

[최종보고서] 2023.04.19

Dr.PO

이미지 인식을 통한 AI 수리 견적서

POSCO AI & BIG DATA ACADEMY 21기 C1

권구택 / 김현철 / 박수림 / 임지연 / 윤혜연

목차

1. 추진 배경	3
1.1 문제 상황	3
1.2 논문 사례	6
1.3 기업 사례	10
2. 프로젝트 소개	12
2.1 프로젝트 소개	12
2.2 프로젝트 목표	14
3. 구조도	15
3.1 개발 환경	15
3.2 프로세스	17
4. 적용 기술 및 기능	18
4.1 Sementic Segmentation	18
4.2 Object Detection	20
5. 제작 과정 및 결과	22
5.1 데이터 수집 및 정제	22
5.2 U-Net 모델 구축 – 손상종류	25
5.3 YOLOv5 모델 구축 – 손상부위	27
5.4 수리비 산정 모델 구축	31
5.5 V5 앱 구현	34
6. 결론	39
6.1 한계점 및 개선사항	39
6.2 기대효과 및 활용방안	41
7. 참고문헌	42
8. 상호평가	43

1. 추진 배경

1.1 문제 상황

1) 국내 차량 등록 및 교통사고 건수 증가

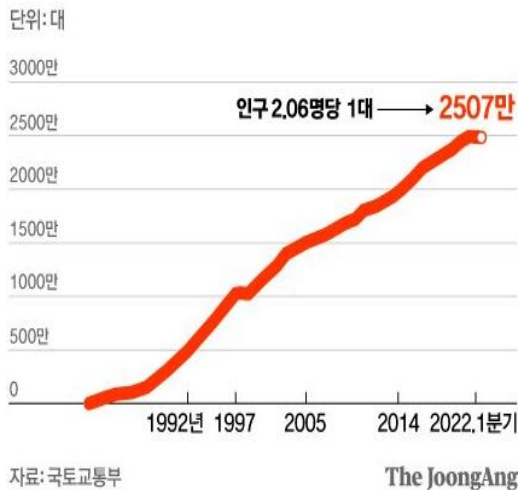
국내 등록 자동차가 처음으로 2500만대를 넘어섰다. 국토교통부 조사에 따르면, 1992년부터 2022년까지 10년간 국내 자동차 등록 현황은 꾸준히 증가하고 있다. 이는 자동차 한 대당 인구수가 2명이라는 것을 보여주고 있다. 교통사고 건수 또한 2021년 기준 125만건으로, 2020년보다 1만건이 증가하였다.

자동차 등록대수 증가는 국내에서만 국한되는 것이 아니다. 한국의 경우 자동차 1대당 인구수가 두 명이지만 미국이나 일본 등 주요 해외 국가의 경우 자동차 1대당 1~1.8명이 소유할 정도로 전 세계 자동차 시장은 커져 있는 상태이다.

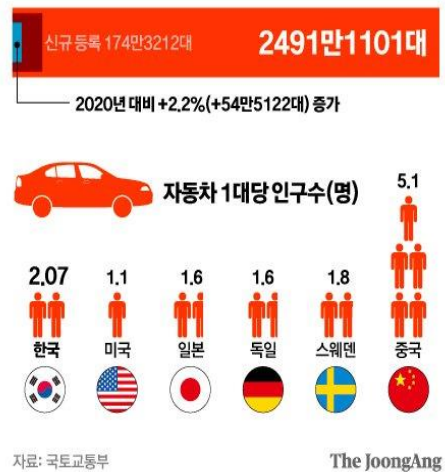
국내이 자동차 등록 수가 증가하고 있는 만큼 해외처럼 자동차 시장이 커질 것을 예상할 수 있다.

따라서, 우리 팀은 시장조사를 통해 자동차 등록하는 사람이 많아질수록 자동차 소비 시장이 지속적으로 증가할 것으로 판단하였고 국내 차량 및 사고 관련 시장에 대한 소비자의 관심이 증가할 것임을 예상할 수 있었다.

자동차등록 현황



2021년말 기준 자동차등록대수

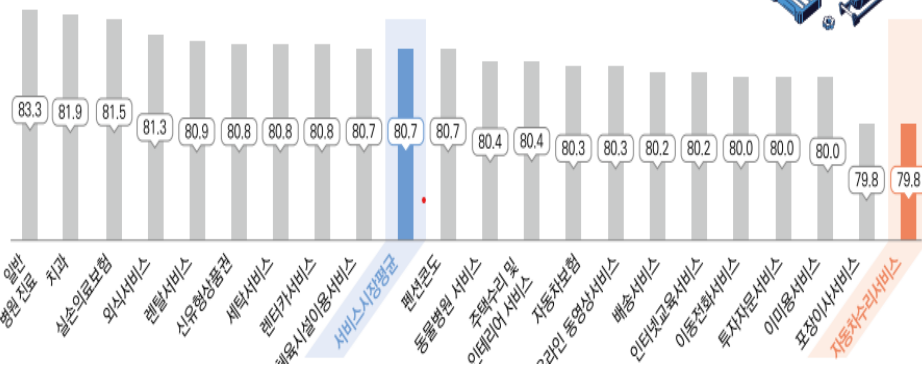


2) 차량 수리 시장, 소비자시장성과지수에서 경고 시장

자동차수리서비스 시장은 21개 서비스 시장 중, 소비자시장성과지수(KCMPI)가 가장 낮아 매년 경고 시장으로 분류되고 있는 실정이다. 차량 관련 시장에 대한 소비자의 관심은 꾸준히 증가하는 반면, '자동차수리서비스'는 2015년부터 2019년까지 연속하여 최하위로 평가된 시장으로 특히 '소비자 불만 및 피해' 경험률(9.1%)로 가장 높았다. 이처럼 차량 수리 업체에 대한 소비자의 신뢰성은 꾸준히 평균에 비해 낮은 수치를 보이고 있다. 이에 따라 소비자의 신뢰성이 지속적으로 최하위 평가를 받는 만큼 시장의 안정성이 위협받는 상황이기에, 차량 수리 시장의 신뢰성 유지 및 재고 방안에 대해 고민할 필요가 있다.

'자동차수리서비스' 시장, 21개 서비스 시장 중 소비자시장성과지수(KCMPI)가 가장 낮아 경고 시장으로 분류

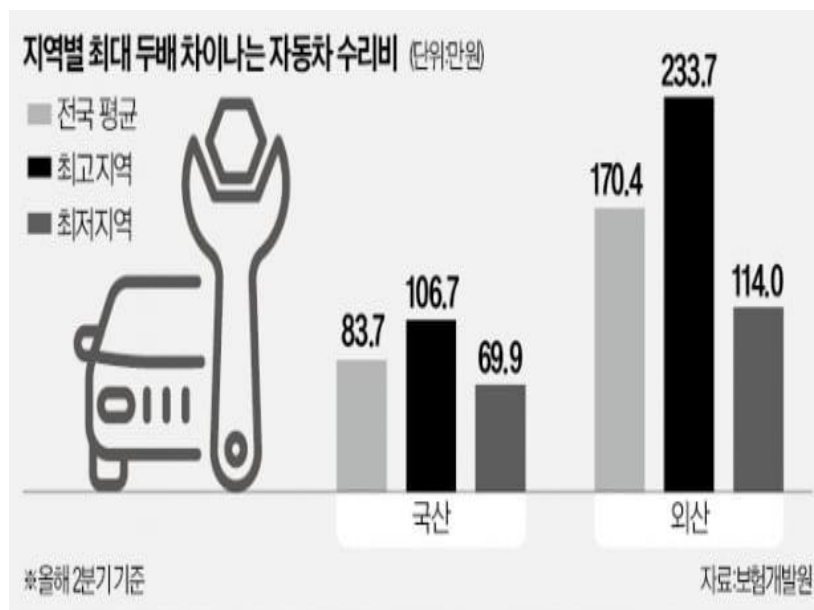
- » 「자동차수리서비스」 시장의 소비자시장성과지수(KCMPI)는 2015년 이후 꾸준히 상승하고 있으나 매년 '경고' 수준
- » 평가항목 중 '선택다양성', '비용효율성'을 제외한 모든 항목이 소비자지향성 경고 영역



구분	2015년	2017년	2019년
자동차수리서비스 소비자시장성과지수(KCMPI)	● 73.9	● 75.9	● 76.1
순위	29위/29개 시장	27위/27개 시장	31위/31개 시장
선택다양성	● 70.2	● 73.3	● 72.8
비용효율성	● 71.6	● 70.9	● 70.1
신뢰성	● 70.1	● 70.8	● 70.4
기대만족도	● 72.1	● 71.3	● 72.0
소비자불만 및 피해(피해 경험률)	● 91.6(8.4%)	● 97.5(3.0%)	● 96.8(9.1%)

3) 표준화된 수리비 산정 시스템의 필요성

보통 차량 수리 업체에 따라 수리비 청구 견적이 상이하다. 하지만 그 중에서도 일부 업체의 과잉 수리로 인해 소비자들이 피해를 입고 있으며, 이와 같은 이유로 소비자의 신뢰성이 꾸준히 낮아지고 있다. 보험 개발원에 따르면 국산차의 평균 수리비는 최근 2년간 10% 증가하였으며, 수입차 수리비는 지역에 따라 최대 2배 넘게 차이 날 만큼 천차만별인 상황이다. 따라서 표준화된 수리비 산정 시스템을 구축함으로써, 차량 수리 업체에 대한 신뢰성을 재고할 수 있다.



최근에는 렌터카, 보험, 차량 관리 등, 자동차와 관련된 시장 규모가 점점 더 증가함에 따라 관련 기술의 연구가 활발해지고 있다. 그 중에서도, 보다 정확하고 객관적인 시각으로 차량 파손 현황을 인식하는 기술의 필요성이 증대되고 있다. 따라서 이러한 사회적 문제에 초점을 맞춰, 차량 손상 현황을 인식하고, 나아가 표준화된 수리비 산정을 주제로 프로젝트를 진행하게 되었다.

1.2 논문 사례

차량 파손 알고리즘의 선행 연구 사례를 조사한 결과는 다음과 같으며, 사용 기술에 따라 VGGNet, WeProov.AI, YOLOv4의 경우로 나눌 수 있다.

1) VGGNet을 이용한 기법¹

- 기술 개요

입력 이미지에 대해 사전 훈련된 VGG 모델을 통해 자동차의 존재 유무를 판단한다. 차량이 존재한다고 판단하면, 차량이 존재하는 것으로 감지된 범위 내에서 파손 여부를 판단한다. 파손이 있을 시 파손 부위와 정도를 판단하는데, 이 때 부위는 전면, 후면, 옆면으로 나뉘고 정도는 세 가지로 나뉜다.

- 사용 기술 및 결과

VGG16과 VGG19의 성능을 비교하기 위해 정밀도, 재현율, F1-score를 측정 기준으로 선정하였다. VGG16은 손상감지, 손상 위치, 손상 심각도에서 각각 94%, 71%, 61%의 정밀도를 가지며, 결과적으로 VGG19보다 더 좋은 성능을 보인다.

TABLE I
PERFORMANCE ANALYSIS OF CAR DAMAGE ASSESSMENT

Pre-trained VGG models	Performances of damage detection			Performances of damage location			Performance of damage severity		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
VGG16	0.94	0.94	0.94	0.71	0.69	0.69	0.61	0.55	0.53
VGG19	0.91	0.91	0.91	0.69	0.66	0.66	0.59	0.54	0.51

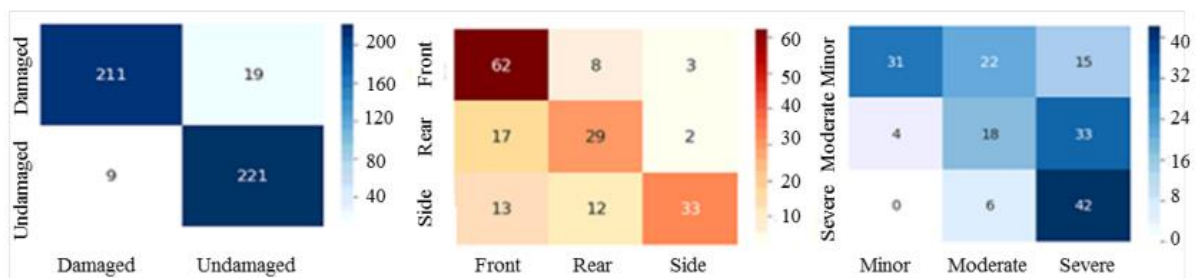


Fig. 2. Confusion matrices for car damage assessment of VGG16

- 연구의 한계점

VGGNet을 활용한 본 연구에서는, 우선 차량 파손 부위가 전면, 후면, 옆면 세 가지 부분에 국

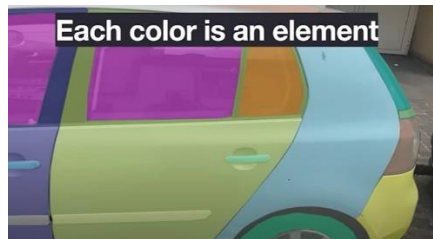
¹ P. M. Kyu and K. Woraratpanya, "Car damage assessment based on VGG models," JSCI8, 2021.

한되어 있다는 한계점이 있다. 또한 각 부위에 존재하는 손상 종류나 개수는 파악하지 못한다.

2) WeProov.AI팀의 독자적인 알고리즘²

- 기술 개요

VGGNet을 이용한 알고리즘과 마찬가지로, 손상된 차량의 이미지를 바탕으로 학습을 진행하였다. 차이점은 차량을 44개의 부분으로 나누어 구분했다는 것인데, 관련 연구 중 차량 부위를 가장 세분화한 모델에 속한다. 손상의 종류 또한 유리 깨짐, 스크래치, 덴트 등 총 18개의 종류로 분류하였다.



- 사용 기술 및 결과

24개의 모듈을 통합한 자체적인 알고리즘을 제작하였고, 이를 이용하여 파손부위와 종류를 검출한다. 인식률이 매우 우수하며 디테일한 판정 결과를 출력한다는 장점이 있다.



2021년 3월 기준, 손상 사이즈에 대한 검출 정확도는 0.5 ~ 1cm의 경우 90%, 1~5cm의 경우 95%, 5cm 이상의 경우 98%를 보인다.

² WeProov.AI, "AI for vehicle inspection, damages & claims management," 2020.

