차량 이미지의 스크래치 위치 파악을 위한 Image Segmentation 모델 개발

AIB 11기 - 김현승

목차

1

프로젝트 개요

배경

SOCAR

Greencar

하루 약 100,000장

시간, 비용

목적

기업에서 제공 받은 데이터 + AI-Hub 데이터

↓

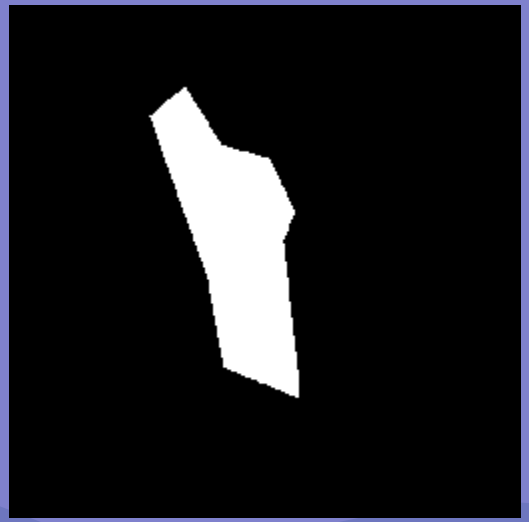
명확한 스크래치는 파악할 수 있는 수준의
Image Segmentation 모델

↓

IoU Score 0.20 이상

구현 내용

“Input Image를 받아서 스크래치가 난 부분은 1,
스크래치 외의 부분은 0으로 예측하는 모델을 만든다.”



개발 환경 및 파이프 라인

+colab

데이터 수집

NEXTLab

AI Hub

데이터 전처리

python

OpenCV

모델링

PYTORCH

U-Net

U-Net3Plus

DeepLabV3

시각화

matplotlib

기대 효과

사용자가 차량의 이미지를 모델에 넣고,
모델은 스크래치 위치를 마스킹 한다.

↓

마스킹 된 이미지를 통해 해당 차량의
스크래치 유무를 판단할 수 있다.

↓

차량 이미지를 검토하는 인원이 일일이
모든 사진을 분류해야 하는 번거로움이 줄어든다.

입력 이미지

출력 이미지

2

프로젝트 팀
구성 및 역할

역할

이미지 라벨링

데이터 수집

데이터 전처리

모델링

시각화

3

프로젝트 수행 절차 및 방법

수행 절차

구분	기간	활동	비고
이미지 라벨링	7/14(목) ~ 7/15(금)	스크래치 이미지 라벨링	Pixel Annotation Tool 사용
데이터 수집	7/18(월) ~ 7/19(화)	기업 데이터, AI-Hub 데이터 수집	약 16,500개
데이터 전처리	7/20(수) ~ 7/22(금)	이미지 크기 조정, 색상 조정, 광원 제거, 마스킹 이미지 생성 등	
모델링	7/25(월) ~ 7/29(금)	U-Net, DeepLabV3 모델링	학습: 약 13,500개 검증: 약 2,800개
모델 개선	8/1(월) ~ 8/4(목)	U-Net 하이퍼 파라미터 조정, U-Net3Plus 모델 학습	Optimizer, Learning Rate, LR Scheduler

사용한 방법

1) 학습에 사용할 데이터가 부족해서 검증 데이터와 테스트 데이터는 동일하게 사용했다.

2) 손실 함수로는 Dice Loss를 사용했고, DeepLabV3 모델만 Binary Cross Entropy를 사용했다.

3) 학습은 차량의 이미지에서 스크래치가 난 부분을 1, 스크래치 외의 부분을 0으로 예측하도록 이루어진다.

4) 데이터는 스크래치가 있는 부분보다 스크래치가 없는 부분이 많은 즉 1의 class보다 0의 class가 훨씬 많은 불균형 데이터이다.

