침수피해 선제적 대응을 위한 우수받이 위치 선정 프로그램 개발: 딥러닝 시뮬레이션

[Connected-Minds]

김지민, 손영신, 연승우, 오준석, 한주윤

지도교수: 정찬기 교수님

목차

팀 소개 > 주재 소개 > 활동 내용 v

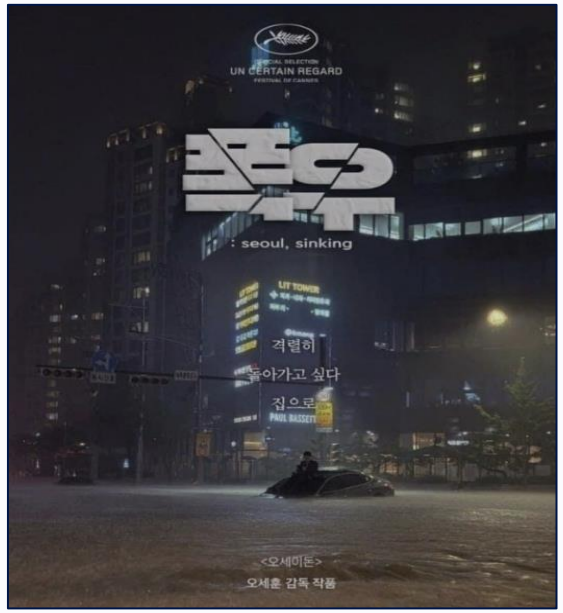
데이터 수집 < 딥러닝 코드 구현 < 시각화

팀 소개

팀장 오준석	팀원 김지민	팀원 손영신	팀원 연승우	팀원 한주윤
국방디지털융합학과 22	국방디지털융합학과 22	국방디지털융합학과 22	국방디지털융합학과 22	국방디지털융합학과 22
프로젝트 총괄	데이터 수집	데이터 수집	수식 정리	데이터 수집
지도학습 구현	데이터 정리	수식 정리	지도학습 구현	데이터 전처리

