



# 컴퓨터 비전 최종 발표

AI융합학부 20201782 김다빈  
20180362 김태우  
20180391 이지훈

# 목차 a table of contents



1

Object Detection 알고리즘의 중요성

2

개발 동기 및 기술 개발 동향

3

주요 개발 내용

4

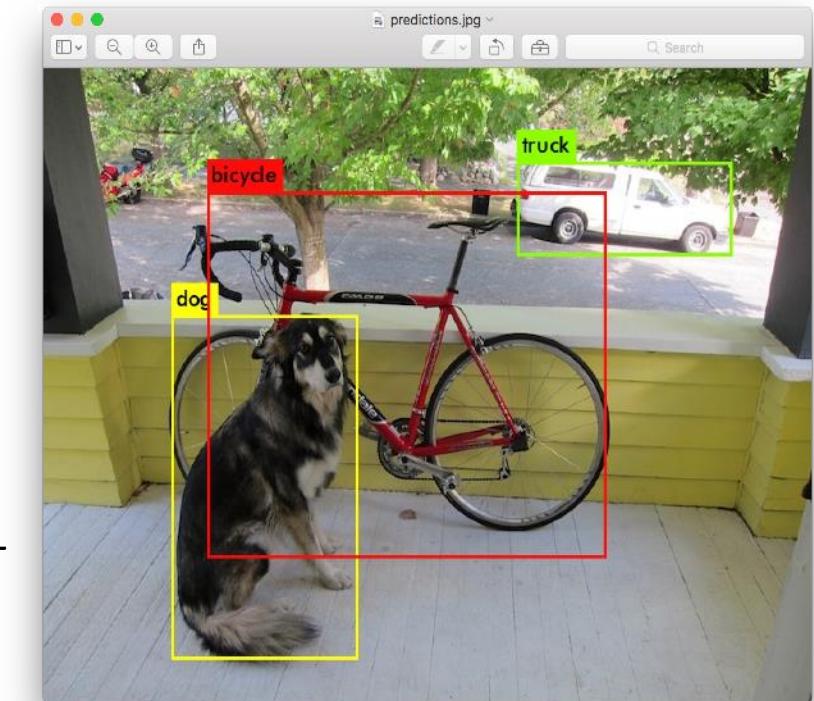
프로젝트 수행 일정 및 팀원 역할 분담

# Object Detection 알고리즘

## Object Detection 이란?

- 컴퓨터 비전과 이미지 처리와 관련된 컴퓨터 기술로서, 디지털 이미지와 비디오로 특정한 계열의 시맨틱 객체 인스턴스(ex. 인간, 건물, 자동차)를 감지하는 일

- 여러 물체에 대해 어떤 물체인지 분류하는 Classification 문제 + 물체가 어디 있는지 Bounding Box를 통해 위치 정보를 나타내는 Localization 문제를 둘 다 해내야 하는 분야



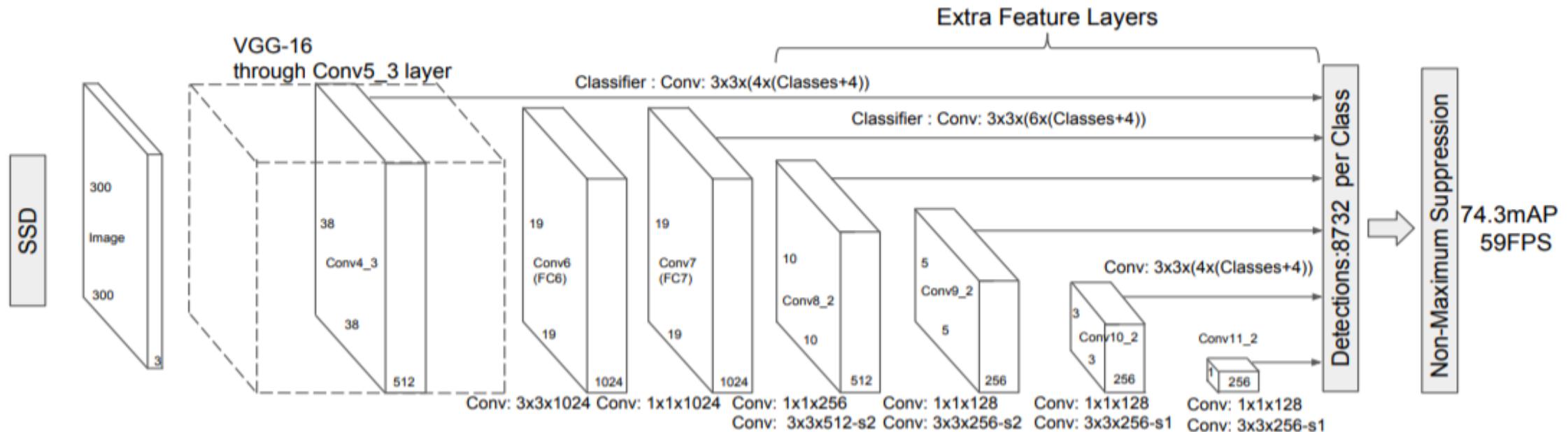
\*Object Detection = 여러가지 물체에 대한 Classification + 물체의 위치정보를 파악하는 Localization

# Object Detection 알고리즘

## Object Detection 중요성?

- 해양 조난자 검출 모델로 드론 영상 데이터를 기반으로 객체와 조난자 여부 탐지 가능
- CCTV 영상 분석하여 교통량 혹은 인구 밀집 지역 분류 가능
- COVID-19 팬데믹 상황에서 마스크 착용자 및 발열 환자를 쉽게 분류
- 농업과 제조업에서 불량 검출 모델을 활용하여 수확한 작물의 품질과 불량 분류  
-> 다양한 분야에서 다양한 용도로 활용 가능한 기술