³ AI for vehicle inspection, damages & claims management - Car damage detection with WeProov.AI

Damage size	Today Average accuracy	Human Average Accuracy	March 2021 Accuracy
0,5 cm to 1cm	74,2 %	80 %	90 %
1 cm to 5 cm	83,5 %	80 %	95 %
More than 5 cm	88,1 %	90 %	98 %

- 연구의 한계점

학습 데이터 셋 양이 백만 장 이상 요구되며, 검출에 소요되는 시간이 평균 2분 정도로 매우 느리기 때문에 비용적인 면에서 다소 비효율적일 수 있다.

3) YOLOv4를 이용한 개선 방안 연구⁴

- 기술 개요

VGGNet 이용 사례, WeProo.AI팀의 사례와 동일하게 차량 파손 이미지를 통한 학습 및 테스트를 거친다. 우선 YOLOv4를 이용하여 차량의 손상과 해당 부위를 검출한다. 이후 각각 검출된 바운딩 박스의 중심좌표, 너비, 높이를 이용한 알고리즘을 적용하여 파손 현황을 도출한다.

- 사용 기술 및 결과

이전 연구들의 파손 클래스와 유사하지만, 차이점은 진흙이 묻어있는 부위를 추가로 검출한 후 전체 파손 사례에서 제외시킨다는 것이다. 또한 파손이 중복 검출되는 문제를 해결하기 위해 중복검출 방지 알고리즘을 YOLOv4 네트워크에 적용했다.

결과는 기존 연구 사례와 비교하였고, 파손 검출 성능, 부위 검출 성능, 알고리즘을 통과한 최종 부위 별 파손 검출 성능의 정밀도, 재현율, F1-score를 평가 지표로 사용하였다

표 4. VGGNet 이용모델과 제안 알고리즘의 성능비교

	Damage Detection			Part Detection			Final Detection		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
Proposed Model	0.94	0.78	0.85	0.95	0.90	0.92	0.93	0.70	0.79
VGGNet Model	0.94	0.94	0.94	0.71	0.69	0.69	-	-	-

⁴ 전종원, 이효섭 and 한희일. (2021). YOLOv4를 이용한 차량파손 검출 모델 개선. 전기전자학회 논문지, 25(4), 750-755

손상 검출에 있어서는 기존 VGGNet 이용 모델에 비해 성능이 떨어지지만, 부위검출성능 측면에서는 더 뛰어난 것을 확인할 수 있다. 또한 손상 검출 부위가 더 다양하고, 그 종류와 개수까지 도출할 수 있다는 면에서 구체적이라는 장점이 있다.



- 연구의 한계점

차량 사진에서 빛의 반사나 블러링이 있을 경우 결과에 큰 영향을 준다. 또한 바운딩 박스의 사각지대로 인해 기존 기술들보다 평균적인 성능이 낮다는 한계가 있다.

각 선행 연구의 이러한 한계점을 보완하여, 손상의 분류를 구체화하는 동시에, 검출에 걸리는 시간을 단축하고 정확도를 높이는 쪽으로 차별화 방향성을 설정했다.

1.3 기업 사례

차량 파손 견적을 예측하는 국내 기업으로는 '카닥', 'AOS 알파'가 있다.

 <p>< 카닥 ></p>	 <p>< AOS 알파 ></p>
<ul style="list-style-type: none"> - 사진을 찍어 어플에 올리면 제휴 업체들을 연결해주는 시스템 - 손상 이미지 인식 - 손상 부위 인식 - 최저가 업체 연결 	<ul style="list-style-type: none"> - 사진을 찍어 어플에 올리면 보험 업체들을 연결해주는 시스템 - 손상 이미지 인식 - 손상 부위 인식 - 보험 업체와 정비소 연결

두 기업은 모두 사진을 찍어 어플에 올리면, 보험이나 수리 등 제휴 업체와 연결해주는 시스템을 구축하였다. 그러나 가격이 카닥의 경우는 가격이 표준화되어 있지 않고, AOS 알파의 경우는 소비자의 상의 없이 업체가 일방적으로 제시한 가격 기준을 따라야 한다는 한계점이 있다.

 <p>< 중국 평안보험 ></p>	 <p>< 영국 트랙터블 ></p>
<ul style="list-style-type: none"> - 사진을 찍어 어플에 올리면 컴퓨터가 견적서를 제시하는 시스템 - 손상 이미지 인식 - 주위 환경 이미지 인식 - 컴퓨터를 통한 견적서 계산 	<ul style="list-style-type: none"> - 사진을 찍어 어플에 올리면 딥러닝을 이용해 수리비를 산출해주는 시스템 - 손상 이미지 인식 - 손상 형태 파악 - 딥러닝을 통한 수리비 산출

이에 반해, 해외 기업들은 이미지 인식을 통해 AI가 가격을 표준화한 시스템을 구축한 것을 확인할 수 있다. 따라서 본 프로젝트에서는 손상 부위와 형태로 AI가 가격을 결정하게 함으로써, 객관성을 높여 국내에서의 차별점을 만들기로 하였다.

2. 프로젝트 소개

2.1 프로젝트 소개


- Dr.Po란?

본 프로젝트는 차를 수리하기 위해 시작된 AI 활용 사례이다. '수리하다'라는 표현을 사람에게 적용한 경우 '의사'로 표현할 수 있으므로 '닥터'라는 명칭을 붙였고, 포스코의 '포'를 합쳐서 Dr. Po라는 이름을 사용하게 되었다.

딥러닝 기술인 Object Detection과 Semantic Segmentation을 통해 이미지를 학습하고, 이를 바탕으로 건적 데이터를 산출한다. 앱 화면을 통해 이를 사용자에게 보여주고, 소비자의 현 위치에 가까운 순으로 정비 업체를 추천해주는 프로젝트이다.

- 프로젝트 소개

지난 수십년간 이뤄진 인공지능의 발전 속도가 최근 들어 급격히 빨라지고 있다. IT 시장은 '지속 가능성'을 모든 전략 기술 트렌드를 통과하는 주제로 주목하고 있다. 이러한 '지속 가능한' IT 시장에 맞춰 우리는 사용자 중심 관점에서 사용자에게 지속적인 서비스를 제공할 수 있도록 만들어야 한다.

 환경(E)이슈	 사회(S)이슈	 지배구조(G)이슈
<ul style="list-style-type: none"> - 기후변화 및 탄소배출 - 대기 및 수질오염 - 생물의 다양성 - 삼림 벌채 - 에너지 효율 - 폐기물 관리 - 물 부족 	<ul style="list-style-type: none"> - 고객만족 - 데이터보호 및 프라이버시 - 성별 및 다양성 - 직원참여 - 지역사회 관계 - 인권 - 노동기준 	<ul style="list-style-type: none"> - 이사회 구성 - 감사위원회 구조 - 노무 및 부패 - 임원보상 - 로비 - 정치 기부금 - 내부 고발자 제도

<자료:금융투자협회, 서울신문>

지속 가능한 서비스를 제공하기 위해서는 소비자의 관점에서 소비자의 니즈에 맞춰야 한다고

판단하였다. 시장의 흐름에 맞게 소비의 니즈가 늘어나고 있는 지속가능한 시장을 파악하기로 하였다. 현재 시장에서 자동차 대수가 증가하고 있는 추세이며 정보 비대칭이 심각한 자동차 애프터마켓을 타겟으로 선정하게 되었다. 우리의 프로젝트는 운전자와 정비업체 양쪽의 편의성을 증대하고자 디지털 기술을 도입하고자 한다. 디지털 기술이 집중하고자 하는 부분은 자동차 외관 수리를 AI가 판단하는 것이다. 자동차 외관 수리 견적은 AI 기술을 활용하여 자동차 파손 이미지를 인식하고 이를 통해 실시간으로 바뀌는 부품비와 정비업체 공임비를 계산하여 사용자에게 제시하고자 한다. 우리 모델의 궁극적인 목적은 이미지 인식만으로 견적서를 계산하여 사용자의 편의를 제공하는 데 있다.

'Dr. Po' 는 사용자가 사용하기 편리하게 어플을 통해 구현하였다. 어플에서 카메라를 통해 자동차 파손 이미지를 인식한 후, 자동차 파손 형태와 부위를 파악하고 사용자 거리 기반 추천 시스템을 탑재함으로써 궁극적인 목적을 달성할 수 있었다.

- 기획 의도

현재 국내에서 자동차 애프터마켓 시장을 선두하는 '카닥'을 벤치마킹하여 프로젝트를 진행하였다.

카닥의 경우 파손 이미지를 인식하여 파손 형태와 파손 부위를 정비업체에 전달해주고 정비업체와 소비자를 연결해주는 O2O 서비스를 진행하고 있다.

카닥의 경우 1) 이미지를 인식하여 파손 형태/부위를 판단한다는 점 2) 정비소 업체와 소비자를 연결해주는 O2O 서비스라는 점 3) 어플을 사용하기 편한 UI/UX 환경을 구성했다는 점 4) 비교 견적을 제시한다는 점에서 우리 팀이 진행하려는 Dr. Po 와 비슷하다고 판단하였고 이를 특징들을 기반으로 모델과 어플을 구현하기로 하였다.

다만, 1) AI 견적서를 제시한다는 점과 2) 소비자의 위치를 기반으로 하여, 거리순으로 정비 업체를 추천해준다는 점을 큰 차별점으로 두고 프로젝트를 기획하였다.

2.2 프로젝트 목표

본 프로젝트의 목표는 다음 다섯 가지와 같다.

1. 파손 부위 이미지를 Object Detection을 통해 학습하는 모델 개발
2. 파손 형태 이미지를 Semantic Segmentation을 통해 학습하는 모델 개발
3. 학습된 데이터를 이용하여 견적서를 추출하는 모델 개발
4. 어플을 적극적으로 활용할 수 있는 서비스를 구현
5. 포스코 AI 빅데이터라는 목적성에 맞게 다량의 데이터셋을 AI를 통해 활용

3. 구조도

3.1 개발 환경

1) PC 환경

- Python

플랫폼에 독립적이며 인터프리터식, 객체지향적, 동적 타이핑 대화형 언어이다. 파이썬의 표준 라이브러리는 프로그래머가 바로 사용할 수 있는 라이브러리 통합환경이다. 본 프로젝트에서 CSV 파일과 데이터 베이스 접속하기 용이하여 사용하게 되었다.

- PyTorch

파이썬 기반의 딥러닝 프레임워크 중 하나로, 동적 계산 그래프로 연산을 처리하여 모델 개발 및 실험을 더욱 쉽게 할 수 있어 사용하게 되었다.

- OpenCV

오픈 소스 컴퓨터 비전 라이브러리로, 이미지 및 비디오 처리를 위한 다양한 함수와 알고리즘을 제공한다. 이미지 및 비디오 처리 함수를 제공하여, 이미지의 전처리, 후처리 및 분석을 수행 가능하게 하므로 사용하였다.

- Semantic Segmentation

이미지 분할 기술 중 하나로, 이미지를 픽셀 단위로 분할하여 각 픽셀이 어떤 클래스에 속하는지 판별하는 기술이다. 파손 형태를 픽셀 단위로 분할하여, 파손 형태에 따라 각각의 클래스로 분류하기 위해 사용하였다.

- Fire Base

구글에서 제공하는 모바일 및 웹 앱 개발 플랫폼으로, 데이터베이스, 스토리지, 인증, 호스팅, 푸시 알림 등의 다양한 기능을 제공한다. 데이터의 변경사항이 발생하면 즉시 반영되어 사용자에게 실시간으로 보여지게 하기 위해 사용하였다.

- Ubuntu 18.04

Ubuntu 는 안정성이 높은 운영체제 중 하나로 GUI가 사용하기 쉬운 인터페이스로 개선되어 있다.

또, 설치 및 업그레이드가 쉽게 이루어지기 때문에 사용자 편의성이 좋다. 특히, 다양한 오픈 소스 소프트웨어를 지원하여 다양한 프로그래밍 언어와 개발 도구를 사용할 수 있기에 활용하였다.

- Linux

안정적이며 개인화가 가능하며, 다양한 프로그래밍 언어와 개발도구를 지원하는 운영체제로써 많은 사용자들이 선택하고 있기에 사용하였다.

2) APP 환경

- Kotlin

Java와 비교하였을 때 간결하고 생산적인 언어로 같은 기능을 구현한다고 하였을 때 비교적 적은 양의 코드로 구현할 수 있다. 또한, 자바보다 훨씬 간결하고 문법적으로 직관적이다. 따라서 Kotlin을 사용하면 빠른 개발과 디버깅이 가능하기 때문에 사용하였다.

- Figma

UI/UX 디자인 툴로서, 디자인과 프로토 타입 작업을 할 수 있는 툴이기 때문에 안드로이드 스튜디오에서 사용될 디자인을 만들 때 프로토 타입을 함께 만들 수 있다. 따라서, 디자인과 개발 작업이 통합되어 진행될 수 있다고 판단하여 사용하였다.

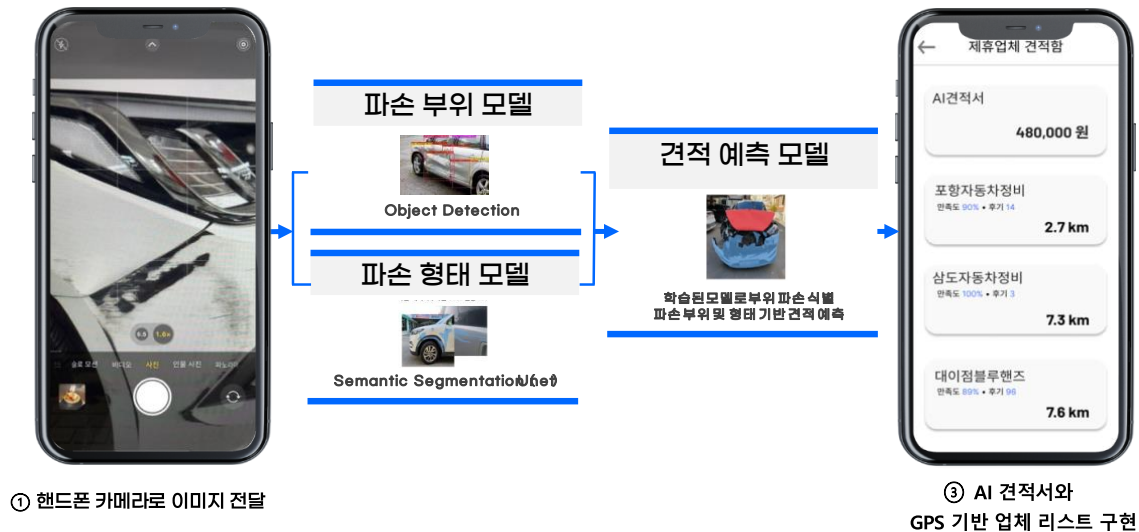
- Android Studio

안드로이드 어플리케이션 개발을 위한 통합 개발 환경을 제공한다. 특히, 코드 자동완성, 디버깅, 테스트 등의 기능들을 제공하여 개발 생산성을 높일 수 있기 때문에 사용하였다.