주제 소개

배경

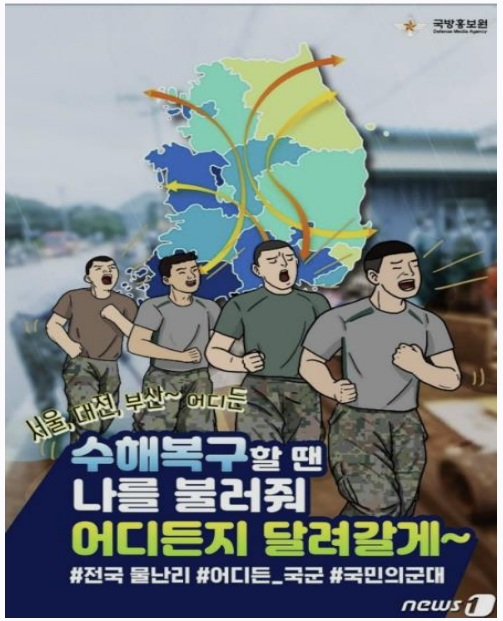


강남역 침수

2022년 강남역 일대 침수

강남 지형 특성

오목하고 지대가 낮은 항아리 지형

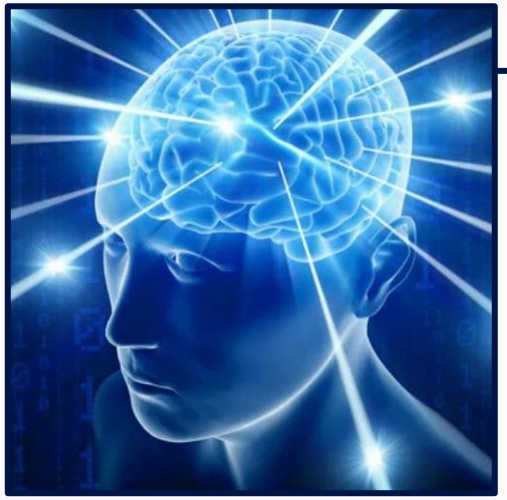


군인 복지

장병들의 대민지원 횟수 감소

주제 소개

아이디어



선제적 대응을 위한 Neural Network

↓

이전의 데이터를 학습을 시켜, 침수 상황을 예측

↓

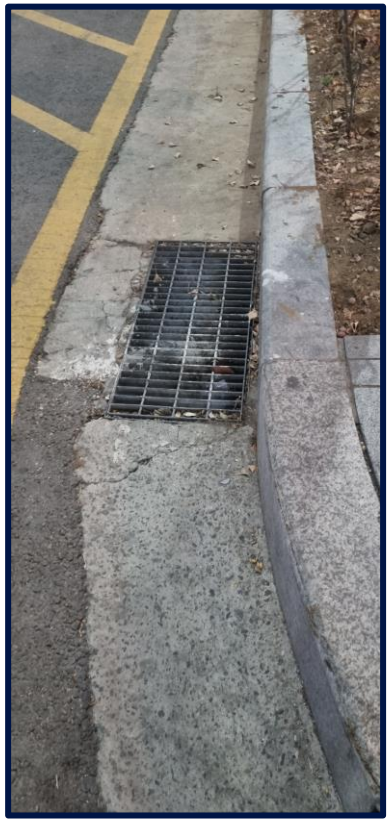
이후 우수받이를 신설해 침수를 예방

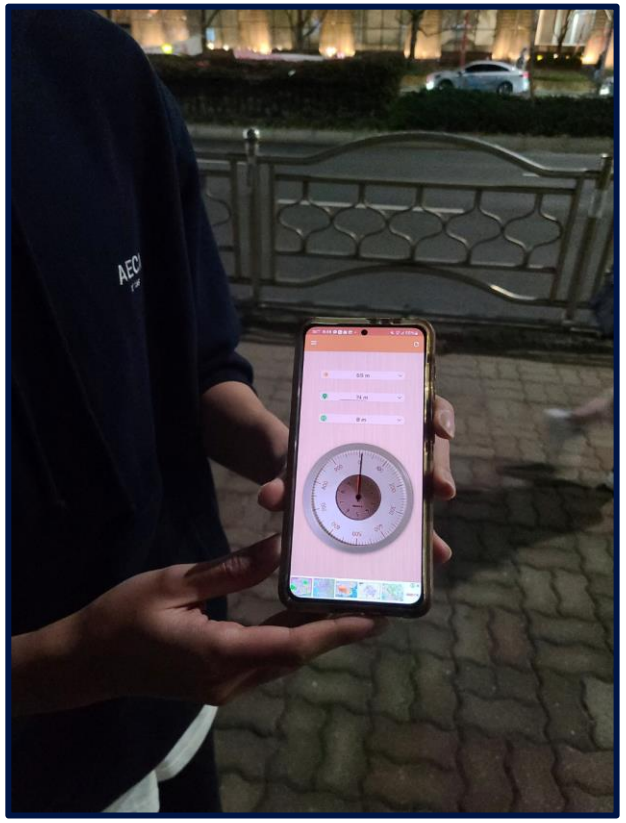
활동 내용

- 1주차 : 참고문헌 조사
- 2~5주차 : 데이터 수집 및 가공
- 6~13주차 : 딥러닝 코드 구현
- 14~16주차 : 딥러닝 학습

데이터 수집

답사를 통해 도로 노면의 우수받이 종류와 설치 방식에 대해 조사하였다.





