CNNs을 통한 이륜차 사고다발지 예측

Prediction of two-wheeled vehicle accidents through CNNs

제출일	2022, 12, 14
과목	산학캡스톤디자인2
담당교수	김판구
팀	8팀
팀구성원	이근일(20154254)
	임지훈(20164372)
	김성호(20164327)
	노순건(20171614)
프로젝트 주소	https://github.com/Nor-s/road-accident-CNN

목차

1. 서론

1.1 연구동기: 이륜차 사고 증가 및 위험성

배달을 기반으로 이루어지는 산업이 증가하고 있다. 이는 경제 전반에 많은 영향을 미치고 있으며, 교통 측면에서도 새로운 문제점들을 야기하고 있다. 특히 사고의 증가와 같은 교통 측면의 문제는 일시적인 현상이 아니고 향후 배달 기반 경제의 확산이 더욱 가속화됨에 따라 더욱 심각한 사회적 문제가 일어날 수 있다.

배달의 핵심 교통수단인 이륜차는 구매 및 유지비용 측면에서 일반 자동차에 비해 저렴하여 쉽게 접근할 수 있다는 장점을 가지고 있다. 이러한 장점으로 배달을 업으로 삼는 노동자들이 증가하였다. 이로 인해 도로를 주행하는 이륜차들이 꾸준히 증가하여 전체 교통사고 건수 감소에도 이륜차 사고는 증가하는 현상이 일어나고 있는 현황이다.[1]

이러한 현황에서 이륜차 사고의 경우 사륜차 교통사고보다 더욱 위험[2]하기에 이륜차 사고의 원인을 파악하여 사고다발지를 예측에 관한 연구할 필요가 있다.

1.2 연구목적

사고는 여러가지 요인에 의해서 발생한다. 이 연구에서는 운전자의 부주의가 일어날 수 있는 환경을 제공하는 도로의 형태와 공간적 특성을 분석하고, CNNs를 활용하여 공간적 특성이 담긴 이미지로 이륜차의 사고 위험도를 예측하여 더 안전한 도로를 만들고자 하는 것이 연구의 목적이다.

[1] 류찬희, “전체 교통사고 건수에도 이륜차 사고는 되레 증가”, 서울신문, 2022.08.03, https://www.seoul.co.kr/news/newsView.php?id=20220803500036

[2] 성다혜, “위험천만 이륜차 교통사고…치사율 사륜차의 4배”, 춘천사람들, 2020.05.18, http://www.chunsa.kr/news/articleView.html?idxno=48421

1.3 관련연구

가) 국도교차로 형태에 따른 교통사고 특성에 관한 연구(1998)[3]

해당 연구는 국도교차로 형태와 사고 발생율에 관한 연구이다. 도로의 교차각과 시거가 나쁘면 사고 발생율이 증가한다는 연구 결과를 보여주었다. 교차각은 교차로에서의 각도를 의미한다. 시거는 운전자가 도로 전방을 살펴볼 수 있는 거리를 나타낸다.

나) Combining Satelite Imagery and Open Data to Map Road Safety(2017)[4]

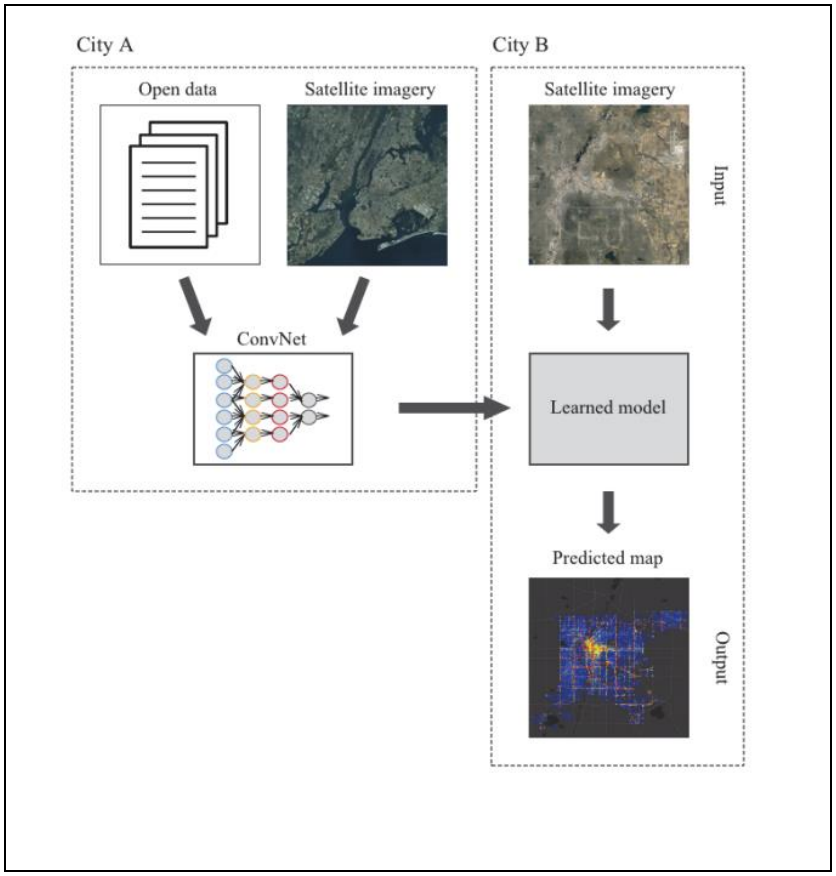

그림1.1 관련연구 나)의 시스템

해당 연구는 CityA 사고지역의 위성 이미지를 CNNs로 모델로 학습시켜 CityA 와 CityB 도로의 위험도를 예측하는 연구이다. 모델은 CityA에서 78%의 정확도를 보여주며, CityB에서 72% 정확도를 보여준다.

[3] 김홍상, 국도교차로 형태에 따른 교통사고 특성에 관한 연구, 한국과학재단, 1998

[4] Alameen Najjar, Shun'ichi Kaneko, Yoshikazu Miyanaga, Combining Satellite Imagery and Open Data to Map Road Safety, Vol. 31 No. 1 (2017): Thirty-First AAAI Conference on Artificial Intelligence, 2017

다) 이륜차사고 영향요인 종합분석을 통한 위험지역 도출 - 서울시 강남구를 중심으로(2021)[5]

이륜차 교통사고에 영향을 미치는 다양한 요인들을 활용하여 이륜차 교통사고 예측 모델을 구축하여 위험도가 높은 지역에 대한 연구이다, 교통, 공간, 인구데이터와 같은 데이터를 수집하여 OLS 회귀분석과 지리적 가중 회귀분석을 통해 예측한다.

라) Inferring high-resolution traffic accident risk maps based on satellite imagery and GPS trajectories (2021)[6]

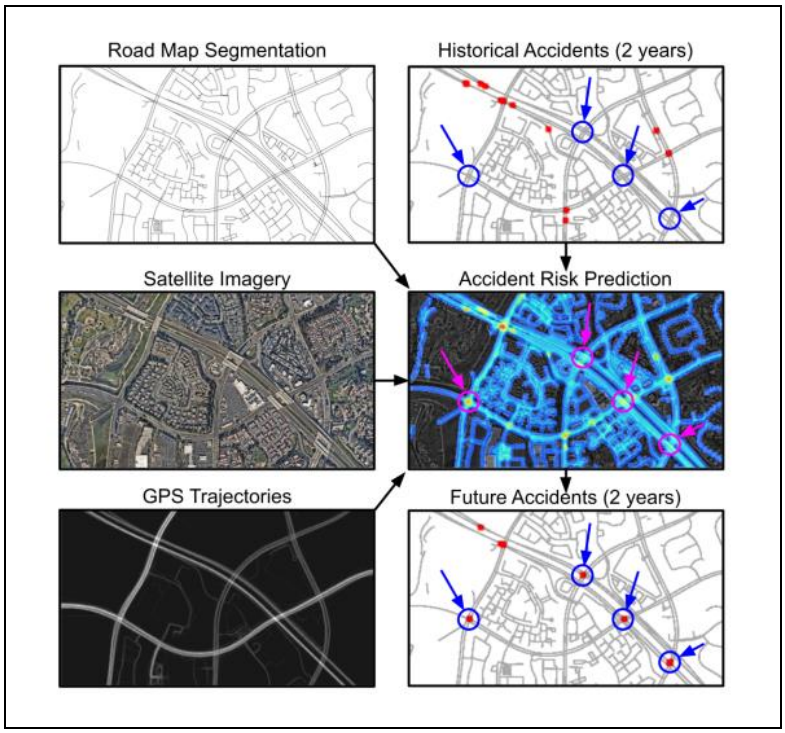

그림1.2 관련연구 라)의 end to end 시스템

위성 이미지, 도로 세그먼트, 사고 기록, GPS 추적 이미지를 통해 사고 위험도를 예측한 사고 위험 예측 맵을 생성하는 연구이다. 그림1.2에 해당하는 모델은 평가지표로 RMSE과 AP를 사용하는 회귀모델이다.

