지능화 캡스톤 프로젝트

프로젝트 #2 결과 발표

YOLOv8을 이용한 해상 객체 검출

2024. 6. 10.

충북대학교 산업인공지능학과

[4조] 최현동, 사수진, 이찬희

INDUSTRIAL AI RESEARCH CENTER

프로젝트 수행체계

수행방법

- 업무 분장을 통해 프로젝트를 진행하고 있습니다.
- 소통은 카카오톡으로 진행하며, 수요일마다 만남을 가져 진행을 하고 있습니다.

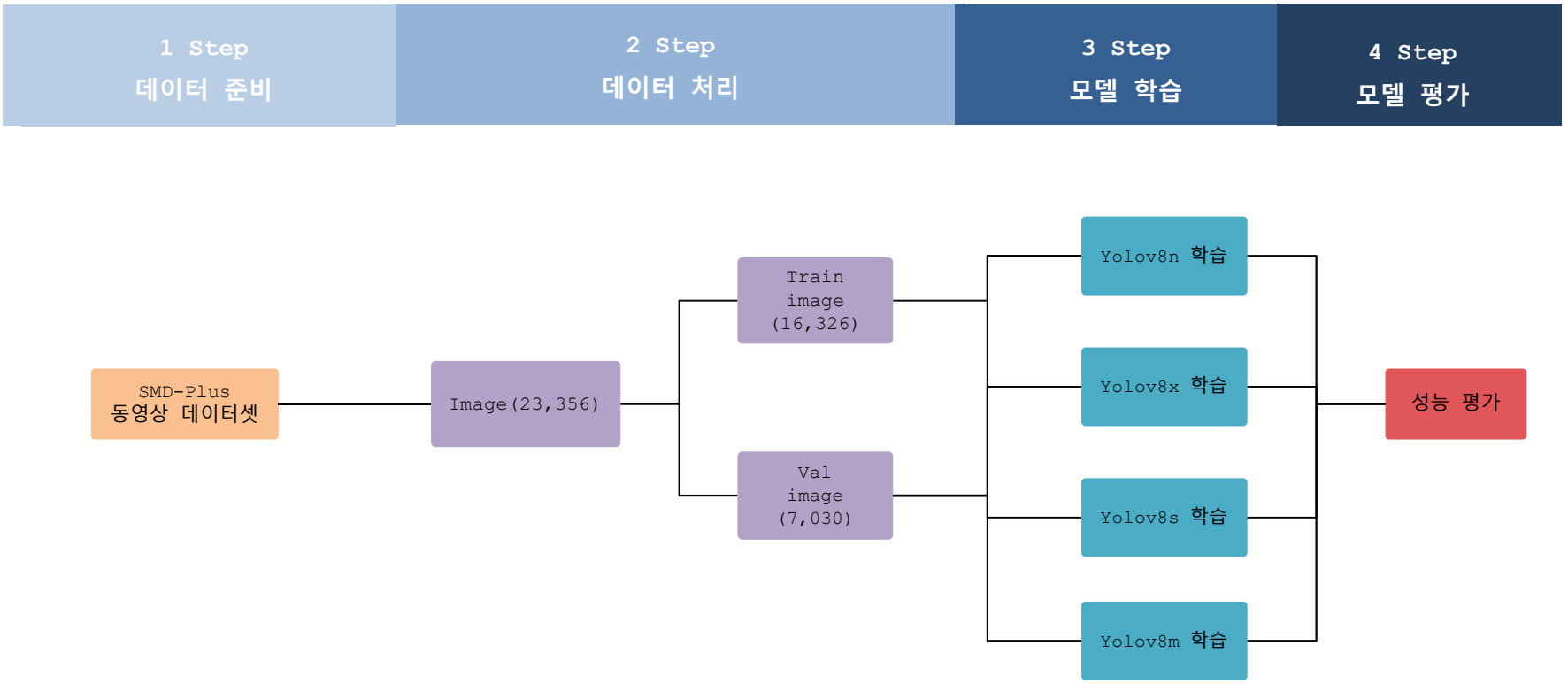
업무분장

이름	수행내용	비고
최현동	• 데이터 학습	
이찬희	• 데이터 학습 및 발표자료 작성 • 결과 발표	
사수진	• 데이터 수집 및 레이블링 작업 • 데이터 학습	