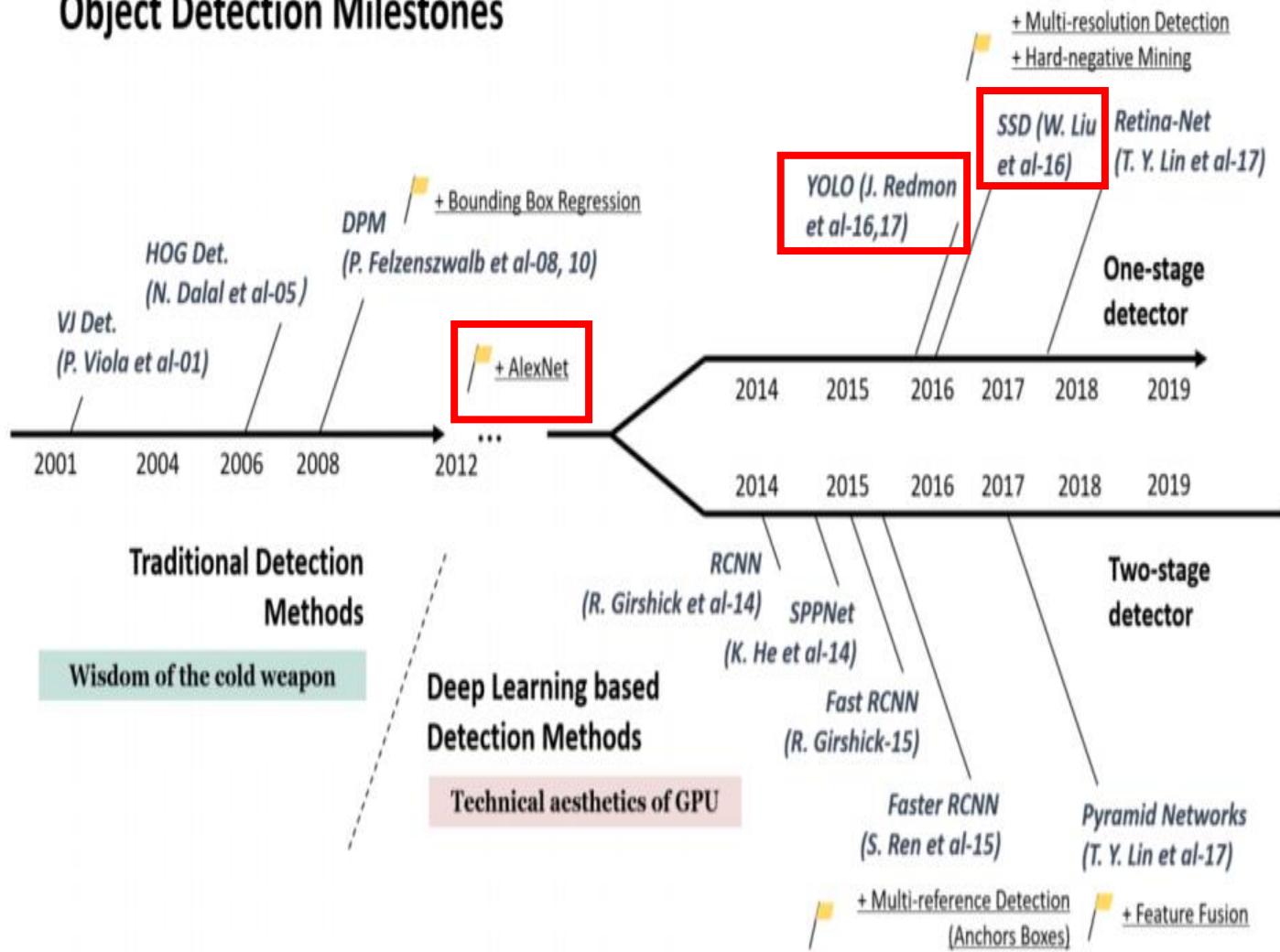
# 개발 동기



기존 SSD 알고리즘 인공 신경망의 구조를 변경하거나 추가된 차별화된 네트워크를 구성하여 Object Detection의 주 성능 지표로 사용되는 mAP의 성능을 높이기 위해 개발

# 기술 개발 동향

## Object Detection Milestones



2012년 Alex Net의 탄생을 기점으로 딥러닝을 활용하여 Object Detection을 하기 시작



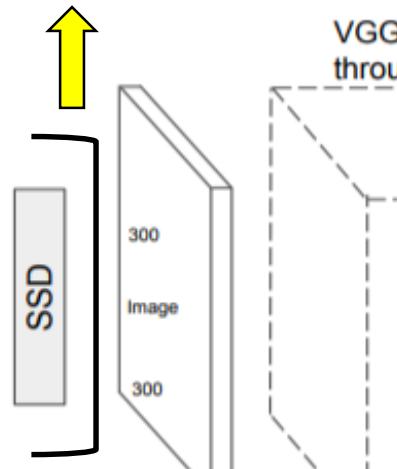
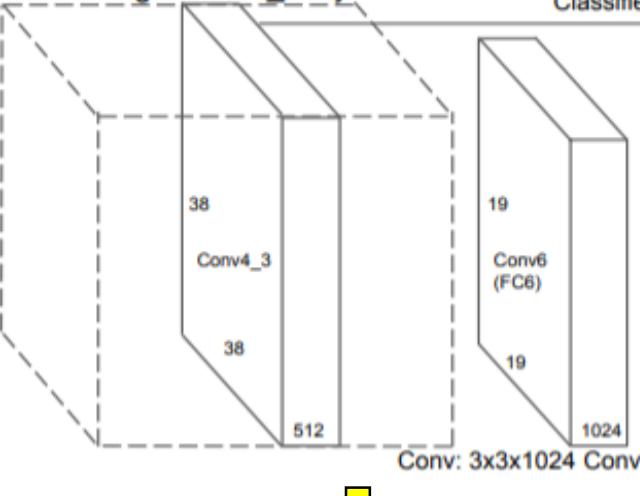
Two-stage detector은 객체가 있을 것 같은 위치를 찾아 정보를 제공해주는 과정과 데이터를 분류하는 과정을 2단계로 수행



One-stage detector는 객체가 있을 것으로 추정되는 위치 찾는 과정을 생략하고 진행 Classification을 하기 위해 만들어지는 feature map에서 동시에 영역 제안을 진행

