket



모든 GT Bbox와 IoU 계산

No	GT Class	Pd Class	IoU
1	Person		0
2	Person		0.01
3	Person	Person	0.78
4	Tennis Racket		0
5	Tennis Racket		0.25

IoU>=0.5

GT Class == Pd Class

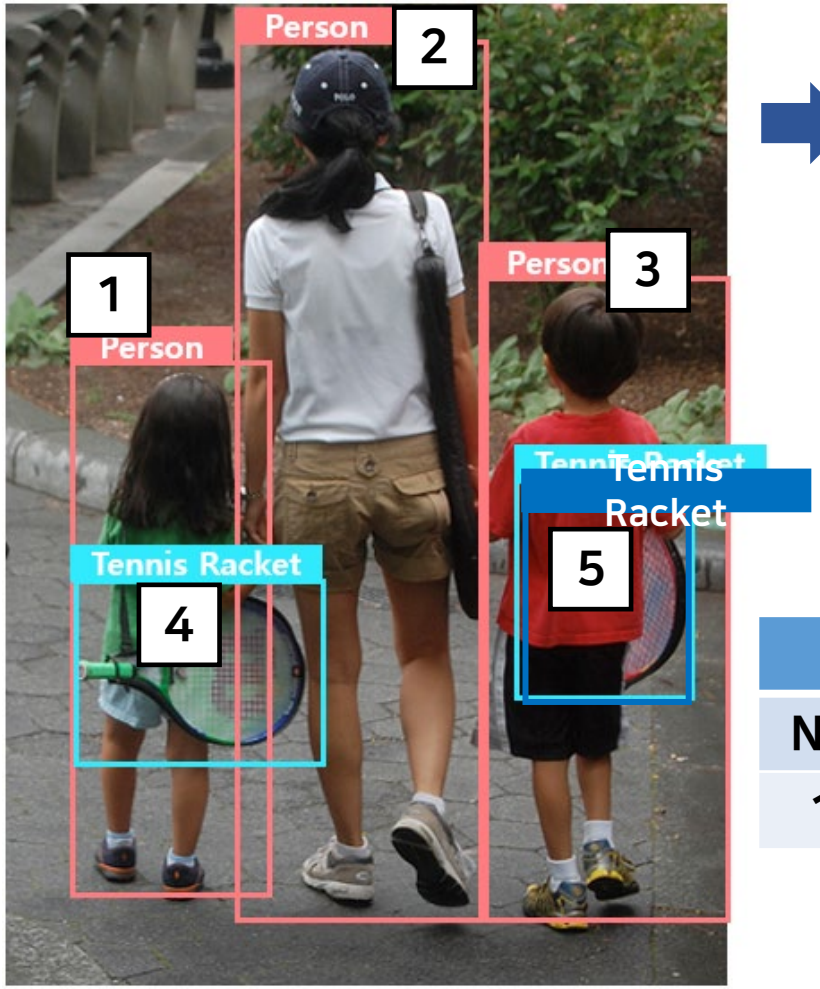
Person Class						
No	Predicted Class	Confidence	IoU	TP/FP	Acc TP	ACC FP
1	Person	0.89	0.89	TP	1	0
2	Person	0.85	0.89	TP	2	0
3	Person	0.56	0.78	TP	3	0

Counting TP/FP per Object (Tennis Racket Class)

PIAI Research Department

GT Person — Predict Person

GT Tennis Racket — Predict Tennis Racket



모든 GT Bbox와 IoU 계산

No	GT Class	Pd Class	IoU
1	Person	Tennis Racket	0
2	Person		0
3	Person		0.21
4	Tennis Racket		0
5	Tennis Racket		0.97

IoU>=0.5

GT Class == Pd Class

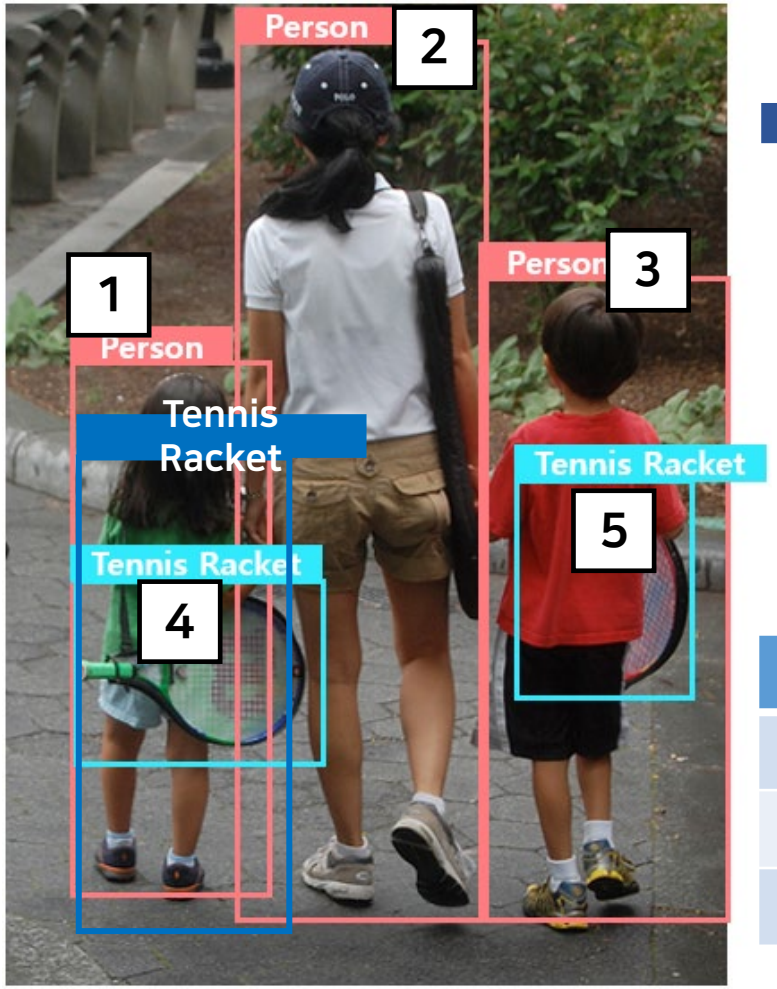
Tennis Racket Class						
No	Predicted Class	Confidence	IoU	TP/FP	Acc TP	ACC FP
1	Tennis Racket	0.77	0.97	TP	1	0

Counting TP/FP per Object (Tennis Racket Class)

PIAI Research Department

GT Person — Predict Person

GT Tennis Racket — Predict Tennis Racket



모든 GT Bbox와 IoU 계산

No	GT Class	Pd Class	IoU
1	Person		0.77
2	Person	Tennis Racket	0.21
3	Person		0
4	Tennis Racket		0.53
5	Tennis Racket		0.25

IoU>=0.5

No.1 IoU >= No.2 IoU

GT Class != Pd Class

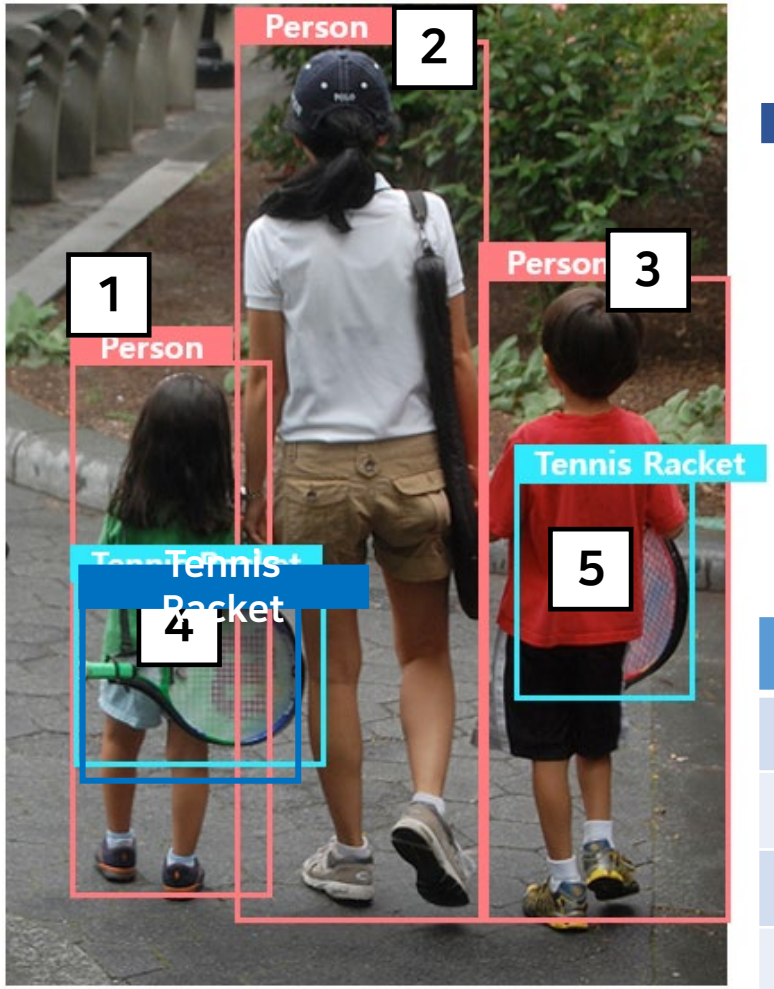
Tennis Racket Class						
No	Predicted Class	Confidence	IoU	TP/FP	Acc TP	ACC FP
1	Tennis Racket	0.77	0.97	TP	1	0
2	Tennis Racket	0.38	0.77	FP	0	1

Counting TP/FP per Object (Tennis Racket Class)