모델 개발 프로세스

우수성/차별성

- *우수한 성능의 모델을 개발하기 위해 취한 차별적인 방법론*
- *각 단계별로 그림으로 도식화...*
- *이후 장편부터 구체적으로 설명*



데이터셋

데이터셋의 구성

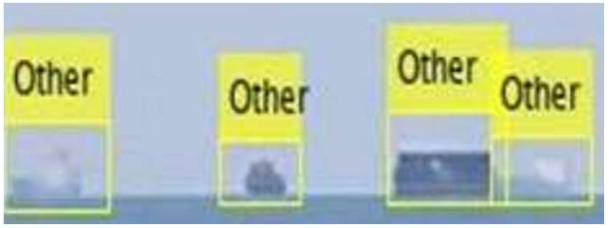
- SMD-Plus 공시 데이터셋 다운로드 : 총 51개의 동영상
- 51개의 동영상을 Train(37개) , Val(14개)로 나누고, Train 영상을 16,326개의 이미지로, Val 영상을 7,030개로 변환하였습니다.

[train]
Boat count: 2803
Vessel-ship count: 1722
Ferry count: 80163
Kayak count: 9419
Buoy count: 3115
Sail-boat count: 1360
other count: 16240

[val]
Boat count: 628
Vessel-ship count: 1935
Ferry count: 45722
Kayak count: 4603
Buoy count: 683
Sail-boat count: 566
other count: 8756

Boat

Ferry



Others

모델 학습방법

딥러닝 학습 환경

Hardware : 1) CPU : 11th Gen Intel® Core™ i7-11800H @ 2.3GHz (16 CPUs), ~2.3GHz

2) GPU : NVIDIA GeForce RTX 3060 Laptop GPU

3) Memory : 16GB RAM

4) epochs = 30,
batch = 10,
imgsz = 640,

5) 모델 : YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8x

```python
import multiprocessing
from ultralytics import YOLO

if __name__ == '__main__':
    multiprocessing.freeze_support() #windows에서는 필수
    #Load a model
    #model = YOLO("yolov8n.yaml") # build a new model from scratch
    model = YOLO(model='yolov8n.pt', task='detect')  # load a pretrained model (recommended for training)

    # Use the model
    model.train(data=r"C:\\Users\\Chan's Victus\\Desktop\\pythonProject\\capstonD\\data\\dataset\\data.yaml",
                epochs = 30,
                batch = 10,
                imgsz = 640,
                ) # train the model

    model.val() # evaluate model performance on the validation set
    model(r"C:\Users\Chan's Victus\Desktop\pythonProject\capstonD\data\dataset\images\train\MVI_0788_VIS_frame0.jpg") #predict on an image
    success = model.export(format='onnx') # export the model to ONNX format
```

제안하는 모델

Best 모델의 학습방법

- YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8x 4가지 모델 학습 비교
- epochs = 30, batch = 10, imgsz = 640,

학습 출력 결과

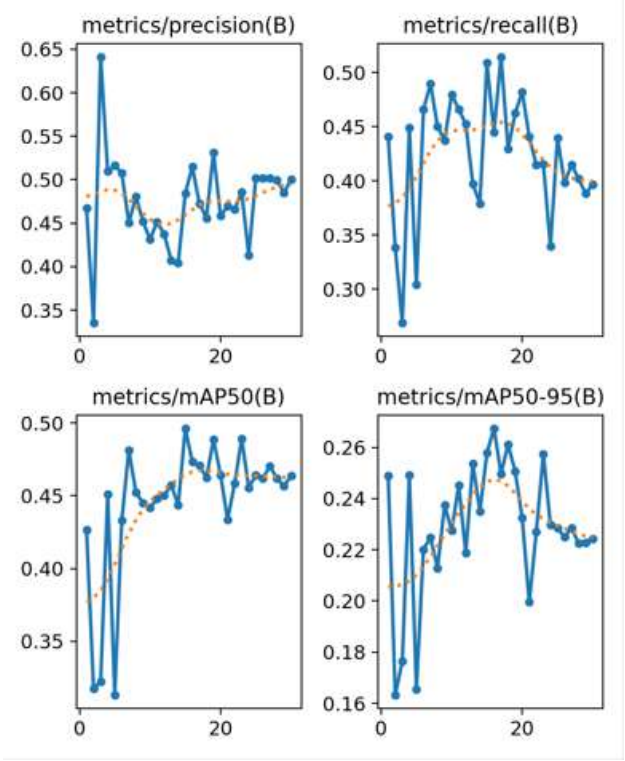
- 4가지 Model 학습 결과 각 항목별 BEST 모델은 아래와 같습니다.

