승용차 검출

인공지능 팀프로젝트

프로젝트 과정

승용차 검출

01 데이터 수집

02 데이터 라벨링

03 데이터 학습

04 학습 결과 분석

05 문제점 분석

데이터 수집

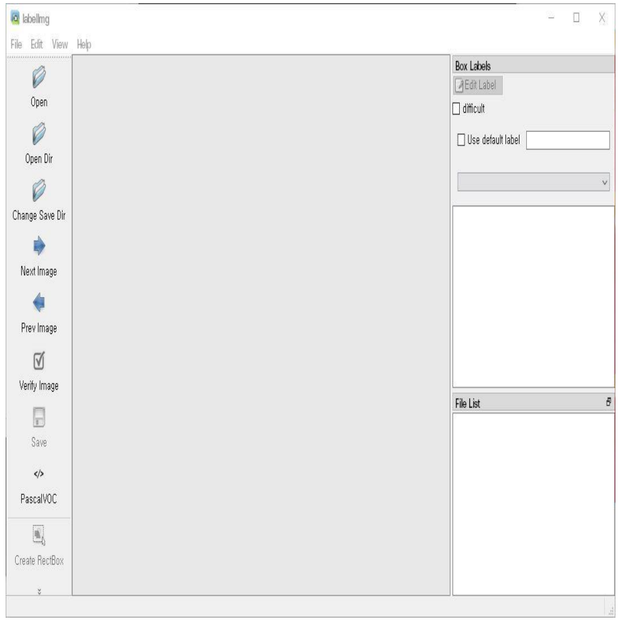
학습용 데이터는 총 900장 수집 하였습니다. 정확한 학습을 위해 여러 각도와 여러 방향의 데이터를 수집하였습니다. 데이터 수집 방법은 포털 사이트에서 직접 사진을 모으거나, 웹크롤링 방법으로 데이터를 수집하였습니다.

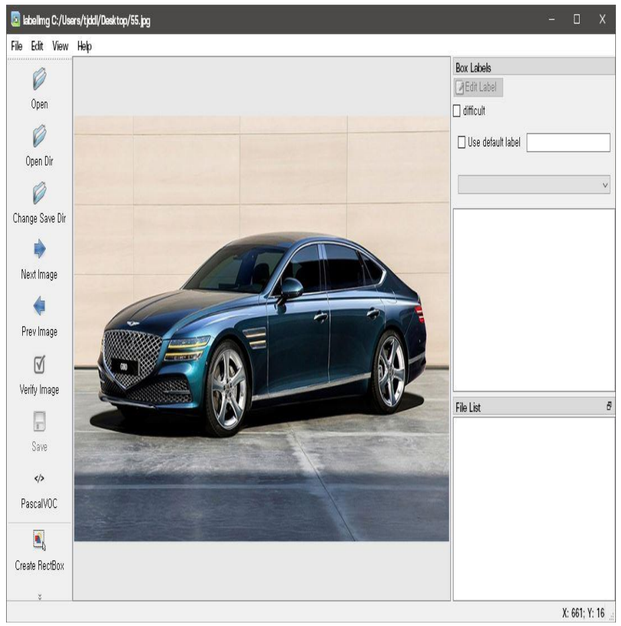
데이터 라벨링

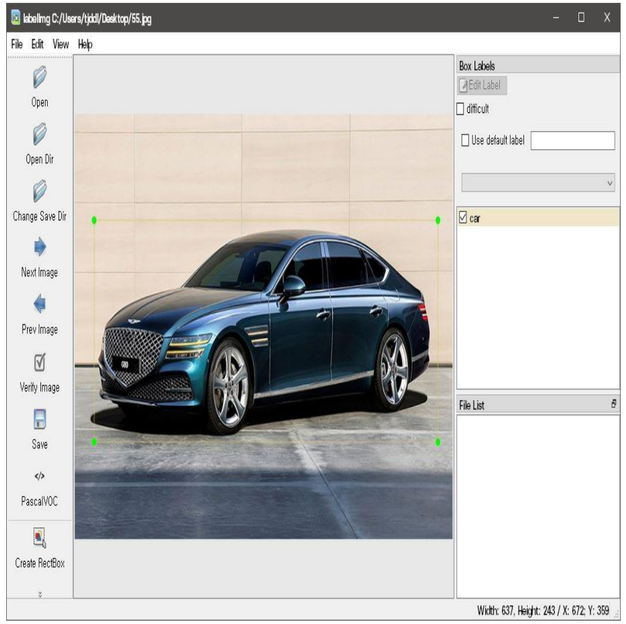
데이터 라벨링은 입력할 학습 데이터가 많기 때문에 팀원이 분담해서
각자 라벨링 툴으로 라벨링을 진행하였습니다.