PIAI Research Department

GT Person — Predict Person

GT Tennis Racket — Predict Tennis Racket



모든 GT Bbox와 IoU 계산

No	GT Class	Pd Class	IoU
1	Person	Tennis Racket	0.3
2	Person		0.12
3	Person		0
4	Tennis Racket		0.84
5	Tennis Racket		0

IoU>=0.5

GT Class == Pd Class

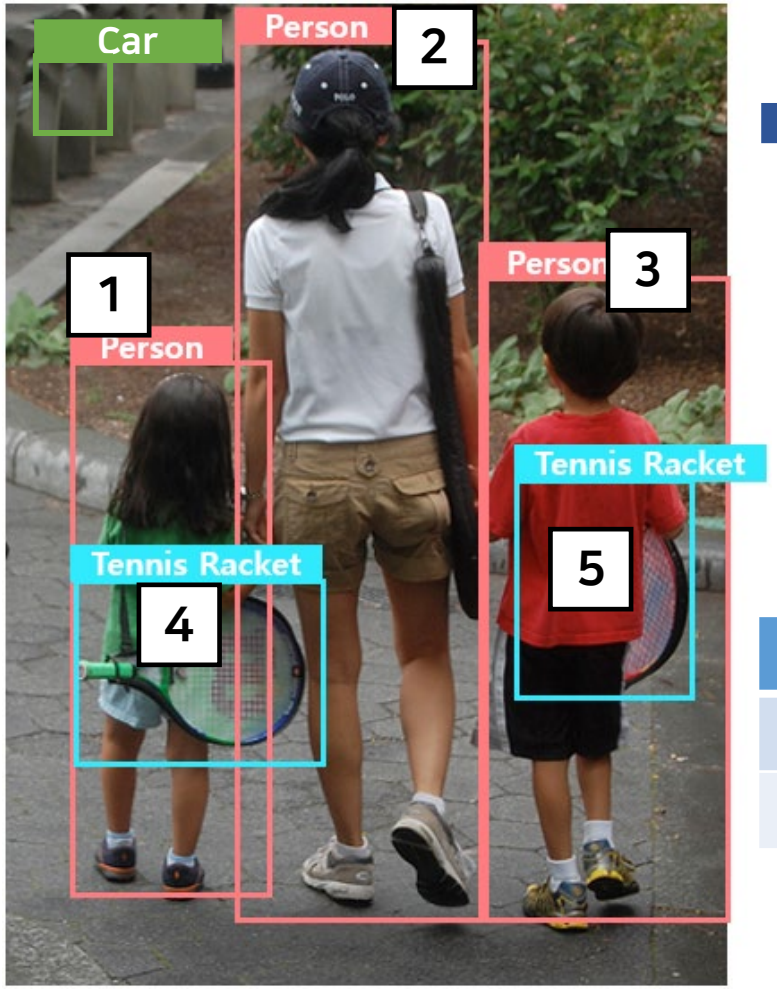
Tennis Racket Class						
No	Predicted Class	Confidence	IoU	TP/FP	Acc TP	ACC FP
1	Tennis Racket	0.77	0.97	TP	1	0
2	Tennis Racket	0.38	0.77	FP	1	1
3	Tennis Racket	0.36	0.84	TP	2	1

Counting TP/FP per Object (Car Class)

PIAI Research Department

GT Person — Predict Person

GT Tennis Racket — Predict Tennis Racket



모든 GT Bbox와 IoU 계산

No	GT Class	Pd Class	IoU
1	Person	Tennis Racket	0
2	Person		0
3	Person		0
4	Tennis Racket		0
5	Tennis Racket		0

Car Class						
No	Predicted Class	Confidence	IoU	TP/FP	Acc TP	ACC FP
1	Car	0.55	0	FP	0	1

Recall / Precision

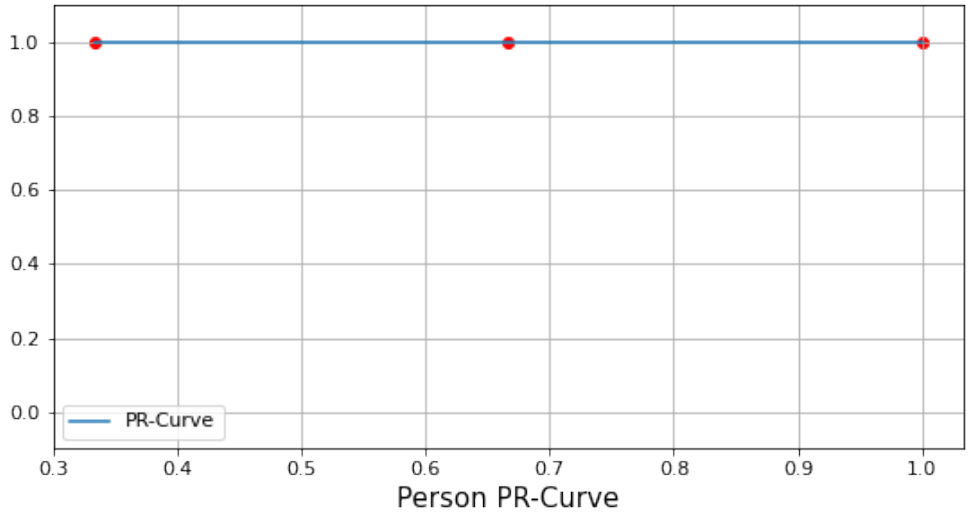
PIAI Research Department

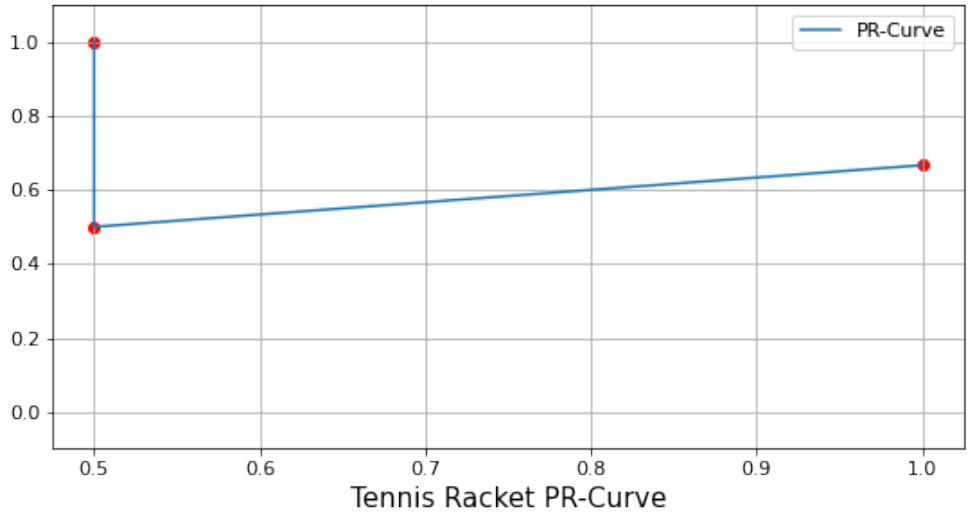
$$Recall = \frac{TP}{TP + FN} = \frac{TP}{All\ Ground\ Truths} \qquad Precision = \frac{TP}{TP + FP} = \frac{TP}{All\ Detections}$$

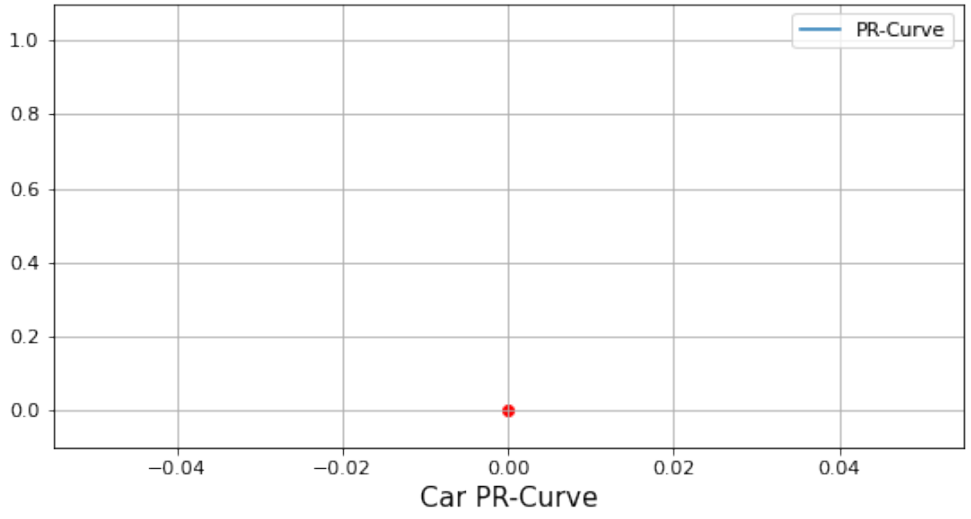
Person Class								
No	Predicted Class	Confidence	IoU	TP/FP	ACC TP	ACC FP	Recall	Precision
1	Person	0.89	0.89	TP	1	0	0.33	1.0
2	Person	0.85	0.89	TP	2	0	0.66	1.0
3	Person	0.56	0.78	TP	3	0	1.0	1.0
Tennis Racket Class								
No	Predicted Class	Confidence	IoU	TP/FP	ACC TP	ACC FP	Recall	Precision
1	Tennis Racket	0.77	0.97	TP	1	0	0.5	1.0
2	Tennis Racket	0.38	0.77	FP	1	1	0.5	0.5
3	Tennis Racket	0.36	0.84	TP	2	1	1.0	0.66
Car Class								
No	Predicted Class	Confidence	IoU	TP/FP	ACC TP	ACC FP	Recall	Precision
1	Car	0.55	0	FP	0	1	0	0