데이터 수집

서초구청 안전건설교통국 물관리과의 협조로

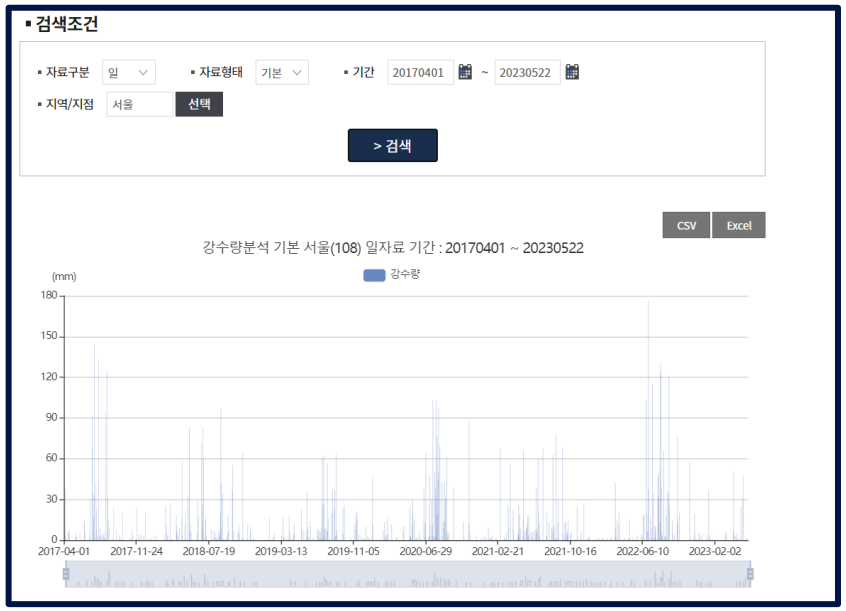
우수관거 지도망과 우수받이 위치 정보 획득

데이터 수집

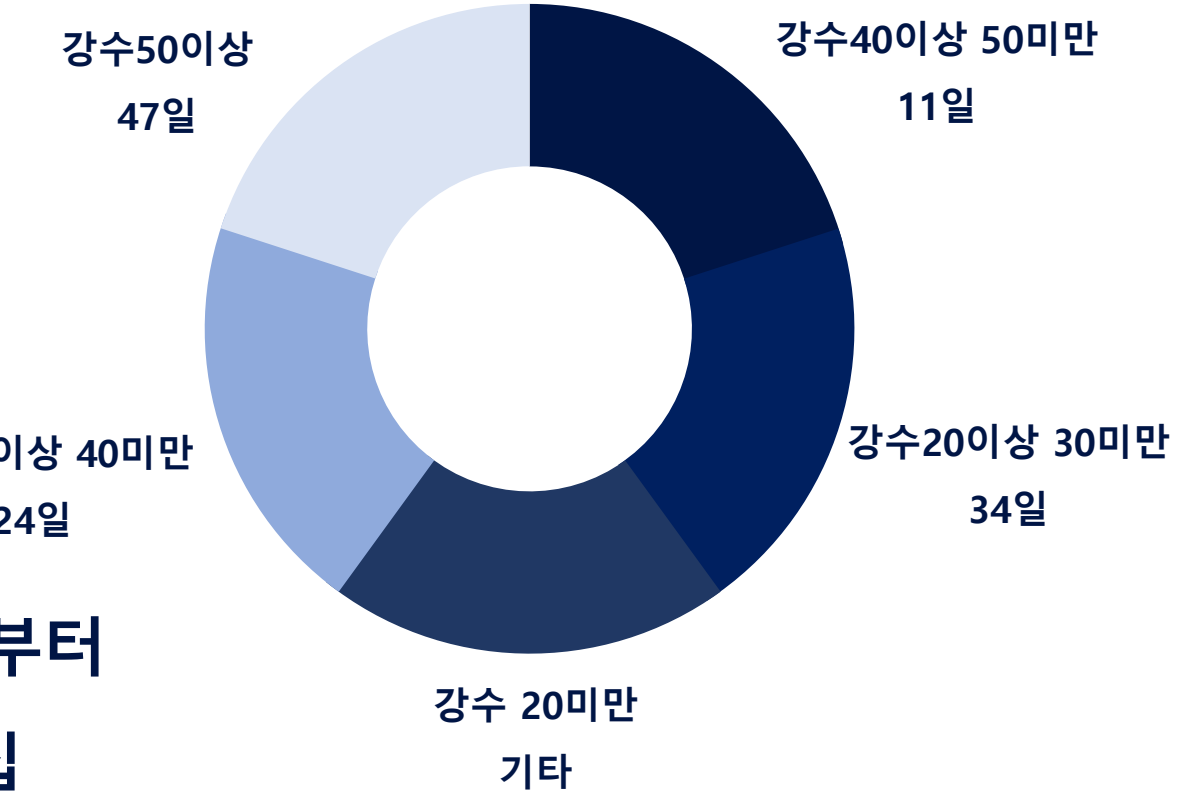
브이월드 데스크톱 3.0 프로그램을 이용해

지형의 고도와 건물 및 도로의 폭을 구함

데이터 수집

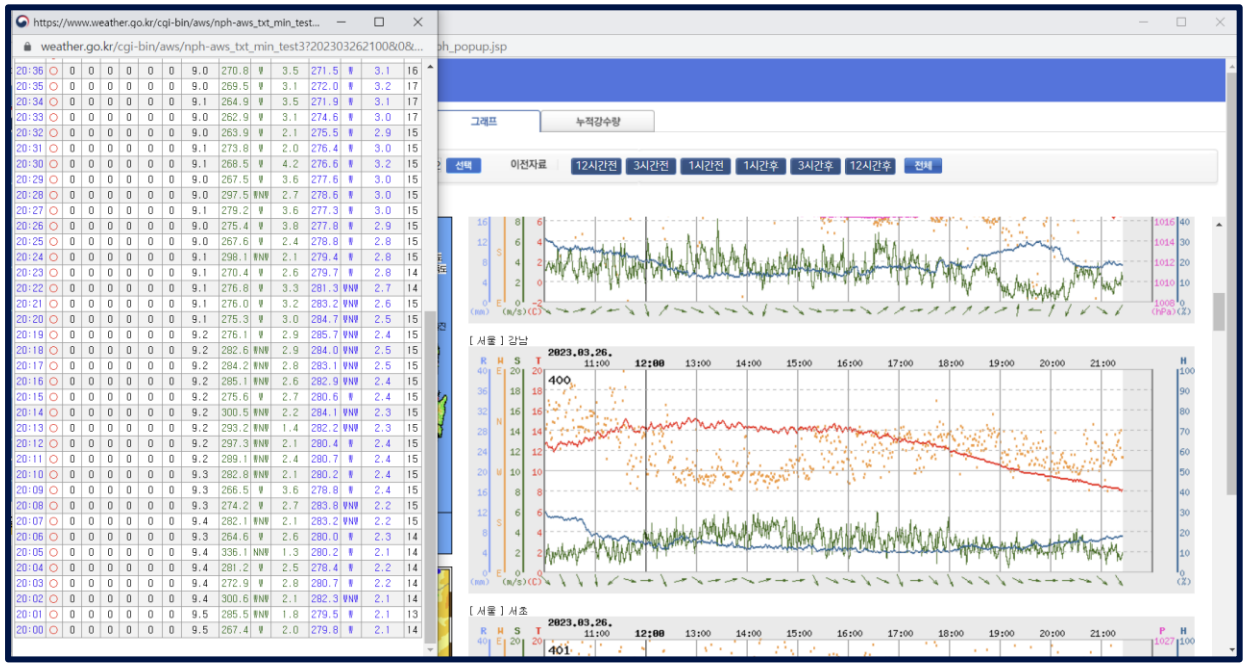


2017년 4월 1일~2023년 5월 22일



기상청 기상자료개방포털에서 2017년 4월부터 2023년 5월까지 일별 강수량 데이터를 수집

데이터 수집



기상청 날씨누리의 지역별 상세 관측 자료실에서

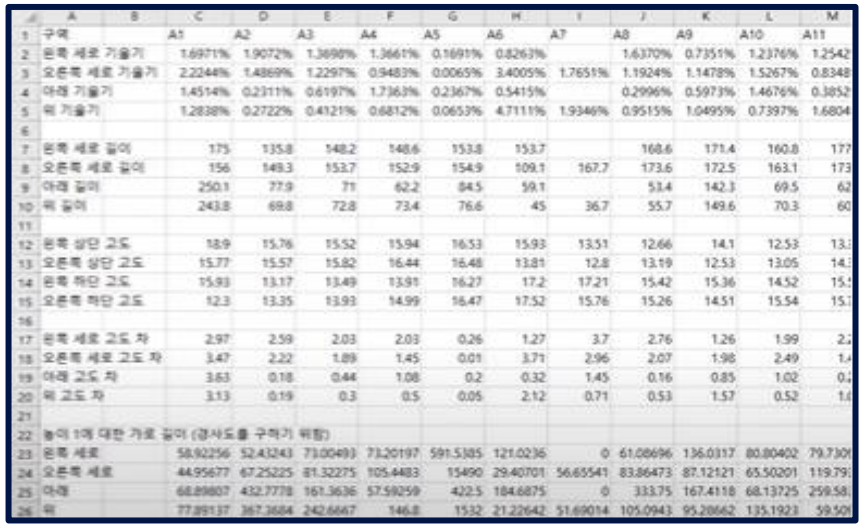
10분당 측정한 강수량 데이터 수집

데이터 가공

강남역 일대 블록화

강남역 일대를 13개의 블록으로

나누고, 각 블록의 고도 구하기



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	구역		A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11
2	왼쪽 세로 기울기		1.6971%	1.9072%	1.3698%	1.3661%	0.1691%	0.8263%		1.6370%	0.7351%	1.2376%	1.2542
3	오른쪽 세로 기울기		2.2244%	1.4869%	1.2297%	0.9483%	0.0065%	3.4005%	1.7651%	1.1924%	1.1478%	1.5267%	0.8348
