YOLOv4

: Optimal Speed and Accuracy of Object Detection

IVPG Lab Seminar 2022.05.18

세종대학교 지능기전공학부

18학번 장윤정

Introduction

〈Background〉

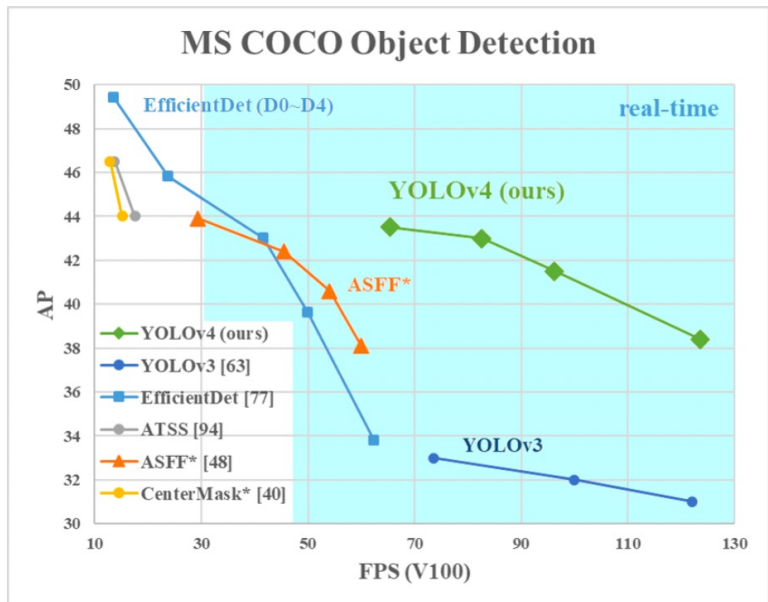
- 정확한 최신 모델들은 real-time으로 작동할 수 없으며, 큰 mini-batch로 학습하기 위해 많은 GPU가 필요함.
- 이러한 object detector들은 추천 시스템에만 적용할 수 있고, 자동차 충돌 경고 같은 real-time 시스템에는 더 빠른 모델이 필요함.

〈Goal〉

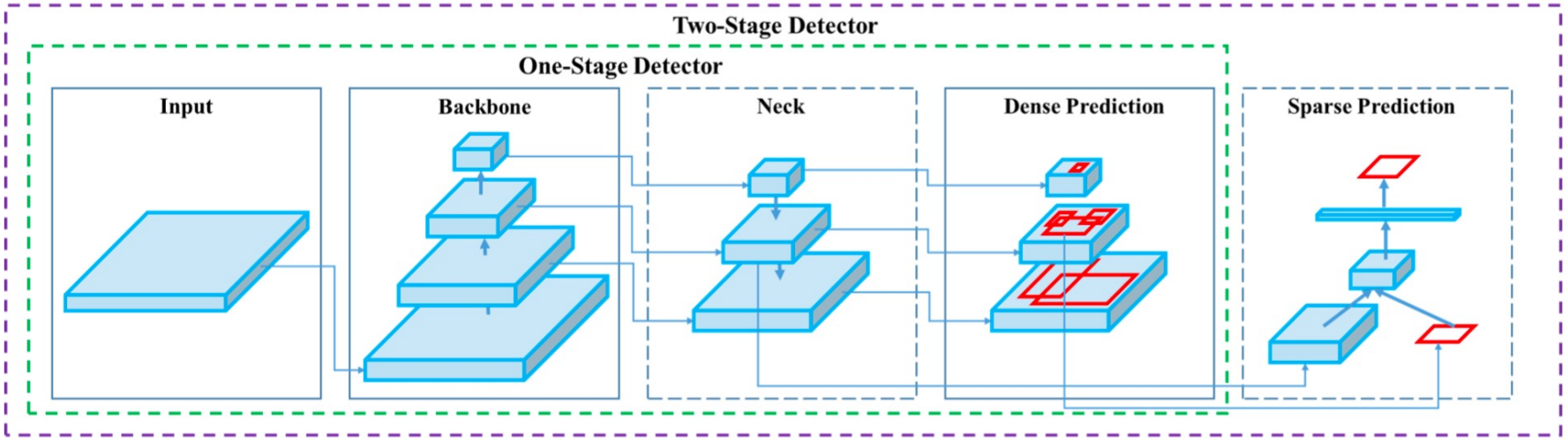
- The main goal of this work is designing a fast-operating speed of an object detector in production systems and optimization for parallel computations, rather than the low computation volume theoretical indicator (BFLOP).
  : object detector의 빠른 연산 속도와 병렬 계산을 위한 최적화를 설계하는 것! → 하나의 GPU로만 학습 가능한 real-time object detector가 목표!