P-R Curve

PIAI Research Department

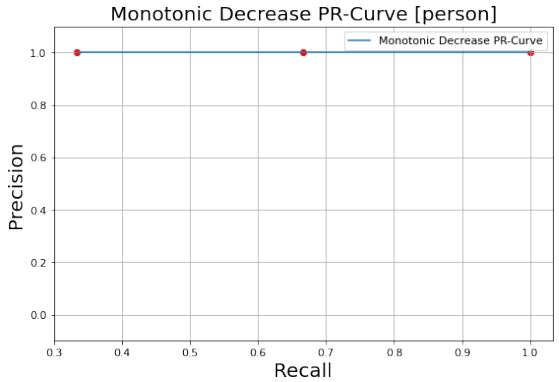


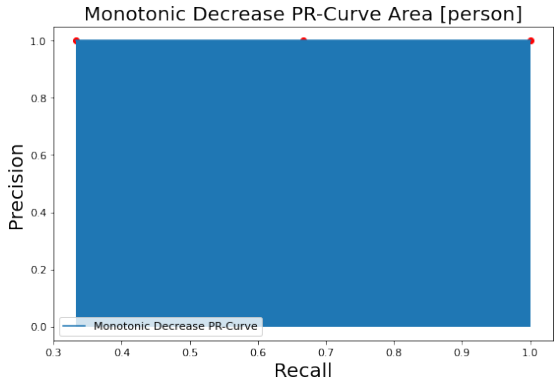


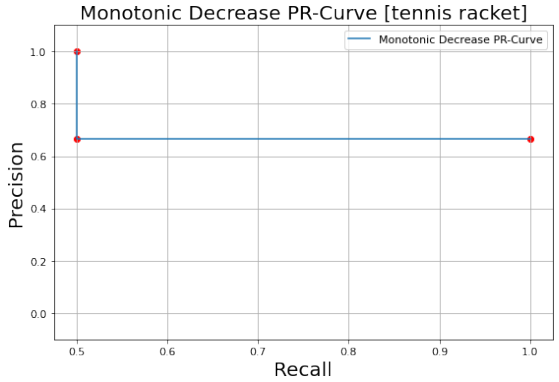


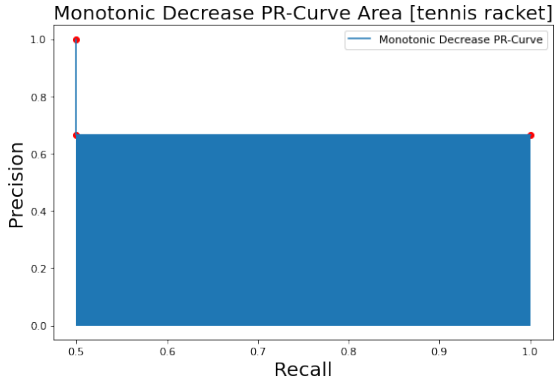
AP & mAP

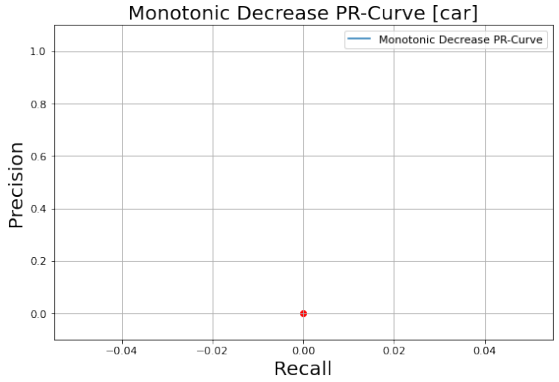
PIAI Research Department

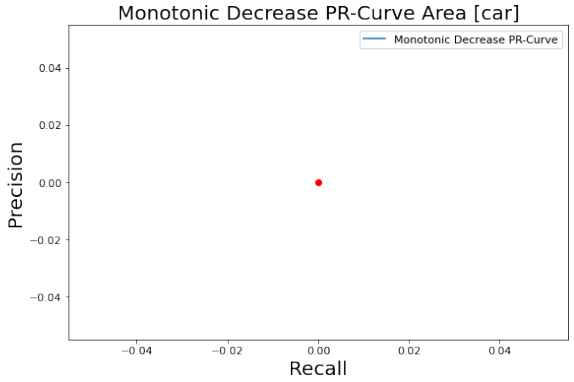












Person AP : 0.67

Tennis Racket AP : 0.33

Car AP : 0

평균

mAP : 33.33%

CV Homework 1

PIAI Research Department

- Ozgenel 데이터 셋의 학습 / 평가 / 분석
- 제출 파일 : yolov5 폴더, 보고서 파일을 함께 압축한 zip 파일 ex) HW_김다현.zip
  - yolov5 폴더 내의 모든 파일을 함께 압축하여 제출할 것
  - 수정한 모든 내용이 포함되어야 함
- 보고서(80%) – mAP 성능(20%)을 종합적으로 평가
  - 테스트 데이터셋에 대한 mAP@.5 점수는 0.992 이상 권장
- 보고서 내용
  - train 종료 시점의 터미널 화면, test 실행한 터미널 화면 캡처하여 첨부
  - train 종료 후 생성되는 PR_curve.png, results.png 첨부
  - 성능 개선을 위해 실험한 내용 및 학습 전략
  - 성능 분석 : 잘못된 검출 케이스 확인 / 개선할 수 있는 방향 제안
- 제출 시 최저점수가 40점이므로 늦더라도 제출 할 것 (미제출 : 0점)
- 표절 심증이 있을 시, 최저점수 반영

*** train, valid, test 데이터셋 구성 변경은 금합니다.**

CV Homework 1

PIAI Research Department

Ozgenel(METU): Concrete Crack Images

Image size : 224 × 224

Dataset

	Crack	Non-Crack	Total
ORIGIN	20000	20000	40000

	Crack	Non-Crack	Total
Train	5400	600	6000
Valid	1800	200	2000
Test	180	20	200

*** Labels 데이터의 외부 유출은 금합니다.**

https://www.academia.edu/download/60483368/2018-Performance_Comparison_of_Pretrained_Convolutional_Neural_Networks_on_Crack_Detection_in_Buildings-_ISARC201820190904-28888-14zdu4h.pdf

1. YOLO v5 다운로드

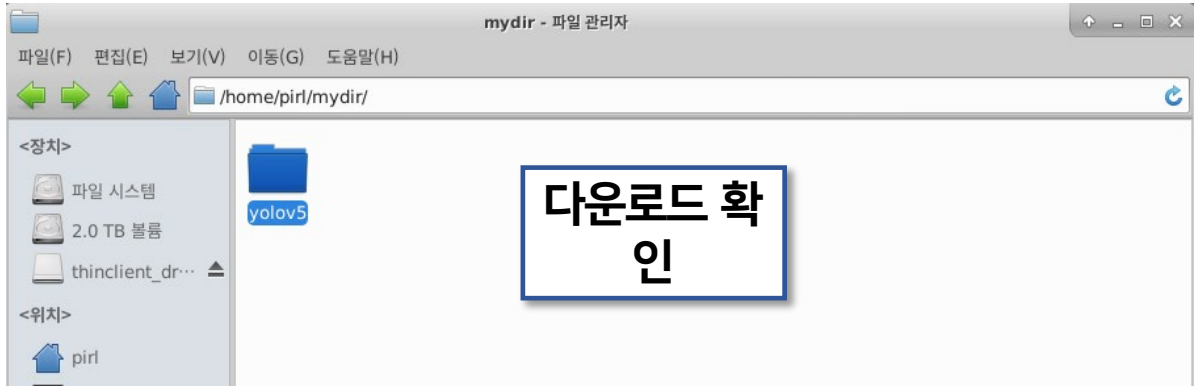
1. YOLOv5 requirements 가 모두 설치된 가상환경 Activate (ex. cv_edu2)
   - conda activate cv_edu2

2. 가상환경에 Git 설치 (설치되어 있을 시 Pass)
   - conda install git

```
(cv_edu2) pirl@pirl-PowerEdge-R740:~$ conda install git
```

3. yolov5를 다운받을 디렉터리로 이동 후 git clone으로 다운
   - cd /원하는 디렉토리/
   - git clone https://github.com/ultralytics/yolov5.git

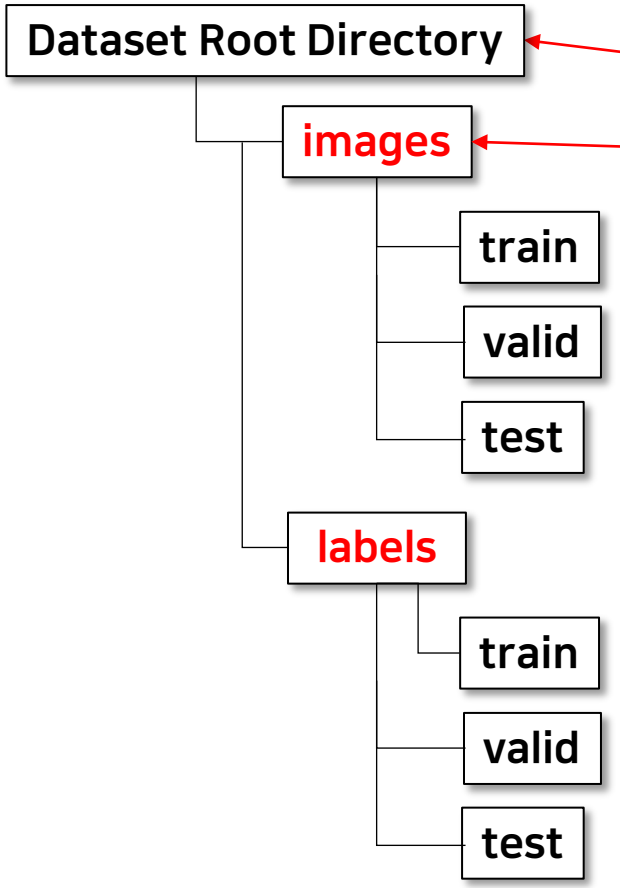
```
(cv_edu2) pirl@pirl-PowerEdge-R740:~$ mkdir mydir
(cv_edu2) pirl@pirl-PowerEdge-R740:~/mydir$ git clone https://github.com/ultralytics/yolov5.git
```