4	아래 기울기		1.4514%	0.2311%	0.6197%	1.7363%	0.2367%	0.5415%		0.2996%	0.5973%	1.4676%	0.3852
5	위 기울기		1.2838%	0.2722%	0.4121%	0.6812%	0.0653%	4.7111%	1.9346%	0.9515%	1.0495%	0.7397%	1.6804
6													
7	왼쪽 세로 길이		175	135.8	148.2	148.6	153.8	153.7		168.6	171.4	160.8	177
8	오른쪽 세로 길이		156	149.3	153.7	152.9	154.9	109.1	167.7	173.6	172.5	163.1	173
9	아래 길이		250.1	77.9	71	62.2	84.5	59.1		53.4	142.3	69.5	62
10	위 길이		243.8	69.8	72.8	73.4	76.6	45	36.7	55.7	149.6	70.3	60
11													
12	왼쪽 상단 고도		18.9	15.76	15.52	15.94	16.53	15.93	13.51	12.66	14.1	12.53	13.3
13	오른쪽 상단 고도		15.77	15.57	15.82	16.44	16.48	13.81	12.8	13.19	12.53	13.05	14.1
14	왼쪽 하단 고도		15.93	13.17	13.49	13.91	16.27	17.2	17.21	15.42	15.36	14.52	15.5
15	오른쪽 하단 고도		12.3	13.35	13.93	14.99	16.47	17.52	15.76	15.26	14.51	15.54	15.1
16													
17	왼쪽 세로 고도 차		2.97	2.59	2.03	2.03	0.26	1.27	3.7	2.76	1.26	1.99	2.2
18	오른쪽 세로 고도 차		3.47	2.22	1.89	1.45	0.01	3.71	2.96	2.07	1.98	2.49	1.4
19	아래 고도 차		3.63	0.18	0.44	1.08	0.2	0.32	1.45	0.16	0.85	1.02	0.2
20	위 고도 차		3.13	0.19	0.3	0.5	0.05	2.12	0.71	0.53	1.57	0.52	1.6
21													
22	높이 1개 대한 가로 길이 (경사도를 구하기 위함)												
23	왼쪽 세로		58.92256	52.43243	73.00493	73.20197	591.5385	121.0236	0	61.08696	136.0317	80.80402	79.7309
24	오른쪽 세로		44.95677	67.25225	81.32275	105.4483	15490	29.40701	56.65541	83.86473	87.12121	65.50201	119.79
25	아래		68.89807	432.7778	161.3636	57.59259	422.5	184.6875	0	333.75	167.4118	68.13725	259.58
26	위		77.89137	367.3684	242.6667	146.8	1532	21.22642	51.69014	105.0943	95.28662	135.1923	59.50