프로젝트 과정

labelImg



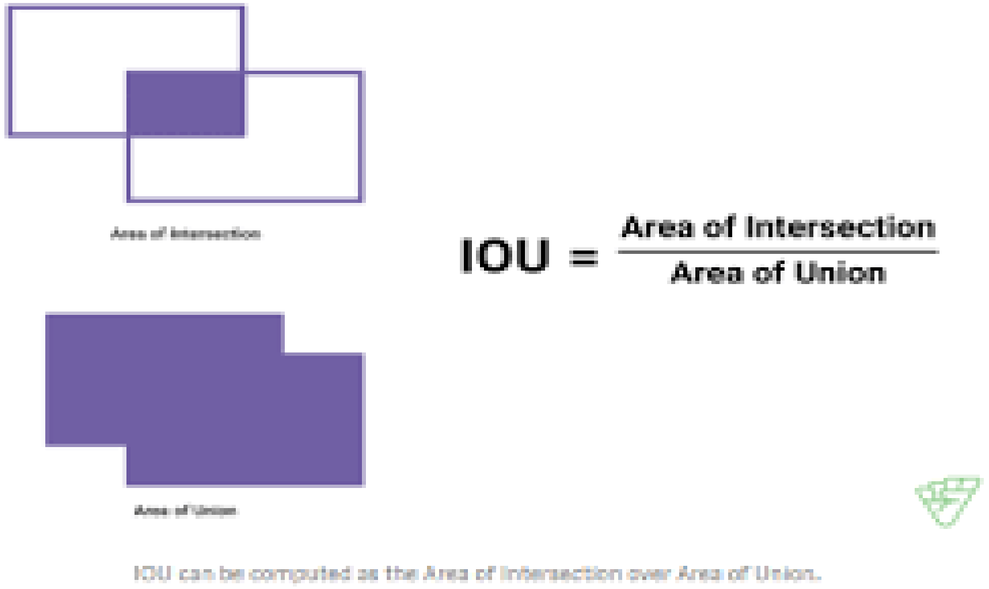




데이터 학습

데이터 학습을 위한 파이썬 코드는 수업교재의 58_02.py 예제 파일을 참고하였습니다. 입력 데이터와 출력 부분, 학습 데이터를 학습할 때의 하이퍼 파라미터를 변경하여 최적으로 학습하고 영상을 검출하는 코드를 파이썬으로 작성했습니다.

프로젝트 과정



IOU(Intersection Over Union)함수는 2개의 바운딩박스(훈련데이터의 박스, 모델의 예측 출력박스)의
교집합(intersect_area)을 합집합(Union_area)영역으로 나누어 계산합니다. 바운딩박스가 완전히 겹치면 1, 분리되어있으면 0입니다.

학습 결과 분석

데이터 학습을 마치고 인공지능이 학습한 내용으로 승용차 검출을 해보았습니다.
실행 결과와 그래프를 바탕으로 학습 결과를 분석하였습니다.

프 로 젝 트　과 정

학습 데이터 분석

위의 16개의 이미지들은 학습용 데이터들입니다.
빨간색 박스는 저희가 labelImg를 통해서
바운딩 박스를 생성했던 부분입니다.

즉, 데이터 학습을 위한 train data의 바운딩 박스를 표시한 부분이 빨간색 박스입니다.

그리고 파란색 박스는 인공지능이 학습한 내용으로 승용차라고 판단되는 부분을 보여줍니다.

16개의 결과만 임의로 추출하여 출력했을 때,
train data의 바운딩 박스와 비슷하게 파란색 사각형 표시를 하고 있음을 알 수 있습니다.

프로젝트 과정

테스트 데이터 분석

Test data를 직접 촬영하여 프로그램을 실행한 결과 입니다.

16장의 사진은 공지하신 바와 같이직접 촬영하였습니다.

라벨링하지 않은 데이터들을 넣어 train data가 아닌
사진에서도 인공지능이 승용차를 찾아낼 수 있는지
확인해봤습니다.

위와 같이 파란색 박스로 보여주는데 바운딩 박스를 표시합니다.

승용차가 가깝게 찍힌 사진들은 정확도가 높게 승용차를
찾아냈고 조금 흐릿하거나 멀리에서 찍힌 승용차들도
선명한 자동차에 비해서는 정확도가 떨어지지만 전체적인
형태는 찾아 출력 했습니다.

