인공지능 기술의 대중화(AI Democratization)를 위한

TANGO 커뮤니티 제1회 컨퍼런스

TANGO

객체 검출 신경망 구조 탐색 기술

성명 김중헌 소속 고려대학교

주관 ETRI 한국전자통신연구원 (TANGO) 주최 과학기술정보통신부 IITP 정보통신기획평가원 후원 lablup weda tesla System 한국인공지능협회 SNUH 서울대학교병원 고려대학교 KOREA UNIVERSITY 홍익대학교 HONGIK UNIVERSITY CAU 중앙대학교 Georgia Tech

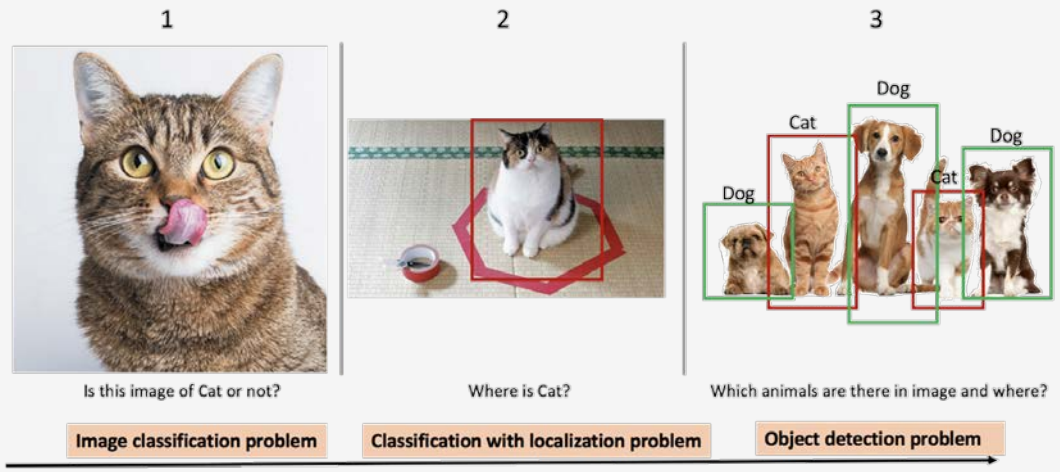
TANGO

목차

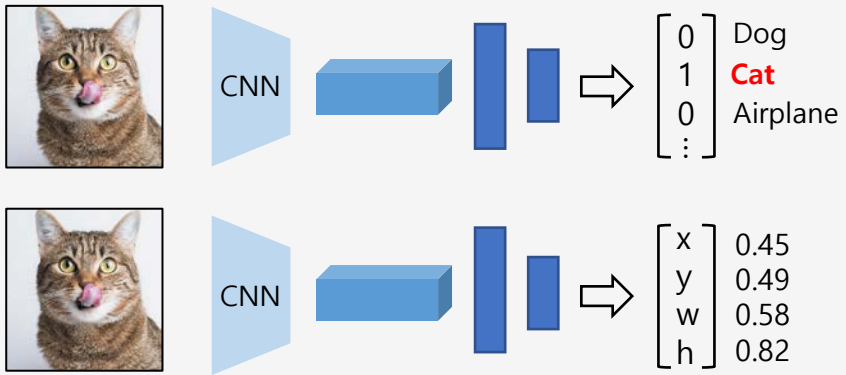
1. 객체 검출 신경망 – 객체 검출

객체 검출(Object Detection) 기술

- 컴퓨터 비전, 영상 처리 관련 기술
- 분류(classification): 이미지의 분류
  분류 + 지역화(localization): 이미지의 분류와 위치 예측
  검출(detection): 이미지 내의 객체 분류 및 위치 예측
- **다수의 객체**에 대한 분류 + 지역화



- 고전 객체 검출 알고리즘
  - 특징 공학(feature engineering) 활용
  - 피처 분포 경계 결정 (boundary decision)
- 딥러닝 활용 알고리즘
  - Convolutional Neural Network (CNN) 활용
  - Detection과 recognition 통합 처리
- 분류 + 회귀(regression) 문제
  - 클래스는 분류, 지역화는 위치 regression



GitBook, “https://deepbaksuvision.github.io/Modu_ObjectDetection/”