[5] 이한빈, 김휘언, 김세영, 정승환, 소현기, 김태영 and 박광헌. (2021). 이륜차사고 영향요인 종합분석을 통한 위험지역 도출 - 서울시 강남구를 중심으로. 디지털문화아카이브지, 4(2), 136-152.

[6] S. He, M. A. Sadeghi, S. Chawla, M. Alizadeh, H. Balakrishnan and S. Madden, "Inferring high-resolution traffic accident risk maps based on satellite imagery and GPS trajectories," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 11957-11965, doi: 10.1109/ICCV48922.2021.01176.

1.4 관련연구와의 차이점

그림1.3 교차각, 너비, 길이, 건물 등의 정보가 있는 위성 이미지

본 연구는 사고율에 영향을 주는 요인을 분석하는 연구 가)와는 다르게 분석이 주된 연구는 아니며, 해당 연구의 결론인 시거와 교차각 등 다양한 정보가 포함된 위성 이미지를 사용하여 CNNs 모델을 통해 도로의 위험도를 예측한다. 여기서 시거는 시야를 의미하므로 도로의 교차각과 도로의 너비 등 위성 이미지에 포함된 정보에 의해 결정된다.

그림1.4 도로가 나무에 의해 가려진 위성 이미지

이렇게 CNNs 모델을 통한 본 연구는 연구 나)와 같이 CNNs를 사용하여 수행한다. 연구 나)에서는 위성 이미지 하나만을 사용한다. 하지만, 위성 이미지는 너무 방대한 정보를 가지고 있으며, 그림1.4에서 보여지듯이 다양한 요인에 의해 도로의 특징을 추출할 수 없다. 따라서 적절하지 않은 데이터로 인해 학습율이 저하될 가능성이 높다. 본 연구에서는 이러한 문제를 해결하기 위해 다양한 시도를 하였다. 간단히 CNNs 모델 하나만 사용하는 것이 아니라, 앙상블 모델과 같이 여러 분류기가 포함된 모델, 설명력이 높은 데이터, 설명력을 높이기 위한 모델을 설계하는 등 더 높은 정확도를 얻기 위한 시도를 하였다.

연구 다)와의 차이점은 앞에서 이야기했듯이 회귀모델이 아닌 CNNs 모델의 사용, 학습데이터가 질적변수나 양적변수가 아닌 이미지 데이터를 사용하여 이륜차 사고다발지를 예측한다.

연구 라) 또한 CNNs 모델을 사용한다. 사용한 모델의 입력으로 들어가는 GPS 궤적은 교통의 밀도, 속도 및 흐름 같은 정보를 포함하고 있는 이미지이다. 이러한 데이터는 비교적 최근 교통 데이터를 수집하는 국내에서 수집하기 어렵다. 사고 기록 이미지 역시 사망 사고의 좌표데이터만 제공하는 국내에서 수집하기 어려운 데이터이다. 이러한 데이터들을 입력으로 삼기 때문에 국내 데이터에 적용하기 어려우므로 국내에 맞는 모델을 설계할 필요가 있다. 또한 연구 라)는 회귀 모델 본 연구는 분류 모델이라는 점에서 차이가 있다.

2. 본론

2.1 연구 일정

세부내용	수행기간(월)																비고
	09				10				11				12				
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	
계획수립	■	■															
자료 조사 및 자료의 유효성 검사			■	■	■												
데이터 수집 및 데이터 탐색						■	■										
모델 설계								■									
모델 학습									■	■							
모델 성능 평가 및 개선											■	■	■				

2.2 구현 환경

환경	이름
운영체제	Windows 11 (버전: 22H2)
개발언어	Python 3.8.11
인공지능 프레임워크	Tensorflow 2.9.1 (cuda 11.2)
GPU	GTX 1660 super 6.0g

주 사용 라이브러리 및 API	설명
Osmnx 1.2.2	도로 이미지 수집 (도로 네트워크)
Folium 0.12.1	도로 이미지 수집 및 시각화(위성)
Google Earth Engine API	도로 이미지 수집(높이)
Pandas 1.4.3	CSV 파일 처리
Opencv-python 4.5.3.56	이미지 읽기
Matplotlib	데이터 시각화
geopandas 0.11.0	도로 geojson 처리
Clean-fid 0.1.32	상관성 평가를 위한 FID 계산

그 외에 데이터 탐색 등 지형 데이터를 분석 및 추출하기위해 QGIS라는 지리정보 체계 응용프로그램을 사용하였다,

2.3 최종 시스템 구성도

개발 중 모델의 구조는 여러 번 수정을 거쳤다. 2.3에서는 최종 모델을 간략하게 설명한다.

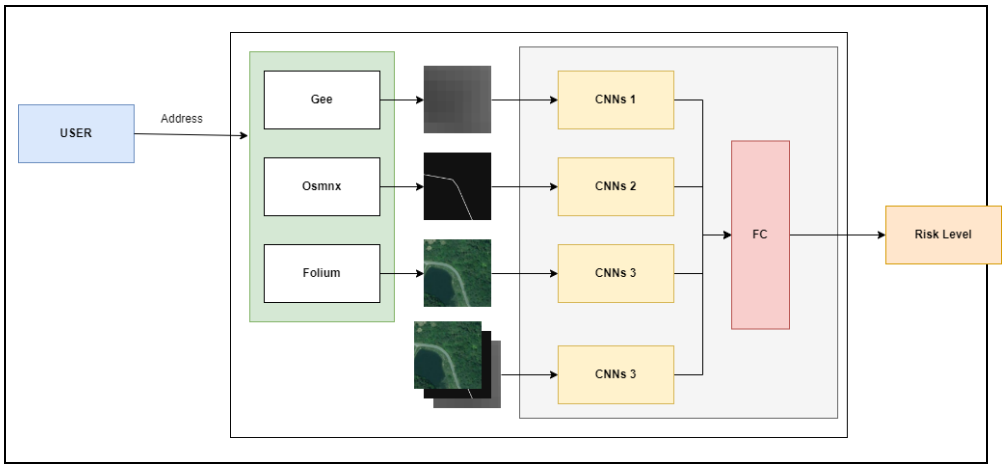

그림2.1 최종 시스템 구성도

최종 시스템 구성도는 그림 1.5와 같다. 먼저 사용자가 주소를 입력한다. 입력받은 주소를 위도 경도 좌표값으로 변환하여 Google Earth Engine API 를 사용하여 DSM(해발고도, 높이) 이미지를 얻고, OSMNX 라이브러리를 사용하여 도로 네트워크 이미지를 얻고, Folium 라이브러리를 사용하여 위성 이미지를 얻는다.

이렇게 얻은 이미지를 각각 CNNs 모델에 넣고 3개의 이미지를 쌓은 9채널 이미지를 CNNs 모델에 넣은 후, 모든 CNNs 결과물에 Dropout을 거친후 FC에 합쳐 Risk Level 을 예측한다.

2.4 구현 내용

2.4.1 데이터 수집 및 처리

가) CSV 데이터 수집

데이터 이름	제공 기관	URL
이륜차사고다발지 2016~2021	도로교통공단	https://www.data.go.kr/data/15105286/openapi.do
서울시 CCTV	Local Data	https://www.localdata.go.kr/lif/lifeCtacDataView.do?opnEtcSvcId=12_04_08_E
서울시 과속방지턱	Local Data	https://www.localdata.go.kr/lif/lifeCtacDataView.do?opnEtcSvcId=12_04_06_E
서울시 횡단보도	서울 열린데이터 광장	http://data.seoul.go.kr/dataList/OA-15554/S/1/datasetView.do
서울시 신호등	서울 열린데이터 광장	http://data.seoul.go.kr/dataList/OA-15554/S/1/datasetView.do;jsessionid=4B75D6E2529334747BAE40CC7FB013D5.new_portal-svr-11

위의 표에서 보여지듯이 국내 여러 기관이 제공하는 공공데이터를 수집하였다. 또한 이륜차 사고와 관련이 있을 것이라고 생각한 변수 CCTV, 과속방지턱, 횡단보도, 신호등과 관련한 데이터를 수집하였다.