〈Contribution〉

- 누구나 1080Ti, 2080Ti 정도의 GPU 하나만 있으면 빠르고 정확한 object detector 학습 가능
- object detector 학습에 있어서 “Bag-of-Freebies” 및 “Bag-of-Specials” 방법론들의 영향 확인
- CBN, PAN, SAM 등의 SOTA 방법론들을 수정하여 효율적인 학습 가능



Related work - Object detection models



- Backbone : **Detector가 GPU에서 동작하는지 / CPU에서 동작하는지에 따라 Backbone 구분 가능**
  - GPU platform : VGG, ResNet, DenseNet, EfficientNet-B0/B7, CSPResNeXt50, CSPDarknet53 등
  - CPU platform : SqueezeNet, MobileNet, ShuffleNet 등
- Neck : **Backbone과 Head 사이에서 다양한 크기의 feature map을 수집하는데 사용됨**
  - Additional blocks : SPP(Spatial Pyramid Pooling), ASPP, RFB, SAM(Self Attention Module) 등
  - Path-aggregation blocks : FPN, PAN(Path Augmented Network), NAS-FPN, BiFPN 등
- Head : **one-stage detector와 two-stage detector로 구분 가능**
  - Anchor-based Two-stage detector : fast R-CNN, faster R-CNN, R-FCN, Libra R-CNN 등 R-CNN 시리즈
  - Anchor-free Two-stage detector : RepPoints 등
  - Anchor-based One-stage detector : YOLO 시리즈, SSD, RetinaNet 등
  - Anchor-free One-stage detector : CenterNet, CornerNet, FCOS 등

Related work - Bag of Freebies

Bag of Freebies(BoF)

- Inference에 드는 cost는 증가시키지 않으면서 성능 향상
- Training 전략만 바꾸거나, Training cost만 증가시키는 방법들

Data augmentation

- Photometric distortion
- Geometric distortion
- Simulating object occlusion
  - Image level : Random erase, CutOut, Hide-and-seek, Grid mask
  - Feature map level : DropOut, DropConnect, DropBlock
- Using multiple images
  - MixUp, CutMix
- GAN

Semantic distribution bias

- Class Imbalance
  - Two-stage detector
    : Hard negative example mining, OHEM
  - One-stage detector
    : Focal loss
- One-hot hard representation
  - Label smoothing

Objective function

- Anchor-based method
  - MSE(Mean Squared Error)
- IoU-based method
  - GIoU(Generalized IoU)
  - DIoU(Distance IoU)
  - CIoU(Complete IoU)

Related work - Bag of Freebies

Bag of Freebies(BoF)

- Inference에 드는 cost는 증가시키지 않으면서 성능 향상
- Training 전략만 바꾸거나, Training cost만 증가시키는 방법들

Data augmentation

이미지의 가변성을 증가시켜서 모델이 다른 환경에서 얻은 이미지에 대해 더 견고해지도록

- Photometric distortion
- Geometric distortion
- Simulating object occlusion
  - Image level : Random erase, CutOut, Hide-and-seek, Grid mask
  - Feature map level : DropOut, DropConnect, DropBlock
- Using multiple images
  - MixUp, CutMix
- GAN

Semantic distribution bias

- Class Imbalance
  - Two-stage detector
    : Hard negative example mining, OHEM
  - One-stage detector
    : Focal loss
- One-hot hard representation
  - Label smoothing

Objective function

- Anchor-based method
  - MSE(Mean Squared Error)
- IoU-based method
  - GIoU(Generalized IoU)
  - DIoU(Distance IoU)
  - CIoU(Complete IoU)

Related work - Bag of Freebies

Bag of Freebies(BoF)

- Inference에 드는 cost는 증가시키지 않으면서 성능 향상
- Training 전략만 바꾸거나, Training cost만 증가시키는 방법들

Data augmentation

- Photometric distortion
- Geometric distortion
- Simulating object occlusion
  - Image level : Random erase, C
  - Feature map level : DropOut,
- Using multiple images
  - MixUp, CutMix
- GAN

이미지의 밝기, 대비, 색조, 채도 조절, 노이즈 추가

Semantic distribution bias

- Class Imbalance
  - Two-stage detector
    : Hard negative example mining, OHEM

- MSE(Mean Squared Error)
- IoU-based method
  - GIoU(Generalized IoU)
  - DIoU(Distance IoU)
  - CIoU(Complete IoU)

(그림 출처 : https://herbwood.tistory.com/24)

Related work - Bag of Freebies

Bag of Freebies(BoF)

- Inference에 드는 cost는 증가시키지 않으면서 성능 향상
- Training 전략만 바꾸거나, Training cost만 증가시키는 방법들

Data augmentation

- Photometric distortion
- Geometric distortion
- Simulating object occlusion
  - Image level : Random erase, C
  - Feature map level : DropOut,
- Using multiple images
  - MixUp, CutMix
- GAN

이미지 랜덤하게 크기 변경, 일부분 자르기, 뒤집기, 회전

Semantic distribution bias

- Class Imbalance
  - Two-stage detector
    : Hard negative example mining, OHEM

- MSE(Mean Squared Error)
- IoU-based method
  - GIoU(Generalized IoU)
  - DIoU(Distance IoU)
  - CIoU(Complete IoU)

(그림 출처 : https://herbwood.tistory.com/24)