2. 학습/검증/테스트 데이터셋 준비

- YOLOv5에서 이미지 데이터는 images 폴더 내에, 라벨 데이터는 labels 폴더 내에 위치시켜야 읽어올 수 있다.
- YOLOv5는 yaml 파일을 통해 학습/검증/테스트 데이터셋이 위치한 디렉토리 위치를 알아낸다.
- Images 폴더 내 구조와 labels 폴더 내 구조가 같아야 yolov5가 라벨 데이터를 읽을 수 있다. (yaml에서는 images 위치만 명시하기 때문)

<YOLO v5 학습 시 지켜야 할 디렉터리 구조>



<coco128.yaml file 예시>

```yaml
# Train/val/test sets as 1) dir: path/to/imgs, 2) file: path/to/imgs.txt, or 3) list: [path/to/imgs1, p
path: ../datasets/coco128  # dataset root dir
train: images/train2017  # train images (relative to 'path') 128 images
val: images/train2017  # val images (relative to 'path') 128 images
test:  # test images (optional)

# Classes
nc: 80  # number of classes
names: ['person', 'bicycle', 'car', 'motorcycle', 'airplane', 'bus', 'train', 'truck', 'boat', 'traffic
        'fire hydrant', 'stop sign', 'parking meter', 'bench', 'bird', 'cat', 'dog', 'horse', 'sheep',
        'elephant', 'bear', 'zebra', 'giraffe', 'backpack', 'umbrella', 'handbag', 'tie', 'suitcase',
        'skis', 'snowboard', 'sports ball', 'kite', 'baseball bat', 'baseball glove', 'skateboard', 'su
        'tennis racket', 'bottle', 'wine glass', 'cup', 'fork', 'knife', 'spoon', 'bowl', 'banana', 'ap
        'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza', 'donut', 'cake', 'chair', 'couc
        'potted plant', 'bed', 'dining table', 'toilet', 'tv', 'laptop', 'mouse', 'remote', 'keyboard',
        'microwave', 'oven', 'toaster', 'sink', 'refrigerator', 'book', 'clock', 'vase', 'scissors', 't
        'hair drier', 'toothbrush']  # class names


# Download script/URL (optional)
download: https://ultralytics.com/assets/coco128.zip
```

2. 학습/검증/테스트 데이터셋 준비

PIAI Research Department

- 실제 학습에 사용된 yolov5/data/crack.yaml과 데이터셋 디렉토리 예시이니 참조할 것.

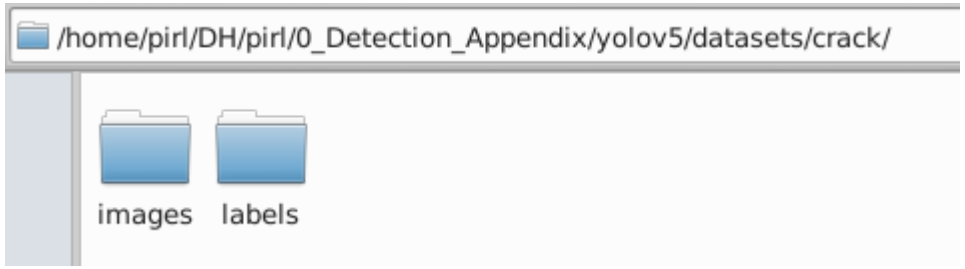
1. yolov5/data/coco128.yaml를 수정하여 yolov5/data/crack.yaml을 만듦

```
path: datasets/crack  # dataset root dir
train: images/train  # train images (rela
val: images/train  # val images (relative
test: images/test # test images (optiona

# Classes
nc: 1  # number of classes
names: ['crack']  # class names
```

내 데이터셋에 대한 클래스 개수와 이름 꼭 명시할 것

2. 데이터셋은 yolov5 폴더 내에 datasets라는 새로운 폴더를 만듦.

→ yolov5/datasets/crack 내 존재하는 폴더



→ yolov5/datasets/crack/images 내에는 train, valid, test 폴더가 존재
- train 폴더 내에는 학습시킬 이미지 데이터가 존재(.jpg 등)
- valid 폴더 내에는 검증할 이미지 데이터가 존재(.jpg 등)
- test 폴더 내에는 테스트할 이미지 데이터가 존재(.jpg 등)

→ yolov5/datasets/crack/labels 내에도 train, valid, test 폴더가 존재
- train 폴더 내에는 학습시킬 라벨 데이터가 존재(.txt)
- valid 폴더 내에는 검증할 이미지 데이터가 존재(.txt)
- test 폴더 내에는 테스트 할 라벨 데이터가 존재(.txt)

3. 모델 선정 및 모델 yaml 수정

PIAI Research Department

- yolov5 중 s/n/m/l/x 중 사용할 모델을 결정한 후 yolov5/models/사용할모델.yaml을 수정

예시) yolov5/data/yolov5s.yaml 수정 후 저장

열기(O) + 2.0 TB 볼륨 ~/DH/pirl/0_Detection_Appendix/yolov5/models 저장(S)

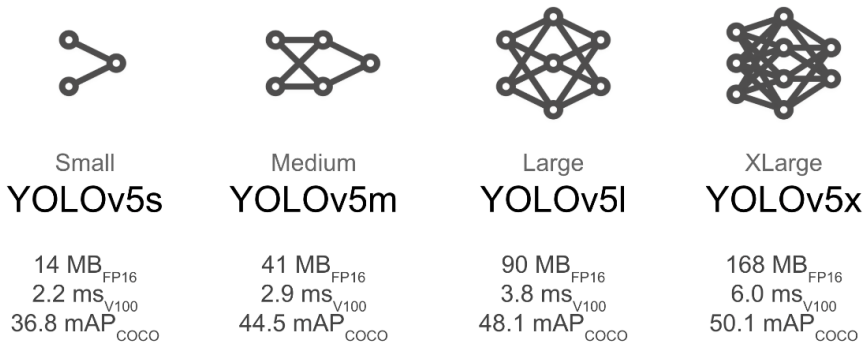
*crack.yaml | coco128.yaml | yolov5s.yaml

```
# YOLOv5 🚀 by Ultralytics, GPL-3.0 license

# Parameters
nc: 1  # number of classes
depth_multiple: 0.33  # model depth multiple
width_multiple: 0.50  # layer channel multiple
anchors:
  - [10,13, 16,30, 33,23]  # P3/8
  - [30,61, 62,45, 59,119]  # P4/16
  - [116,90, 156,198, 373,326]  # P5/32

# YOLOv5 v6.0 backbone
backbone:
  # [from, number, module, args]
  [[-1, 1, Conv, [64, 6, 2, 2]],  # 0-P1/2
   [-1, 1, Conv, [128, 3, 2]],  # 1-P2/4
   [-1, 3, C3, [128]],
   [-1, 1, Conv, [256, 3, 2]],  # 3-P3/8
   [-1, 6, C3, [256]],
   [-1, 1, Conv, [512, 3, 2]],  # 5-P4/16
   [-1, 9, C3, [512]],
   [-1, 1, Conv, [1024, 3, 2]],  # 7-P5/32
   [-1, 3, C3, [1024]],
   [-1, 1, SPPF, [1024, 5]],  # 9
  ]

# YOLOv5 v6.0 head
head:
  [[-1, 1, Conv, [512, 1, 1]],
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [[-1, 6], 1, Concat, [1]],  # cat backbone P4
   [-1, 3, C3, [512, False]],  # 13

   [-1, 1, Conv, [256, 1, 1]],
```

"/home/pirl/DH/pirl/0_Detection_Appendix/yolov5/models/yolov5s.yaml" 파일을 읽어들이는 중입니다… YAML 탭 너비: 8 20행, 42열 삽입