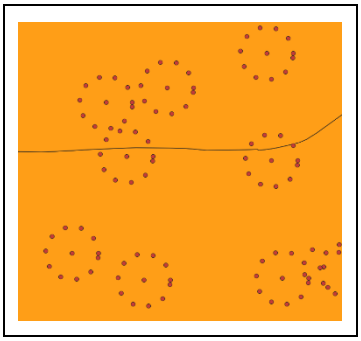

그림2.2 사고다발지

오버 샘플링

이륜차 사고다발지의 데이터는 1169개의 좌표 데이터로, 학습하기에 충분하지 못한 데이터양이라고 판단했기에 그림 1.6과 같이 사고다발지 중심점에서 100m 떨어진 지점 또한 수집하였다.

나) 이미지 데이터 수집

```
fig, ax = ox.plot_figure_ground(
                        point=point,
                        dist=self.dist,
                        filepath =fp,
                        network_type=self.network_type,
                        street_widths=self.street_widths,
                        dpi=self.dpi,
                        save=True,
                        show=False,
                        close=True,
                        )
```

코드 2.1 osmnx를 사용하여 도로 네트워크 이미지를 수집

수집한 이륜차 교통사고 다발지역 데이터의 다발지역 선정조건이 반경 100m 내 대상사고 4건 이상 발생지역이다. 따라서, dist 인자에 100을 넘겨 이미지를 수집할 때 200m X 200m 의 정사각형 지역을 수집하였다. 하지만, 수집한 후 직접 이미지를 그림2.1과 대조해본 결과 약간의 차이가 발생하였다. 이는 dist 값을 줄이며 확인했으며 최종적으로 dist 인자에 80을 넘겨주었다.

```
m = folium.Map(location=[lat, lon],
                    zoom_start=self.zoom_start,
                    tiles= self.tiles,
                    attr= self.attr
                    )
filename = f'{lat}_{lon}'
img = m._to_png()
img = np.fromstring(img, dtype = np.uint8)
img = cv2.imdecode(img, cv2.IMREAD_COLOR)
img = center_crop(img, self.reszie_wh)
```

코드 2.2 folium을 사용하여 위성 이미지를 수집

Folium의 타일을 vworld api를 통해 위성 타일로 변경하고 코드 2.2의 7번줄의 to_png()로 이미지를 수집하였다. Folium은 osmnx와 같이 범위를 지정하는 기능은 없었고, zoom의 정도로 크기를 조절할 수 있었기에 직접 osmnx에서 수집한 이미지와 대조하여 zoom을 맞추고 범위를 10번줄의 center_crop 함수로 맞추었다.

```
u_poi = ee.Geometry.Point(lon, lat)
lyon = u_poi.buffer(self.meters)  # meters
alos = ee.ImageCollection('JAXA/ALOS/AW3D30/V3_2').select('DSM')
proj = alos.first().select(0).projection()
alos = alos.mosaic().setDefaultProjection(proj)
url = alos.getThumbUrl({'min': self.min,
                                    'max': self.max,
                                    'region': lyon,
                                    'dimensions': self.dimensions,
                                    'format': 'png',
                                    'crs': self.crs,
                                    'bestEffort': True,
                                    })
```

코드 2.3 Google Earth Engine을 사용하여 높이 이미지를 수집

높이 이미지는 Google Earth Engine을 사용하여 수집하였다. self.meters의 값을 100으로 설정하여 좌표값을 중심으로하는 반경 100m를 수집하였다.

다) CSV 데이터 처리

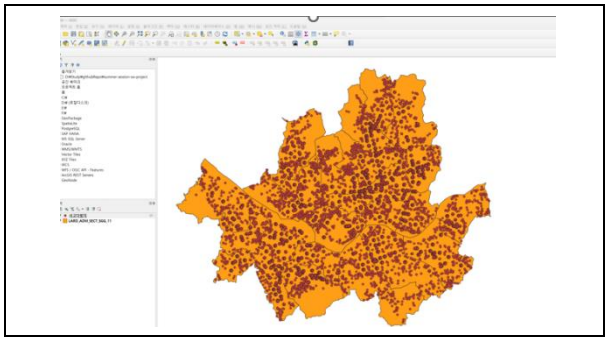

그림2.3 QGIS 화면, 사고다발지 데이터

수집한 서울시 공간데이터를 분석하기 위해 사고다발지 데이터와 cctv, 과속방지턱, 횡단보도, 신호등 데이터를 합칠 필요가 있었다. 각 시설물의 개수를 변수로 삼기위해 QGIS를 사용하였다.

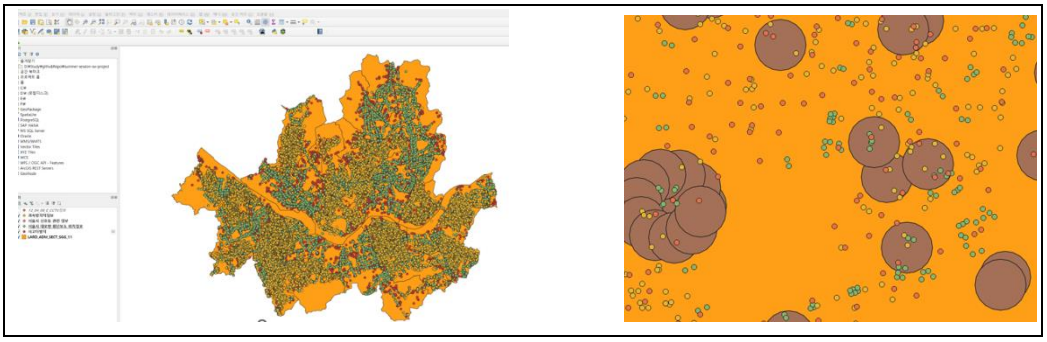

그림2.4 QGIS 화면, 사고다발지 데이터, cctv, 과속방지턱, 횡단보도, 신호등 데이터

수집한 csv 데이터를 전부 QGIS에 올린 화면은 그림 2.4와 같다. 이 상태에서 사고 다발지를 기준으로 버퍼 생성 후, 버퍼 안에 해당 변수들이 얼마나 있는지 카운팅하였다.

사고다발지FID	사망자수	중상자수	경상자수	부상신고자수	bumps	cctvs	walk	light
104688	0	19	27	7	0	0	7	69
104683	0	6	16	11	0	2	8	35
104684	0	6	13	9	0	0	12	47
104573	0	9	12	3	2	1	7	62
104773	0	12	16	3	2	2	7	46

그림2.5 합친 CSV 결과

공간데이터를 합친 결과는 그림 2.5와 같다.

2.4.2 데이터 탐색 및 변수 선택

가) 상관분석 및 도표 분석

```
1  Import seaborn as sb
2  sb.heatmap(df2.corr(),
3             annot = True,
4             cmap = 'Greens', #색상
5             vmin = -1, vmax=1
6            )
```

코드 2.4 df2 의 상관분석 결과를 히트맵으로 시각화

Matplotlib기반 패키지인 Seaborn을 사용, Pandas의 dataframe의 corr 멤버 함수를 사용하여 상관계수를 시각화 하였다.

그림2.6 상관분석 결과

2.4.1에서 합친 csv 파일을 상관분석해본 결과 횡단보도와 신호등의 상관계수가 비교적 높게 나왔다. 이는 횡단보도가 많거나 신호등이 많으면 사고가 일어날 확률이 높아진다고 해석할 수 있지만, 이러한 결과가 나온 이유를 다르게 해석하였다.

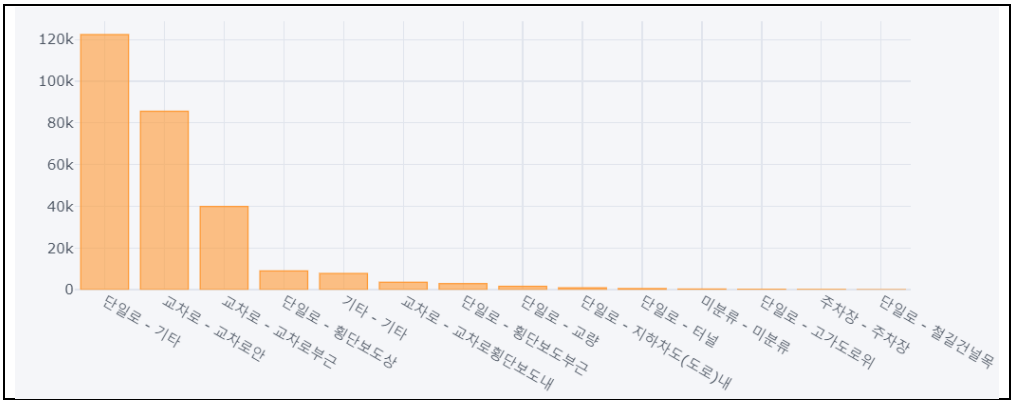

그림2.7 이륜차 사고다발지의 도로형태를 막대그래프로 시각화

횡단보도와 신호등은 교차로에 시설물의 특성상 교차로에 많이 배치되어 있다. 교차로일 경우 이륜차 사고가 많이 일어날 것이라는 가정을 세우고, 사고 다발지 데이터의 도로형태를 시각화 하였다. 기타를 제외하면 교차로에서 이륜차가 가장 많이 발생한다는 것을 확인할 수 있었다. 따라서 “위성”, “도로 네트워크” 이미지에 포함되어 있으므로 수집한 서울시 공간데이터 변수들은 학습에서 제외시켰다.