1. Precision은 YOLOv8s
2. Recall은 YOLOv8m
3. mAP50은 YOLOv8s
4. mAP50-95는 YOLOv8m

model	metrics/precision(B)	metrics/recall(B)	metrics/mAP50(B)	metrics/mAP50-95(B)	학습시간(분)
YOLOv8n	0.54233	0.48382	0.52854	0.29584	90
YOLOv8s	0.80416	0.51226	0.60125	0.35279	120
YOLOv8m	0.67148	0.59172	0.57735	0.35853	270
YOLOv8x	0.53849	0.54212	0.56255	0.29509	780

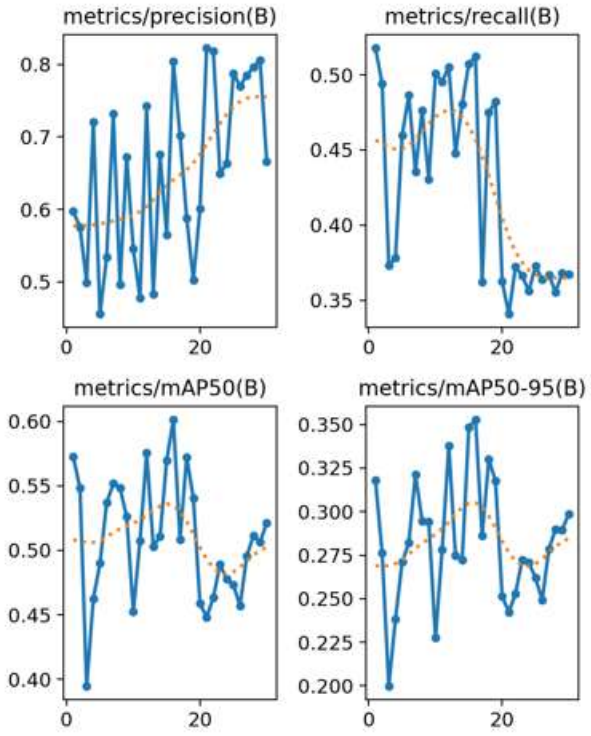
제안하는 모델

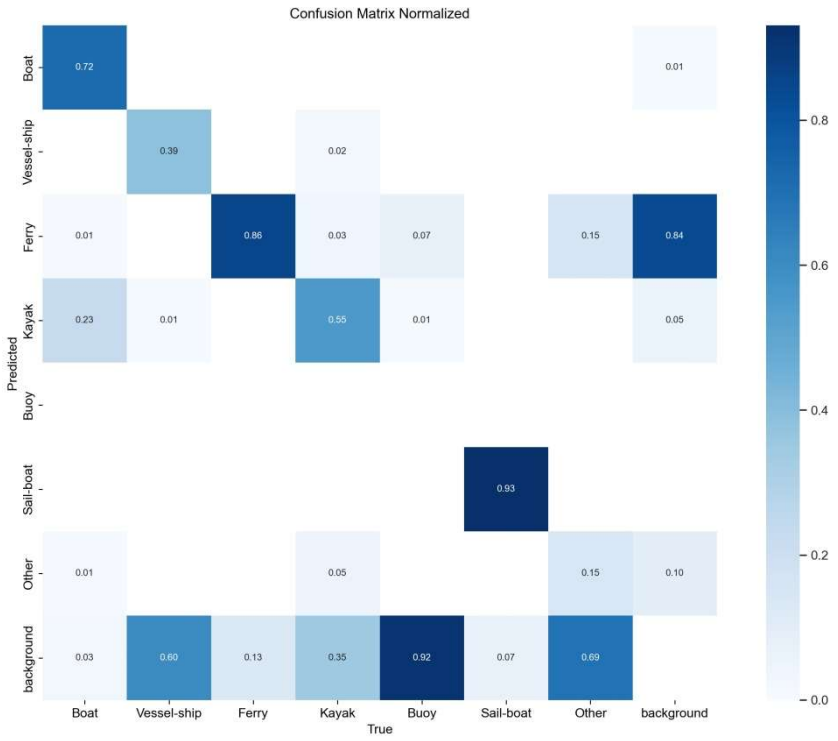
YOLOv8n 학습 곡선 및 성능



제안하는 모델

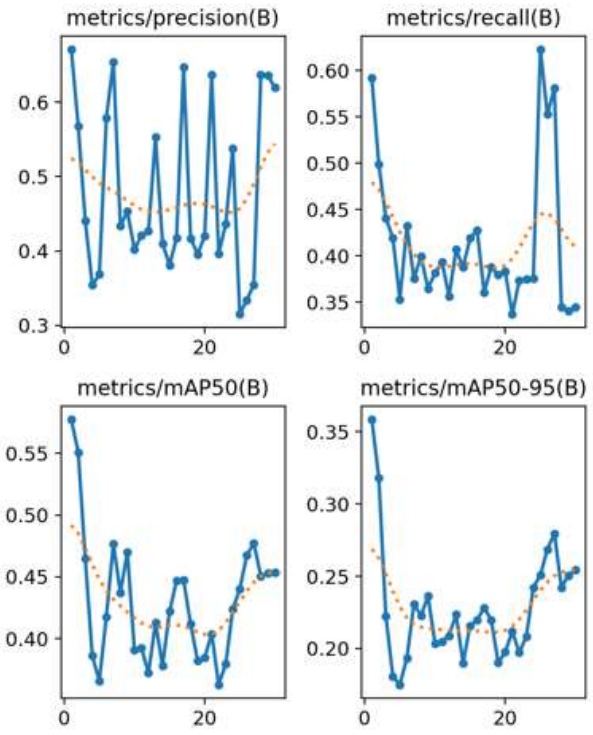
YOLOv8s 학습 곡선 및 성능





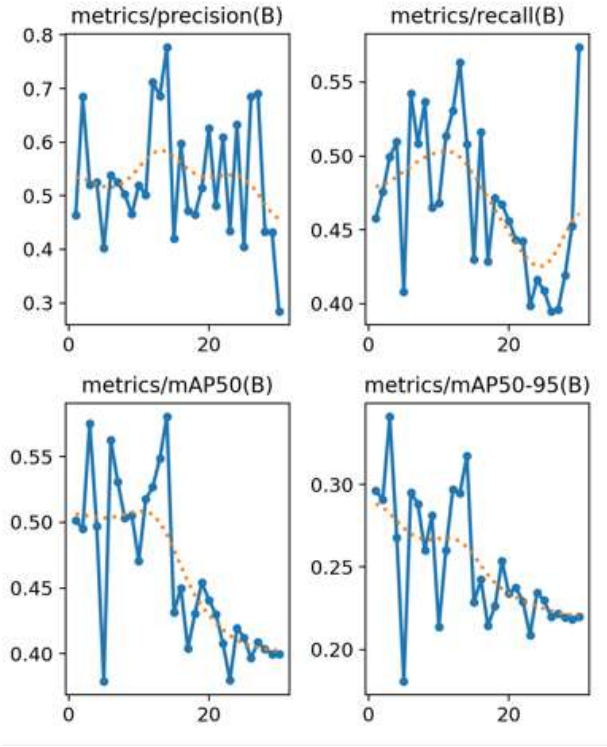
제안하는 모델

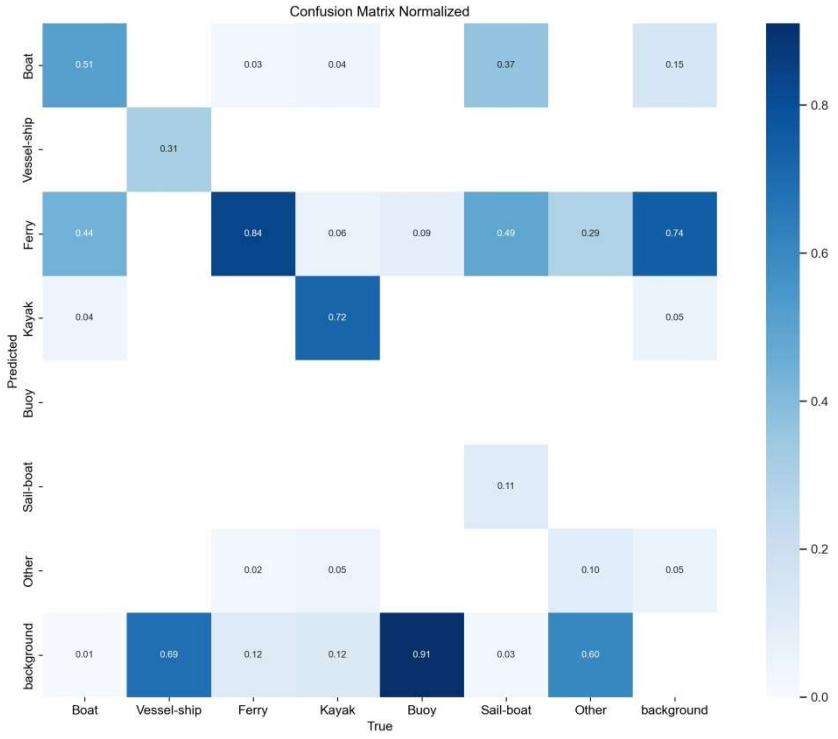
YOLOv8m 학습 곡선 및 성능



제안하는 모델

YOLOv8x 학습 곡선 및 성능





제안하는 모델

YOLOv8s 검출이미지 및 개선점