1. 객체 검출 신경망 – 신경망 구조

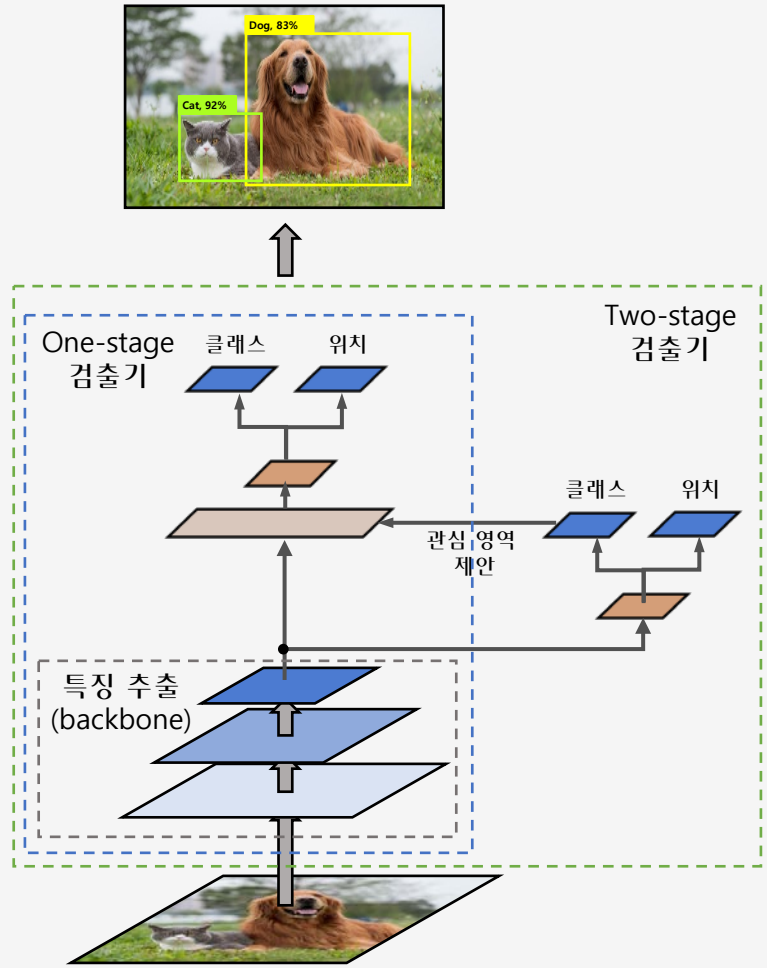
객체 검출 신경망 구조

- 영역 제안(region proposal): 객체가 있을 만한 영역 제안
  - Sliding window
  - Selective search
  - Region Proposal Network (RPN)

- Two-stage 검출기
  - 영역 제안 단계 → 분류 및 지역화 단계
  - R-CNN, Fast R-CNN, Faster R-CNN, …
  - 비교적 높은 정확도, 학습 및 추론 시간 지연

- One-stage 검출기
  - 영역 제안 단계 없이 분류 및 지역화
  - 균일한 grid + 앵커 박스(anchor box) 활용
  - YOLO, SSD, …
  - 비교적 낮은 정확도, 빠른 추론 시간

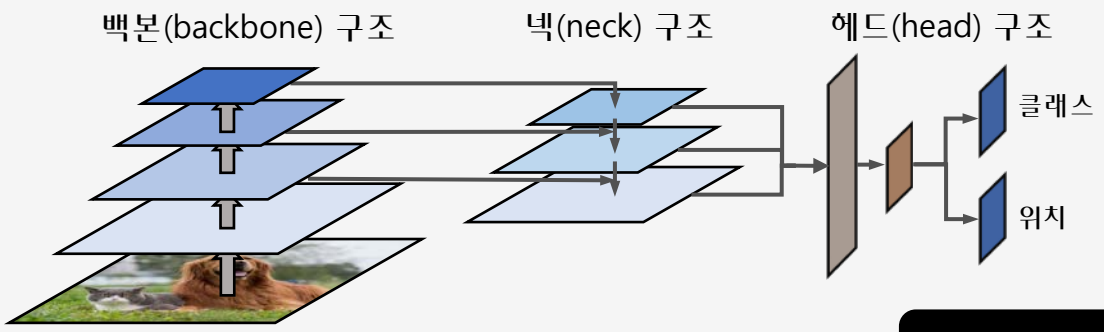
- 실시간성(real-time) 충족 가능한 One-stage 방식이 활발하게 연구



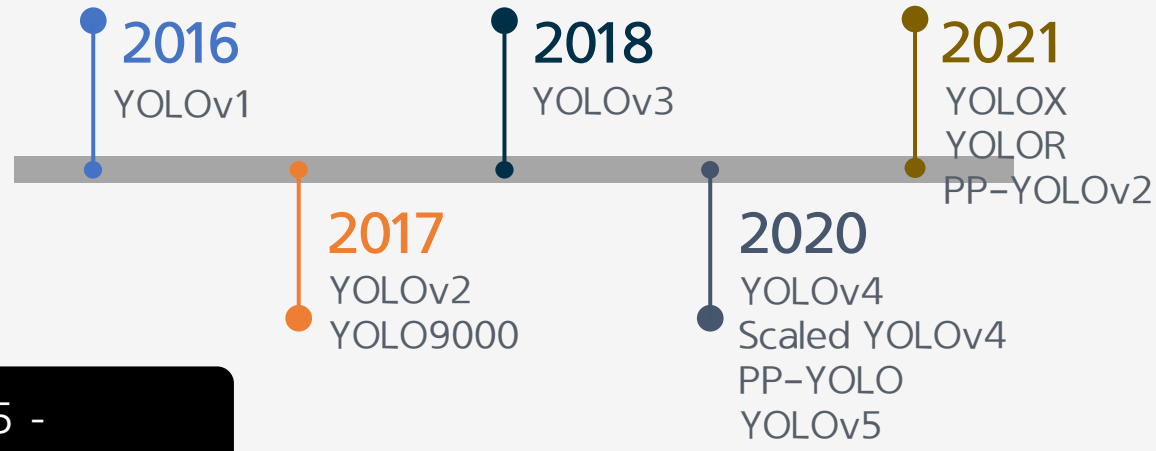
1. 객체 검출 신경망 - YOLO

You Only Look Once (YOLO)

