

# 딥러닝 기반의 쓰레기 자동 분류 시스템



2조. 기브 미 Trash

# CONTENTS

1. 주제 선정 이유
2. 프로젝트 목표
3. 순서도
4. 소프트웨어
5. 모델 채택 이유
6. 데이터 선정 및 학습 과정
7. 모델 결과 비교
8. 시연 영상
9. 개선 사항

Q&A



# 팀원 소개

이제희

조장  
로봇 팔 알

백준형

로봇 팔(모터)  
제어  
딥러닝 데이터  
가공  
발표 자료 제작

강민구

딥러닝 데이터  
(수집, 정제, 가공,  
학습)  
발표 자료 제작

김정현

딥러닝 데이터  
(수집, 정제, 가공,  
학습)  
발표 자료 제작



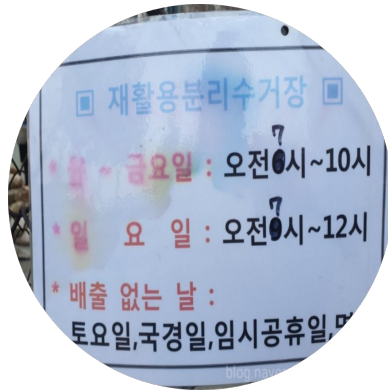
# 주제 선택 이유



플라스틱류 및 유리류 등의 배출량 증가



분리수거장 정리 요소 증가



정해진 분리수거 일자



# 주제 선택 이유



OpenMANIPULATOR-X(왼쪽)과  
RealSense D435i(오른쪽) 활용


정리가 되어있고, 라벨링이 되어있는 데이터

**AI Hub**  
AI 데이터찾기 AI 개발지원 참여하기 정보공유 고객지원 AI 허브소개

로그인 회원가입

데이터 분야

AI 데이터찾기 > 데이터 분야



#환경 #환경오염 #폐기물 #산업폐기물 #AI데이터

## 생활 폐기물 이미지

분야 재난안전환경 유형 비디오, 이미지

갱신년월 : 2023-04  
용량 : 331.07 GB

구축년도 : 2020 조회수 : 4,002 다운로드 : 2,145

다운로드

↓ 샘플 데이터 ?

관심데이터 등록 28

※ 내국인만 데이터 신청이 가능합니다.