# 주요 개발 내용(Outline)

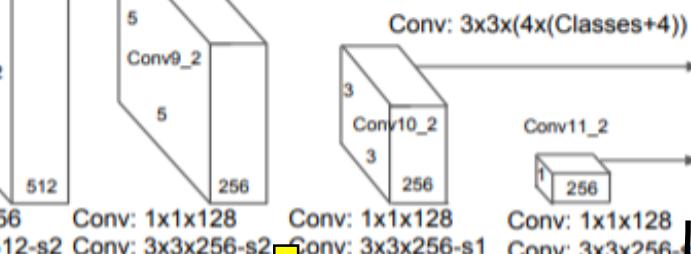
#0 Data augmentation 설정

VGG-16  
through Conv5\_3 layer

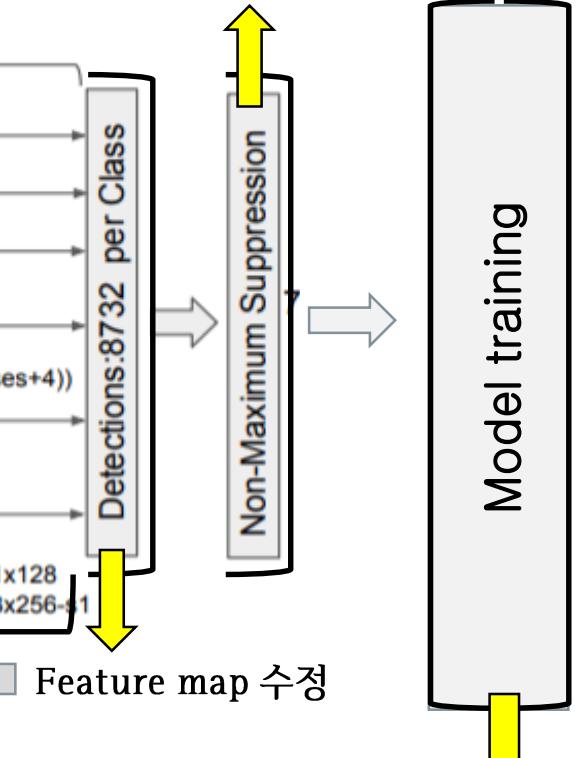
Classifier : Conv: 3x3x(4x(Classes+4))

Extra Feature Layers

Classifier : Conv: 3x3x(6x(Classes+4))



#4 Bounding box 생성 조건 설정



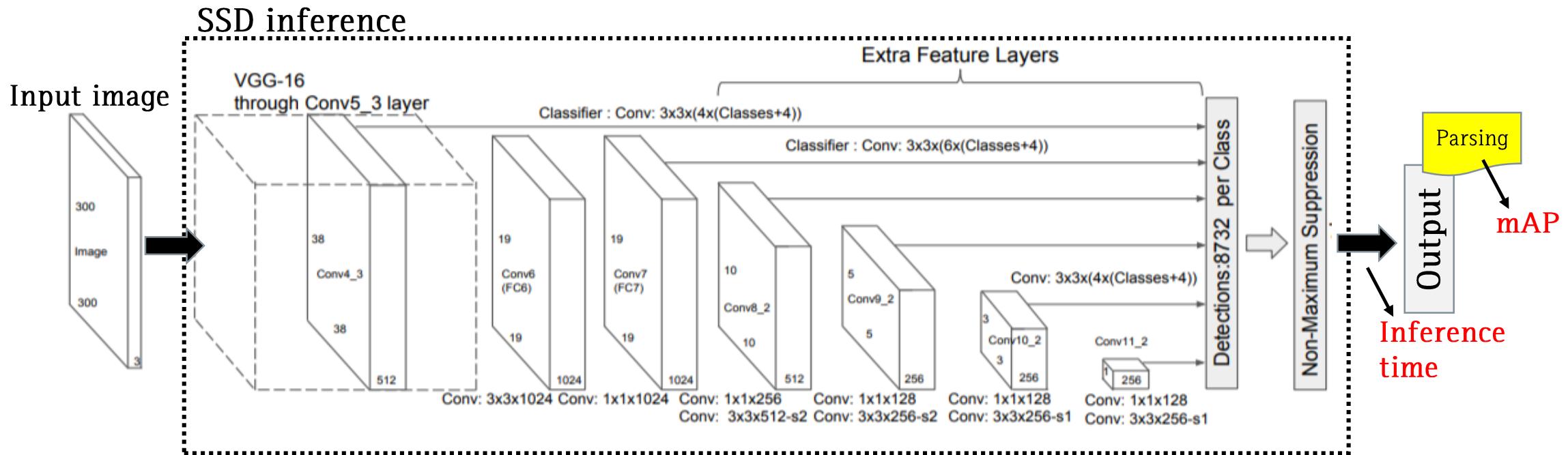
#1 Backbone model 설정

#2 Extras layers 설정

#3 Feature map 설정

#5 Training options 설정

# 주요 개발 내용(Outline)



## #Main approach

- ✓ #1. Inference time과 mAP 두 주요 지표를 동시에 향상 시키는 것은 어려울 것이다?  
학부생 수준에서는 어려울 것이라 판단, inference time을 증가시키는 방향으로 진행  
ex) extra 계층 추가, extra filter 개수 증가, feature map 증가
- ✓ #2. SSD paper 참조, mAP를 떨어트릴 수 있는 요인들을 하나하나 제거하는 방향으로 진행  
ex) 마지막 feature map 제거, conf\_thresh 증가
- ✓ #3. Other ?

# 주요 개발 내용(아이디어 제시)

- 1) backbone 모델 일부 parameter 값 조정
- 2) Feature map 개수 증감
- 3) Extras 계층 parameter 값 수정 (1. filter 개수 늘리기, 2. 모델의 크기를 늘리기)
- 4) Detect class hyper parameter 값 수정 (nms\_thresh, conf\_thresh, top\_k...)

# 주요 개발 내용(아이디어 제시)

5) BBox configure -> cfg에서 수정(min, max, ratio)

