AI야! 교통안전을 부탁해~

2021 인공지능 학습용 데이터 해커톤 대회

보행안전 도로시설물 부문

2021.12.13

김건희, 김지은, 오형석, 이현진, 임다현, 최진규

INDEX

Hackhathon

PART 01

과제설명

채점기준 - 프로세스

PART 02

전처리

양호 / 불량 분류
이미지 자르기

과제 설명

기준	반영률
이상 이미지/프레임 탐지	35%
이상 이미지/프레임 분류	35%
이미지/프레임 內 이상 영역 탐지	20%
기타 창의성, 활용성 등	10%

- 평가용 데이터의 이미지/프레임은 원칙적으로 단일 클래스로 정의
  (복수 클래스 정의가 가능한 경우들에 대해서는 분류 예측 결과가 복수 클래스 중 하나에 포함되기만 하면 분류 예측을 성공한 것으로 간주)
- 최종 프로그램의 성능 평가 점수의 총점은 100점
  (이상 이미지/프레임 탐지 40점, 분류 40점, 이미지/프레임 내에서의 이상 영역 탐지 20점) - F1 score 기반
- 심사용 테스트 데이터의 양은 제공되었던 기본 데이터 양의 10% 수준
- 프로그램의 처리 속도에 대한 제한은 크게 없음
  심사용 테스트 프로그램 정상 구동 후 2시간 초과시에도 Inference가 완료되지 않은 데이터 분에 대해서는 미판정 처리
- 심사 중 프로그램 및 파라미터 등 일체의 중간 수정은 불가

과제 설명

데이터 전처리

이미지 분류

.json파일 기준
양호/불량 파일 구분

이미지 자르기

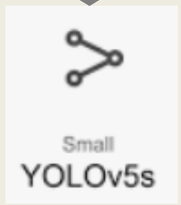
.json파일의
시설물 좌표대로 자르기

불량인 경우
불량부분의 이미지는
part파일로 이동

도로시설물 인식

시설물 좌표 라벨링
augmentation

600장
30개 카테고리
18000장



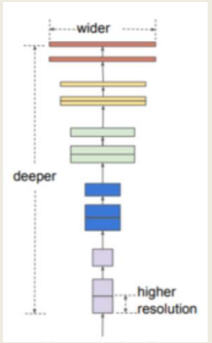
모델 훈련

시설물 인식

도로시설물 양호 / 불량 판별

train / test 분류
augmentation

모델 선정



EfficientNet
b0와 b1 사용

튜닝

batch size
normalize
learning rate
weight decay
epoch
optimizer

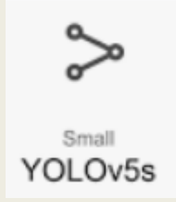
30가지 시설물의 best.pt

도로시설물 인식 결과 기반 양호 / 불량 판별

도로시설물 불량부분 판별

. json파일 기반 불량부분 좌표
좌표계 변환 COCO -> YOLO

도로시설물 양호 / 불량 판별 결과 불량일 경우



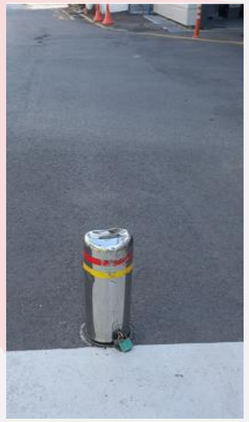
모델 훈련

불량 부분 인식

전처리

- 01 Bollard_스테인리스
- 02 Bollard_탄성고무
- 03 시선유도봉_ 2줄
- 04 시선유도봉_3줄
- 05 보행자용 방호울타리
- 06 교량용 방호울타리
- 07 턱낮추기
- 08 경사로
- 09 점자블럭
- 10 (도로안내표지) 지주
- 11 보도(시멘트 콘크리트)
- 12 보도 블록
- 13 자전거 도로
- 14 연석
- 15 무단횡단 방지 울타리
- 16 맨홀
- 17 중앙차로 버스정차대
- 18 가로변 버스정차대
- 19 가로변 택시정차대
- 20 방음벽
- 21 현장 신호제어기
- 22 보행자 작동신호기
- 23 시각장애인용 음향신호기
- 24 과속방지턱
- 25 횡단보도
- 26 고원식횡단보도
- 27 통합표지
- 28 정주식, 부착식 표지
- 29 (가로등) 지주
- 30 (전봇대) 빗금표시