- Google API

구글에서 제공하는 다양한 서비스들을 활용하기 위한 인터페이스를 제공하여 구글의 다양한 서비스들을 쉽게 활용하기 위해 사용하였다. 데이터 사용 측면에서, 구글 API를 사용하면 구글이 보유하고 있는 방대한 데이터를 활용할 수 있다. 구글 맵 API를 사용하여 애플리케이션에 지도 서비스를 제공하기 위해 사용하였다.

3.2 프로세스



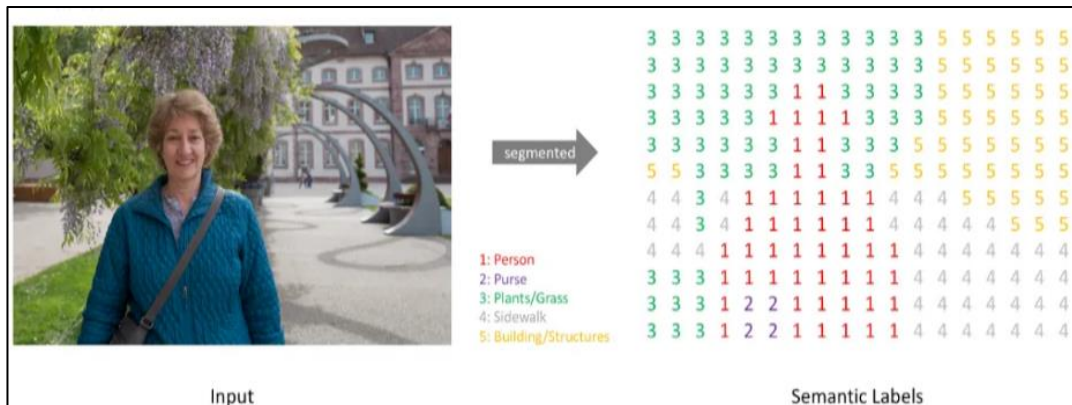
사용자가 어플에 자동차 파손 이미지를 업로드하면 파손 형태와 부위를 파악하여 견적을 예측한다.

파손 부위 모델은 Object Detection에서 Bounding Box를 사용해 만들었다. 파손 이미지를 통해 a 필러, b 필러, c 필러, 뒷도어, 뒷범퍼, 뒷유리 등 총 32개의 파손 부위 중 하나를 예측할 수 있다. 파손 형태 모델은 Semantic Segmentation을 이용해 만들었다. 이를 통해 파손 이미지 형태가 스크래치, 이격, 파손, 찌그러짐 중 하나인 것을 예측할 수 있다. 예측한 값을 이용하여 예상 견적을 생성하고 Fire Base로 데이터를 어플에 전송한다. 어플에서는 AI가 예상해준 견적서를 사용자에게 보여준다. 여기에 Google API로 GPS를 활용하여 사용자의 위치를 파악하여 사용자 최단 거리 기반으로 업체를 추천함으로써 정비소와 사용자를 연결해주는 O2O 서비스를 구현할 수 있다.

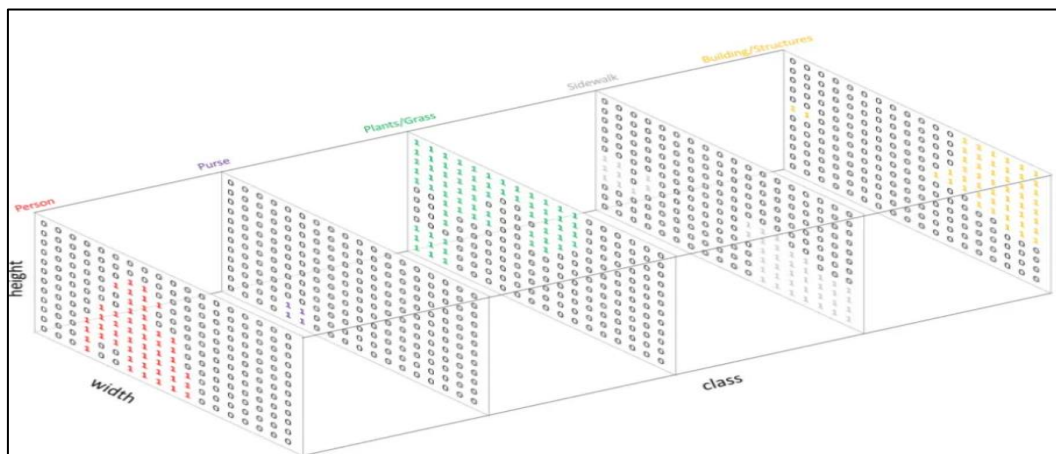
4. 적용 기술 및 기능

4.1 Sementic Segmentation

입력된 이미지를 픽셀별(pixel-wise)로 분류하는 문제이다. 최근 컴퓨터 비전에서 많이 다뤄지는 문제이며, input으로 이미지가 사용되고 output으로 픽셀 당 클래스를 반환한다는 특징이 있다.



▲ Sementic Segmentation의 결과



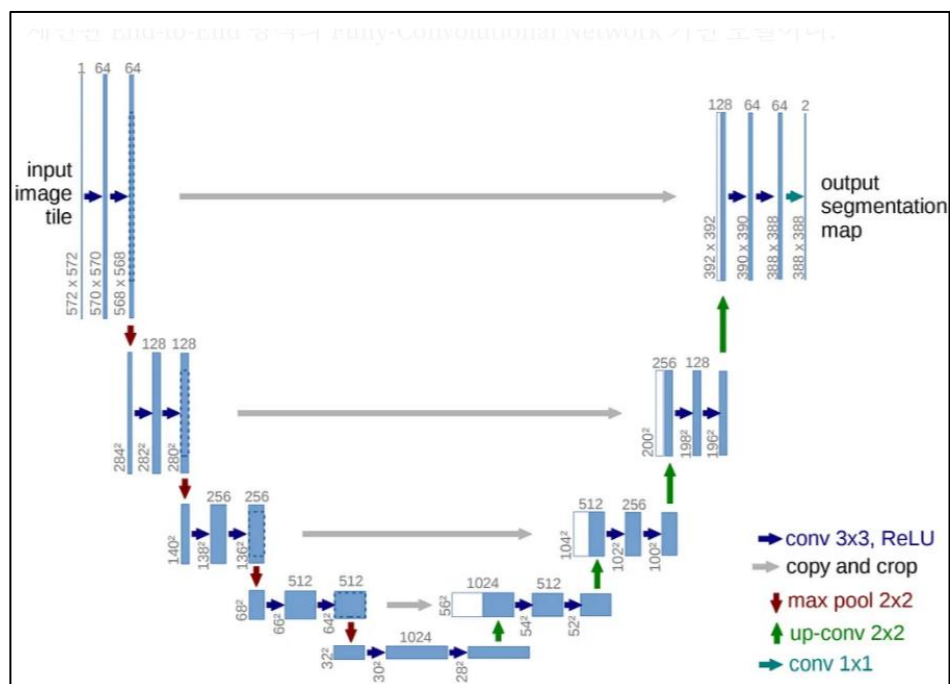
▲ Output 구성

Sementic Segmentation 모델에 이미지가 입력되면 One-Hot Encoding으로 각 class에 대한 출력 채널을 만들어 segmentation을 구성한다. 이후 Class의 개수만큼 만들어진 채널을 argmax를 통해 하나의 출력물로 만들어낸다. 이는 input에 대해서 하나의 레이블을 예측하는 Classification(분류)나 예측할 위치에 Bounding Box를 그려 Box 속 물체를 예측하는 Localization/Detection와 차이가 있다. 이번 프로젝트에서는 차량의 파손 종류, 파손 부위를 탐지하고, 픽셀 영역을 산출해 내어 건적 산정에 용이하도록 Sementic Segmentation 기술을 활용할 예정이다.

구축할 모델을 선정하기 위해 자료조사를 진행하였다. 2019년 SOCAR에서 개발한 딥러닝 기반 차량 파손 탐지 모델에 따르면, 고려 대상인 U-Net Model과 DeepLab V3 Model 중 최적 모델을 선정하기 위해 두 모델에 대해 데이터 전처리, 하이퍼파라미터 튜닝 등을 배제한 순수 베이스라인을 구현하였다. 그 결과 mIoU가 U-Net에서 92.2, DeepLab V3는 80.7의 성능을 나타내어 개발 과정에서 U-Net Model을 선정하였다. 우리 팀은 수집한 데이터의 출처가 본디 SOCAR에서 생성한 데이터이며, 차량의 파손 종류와 파손 부위를 탐지한다는 프로젝트 목적이 같기에 이를 벤치마킹하기로 결정하였다. 따라서 U-Net Model을 구축하여 분석에 활용하였다.

- U-Net

U-Net은 의학분야에서 Image Segmentation을 목적으로 제안된 Fully-Convolutional Network 기반 모델이다. 이미지의 컨텍스트 정보를 얻기 위한 Contracting Path와 정확한 Localization을 위한 Expanding Path가 대칭으로 구성되어 있다. (Encoder – Decoder 구조)



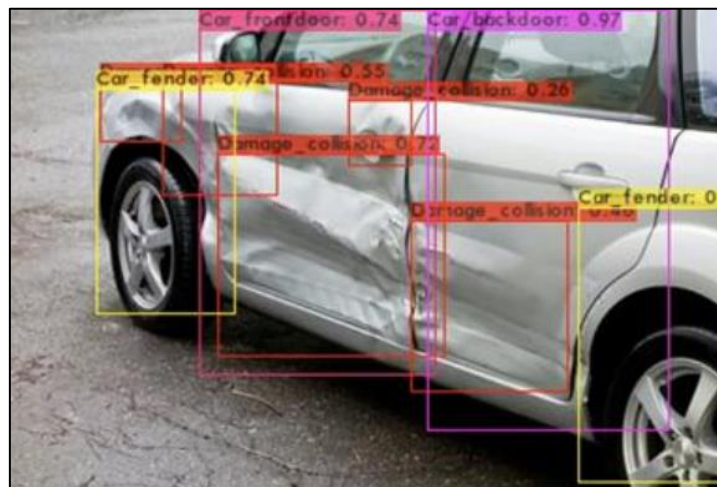
▲ U-Net의 구성

Contracting Path에서는 3X3필터와 Maxpooling을 통해 Down-sampling을 진행하며 Feature-map의 size를 절반씩 줄여나간다. Expanding Path에서는 Contracting Path와 반대의 연산을 진행하며 Up-sampling을 진행한다. 이후 마지막 레이어에는 1X1 convolution 연산을 통해 픽셀당 클래스를 얻을 수 있다. U-Net은 학습 속도가 빠르다는 장점과 함께 Contracting Path 부분에 잘 학

습된 VGGNet, EfficientNet 등의 모델 구조를 encoder로 지정할 수 있는 장점이 있다.

4.2 Object Detection

컴퓨터 비전과 이미지 처리와 관련된 컴퓨터 기술로서, 이미지 및 비디오 내에서 유의미한 특징 객체를 감지하는 작업을 수행한다. 얼굴 인식(Face Detection), 비디오 추적(Video Tracking), People Counting 등 다양한 분야의 문제를 해결하기 위해 사용된다.

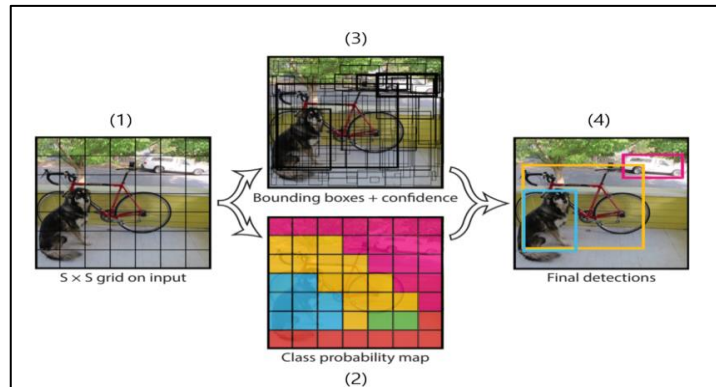


▲ Object Detection Class 예측 예시

Object Detection은 Classification과 Box를 통한 localization을 동시에 수행하며 물체라 판단되는 부분에 Bounding Box를 그려 그 부분이 물체인지 아닌지 판별한다. 또한 일반적인 Classification과 달리 다중 클래스 예측이 가능하다. 본 프로젝트에서는 이미지 속 BBOX가 그려진 부분을 추출하여 차량 부위를 예측하는 모델을 구축할 예정이다.

- **YOLOv5**

You Only Look Once의 약자로 처음 One-stage-detection 방법을 고안해 실시간으로 객체 탐지가 가능한 모델이다. 현재 YOLOv8까지 등장했으며 특징으로는 첫째, 이미지 전체를 한 번만 보며 통합된 모델을 사용해 간단하다. 마지막으로 기존 모델보다 빠른 성능으로 실시간 객체 검출이 가능하다. 아래는 YOLO의 작동과정을 그림으로 나타낸 것이다.



▲ YOLO의 작동 과정

YOLO에 이미지가 입력되면 $S \times S$ 의 그리드를 생성하여 각 객체별로 Ground Truth에 속할 확률 맵을 생성한다. 이후 Bounding box와의 신뢰도를 계산하여 최종적으로 객체 탐지를 수행한다.

YOLO의 여러 버전들 비교분석 결과 YOLOv5버전을 분석에 활용하였다. 선정 이유로는 우선 구현이 타 YOLO모델보다 쉽고, APP에서 사진을 찍거나 업로드되면 빠른 속도로 차량부위를 예측해야 하기 때문이다. 아래는 YOLOv3부터 YOLOv5까지의 특징이 기술되어 있는 표이다.