나) 수집한 이미지와 사고와 관계

수집한 이미지들이 이륜차 사고와 관계가 있는지 검증하기 위해 FID(Frechet Inception Distance)를 사용하였다. FID는 이미지의 평균과 공분산을 계산하여 확률분포로 요약한 후, 두 분포의 거리를 Frechet distance를 사용하여 측정하는 방식이다. 여기서 확률분포는 이미지의 특징을 설명한다.

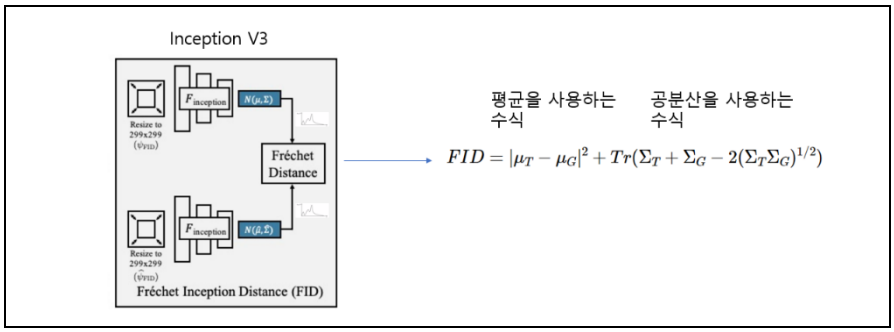

그림2.8 FID[7]

FID 의 동작 방식은 다음과 같다. 이미지 셋에서 이미지를 요약하기 위한 사전 학습된 Inception V3 모델로 확률분포를 형성한 후, 그 평균과 공분산을 이용하여 거리를 계산한다. 여기서 평균을 사용하는 수식은 얼마나 분포가 비슷한 지를 나타내고, 공분산을 사용하는 수식은 특징들 간의 통계가 얼마나 비슷한 지를 나타낸다.

```
1  from cleanfid import fid
2  score = fid.compute_fid(random_80, accident_80, num_workers=0)
3  print(score)
```

코드 2.5 오픈소스 clean-fid를 사용하여 fid 계산

FID 는 오픈소스 clean-fid를 통해 계산하였다. Clean-fid 와 다른 fid와의 차이점은 안티에일리어싱에 있다. Clean-fid는 Inception v3 모델로 확률분포를 형성하기 전에 다운 샘플링하는 과정에서 안티에일리어싱을 통해 이미지의 손실을 최소화한다.

이렇게 계산된 결과는 다음과 같다.

사고 - 비 사고	사고 - 전체	비 사고 - 전체
51.5294	34.8313	5.1256

전체는 사고 와 비 사고를 포함하는 국내 도로에서 무작위적으로 샘플링한 이미지 집합이다. 사고와 비 사고는 “이륜차 사고”의 조건으로부터 분류되어 수집된 이미지 집합이다. 만약, 사고라는 조건으로부터 수집한 ‘사고’의 이미지 집합의 분포가 전체 이미지 집합의 분포와 FID 가 크다면 사고라는 조건이 이미지 집합에 영향을 끼쳤다고 볼 수 있다.

[7] 이미지 출처: https://github.com/GaParmar/clean-fid

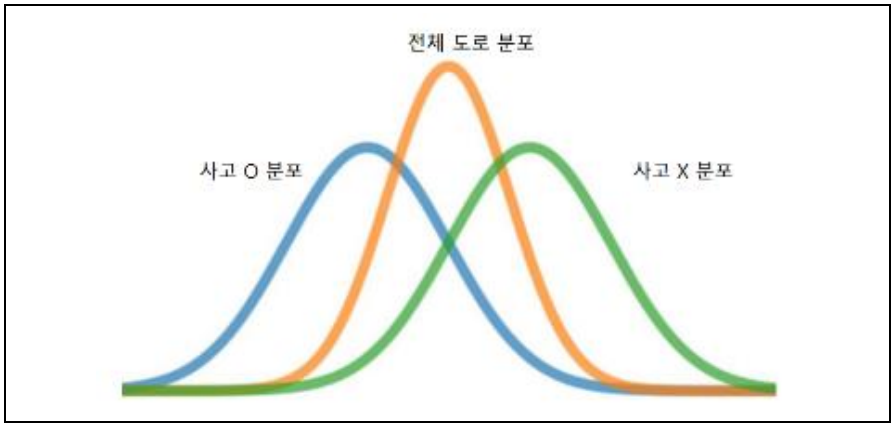

그림2.9 FID

FID 계산 결과를 시각화한 그림 2.9를 보면 사고라는 조건이 이미지 집합에 영향을 준 것을 확인할 수 있다. 이는 사고와 사고 도로 이미지 간에 상관관계가 있다는 것을 의미한다. 그러므로 이들은 변수로 사용하기에 적절하다.

2.4.3 1 차 모델 학습 및 결과

먼저 OSMNX로 수집한 도로 네트워크 이미지 수집한 이미지들을 2개의 클래스로 나누고 간단한 모델을 학습시켜보았다.

가) 라벨링: 2개의 클래스

- 사고 다발지역
- 비 사고다발지역

나) 1차 모델 학습

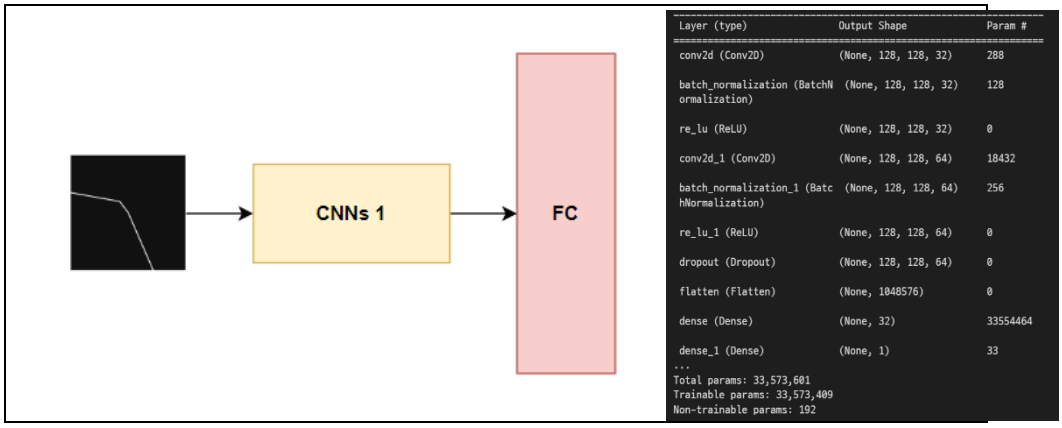

그림2.10 클래스 2개, 이미지 한 개가 인풋으로 들어가는 모델

특징을 추출하는 Conv 층과 분류하는 Dense 층을 간단하게 구성하여 모델을 학습시켰다.

다) 결과 (30 epoch)

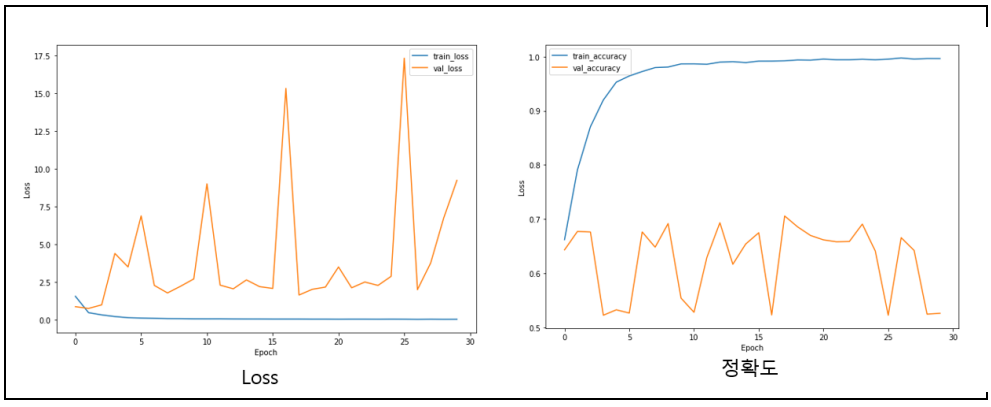

그림2.11 1차 모델학습 결과

	Train	Validation	Test
Accuracy	99.2%	70.6%	69.8%

99%의 정확도를 보이는 Train 데이터에 과적합 문제가 발생하였다.