목록

데이터 개요

데이터 변경이력

버전	일자	변경내용	비고
1.0	2021-06-25	데이터 최초 개방	

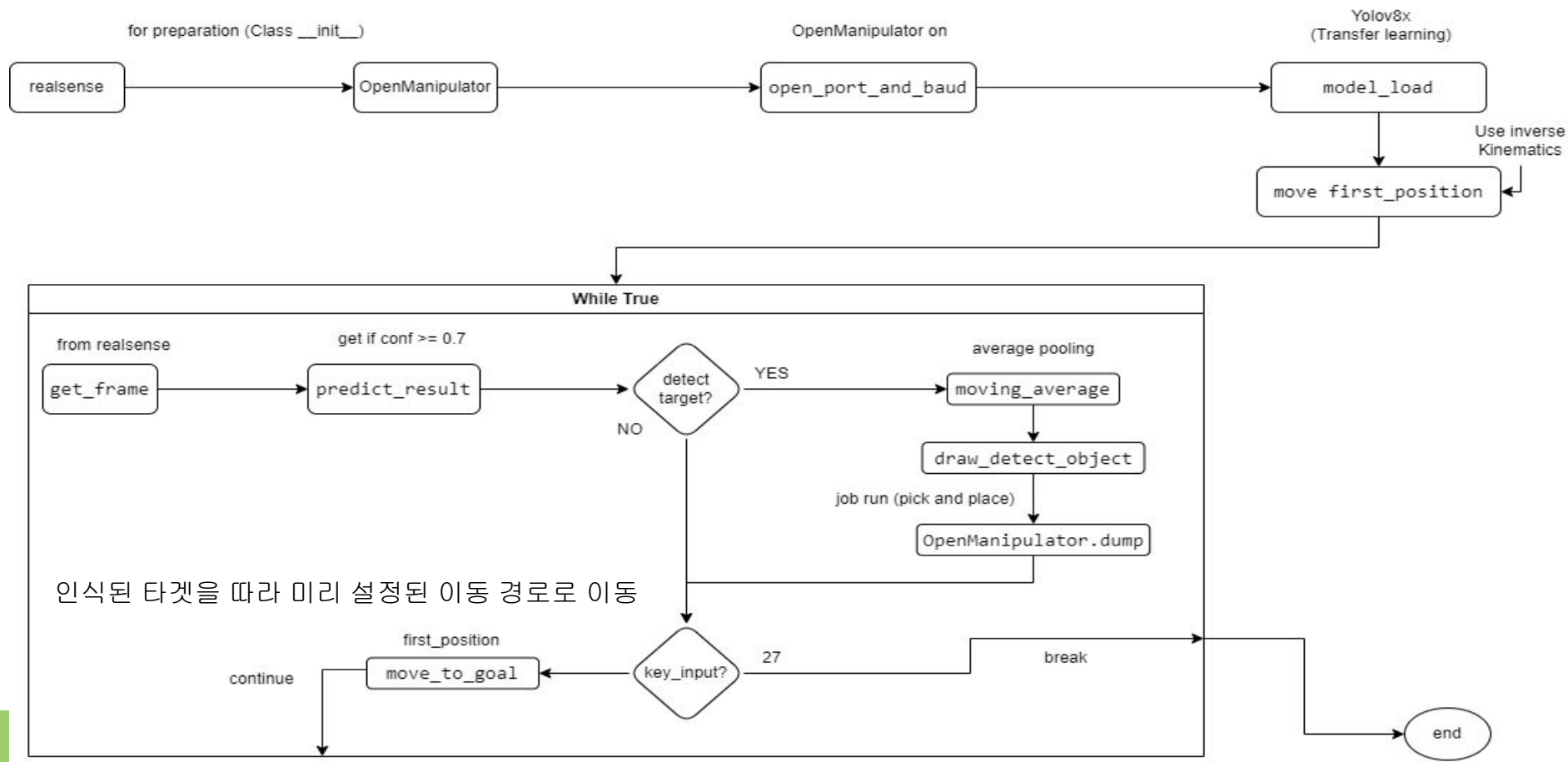
데이터 히스토리

# 프로젝트 목표

“YOLO를 이용해 객체를 분류하고  
로봇을 제어해서  
쓰레기를 자동 분류해보자!”



# 순서도



# 소프트웨어 / 하드웨어

## 모델 학습



YOLOv8 (n,s,l,x)



Google colab (PRO) \* 2



Jupyter Notebook



VS Code

## 예측



YOLOv8x (transfer learning)



OpenCV



RealSense D435





# 소프트웨어 / 하드웨어

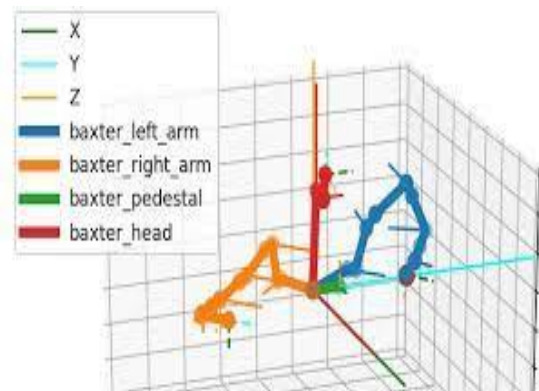
## Dynamixel Control



Dynamixel-sdk



OpenMANIPULATOR-X  
(XM430-W350-T\*5)



Ikpy



# 모델 채택 이유

## Python

YOLOv8 may also be used directly in a Python environment, and accepts the same [arguments](#) as in the CLI example above:

```
from ultralytics import YOLO

# Load a model
model = YOLO("yolov8n.yaml") # build a new model from scratch
model = YOLO("yolov8n.pt") # load a pretrained model (recommended for training)

# Use the model
model.train(data="coco128.yaml", epochs=3) # train the model
metrics = model.val() # evaluate model performance on the validation set
results = model("https://ultralytics.com/images/bus.jpg") # predict on images
success = model.export(format="onnx") # export the model to ONNX format
```

[Models](#) download automatically from the latest Ultralytics [release](#). See [YOLOv8 Python Docs](#) for more examples.