1. 불량, 양호, 불량부분 분류



불량

양호

.json파일 annotation 값 중
ls_defect : 양호(0) vs 불량(1)

- 0
- 1

- 0
- 1
- part

불량 시설물의 json 파일의
annotation 값이 여러 개인 경우
1 : 불량인 시설물 전체 자른 파일
Part : 불량 부분만을 모아둔 파일

2. 이미지 자르기

.json파일 annotation 값 중
annotation_box : bbox

```
[930.2178352499878,
1185.1199938322368,
406.4431903139217,
1673.4315789473683]
```

annotation_info의 정보대로 이미지 자르기

annotation_box : polygon

```
[492.8675771485776,345.5820933910505],
[629.6508858659785,1033.599519939833],
[597.2548390644888,1037.1923002351268],
[651.248250400305,1069.527322892772],
[750.2361711826345,1055.1562017115964]
```

annotation_info의
x값 최대, 최소
y값 최대, 최소 → 직사각형 형식으로 이미지 자르기

Hackhathon

PART
03

도로 시설물 인식

Image Augmentation
YoLov5

Data Augmentation

Image Augmentation?

데이터의 양을 늘리기 위해 **원본에 각종 변환을 적용**하여 **개수를 증강**시키는 기법이다.

2가지 다른 방법을 사용하여 Image Augmentation을 진행하였다.
#1. 인식에서의 Augmentation, #2. 불량 / 양호 판별을 위한 Augmentation

Data Augmentation

#1. 인식

불량인 사진과 양호인 사진의 각기 다른 Augmentation 옵션을 부여하고 진행.
roboflow 를 사용하여 진행.

불량

Image Size : 416 x 416

Brightness : Between -15 % and 15 %
Blur : Up to 2.25px
Noise : Up to 3% of pixels

Training Set : 92% / Training Set : 8%

Output Image quantity : x3amount

양호

Image Size : 416 x 416

Brightness : Between -15 % and 15 %
Blur : Up to 2.25px

Training Set : 92% / Training Set : 8%

Output Image quantity : x3amount

YoLOv5s

YoLov5

YOLOv5 🚀 (You Only Look Once)
: ultralytics 이라는 곳에서 제공해주는 Object Detection 무료 오픈소스다.

As is

1. 주어진 30개 카테고리 이미지를 roboflow 를 통해 YoLOv5에 Training 시킬 수 있도록 전처리를 한다.
   카테고리별 280 ~ 300개의 이미지,
   대략 10000장의 데이터셋을 구축하여
   Training을 실시했다.

2. 환경은 GoogleColab(Pro) 에서 실시하였으며, RunTimeError를 방지하기 위해, 최대 Epoch를 180번으로 지정했다.

3. 카테고리별 인식률을 F1-Score를 통해 확인한다.

4. 2번의 Training 결과 얻어진 Model을 통해 임의의 Test 데이터를 검증한다.