필수 : 내 데이터셋에 대한 클래스 개수로 꼭 변경할 것

옵션 : 앵커박스 또는 모델을 수정하고 싶을 시 변경



4. 하이퍼파라미터 수정

PIAI Research Department

- yolov5/data/hyps 내에서 사용할 yaml 파일 선택한 후 해당 파일을 수정

예시) yolov5/data/hyps/hyp.scratch-low.yaml수정 후 저장

```yaml
lr0: 0.01  # initial learning rate (SGD=1E-2, Adam=1E-3)
lrf: 0.01  # final OneCycleLR learning rate (lr0 * lrf)
momentum: 0.937  # SGD momentum/Adam beta1
weight_decay: 0.0005  # optimizer weight decay 5e-4
warmup_epochs: 3.0  # warmup epochs (fractions ok)
warmup_momentum: 0.8  # warmup initial momentum
warmup_bias_lr: 0.1  # warmup initial bias lr
box: 0.05  # box loss gain
cls: 0.5  # cls loss gain
cls_pw: 1.0  # cls BCELoss positive_weight
obj: 1.0  # obj loss gain (scale with pixels)
obj_pw: 1.0  # obj BCELoss positive_weight
iou_t: 0.20  # IoU training threshold
anchor_t: 4.0  # anchor-multiple threshold
# anchors: 3  # anchors per output layer (0 to ignore)
```

```yaml
hsv_h: 0.015  # image HSV-Hue augmentation (fraction)
hsv_s: 0.7  # image HSV-Saturation augmentation (fraction)
hsv_v: 0.4  # image HSV-Value augmentation (fraction)
degrees: 0.0  # image rotation (+/- deg)
translate: 0.1  # image translation (+/- fraction)
scale: 0.5  # image scale (+/- gain)
shear: 0.0  # image shear (+/- deg)
perspective: 0.0  # image perspective (+/- fraction), range 0-0.001
flipud: 0.0  # image flip up-down (probability)
fliplr: 0.5  # image flip left-right (probability)
mosaic: 1.0  # image mosaic (probability)
mixup: 0.0  # image mixup (probability)
copy_paste: 0.0  # segment copy-paste (probability)
```

2. 모델 학습 및 검증

- yolov5 모델 학습

```
$ python train.py --data ozgenel.yaml --cfg yolov5s.yaml --weights yolov5s.pt
 --imgsz 228 --batch-size 32 --hyp hyp.scratch-low.yaml --epochs 3
```

- --data : 데이터셋의 경로, class 종류 등의 정보 파일
- --cfg : 학습 모델의 구조 정보 파일
- --weights : 기존에 학습된 weights 파일
- --imgsz : resize할 이미지 크기
- --batch-size : 한번에 학습시킬 데이터의 크기
- --hyp : 학습에 사용할 하이퍼파라미터 정보 파일

```
      Epoch    GPU_mem   box_loss   obj_loss   cls_loss  Instances       Size
      97/99     0.726G    0.01755    0.01103          0         45        256: 100%|██████████| 375/375 [00:49<00:00,  7.53it/s]
                 Class     Images  Instances          P          R      mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:32<00:00,  1.96it/s]
                   all       2000       2061      0.961      0.956      0.984       0.84

      Epoch    GPU_mem   box_loss   obj_loss   cls_loss  Instances       Size
      98/99     0.726G    0.01765    0.01097          0         36        256: 100%|██████████| 375/375 [00:49<00:00,  7.55it/s]
                 Class     Images  Instances          P          R      mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:47<00:00,  1.34it/s]
                   all       2000       2061      0.966      0.951      0.984      0.841

      Epoch    GPU_mem   box_loss   obj_loss   cls_loss  Instances       Size
      99/99     0.726G    0.01717    0.01092          0         41        256: 100%|██████████| 375/375 [00:41<00:00,  8.95it/s]
                 Class     Images  Instances          P          R      mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:47<00:00,  1.32it/s]
                   all       2000       2061      0.966      0.951      0.984      0.841

100 epochs completed in 2.495 hours.
Optimizer stripped from runs/train/exp9/weights/last.pt, 14.3MB
Optimizer stripped from runs/train/exp9/weights/best.pt, 14.3MB

Validating runs/train/exp9/weights/best.pt...
Fusing layers...
YOLOv5s summary: 213 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs
                 Class     Images  Instances          P          R      mAP@.5 mAP@.5:.95: 100%|██████████| 63/63 [00:49<00:00,  1.28it/s]
                   all       2000       2061      0.969      0.953      0.983      0.842
```

2. 모델 학습 및 검증

- yolov5 모델 검증

```
$ python val.py --weights yolov5s.pt --data ozgenel.yaml --imgsz 228 --task test
```

```
(pytorch) piai@piai-Precision-Tower-5810:~/바탕화면/yolov5-master$ python val.py --weights ./runs/train/exp9/weights/best.pt --data ./data/ozgenel.yaml --task test --imgsz 228
val: data=./data/ozgenel.yaml, weights=['./runs/train/exp9/weights/best.pt'], batch_size=32, imgsz=228, conf_thres=0.001, iou_thres=0.6, task=test, device=, workers=8, single_cls=False, augment=False, verbose=False, save_txt=False, save_hybrid=False, save_conf=False, save_json=False, project=runs/val, name=exp, exist_ok=False, half=False, dnn=False
YOLOv5 🚀 2022-8-31 Python-3.8.13 torch-1.10.2 CUDA:0 (NVIDIA GeForce RTX 3080, 10010MiB)

Fusing layers...
YOLOv5s summary: 213 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs
WARNING: --img-size 228 must be multiple of max stride 32, updating to 256
test: Scanning '/home/piai/바탕화면/datasets/crack/labels/test.cache' images and labels... 200 found, 0 missing, 20 empty, 0 corrupt: 100%|██████████| 200/200 [00:00<?, ?it/s]
               Class     Images  Instances          P          R      mAP@.5 mAP@.5:.95: 100%|██████████| 7/7 [00:01<00:00,  5.58it/s]
                 all        200        214      0.979      0.935      0.989      0.858
Speed: 0.0ms pre-process, 0.8ms inference, 1.6ms NMS per image at shape (32, 3, 256, 256)
```

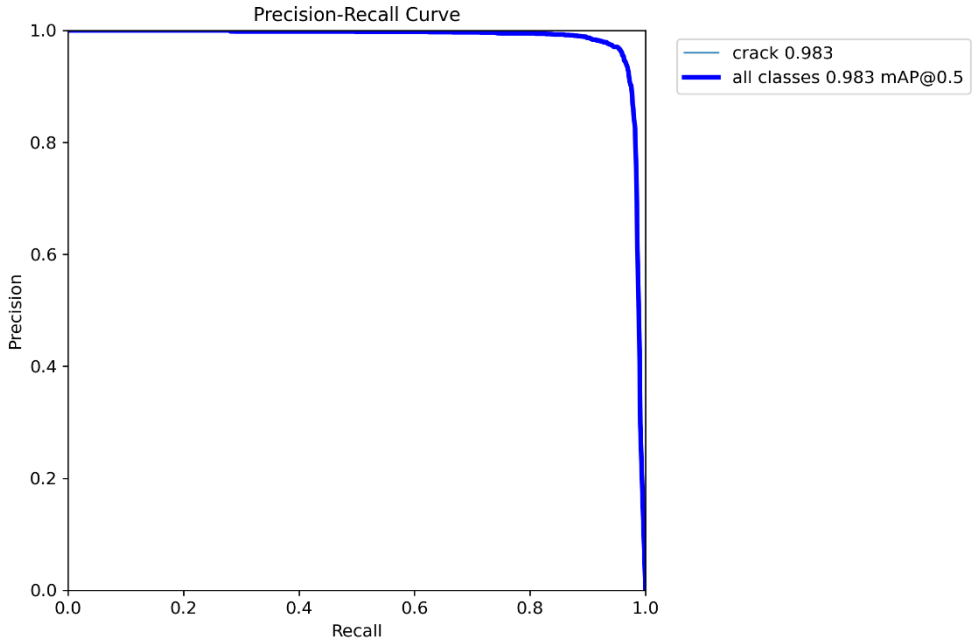
Class	Images	Instances	P	R	mAP@.5	mAP@.5:.95:
all	200	214	0.979	0.935	0.989	0.858