- 2016년 CVPR, Joseph Redmond 공개
  - YOLOv3 이후 연구 커뮤니티 탈퇴
- 대표적인 One-stage 검출기
  - 단일 신경망으로 클래스 예측과 객체 위치 추론
  - 실시간(Real-time) 영상 처리 가능
- 신경망 구조
  - 백본 구조: 피처 추출, 상대적으로 크고 무거움
  - 넥 구조: 다양한 수준의 피처 가공
  - 헤드 구조: 클래스 분류, bounding box 추측



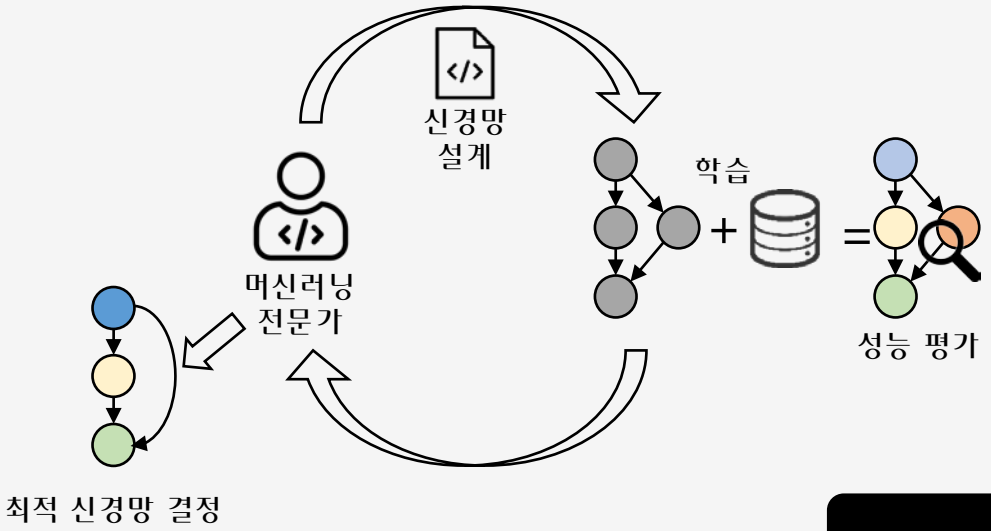
- YOLOv4
  - Bag of Freebies (BoF): 신경망 구조 변경 없이 학습 방법 개선으로 성능 향상 시도
  - Bag of Specials (BoS): 신경망 구조 변경으로 성능 향상 시도
- Cross-Stage-Partial-Connection 백본 구조 제안
  - 입력 채널의 절반만 신경망을 통과, 연산량 감소
- 다양한 연구진들이 YOLO 계열 신경망 연구 진행 중



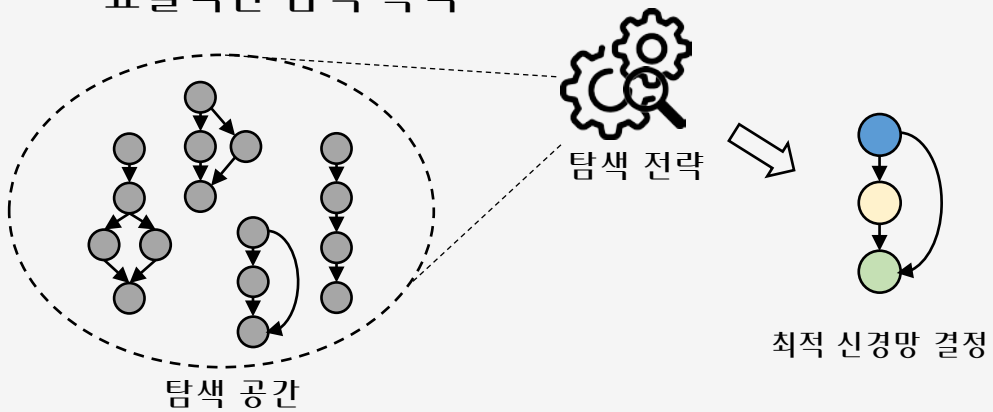
2. 신경망 구조 탐색 - 개요

신경망 구조 탐색(Neural Architecture Search)

- AutoML (Automated Machine Learning)
  - 반복적인 머신러닝 모델 개발 작업의 자동화
  - 데이터 준비, 피처 생성, 모델 생성, 모델 평가
- 심층 신경망 모델 생성 및 평가
  - 전문가의 지식에 의존적, 반복적인 작업 비용
  - 신경망 설계 → 학습 → 성능 평가 및 결정



