

출석체크 AI

전 화 기

20141713 허 찬

20151476 이원석

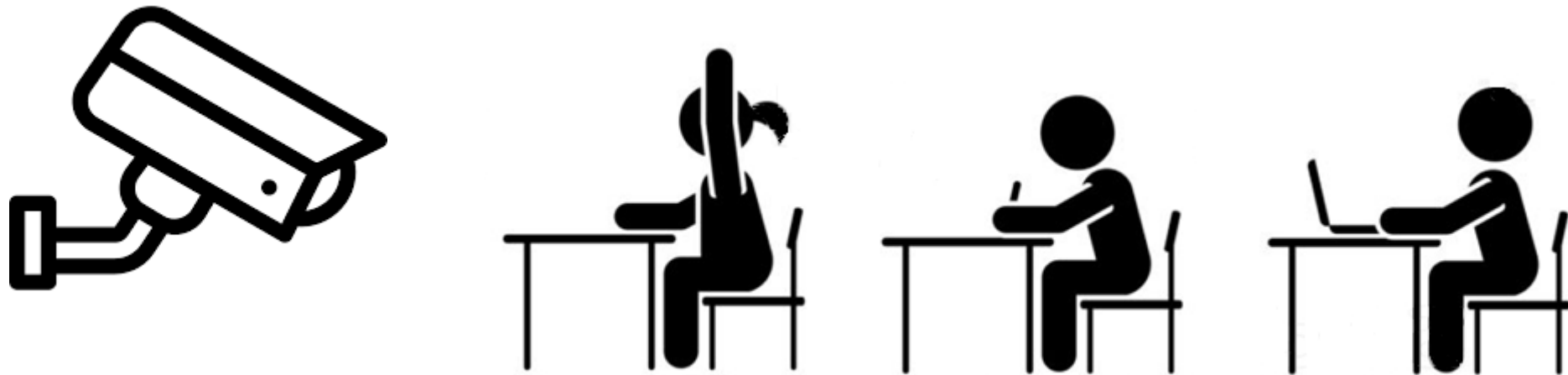
20151739 공대현

◆ 문제 인식

- 대학 수업 내 출결 관련 '출퇴', '대리출석' 등 다양한 문제 발생
- 교수님이나 조교가 일일이 학생들의 출결을 확인해야하는 번거로움
- ✓ 출석체크의 문제 해결 및 효율성, 편리성 개선

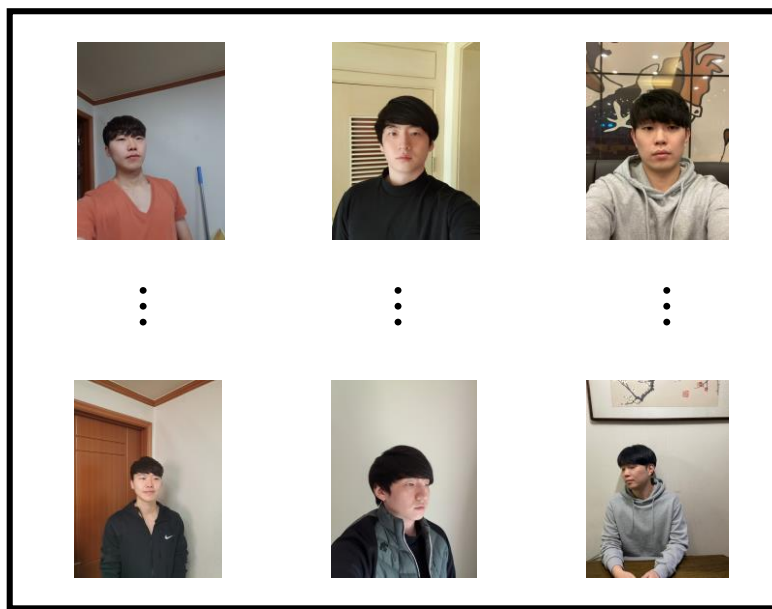
◆ 해결 방안

- 강의실 내 CCTV를 이용하여 실시간으로 학생이 수업에 임하고 있는지 확인
- 총 강의 시간동안 학생이 있는 시간을 확인하여 자동으로 출석, 지각, 결석, 출퇴 구분