mAP@.5:.95:
0.5부터 0.95사이에서 IoU threshold 값을 조정하여 구한 mAP의 평균

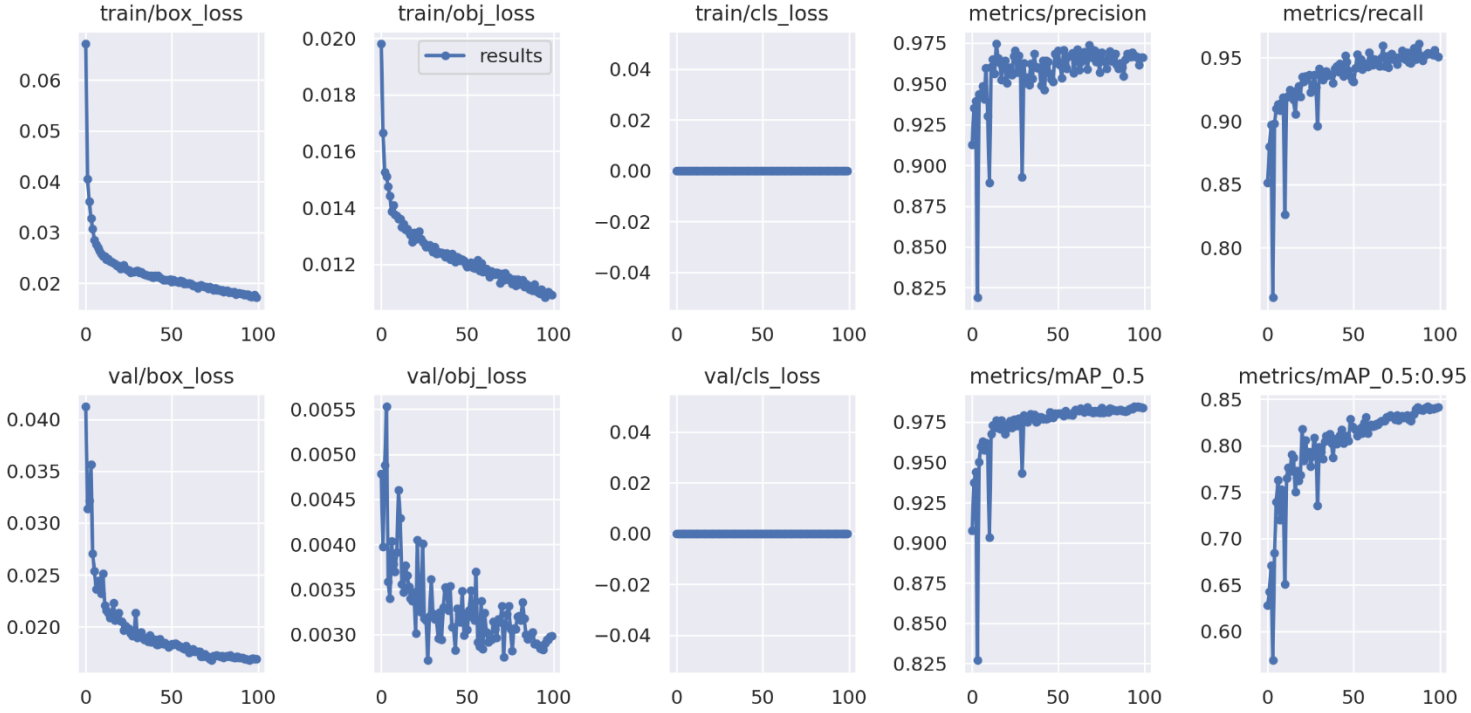
CV Homework 1

PIAI Research Department

- **PR_curve**



- **results**



PIAI

Appendix – Annotation by labelImg

1. 기본 환경 세팅 – LabelImg 설치

- LabelImg 프로그램은 Object Detection 라벨링 툴이며 pip3 명령어로 쉽게 설치 가능
  ※ LabelImg Github : https://github.com/tzutalin/labelImg

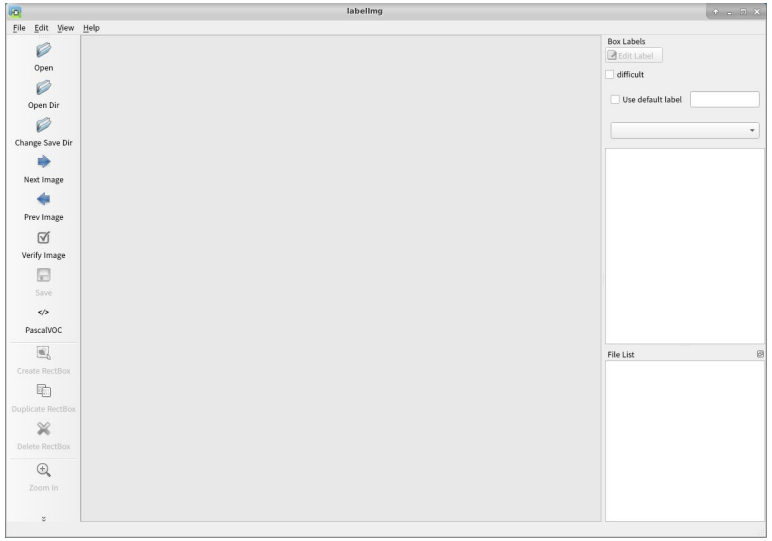
1. 가상환경 생성 및 가상환경 실행
   - conda create -n labelImg python=3.8
   - conda activate labelImg

2. 생성 가상환경 내 lablImg 설치
   - pip3 install labelImg

```
(labelImg) pirl@pirl-PowerEdge-R740:~$ pip3 install labelImg
```

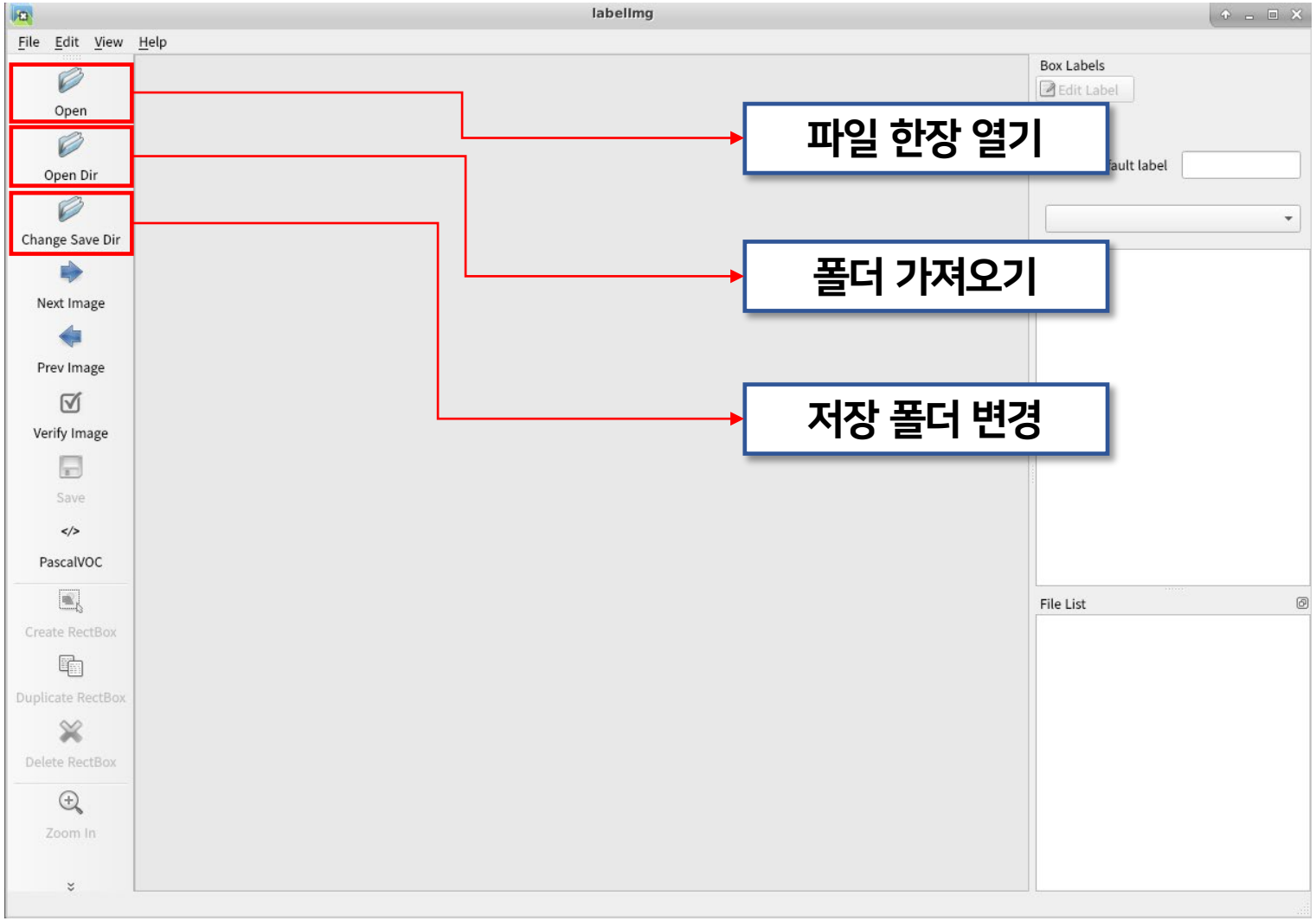
3. 생성 가상환경 내에서 labelImg 실행
   - labelImg

```
(labelImg) pirl@pirl-PowerEdge-R740:~$ labelImg
```