YOLOv3	<ul style="list-style-type: none"> - 백본 아키텍처 Darknet 53 기반 - multi-scale에서 feature를 추출 - class 예측 시에 개별 클래스 별로 Logistic Regression 사용
YOLOv4	<ul style="list-style-type: none"> - v3보다 vAP, FPS가 각각 10%, 12% 증가 - v3보다 WRC, CSP 등을 사용해 성능 향상 - CSPNet 기반의 backbone을 설계하여 사용
YOLOv5	<ul style="list-style-type: none"> - v4에 비해 낮은 용량과 빠른 속도, 성능은 비슷함 - YOLOv4와 같은 CSPNet 기반의 backbone을 설계 - Pytorch 기반 구현이기 때문에 이전 버전들과 다름

▲ YOLO의 버전별 특징

5. 제작 과정 및 결과

5.1 데이터 수집 및 정제

1) AI HUB 차량 파손 이미지 데이터

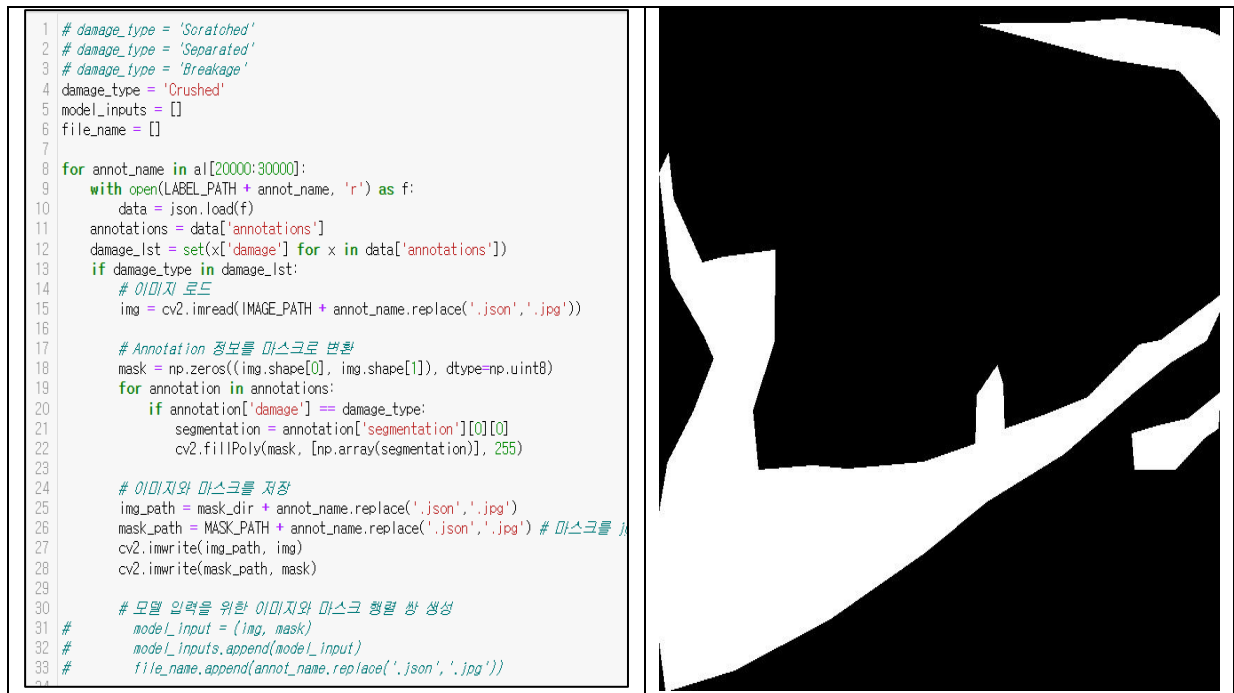
</

▲ AI HUB 사이트 캡처 및 Ground Truth 분포

사고 자동차 이미지를 수집한 AI 데이터셋으로, 차량 외관 부품의 Segmentation과 손상 유형 및 범위를 이해할 수 있는 형태로 가공된 다양한 차종 및 부품에 대한 학습데이터이다.

예측해야 할 Ground Truth는 총 4개로 각각 Scratch(스크래치), Crushed(찌그러짐), Breakage(파손), Separated(이격)으로 구성되어 있으며, 클래스 간 불균형이 존재한다.

각 Ground Truth의 설명을 덧붙이자면, 스크래치의 경우, 차량의 표피 성분의 변화 없이 일부의 긁힘 상태를 나타낸다. 찌그러짐은 차량 외관의 절제상 결합이 발생한 수준으로 판단하며 파손은 외부 충격 등으로 인해 차량의 형태 일부, 혹은 전체가 기존의 형태와 완전히 달라진 경우로 정의한다. 마지막으로 이격은 차량의 각 부위가 외부 충격으로 인해 어긋남이 발생한 경우로 정의한다.



▲ 마스크 생성 코드와 생성된 마스크 이미지

원본데이터에는 이미지 데이터와 json 파일이 함께 저장되어 있다. Segmentation 문제의 경우 Ground Truth의 바운딩 박스, 즉 세그멘테이션을 예측하는 문제이기 json 파일의 segmentation 좌표를 추출해내어 Ground Truth를 생성하는 마스킹을 진행하였다.

마스킹을 진행할 때, 손상 종류인 Scratched(스크래치), Separated(이격), Breakage(파손), Crushed(찌그러짐)에 나누어 마스크 이미지를 저장하였고, 추후 데이터셋 클래스 정의 시 마스크 이미지를 흑백(2차원)으로 로드하여 백색이면 1, 이외는 0으로 이진화를 수행하였다.

각 이미지의 크기는 800 X 600 이상으로 구성되어 있으며, 사이즈가 통일돼 있지 않다는 문제가 있다. 이러한 문제를 마스킹은 원본 이미지 사이즈에 맞게 진행하고, 데이터 로드 시에 이미지 사이즈를 맞춤으로써 해결할 수 있었다.

이미지 데이터를 활용할 모델은 총 2개로, 손상종류와 손상부위를 분류한다. 손상부위는 앞범퍼, 뒷범퍼 등 총 32가지의 라벨로 구성되어 있으며 자세한 사항은 AI HUB 사이트에 데이터 구축 가이드라인 내지 데이터 설명서에서 확인할 수 있다.

2) 차량 부위 이미지 데이터

AI HUB의 차량파손 이미지 데이터는 손상 종류 이외에도 손상 부위가 라벨링 되어 있는 이미지 데이터가 존재한다. 예측해야 할 클래스는 '앞범퍼', '뒷범퍼', '앞도어(좌)', '앞도어(우)' 등 크게

차량부위의 앞면, 뒷면, 옆면, 타이어/휠에 속하는 부위들로, 총 32 가지가 존재한다. 자세한 사항은 사이트에서 확인할 수 있다.

데이터 종류		Annotation 규모	결과물 규모
사고자동차 이미지	손상 종류	2,013,758 건	504,450 장
	차량 부위	179,357 건	128,244 장

▲ 차량 부위 이미지 데이터셋 규모

Object Detection 을 위해 라벨 데이터를 txt 확장자로 변환하는 작업을 수행하였다. 우선 json 파일의 annotation 을 모두 불러와 데이터프레임으로 저장하였다. 이후 예측할 part(차량 부위) 가 존재하는 annotation 을 제외한 다른 annotation 들은 모델 학습에 도움이 되지 않을뿐더러, 모델의 성능을 저하시킬 수 있기 때문에 모두 분석에서 제외하였다. 남은 annotation 의 bbox 좌표값을 이용하여 Bounding Box 를 도식화할 수 있었다.

마지막으로 도출한 part 데이터셋에 라벨을 붙여 Ground truth 를 만든 후 8:1:1 비율로 훈련, 검증, 테스트 데이터셋으로 나누어 txt 파일로 저장하였다.

3) 건적 예측용 데이터셋

건적 예측을 위해 AI HUB 에서 사고 건적서 파일(json)과 이미지 데이터의 json 파일을 함께 가공하였다. 건적서 파일에는 크게 차량정보, 수리비 정산정보, 수리내역으로 구성되어 있으며 json 파일에는 이미지파일명, 사고 ID, segmentation 좌표, 차량연식, 색상, 수리방법 등 다중 딕셔너리로 구성되었다. 건적서 파일은 총 125,006 개이며 라벨 데이터는 train 의 라벨 데이터만을 사용하였다.

	차량정보	수리비 정산정보	수리내역	
0	{제작사/차종: 'FORD', '차량명칭: '포드', '모델: '기본형', ...	{공임: {탈착교환: '227,370', '판금수리: '363,000', ...	{[No: '1 U', '작업항목 및 부품명: '앞범퍼(스틸형)하단', '작업...	<pre> "images": { "id": 1, "width": 1900, "height": 1000, "file_name": "0000734_as-0000163.jpg" }, "annotations": [{ "id": 2, "image_id": 1, "category_id": "as-0000163", "segmentation": [...], "area": 1252.5, "bbox": [627, 342, 86, 57], "damage": "Scratched", "part": null, "year": 2015, "color": "White", "level": 1, "repair": ["Front bumper:repair,coating", "Front fender(R):sheet_metal,coating", "Head lights:exchange"] }, { "id": 5, "image_id": 1, "category_id": "as-0000163", "segmentation": [...], "area": 170492, "bbox": [42, 199, 694, 286], "damage": null, "part": "Front bumper", "year": 2015, "color": "White", "level": 1, "repair": "Front bumper:repair,coating" }] </pre>
1	{제작사/차종: 'BMW', '차량명칭: 'BMW 5시리즈', '모델: '5...}	{공임: {탈착교환: '152,460', '판금수리: '570,900', ...	{[No: '1', '작업항목 및 부품명: '뒤범퍼(단독작업)', '작업...	
2	{제작사/차종: 'CITROEN', '차량명칭: '씨트로엥', '모델: '기...	{공임: {탈착교환: '490,710', '판금수리: '650,100', ...	{[No: '1 U', '작업항목 및 부품명: '앞범퍼', '작업: 탈착...	
3	{제작사/차종: 'BMW', '차량명칭: 'BMW 520D', '모델: '기...	{공임: {탈착교환: '165,760', '판금수리: '1,664,000', ...	{[No: '1 U', '작업항목 및 부품명: '앞범퍼', '작업: 탈착...	
4	{제작사/차종: 'BMW', '차량명칭: 'NEW BMW 740Li', '모델: ...}	{공임: {탈착교환: '241,600', '판금수리: '2,563,520', ...	{[No: '1', '작업항목 및 부품명: '도어(뒤우,자동문)일체인지', ...}	

▲ 사고차량 건적서 파일 및 annotation 딕셔너리

사고차량 견적서의 차량정보, 수리비 정산정보, 수리내역의 행들을 각각 데이터프레임으로 변환 후 결합하였다. 마찬가지로 json 파일(이미지 annotation) 또한 데이터프레임으로 변환하였다.

차량정보 데이터프레임의 경우 '제작사/차종', '모델명', '입고날짜', '도장작업시간' 등의 컬럼이 존재하였고, 수리비 정산정보 데이터프레임의 경우 '탈착교환', '총계' 등 수리유형별 총 금액의 정보가 포함되었다.

견적서 데이터에 손상종류, 손상부위 등 이미지 데이터의 정보를 반영하기 위해 두 데이터프레임을 또다시 결합할 필요가 있었다. 이에 사고차량 견적 데이터의 파일명과 json 파일의 사고 ID 를 기준으로 매칭하여 통합 데이터셋을 생성하였고, 분석에 활용하지 않는 컬럼들은 모두 제외하였다. 본디 행이었던 차량 부위 클래스를 One-Hot encoding 과 pivot 을 통해 컬럼으로 변환하였고, 4 가지 손상종류 클래스 또한 컬럼으로 추가하였다. 손상종류 컬럼의 값은 json 파일의 segmentation 좌표를 이어 만든 폴리곤의 면적으로 채웠다.

id	brand	Breakage	Crushed	Scratched	Separate repair	공임	Roof	Windshield	Bonnet	Bumper	C pillar	Rear bumper	Rocker panel	Trunk lid	A pillar	Rear lamp	Rear door	Rear Wheel	Head light	Side mirror	Front fender	Front door
as-000001	BMW	5846.5	0	224559	0	[Rear bu	1515360	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
as-000002	CITROEN	0	0	397134	63774.5	[Front bu	2827810	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0
as-000003	BMW	0	0	9618	8546.5	[Front bu	2589760	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

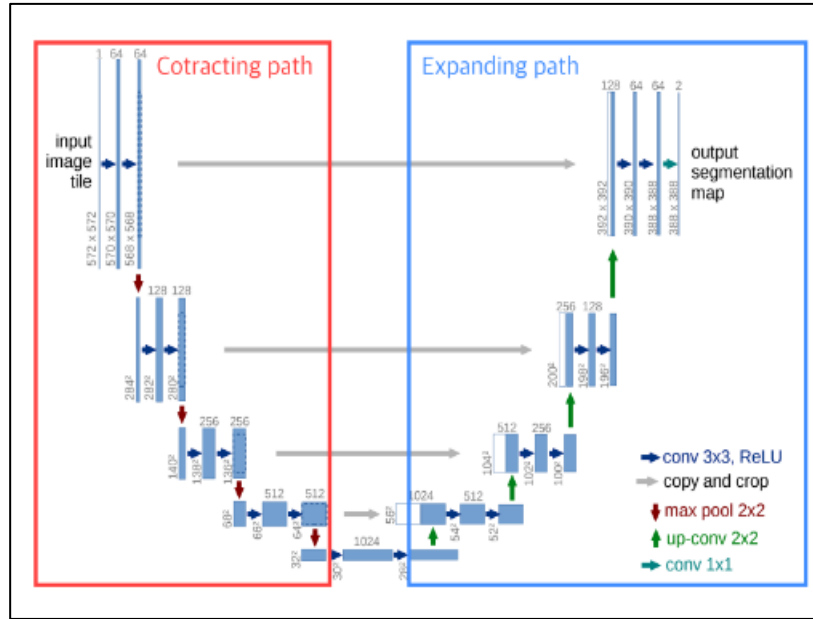
▲ 견적 예측용 통합 데이터셋

위 그림은 견적 예측용 모델에 사용할 통합 데이터셋의 컬럼이다. 예측한 label 은 '공임' 컬럼이며 독립변수로 문자형 변수인 id, brand, repair 컬럼을 제외한 모든 변수를 사용하였다. 사고 ID 를 기준으로 두 파일을 통합하였기 때문에 사고 ID 가 겹치지 않는 행들은 제외되어 총 105,848 개의 행과 30 개의 열을 가지게 되었다.