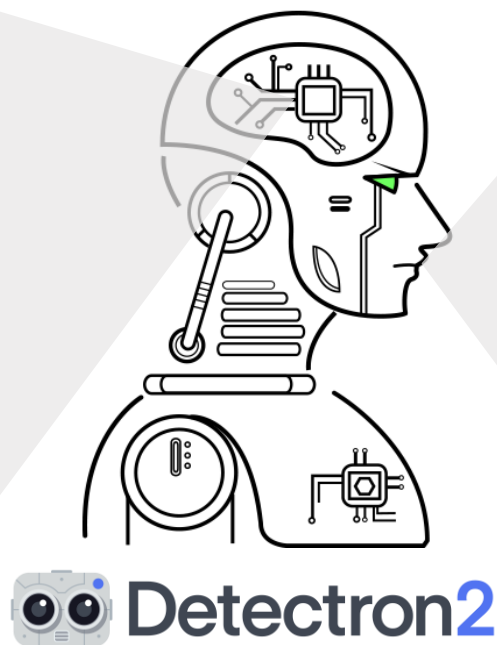


◆ 설계 계획

- ① 학생들의 얼굴 사진으로 Dataset 생성
- ② Object Detection을 이용하여 Neural Network 학습
- ③ 실제 강의실을 배경으로한 동영상으로 출석시스템 Test



[Image Dataset]



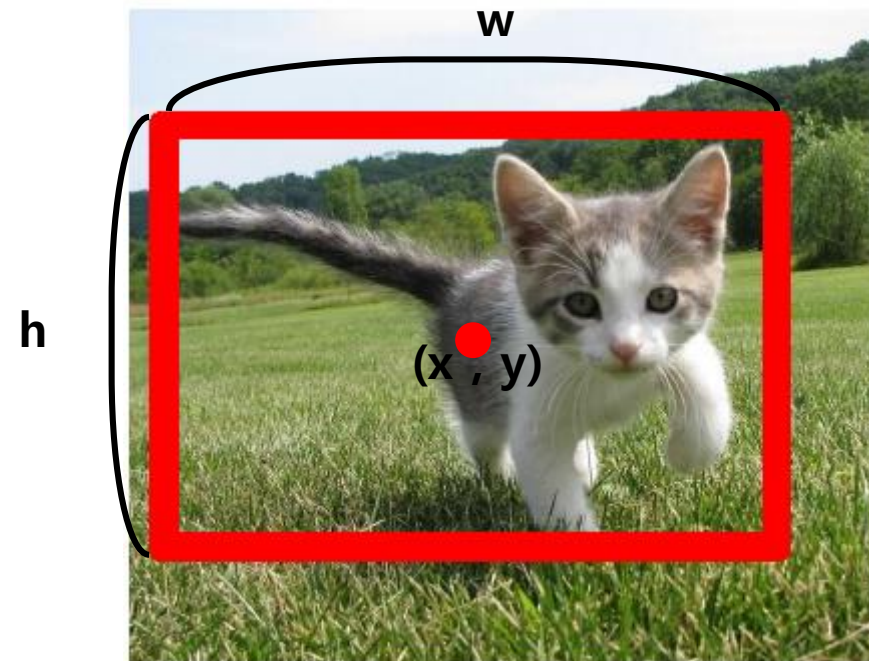
이름	강의 시작	출석 시간	출결 결과
공대현	O	75분	출석
이원석	O	25분	출퇴
허찬	X	65분	지각