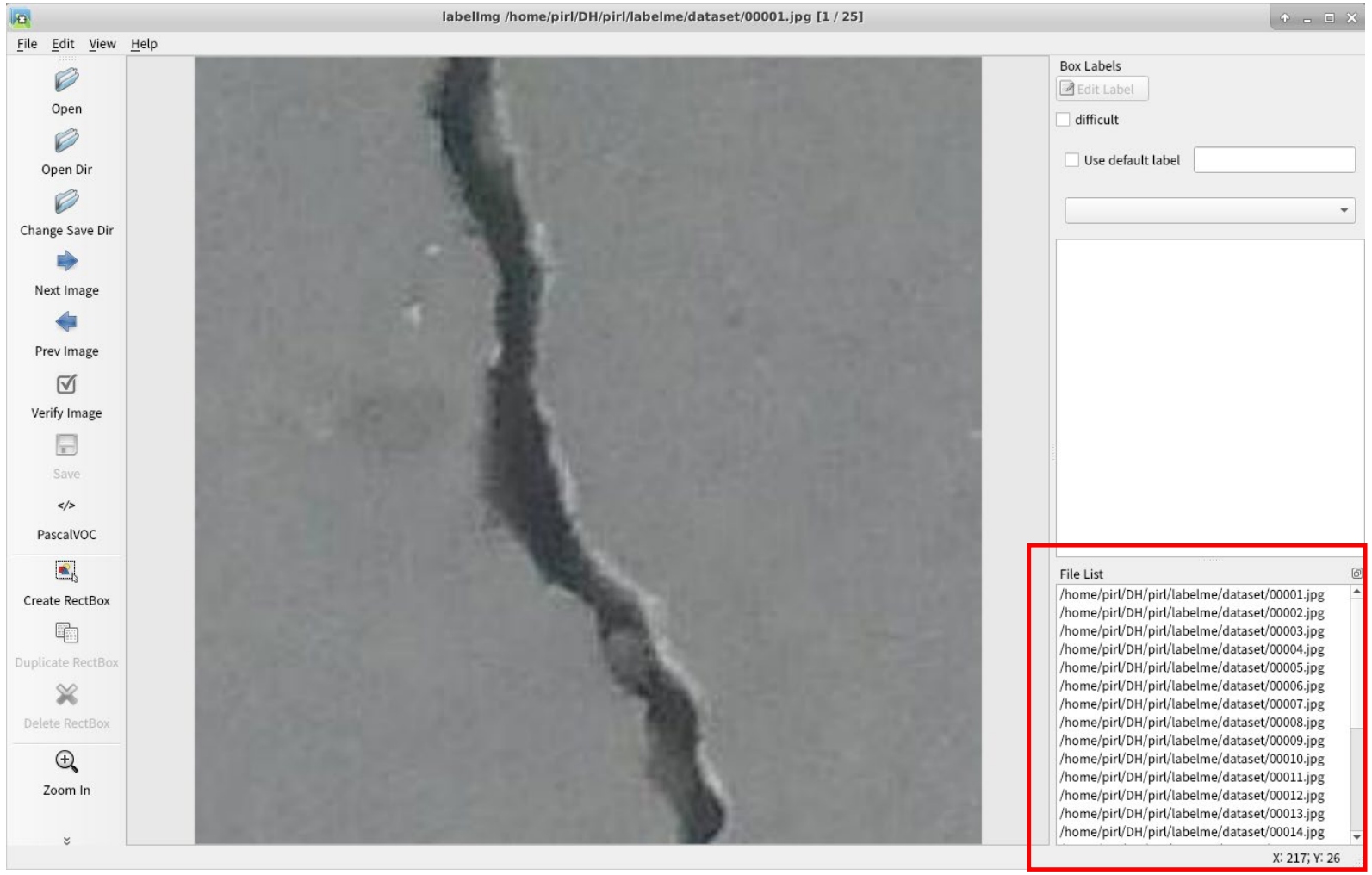
2. LabelImg 기능 – 파일 및 디렉토리 열기

PIAI Research Department



2. LabelImg 기능 – Open Dir 및 단축키

PIAI Research Department

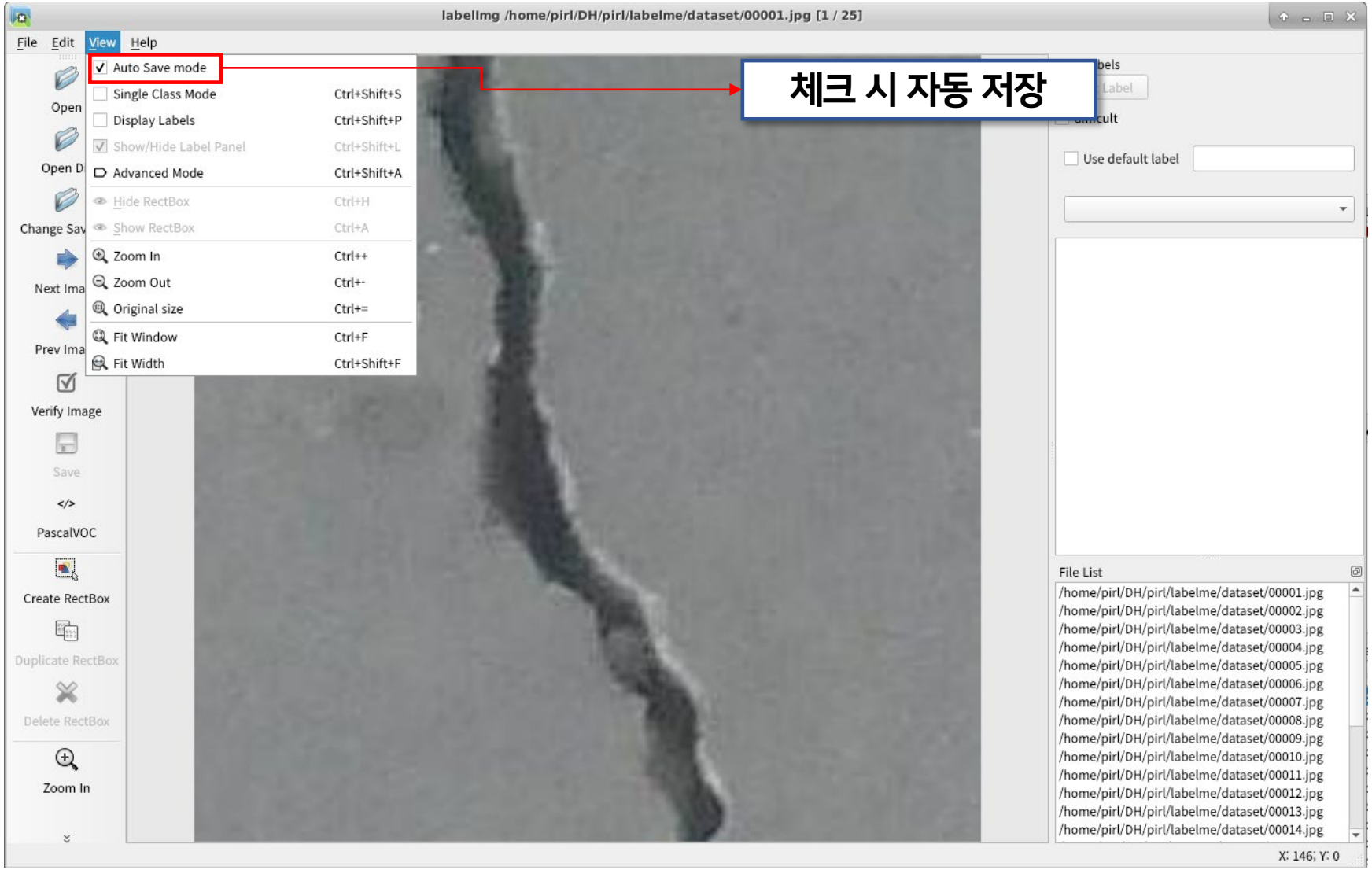


Ctrl + u	Load all of the images from a directory
Ctrl + r	Change the default annotation target dir
Ctrl + s	Save
Ctrl + d	Copy the current label and rect box
Ctrl + Shift + d	Delete the current image
Space	Flag the current image as verified
w	Create a rect box
d	Next image
a	Previous image
del	Delete the selected rect box
Ctrl++	Zoom in
Ctrl--	Zoom out
↑→↓←	Keyboard arrows to move selected rect box

2. LabelImg 기능 – 자동 저장 모드

PIAI Research Department

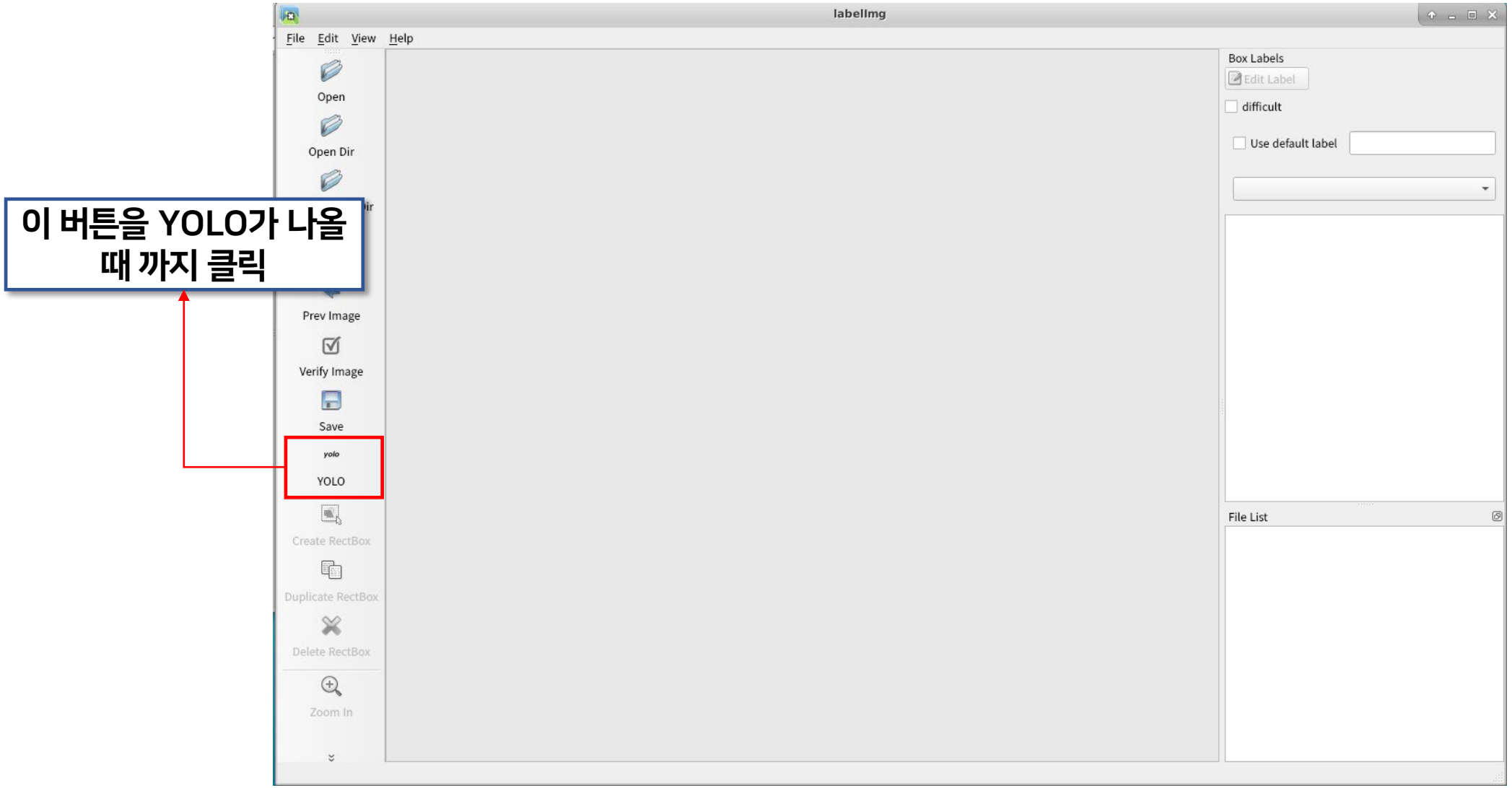
- 저장을 수동적으로 하지 않고 이전/다음 이미지로 넘어가면 자동적으로 저장하는 모드