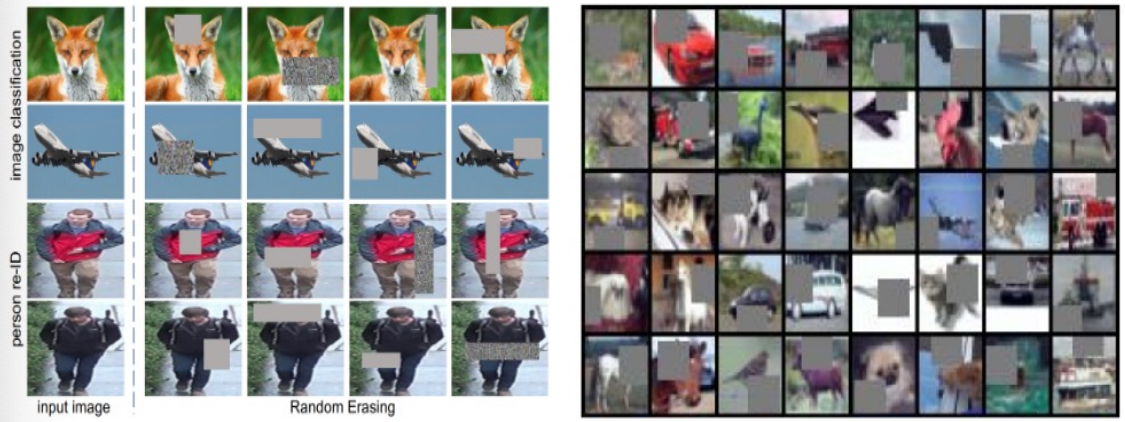
Related work - Bag of Freebies

Bag of Freebies(BoF)

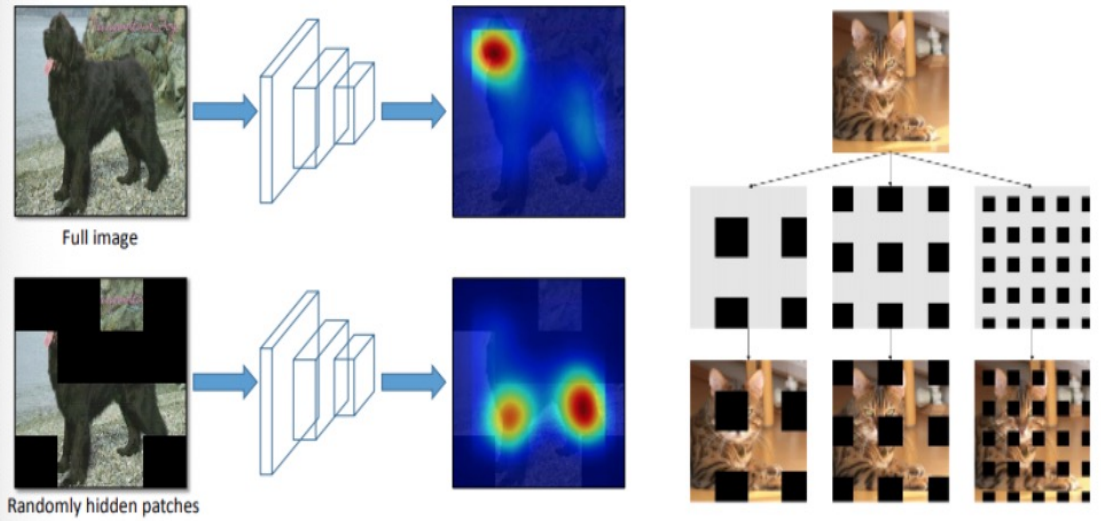
- Inference에 드는 cost는 증가시키지 않으면서 성능 향상
- Training 전략만 바꾸거나, Training cost만 증가시키는 방법들

Data augmentation

- Photometric distortion
- Geometric distortion
- Simulating object occlusion → 객체가 겹치는 상황을 가정하고, 이런 상황에 모델이 잘 대처하도록 하기위해 사용하는 방법
  - Image level : Random erase, CutOut, Hide-and-seek, Grid mask
  - Feature map level : DropOut, DropConnect, DropBlock
- Using multiple images
  - MixUp, CutMix
- GAN



[그림 24] Random Erase(좌), Cutout(우)



[그림 25] Hide-and-Seek(좌), Grid Mask(우)

(그림 출처 : https://herbwood.tistory.com/24)

Related work - Bag of Freebies

Bag of Freebies(BoF)

- Inference에 드는 cost는 증가시키지 않으면서 성능 향상
- Training 전략만 바꾸거나, Training cost만 증가시키는 방법들