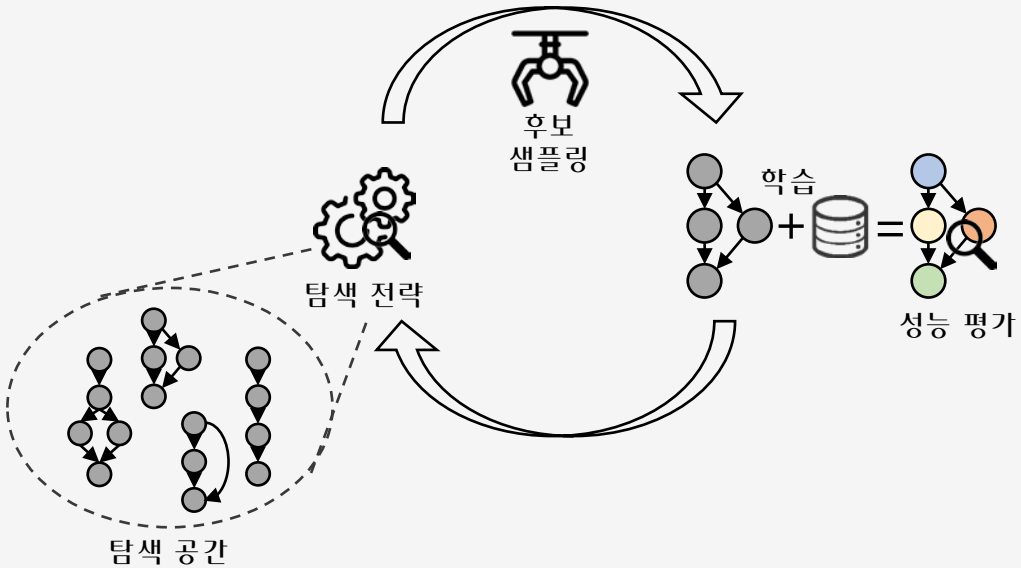
- 신경망 구조 탐색
  - 심층 신경망 구조 설계 과정의 자동화
  - 타겟 데이터/태스크 최적 구조 탐색 목적
- 탐색 공간(search space)
  - 후보 신경망 구조들의 집합
  - 활용 operation, 신경망 구성 규칙 정의
- 탐색 전략(search strategy)
  - 탐색 공간을 탐색하는 알고리즘
  - 효율적인 탐색 목적