- 해당 결과를 보면 Label 데이터와 차이를 보이고 있는데, 학습 결과가 생각처럼 높게 나타나지 않아, 하이퍼 파라미터 튜닝 작업으로 성능을 좀 더 개선시킬 필요가 있어 보입니다.

label

검출 결과

모델 개발의 차별성

정확도 향상 방법

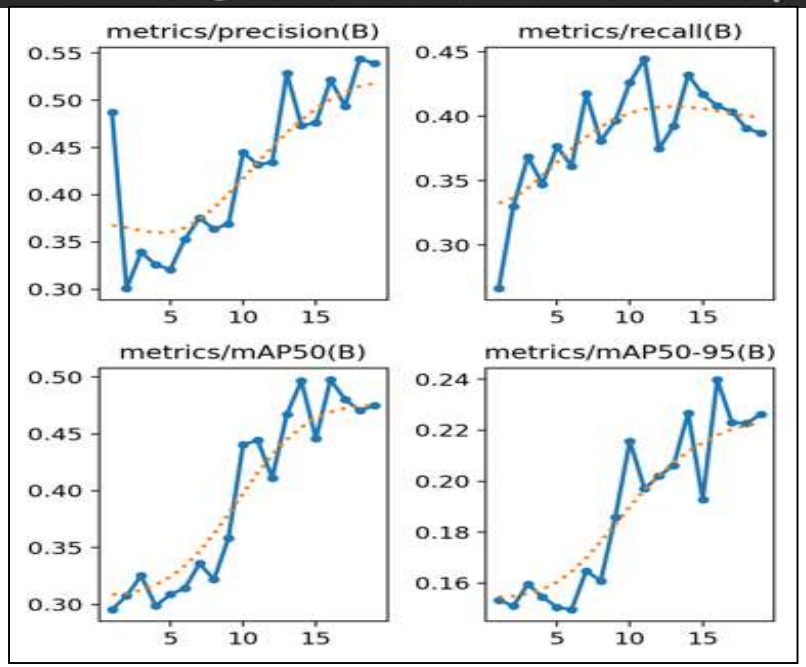
- 하이퍼 파라미터 최적화 (Optuna)를 통해 하이퍼 파라미터 튜닝 작업 시도.
- mAP50이 Maximize
- *Epoch : 10, 20*
- *Batch Size : -1, 10, 16*
- *img_size : 600, 800*
- *Learning rate: 0.00001, 0.01*