6) BBox 개수 늘리기

7) Overfitting 방지 정규화 기술 적용하기

- Data augmentation -> RandomSampleCrop 제거
- 배치 정규화
- Early Stopping
- Drop Out

8) 이미지 사이즈 300-> 512

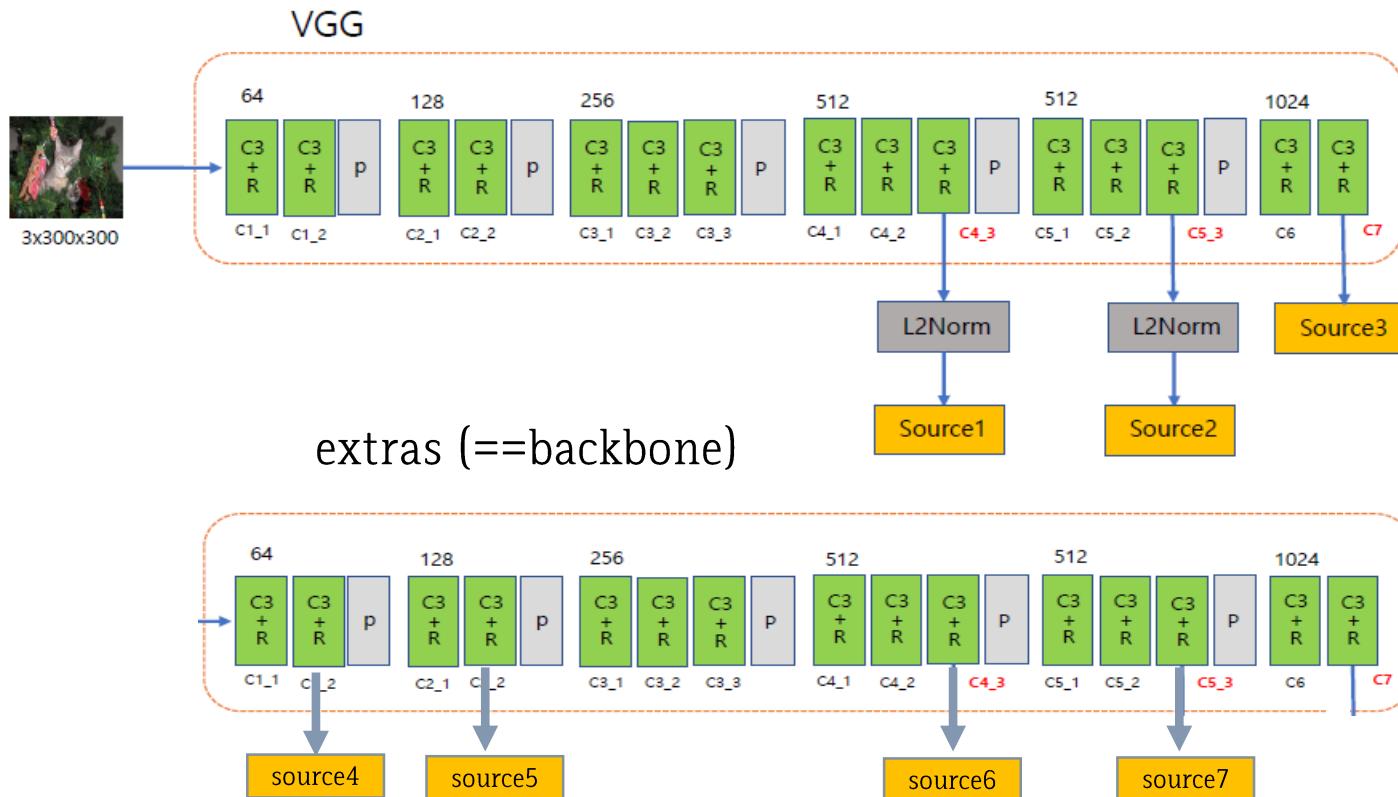
9) 기타

- optimizer-> cosine annealing으로 변경
- loss 함수 -> focal loss로 변경

# 주요 개발 내용(#1)

## backbone 모델 일부 parameter 값 조정

- Backbone 모델의 구조를 조금이라도 수정하면 주어진 가중치를 사용할 수 없었음
- 중간 과제의 경향성을 따라서, backbone은 건드리지 않는 대신 extras를 backbone 처럼 바꾸어 보는 방식으로 시도
- Extras 구조를 주어진 backbone과 동일하게 하여, **backbone + backbone의 구조를 가지는 SSD 모델 개발 후 테스트 (mAP 56.6)**



# 주요 개발 내용(#1)

## extras 계층 parameter 값 수정하기

filter 개수 늘리기(추론 시간은 걸리지만 모델의 표현력↑)



C3: 3x3 kernel의 convolution layer  
 C1: 1x1 3x3 kernel의 convolution layer  
 R: Relu activation function  
 P: Max Pooling layer

# 주요 개발 내용(#1)

## Dbox 개수 늘리기

-작은 객체에 대한 검출의 정확도를 높이기 위해 기존 DBox 개수인 4개에서 6개로 증가시킴

```
ssd_cfg = {
    'num_classes': 21, # 배경 클래스를 포함한 총 클래스 수
    'input_size': 300, # 이미지의 입력 크기
    'bbox_aspect_num': [6, 6, 6, 6, 4, 4, 4], # 출력할 DBox의 화면비의 종류
    'feature_maps': [38, 19, 19, 19, 9, 2, 1], # 각 source의 이미지 크기
    'steps': [8, 16, 16, 16, 100, 300, 300],
    'min_sizes': [30, 60, 60, 60, 213, 264, 264], # DBOX의 크기(최소)
    'max_sizes': [60, 111, 111, 111, 264, 315, 315], # DBOX의 크기(최대)
    'aspect_ratios': [[2, 3], [2, 3], [2, 3], [2, 3], [2], [2], [2]],
}
```

# 주요 개발 내용(#1)

Feature map 개수 늘리기(6개->7개 : 표본 늘리기)



C3: 3x3 kernel의 convolution layer  
 C1: 1x1 3x3 kernel의 convolution layer  
 R: Relu activation function  
 P: Max Pooling layer

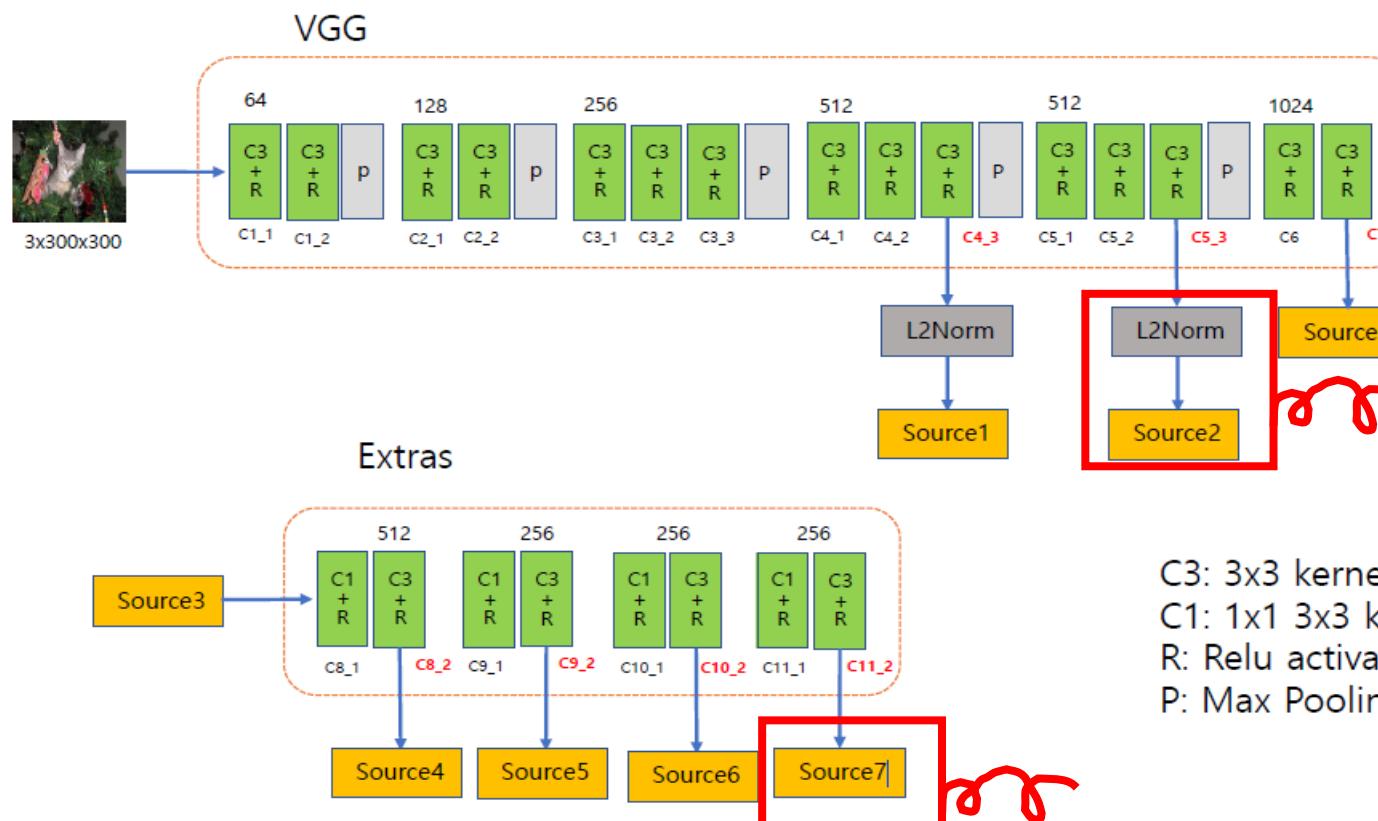
# 주요 개발 내용(#2)