YOLOv8

## Inference with detect.py

`detect.py` runs inference on a variety of sources, downloading [models](#) automatically from the latest YOLOv5 [release](#) and saving results to `runs/detect`

```
python detect.py --weights yolov5s.pt --source 0 # webcam
                                             img.jpg # image
                                             vid.mp4 # video
                                             screen # screenshot
                                             path/ # directory
                                             list.txt # list of images
                                             list.streams # list of streams
                                             'path/*.jpg' # glob
                                             'https://youtu.be/Zgi9g1ksQHc' # YouTube
                                             'rtsp://example.com/media.mp4' # RTSP, RTMP, HTTP stream
```

## Training

The commands below reproduce YOLOv5 [COCO](#) results. [Models](#) and [datasets](#) download automatically from the latest YOLOv5 [release](#). Training times for YOLOv5n/s/m/l/x are 1/2/4/6/8 days on a V100 GPU ([Multi-GPU](#) times faster). Use the largest possible, or pass for YOLOv5 [AutoBatch](#). Batch sizes shown for V100-16GB. `--batch-size --batch-size -1`

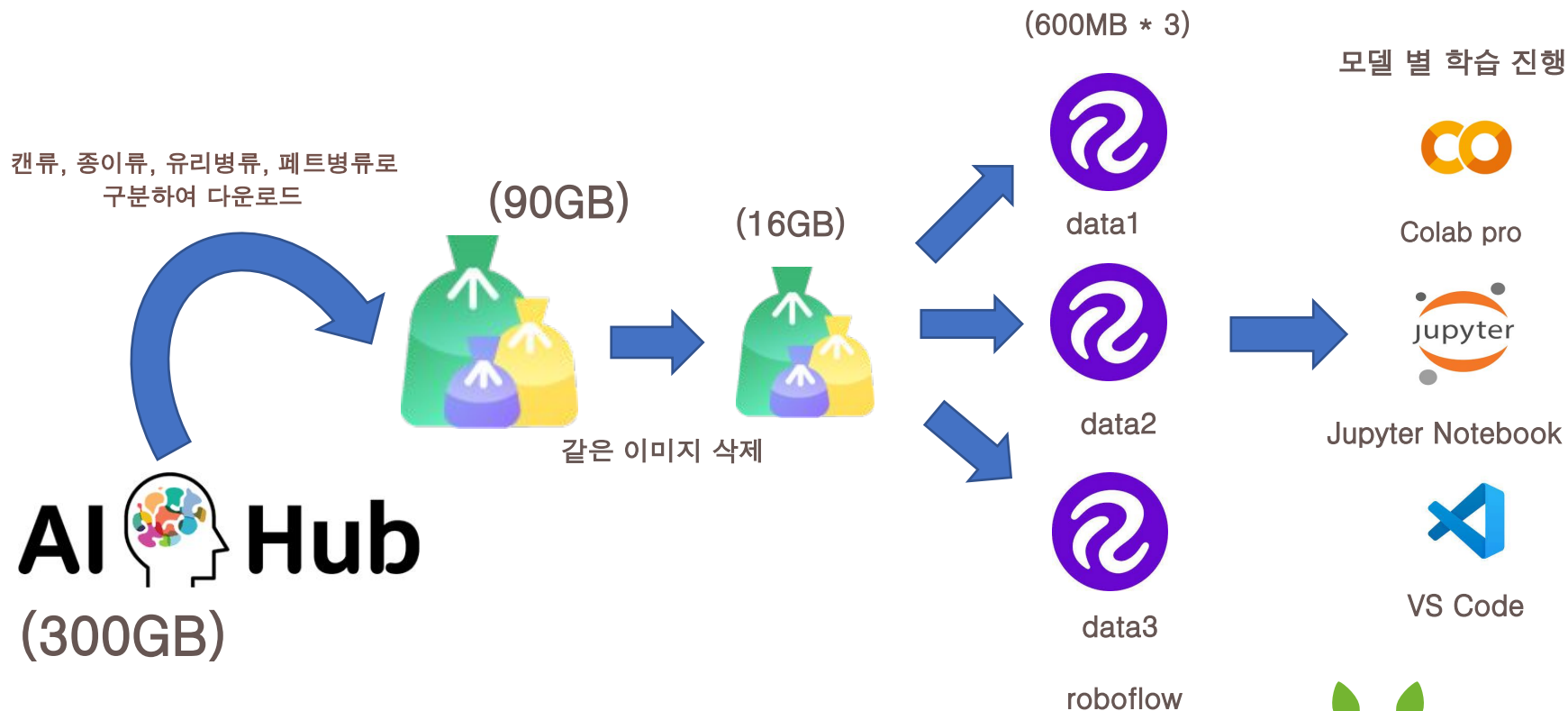
```
python train.py --data coco.yaml --epochs 300 --weights '' --cfg yolov5n.yaml --batch-size 128
                  yolov5s
                  yolov5m
                  yolov5l
                  yolov5x
```