2.4.4 2 차 모델 학습 및 결과

과적합 문제를 해결하기 위해 tensorflow의 ImageDataGenerator 함수를 사용한 데이터 증강 및 Max Polling Layer 와 Global Average Pooling Layer를 추가하였다.

가) 라벨링: 2개의 클래스

- 사고 다발지역
- 비 사고다발지역

나) 2차 모델 학습

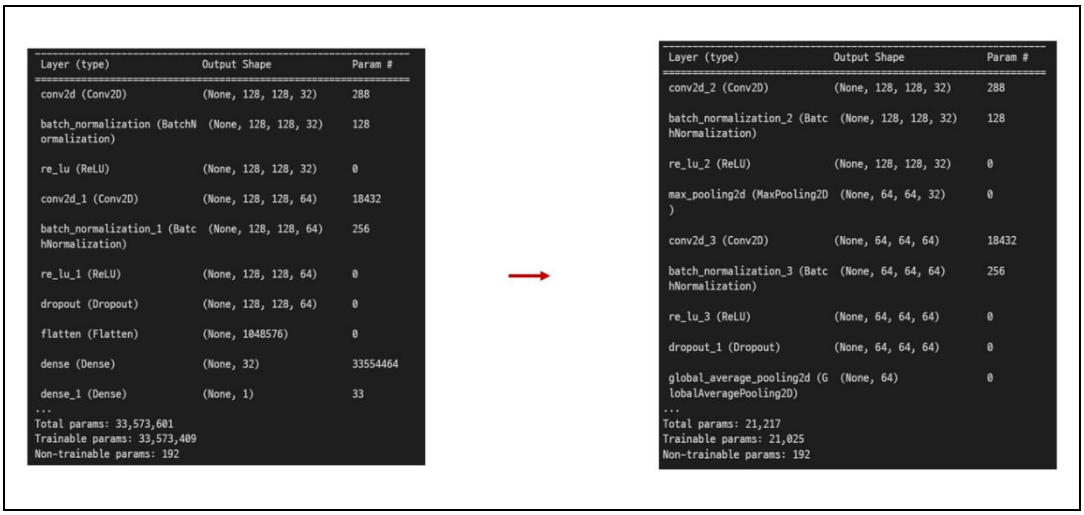

그림2.12 33,573,601에서 21,217 로 파라미터의 수를 줄임

특징을 추출하는 Conv 층과 분류하는 Dense 층을 간단하게 구성하여 모델을 학습시켰다.

다) 결과 (30 epoch)

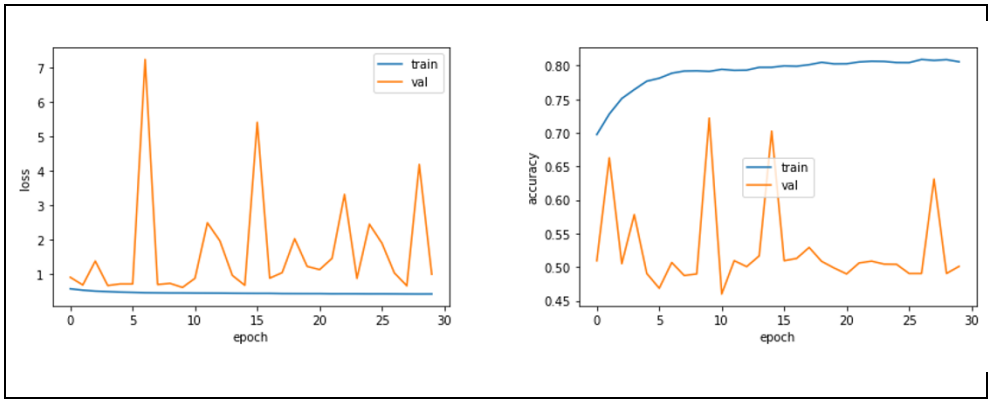

그림2.13 2차 모델 학습 결과

1차 모델학습과 같이 Train 데이터에 과적합 문제가 발생하였다.

2.4.5 3 차 모델 학습 및 결과

모델 구조 자체에 문제가 있다고 생각하여, 사전학습 모델인 VGG16 을 사용하여 학습하였다. VGG16모델은 깊은 네트워크를 사용하여 성능을 향상시킨 모델로 유명하다. 먼저 VGG16 모델로 학습시키고 결과를 확인한 후 과적합 문제가 어느정도 해결되어 사전 학습 모델의 일부분만 학습시키는 파인 튜닝을 한 후 학습을 시킨 결과이다.

가) 3차 모델 학습

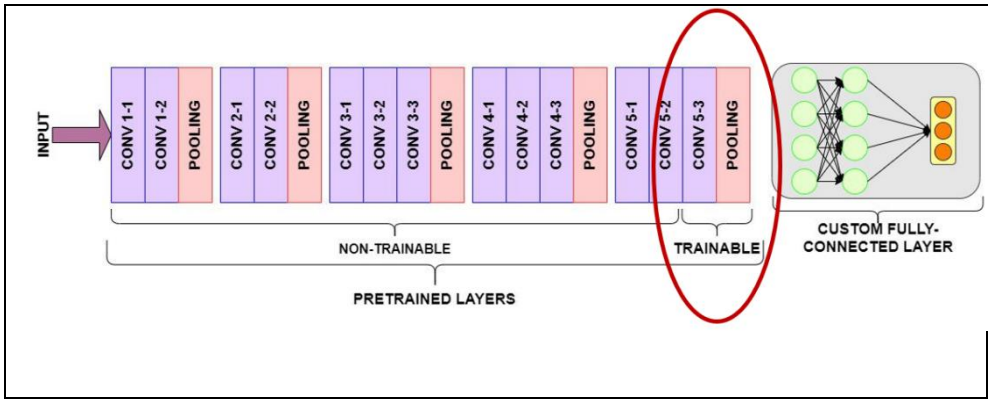

그림2.14 3차 모델 [8]

CNNs 모델을 VGG16 사전 모델로 교체한 후 모델을 학습시켰다.

나) 결과 (30 epoch)

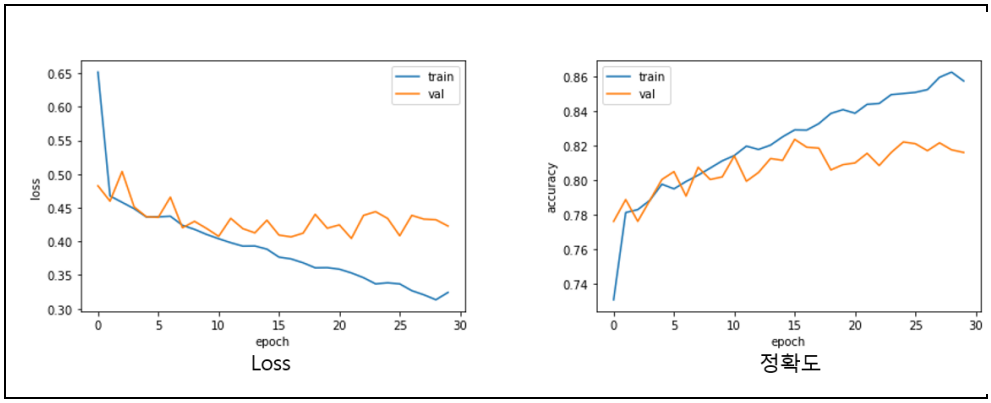

그림2.15 3차 모델 학습 결과

	Train	Validation	Test
Accuracy	82.91%	82.36%	83.66%

모델 학습 결과 2개의 클래스를 분류 정확도 83.66%로 유의미한 정확도가 나왔다. 또한 이전에 있었던 과적합 문제가 해결되어 Train 데이터와 함께 학습이 진행된 결과를 확인할 수 있었다.

[8] 이미지 출처: https://neurohive.io/en/popular-networks/vgg16/

2.4.6 4 차 모델 학습 및 결과

2개의 클래스를 분류하는 3차 모델이 유의미한 결과를 보여주었기에 하나의 클래스를 더 추가하여 동일한 모델에 인풋과 아웃풋 층만 변경한 후 학습시켰다.