Feature map 개수 줄이기(마지막 feature map 제거 (6개->5개))

Prediction source layers from:						mAP		# Boxes
conv4_3	conv7	conv8_2	conv9_2	conv10_2	conv11_2	use boundary boxes?	Yes	
✓	✓	✓	✓	✓	✓	74.3	63.4	8732
✓	✓	✓	✓	✓		<b>74.6</b>	63.1	8764
✓	✓	✓	✓			73.8	68.4	8942
✓	✓	✓				70.7	69.2	9864
✓	✓					64.2	64.4	9025
						62.4	64.0	8664

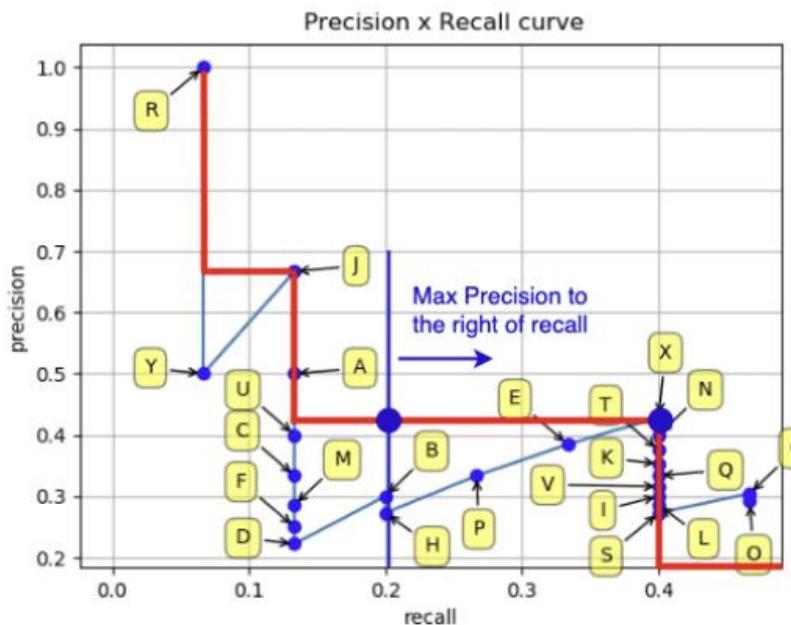
# 주요 개발 내용(#2)

Feature map 개수 줄이기(마지막 feature map 제거 (6개->5개))



C3: 3x3 kernel의 convolution layer  
 C1: 1x1 3x3 kernel의 convolution layer  
 R: Relu activation function  
 P: Max Pooling layer

# 주요 개발 내용(#2)



- AP(Average Precision): Curve graph 하단의 면적

$$AP = \int_0^1 p(r)dr$$

현 mAP 코드는 각 recall 값에서 precision 최댓값만을 가져와서 면적을 구함

## # Precision의 정의를 활용하여 mAP를 올릴 수 있을까?

$$Pre = \frac{TP}{TP+FP}$$

- TP를 증가시킨다?  
IOU threshold는 mAP 코드에서 정해져 있기 때문에 수정 불가
- TP+FP 를 감소시킨다? 따라서 bounding box를 만드는 기준을 엄격하게 설정  
→ conf\_thresh 증가하는 방향으로 진행



위와 같이 conf thresh가 낮은 Bounding Box를 제거할 수 있다.