YOLOv5

# 모델 채택 이유

Model	size (pixels)	mAP <sup>val</sup> 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

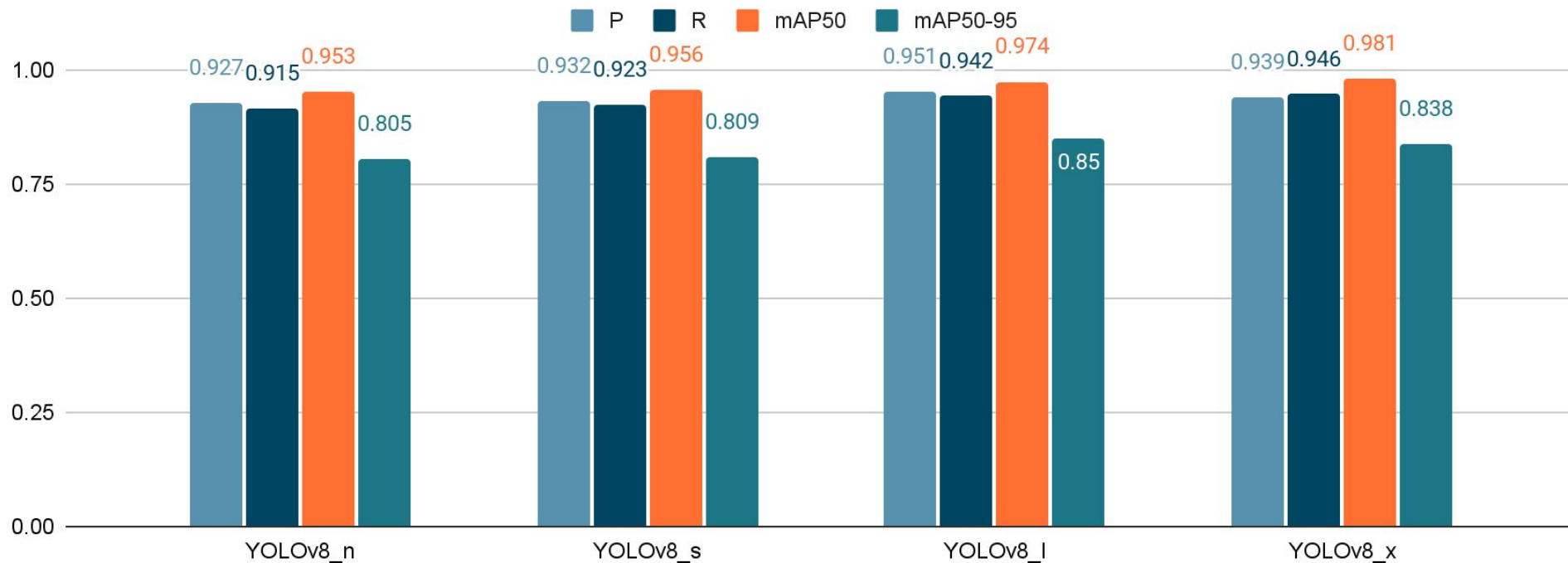
# 데이터 선정 및 학습 과정



# 모델 결과 분석

학습이 끝난 후 model summary 항목을 비교

All



# 모델 결과 비교

실제 웹캠(or 리얼센스)에 연결하여 결과 확인



YOLOv8x

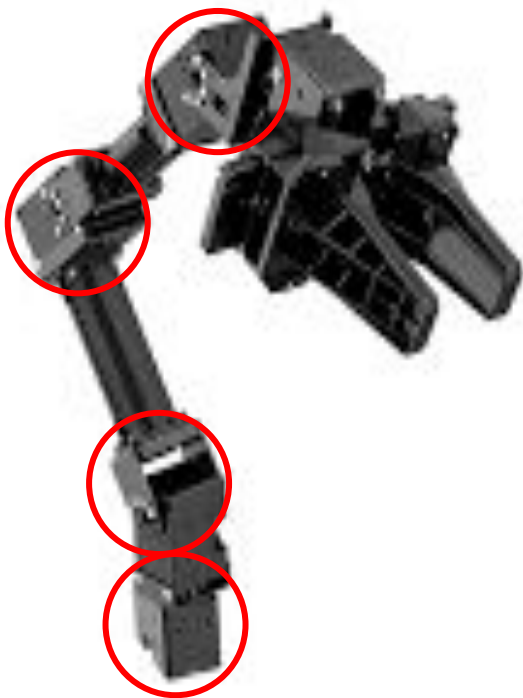


YOLOv8l





# 로봇 제어



```
class ikpy.link.URDFLink(name: str, origin_translation: <MagicMock
id='139836115923216'>, origin_orientation: <MagicMock
id='139836115718096'>, rotation: Optional[<MagicMock
id='139836115743568'>] = None, translation: Optional[<MagicMock
id='139836115745296'>] = None, bounds=None, angle_representation='rpy',
use_symbolic_matrix=True, joint_type: str = 'revolute') [source]
```

Bases: `ikpy.link.Link`

Link in URDF representation.

- Parameters:
- **name** (*str*) – The name of the link
  - **bounds** (*tuple*) – Optional : The bounds of the link.  
Defaults to None
  - **origin\_translation** (*numpy.array*) – The translation vector. (In URDF, attribute "xyz" of the "origin" element)
  - **origin\_orientation** (*numpy.array*) – The orientation of the link. (In URDF, attribute "rpy" of the "origin" element)
  - **rotation** (*numpy.array*) – The rotation axis of the link. (In URDF, attribute "xyz" of the "axis" element)
  - **angle\_representation** (*str*) – Optional : The representation used by the angle. Currently supported representations : rpy. Defaults to rpy, the URDF standard.
  - **use\_symbolic\_matrix** (*bool*) – whether the transformation matrix is stored as a Numpy array or as a Sympy symbolic matrix.
  - **joint\_type** (*str*) – The URDF "type" attribute of the joint. Only support for revolute and prismatic joint for the moment