가) 라벨링: 3개의 클래스

- 1: 비 사고 다발지역
- 2: 사고 다발지역 약 ~ 중
- 3: 사고 다발지역 강

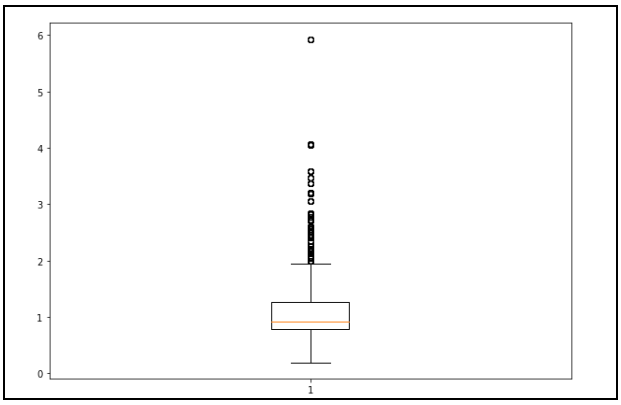

그림2.16 사고 다발지 박스 플롯

2번 클래스와 3번 클래스의 기준은 사고다발지의 3사분면으로 설정하였다. 위험도가 더 높은 사고는 더 적게 일어나며, 너무 기준 값을 높게 잡으면 데이터 양이 부족하고, 너무 낮게 잡으면 위험도를 적절하게 나타낼 수 없을 것이라 생각하였기 때문이다.

나) 결과 (100 epoch)

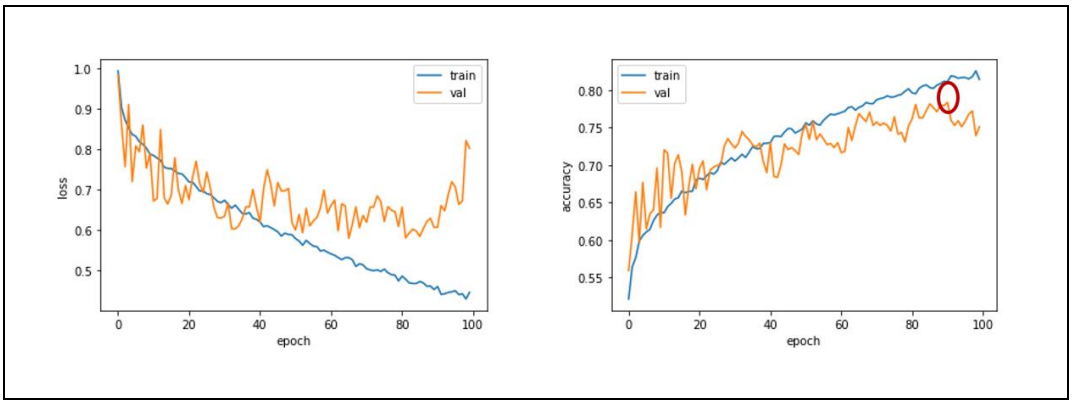

그림2.17 4차 모델 학습 결과

	Train	Validation	Test
Accuracy	81.13%	78.38%	77.17%

2개의 클래스를 분류하는 3차 모델보다 정확도가 떨어진 것을 확인할 수 있었다.

2.4.7 최종 모델 학습 및 결과

클래스를 하나 더 추가하였기에 4차 모델의 성능이 3차 모델의 정확도보다 떨어졌다. 이를 해결하기 위해 앞서 수집한 위성 사진, 도로 네트워크 사진, 높이 사진 총 3가지 이미지를 사용하여 학습시켰다.

위성사진은 도로에 관한 전체 적인 정보와 횡단보도와 같은 노면표시 또는 방지턱, 주위 건물들 등의 정보를 포함한다. 높이 사진은 위성사진에 포함되지 않은 경사와 같은 정보를 포함한다. 도로 네트워크 사진은 도로만을 표시하는 이미지로 위성사진에서 발생할 수 있는 구름 또는 나무로 의한 데이터 품질 저하를 방지하는 정보를 포함한다.

가) 최종 모델 학습

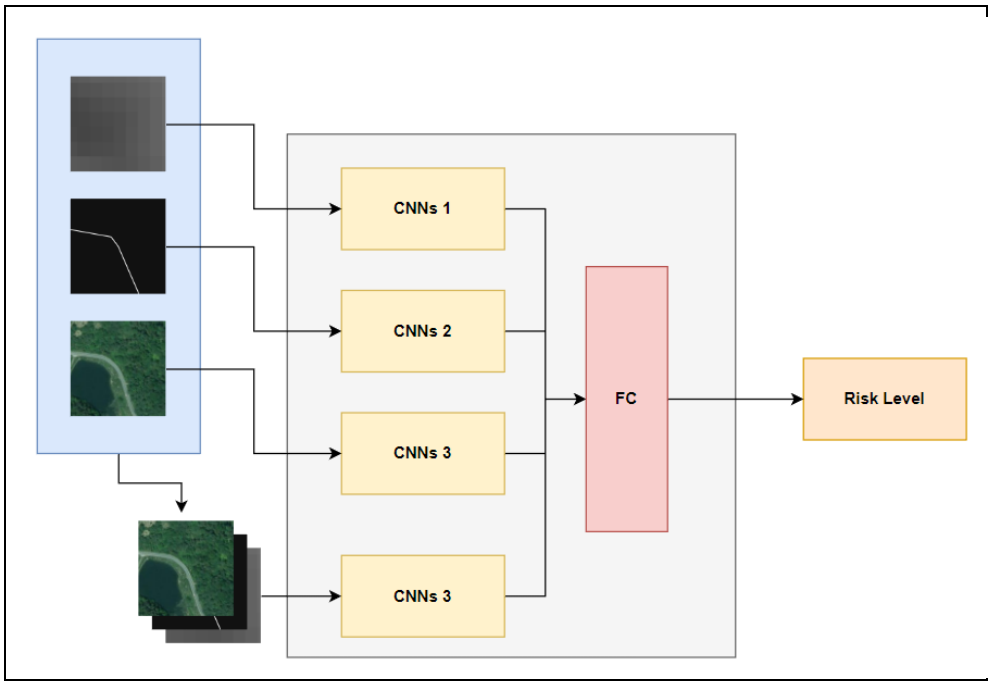

그림2.18 최종 모델 구조 4개의 CNN을 통해 위험 정도(클래스 3개)를 예측한다.

입력 데이터가 증가하여 모델 구조를 변경할 필요가 있었고, 여러 분류기를 사용하여 정확도를 올리는 앙상블 기법을 모방하여 각각의 이미지만을 보고 분류하는 CNN 모델 3개와 3개의 이미지를 합쳐서 종합적으로 특징을 추출하여 분류하는 CNN 모델 하나 총 4개의 CNN 모델을 사용하였다. 또한 각각의 CNN 모델을 합치는 층 바로 전에 Drop 층을 두어 과적합을 방지하도록 모델을 구성하였다.

```
def model_maker(num_class, input_shape=(224, 224, 3)):
    base_model = tf.keras.applications.MobileNetV2(…
    base_model2 =tf.keras.applications.MobileNetV2( …
    base_model3 =tf.keras.applications.MobileNetV2( …
    base_model._name = 'mobilenetv2_1'
    base_model2._name = 'mobilenetv2_2'
    base_model3._name = 'mobilenetv2_3'

    for layer in base_model.layers[:130]:
        layer.trainable = False
    for layer in base_model2.layers[:130]:
        layer.trainable = False
    for layer in base_model3.layers[:130]:
        layer.trainable = False
    sat_input = tf.keras.Input(shape=input_shape)
    sat_custom_model = base_model(sat_input)
    sat_custom_model = tf.keras.layers.GlobalAveragePooling2D()
                                                    (sat_custom_model)
    sat_custom_model = tf.keras.layers.Dropout(0.2)(sat_custom_model)
    sat_custom_model = tf.keras.layers.Dense(256,
                        activation='relu')(sat_custom_model)
    dem_input = tf.keras.Input(shape=input_shape)
    dem_custom_model = base_model2(dem_input)
    dem_custom_model = tf.keras.layers.GlobalAveragePooling2D()
                                                    (dem_custom_model)
    dem_custom_model = tf.keras.layers.Dropout(0.2)(dem_custom_model)
    dem_custom_model = tf.keras.layers.Dense(256,
                        activation='relu')(dem_custom_model)
    road_input = tf.keras.Input(shape=input_shape)
    road_custom_model = base_model3(road_input)
    road_custom_model = tf.keras.layers.GlobalAveragePooling2D()
                                                    (road_custom_model)
    road_custom_model =tf.keras.layers.Dropout(0.2)(road_custom_model)
    road_custom_model = tf.keras.layers.Dense(256, activation='relu')
                        (road_custom_model)
    sum_custom_model = tf.keras.layers.concatenate
                          ([sat_input, road_input, dem_input])
    sum_custom_model = tf.keras.layers.Conv2D(32, (3, 3),
                        activation='relu')(sum_custom_model)
    sum_custom_model = tf.keras.layers.Conv2D(64, (3, 3),
                        activation='relu')(sum_custom_model)
    sum_custom_model =  tf.keras.layers.MaxPool2D(pool_size=(2,
                        2))(sum_custom_model)
    sum_custom_model = tf.keras.layers.Conv2D(128, (3, 3),
                        activation='relu')(sum_custom_model)
    sum_custom_model =  tf.keras.layers.MaxPool2D(pool_size=(2,
                        2))(sum_custom_model)
    sum_custom_model = tf.keras.layers.Conv2D(256, (3, 3),
                        activation='relu')(sum_custom_model)
    sum_custom_model = tf.keras.layers.GlobalAveragePooling2D()
                                                    (sum_custom_model)
    sum_custom_model = tf.keras.layers.Dropout(0.2)(sum_custom_model)
    sum_custom_model = tf.keras.layers.Dense(256, activation='relu')
                                                    (sum_custom_model)
    custom_model = tf.keras.layers.concatenate([sat_custom_model,
          dem_custom_model, road_custom_model, sum_custom_model])
    custom_model = tf.keras.layers.Dense(64, activation='relu')
                                                    (custom_model)
    custom_model = tf.keras.layers.Dense(32, activation='relu')
                                                    (custom_model)
    predictions = tf.keras.layers.Dense(num_class,
                        activation='softmax')(custom_model)
    return tf.keras.Model(inputs=[sat_input, dem_input, road_input],
                          outputs=predictions)
```