To Be

30개의 카테고리에서의 F1-Score

⬇

대략 0.7

*출처 : https://github.com/ultralytics/yolov5

Hackhathon

PART
04

도로시설물
양호 / 불량 판별

Image Augmentation
CNN 개념 정리
EfficentNet

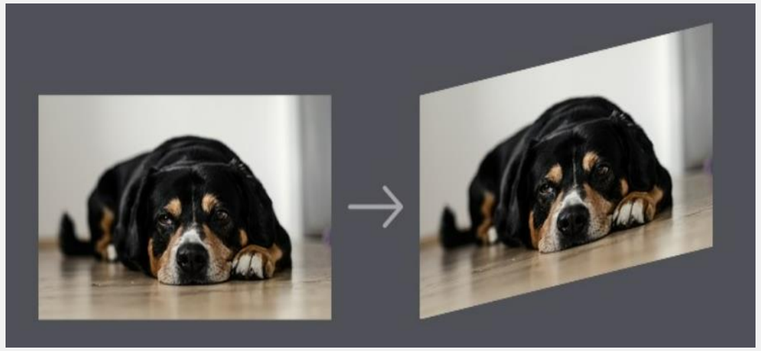
Data Augmentation

#2. 불량 / 양호

Python 내에서 좌우 반전, 기울이기, 노이즈 추가해서 추가적으로 Augmentation을 진행.