2. 신경망 구조 탐색 - One-shot 탐색

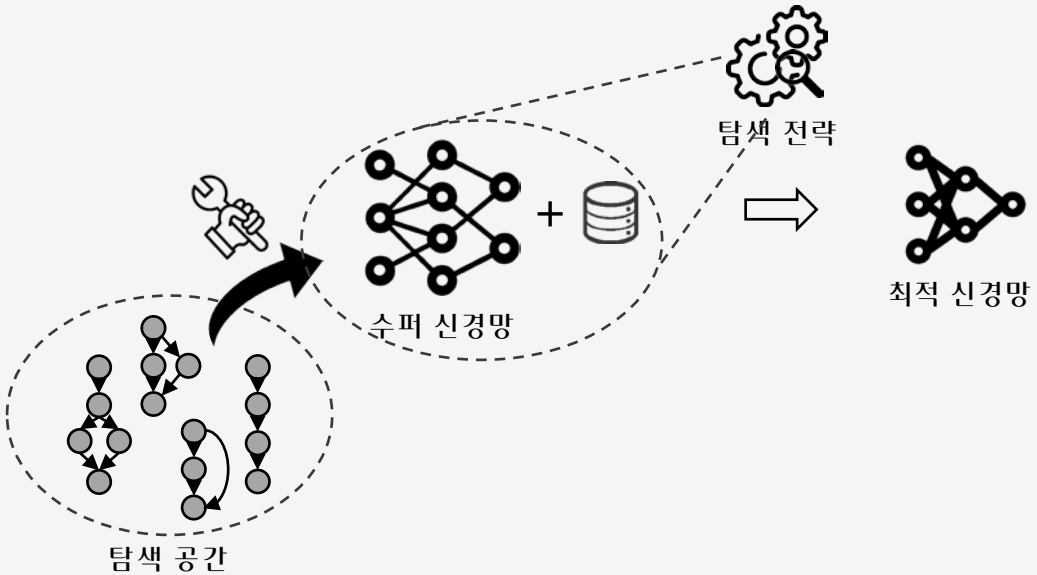
One-shot 탐색 전략

➢ 초기 신경망 구조 탐색 전략



- 반복적인 후보 신경망 구조 샘플링
- 학습 및 성능 평가, 최적 신경망 구조 결정
- 탐색 공간 크기에 비례하는 탐색 비용

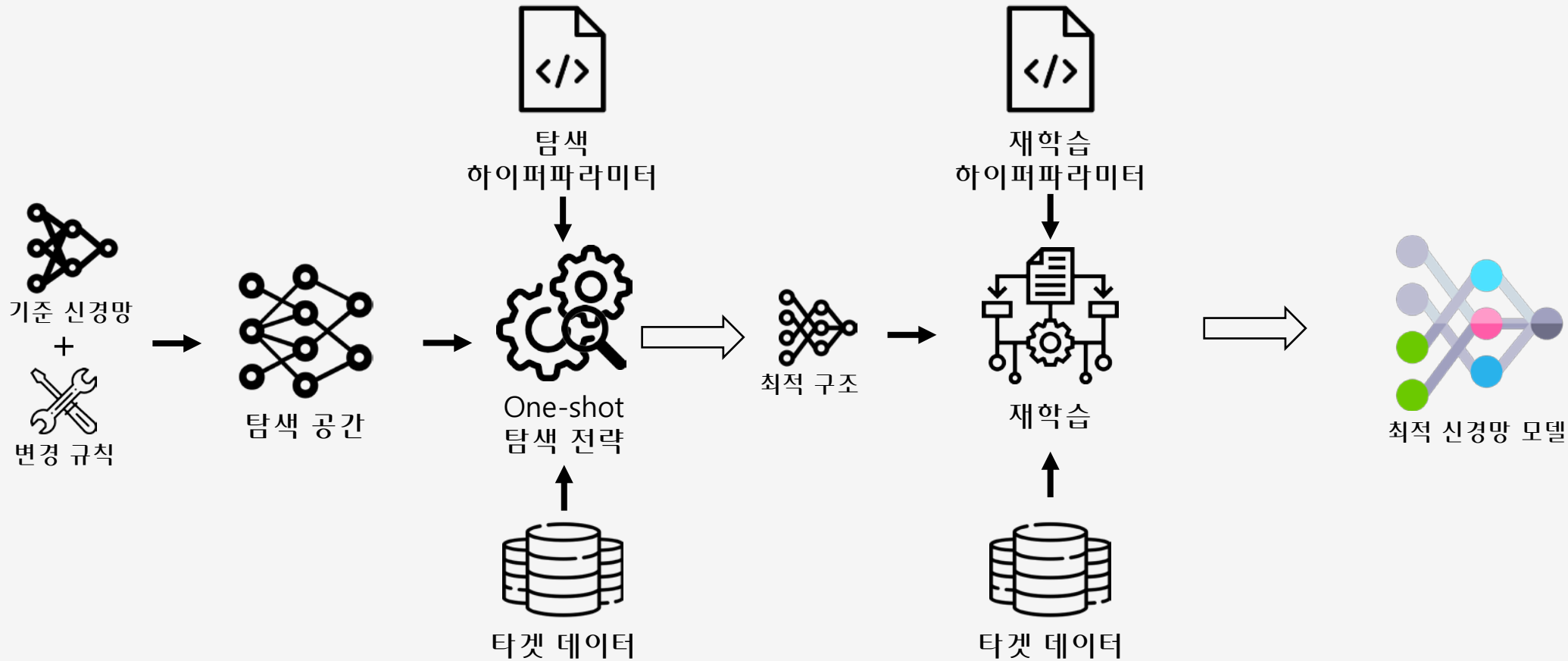
➢ One-shot 탐색 전략



- 후보 신경망들을 하나의 큰 신경망으로 구성
- 수퍼 신경망의 학습, 서브 신경망 탐색
- 최적의 서브 신경망 구조 결정

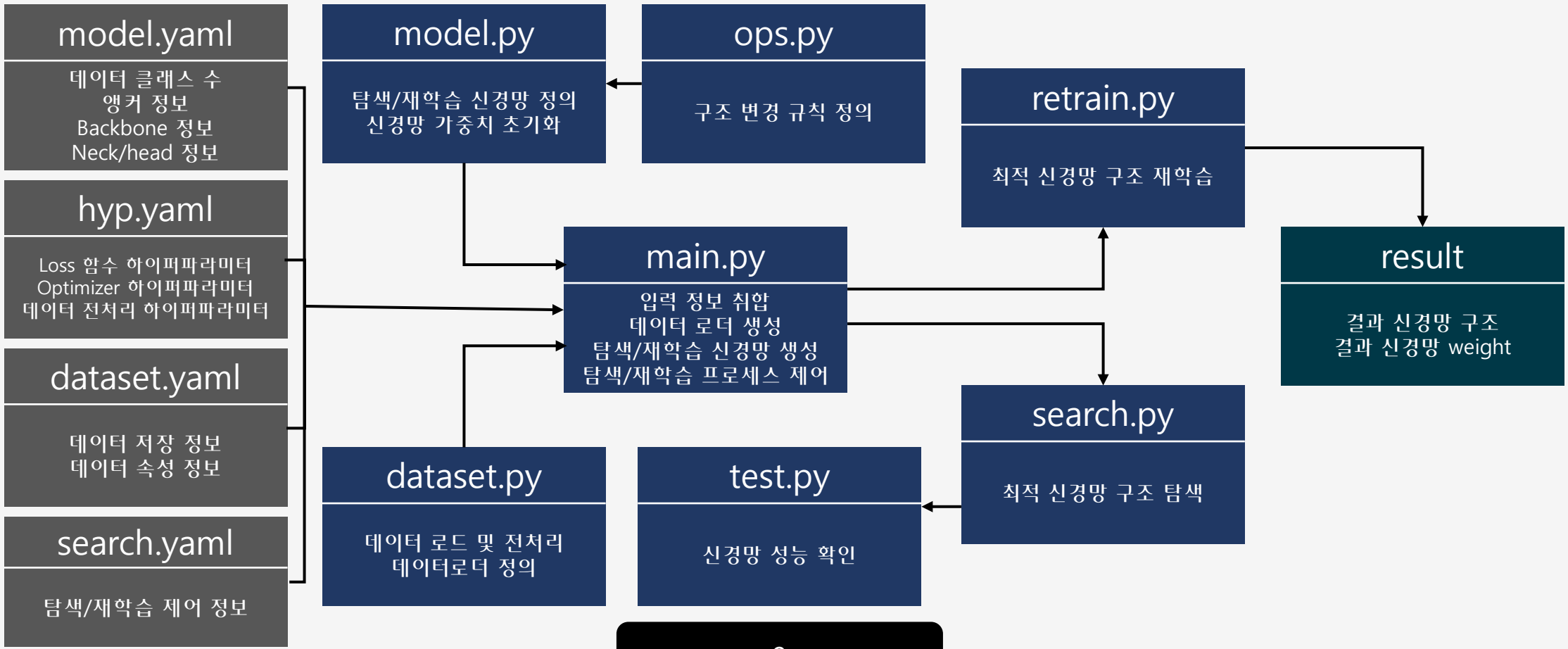
3. 객체 검출 신경망 구조 탐색 기술 – 설계

객체 검출 신경망 구조 탐색 프레임워크



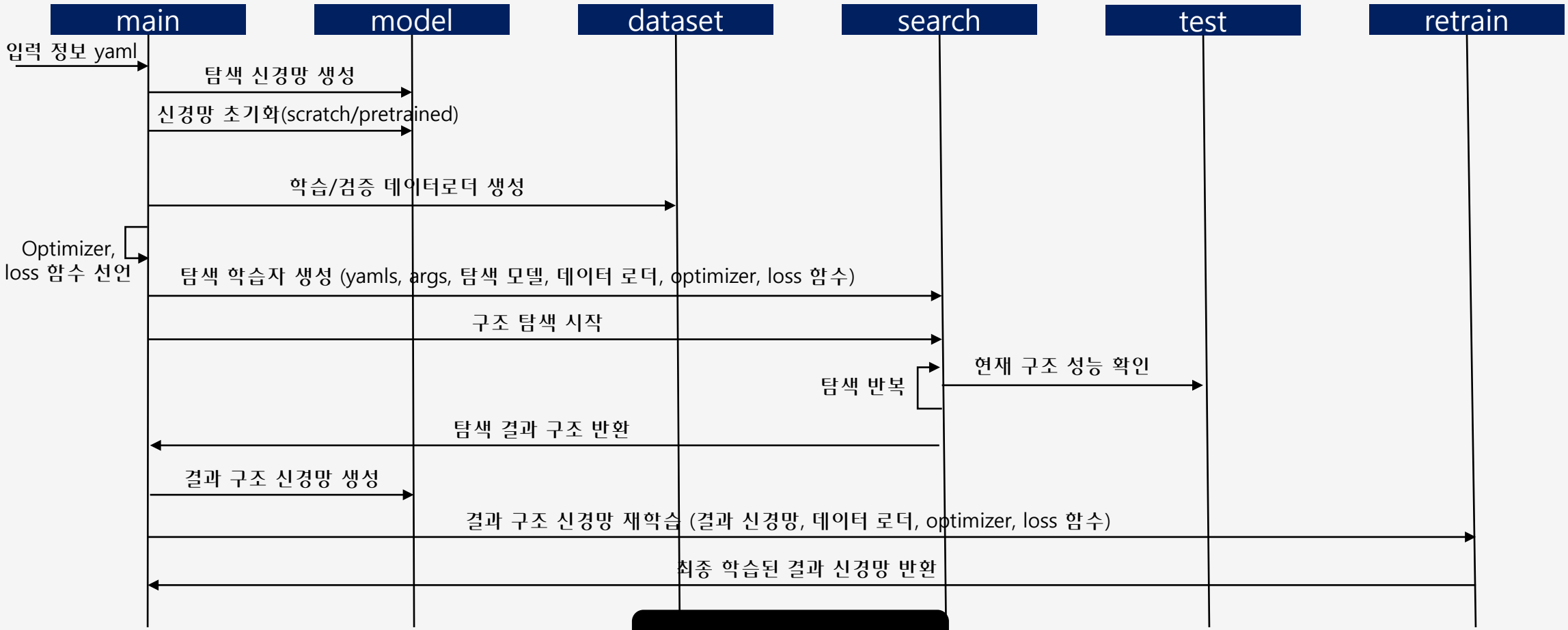
3. 객체 검출 신경망 구조 탐색 기술 – 설계

객체 검출 신경망 구조 탐색 프레임워크 모듈



3. 객체 검출 신경망 구조 탐색 기술 – 설계

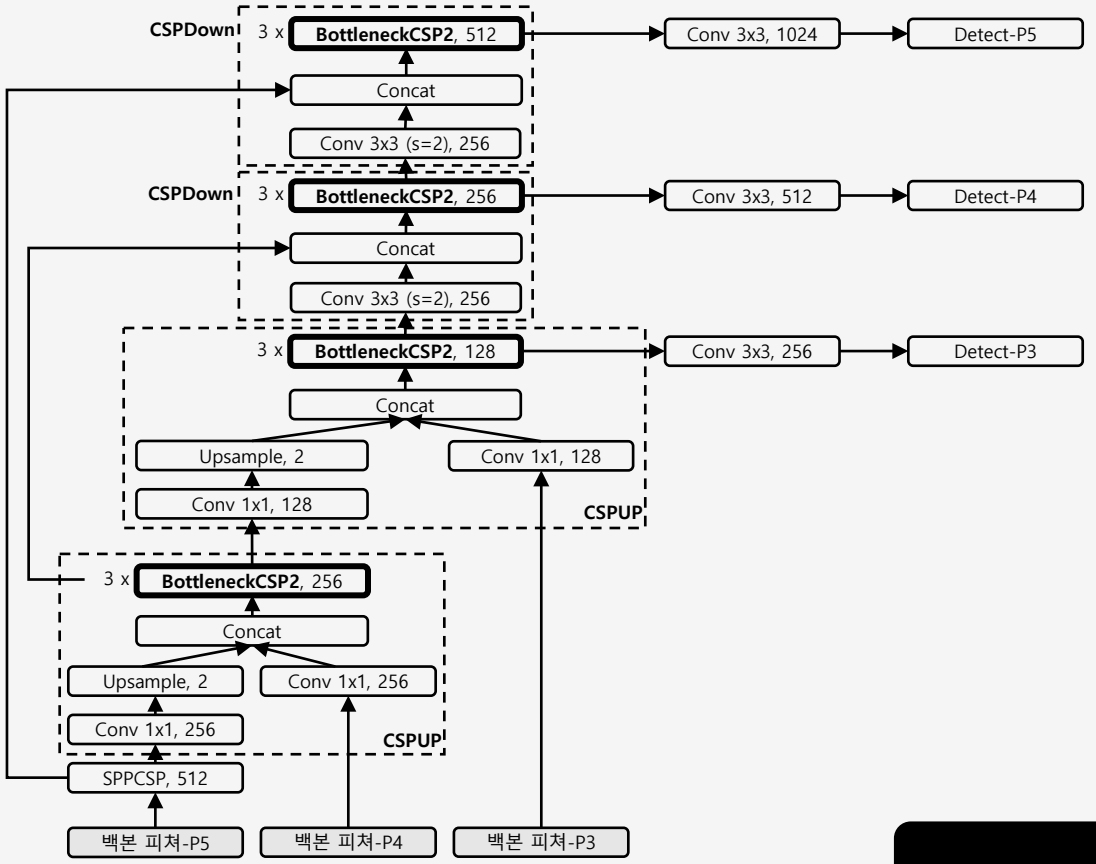
객체 검출 신경망 구조 탐색 프레임워크 시퀀스



3. 객체 검출 신경망 구조 탐색 기술 – 활용

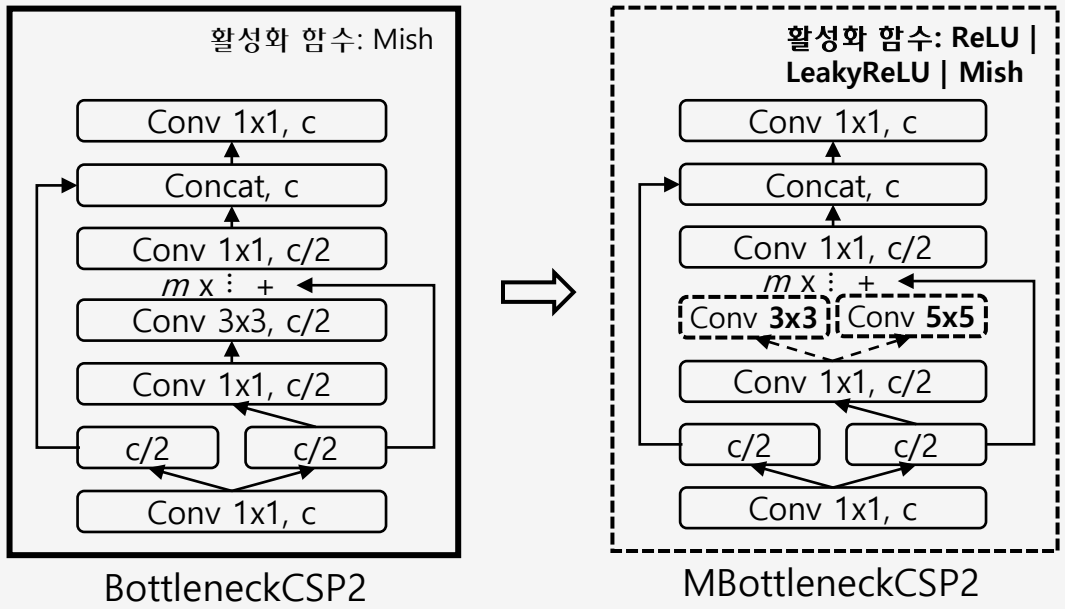
객체 검출 신경망 구조 탐색 공간 정의

➢ 기준 신경망 구조 – YOLOv4-P5 활용



➢ 변경 규칙

- Convolutional 커널 크기 변화: 3x3 또는 5x5
- 활성화 함수 변화: ReLU, LeakyReLU 또는 Mish



BottleneckCSP2 MBottleneckCSP2

3. 객체 검출 신경망 구조 탐색 기술 – 활용

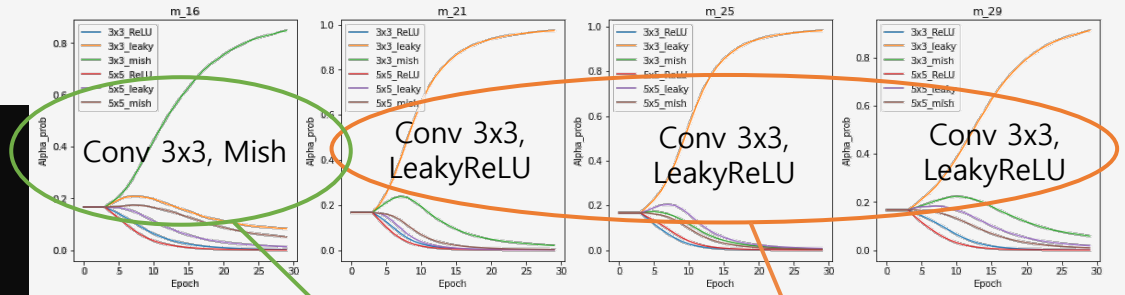
객체 검출 신경망 구조 탐색

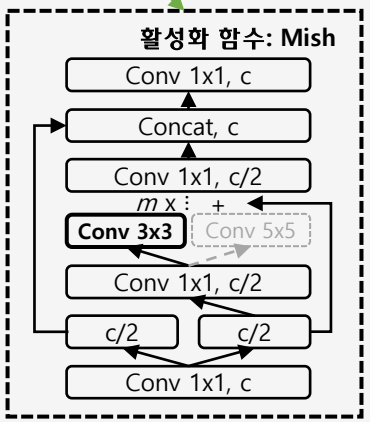
➢ 신경망 구조 탐색 공간

```
# yolov4-p5 head
# na = len(anchors[0])
head:
  [[-1, 1, SPPCSP, [512]], # 11
   [-1, 1, Conv, [256, 1, 1]],
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [8, 1, Conv, [256, 1, 1]], # route backbone P4
   [[-1, -2], 1, Concat, [1]],