4

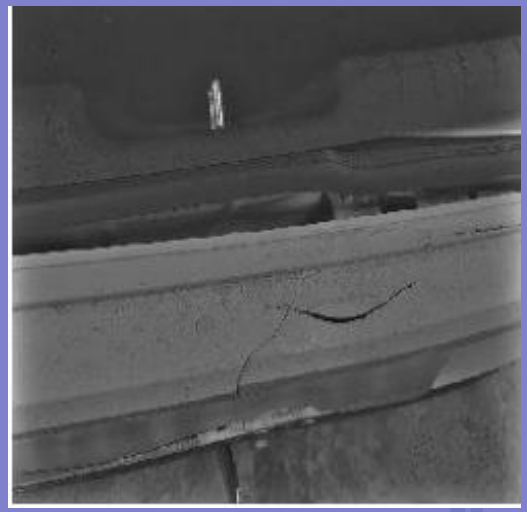
프로젝트 수행 결과

데이터 소개 및 전처리

기업 3,000개 + AI-Hub 13,000개 = 학습: 약 13,500개
검증: 약 2,800개

RGB

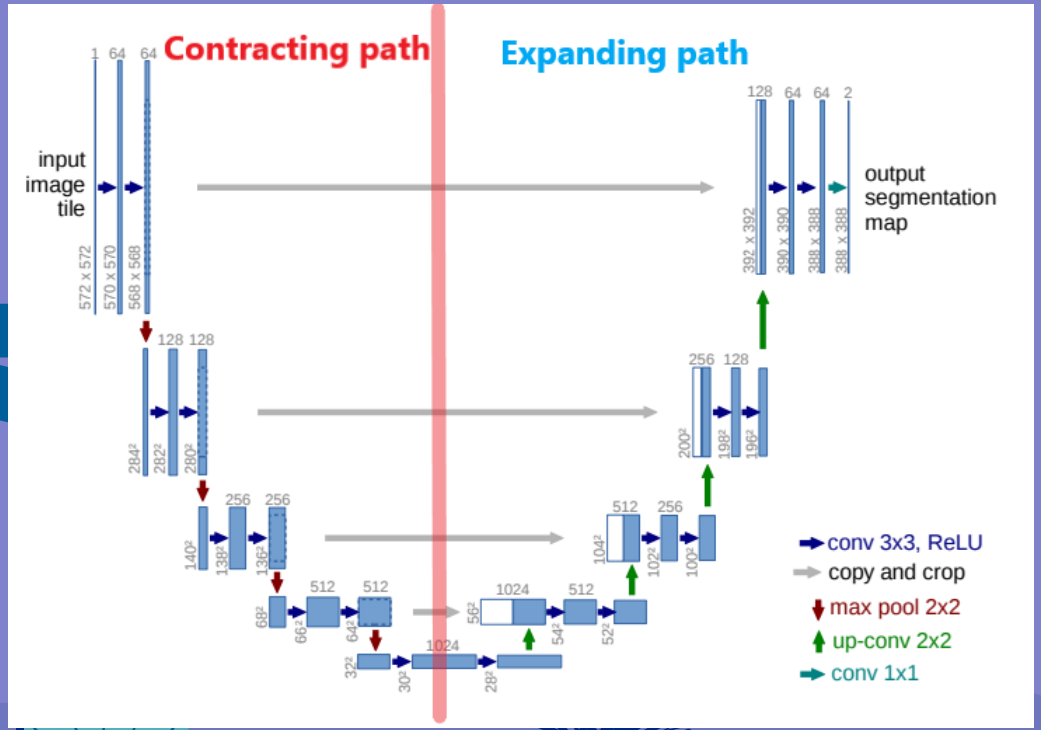
Gray



No light

모델 설명

U-Net: Segmentation 모델 중 하나로 이전 층의 공간 정보를 최대한 보존하면서 이미지를 픽셀 단위로 분류 시키는 모델이다.



Encoding(Down Sampling)

Decoding(Up Sampling)

모델 설명

Encoding(Down Sampling)

1) Input image를 받는다.

2) 3 x 3 합성곱 연산 수행 + ReLU(피처맵의 크기는 줄고, 채널의 수는 늘어난다.)

→ 2)번의 과정을 2회 반복한다.

3) 2 x 2 Max pooling 연산을 수행하여 피처맵의 크기를 절반으로 줄인다.

→ 위 과정을 4번 반복하여 채널의 수는 늘리면서 피처맵의 크기는 줄인다.

Decoding(Up Sampling)

5) Up sampling을 통해 Down sampling과는 반대로 채널의 수는 줄이고, 피처맵의 크기는 키운다.
→ 매칭되는 down sampling의 피처맵의 일부를 up sampling 과정에서 합쳐서 이전의 위치 정보가 손실되지 않게 한다.

6) 위 과정을 4번 반복한다.

7) 최종 결과물로 class 크기 만큼의 채널수를 가진 원본과 비슷한 크기의 피처맵이 만들어 진다.