좌우 반전

기울이기

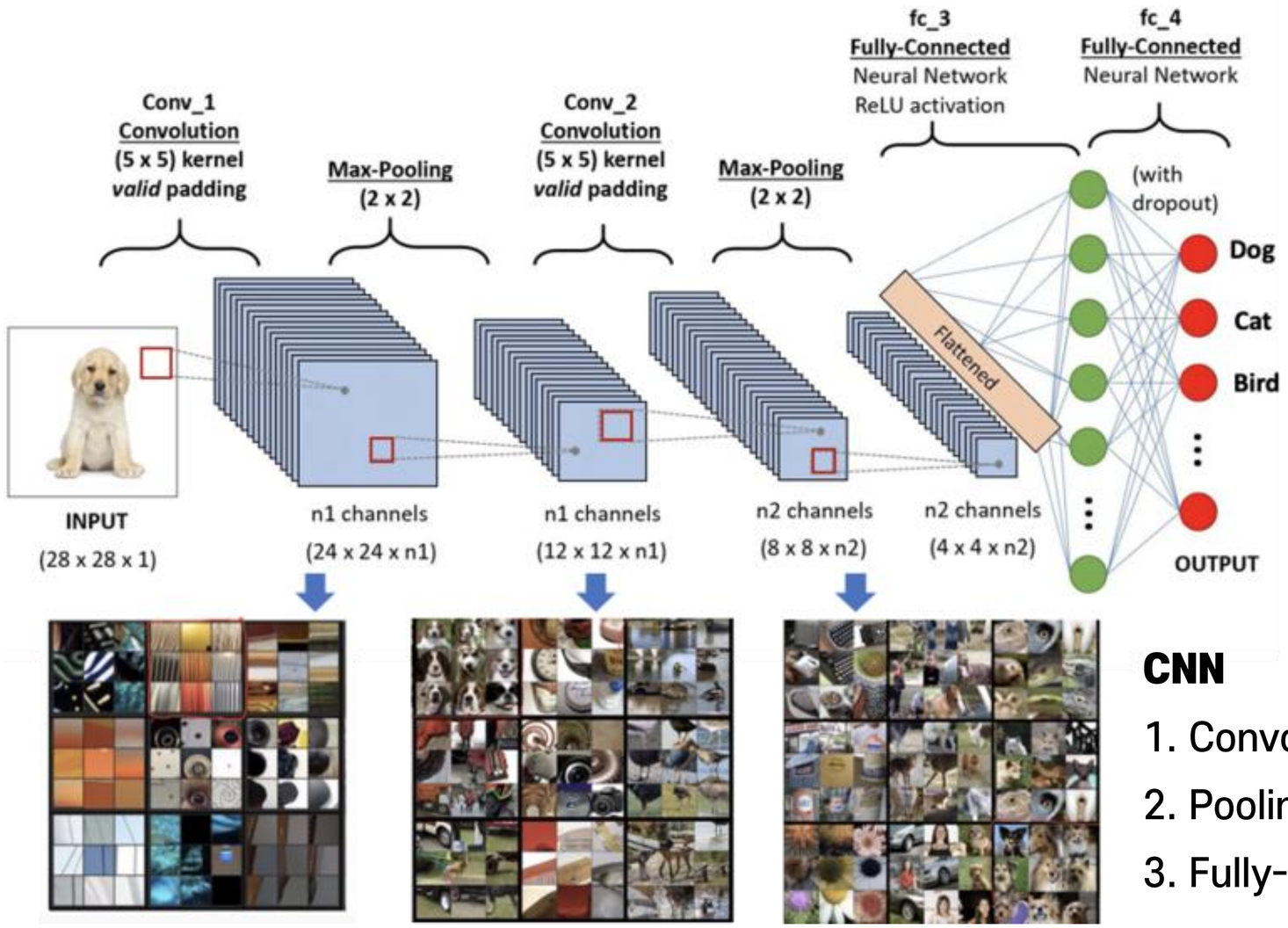


노이즈 추가

데이터를 Train / Test 로 나눈 후, Train 데이터에만 +300장 Augmentation을 적용.

그 후 *Efficient Net을 이용해 학습시키기 전에 Efficient Net Version에 맞게끔
이미지의 크기를 조정해주었다.

CNN 개념 정리



CNN

1. Convolution layer : 특징 추출(feature extraction)
2. Pooling layer : 특징 추출(feature extraction)
3. Fully-connected layer : 분류(classificaiton)

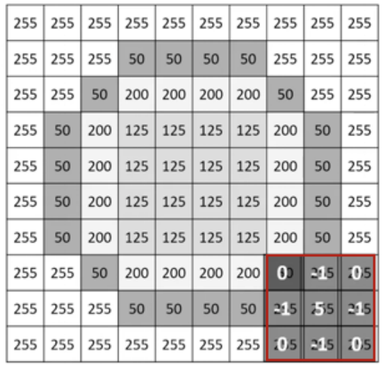
이미지 분류 딥러닝 모델은 대부분 **CNN**을 **수정/보완하거나 발전시킨 형태**

