YOLOv5를 이용한 과적 차량 탐지

AI 16기 조예찬

서론

과적차량 탐지는 도로 안전에 매우 중요한 역할을 합니다. 과적차량은 차선 침범, 교통 체증, 교통 사고 등 다양한 문제를 유발할 수 있기 때문입니다. 또한, 현재 우리나라에서는 대형화물차를 제한하기 위해 규제가 강화되고 있습니다.

따라서, 이번 프로젝트는 YOLOv5 모델을 이용하여 과적차량을 자동으로 탐지하고 이를 활용하여 교통 안전을 강화하는 것을 목표로 합니다. YOLOv5는 높은 정확도와 빠른 속도로 객체를 탐지할 수 있는 모델입니다.

이번 프로젝트를 통해 과적차량 탐지 기술의 발전에 기여하고, 도로 안전에 기여함으로써 국민의 생명과 재산을 보호하는 데 도움을 줄 것으로 기대합니다.

모델 소개

YOLOv5는 You Only Look Once(한 번만 보면 된다)라는 이름에서 유래한 물체 감지 알고리즘입니다. YOLOv5는 YOLOv4의 개선 버전으로, 기존 모델보다 더 높은 정확도와 빠른 속도를 가지고 있습니다.

- 빠른 속도: YOLOv5는 실시간으로 객체를 탐지할 수 있을 만큼 빠른 탐지 속도를 보여줍니다.
- 높은 정확도: YOLOv5는 이전 버전의 모델들보다 더욱 높은 정확도를 가지고 있습니다.
- 자동 스케일: YOLOv5는 다양한 크기의 이미지를 자동으로 스케일링해 탐지할 수 있습니다.
- PyTorch 기반: YOLOv5는 PyTorch 기반으로 개발되어 있어서 사용하기 수월합니다.

데이터셋 소개

주어진 데이터셋 트리구조

```
New_Sample
├── 원천 데이터
│   └── TS1.대형차
│       ├── 불법 차량
│       │   └── *.jpg
│       └── 정상 차량
│           └── *.jpg
└── 라벨링 데이터
    └── TS1.대형차
        ├── 불법 차량
        │   └── *.json
        └── 정상 차량
            └── *.json
```

모델에서 요구하는 트리구조

```
New_Sample
├── image
│   ├── train
│   │   └── *.jpg
│   └── val
│       └── *.jpg
└── labels
    ├── train
    │   └── *.txt
    └── val
        └── *.txt
```

Label 구조


```
"FILE":[{
        "FILE_NAME":"A01_B01_C04_D01_0703_E06_F07_1_1.jpg",
        "COLLECTIONMETHOD":"CCTV",
        "DAY/NIGHT":"주",
        "LANE":"다선",
        "ROADNUMBER":"아트센터대로",
        "PLACE":"아트센터교2(송도동73)",
        "IDCODE":"F07",
        "DATE":"2021.07.03",
        "WEATHER":"맑은날",
        "RESOLUTION":"1920*1080",
        "MAKE":"한화테크윈",
        "MODELNAME":"XNO-6020R",
        "FILESIZE":"333392",
        "BOUNDINGCOUNT":"1",
        "ITEMS":[
                {
                "DRAWING":"Box",
                "SEGMENT":"대형차",
                "BOX":"722.02,479.77,456.74,307.75",
                "POLYGON":"",
                "PACKAGE":"불법차량",
                "CLASS":"적재불량",
                "COVER":"덮개개방",
                "COURSE":"후면우측",
                "CURVE":"정상주행"
                }
```

모델 훈련

모델 훈련의 개요

1. label의 데이터를 yolov5가 사용할 수 있는 형태의 txt파일로 만들어준다.
({레이블} {x_center} {y_center} {width} {height} 형태)

2. 이미지를 train과 validation으로 나누어 그 정보를 txt파일에 저장한다.

3. 레이블과 이미지셋 정보를 담은 yaml파일을 생성한다.

4. yaml파일과 yolov5s모델을 통해 학습시킨다.