2. LabelImg 기능 – YOLO Format Annotation

PIAI Research Department

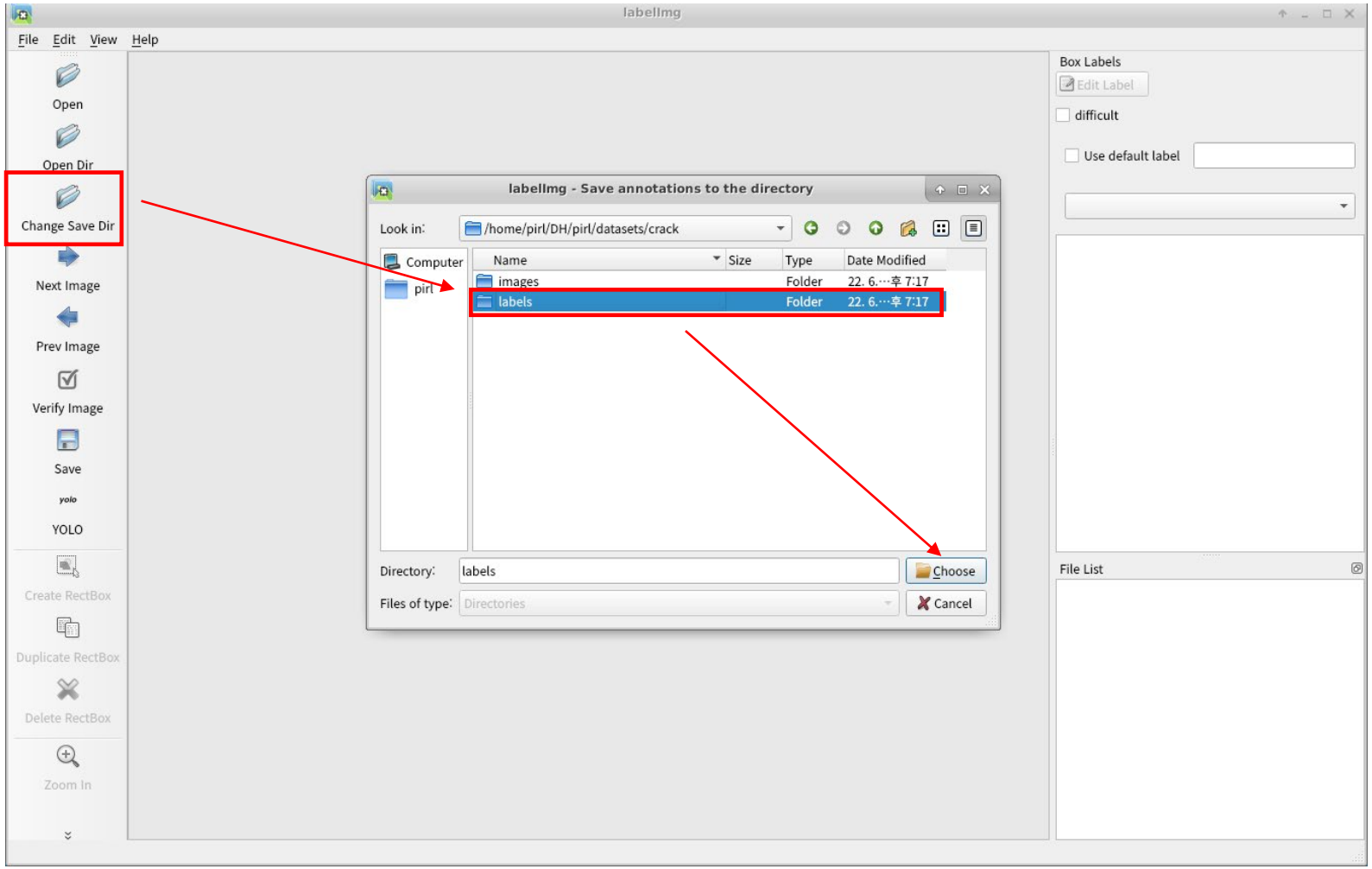
- LabelImg 사용 시 YOLO 형태로 Annotation 하고 싶다면 반드시 좌측의 버튼을 눌러 설정해 주어야함



2. LabelImg 기능 – 라벨 저장 위치

PIAI Research Department

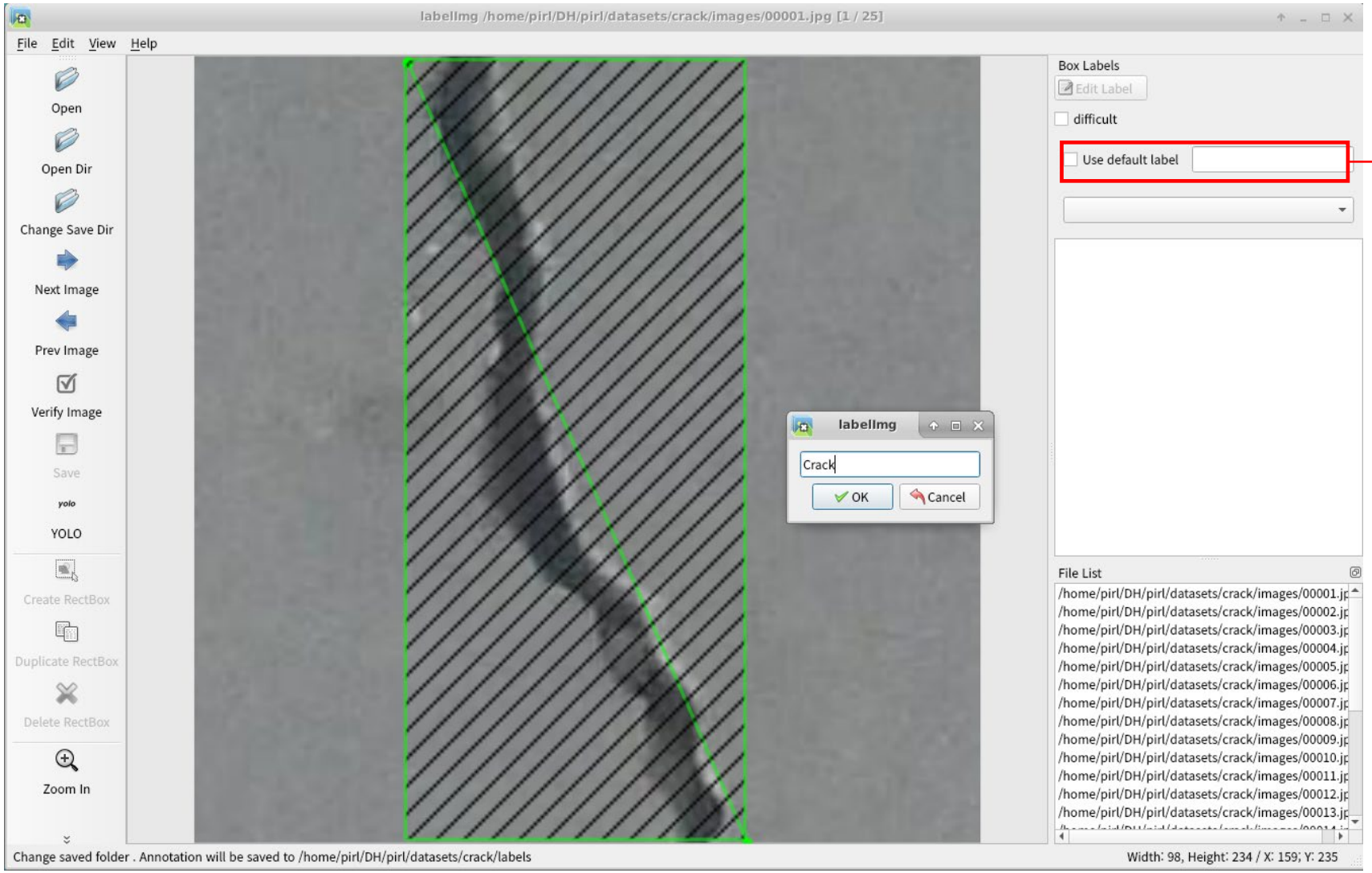
- YOLO v5를 학습시킬 때 이미지 파일과 txt로 된 라벨 파일을 다른 폴더에 두어 학습시키므로, labels 폴더를 만들어 라벨 txt 파일을 저장 (이미지 파일들은 images 폴더에 저장)



2. LabelImg를 활용한 라벨링

PIAI Research Department

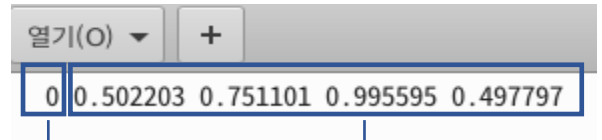
- w를 눌러 Bbox를 그린 뒤 라벨을 부여



Default Label을 설정해서 사용할 수도 있음

이미지 하나당 한 개의 .txt 파일 생성

예시)



클래스 번호

BBox좌표