5.2 U-Net 모델 구축 – 손상종류

1) 사전학습(Pre-Trained)

AI HUB에서 제공하는 Pre-Trained Model을 가중치로 사용하였다. 사전학습모델은 ImageNet 데이터셋으로 학습되었다. Encoder의 역할을 수행하여 Down Sampling 역할을 담당하는 Contracting Path부분은 ResNet34로 지정하여 구축하였다. 여기서, Pre-Trained Model이 ResNet34 기반으로 학습되었기 때문에 Encoder로 다른 모델은 사용할 수 없다.



▲ UNet의 Contracting Path

2) 모델 설명

손상종류 분류모델은 총 4개로, 앞서 언급했던 Ground Truth들에 대한 모델이다. 각 모델의 Class number는 2로, 마스크 이미지의 Segment와 그 외는 배경으로 분류하였다. 모델을 따로 구축한 이유는 원본 이미지 데이터의 데이터 불균형 문제를 해결하기 위한 것이다. Class 불균형으로 인해 모델의 성능이 낮아질 수 있기 때문이다. 성능 지표로는 mIOU와 Accuracy를 사용하였다.

$$MIOU(\%) = 100 * \frac{1}{k} \sum_c \frac{TG_c}{TG_c + FE_c + FG_c}$$

▲ MIOU 계산식

* mIOU(Mean Intersection over union): 객체별 참값의 영역과 추정값의 영역의 교집합을 합집합으로 나눈 값
성능 검정은 원본 이미지 데이터와 겹치지 않는 AI HUB 사이트의 샘플 데이터를 이용하였다. 각각의 사전학습모델의 성능은 표에서 확인할 수 있다.

Predict Ground Truth	MIOU(%)	Accuracy(%)
Scratch(스크래치)	52.6	95.22
Separate(이격)	50.73	94.29
Crush(파손)	51.08	95.09
Breakage(찌그러짐)	40.59	74.72

▲ 모델별 MIOU, Accuracy

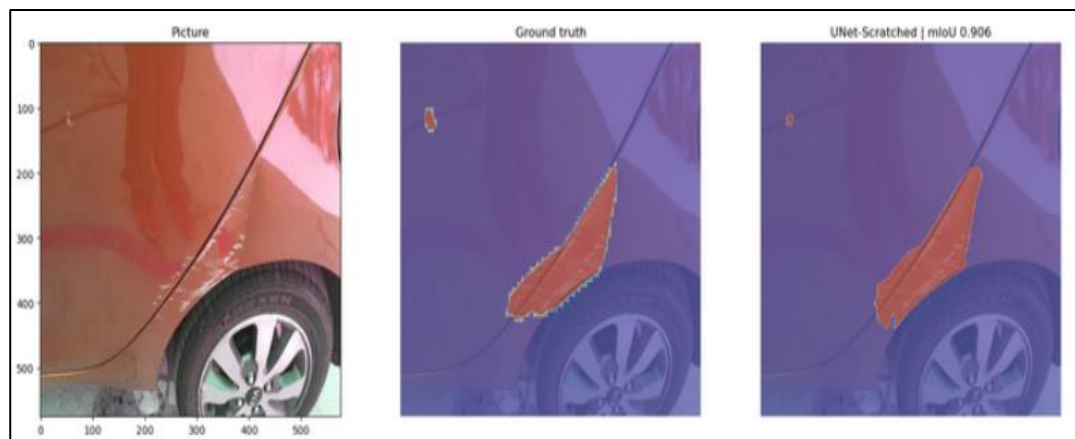
전체적으로 MIoU의 성능은 50% 초반으로 나타났다. Breakage 모델의 경우, 야간에 찍힌 사진과 픽셀 범위가 큰 사진들이 있어 성능이 타 모델에 비해 낮은 것으로 판단된다. Accuracy 또한 급격히 낮은 것을 알 수 있다.

3) 전이학습(Transfer Learning)

성능을 증강시키기 위해 앞서 구축한 모델에 추가적으로 학습을 진행하였다. 테스트는 이전과 같이 AI HUB의 샘플데이터를 이용하였고, 전체적으로 MIoU와 Accuracy가 증가함을 알 수 있었다. 특히 Breakage의 성능이 이전에 비해 크게 증가하였다. 다만, GPU Memory 문제와

Predict Ground Truth	MIoU(%)	Accuracy(%)
Scratch(스크래치)	59.5	96.8
Separate(이격)	58.59	98.22
Crush(파손)	56.28	95.58
Breakage(찌그러짐)	57.28	97.17

▲ 전이학습된 모델별 MIoU, Accuracy



▲ UNet 예측 결과 예시

5.3 YOLOv5 모델 구축 – 손상부위

YOLOv5 모델을 구축하기 위해 앞서 annotations json 파일의 Bounding Box 좌표들을 추출해내어 파손 부위 Ground Truth를 생성하였다. 하지만 서버의 용량부족 및 메모리 초과 문제 인해 모든 이미지 데이터를 활용하지 못하고, 약 12,000개의 이미지만을 Custom Dataset으로 생성하였다.

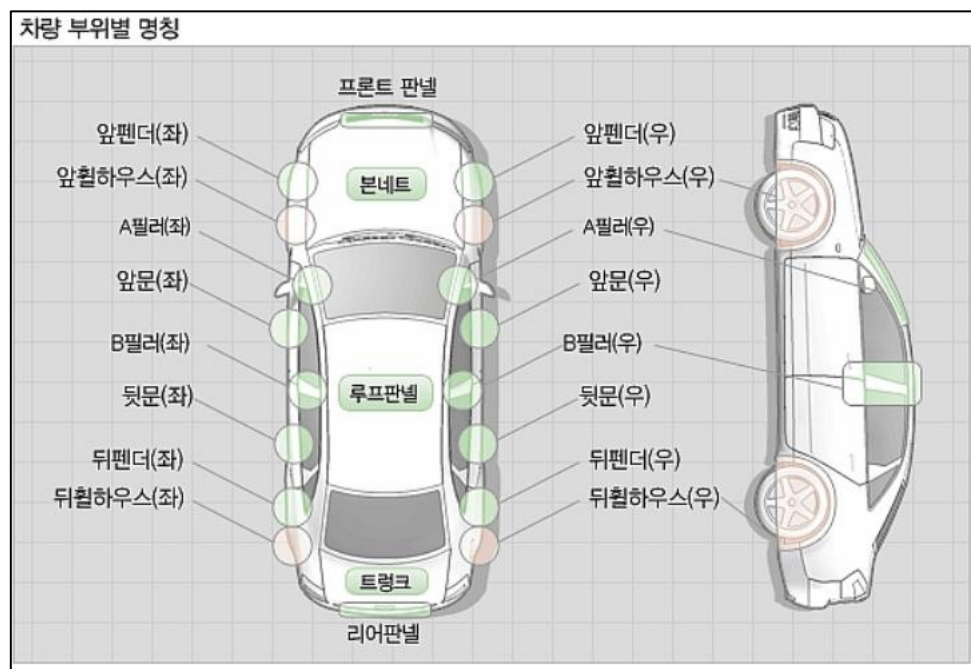
이후 YOLOv5x 모델과 YOLOv5s 모델을 각각 epochs 300, 100씩 학습시켜 손상 부위 탐지모델을 구축하였다. 성능 지표는 mAP를 활용하였으며, 학습용 Terminal에 노출되는 mAP@.5, mAP@.95

둘 중 mAP@.5를 최종 성능으로 결정하였다.

* mAP : 정밀도-재현율 곡선 아래 면적

YOLOv5s 모델은 epochs 수가 증가하면서 오히려 성능이 감소하는 양상을 보였다. 그리하여 YOLOv5x 모델을 최종 모델로 선정하고 학습을 진행하였다. 모델 예측 결과, 부족한 데이터셋 수에 비해 많은 Ground Truth로 훈련 세트에서는 mAP가 약 0.8의 성능을 나타냈지만, 테스트 세트에서의 mAP는 약 0.3으로 YOLOv5x가 매우 Overfitting된 결과를 나타냈다.

이에 Ground Truth의 수를 줄이고 클래스 불균형 문제를 해결하기로 결정하였다. 예시로 수리비 산정 시, 좌측 후미등과 우측 후미등의 가격은 다르지 않을 테니 (좌), (우)로 분리된 Ground Truth를 하나로 결합하는 방식이다.



▲ (좌), (우)로 구분된 Ground Truth

차량 부위별 통합을 통해 기존 32가지였던 Ground Truth의 수를 20가지로 줄일 수 있었다. 이후 YOLOv5x의 epochs를 300으로 지정 후 다시 학습을 진행하였다.

```

300 epochs completed in 17.110 hours.
Optimizer stripped from runs/train/exp5/weights/last.pt, 173.2MB
Optimizer stripped from runs/train/exp5/weights/best.pt, 173.2MB

Validating runs/train/exp5/weights/best.pt...
Fusing layers...
YOLOv5x summary: 444 layers, 86301265 parameters, 0 gradients, 204.2 GFLOPs

```

Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95: 100%
all	11451	19217	0.955	0.802	0.849	0.77
Trunk lid	11451	922	0.986	0.988	0.995	0.949
Rear bumper	11451	3033	0.963	0.99	0.992	0.954
Front fender	11451	2453	0.97	0.988	0.993	0.926
Front bumper	11451	4881	0.978	0.988	0.992	0.967
Rear door	11451	1107	0.969	0.981	0.992	0.941
Front door	11451	876	0.946	0.943	0.979	0.901
Bonnet	11451	902	0.965	0.983	0.991	0.936
Rocker panel	11451	570	0.92	0.925	0.964	0.847
Rear fender	11451	1602	0.972	0.988	0.993	0.916
Rear lamp	11451	499	0.932	0.976	0.988	0.848
Front Wheel	11451	468	0.969	0.987	0.993	0.945
Head lights	11451	1176	0.932	0.969	0.985	0.847
Rear Wheel	11451	192	0.975	0.995	0.995	0.948
Side mirror	11451	448	0.987	0.967	0.991	0.901
Roof	11451	5	1	0	0.0061	0.00488
Windshield	11451	19	0.85	1	0.988	0.871
Rear windshield	11451	32	0.778	0.938	0.952	0.869
C pillar	11451	16	1	0.438	0.839	0.52
A pillar	11451	9	1	0	0.108	0.0729
Undercarriage	11451	7	1	0	0.242	0.232

```

Results saved to runs/train/exp5
wandb: Waiting for W&B process to finish... (success).

```

▲ 재학습한 YOLOv5x의 훈련 세트에 대한 성능

클래스 20개로 재학습한 YOLOv5x의 mAP_0.5는 0.84885로 이전 모델에 비해 증가한 경향을 보였다. 하지만 클래스의 편차가 심하여 클래스 불균형 문제를 완벽히 해결하진 못하였다.

```

test: Scanning '/home/pi/Desktop/model2_final/datasets/customdata/labels/test' images and labels...
test: New cache created: /home/pi/Desktop/model2_final/datasets/customdata/labels/test.cache

```

Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95: 100%
all	1431	2343	0.66	0.427	0.469	0.395
Trunk lid	1431	116	0.587	0.517	0.539	0.485
Rear bumper	1431	399	0.805	0.812	0.809	0.695
Front fender	1431	287	0.733	0.688	0.729	0.618
Front bumper	1431	607	0.856	0.874	0.875	0.787
Rear door	1431	110	0.652	0.6	0.596	0.538
Front door	1431	112	0.735	0.438	0.549	0.481
Bonnet	1431	90	0.619	0.522	0.54	0.471
Rocker panel	1431	76	0.539	0.355	0.482	0.371
Rear fender	1431	163	0.621	0.607	0.58	0.478
Rear lamp	1431	65	0.481	0.292	0.32	0.236
Front Wheel	1431	72	0.641	0.556	0.618	0.568
Head lights	1431	140	0.484	0.286	0.283	0.218
Rear Wheel	1431	34	0.543	0.324	0.308	0.289
Side mirror	1431	61	0.852	0.623	0.691	0.508
Roof	1431	1	1	0	0	0
Windshield	1431	5	0.722	0.2	0.233	0.186
Rear windshield	1431	4	0	0	0.281	0.188
A pillar	1431	1	1	0	0	0

```

Speed: 0.0ms pre-process, 2.7ms inference, 1.3ms NMS per image at shape (32, 3, 256, 256)
Results saved to runs/val/exp4

```

▲ 재학습한 YOLOv5x의 테스트 세트에 대한 성능

테스트 세트에서도 이전 모델에 비해 mAP_0.5가 0.469로 이전 모델에 비해 mAP_0.5가 최소 0.1 이상 증가하였다.

YOLOv5x 모델을 구축했지만, annotation에서 가져온 Bounding Box의 좌표가 손상된 부위가 아닌 부품 자체의 위치에 박스가 쳐져 있는 문제를 발견했다. 그리하여 Bounding Box로 Ground Truth를 만드는 것이 아닌, 손상된 위치가 그려진 Segment를 토대로 Ground Truth를 생성한 후 YOLOv5x 모델을 epochs 100으로 학습하였다.

```
Run summary:
  best/epoch 99
  best/mAP_0.5 0.67212
  best/mAP_0.5:0.95 0.5687
  best/precision 0.8251
  best/recall 0.60192
  metrics/mAP_0.5 0.67199
  metrics/mAP_0.5:0.95 0.56869
  metrics/precision 0.82513
  metrics/recall 0.60192
  train/box_loss 0.03275
  train/cls_loss 0.00853
  train/obj_loss 0.01359
  val/box_loss 0.01774
  val/cls_loss 0.00285
  val/obj_loss 0.00551
  x/lr0 2e-05
  x/lr1 2e-05
  x/lr2 2e-05
```

▲ YOLOv5x의 훈련 세트에 대한 성능

학습한 YOLOv5x의 훈련 세트에 대한 성능은 MAP_0.5 기준으로 0.672로 나타났다. 이전 모델에 비해 mAP가 감소했지만, 테스트 세트에 대한 성능이 이전 모델보다 증가함으로써 과대적합 문제를 해결한 것으로 판단하였다.

```
test: Scanning '/home/piat/Desktop/model2_final/datasets/customdata/labels/test' images and labels...
test: New cache created: /home/piat/Desktop/model2_final/datasets/customdata/labels/test.cache
```

Class	Images	Labels	P	R	mAP@0.5	mAP@0.5:0.95	100%
all	1431	2343	0.605	0.557	0.508	0.415	
Trunk lid	1431	116	0.383	0.741	0.548	0.474	
Rear bumper	1431	399	0.709	0.932	0.846	0.717	
Front fender	1431	287	0.598	0.819	0.772	0.619	
Front bumper	1431	607	0.785	0.937	0.904	0.798	
Rear door	1431	110	0.473	0.764	0.633	0.535	
Front door	1431	112	0.491	0.577	0.566	0.475	
Bonnet	1431	90	0.371	0.678	0.531	0.442	
Rocker panel	1431	76	0.446	0.529	0.504	0.38	
Rear fender	1431	163	0.504	0.773	0.617	0.493	
Rear lamp	1431	65	0.303	0.446	0.4	0.246	
Front wheel	1431	72	0.519	0.722	0.64	0.57	
Head lights	1431	140	0.303	0.426	0.337	0.222	
Rear wheel	1431	34	0.403	0.5	0.288	0.25	
Side mirror	1431	61	0.694	0.689	0.721	0.525	
Roof	1431	1	1	0	0	0	
Windshield	1431	5	1	0	0.241	0.227	
Rear windshield	1431	4	0.901	0.5	0.579	0.471	
A pillar	1431	1	1	0	0.0237	0.0237	

```
Speed: 0.0ms pre-process, 2.6ms inference, 1.4ms NMS per image at shape (32, 3, 256, 256)
Results saved to runs/val/exp5
```

▲ YOLOv5x의 테스트 세트에 대한 성능

테스트 세트에서의 mAP_0.5는 약 0.508로 이전 모델의 테스트 세트에 대한 성능에 비해 약 0.03의 성능이 개선되었다. 훈련 세트와 테스트 세트에서의 성능이 크게 차이 나지 않는 것으로 보아 앞서 언급한 것처럼 과대적합 문제를 해결한 것으로 판단된다.