   #  [-1, 3, BottleneckCSP2, [256]], # 16
   [-1, 3, MBottleneckCSP2, [256]],
   [-1, 1, Conv, [128, 1, 1]],
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [6, 1, Conv, [128, 1, 1]], # route backbone P3
   [[-1, -2], 1, Concat, [1]],
   #  [-1, 3, BottleneckCSP2, [128]], # 21
   [-1, 3, MBottleneckCSP2, [128]],
   [-1, 1, Conv, [256, 3, 1]],
   [-2, 1, Conv, [256, 3, 2]],
   [[-1, 16], 1, Concat, [1]],  # cat
   #  [-1, 3, BottleneckCSP2, [256]], # 25
   [-1, 3, MBottleneckCSP2, [256]],
   [-1, 1, Conv, [512, 3, 1]],
   [-2, 1, Conv, [512, 3, 2]],
   [[-1, 11], 1, Concat, [1]],  # cat
   #  [-1, 3, BottleneckCSP2, [512]], # 29
   [-1, 3, MBottleneckCSP2, [512]],
   [-1, 1, Conv, [1024, 3, 1]],
   # Detect(P3, P4, P5)
   [[22,26,30], 1, Detect, [nc, anchors]],
   ]
```

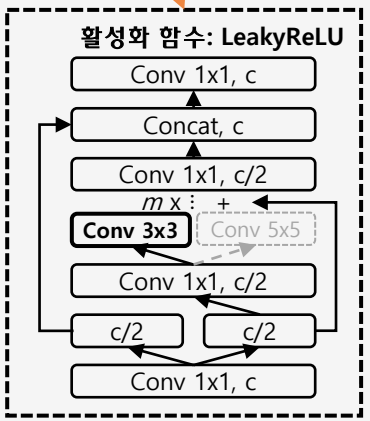
```
                 params  module                                  arguments
  0                928  models.ops_syolov4.Conv                 [3, 32, 3, 1]
  1              18560  models.ops_syolov4.Conv                 [32, 64, 3, 2]
  2              19904  models.ops_syolov4.BottleneckCSP        [64, 64, 1]
  3              73984  models.ops_syolov4.Conv                 [64, 128, 3, 2]