◆ Object Detection



DOG, DOG, CAT

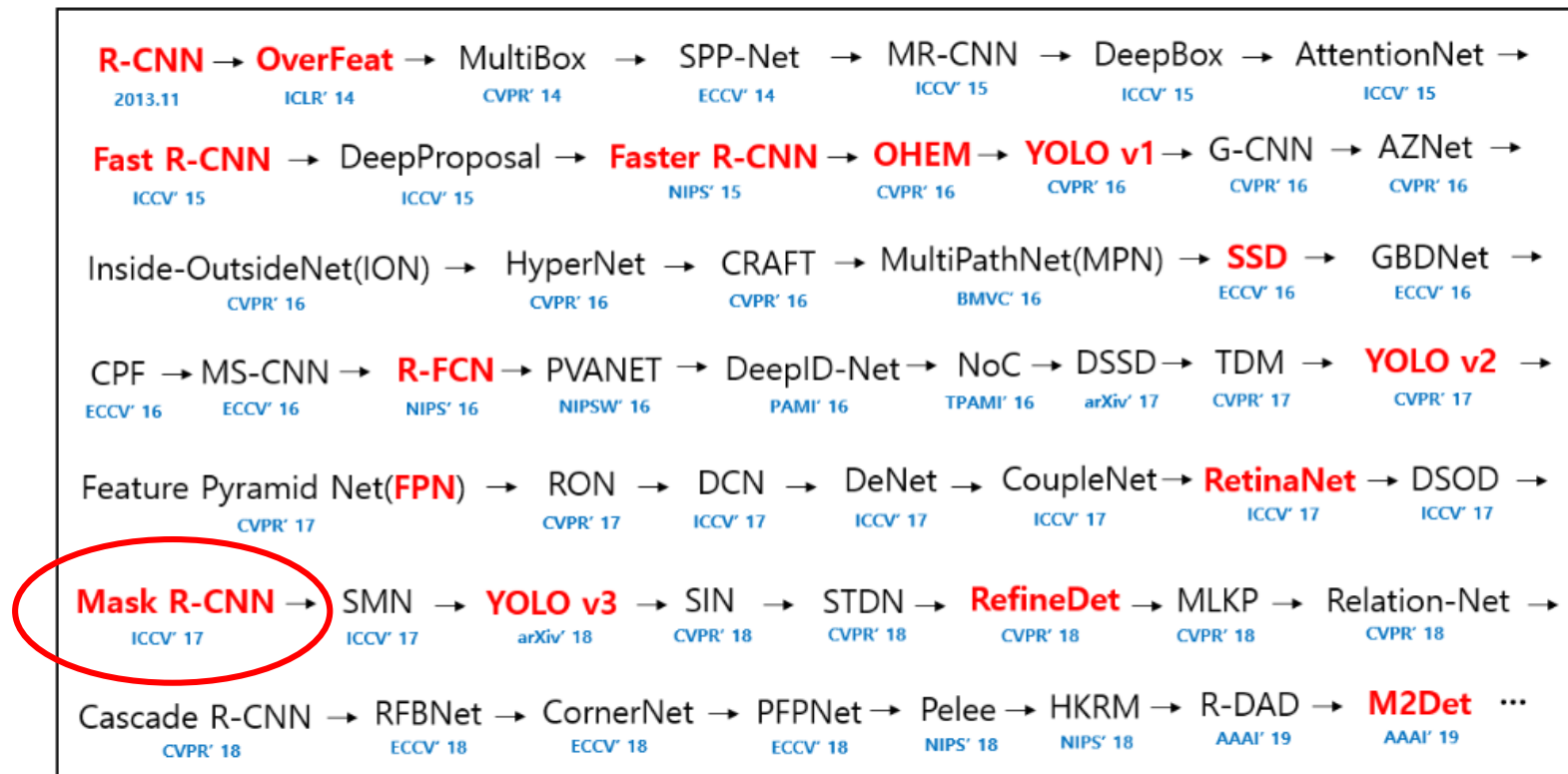
- Input : Images (require good resolutions)
- Output : Bounding box (Bbox) & Class of objects
(require multiple labels of not fixed number)



- Regression할 대상 : Bbox $\Rightarrow (x, y, w, h)$
 (x, y) = coordinate of center of Bbox
 w = width of Bbox , h = height of Bbox