5.4 수리비 산정 모델 구축

앞서 생성한 수리비 산정 데이터 셋을 모델에 학습시켜 예상 수리비를 도출하였다. 모델 학습 시 분석에서는 문자형 컬럼인 id(사고ID), repair(수리 내역), Brand(차량 브랜드)를 제외하였다. 사고 ID의 경우, 식별자이기 때문에 제외하였고, repair 변수는 수리 내역이 여러 개 있는 경우, 리스트 형태로 입력되어 분석에 활용하기 적절치 않아 제외하였다. 또한 Brand 변수의 경우 One-hot Encoding을 진행할 시 생성되는 컬럼이 약 30개가 되어 모델 성능에 악영향을 미칠 것이라 판단돼 제외하였다. 따라서 모든 숫자형 변수를 독립변수로 두고, 공임 변수를 Target으로 선정하여 예측을 수행하였다.

변수명	비고	변수명	비고
Id	분석에서 제외	Side mirror	One-hot Encoding
Brand	분석에서 제외	Front fender	"
Repair	분석에서 제외	Front door	"
공임	Target	B pillar	"
Roof	One-hot Encoding	Front Wheel	"
Windshield	"	Rear fender	"
Bonnet	"	Front Bumper	"
Bumper	"	Undercarriage	"
C pillar	"	Rear windshield	"
Rear bumper	"	Breakage	segment 면적
Rocker panel	"	Crushed	segment 면적
Head lights	"	Scratched	segment 면적
		Separated	segment 면적

▲ 활용 변수 및 분석에서 제외된 변수

모델 학습 전, 훈련:검증:테스트 데이터 비율을 각각 8:1:1로 나누어 준 후, 변수 간 관측값의 편차를 줄이기 위해 Breakage, Crushed, Scratched, Separated 컬럼에 Min-Max 정규화 기법을 적용시

켜 0부터 최대 1의 값을 갖도록 스케일링을 진행하였다

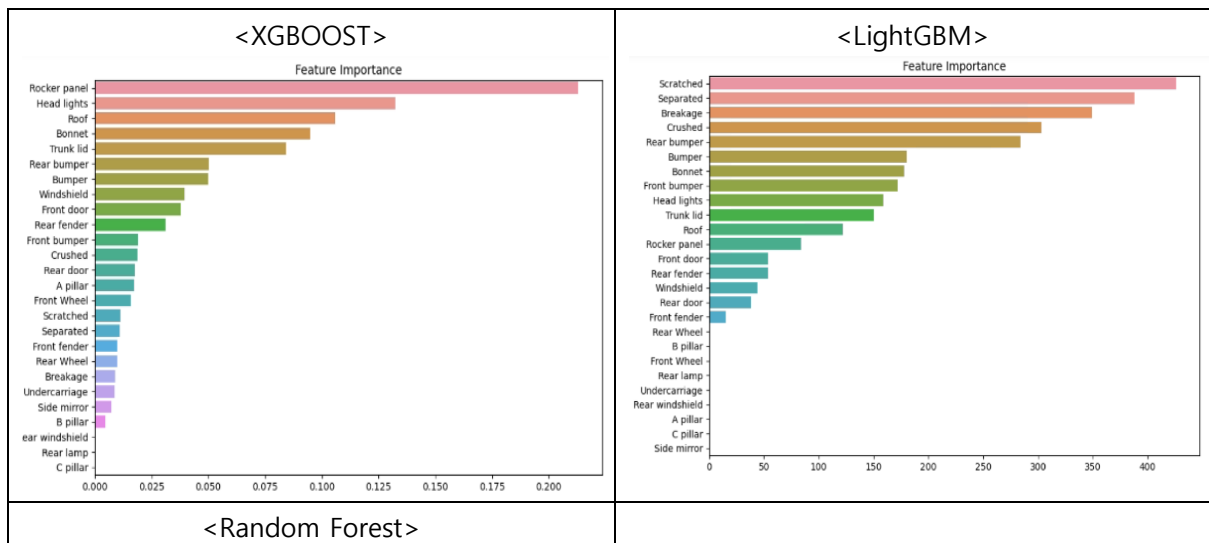
- 활용 모델

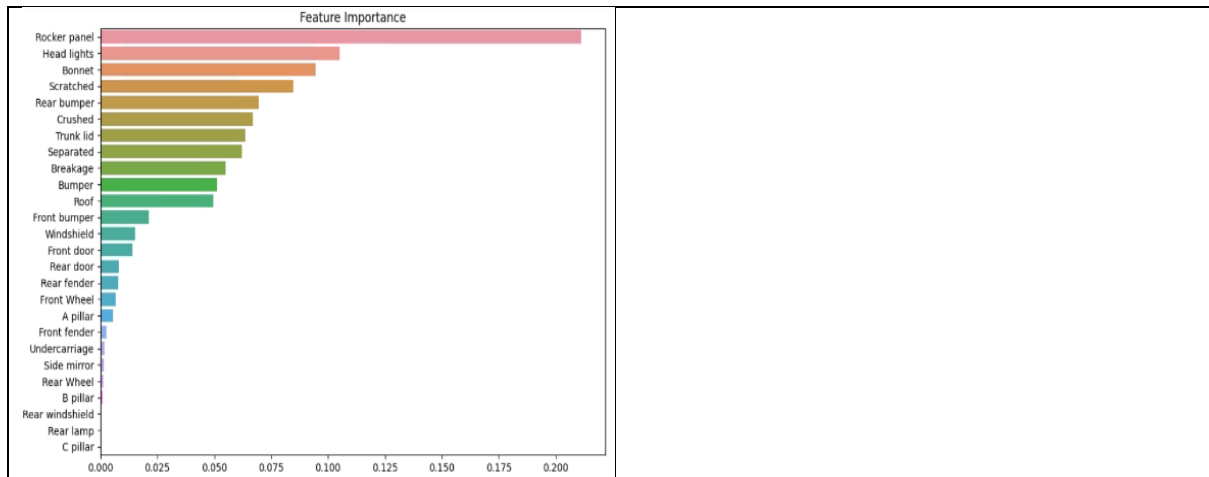
분석에는 XGBOOST, LightGBM, Random Forest, MLPRegressor로 총 4개 모델을 활용하였다. 성능 지표는 RMSE로, 오차제곱합의 제곱근을 뜻한다. 산정 방식은 이와 같다.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

▲ RMSE 연산식

모델 적합 시 MLPRegressor를 제외한 모든 모델에 Random Search, K-fold CV를 적용시켰다. K의 개수는 모두 4이고, Random Search의 반복 횟수는 5로 모두 동일하였다. 각 모델의 RMSE를 비교하여 최적의 모델을 선정하였다. 선정 기준은 테스트 데이터에 대한 RMSE가 가장 낮은 모델이고, MLPRegressor를 제외한 각 모델의 변수 중요도를 시각화하였다.





▲ 각 모델의 변수 중요도

각 모델에서 조정한 하이퍼파라미터는 이와 같다.

모델명	하이퍼파라미터
XGBOOST	<ul style="list-style-type: none"> - max_depth = randint(6, 10) - n_estimators = randint(50, 200) - learning_rate = uniform(0.01, 0.2) - gamma = uniform(0, 5)
LightGBM	<ul style="list-style-type: none"> - n_estimator = [100, 200, 300] - max_depth = [10, 20, 30, -1] - learnig_rate = [0.01, 0.05, 0.1] - num_leaves = [31, 41, 51] - min_child_samples = [20, 30 ,40]
Random Forest	<ul style="list-style-type: none"> - n_estimators = [100, 300, 500] - max_depth = [6, 10, 15]

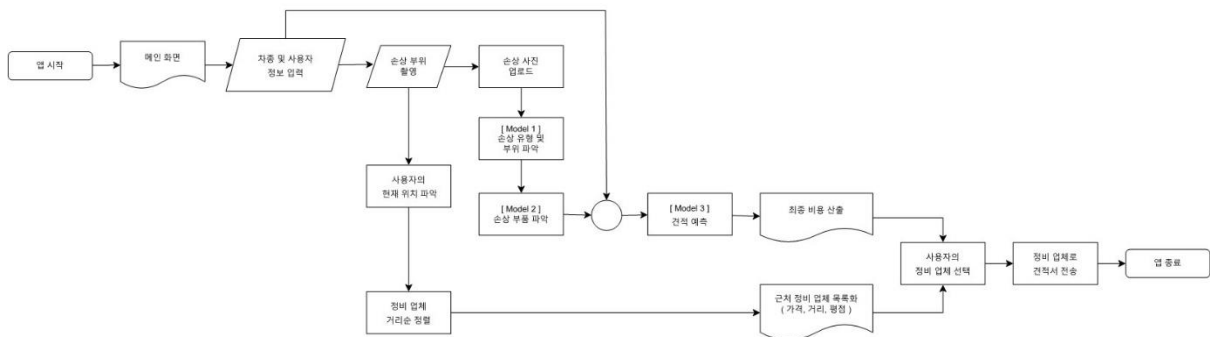
▲ 하이퍼파라미터 튜닝 세부 내용

모델명 : 결과	모델명 : 결과
<p><XGBOOST></p> <p>검증 데이터에 대한 예측 결과: 훈련 데이터에 대한 RMSE: 308681.6654902415 검증 데이터에 대한 RMSE: 317592.8916230205 테스트 데이터에 대한 RMSE: 347683.77755271445</p>	<p><LightGBM></p> <p>검증 데이터에 대한 예측 결과: 훈련 데이터에 대한 RMSE: 334686.24682819896 검증 데이터에 대한 RMSE: 314554.1923086802 테스트 데이터에 대한 RMSE: 347689.08705331414</p>
<p><Random Forest></p> <p>검증 데이터에 대한 예측 결과: 훈련 데이터에 대한 RMSE: 307630.65384401684 검증 데이터에 대한 RMSE: 319906.2206864982 테스트 데이터에 대한 RMSE: 382323.5350505796</p>	<p><MLP Regressor></p> <p>예측 결과: 훈련 데이터에 대한 RMSE: 379090.09284997464 검증 데이터에 대한 RMSE: 352255.28556274716 테스트 데이터에 대한 RMSE: 382323.5350505796</p>

테스트 세트에서 가장 성능이 좋은 XGBOOST 모델을 최종 모델로 선택한 후, 수리비 산정모델로 이용하였다. (변경될 수 있음)

5.5 V5 앱 구현

앞에서의 과정을 인터페이스로 표현하기 위해 앱으로 구현해보았다. 전체 과정은 다음과 같다.



1) 구현 과정 설명

앱을 실행하면 시작화면이 나타나며, “견적 받기” 버튼을 누르면 차량 번호를 입력하고 개인 또는 법인/리스 소유주 구분을 선택할 수 있다. 개인인 경우에는 소유자명을 입력하고 차량 브랜드를 선택한다. “시작하기” 버튼을 누르면 견적서 정보를 입력하는 화면으로 전환된다.

사용자는 차량 외부수리 또는 기타 수리 여부를 선택하고, 차량 손상 부분을 확인하기 위해 이미지를 촬영하여 업로드한다. 또한 GPS를 기반으로 하여 근처 정비소를 탐색하고, 포항 지역 내에서 주변 정비소를 확인할 수 있다. 이때, 사용자가 업로드한 이미지를 기반으로 견적 가격을 예측하는 AI 견적서와 정비소별 견적서가 생성된다.

견적서는 AI 견적서와 정비소별 견적서로 나뉘어져 있으며, 사용자는 업로드한 이미지를 기반으로 산출된 AI 견적서와 거리순으로 정렬된 정비소 견적서를 한눈에 확인할 수 있다. 또한, 선택한 견적서를 클릭하면 해당 견적서의 세부 사항을 확인할 수 있다. 이를 통해 사용자는 간편하게 견적 가격을 비교하고, 선택한 견적서의 세부 내용을 확인할 수 있다.

2) 작업환경 및 기능 설명

① 작업환경 설명



해당 앱은 안드로이드 스튜디오에서 Kotlin으로 작업했다. 작업환경을 iOS Swift가 아닌 안드로이드 Kotlin으로 선택한 이유는 다양한 이유가 있다. 먼저, 안드로이드는 다양한 스마트폰 제조사들이 사용하고 있어 다양한 종류의 디바이스에서 동작할 수 있기 때문이다. 이는 다양한 시장 점유율을 가진 스마트폰들에서 앱을 구현하고자 할 때 큰 장점이 될 수 있다. 또한, 안드로이드는 개방성이 높아 개발자들이 더 많은 커스터마이징과 개발 가능한 영역이 있다. 다양한 앱 스토어와의 호환성이 있어 개발자가 원하는 앱 배포 경로를 선택할 수 있다. 안드로이드 앱은 Java, Kotlin, C++, Python 등 다양한 언어로 개발이 가능하며, 다양한 도구와 라이브러리가 제공되어 있어 편리하게 개발할 수 있다.

iOS의 경우, 하드웨어와 소프트웨어의 일관된 통합이 이루어져 일관된 사용자 경험을 제공한다. 또한, iOS는 애플의 강력한 보안 기술로, 개발자와 사용자의 개인 정보 보호에 대한 안정성이 높다. 애플은 개발자들에게 애플의 개발 도구와 라이브러리를 제공하여 개발을 용이하게 한다. 이는 애플의 다양한 디바이스들이 있기 때문에 특정 하드웨어 특성을 활용한 앱을 개발할 수 있다는 장점이 있다.