  4             161152  models.ops_syolov4.BottleneckCSP        [128, 128, 3]
  5             295424  models.ops_syolov4.Conv                 [128, 256, 3, 2]
  6            2614016  models.ops_syolov4.BottleneckCSP        [256, 256, 15]
  7            1180672  models.ops_syolov4.Conv                 [256, 512, 3, 2]
  8           10438144  models.ops_syolov4.BottleneckCSP        [512, 512, 15]
  9            4720640  models.ops_syolov4.Conv                 [512, 1024, 3, 2]
 10           20728832  models.ops_syolov4.BottleneckCSP        [1024, 1024, 7]
 11            7610368  models.ops_syolov4.SPPCSP               [1024, 512, 1]
 12             131584  models.ops_syolov4.Conv                 [512, 256, 1, 1]
 13                  0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 14             131584  models.ops_syolov4.Conv                 [512, 256, 1, 1]
 15                  0  models.ops_syolov4.Concat               [1]
 16           23230464  models.ops_syolov4.MBottleneckCSP2      [512, 256, 3]
 17              33024  models.ops_syolov4.Conv                 [256, 128, 1, 1]
 18                  0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 19              33024  models.ops_syolov4.Conv                 [256, 128, 1, 1]
 20                  0  models.ops_syolov4.Concat               [1]
 21            5815296  models.ops_syolov4.MBottleneckCSP2      [256, 128, 3]
 22             295424  models.ops_syolov4.Conv                 [128, 256, 3, 1]
 23             295424  models.ops_syolov4.Conv                 [128, 256, 3, 2]
 24                  0  models.ops_syolov4.Concat               [1]
 25           23230464  models.ops_syolov4.MBottleneckCSP2      [512, 256, 3]
 26            1180672  models.ops_syolov4.Conv                 [256, 512, 3, 1]
 27            1180672  models.ops_syolov4.Conv                 [256, 512, 3, 2]
 28                  0  models.ops_syolov4.Concat               [1]
 29           92860416  models.ops_syolov4.MBottleneckCSP2      [1024, 512, 3]
 30                -1  1   4720640  models.ops_syolov4.Conv                 [512, 1024, 3, 1]
 31      [22, 26, 30]  1    610300  model.Detect                            [80, [[13, 17, 31, 25
, 24, 51, 61, 45], [48, 102, 119, 96, 97, 189, 217, 184], [171, 384, 324, 451, 616, 618, 800, 800
]], [256, 512, 1024]]
```

➢ 아키텍처 파라미터 업데이트 결과





MBottleneckCSP2



MBottleneckCSP2

감사합니다.