모델 훈련

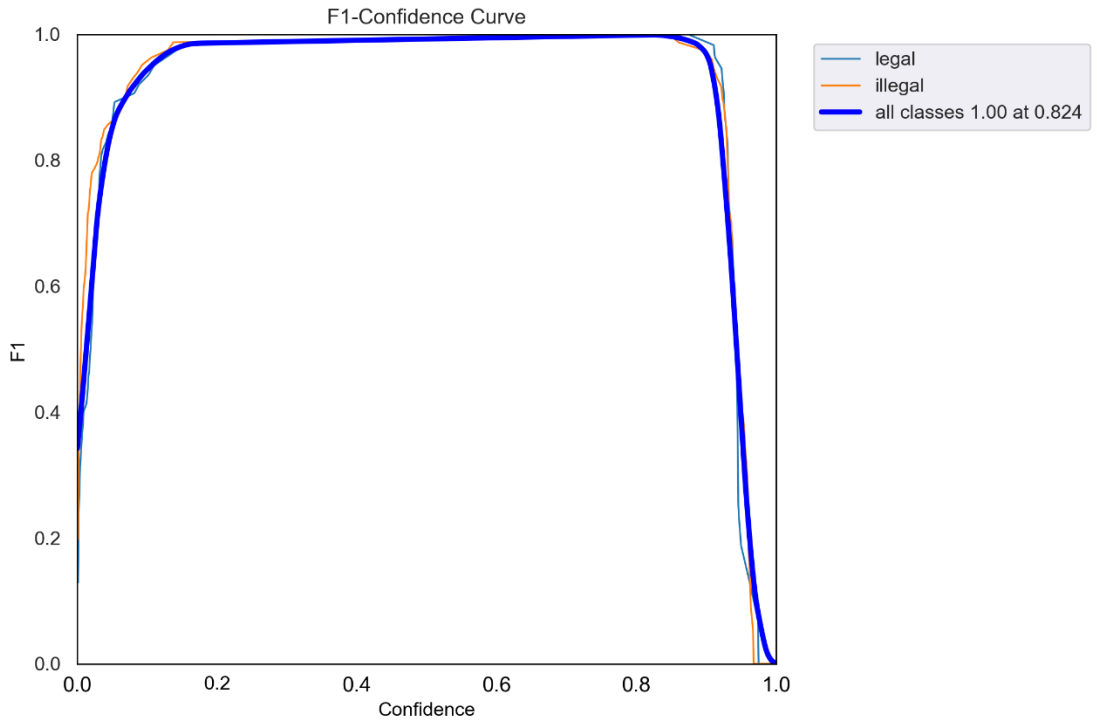
```
# Label 데이터의 json을 txt로 변환
json_files = [f for f in os.listdir('.\data\json') if f.endswith('.json')]


for json_file in json_files:
    with open(os.path.join('.\data\json', json_file), encoding='utf-8') as f:
        data = json.load(f)
        items = data['FILE'][0]['ITEMS']
        img_width, img_height = map(
            int, data['FILE'][0]['RESOLUTION'].split("*"))

        classes = {'정상차량': 0, '불법차량': 1}

        for item in items:
            class_name = item["PACKAGE"]
            class_id = classes[class_name]

            box = item['BOX'].split(',')
            x1, y1, x2, y2 = map(float, box)

            dw, dh = 1. / img_width, 1. / img_height

            x, y, w, h = x1 + x2 / 2.0, y1 + y2 / 2.0, x2, y2

            x_center, width = x * dw, w * dw
            y_center, height = y * dh, h * dh

            yolo_label = f"{class_id} {x_center} {y_center} {width} {height}"

            filename = os.path.splitext(json_file)[0] + '.txt'
            with open(os.path.join('.\dataset\labels', filename), 'w') as f:
                f.write(yolo_label)
```

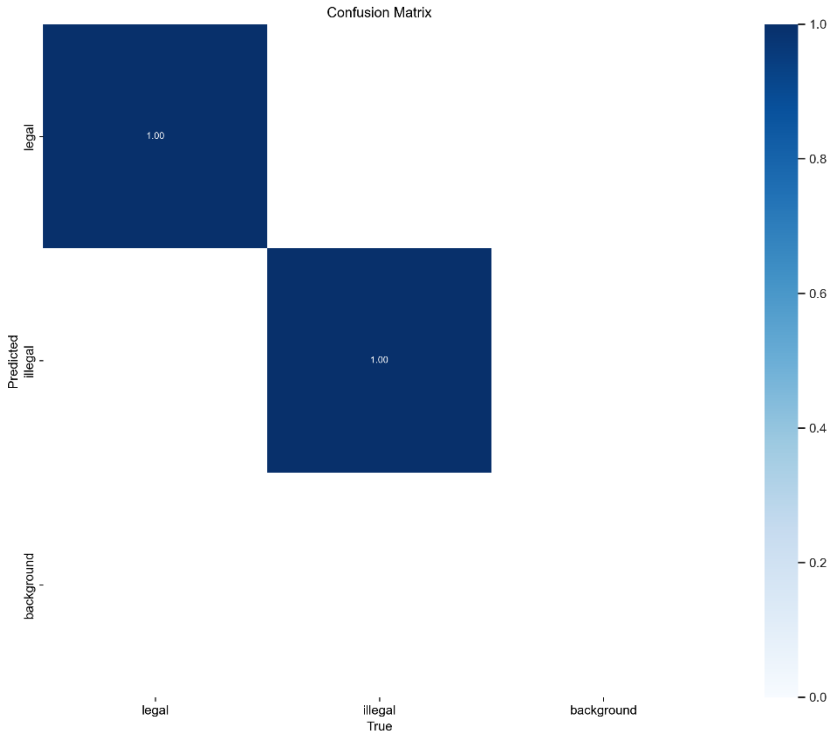
```
# 이미지 리스트를 받아오기
def get_image_list(directory):
    return [os.path.join(directory, f) for f in os.listdir(directory) if f.endswith('.jpg')]

# 이미지 리스트를 test셋과 val셋으로 나누기
def split_data(train_size=0.8):
    img_list = get_image_list('./dataset/images')
    train_img_list, val_img_list = train_test_split(img_list, test_size=1-train_size)
    with open('./dataset/train.txt', 'w') as f:
        f.write('\n'.join(train_img_list) + '\n')
    with open('./dataset/val.txt', 'w') as f:
        f.write('\n'.join(val_img_list) + '\n')

split_data()
```

```
# 레이블과 데이터에 맞는 yaml파일을 만든 뒤 학습을 진행

!python ./models/yolov5/train.py --img 640 --batch 16 --epochs 50 --data ./dataset/data.yaml --cfg ./models/yolov5/models/yolov5s.yaml --weights yolov5.pt --name final_truck_yolov5s_results
```