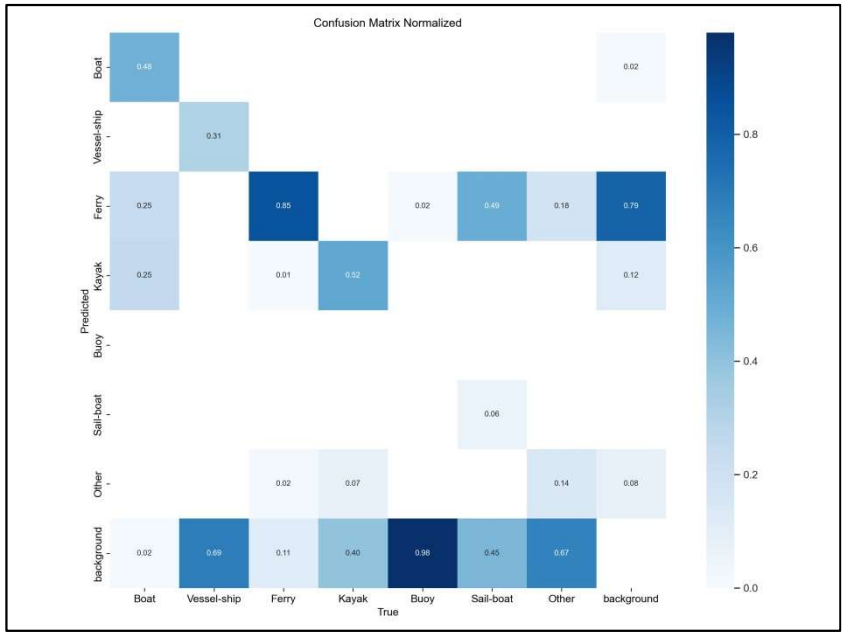
```python
def objective(trial):
    # Define the hyperparameters to optimize
    epochs = trial.suggest_int('epochs', 10, 20)
    batch_size = trial.suggest_categorical('batch_size', [-1, 10, 16])
    img_size = trial.suggest_int('img_size', 600, 800)
    learning_rate = trial.suggest_loguniform('learning_rate', 1e-5, 1e-2)
    # Evaluate the model performance on the validation set and return a metric to optimize
    results = model.val()
    # Use mAP(0.5) as the metric to optimize
    return results.box.map50  # Adjust this based on the actual results structure
```

모델 개발의 차별성

하이퍼 파라미터(Optuna) 최적화 결과

```
Best trial:
  Value: 0.5883412770393605
  Params:
    epochs: 19
    batch_size: 10
    img_size: 656
    learning_rate: 5.40815076230637 34e-05
```





model	metrics/precision(B)	metrics/recall(B)	metrics/mAP50(B)	metrics/mAP50-95(B)
YOLOv8n	0.54233	0.48382	0.52854	0.29584
YOLOv8s	0.80416	0.51226	0.60125	0.35279
YOLOv8m	0.67148	0.59172	0.57735	0.35853
YOLOv8x	0.53849	0.54212	0.56255	0.29509
YOLOv8s 하이퍼 파라미터 최적화	0.52152	0.40803	0.49774	0.23982

결과 및 토의

결과 요약

- 현재까지 진행사항으로는 YOLO의 4가지 모델(n,s,m,x)을 비교 하였으며, 그 중 s 모델이 해당 데이터에 대해 4 모델 중 가장 나은 성능을 보였습니다.
- YOLOv8s모델을 가지고 하이퍼 파라미터 최적화를 한 결과가 이전 튜닝 전 결과보다 낮게 나왔습니다.

고찰

- 4개의 모델과 하이퍼 파라미터 최적화를 한 결과에서 confusion matrix를 보게 되면 Bouy label을 아예 검출하지 못하는 걸로 나옵니다.
- Label에서 Bouy를 Other로 합치거나 제거한 채 학습했으면 좀 더 좋은 결과가 나타나지 않았을까 생각합니다.

감사합니다