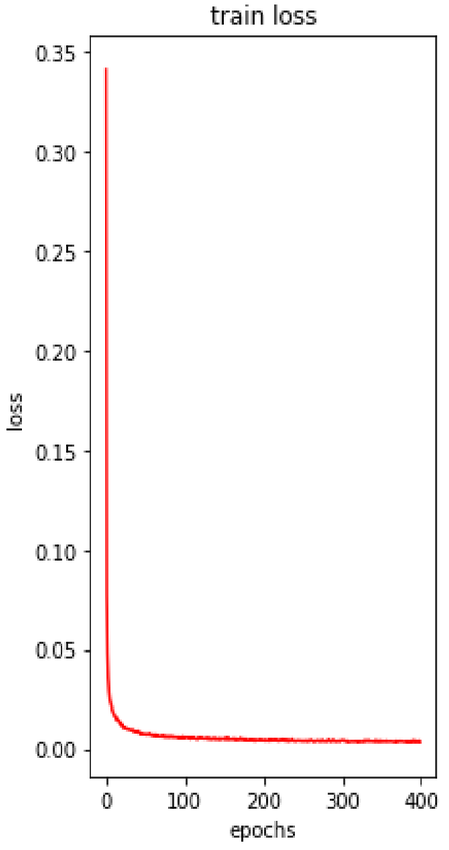
프 로 젝 트 과 정

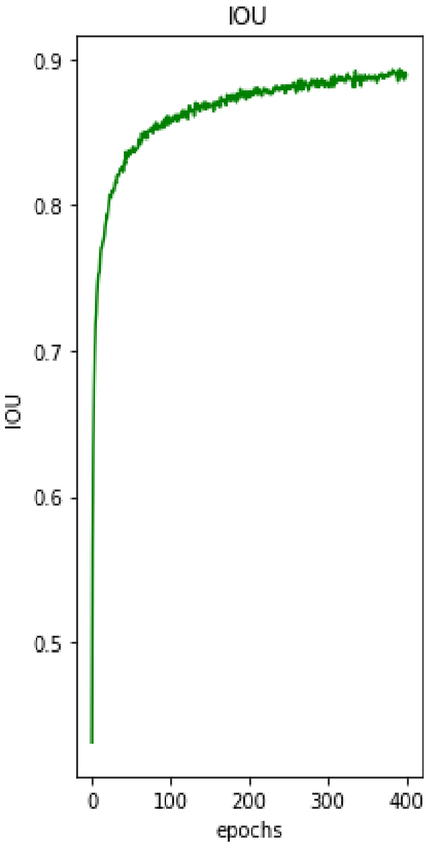
그래프 분석

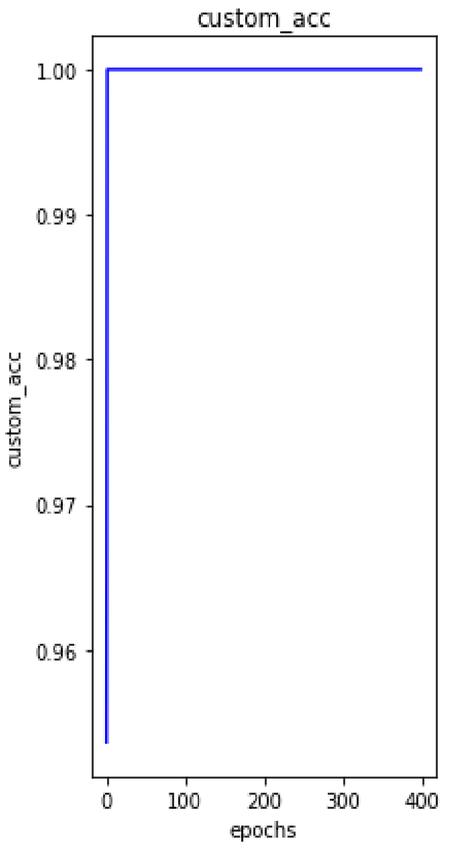
Learning_rate=0.001
epochs=400
batch_size=8
hyper parameter를 설정한 후 학습했습니다.

epochs=400에서의 Train dat evaluate
Loss=0.0047
IOU=0.8895
custom_acc(클래스별 분류 정확도)=1.0

Test data
Loss=0.2895
IOU=0.4093
정확도=1.0







그래프 분석

Epochs = 400

문제점 분석

학습 결과 분석 내용을 바탕으로 문제점을 분석하여
성능 향상 방안을 생각해보았습니다.