두 모바일 운영체제는 각각 장단점이 있지만, iOS의 경우 커스터마이징의 제한성, 디바이스 간 호환성과 종류를 고려하였을 때, 앱 커스터마이징과 하드웨어와의 연결 및 다양한 종류가 우선순위인 상황에서 적합하지 않다고 판단하였다.



안드로이드에서 지원하는 다양한 개발 언어 중 Kotlin을 선택한 이유는 먼저 안드로이드 앱 개발을 위한 공식 언어로 지원되어 개발 생산성이 높기 때문이다. 또한 자바와 Java와 상호 운용성이 뛰어나고 간결하고 표현적인 문법을 가지고 있으며 null 안정성, 확장 함수 등의 현대적인 기능을 제공하여 개발 품질을 향상시킬 수 있다. 개발 환경과 요구 사항에 따라 언어를 선택해야 하지만, Kotlin은 안드로이드 앱 개발을 위한 공식 언어로서 뛰어난 선택이며, 다양한 기능과 호환성, 간결한 문법 등의 장점을 가지고 있다. 안드로이드 앱 개발에 특화된 생산성과 품질 향상을 기대할 수 있으며 이미 자바로 개발된 코드를 확장하거나 Kotlin 코드를

Java로 호출하는 것이 가능하다.

② 기능 설명

앞의 구현 과정에서 설명했듯이, 전체적인 프로세스는 위와 같다. 이 부분은 전체적인 흐름에서 더 나아가, 기능적인 부분에 대한 설명이다.

차량 정보 및 업로드한 이미지를 이용하여 "Semantic Segmentation"을 적용한 모델에 이미지를 입력한다. "Semantic Segmentation"은 이미지에서 픽셀 단위로 객체의 클래스를 예측하는 기술로, 차량 이미지에서 손상 부품을 식별하기 위해 사용된다. "Semantic Segmentation" 모델은 이미지를 분석하여 손상 부품의 위치와 영역을 예측한다. 이를 토대로 손상 부품과 정도에 대한 정보를 추출한다. 예를 들어, 차량의 앞 범퍼가 손상되었고, 그 정도가 스크래치 수준이라면 해당 정보를 추출한다. 추출된 손상 부품과 정도에 대한 정보를 기반으로 가격 예측 모델에 입력하여 견적서를 생성한다. 가격 예측 모델은 손상 부품과 정도에 따라 해당 부품의 예상 가격을 산정하는 모델로, 사전에 학습된 데이터를 기반으로 예측을 수행한다.

가격 예측 모델이 생성한 견적 정보를 JSON 파일 형태로 안드로이드 파일에 전달한다. JSON은 경량 데이터 교환 형식으로, 안드로이드에서 쉽게 파싱하여 사용할 수 있다. 안드로이드 애플리케이션 내에서 전달받은 JSON 파일을 파싱하여 브랜드명, 손상 부품, 손상 부품 가격, 공임비 등의 정보를 매칭시켜 견적서를 생성한다. 이를 통해 사용자는 손상된 차량의 견적서를 확인할 수 있다. 이와 같은 과정을 통해 사용자가 업로드한 차량 이미지를 기반으로 손상 부품을 식별하고, 해당 부품의 가격 예측을 통해 견적서를 생성하는 AI 견적서 생성 시스템이 동작하게 된다.

3) 한계점 및 개발일지

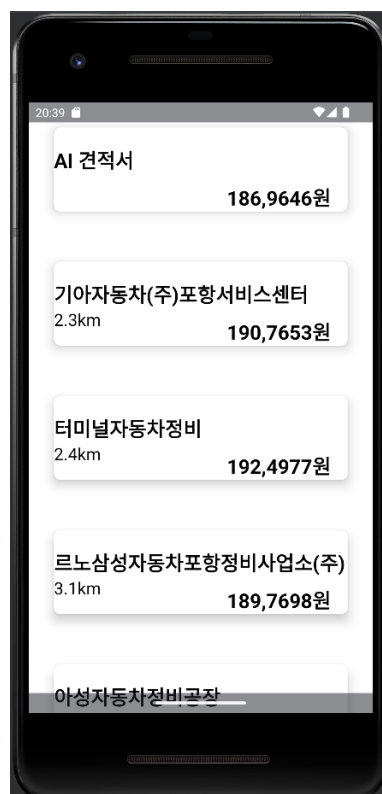
① 한계점

에뮬레이터로 구현할 때에는 카메라 촬영도 되고, 갤러리에서 이미지 업로드도 가능했는데 하드웨어로 시뮬레이트했을 때 카메라 권한 설정만 가능하고 기능이 연동되지 않았다. 그래서 외부 저장소의 이미지를 갤러리와 연결하여 이미지를 업로드하는 것으로 구현하였다.

```
val tw = LatLng( latitude: 35.99686753, longitude: 129.3543555)
mMap.addMarker(MarkerOptions()
    .position(tw)
    .title("(주)태원자동차정비"))
mMap.moveCamera(CameraUpdateFactory.newLatLng(tw))
```

또한, Open Api를 통해 지도에 정비소 목록에 대한 json파일을 불러와서 지도에 마커를 추가하고 싶었는데, 연결되지 않아서 하나씩 매칭시켜 마커를 추가하였다. 결과적으로는 같은 결과이지만, 하드코딩으로 구현한 것이기에 비효율적으로 구동되어 전송 속도가 느린 것이 한계점이다.

가격 예측 모델을 불러오면서 초기값 설정 오류로 인하여 브랜드와 부품 예측이 잘 안 되는 이슈가 있었다. 따라서 모델링을 돌린 값을 텍스트로 집어넣어 결과를 도출하는 방식으로 방법을 바꾸었다.



② 개발일지

앱 개발에 대한 경험이 전혀 없는 비전공자로서, 안드로이드 앱을 구현하기 위해 도전하게 되었다. 초기에는 안드로이드 스튜디오 설치하는 것만으로도 많은 시간이 소요되었다. 또한, Java나 코틀린과 같은 언어에 대한 경험이 없어 어려움을 겪었다. UI 디자인은 피그마를 사용하여 구현하였고, 전체적인 구현은 안드로이드 스튜디오에서 진행되었다.

어려움에도 불구하고, 구글 검색, 유튜브, Git Hub 등을 적극 활용하여 관련 예제를 찾아가며 하나하나 적용해보는 과정을 수행하였고, 함수가 'deprecated'된 경우를 확인하고 오류를 해결하기 위해 노력하였다. 첨부한 파일을 통해 확인할 수 있는 소스코드는 자바나 객체 언어에 대한 구조를 이해한다면 본 개발자보다 더 간결하게 코드를 작성할 수 있을 것이라 기대한다. 앱 구현시 DB를 쌓게 된다면, DB 구조 또한 잘 파악하는 것이 중요하다. SQLite을 공부하면 좋을 것 같다. 안드로이드 스튜디오 설치 시 컴퓨터가 뜨거워지고 오류가 발생할 수 있다는 점을 유의해야한다. 마지막으로, 당연히 경험이 있는 개발자가 개발을 담당하는 것이 가장 좋겠지만, 만약 그럴 수 없는 상황이라면 초보자도 간단한 부분은 다양한 구글링과 검색을 통해 앱을 만들어낼 수 있을 것이다.

6. 결론

6.1 한계점 및 개선사항

1) 부족한 서버 저장용량

인공지능 연구원 측에서 서버 주소를 받았지만, 예상보다 서버에서 사용할 수 있는 용량이 적었다. 따라서 AI HUB의 모든 이미지 데이터를 활용하지 못하고, 그 중에 일부 이미지만 분석에 활용할 수 있었다. 이러한 문제 때문에 구축한 모델이 기대보다 낮은 성능을 보였다.

➔ 개선 방안

외장하드 구매나 저장용량을 증강시킬 수 있는 방법들을 찾길 권장한다. 또는 Data Augmentation 옵션을 적극 활용하여 적은 이미지 데이터에서도 성능을 증강시킬 수 있는 방안을 활용하는 것도 좋은 방법이다. 데이터를 분할하여 조금씩 학습시키는 것도 좋은 방안이다.

2) GPU Memory 초과

Local PC와 서버에서 모두 CUDA GPU를 사용하였는데 이 GPU의 메모리 가용량이 매우 낮아 U-Net의 경우 CPU로 모델을 학습할 수밖에 없었다. 시간이 많이 들었음에도 불구하고 모델의 성능은 mIoU 기준 약 10%밖에 증가시키지 못했다.

➔ 개선 방안

부족한 GPU 가용량을 최적화하는 방안을 모색하거나, U-Net보다 학습속도가 빠른 모델을 활용하는 것을 권장한다.

3) BBox로 인한 문제

Object detection에서 BBox를 사용했는데, 객체가 아닌 곳까지 Box가 포함하고 있는 경우 또는 객체 전체를 BBox가 잡지 못하는 경우 사각지대가 발생하였다. 또한 한 사진 내에서 인접해있는 여러 개의 손상을 감지하면서, BBox가 겹치는 중복 검출 현상이 발생하기도 했다.

➔ 개선 방안

라벨링 재작업, 차량 부위 추가 학습을 통해 바운딩 박스의 사각 지대를 감소시킬 수

있다. 또한 중복 검출 현상은 YOLO 네트워크에 중복 검출 방지 알고리즘을 적용함으로써 해결할 수 있다.

4) YOLOv5x 성능 개선

프로젝트 막바지에 이르러 학습한 YOLOv5x 모델이 이전의 과대적합 문제를 해결한 것으로 사료된다. 더불어 epochs 수를 증가시켰으면 더욱 좋은 성능의 모델이 될 수 있을 것이다.

6.2 기대효과 및 활용방안

1) 인슈어테크 분야 활용 가능성

인슈어테크 시장은 보험(insurance)과 기술(technology)의 합성어로 기술과 보험업의 융합이다. 상품개발과 계약체결, 고객관리 등 각 분야의 보험 업무에 사물인터넷(IoT), 빅데이터, 인공지능(AI) 등 정보기술(IT)을 융합해 효율적이고 혁신적인 서비스를 제공하는 것을 말한다.

본 프로젝트에서 제안한 모델을 사용할 시, AI가 사진을 바탕으로 부품 종류, 손상 유형 및 손상위치를 인식하고 수리비 견적 산출까지 자동 처리해준다. 이러한 기술을 인슈어테크에 적용한다면, 보험 처리에 있어서 접근성과 편의성을 높일 수 있을 것이다. 현재 구현 상태로는 수리가 필요한 부품을 스스로 인식하고, 보험사 직원 등 관련된 사람에게 자동으로 송부하는 기능 정도를 고려해볼 수 있다.

여기에 추가적으로 사고차량 번호판을 자동 인식하는 OCR 기술을 연동할 수 있다. 이를 통해 보험 계약 정보와 자동연결, 보상 업무 처리 시간 손실을 최소화할 수 있다는 점에서, 보험 업무 처리에서의 효율성 증대 또한 기대해볼 수 있다.

2) 소비 시장 전체에서의 신뢰도 향상

현재 자동차 대수가 증가하면서 국내 인구 2명 중 1명이 차를 소유할 정도로 소비 시장은 커지고 있는 추세이다. 하지만, 소비자시장성과지수(KCMPI)는 평가 항목 중 '선택 다양성', '비교 용이성'을 제외한 모든 항목이 소비자지향성경고 영역에 해당될 정도로 시장에 대한 신뢰도는 낮은 상태이다. 따라서 AI를 활용한 객관적인 수리 견적서를 소비자들에게 전달해줌으로써 소비자 개개인의 신뢰를 얻을 수 있으며, 나아가 소비 시장 전체에서의 신뢰도 또한 높일 수 있다.

7. 참고문헌

1. 강갑생, 강병철. "[Data & Now] 2022 년 1 분기 자동차 누적 등록 대수...2507 만 180 대" 중앙일보. 2022 년 4 월 14 일. <https://www.joongang.co.kr/article/25063287#home>
2. 이인혁. "과잉 수리·허위 청구에 줄줄 새는 車 보험금" 한경 금융. 2022년 10월 10일. <https://www.hankyung.com/economy/article/2022101073281>
3. 박윤호. "보험개발원, AI 기반 AOS알파 시스템 고도화 작업 착수" 전자신문. 2021년 4월 22일. <https://www.etnews.com/20210422000160>
4. "「자동차수리서비스」 시장, 21개 서비스 시장 중 소비자시장성과지수(KCMIPI)가 가장 낮아 경고 시장으로 분류." 한국소비자원. 2022년 6월 23일, 2023년 04월 19일 접속, <https://www.kca.go.kr/home/sub.do?menukey=6055&mode=view&no=1003327339>
5. "AI 이미지인식을 통한 수리비 자동견적 시스템 구축 및 활용방안." 한국화재보험협회. 2019년 3월, 2023년 4월 19일 접속, <https://www.kfpa.or.kr/webzine/201903/sub/disasters4.html>
6. "지금 뜨는 인슈어테크." 일분톡 블로그. 2021년 6월 23일, 2023년 4월 19일 접속, <https://m.post.naver.com/viewer/postView.naver?volumeNo=31826345&memberNo=1271504>
7. "Semantic Segmentation을 활용한 차량 파손 탐지 딥러닝 모델 개발기." 쏘카 테크 블로그. 2020년 2월 13일, 2023년 4월 19일 접속, <https://tech.socarcorp.kr/data/2020/02/13/car-damage-segmentation-model.html>
8. "[딥러닝] Object Detection with YOLO." Hhhong.log 블로그. 2022년 1월 12일, 2023년 4월 19일 접속, <https://velog.io/@hhhong/Object-Detection-with-YOLO>
9. "DeepLab v3 (Rethinking Atrous Convolution for Semantic Image Segmentation)." JINSOL KIM 블로그. 2019 년 11 월 4 일, 2023 년 4 월 19 일 접속, <https://gaussian37.github.io/vision-segmentation-deeplabv3/>
10. 최태범. "보험시장 뒤흔드는 첨단메기...'인슈어테크 스타트업' 뜬다[빅트렌드]" 머니투데이. 2021년 7월 4일. <https://news.mt.co.kr/mtview.php?no=2021070115434765048>
11. 김경희. "국내 기업 10곳 중 9곳 'AI 추가 도입 계획 있다'...사내 업무효율 개선에 가장 기대" 조선일보. 2019년 11월 21일. https://digitalchosun.dizzo.com/site/data/html_dir/2019/11/21/2019112180134.html