# 시연 영상



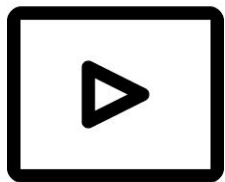


# 개선 시도

겹쳐 있는 이미지는 인식 하지 못함



리얼센스로  
촬영된 영상 저장



일정 주기마다  
프레임 저장



200장



이미지 라벨링



roboflow

600장



학습 진행



Colab Pro



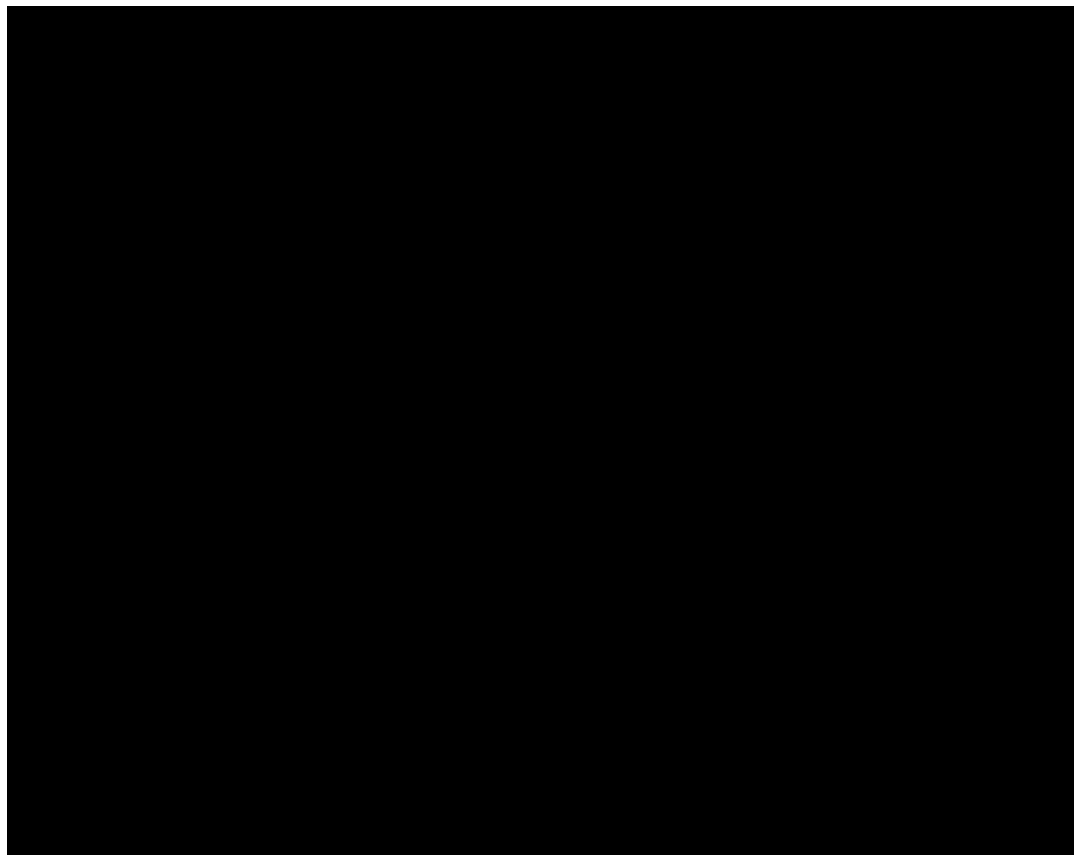
VS Code



Jupyter Notebook



# 개선 시도



시연 때 사용한 이미지에 대한 인식을



But, 일반적인 생활 폐기물에  
대한 인식을



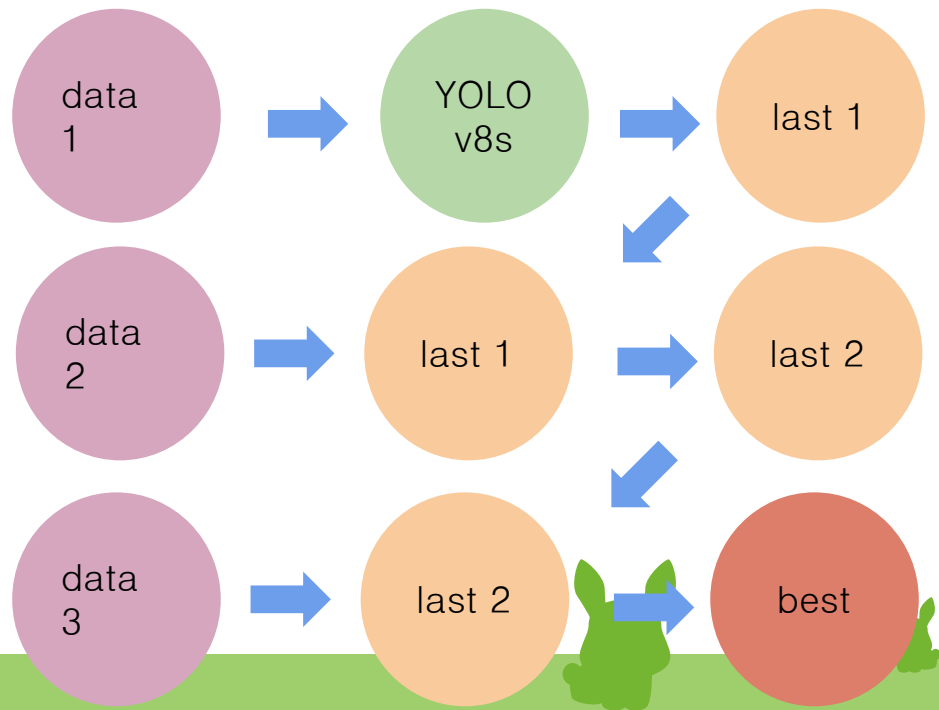
과적합 의심



# 개선 시도

학습 데이터에 비해 학습 할 파라미터가 많아 과적합이 생길 수도 있다고 판단, YOLOv8s 모델로 학습  
처음에 만든 3개의 데이터셋을 YOLOv8s를 통해 순서대로 학습

model	params (M)
YOLOv8n	3.2
YOLOv8s	11.2
YOLOv8m	25.9
YOLOv8l	43.7
YOLOv8x	68.2



# 개선 시도



이전에 비해 물체를 비교적 잘 잡는다.



# 추후 개선 사항

- 투명한 페트병을 유리병으로 인식하는 오인식 문제
- 리얼센스의 depth 기능을 사용해 물건을 집을 수 있도록 개선
- 로봇이 이동할 때, 바닥을 찍고 이동하는 문제



# Q&A