코드 2.6 최종 모델 코드

나) 결과 (100 epoch)

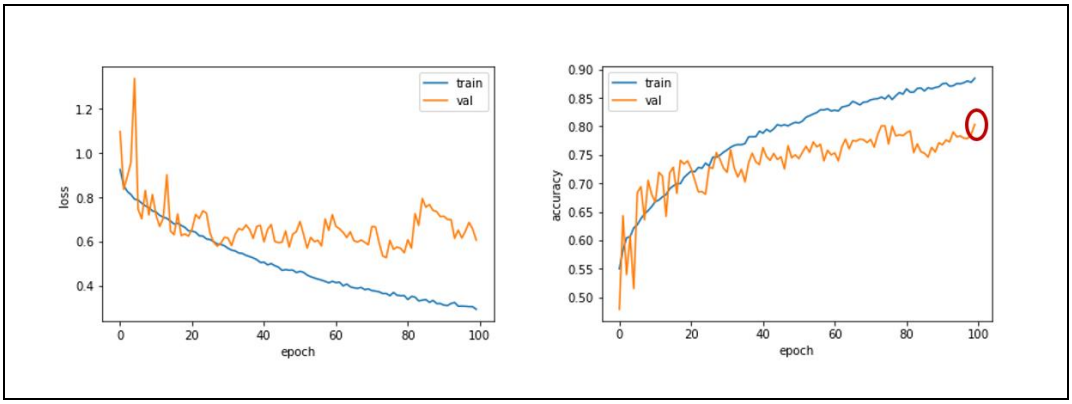

그림2.19 최종 모델 학습 결과

	Train	Validation	Test
Accuracy	88.47%	80.34%	81.91%

5차 모델의 정확도 보다 약 3% 증가한 것을 확인할 수 있었다.

다) 결과 (300 epoch)

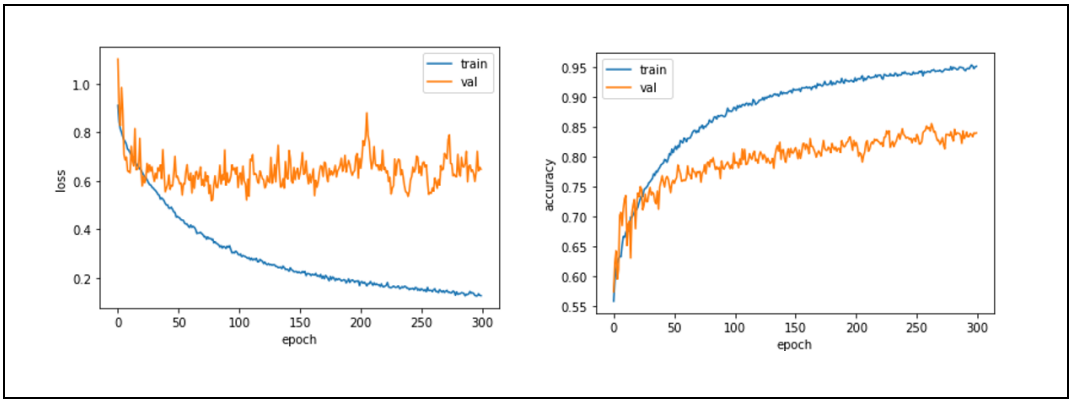

그림2.20 4차 모델 학습 결과

	Train	Validation	Test
Accuracy	94.52%	85.58%	84.37%

100 epoch 에서 300 epoch로 증가시켜 학습을 진행하였다. 그 결과 84%의 정확도를 얻을 수 있었다.

2.4.8 시각화: 광주광역시 이륜차 사고다발지 예측

84%의 정확도의 모델을 사용하여 광주광역시 도로의 이륜차 사고다발지 예측을 QGIS, Folium 라이브러리, GeoPandas를 사용하여 해보았다.

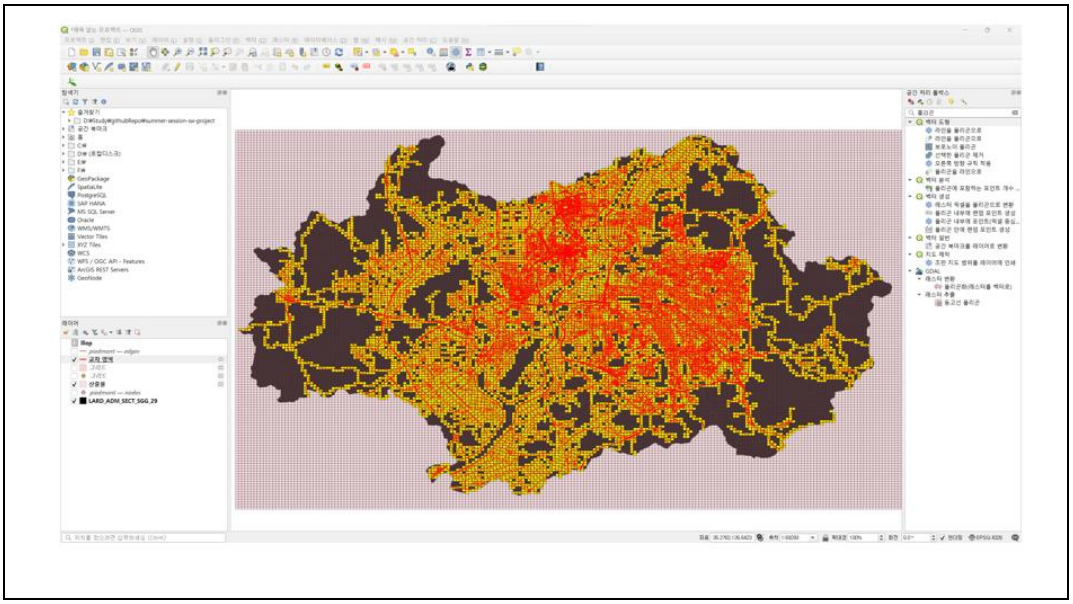

그림2.21 QGIS를 사용하여 도로 데이터 추출

먼저, OpenStreetMap의 Road 데이터를 사용하여 도로영역을 QGis 로 200m X 200m 그리드 데이터로 추출하였다.

```
df.head()
```

Python

id	left	top	right	bottom	geometry	center	lon	lat
22847.0	1.412851e+07	4.189621e+06	1.412871e+07	4.189821e+06	MULTIPOLYGON (((126.91951 35.18993, 126.91951 ...	[(126.91860894594058, 35.18919202300463)]	126.918609	35.189192
22846.0	1.412851e+07	4.189821e+06	1.412871e+07	4.190021e+06	MULTIPOLYGON (((126.91951 35.19139, 126.91951 ...	[(126.91860894594059, 35.19066031257187)]	126.918609	35.190660
22845.0	1.412851e+07	4.190021e+06	1.412871e+07	4.190221e+06	MULTIPOLYGON (((126.91951 35.19286, 126.91951 ...	[(126.91860894594056, 35.19212857560577)]	126.918609	35.192129
22819.0	1.412851e+07	4.195221e+06	1.412871e+07	4.195421e+06	MULTIPOLYGON (((126.91951 35.23103, 126.91951 ...	[(126.91860894594058, 35.23029409969932)]	126.918609	35.230294
22818.0	1.412851e+07	4.195421e+06	1.412871e+07	4.195621e+06	MULTIPOLYGON (((126.91951 35.23250, 126.91951	[(126.91860894594059, 35.23176164615013)]	126.918609	35.231762