◆ Object Detection

- Object Detection중 Mask RCNN을 사용

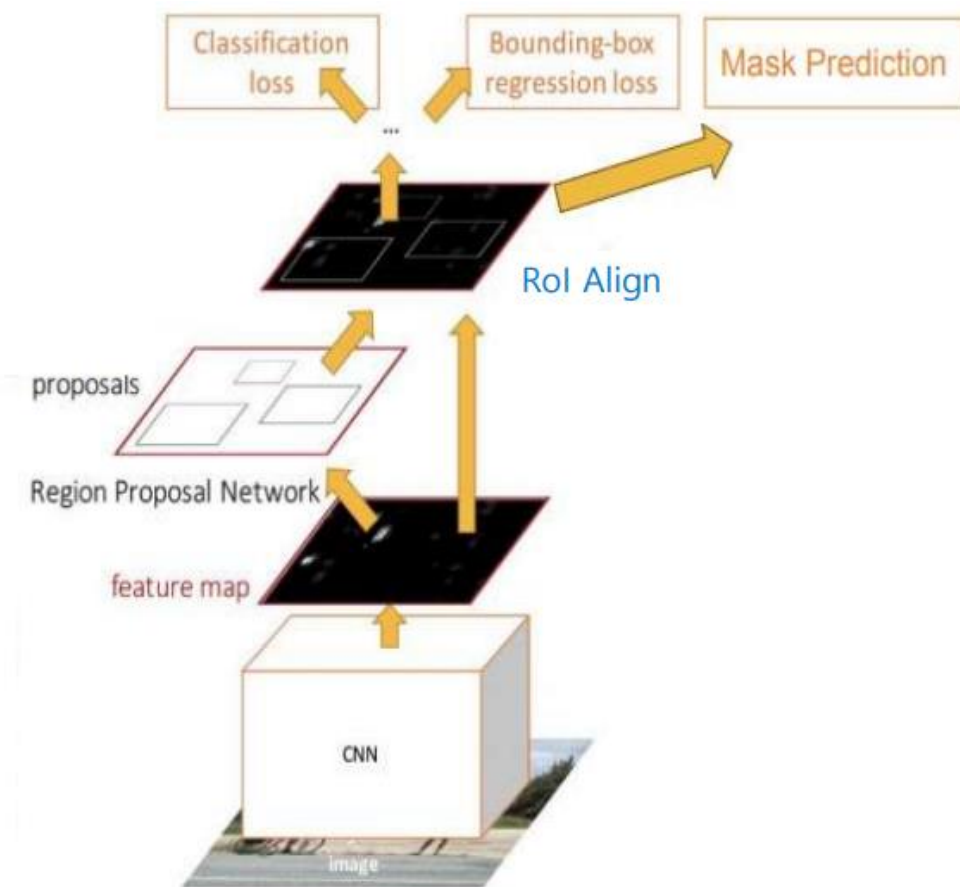
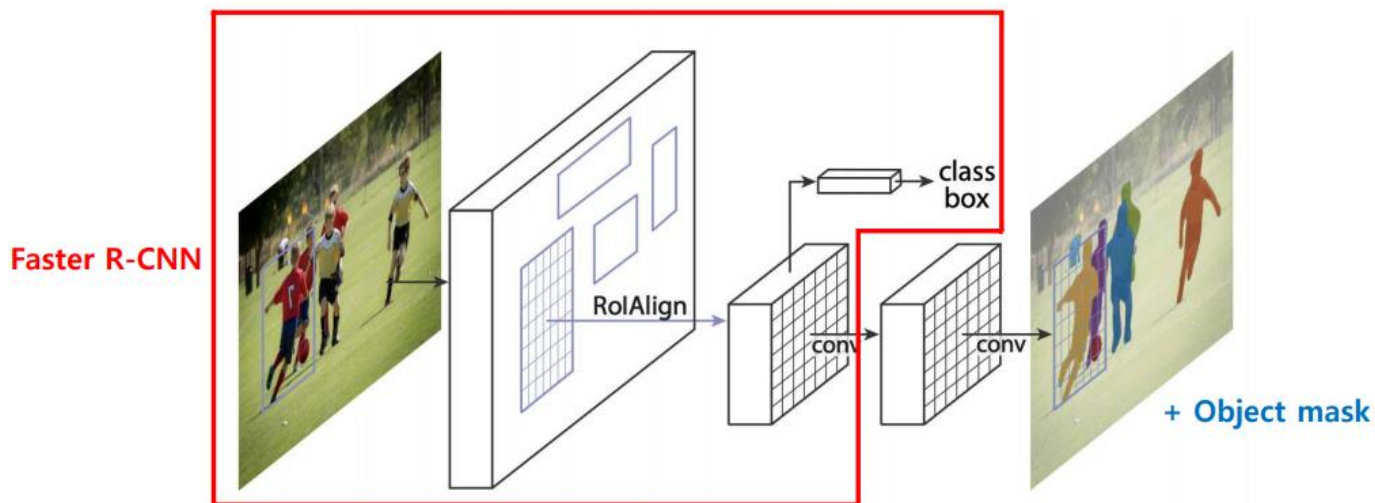


Object detection list from 2014 to now

◆ Object Detection

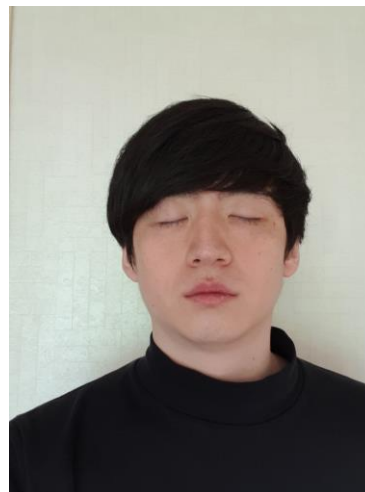
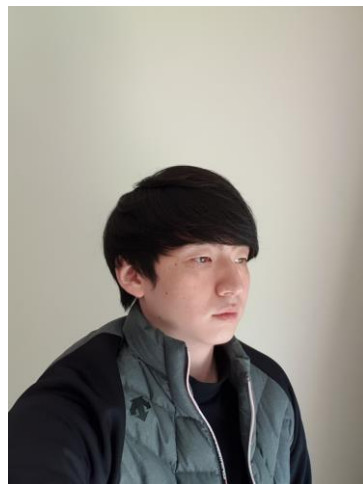
• Mask R-CNN Architecture