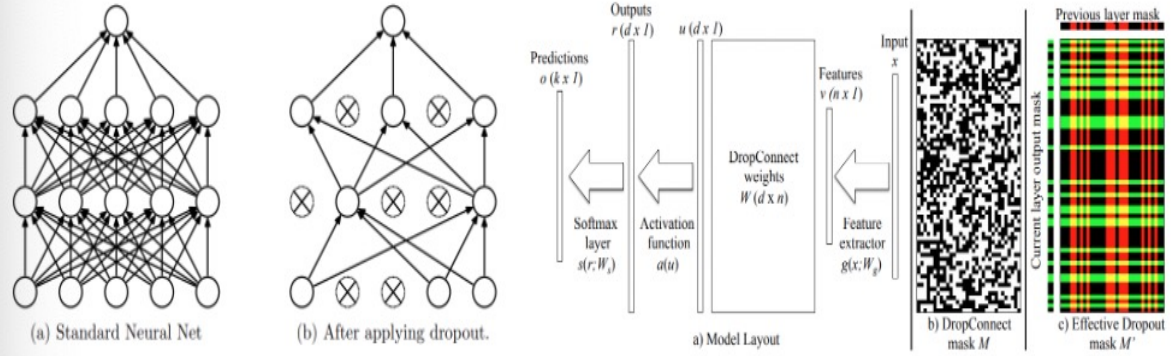
Data augmentation

- Photometric distortion
- Geometric distortion
- Simulating object occlusion

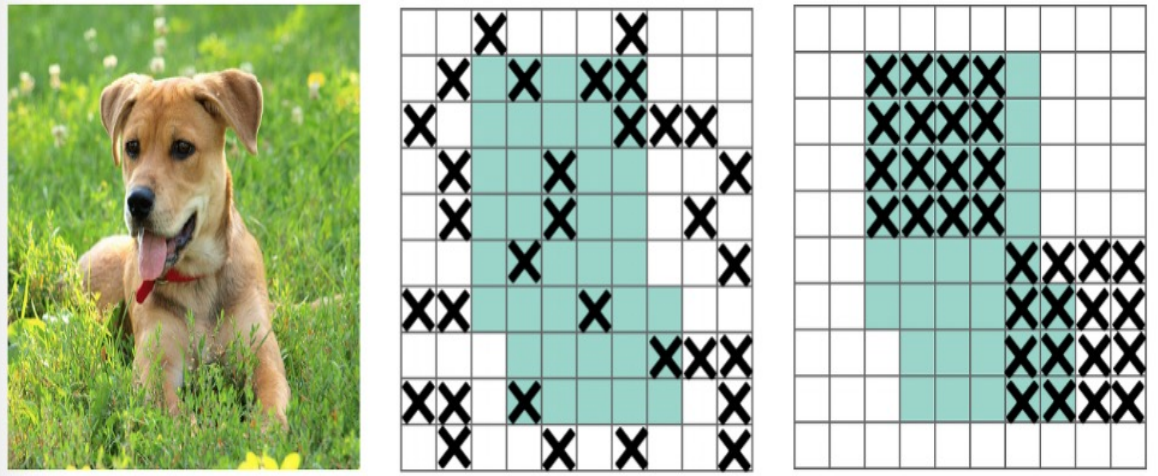
  객체가 겹치는 상황을 가정하고, 이런 상황에 모델이 잘 대처하도록 하기위해 사용하는 방법

  - Image level : Random erase, CutOut, Hide-and-seek, Grid mask
  - Feature map level : DropOut, DropConnect, DropBlock

    Regularization!
- Using multiple images
  - MixUp, CutMix
- GAN



[그림 26] Dropout(좌), DropConnect(우)



[그림 27] DropBlock

(그림 출처 : https://herbwood.tistory.com/24)

Related work - Bag of Freebies

Bag of Freebies(BoF)

- Inference에 드는 cost는 증가시키지 않으면서 성능 향상
- Training 전략만 바꾸거나, Training cost만 증가시키는 방법들

Data augmentation

- Photometric distortion
- Geometric distortion
- Simulating object occlusion
  - Image level : Random erase, CutOut, Hide-and-seek, Grid mask
  - Feature map level : DropOut, DropConnect, DropBlock
- Using multiple images → 다수의 이미지를 함께 사용
  - MixUp, CutMix
- GAN

Semantic distribution bias

- Class Imbalance
  - Two-stage detector
    : Hard negative example mining, OHEM
  - One-stage detector
    : Focal loss
- One-hot hard representation
  - Label smoothing

Objective function

- thod
  uared Error)
  d
  zed IoU)
  IoU)
  e IoU)

(그림 출처 : https://herbwood.tistory.com/24)

Related work - Bag of Freebies

Bag of Freebies(BoF)

- Inference에 드는 cost는 증가시키지 않으면서 성능 향상
- Training 전략만 바꾸거나, Training cost만 증가시키는 방법들

Data augmentation

- Photometric distortion
- Geometric distortion
- Simulating object occlusion
  - Image level : Random erase, CutO
  - Feature map level : DropOut, Drop
- Using multiple images
  - MixUp, CutMix
- GAN → Texture bias를 해결하기 위해 texture만 변형시킨 이미지

Semantic distribution bias

- Class Imbalance
  - Two-stage detector
    : Hard negative example mining, OHEM
  - One-stage detector
    : Focal loss
- One-hot hard representation
  - Label smoothing

(그림 출처 : https://herbwood.tistory.com/24)

Related work - Bag of Freebies

Bag of Freebies(BoF)

- Inference에 드는 cost는 증가시키지 않으면서 성능 향상
- Training 전략만 바꾸거나, Training cost만 증가시키는 방법들

Data augmentation

- Photometric distortion
- Geometric distortion
- Simulating object occlusion
  - Image level : Random erase, CutOut, Hide-and-seek, Grid mask
  - Feature map level : DropOut, DropConnect, DropBlock
- Using multiple images
  - MixUp, CutMix
- GAN

Semantic distribution bias

- Class Imbalance → Positive/Negative sample 사이의 class imbalance 문제를 해결하는 방법
  - Two-stage detector
    : Hard negative example mining, OHEM
  - One-stage detector
    : Focal loss → Easy negative는 down-weight 해서 hard negative sample을 집중적으로 학습할 수 있게 해주는 loss function
- One-hot hard repre
  - Label smoothing