성능 평가 및 결과

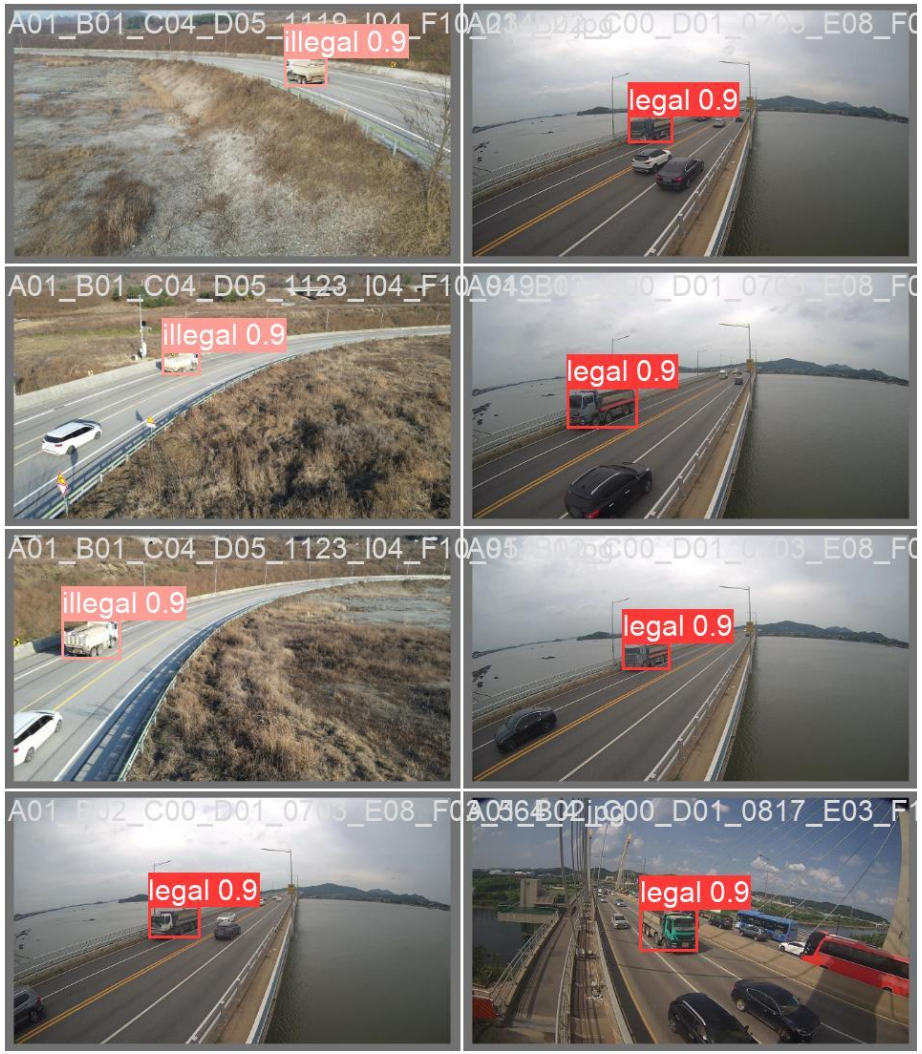
F1 Curve와 Confusion Matrix를 확인했을 때, 학습의 결과는 괜찮게 나옴을 확인하였다.





성능 평가 및 결과

Val set와 아예 다른 data를 사용했을 때 얼추 과적차량을 구분하는 것으로 보인다.



결론

트럭이 아닌 버스나 승용차를 인식하는 문제가 있다.

이번 모델을 학습시킬 때 사용한 데이터의 수는 170개 였다.

더 나은 모델을 얻기 위해서는 다양한 차종과 상황에 대한 데이터가 필요할 것 같다.