- 기존의 Object Detection Model인 Faster R-CNN에 추가적으로 Object Mask 추가



◆ Data Preprocessing


- Image Dataset



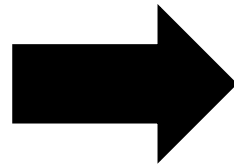
• • • • •

Original Image 다양하게 1인당 60장씩 준비

◆ Data Augmentation

- 데이터 증식을 위해 'Supervisely' application 자체 언어 DTL 사용
-  **SUPERVISELY** (<https://supervise.ly/>)

- 4가지 object로 변환 내용 정의
- (1) Action – define type of operation
 - (2) Source – input data
 - (3) Destination – output data
 - (4) Setting – transformation settings



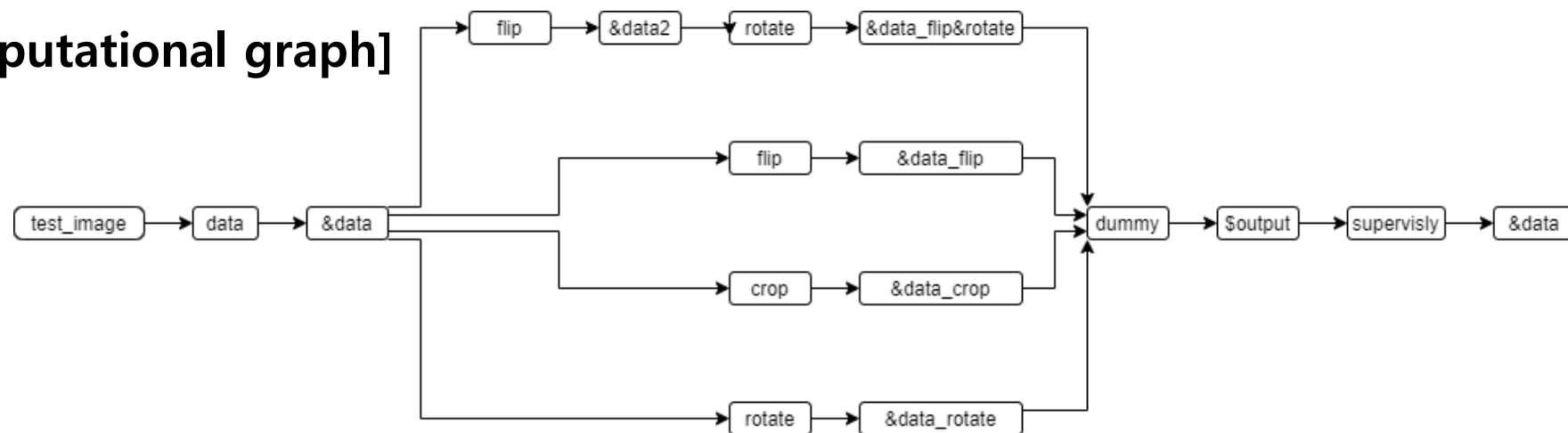
- 코드 format

```
{  
    "action": "data",  
    "src": [  
        "my_project/*"  
    ],  
    "dst": "$data",  
    "settings": {  
        "classes_mapping": "default"  
    }  
}
```

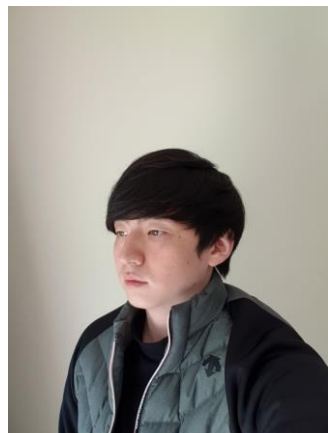

◆ Data Augmentation

- Transformation method 중 crop, flip, rotate의 조합으로 데이터 5배 증식

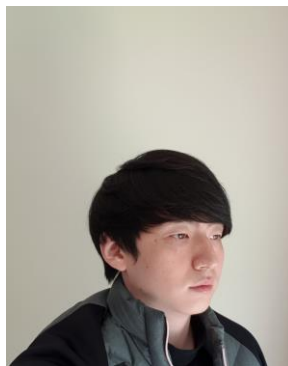
[computational graph]



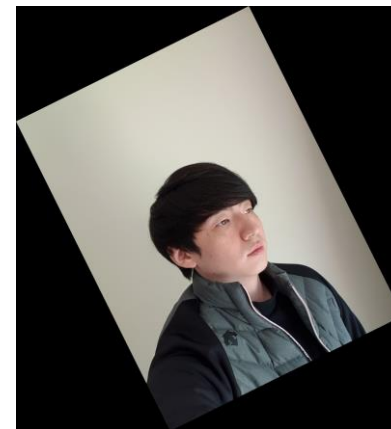
[Original Image]