Objective function

- Anchor-based method
  - MSE(Mean Squared Error)
- IoU-based method
  - GIoU(Generalized IoU)
  - DIoU(Distance IoU)
  - CIoU(Complete IoU)

Related work - Bag of Freebies

Bag of Freebies(BoF)

- Inference에 드는 cost는 증가시키지 않으면서 성능 향상
- Training 전략만 바꾸거나, Training cost만 증가시키는 방법들

Data augmentation

- Photometric distortion
- Geometric distortion
- Simulating object occlusion
  - Image level : Random erase, CutOut, Hide-and-seek, Grid mask
  - Feature map level : DropOut, DropConnect, DropBlock
- Using multiple images
  - MixUp, CutMix
- GAN

Semantic distribution bias

- Class Imbalance
  - Two-stage detector
    : Hard negat
  - One-stage de
    : Focal loss
- One-hot hard representation
  - Label smoothing

서로 다른 카테고리가 어느 정도의 관계를 가지는지 표현하기 어렵다는 문제 (주로 labeling 시에 발생)

Label을 0,1이 아니라 smooth하게 부여

Objective function

- Anchor-based method
  - MSE(Mean Squared Error)
- IoU-based method
  - GIoU(Generalized IoU)
  - DIoU(Distance IoU)
  - CIoU(Complete IoU)

Related work - Bag of Freebies

Bag of Freebies(BoF)

- Inference에 드는 cost는 증가시키지 않으면서 성능 향상
- Training 전략만 바꾸거나, Training cost만 증가시키는 방법들

Data augmentation

- Photometric distortion
- Geometric distortion
- Simulating object occlusion
  - Image level : Random er
  - Feature map level : DropOut, DropConnect, DropBlock
- Using multiple images
  - MixUp, CutMix
- GAN

Semantic distribution bias

- Class Imbalance
  - Two-stage detector
    : Hard negative example mining, OHEM
  - One-stage detector
    : Focal loss
- One-hot hard representation
  - Label smoothing

Bounding box regression을 수행하기 위한 loss function

Objective function

- Anchor-based method
  - MSE(Mean Squared Error)
- IoU-based method
  - GIoU(Generalized IoU) 두 객체 사이의 거리 고려
  - DIoU(Distance IoU) 두 객체의 중심까지의 거리 추가
  - CIoU(Complete IoU) 두 객체 사이의 종횡비 추가

Related work - Bag of Specials

Bag of Specials(BoS)

- Inference에 드는 cost를 약간 증가시키지만, 정확도 크게 향상
- plugin modules(추가 모듈) 또는 post-processing(후처리) 방법

Enhance receptive field

- SPP(Spatial Pyramid Pooling)
- ASPP(Astrous Pyramid Pooling)
- RFB(Receptive Field Block)

Attention module

- SE(Squeeze-and-Excitation)
- SAM(Self-Attention Module)

Feature integration

- SFAM(Scale-wise Feature Aggregation Module)
- ASFF(Adaptively Spatial Feature Fusion)
- BiFPN

Activation function

- LReLU(Leaky ReLU)
- PReLU(Parametric ReLU)
- ReLU6
- SELU(Scaled Exponential Linear Unit)
- Swish
- hard-Swish
- Mish

Post-processing method

- Greedy NMS
- Soft NMS
- DIoU NMS

Related work - Bag of Specials

Bag of Specials(BoS)

- Inference에 드는 cost를 약간 증가시키지만, 정확도 크게 향상
- plugin modules(추가 모듈) 또는 post-processing(후처리) 방법

Enhance receptive field

Backbone에서 얻은 feature map에 대한 receptive field를 키우는 방법

- SPP(Spatial Pyramid Pooling)
- ASPP(Astrous Pyramid Pooling)
- RFB(Receptive Field Block)

le

- SE(Squeeze-and-Excitation)
- SAM(Self-Attention Module)

Activation function

- LReLU(Leaky ReLU)
- PReLU(Parametric ReLU)
- ReLU6
- SELU(Scaled Exponential Linear Unit)
- Swish
- hard-Swish
- Mish

Feature integration

- SFAM(Scale-wise Feature Aggregation Module)
- ASFF(Adaptively Spatial Feature Fusion)
- BiFPN

Post-processing method

- Greedy NMS
- Soft NMS
- DIoU NMS

Related work - Bag of Specials

Bag of Specials(BoS)

- Inference에 드는 cost를 약간 증가시키지만, 정확도 크게 향상
- plugin modules(추가 모듈) 또는 post-processing(후처리) 방법