출력값은 픽셀 단위의 확률값(Ex. 해당 픽셀이 어느 클래스에 속하는지)

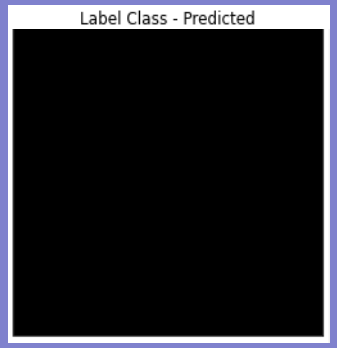
성능

모델(하이퍼 파라미터)		Loss	IoU Score	비고
U-Net	Epochs 150, Batch 4, Lr 0.001, Adam	0.7082	0.1842	기업 데이터만 사용
U-Net	Epochs 100, Batch 4, Lr 0.001, Adam	0.6272	**0.1916**	AI-Hub 데이터 추가
DeepLabV3	Epochs 50, Batch 16, Lr 0.001, Adam	0.5212(BCE)	0.1470	성능 저하
U-Net	Epochs 100, Batch 4, Lr 0.001, Adam	0.6483	0.1808	광원 제거

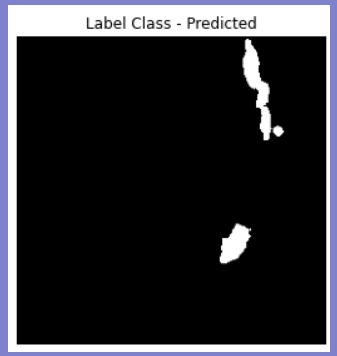
성능

모델(하이퍼 파라미터)		Loss	IoU Score	비고
U-Net	Epochs 100, Batch 4, Lr 0.001, **RMSP**, **CosineAnnealingLR**	0.6094	0.2042	Gray scale
U-Net	Epochs 100, Batch 4, **Lr 0.0001**, Adam, **CosineAnnealingWarm Restarts**	0.6039	**0.2107**	최고 성능
U-Net	Epochs 10, Batch 4, Lr 0.0001, Adam, Lambda LR	0.6858	0.1602	비교 모델
U-Net3Plus	Epochs 10, Batch 4, Lr 0.001, Adam, Lambda LR	0.6542	**0.1831**	비교 모델

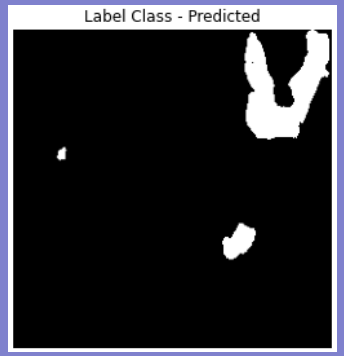
결과 시각화



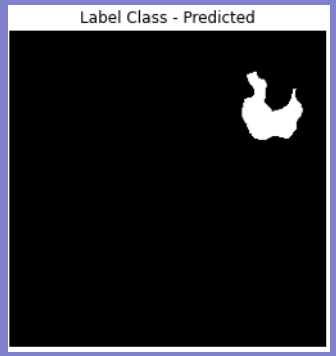
BCE



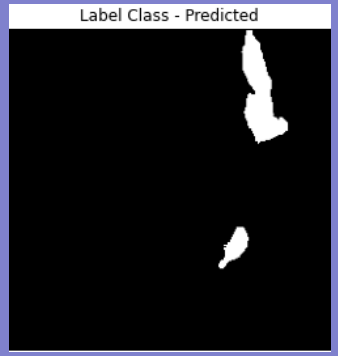
U-Net(초기)



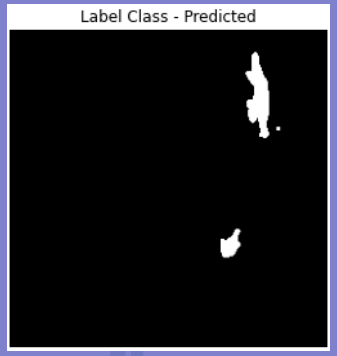
U-Net(광원제거)



DeepLabV3



U-Net(Best)



U-Net3Plus

5

자체 평가 의견

한계 및 해결방안

한계	해결 방안
스트레치 외의 손상에 대해서는 분류하지 못했다.	스크래치 외의 손상 데이터를 제공 받는다면 찍힘, 벌어짐 등과 같은 다른 손상도 다중 분류로 해결할 수 있다.
데이터 양이 적어서 과적합을 해결하기 어려웠다.	더 많은 시간과 좋은 GPU 환경이 주어진다면 AI-Hub에서 제공하는 100GB 이상의 데이터를 사용할 수 있고, 성능이 좋은 U-Net3Plus 모델을 더 깊게 학습시킴으로써 지금보다 더 좋은 모델을 만들 수 있다.
U-Net3Plus 모델을 더 깊게 학습시키지 못했다.	

느낀점

1) 성능을 많이 높이진 못했지만 도입부에 세웠던 IoU Score 목표를 달성해서 뿌듯했다.

2) 기업과 연계하여 진행한 프로젝트 덕분에 현업에 사용되는 데이터를 직접 사용할 수 있어서 좋았다.

3) 모델의 성능을 높이기 위해 진행했던 전처리나, 하이퍼 파라미터 튜닝들이 현업에서 내가 해야 할 일이기 때문에 좋은 경험이 됐다.

Thank You