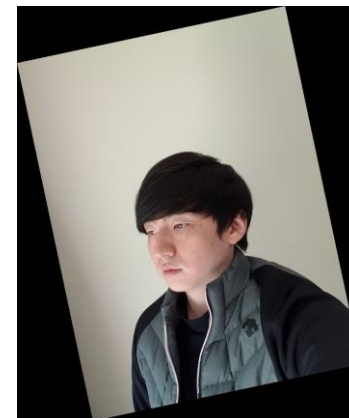
[Flip]



[Crop]



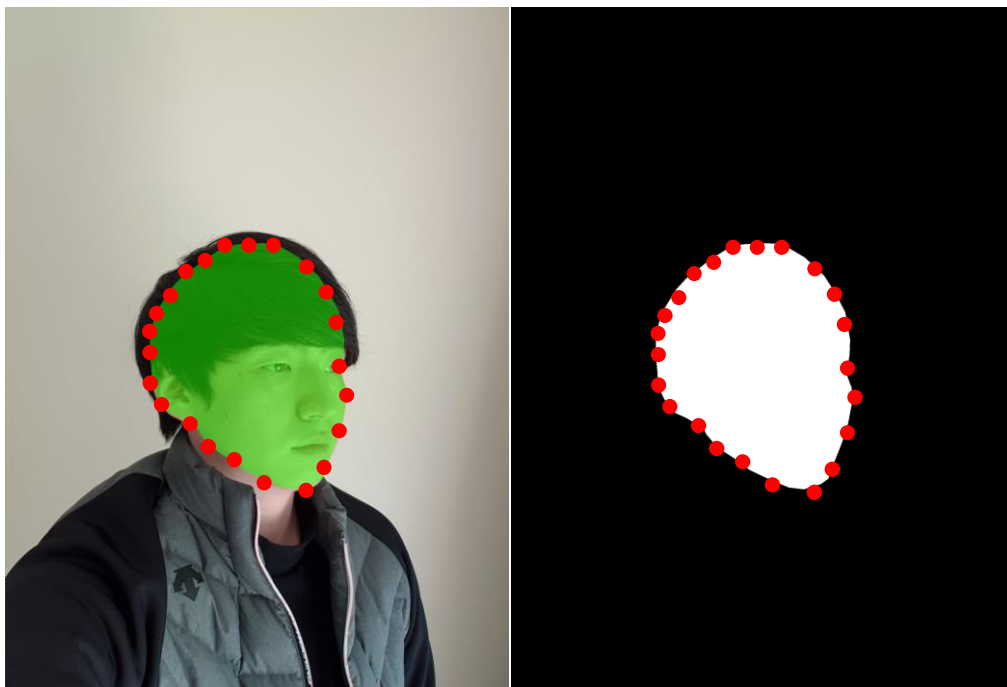
[Rotate]



[Rotate & Flip]

◆ Dataset 파일 변환

- Image File Dataset → COCO Dataset(json file)
- Image와 Annotation을 함께 저장하고 있는 형태의 dataset
- COCO : Common Objects in COntext
- (github.com/waspinator/pycococreator)



pycococreator.py

```
CATEGORIES = [  
{ 'id': 1,  
  'name': 'Kong',  
  'supercategory': 'face'},  
{ 'id': 2,  
  'name': 'Lee',  
  'supercategory': 'face'},  
{ 'id': 3,  
  'name': 'Huh',  
  'supercategory': 'face'}]
```

JSON File

```
"annotations": [{ "id": 2, "image_id": 2,  
  "category_id": 2, "iscrowd": 0, "area": 22695,  
  "bbox": [380.0, 426.0, 147.0, 225.0],  
  "segmentation": [[445.0, 650.5, 416.0, 649.5,  
    390.5, 625.0, 379.5, 608.0, 379.5, 583.0,  
    419.5, 456.0, 441.0, 431.5, 475.0, 425.5,  
    502.0, 433.5, 517.5, 450.0, 526.5, 486.0,  
    486.5, 613.0, 472.0, 636.5, 445.0, 650.5]],  
  "width": 960, "height": 1280}
```