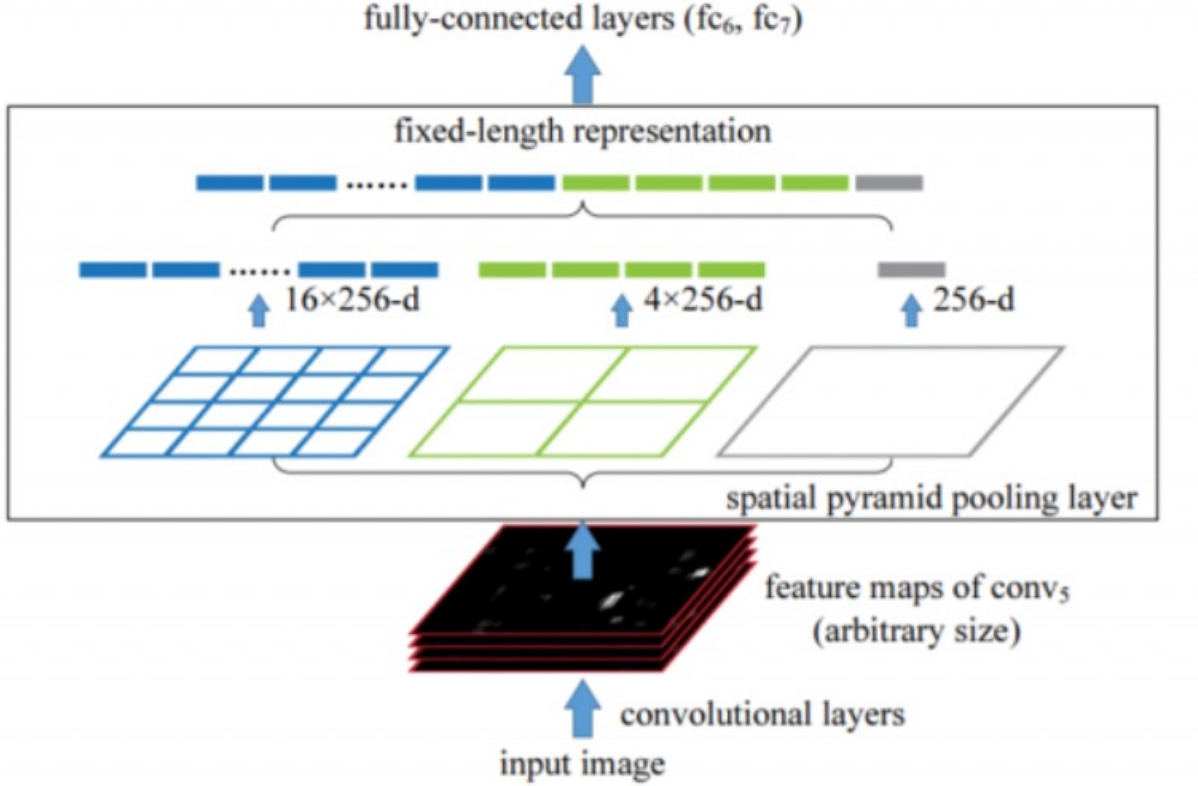
Enhance receptive field

- SPP(Spatial Pyramid Pooling)
- ASPP(Astrous Pyramid Pooling)
- RFB(Receptive Field Block)

conv layer의 마지막 feature map을 고정된 크기의 그리드로 분할해 max pooling하여 고정된 크기의 representation 벡터를 출력하는 방법

Attention

Activation function




```
input_data = convolutional(input_data, (1, 1, 1024, 1024), activate_type="mish")
input_data = convolutional(input_data, (1, 1, 1024, 512))
input_data = convolutional(input_data, (3, 3, 512, 1024))
input_data = convolutional(input_data, (1, 1, 1024, 512))

max_pooling_1 = tf.keras.layers.MaxPool2D(pool_size=13, padding='SAME', strides=1)(input_data)
max_pooling_2 = tf.keras.layers.MaxPool2D(pool_size=9, padding='SAME', strides=1)(input_data)
max_pooling_3 = tf.keras.layers.MaxPool2D(pool_size=5, padding='SAME', strides=1)(input_data)
input_data = tf.concat([max_pooling_1, max_pooling_2, max_pooling_3, input_data], axis=-1)

input_data = convolutional(input_data, (1, 1, 2048, 512))
input_data = convolutional(input_data, (3, 3, 512, 1024))
input_data = convolutional(input_data, (1, 1, 1024, 512))

return route_1, route_2, input_data
```

...ial Feature Fusion)

- Soft NMS
- DIoU NMS

Related work - Bag of Specials

Bag of Specials(BoS)

- Inference에 드는 cost를 약간 증가시키지만, 정확도 크게 향상
- plugin modules(추가 모듈) 또는 post-processing(후처리) 방법

Activation function

- LReLU(Leaky ReLU)
- PReLU(Parametric ReLU)
- ReLU6
- SELU(Scaled Exponential Linear Unit)
- Swish
- hard-Swish

Enhance receptive field

- SPP(Spatial Pyramid Pooling)
- ASPP(Astrous Pyramid Pooling)
- RFB(Receptive Field Block)

Attention module

- SE(Squeeze-and-Excitation)
- SAM(Self-Attention Module)

Feature에서 상대적으로 중요한 값을 강조하여
매우 적은 연산량 증가로 정확도 향상 가능

Feature integration

- SFAM(Scale-wise Feature Aggregation Module)
- ASFF(Adaptively Spatial Feature Fusion)
- BiFPN

Post-processing method

- Greedy NMS
- Soft NMS
- DIoU NMS

Related work - Bag of Specials

Bag of Specials(BoS)

- Inference에 드는 cost를 약간 증가시키지만, 정확도 크게 향상
- plugin modules(추가 모듈) 또는 post-processing(후처리) 방법

Enhance receptive field

- SPP(Spatial Pyramid Pooling)
- ASPP(Astrous Pyramid Pooling)
- RFB(Receptive Field Block)