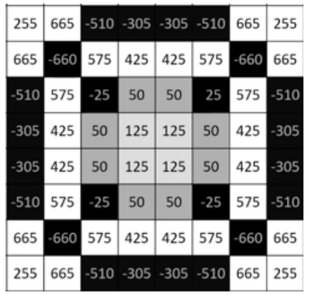
CNN 개념 정리

Convolution

입력값 이미지의 모든 영역에 필터를 적용해 *Convolution연산 처리하는 것

Filter를 통과한 이미지는 색상, 선, 형태, 경계 등의 특징이 뚜렷해짐

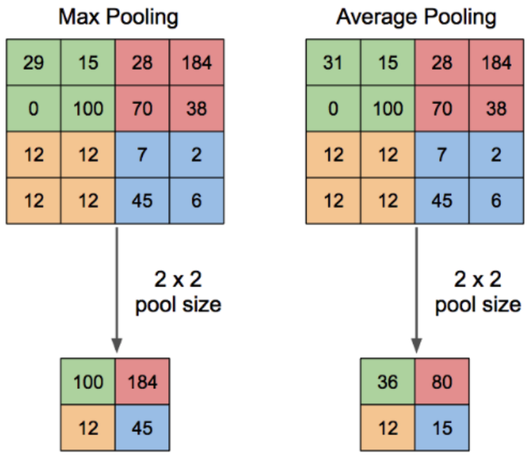




Pooling

Convolution의 결과값 중

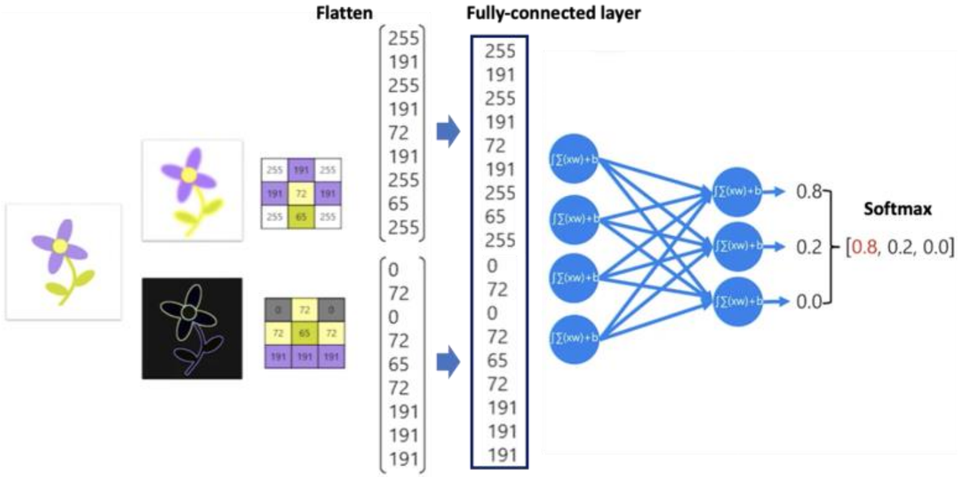
유의미한 부분만 선택해 크기를 줄이는 과정



Fully-Connected layer

결과값을 하나의 벡터로 연결하여

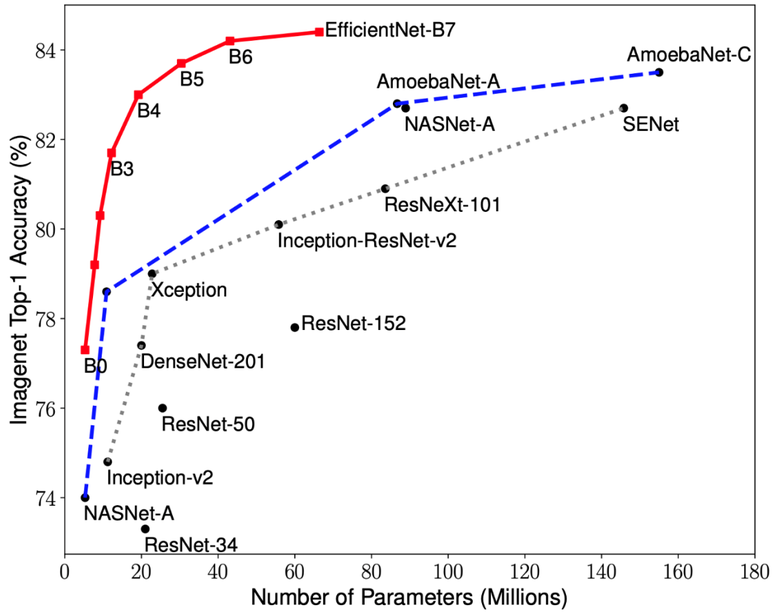
가장 확률이 높은 class를 산출



* 출처 : https://cs231n.github.io/convolutional-networks/#add

분류 모델 선정

ResNet　VGGNet　GoogleNet　EfficientNet



EfficientNet 선정 이유

① 딥러닝 분야의 최신 모델(2019)

② 파라미터의 수 대비 분류 성능이 뛰어남
(연산량 ↓ 성능↑)

학습 시간 대비 성능을 고려해
EfficientNet-B0, B1 사용

Efficient Net

모델 성능을 높이기 위한 3가지 방법

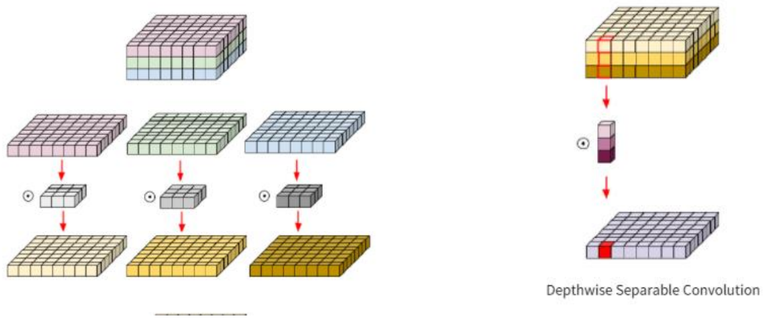
- Layer의 개수 ↑
- Filter의 개수 ↑
- Image의 해상도 ↑