# 주요 개발 내용(#2)

학습에 사용한 최적화 알고리즘과 사용된 hyperparameters 설정

ssd\_model code의 conf\_thresh , calculating code에서 data\_confidence\_level

-> (0.01, 0.1, 0.5, 0.6 여러 숫자로 테스트 진행)

```
class Detect(Function):
    # 조건 이상의 값을 추출
    data_confidence_level = 0.5

    #def __init__(self, =0.01, top_k=200, nms_thresh=0.45):
    #    self.softmax = nn.Softmax(dim=-1) # conf를 소프트맥스 함수
    #    self.conf_thresh = conf_thresh # conf가 conf_thresh=0.0
    #    self.top_k = top_k # nm_supression으로 conf가 높은 top_k까지만 선택이므로, top_k = 200
    #    self.nms_thresh = nms_thresh # nm_supression으로 IOU가 nms_thresh=0.45보다 크면 동일한 물체의 BBox로 간주

    #def forward(self, loc_data, conf_data, dbox_list ):

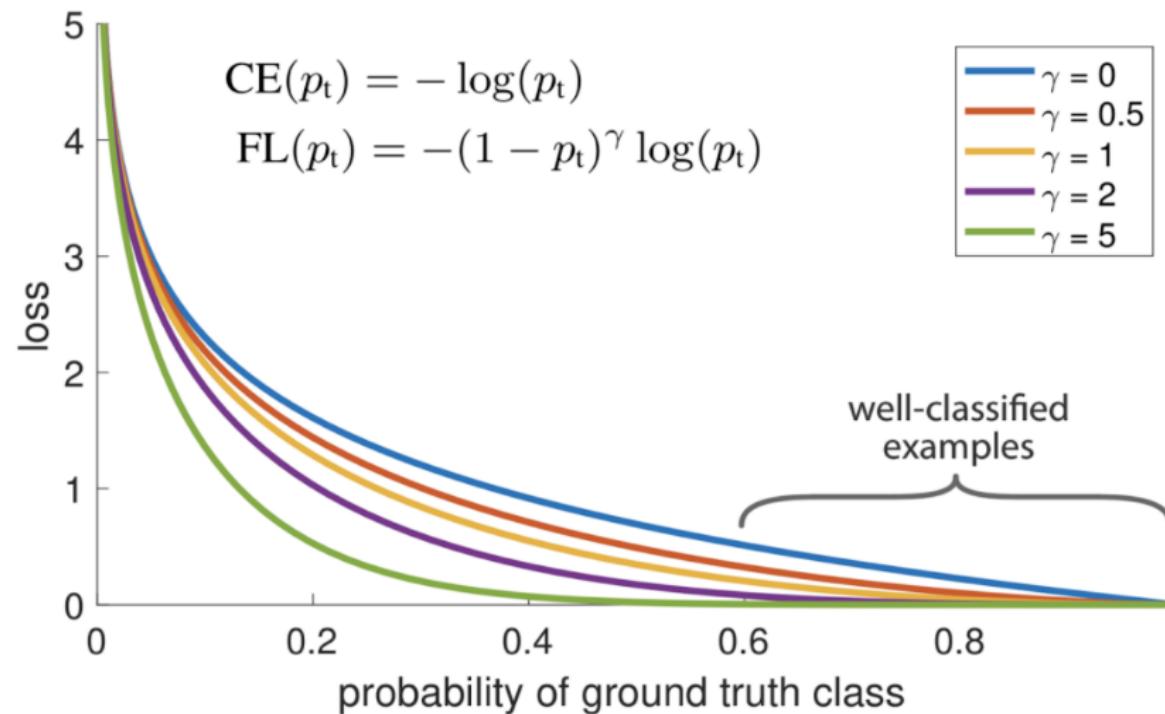
    # HOUN
    #def forward(self, loc_data, conf_data, dbox_list, conf_thresh=0.01, top_k=200, nms_thresh=0.45): #현 pytorch version에 맞게 수정함

    # HOUNING
    @staticmethod
    def forward(self, loc_data, conf_data, dbox_list, conf_thresh=0.1, top_k=200, nms_thresh=0.45): #현 pytorch version에 맞게 수정함
```

# 주요 개발 내용(#3)

## Overfitting 방지 정규화 기술 적용하기

- loss 함수 -> focal loss로 변경 (학습 중 클래스 불균형 문제 해결)



# 주요 개발 내용(#3)

## Overfitting 방지 정규화 기술 적용하기

- 배치 정규화 적용

```

bn1 = nn.BatchNorm1d(32)
bn2 = nn.BatchNorm1d(32)
bn3 = nn.BatchNorm1d(32)
bn4 = nn.BatchNorm1d(32)
bn5 = nn.BatchNorm1d(32)
bn6 = nn.BatchNorm1d(32)
bn7 = nn.BatchNorm1d(32)
bn8 = nn.BatchNorm1d(32)

layers += [nn.Conv2d(in_channels, cfg[0], kernel_
layers += [nn.Conv2d(cfg[0], cfg[1], kernel_size=
layers += [nn.Conv2d(cfg[1], cfg[2], kernel_size=

```

## Dropout 적용

```

# vgg를 끝까지 계산하여, source2를 작성하고, sources에 추가
for k in range(23, len(self.vgg)):
    x = self.vgg[k](x)

sources.append(x)
x = F.dropout(x, 0.5) # extras의 conv와 ReLU를 계산
# source3~6을 sources에 추가
for k, v in enumerate(self.extras):
    x = v(x)
    nn.BatchNorm1d(32)
    x = F.leaky_relu(x, inplace=True)
    # x = F.relu(v(x), inplace=True)
    # x = F.dropout(x, 0.5) # 5 11 17
    if k ==1:
        sources.append(x)
    elif k ==3:
        sources.append(x)
    elif k ==5:
        sources.append(x)

```

# 주요 개발 내용(#3)

## A. 이미지의 전처리(image size 300->512)

-> SSD: Single Shot MultiBox Detector Paper 참고하여 300을 512로 수정

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

# 주요 개발 내용(#3)

## A. 이미지의 전처리(image size 300->512)

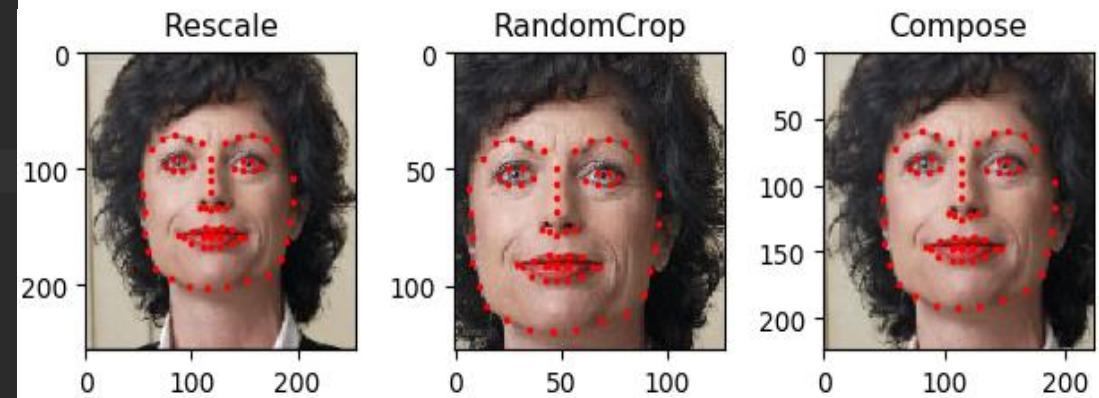
```
# SSD512 설정
ssd_cfg = [
    'num_classes': 21, # 배경 클래스를 포함한 총 클래스 수
    'input_size': 512, # 이미지의 입력 크기 #####
    'bbox_aspect_num': [4, 6, 6, 6, 6], # 셀 당 DBox의 개수
    'feature_maps': [64, 32, 16, 8, 4], # 각 source 피처맵크기
    'steps' : [8, 16, 32, 64, 128],
    'min_sizes': [20, 51, 133, 215, 296], # DBOX의 크기(최소)
    'max_sizes': [51, 133, 215, 296, 378], # DBOX의 크기(최대)
    'aspect_ratios': [[2], [2, 3], [2, 3], [2, 3], [2, 3]],
```

```
layers += [nn.Conv2d(in_channels, cfg[0], kernel_size=(1))]
layers += [nn.Conv2d(cfg[0], cfg[1], kernel_size=(3), stride=2, padding=1)]
layers += [nn.Conv2d(cfg[1], cfg[2], kernel_size=(1))]
layers += [nn.Conv2d(cfg[2], cfg[3], kernel_size=(3), stride=2, padding=1)]
layers += [nn.Conv2d(cfg[3], cfg[4], kernel_size=(1))]
layers += [nn.Conv2d(cfg[4], cfg[5], kernel_size=(3), stride=2, padding=1)]
layers += [nn.Conv2d(cfg[5], cfg[6], kernel_size=(1))]
layers += [nn.Conv2d(cfg[6], cfg[7], kernel_size=(3))]
```