Attention module

- SE(Squeeze-and-Excitation)
- SAM(Self-Attention Module)

Activation function

- LReLU(Leaky ReLU)
- PReLU(Parametric ReLU)
- ReLU6
- SELU(Scaled Exponential Linear Unit)
- Swish
- hard-Swish
- Mish

Feature integration

Feature map을 통합하기 위해 연구된 방법들
(YOLOv3의 FPN보다 경량화)

- SFAM(Scale-wise Feature Aggregation Module)
- ASFF(Adaptively Spatial Feature Fusion)
- BiFPN

ethod

- Greedy NMS
- Soft NMS
- DIoU NMS

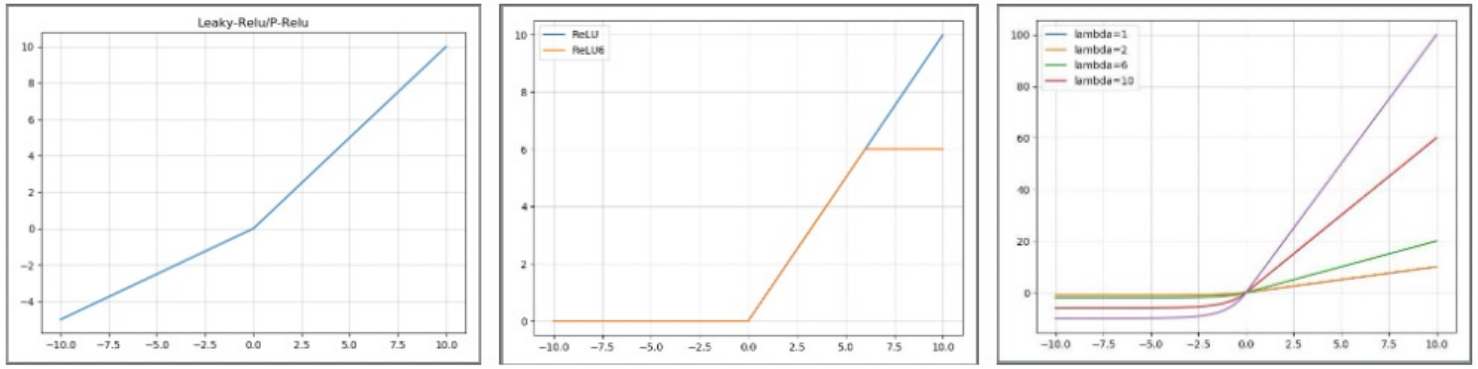
Related work - Bag of Specials

좋은 활성화 함수는 gradient를 효율적으로 전파, cost 적게 발생

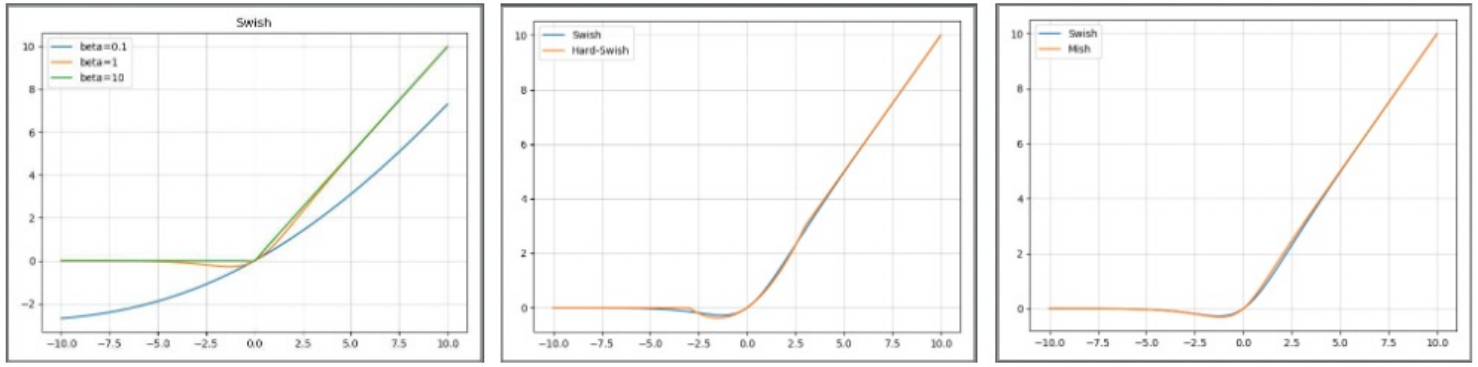
Activation function

- LReLU(Leaky ReLU)
- PReLU(Parametric
- ReLU6 최대값 6
- SELU(S Self-normalizing 효과 Unit)
- Swish 깊은 layer 학습 때 좋은 성능을 보임
- hard-Swish 연산량 down
- Mish 약간의 음수를 허용하여 gradient가 잘 흐르도록 설계

Dying ReLU 해결을 위해 음수값에 대해 gradient가 0이 되지 않도록 작은 값을 곱함



[그림 35] Leaky ReLU/PReLU(좌), ReLU6(중간), SeLU(우) 그래프



[그림 36] Swish(좌), hard Swish(중간), Mish(우) 그래프

Post-processing method

- Greedy NMS
- Soft NMS
- DIoU NMS

(그림 출처 : https://herbwood.tistory.com/24)