◆ Detectron2를 이용하여 Train

```
from detectron2.modeling.meta_arch.build import build_model
import torch.nn as nn
from detectron2.data.datasets import register_coco_instances
register_coco_instances("Acheck", {}, "./Acheck_hair.json" "./img_hair")
from detectron2.data import MetadataCatalog ① ②
MetadataCatalog.get("Acheck").thing_classes = ["Kong", "Lee", "Huh"]
Acheck_metadata = MetadataCatalog.get("Acheck")

from detectron2.data import DatasetCatalog
dataset_dicts = DatasetCatalog.get("Acheck")

import random
import cv2
from detectron2.utils.visualizer import Visualizer
```

- ① COCO format data json file 사용
- ② 얼굴 이미지 파일
- ③ Class : 3 (Kong, Lee, Huh)

```
from detectron2.engine import DefaultTrainer
from detectron2.config import get_cfg
import os

cfg = get_cfg()
cfg.MODEL.DEVICE='cuda:1'
yaml = "./configs/COCO-InstanceSegmentation/mask_rcnn_R_101_C4_3x.yaml"
weight = "./output/model_final_cascade_with_hair/mask_rcnn_R_101_C4_3x/model_final.pth"
cfg.merge_from_file(
    "./configs/COCO-InstanceSegmentation/mask_rcnn_R_101_C4_3x.yaml"
)
cfg.DATASETS.TRAIN = ("Acheck",)
cfg.DATASETS.TEST = ("Acheck",)
cfg.DATALOADER.NUM_WORKERS = 2
# cfg.MODEL.WEIGHTS = "detectron2://COCO-InstanceSegmentation/mask_rcnn_R_101_FPN_3x/137849600/model_final_f102
cfg.SOLVER.IMS_PER_BATCH = 2
cfg.SOLVER.BASE_LR = 0.002
cfg.SOLVER.MAX_ITER = (
    50000
) # 300 iterations seems good enough, but you can certainly train longer
cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = (
    128
) # faster, and good enough for this toy dataset
cfg.MODEL.ROI_HEADS.NUM_CLASSES = 3 # 3 classes (Kong, lee, Huh)
os.makedirs('./output/model_final_cascade_with_hair/mask_rcnn_R_101_C4_3x', exist_ok=True)
cfg.OUTPUT_DIR = "./output/model_final_cascade_with_hair/mask_rcnn_R_101_C4_3x"
trainer = DefaultTrainer(cfg)
trainer.resume_or_load(resume= "./output/model_final_cascade_with_hair/mask_rcnn_R_101_C4_3x/model_final.pth")
trainer.train()
```

◆ 출석체크 기준

- 동영상의 프레임 속도에 맞춰 현실 시간 계산(2.47frame/s)
- 출석 시간에 따라 출석, 지각, 결석, 출퇴 4가지로 분류
- (실제 강의 시간 75분을 동영상 25초로 가정하여 Test)

상태 수업시간	출석	지각	결석	출퇴
75분	늦지 않고 45분 이상 출석	15분 이하만큼 늦게 출석	15분 초과만큼 늦게 출석	결석하지 않고 45분 이하 출석
25초	늦지 않고 15초 이상 출석	5초 이하만큼 늦게 출석	5초 초과만큼 늦게 출석	결석하지 않고 15초 이하 출석

◆ 출석체크 결과 (https://www.youtube.com/watch?v=GfHGqiB_RqE)

- 좌측 상단에 출석 시간을 frame 단위로 display
- Kong : 10 frame, Lee : 16 frame, Huh : 5 frame



[ex. 수업시간 5초 경과]

◆ 출석체크 결과

- 총 수업 시간 25초가 지난 뒤, 학생들의 출결 상태를 기록
- 출석 시간에 따른 출석율(빈도)에 따라 출석, 출퇴, 지각 결석으로 구분

[05/27 11:18:22 출석체크결과] :