# 주요 개발 내용(#3)

B. RandomSampleCrop을 적용 시 객체가 잘리는 것을 방지하기 위해  
 ->Data Augmentation 기법 중 RandomSampleCrop 삭제(mAP : 69.7->65 감소)

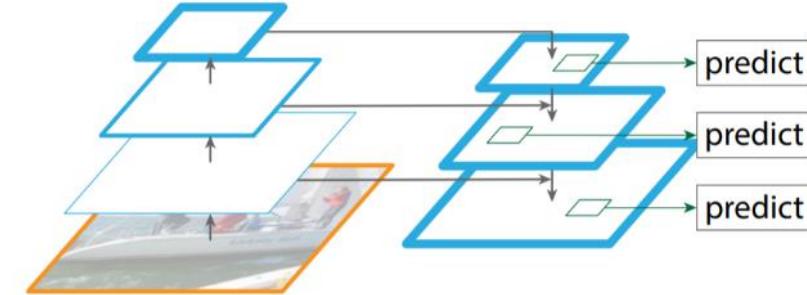
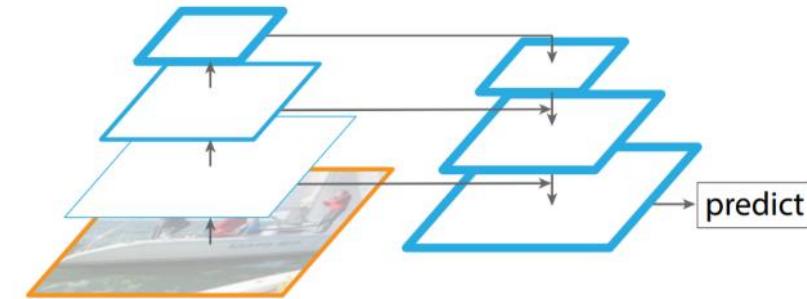
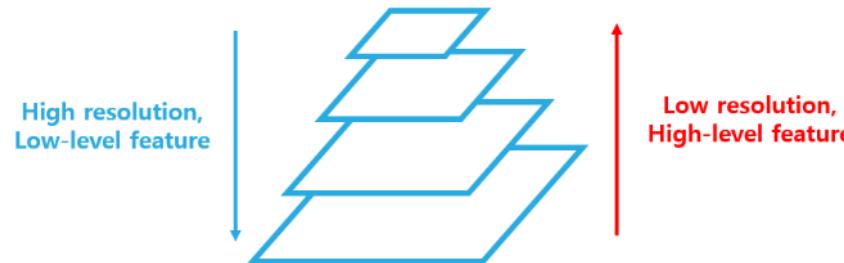
```
def __init__(self, input_size, color_mean):
    self.data_transform = {
        'train': Compose([
            ConvertFromInts(), # int를 float32로 변환
            ToAbsoluteCoords(), # 어노테이션 데이터의 규격화를 반환
            PhotometricDistort(), # 이미지의 색조 등을 임의로 변화시킴
            Expand(color_mean), # 이미지의 캔버스를 확대
            #RandomSampleCrop(), # 이미지 내의 특정 부분을 무작위로 추출
            RandomMirror(), # 이미지를 반전시킨다
            ToPercentCoords(), # 어노테이션 데이터를 0~1로 규격화
            Resize(input_size), # 이미지 크기를 input_size x input_size로 변형
            SubtractMeans(color_mean) # BGR 색상의 평균값을 뺀다
        ]),
        'val': Compose([
            ConvertFromInts(), # intをfloatに変換
            Resize(input_size), # 이미지 크기를 input_size x input_size로 변형
            SubtractMeans(color_mean) # BGR 색상의 평균값을 뺀다
        ])
    }
```



# 주요 개발 내용(#3)- FPN (Feature Pyramid Network)

FPN 구조를 Resnet이 아닌 VGG에 적용

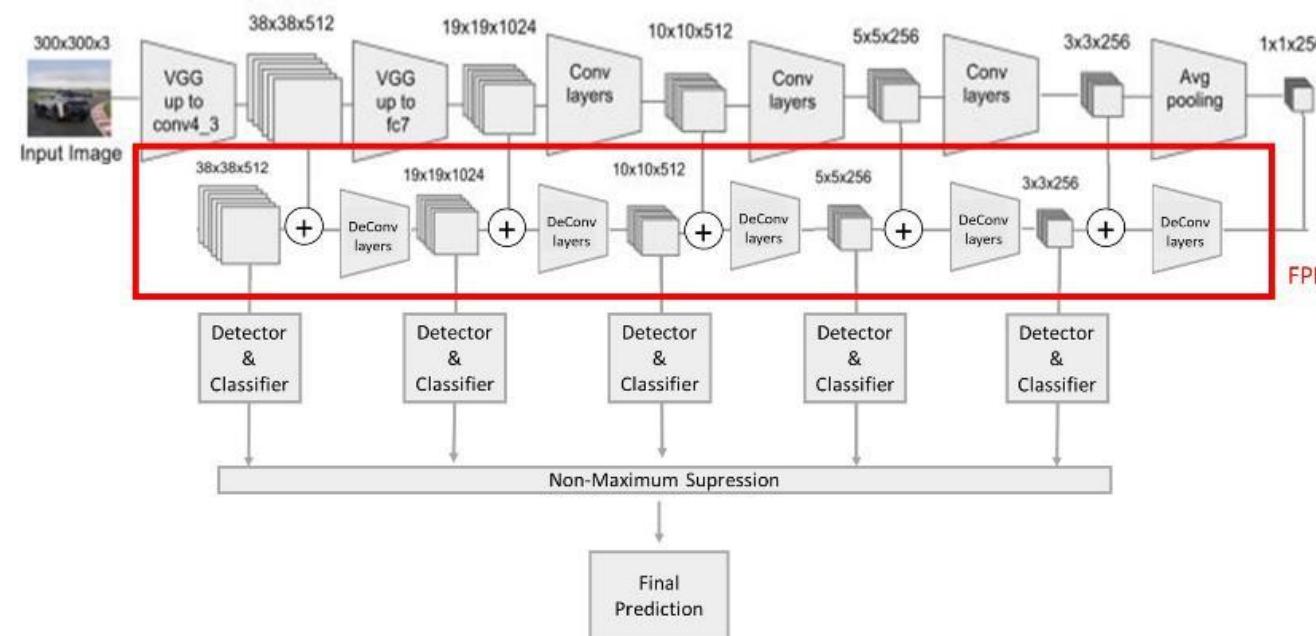
SSD-300	SSD-300+FPN-38	SSD-300+FPN-75	SSD-300+FPN-150
77.4 %	78.45 %	78.95%	79.03 %