8. 상호평가

[권구택]

김현철	우리 조 든든한 조장 구스택! 전공자가 퇴소한 시점에서 우리 조의 코딩은 거의 도맡다시피 했지.. 그런데도 힘든 내색 안하고 프로젝트 끝나는 날까지 고생해줘서 고마워. 과목 공부에 과제에 프로젝트에 취준까지 4가지 일 병행하며 10주 지내느라 고생했다! 너가 가르쳐 준 여러 코딩들, 라이브러리 활용법이라던지 알고리즘이라던지 정말 잘 배웠고 잘 가르쳐줬다고 생각해. 올해 꼭 원하는 회사 취직하길 바랄게. 그리고 떨어지기 금지 떨어지고 개 구데기네~ 금지
박수림	두달이라는 시간동안 든든하게 코딩해준 코천(코딩천재) C1 조장 권구택! 처음에는 다가가기 어려운 사람일 거라고 생각했는데 알면 알수록 새로운 모습을 보여주면서 항상 편안하게 분위기 만들어준 덕분에 프로젝트 하는 내내 부담 없이 웃을 수 있었어! 취준도 같이 병행하느라 힘들었을텐데 티 한 번 안내고 코딩 과제도 시험도 묵묵히 잘 해내는 모습이 솔직히 대단했다! 인정한다! 앞으로도 어디서든 무얼하든지 잘 할 거라고 생각되는 사람이여서 긴 말 필요 없을 것 같지만,,그래도 항상 응원하고! 이번년도 꼭 취뽀해!!!!
임지연	아카데미 기간 동안 항상 조장으로 열심히 우리를 이끌어준 구스택~ 당신 C1조의 MVP야~ 내가 내어줄게~ 처음엔 정 붙이기 힘들겠다 생각했는데 지금 생각하면 다 즐거웠던 것 같아! 우리의 코천 재능기부 구스택 없었으면 프로젝트 못했을거야,,ㅎㅎ 10주 동안 밤 새면서 견뎌줘서 고마워~ 또 옆에서 에러랑 오타 찾아달라고 귀찮게 짱짱 거렸을텐데 챙겨줘서 고마워! 하고 싶은 일 다 잘 될거야! 올해 취업 대박나자~!!
윤혜연	지난 주에 조장 없을 때 빈자리를 크게 느꼈습니다.. 잡담 파티였습니다. 아무튼 두달 동안 조장으로서 너무너무 고생많았어~ 리더라는 게 성향에도 안 맞고, 어떻게 보면 생판 처음보는 사람들 이끌어가는 게 결코 쉬운 게 아닌데 덕분에 무사히 마무리한 것 같아. 근데 성향에 안맞는 거 치곤 잘하던데. 아무튼 우리 조장이라는 게 든든했고 의지도 많이 됐다. 근데 반에서 공놀이랑 리리코 남친 성대모사는 좀 자제해줘. 카톡 배경 사진과도 빠른 시일 내에 원만한 합의 보길 바란다. 여태껏 해왔던 것처럼, 앞으로도 잘 준비해서 뭐든 이뤄내길!

[김현철]

권구택	<p>우리의 예스맨 현철이. 이제 너는 mIOU와 RFM이다. 진짜 처음 해보는 코딩이었을텐데 묵묵히 열심히 따라와주고 의견 존중해줘서 고마워. 너 덕분에 나도 열심히 코딩하고 우리가 모델 구현에 성공한 것 같음. 내가 보기엔 너 코딩 재능 있어. 뭐랄까 든든한 코딩 버팀목이었다. 내가 코딩하다가 놓치는 부분도 항상 잘 캐치해주고 너랑 같이 코딩하면 시너지가 발생하는 것 같았어. 진짜 고생했어</p>
박수림	<p>두달동안 나랑 짝궁한 현철아 너 솔직히 코딩 잘해 ㅎㅎ 항상 내 옆에서 나 모르는 거 도와주고 내가 뭐라해도 흐흐 웃으며 넘어가는 모습,,두달동안 내 옆에 있으면서 행복했니..? 난 너랑 짝궁해서 좋았어! 특히 너가 코딩 천재로 성장하는 모습이 압권이란다 AI 프로젝트 하면서 한계를 느낀다고 했는데 내가 봤을 때는 넌 그 한계도 이겨낼 열정 뽐이야~! 수료 후에 하는 공모전에서 꼭 상 타길 바랄게! 화이팅!!!</p>
임지연	<p>나의 견제러 현철아 너 덕분에 내가 열심히 견제할 수 있었어,,ㅎㅎ 우리 평생 견제하며 살자~ 10주 내내 내 페이스메이커가 되어줘서 그리고 내가 귀찮게 과제 물어봐도 같이 해결해줘서 정말 고마웠어! 내가 항상 장난 걸고 시비 거는 거 받아주느라 고생했고 앞으로도 계속 받아야 해 견제러 현철아 우리 손절하기 전에 내가 너 먼저 손절한다 현철아~ 나랑 같이 평생 견제하며 살자 핫팅~</p>
윤혜연	<p>맨날 새벽까지 남아서 모르는 코딩 문제 풀고 있던 게 기억이 난다. 심지어 과제도 아닌데! 답도 있는데! 이렇게나 저렇게나 코딩 실력이 늘 수 밖에 없을 듯 인정. 모르는 거 물어보면 맨날 서론-본론-결론 길게 설명해줘서 고마워. (feat. 자사를..자사..자사를...) 덕분에 그냥 지나갈 뻔 했던 것도 다시 보고, 수업도 잘 따라갈 수 있었던 것 같다! 그리고 우리가 비록 졸렬 C1이지만 당신은 정말 착해 ㅌㅌ 아무튼 건강하고 주눅들지 않고 쫓대 있는 삶을 살자 현철스! 아 포켓보이 컨셉은...좀.....</p>

[박수림]

김현철	에어팟 끼고 있으면 가끔 울리는 C1사랑스러운존예깜찍이한개의복숭아이모티콘님의 메시지.. 이젠 외워버렸다.. 사실 웃겨서 이름 안바꾸고 있어. 아무튼 10주간 내 짝공하느라 힘들었을 수도 있지만 그냥 좋았다고 해~ 장난이고 빅데이터, AI 프로젝트 간 내가 너에 대해서 느꼈던 건 정말 세밀하고 기획력이 좋다는 점이야. 맨날 헤실대는 수림이가 아닌 느낌...? 프로젝트 방향이 어긋나면 곧바로 잡아주고 주제가 산에 가지 않게 잘 붙잡아준 점이 너무 좋았어. 끝나고 대학원 진학하고 싶다 그랬는데 원하는 대학원 꼭 가길 바라~ 못가면 탈모음
권구택	고생 많았어 수림아. 너 덕분에 우리가 프로젝트를 진행할 때 여러 방면에서 생각하고 좀 더 구체적으로 잘 진행할 수 있었어. 너의 서칭 능력? 그거 칭찬해. 항상 관련 논문이나 자료를 가져와서 설명해주고, 우리가 더 잘 이해할 수 있도록 도와주는 멋진 모습 진짜 훌륭해. 조용히 자기 할거 하고 학생회관 마스코트 수림이 진짜 고생 많았어.
임지연	졸렬 C1조의 디자이너 박수림 없었으면 세 꼭지 못 챙겨갔잖아~ 우리 아카데미 와서 첫날부터 수다 떤 것부터 생각하면 그때부터 잘못됐어. 항상 내 멘탈 챙겨주고 옆에서 할 거 챙겨줘서 고맙다 수림아 너 덕에 프로젝트랑 피피티 정말 잘 마치고 해낼 수 있었어. 너가 있어서 아카데미 견딜 수 있었어~ 너 없으면 길 잃을 뻔 했잖아^^ 대학원 못 가면 탈모 빔 맞음ㅋ
윤혜연	언니 내가 아는 사람 중에 제일 웃겨. 아니 웃기다고 하면 좋아할 것 같아서 말하고 싶지 않아. 아무튼. 나 계획 잘 안세우고 사는 사람인데, 언니가 맨날 회의 준비 뽀세게 해오는 모습 보고 대단하다고 생각했엉. 언니가 주제 방향성 검토 안해줬으면 발표 준비하면서 더 어려움 겪고 있었을 지도 몰라. 그리고 이 참에 디자이너 해보는 건 어때? 미리캔버스 세상에서 제일 잘 이용하는 사람같아. 여기와서 발표하면서 피피티 걱정해본 적 없다~ 아 마지막으로 언니 컴퓨터에 요상한 사진 많던데 다음 기수를 위해 지우고 가 ㅎㅎ 그리고 마지막이니까 예쁘고 상큼하다고 해줄게.....

[임지연]

김현철	견제킹 임지연~~!! 코딩하는 동안 뒤에서 견제하고 같은 팀인데도 견제하는 이 말 많고 텐션 높은 ENFP야! 너 성은 이제 임 아니고 EN이다. AI프로젝트 하는 동안 접해보지도 않았던 코틀린한다고 고생했어. 나였으면 죽어도 못했을거야. 지금 이거 쓰는 동안에도 내꺼랑 줄 수 맞춘다고.. 아무튼 며칠 밤새서 어플 만든다고 기운 빠진 거 볼 때 마다 AI 프로젝트 전의 너가 그림기기도 해. 이제 곧 돌아오겠지..? 계속 밝은 모습 갖길 그리고 끝나고 하는 공모전 너도 상 타고 나도 상타길~
박수림	ENFP 그 자체인 지연이~! Aka. 코틀린 천재 임지연~! 너 덕분에 우리가 빨리 친해질 수 있었고 너 덕분에 두달 동안 웃을 수 있었어 나이는 제일 어려도 나보다 성숙해 보일 때도 많고 그리고 너의 기억력 정말 리스펙 해 너가 몸이 많이 힘들었을 텐데 티 내지 않고 묵묵히 프로젝트 참여할 때마다 걱정됐는데 이제는 잠 많이 자서 다시 건강해지길! 그리고 꼭!!! 어플 때 술 마시지 마라..^^ 수료 후에 하는 공모전에서 너도 꼭 상 타서 인스타로 자랑해줘!! 화이팅!!!
권구택	브라보 임지연!!! 성장형 그 잡채. 이것저것 다 하고 내가 보기엔 너? MVP야. 처음 구현해보는 어플일텐데 진짜 잘해줘서 고마워. ChatGPT VIP 임지연. 진짜 끝까지 포기하지 않고 노력해서 고마워. 조장으로서 부족했을 텐데 내 의견 존중해주고 집중해줘서 고마워. 고맙다 연진아 너 덕분에 이제 웬만한 오류는 놀랍지도 않아.
윤혜연	지금도 골골대고 있는 임지피티.. 제발 건강 챙기면서 열정 발휘하라고~ 너는 포박아에 최적화된 인재같다. 빅데이터 프로젝트 기간에도 모르는 거 있으면 다 설명해주고, 이번에도 앱 구현 다하고. 통계학과 아니라 사실 개발자였지? 솔직히 말해. 아무튼 열심히 프로젝트 해줘서 고맙고 남은 학기 마무리 잘하고, 앞으로 엠비티아이나 심리테스트 이런 건 안해도 될 것 같아~ 차타고 지나가면서 봐도 너는 앵부빠야. 그리고 덕분에 오디오 안 비어서 재미있었다! 너 코틀린 다음으로 결레랑 맞아용 제일 잘 해.

[윤혜연]

김현철	<p>최종발표 2번 그제 대강당이 좋은 윤혜연!! 요새 내 킹받음을 책임지는.. 맡은 일 시키는 일 전부 다 해내는 걸 보고 새삼 대단하다 느꼈어. 자소서까지 겸하면서 말이야. 빅데이터 프로젝트 첫 과제 때 발표하는 모습을 처음 봤던 게 기억나. 정말로 내가 살면서 본 발표자 중에 다섯 손가락 안이었나..? 아닌가 암튼 발표를 되게 잘했었어. 취준 병행하면서 AI 프로젝트도 같이 하기 어려웠을 텐데 프로젝트 기간 동안 고생했어!!! 너 덕분에 여기서 B반 친구들도 더 사귀어 보고, 생파도 가보고 여러모로 재밌는 경험했어. 민우민우랑 세림세림이랑도 더 친해진 거 같고. 그리고 끝나고 나서 하는 현장실습에는 꼭 붙고 떨어지면 키 5cm 작아짐~</p>
박수림	<p>발표왕 윤혜연! 비전공자임에도 혼자 다 해내는 윤혜연!! 갑자기? 짝궁이 사라져서 혼자 하느라 힘들었을텐데 투정 한 번 부리지 않고 모르더라도 척척 해내는 너의 모습을 보고 넌 정말 뭐든 잘 해낼 사람이라고 느꼈어 같은 팀인데 잘 챙겨주지 못한 것 같아서 미안한 마음도 들고 더 같이 놀아보지 못한 것 같아서 아쉽다 자소서 기간이랑 겹쳐서 바쁠텐데 팀 프로젝트에도 열심히 참여하려고 노력하는 모습이 보기 좋았어! 꼭 원하는 곳 취뽀해서 널리널리 자랑해주길!! 파이팅!!</p>
임지연	<p>우리의 발표왕 윤혜연~!! 우리 윤 킹받게 하는 재미로 아카데미 건넸잖아~자소서도 쓰고 프로젝트도 참여하느라 정신 없었을텐데 노력해줘서 고마워~ 항상 주변 사람 챙겨주고 고양이 예뻐하는 모습 기억 남을 것 같아 정말 뭐든 똑부러지고 야무지게 해내는 너라서 뭐든 잘 될 거라고 생각해~! 올해 취업 대박나고 수원에서 봐야~아아~</p>
권구택	<p>우리들의 발표왕 윤혜연! 자소서 기간이랑 겹쳐서 많이 힘들었을텐데 진짜 고생 많았어. 항상 노력하는 모습이 보기 좋았고 너 덕분에 우리 모델 성능 잘 올린 것 같아. 아 그리고 논문 정리? 너무 훌륭했으. 완벽한 발표를 위해 계속 연습하는 모습도 너무 보기 좋았어. 아 그리고 내가 본 사람 중 너가 발표 제일 잘해. 나중에 어딜가도 발표하면 될 듯. 응원할게!!!</p>