그림2.22 추출한 도로 데이터

추출한 데이터를 GeoPandas에서 읽어 그리드의 중심좌표를 얻은 후, 2.4.1에서 다룬 이미지 수집 코드를 사용하여 광주 도로의 이미지를 수집한 후, 코드 2.7의 model.predict 함수를 사용하여 각 그리드의 위험도를 예측하였다.

```
for idx, row in df3.iterrows():
    lat, lon = row['lat'], row['lon']
    filename = f'{lat}_{lon}'
    imgs = get_image(filename)
    if len(imgs) != 3:
        print(-1)
        latlon_dict[filename] = 0
    else:
        pred = model.predict(imgs)
        latlon_dict[filename] = np.argmax(pred, axis= 1)[0]
        print('Predicted: ', latlon_dict[filename])
```

코드 2.7 모델을 사용하여 해당 좌표의 위험 정도 (사고다발지)를 예측

```
center = [35.1595454, 126.8526012]
m = folium.Map(location=center, zoom_start=12)
folium.Choropleth(
    geo_data='./data/final_GW_road_grid.geojson',
    data = df3,
    columns=['id' , 'class'],
    key_on = 'feature.properties.id',
    fill_color='Reds',

).add_to(m)
m
```

코드 2.8 Folium의 Choropleth 함수를 사용하여 예측한 위험도 시각화

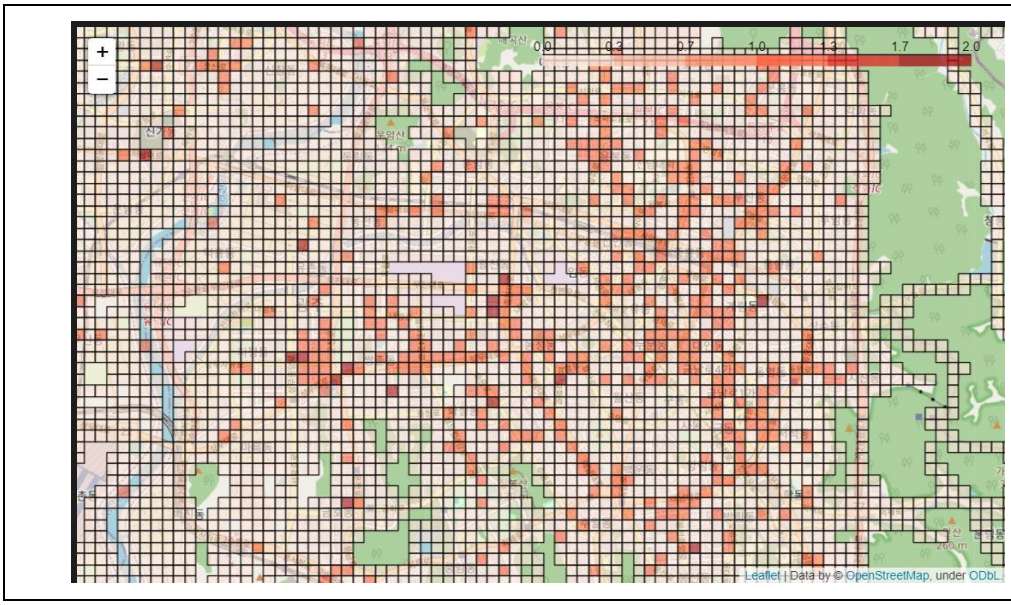

그림2.23 광주 광역시 이륜차 사고 다발지 예측 결과 시각화

코드 2.8의 Choropleth 함수를 통해 코드 2.7로 얻은 위험도를 시각화 한 결과물이 그림 2.23이다.

3. 결론

3.1 결론

프로젝트	정확도	클래스 수	해상도
Combining Satellite Imagery and Open Data to Map Road Safety	78%	3	Google Static Maps API 256×256 pixels, zoom levels (18, 19, and 20).
RoadAccidents	78%	2	200m x 200m
NYC_Traffic_Safety_Project	74.29%	5	?
Traffic-Accidents	82%	2	?

본 연구에서 위성 이미지와 높이 이미지인 DSM, 그리고 도로를 나타내는 도로 구분(도로 네트워크) 이미지를 사용하여 모델을 학습할 수 있었고 이륜차 사고다발지를 예측할 수 있었다.

정확도는 300에포크에서 84%로 유의미한 값을 얻어낼 수 있었다. 깃허브에서 찾을 수 있었던 다른 해외의 오픈소스 프로젝트와 비교해보았을 때 정확도가 본 연구의 결과가 더 높은 것을 확인할 수 있었다. (비교한 해외 오픈소스 프로젝트들은 표지에 있는 프로젝트 주소, README.md 파일에서 확인할 수 있다.)

3.2 기대효과

본 연구는 이륜차 사고다발지의 지형적 특징을 이미지로 학습시켜 지형 및 도로의 특성이 담긴 위성사진과 위험도를 예측하였다. 도로 공사 및 도로 정비를 계획할 때 현재 설계한 도로가 해당 지형에서 안전한지 판단 내릴 수 있는 지표로 사용할 수 있는 기대효과가 있다.

3.3 한계 및 향후 연구계획

본 연구의 한계는 데이터에 있다. 수집한 이륜차 사고 좌표데이터는 본론 구현내용에서 이야기하였듯이 약 1,000여개의 데이터이다. 이는 인공지능 모델을 학습하기 위해 충분하지 않은 데이터 수이다. 데이터를 증강하기 위해 사고 다발지의 중심점에서 100m 떨어진 지점 또한 사고다발지역으로 삼았다. 최종적으로 10,000개의 사고 다발지역과 10,000개의 비 사고 다발지역을 수집하였다. 이러한 오버 샘플링으로 인해 과적합 문제가 발생했을 가능성이 높다.

또 하나의 한계점은 라벨링에 있다. 라벨링을 진행하면서 3사분위수를 기준으로 사고 다발지역의 라벨을 심각한 지역과 덜 심각한 지역으로 나누었다. 이러한 기준은 검증되지 않았으므로 적절히 학습되지 않았을 가능성이 있다.

이러한 한계점들을 극복하기위해 향후 공공데이터에서 사고 데이터의 좌표 변수를 제공하지 않으므로 해외 데이터를 활용하여 데이터의 수를 증강시켜 연구할 계획이다. 또한 라벨링을 하는데 있어 다양한 클러스터링 방법을 적용하여 통계적인 방법으로 적절한 기준 값을 선정하여 연구할 계획이다.

4. 부록

4.1 참고 자료

1. 류찬희, “전체 교통사고 건수에도 이륜차 사고는 되레 증가”, 서울신문, 2022.08.03,
2. 성다혜, “위험천만 이륜차 교통사고…치사율 사륜차의 4배”, 춘천사람들, 2020.05.18,
3. 김홍상, 국도교차로 형태에 따른 교통사고 특성에 관한 연구, 한국과학재단, 1998
4. Alameen Najjar, Shun'ichi Kaneko, Yoshikazu Miyanaga, Combining Satellite Imagery and Open Data to Map Road Safety, Vol. 31 No. 1 (2017): Thirty-First AAAI Conference on Artificial Intelligence, 2017
5. 이한빈, 김휘언, 김세영, 정승환, 소현기, 김태영 and 박광헌. (2021). 이륜차사고 영향요인 종합분석을 통한 위험지역 도출 - 서울시 강남구를 중심으로. 디지털문화아카이브지, 4(2), 136-152.

4.2 팀원 수행 작업

가) 팀원 명단

연번	학과	학년	학번	성명	연락처	역할
1	컴퓨터공학과	4	20154254	이근일	010-9885-5448	팀 장
2	컴퓨터공학과	4	20154323	임지훈	010-9198-5548	팀 원
3	컴퓨터공학과	4	20164327	김성호	010-9179-7863	팀 원
4	컴퓨터공학과	4	20171614	노순건	010-2811-8149	팀 원

나) 수행 작업

이름	작업 내용
이근일	시각화 코드 작성, ppt 작성, 발표, 자료 조사, 제안서 작성
임지훈	EDA, 자료 조사
김성호	EDA, 자료 조사
노순건	이미지 데이터 수집, 모델 설계, 모델 학습, 자료 조사, 관련 연구 조사, 보고서 작성