VGG의 8층과 15층의 Layer을 추출하여 상위 High resolution으로 사용

# 주요 개발 내용(#3)- FPN (Feature Pyramid Network)

일반 Vanilla SSD 300 보다 4mAP 높은 수치



마지막 Feature map을 제거한 후 지금까지 개발한 모델과의 성능 비교

# 주요 개발 내용

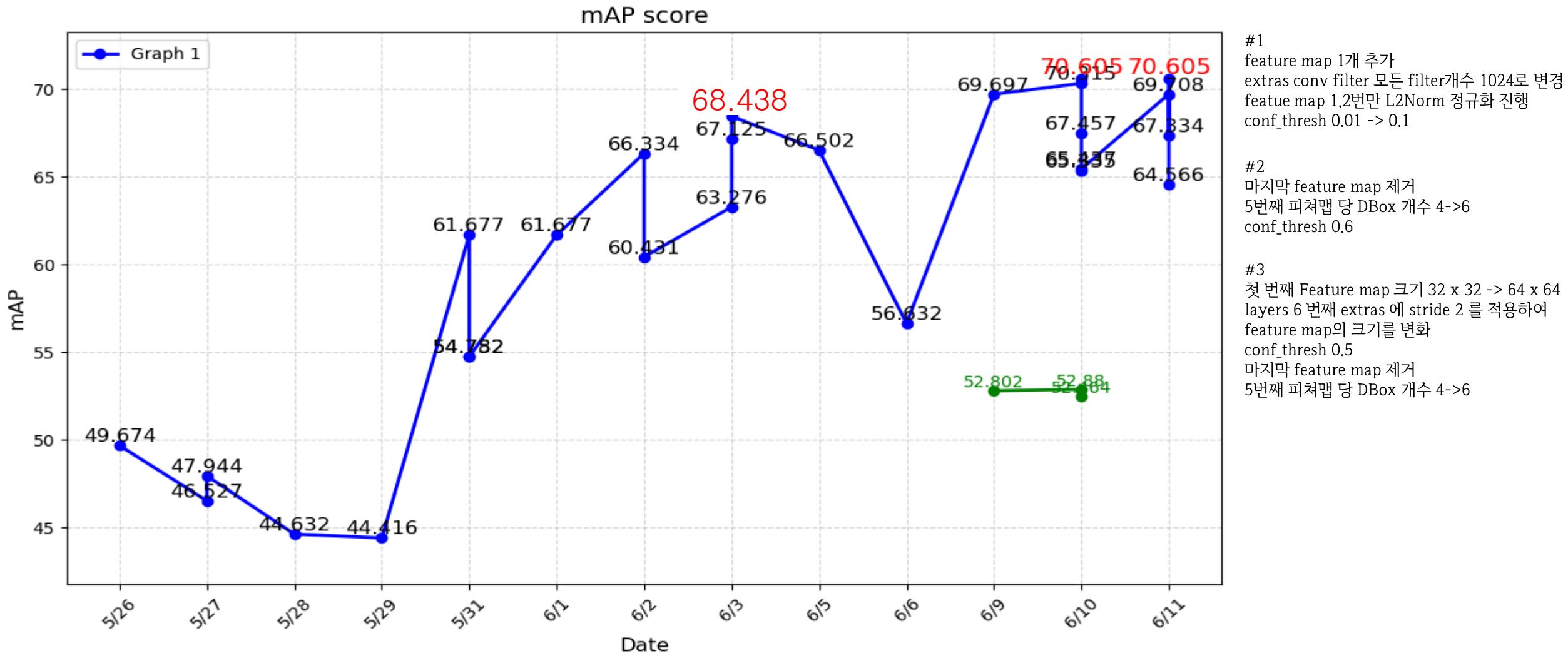
D. object detection을 위한 팀만의 차별화된 인공신경망의 구조 (mAP:72.179)

- 마지막 feature map 제거 + 5번 feature map 당 Dbox 4->6으로 변경
- conf\_thresh 0.6



C3:  $3 \times 3$  kernel의 convolution layer  
 C1:  $1 \times 1$   $3 \times 3$  kernel의 convolution layer  
 R: Relu activation function  
 P: Max Pooling layer

# 날짜별 mAP 그래프



# 프로젝트 일정 및 팀원 역할 분담

## 프로젝트 일정

5/11~5/17 : 프로젝트 문제 파악 및 SSD 알고리즘 모델에 대한 분석

5/18~5/24 : mAP 성능을 높이기 위한 다양한 방법 제시 및 모델 수정 시작

5/25~5/31 : 실습실 컴퓨터를 이용하여 epoch 높여서 테스트 시작

6/1~6/11 : 최적의 모델 선정 및 test , PPT자료 만들기

## 팀원 역할 분담

김다빈 - 모델 수정 및 테스트(extras 계층 filter 개수 추가 및 hyper parameter 조정)

김태우 - 모델 수정 및 테스트(이미지 전처리, 손실 함수를 focal loss함수로 수정)

이지훈 - 모델 수정 및 테스트(extras 계층 크기 추가(extras 계층을 backbone구조와 동일하게 구현)

및 Bbox 개수 늘리기)



# Thank you

출처

<https://herbwood.tistory.com/15>

<https://hugrypiggykim.com/2020/07/19/object-detection%EC%9D%98-%EB%B3%80%EC%B2%9C%EC%82%AC-%EB%85%BC%EB%AC%B8-review/>

<https://bkshin.tistory.com/entry/%EB%85%BC%EB%AC%B8-%EB%A6%AC%EB%B7%BO-SSDSingle-Shot-MultiBox-Detector-%ED%86%BA%EC%95%84%EB%B3%B4%EA%B8%B0>

[https://gaussian37.github.io/dl-concept-focal\\_loss/](https://gaussian37.github.io/dl-concept-focal_loss/)