[ main.py ]



주어진 어노테이션 JSON 파일을 불러와 YOLO학습을 위해 카테고리 ID와 좌표를 라벨링하는 코드이다.

training, detection, validationn 세 개의 데이터 모두 각각 yolo\_\* 폴더에 images., labels 폴더로 분류하여 저장한다.

[ 학습 및 검증 ]

코드 실행이 끝나고 나면 생성된 yolo 맞춤 데이터로 yolo5로 training, detection, validation을 진행한다.

프로젝트 폴더를 터미널로 열어서 yolo5 리포지토리를 클론한 후, 생성된 yolov5 폴더로 이동 후, data.yaml 파일을 다음과 같이 작성하여 저장한다.



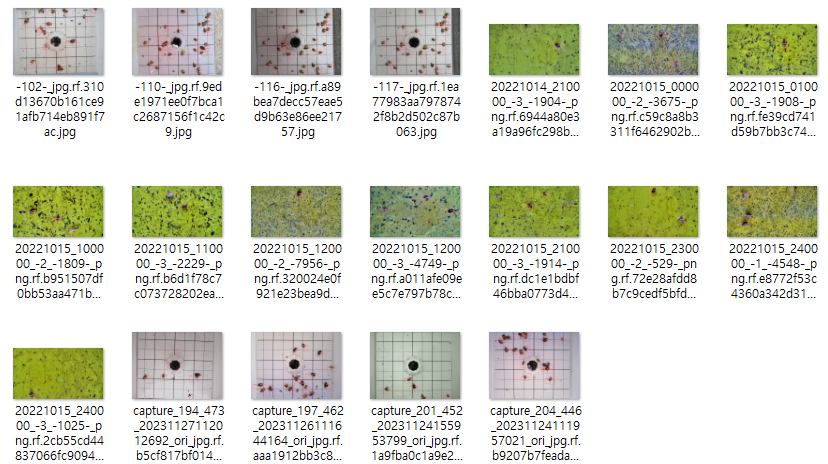
성공적으로 파일을 생성하여 저장한 후에 터미널을 열어 아래 명령어를 실행한다.

**$ python train.py --img 416 --batch 16 --epochs 100 --data data.yaml --weights yolov5s.pt”**

해당 명령어를 실행하여 사전에 생성한 yolo 맞춤 데이터로 학습을 진행하면, yolov5/runs/train/exp 폴더에 학습 수행 진행 과정을 담은 여러 시각화 그래프와 학습 가중치 파일이 저장된 폴더가 생성된다. 해당 학습 가중치 파일을 활용하여 아래의 코드로 detection을 진행한다.

**$ python detect.py --weights runs/train/exp2/weights/best.pt --img 416 --task test**

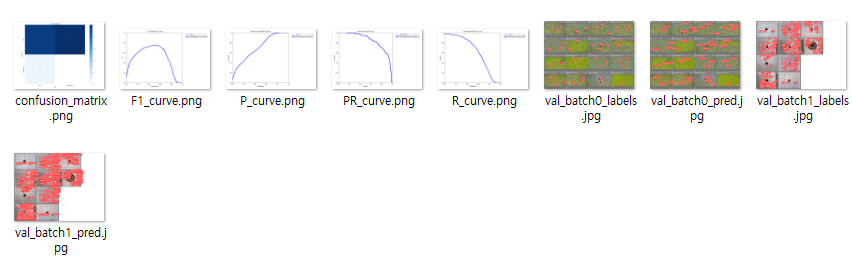
해당 명령을 실행하면, 마찬가지로 yolov5/runs/detect/exp 폴더에 학습한 가중치 파일을 토대로 입력된 이미지들에 대한 탐지를 수행하며, 결과는 아래와 같다.



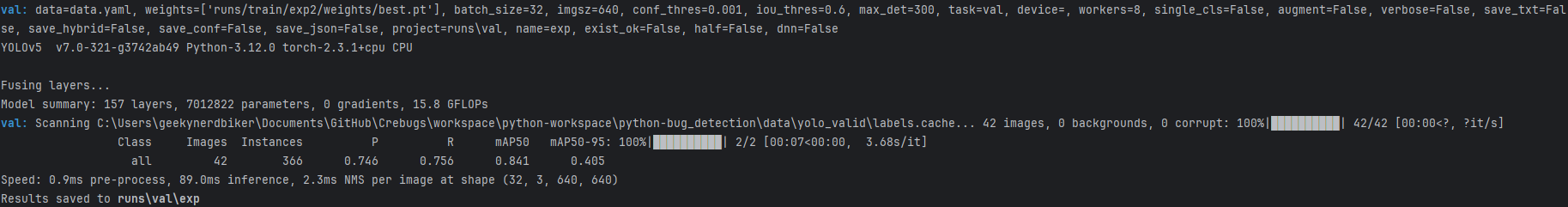
이후 마지막으로 검증을 위해 아래의 명령어를 실행한다.

**$ python val.py --weights runs/train/exp2/weights/best.pt --data data.yaml --task val**

validation 폴더의 images로 성능 검증을 진행하며, 마찬가지로 yolov5/runs/val/exp 폴더에 파일들이 생성되며, 실행 결과는 다음과 같다.



또한 명령 실행으로 인해 출력된 결과를 통해 각종 스코어를 알 수 있으며, 결과는 아래와 같다.



[ 특이사항 ]

데이터 어노테이션 파일과 데이터를 살펴보면, 카테고리가 0, 1 두개가 존재하는데, 실질적으로 데이터를 살펴보면 category\_id가 0인 데이터가 존재하지 않는다. 즉, 하나의 클래스만 존재한다. 그래서 데이터 컨버팅 과정에서 카테고리 아이디를 모두 1이 아닌 0으로 변환하여 라벨링을 진행했다.