총 시간: 24.9초

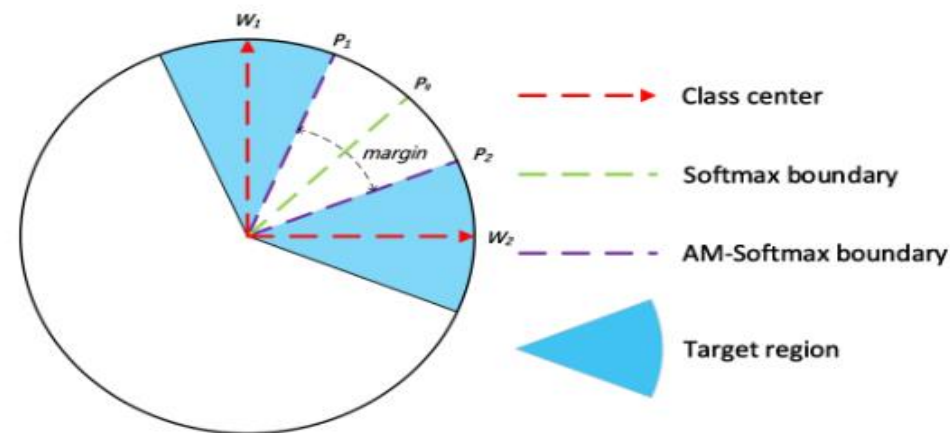
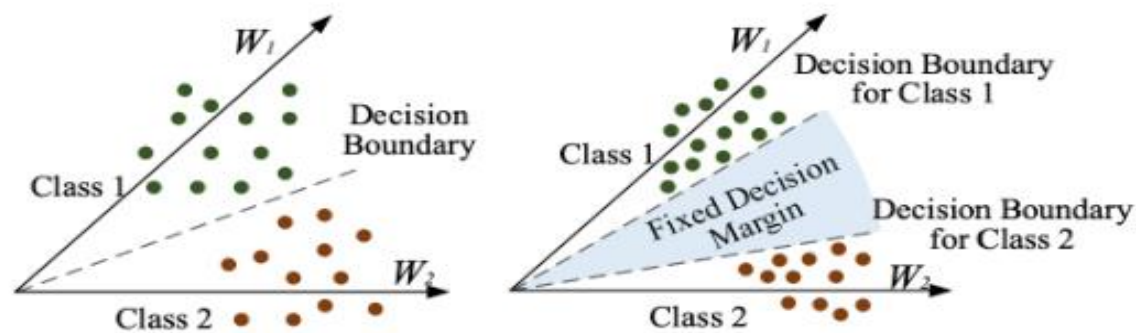
학번	전공	이름	출석시간	빈도	결과
20151739	기계공학과	공대현 (Kong)	25.1초	100%	출석 (attend)
20151476	전자공학과	이원석 (Lee)	10.0초	40.1%	출퇴 (escape)
20141713	화생공학과	허 찬 (Huh)	20.6초	82.8%	지각 (late)
20161739	방송연예학	사 나 (Sana)	0.0초	0.0%	결석 (absence)

[수업시간 종료 후 출결 결과]

◆ 한계점과 개선방법

- Bbox와 Mask Segmentation은 정확하지만, Classification에서 전에 없던 Class를 Train Class 중 하나로 잘못 인식하는 문제 발생

⇒ Classification Loss에 Metric Learning을 적용해야 함.



$$L_{AMS} = -\frac{1}{N} \sum_{i=1}^N \log \left(\frac{e^{s \cdot (\cos \theta_{y_i} - m)}}{e^{s \cdot (\cos \theta_{y_i} - m)} + \sum_{j=1, j \neq y_i}^c e^{s \cdot \cos \theta_j}} \right) = -\frac{1}{N} \sum_{i=1}^N \log \left(\frac{e^{s \cdot (W_{y_i}^T \mathbf{f}_i - m)}}{e^{s \cdot (W_{y_i}^T \mathbf{f}_i - m)} + \sum_{j=1, j \neq y_i}^c e^{s W_j^T \mathbf{f}_i}} \right)$$

<Angular loss를 사용한 Metric Learning>

◆ 참고문헌

- [1] R. Girshick, J. Donahue, T. Darrell, and J. Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation." CVPR (2014)
- [2] R. Girshick. "Fast R-CNN." ICCV (2015)
- [3] S. Ren, K. He, R. Girshick, and J. Sun. "Faster R-CNN: Towards real-time object detection with region proposal networks." NIPS (2015)
- [4] R. Girshick, K. He, G. Gkioxari, P. Dollar. "Mask R-CNN" ICCV (2017)
- [5] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. "Feature pyramid networks for object detection." CVPR (2017)
- [6] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. "Microsoft COCO: Common objects in context." ECCV (2014)
- [7] K CHen. "MMDetection: Open mmlab detection toolbox and benchmark" arXiv:1906.07155 (2019)
- [8] Z. Cai and N. Vasconcelos, "Cascade R-CNN: High Quality Object Detection and Instance Segmentation," in IEEE Transactions on Pattern Analysis and Machine Intelligence.10.1109/TPAMI.2019.2956516 (2019)
- [9] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollar, and K. He. Detectron. <https://github.com/facebookresearch/detectron>, (2018)
- [10] 앨런 비소책. "딥러닝 데이터 전처리 입문"
- [11] 프랑소와 솔레. "케라스 창시자에게 배우는 딥러닝"

감사합니다.

전 화 기

20141713 허 찬

20151476 이원석

20151739 공대현