→ EfficientNet

Compund Scaling : AutoML을 통해
세가지 방법에 대한 최적 조합을 찾은 모델

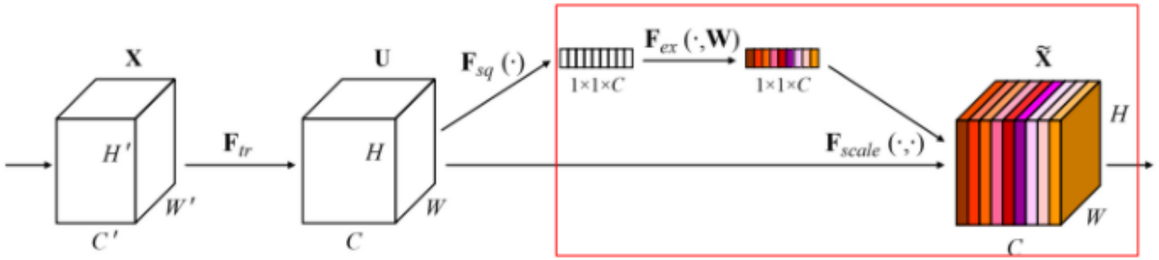
특징 ① MBConV

https://github.com/qubvel/efficientnet



② Squeeze-and-excitation

채널의 상대적 중요도를 산출해
Feature Map에 곱함

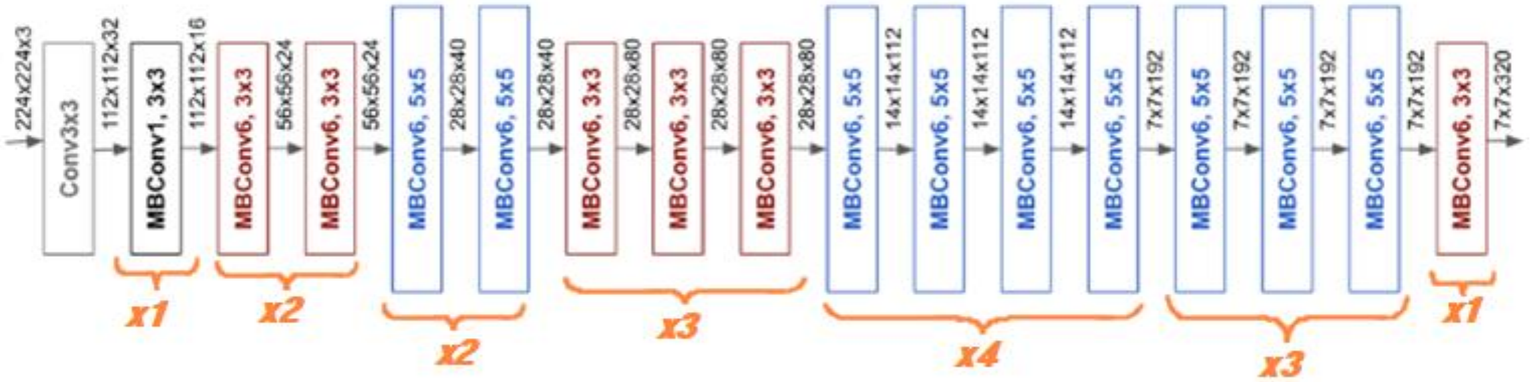


* 출처 : https://github.com/qubvel/efficientnet

Efficient Net Architectrue

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	28×28	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

반복 횟수



* 출처 : https://www.youtube.com/watch?v=fR_0o25kigM&list=PLhhyoLH6IjfxeoooqP9rhU3HJIAVAJ3Vz&index=20

학습 과정

데이터셋 구축

도로시설물
30종
양호/불량
이미지
데이터

Split

Train

비율 80%

Test

비율 20%

Data Augmentation

Augmentation

좌우 반전
기울이기
노이즈 추가

Hyper Parameter

Resize
Normalization
Batch Size
Loss function
Optimizer
Learning Rate
Weight Decay
Epoch