Related work - Bag of Specials

Bag of Specials(BoS)

- Inference에 드는 cost를 약간 증가시키지만, 정확도 크게 향상
- plugin modules(추가 모듈) 또는 post-processing(후처리) 방법

Enhance receptive field

- SPP(Spatial Pyramid Pooling)
- ASPP(Astrous Pyramid Pooling)
- RFB(Receptive Field Block)

Attention module

- SE(Squeeze-and-Excitation)
- SAM(Self-Attention Module)

Activation function

- LReLU(Leaky ReLU)
- PReLU(Parametric ReLU)
- ReLU6
- SELU(Scaled Exponential Linear Unit)
- Swish
- hard-Swish
- Mish

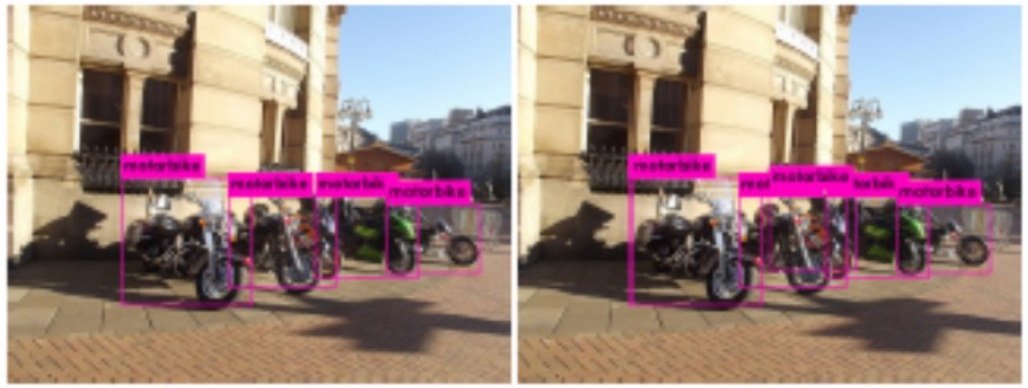
Feature i

Post-processing method

동일한 객체를 중복 예측하는 bbox를 필터링하는 NMS 방법들

- Greedy NMS
- Soft NMS
- DIoU NMS

기존 NMS threshold에 DIoU panalty term을 추가하여 겹친 객체에 대한 성능을 향상시킨 방법



$\mathcal{L}_{CIoU}$ + NMS

$\mathcal{L}_{CIoU}$ + DIoU-NMS

Related work - Bag of Specials

Bag of Specials(BoS)

- Inference에 드는 cost를 약간 증가시키지만, 정확도 크게 향상
- plugin modules(추가 모듈) 또는 post-processing(후처리) 방법

Enhance receptive field

- SPP(Spatial Pyramid Pooling)
- ASPP(Astrous Pyramid Pooling)
- RFB(Receptive Field Block)

Attention module

- SE(Squeeze-and-Excitation)
- SAM(Self-Attention Module)

Feature integration

- SFAM(Scale-wise Feature Aggregation Module)
- ASFF(Adaptively Spatial Feature Fusion)
- BiFPN

Activation function

- LReLU(Leaky ReLU)
- PReLU(Parametric ReLU)
- ReLU6
- SELU(Scaled Exponential Linear Unit)
- Swish
- hard-Swish
- Mish

Post-processing method

- Greedy NMS
- Soft NMS
- DIoU NMS

BoF와 BoS의 많은 방법들 중 최종적으로 선택한 방법은 무엇인지 + YOLOv4 저자들이 직접 개발한 방법은 무엇인지는 다음 시간에!

Methodology - Selection of architecture

Table 1: Parameters of neural networks for image classification.

Backbone model	Input network resolution	Receptive field size	Parameters	Average size of layer output (WxHxC)	BFLOPs (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	512x512	425x425	20.6 M	**1058 K**	31 (15.5 FMA)	62
CSPDarknet53	512x512	725x725	**27.6 M**	950 K	52 (26.0 FMA)	**66**
EfficientNet-B3 (ours)	512x512	**1311x1311**	12.0 M	668 K	11 (5.5 FMA)	26

Parameters, BFLOPs(연산량)이 큰데도 FPS가 좋게 나오는 이유는?
CSP(Cross Stage Partial) 구조 사용!
→ 컴퓨팅파워가 낮은 환경이 감당하기 힘든 연산량을 완화시켜줘서 Capacity 증가해도 감당 가능!

〈Backbone을 결정하는 3가지 조건〉

1) **Higher input network size (resolution)** : 여러 개의 작은 크기의 객체 탐지를 위해
2) **More layers** : 증가된 input network size를 커버하기 위한 Higher receptive field를 위해
3) **More parameters** : 단일 이미지에서 다양한 크기의 여러 객체를 탐지할 수 있는 모델의 더 큰 capacity를 위해

〈YOLOv4 Architecture〉

- **Backbone** : CSPDarknet53
- **Neck** : SPP(receptive field 증가 및 연산 속도 감소를 최소한으로 유지), PAN(YOLOv3의 FPN 대신)
- **Head** : YOLOv3의 anchor-based Head

감사합니다 :)

IVPG Lab Seminar 2022.05.18

세종대학교 지능기전공학부

18학번 장윤정