성능 향상 방안 1

쾌적한 학습 환경

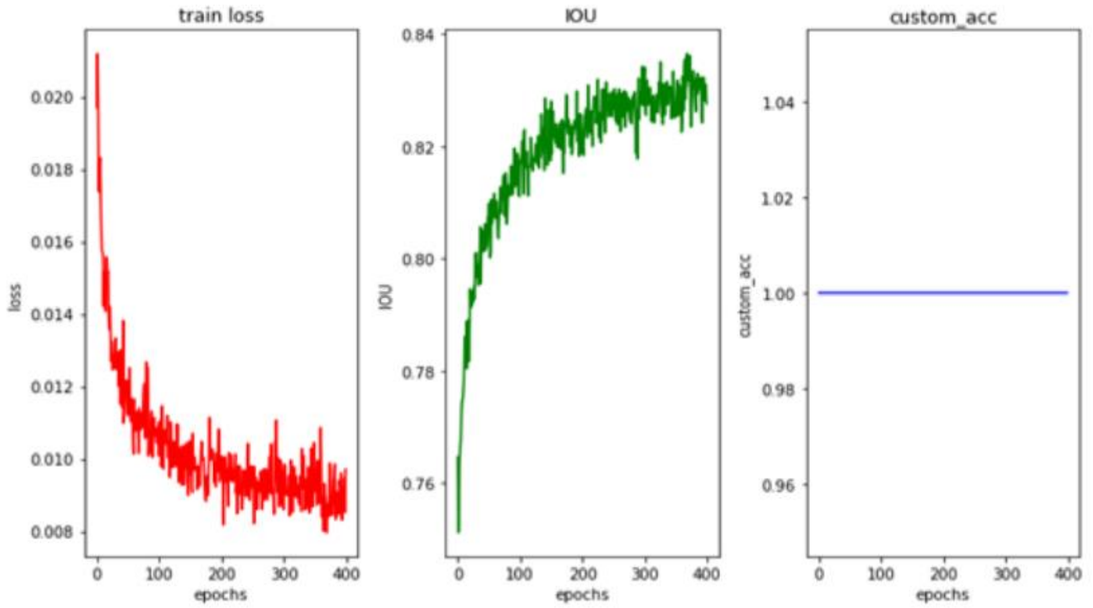
처음에 train data가 부족하여 1000장 이상의 사진을 모았었는데,
프로그램을 돌리는 과정에서 900장이상은 수용하기 힘들어했습니다.
Train data를 900장 밖에 사용하지 못하였고,
한 번 학습하는데에도 40분 정도 시간이 걸렸기 때문에,
RAM같은 하드웨어 조건이 받쳐준다면 더 많은 학습데이터로
빠르게 학습하여 성능을 향상 시킬 수 있다고 생각합니다

성능 향상 방안 2

학습률과 epochs 조절

수업시간에 배운 바와 같이 learning_rate와 epochs를 적절하게 맞춰주어야 할 것 같습니다.
Epochs 역시 컴퓨터의 문제로 큰 수까지 하지 못했지만 epochs를 증가시키면 학습 횟수가 더 증가하기 때문에, 더 높은 정확도와 더 낮은 오차를 출력했을 것입니다.

성능 향상 방안 2



Learning rate를 0.01일 때의 그래프입니다.
0.001일때와 비교했을 때, 그래프가 완만하지 않고
불안정적인 결과를 볼 수 있습니다.

성능 향상 방안 2

```
28/28 - 3s - loss: 0.0078 - IOU: 0.8729 - custom_acc: 1.0000
1/1 - 0s - loss: 0.2147 - IOU: 0.4323 - custom_acc: 1.0000
[0.21470451354980469, 0.43229383230209351, 1.0]
```

Train data의 loss도 이전의 결과보다 덜 감소한 결과를 확인할 수 있습니다.

따라서, learning rate가 낮을수록 정확한 결과와 최소한의 오차로 학습시킬 수 있다는 사실을 알 수 있습니다.

맡은 부분

제가 이 팀프로젝트에서 맡은 부분은 데이터 수집을 편하게 하기위해 크롤링 기능을 만들어 팀원들에게 주었고 같이 데이터를 수집하여 라벨링을 했습니다. 그리고 코드를 짜고 구동하는 역할을 했는데 텐서플로우를 이용하여 데이터들을 학습시켰습니다. 높은 정확도가 나오도록 코드를 분석하고 결과를 만들어내는 역할을 했습니다.

감 사 합 니 다