Model

EfficientNet

Train

Result

Pt파일 저장

최종
Model 선정

Model Test

양호 불량 판별
F1 Score 산출

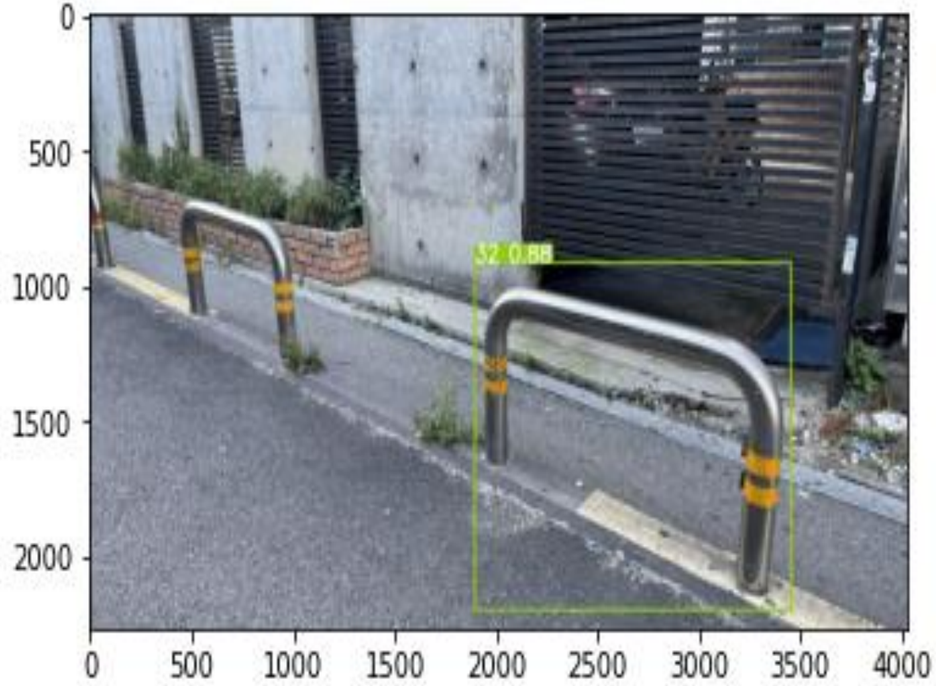
```
[Test Phase] Loss: 1.6835 Acc: 70.0000
precision: 0.7547169811320755
recall: 0.7843137254901961
f1_score: 0.7692307692307692
```

Result

불량 / 양호 판별

최종적으로 EfficientNet을 통과하면 { Label : 0,1, Category : Category명} 을 도출하게 된다.

양호



Label : 0
Category : 32

불량

Label : 1
Category : 26

Hackhathon

PART
05

도로시설물
불량부분 인식

YoLOv5
불량부분 라벨링

YoLOv5

YoLOv5

#2. 불량 부분 검출

As is

EfficientNet을 통과한 후 불량이라고 식별된 이미지만을

1. 30개의 카테고리에서 **불량 이미지만**을 *CoCo좌표계로 존재하는 .JSON 형식의 파일을 *YoLo 좌표계로 변환 후 제공 받은 모든 이미지를 데이터셋으로 구축하였다.

2. 그 후 GoogleColab(Pro) 환경에서180번의 Epoch로 Training을 시켰다.

3. 카테고리별 불량 판별율을 F1-Score를 통해 확인한다.

4. 2번의 Training 결과 얻어진 Model을 통해 임의의 Test데이터를 검증한다.

To Be

*출처 : https://velog.io/@jaeha0725/Object-Detection-Label-COCO-YOLO-KITTI

불량부분 라벨링

불량

Label : 1
Category : 26

불량이라고 인식된 부분의
YoLO 좌표계를 도출하여

CoCo 좌표계로 바꿔준 후,
최종 .txt 파일로 저장한다.

Label : 1 (불량, 불량부분)
Category : 26
Area : [x, y, w, h]

한계점

01

Google Drive
용량 한계

02

Colab 환경에서의
Runtime 한계

03

불량 부분에서
이미지의 유사성
으로 인한
부족한 불량부분
인식률 ↓