데이터 가공

데이터들을 프로그래밍에 사용하

기 적합한 형태로 가공

데이터 가공

```
[ ] # 결측치를 예측할 열을 제외한 열 선택
    df_selected = df[['강수15', '강수60', '강수6H', '강수12H', '월강수']]

[ ] # 결측치가 있는 행 제거
    df_selected = df_selected.dropna()

[ ] # 입력(X)과 타겟(y) 데이터 설정
    X = df_selected[['강수15', '강수60', '강수6H', '강수12H']].values
    y = df['강수3H'].values

[ ] # 각 특성별로 스케일링
    scalers = []
    X_scaled = np.empty_like(X)
    for i in range(X.shape[1]):
        scaler = MinMaxScaler()
        X_scaled[:, i] = scaler.fit_transform(X[:, i].reshape(-1, 1)).flatten()
        scalers.append(scaler)

    y_scaled = MinMaxScaler().fit_transform(y.reshape(-1, 1))

[ ] # 데이터 형태 재구성
    X_reshaped = np.reshape(X_scaled, (X_scaled.shape[0], 1, X_scaled.shape[1]))

[ ] # 결측치를 예측할 데이터 선택
    X_predict = df[df['강수3H'].isnull()][['강수15', '강수60', '강수6H', '강수12H']].values
    X_predict_scaled = np.empty_like(X_predict)
    for i, scaler in enumerate(scalers):
        X_predict_scaled[:, i] = scaler.transform(X_predict[:, i].reshape(-1, 1)).flatten()

    X_predict_reshaped = np.reshape(X_predict_scaled, (X_predict_scaled.shape[0], 1, X_predict_scaled.shape[1]))

[ ] # LSTM 모델 구축
    model = Sequential()
    model.add(LSTM(7, input_shape=(1, X_reshaped.shape[2])))
    model.add(Dense(1))
    model.compile(loss='mean_squared_error', optimizer='adam')
    model.fit(X_reshaped, y_scaled, epochs=10, batch_size=24, verbose=1)

[ ] # 강수량 3H 예측
    y_predict_scaled = model.predict(X_predict_reshaped)
    y_predict = scalers[-1].inverse_transform(y_predict_scaled)

[ ] # 예측 결과를 원본 데이터프레임에 적용
    df.loc[df['강수3H'].isnull(), '강수3H'] = y_predict.flatten()

    # 예측 결과 출력
    print(df[['시간', '강수15', '강수60', '강수6H', '강수12H', '강수3H', '월강수']])
```

결측치 예측

구글 Colab에서 강수량 데이터의

결측치를 예측

```
# 데이터 프레임의 '강수3H' 열을 대체하는 코드
for i in range(df.shape[0]):
    if np.isnan(df.loc[i, '강수3H']):
        if i >= 12:
            df.loc[i, '강수3H'] = df.loc[i-6, '강수60'] + df.loc[i-12, '강수60'] + df.loc[i, '강수60']
```

결측치 검토

결측치가 존재하는 부분의 값을 직접

계산

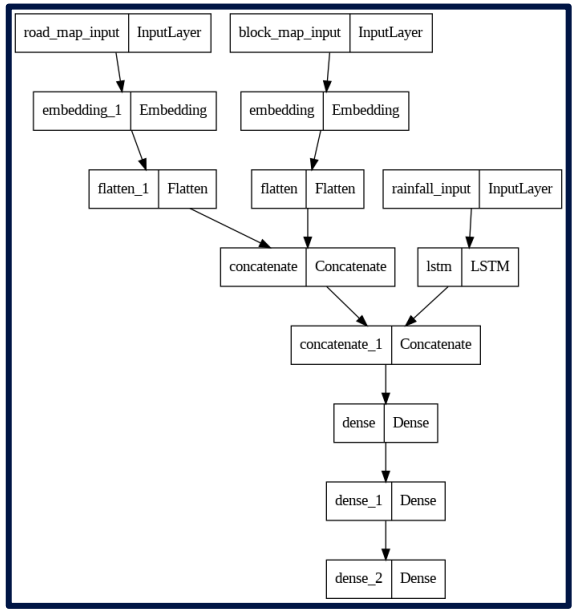
딥러닝 코드 구현

```
class RoadNode:
  def __init__(self, id, stormdrain, length):
    self.id = id
    self.stormdrain = stormdrain
    self.length = length
    self.direction = None

class BlockNode:
  def __init__(self, id, area, coeff):
    self.id = id
    self.area = area
    self.coeff = coeff
    self.directon = None
    self.directions = {
        0 : [], 1 : [], 2 : [], 3 : []
    }
    self.roads = [None] * 4

  def add_block_node(self, block_node, direction):
    self.directions[direction].append(block_node)

  def add_road_node(self, road_node, direction):
    self.roads[direction] = road_node
```

노드 구현

맵 데이터를 사용하기 위한

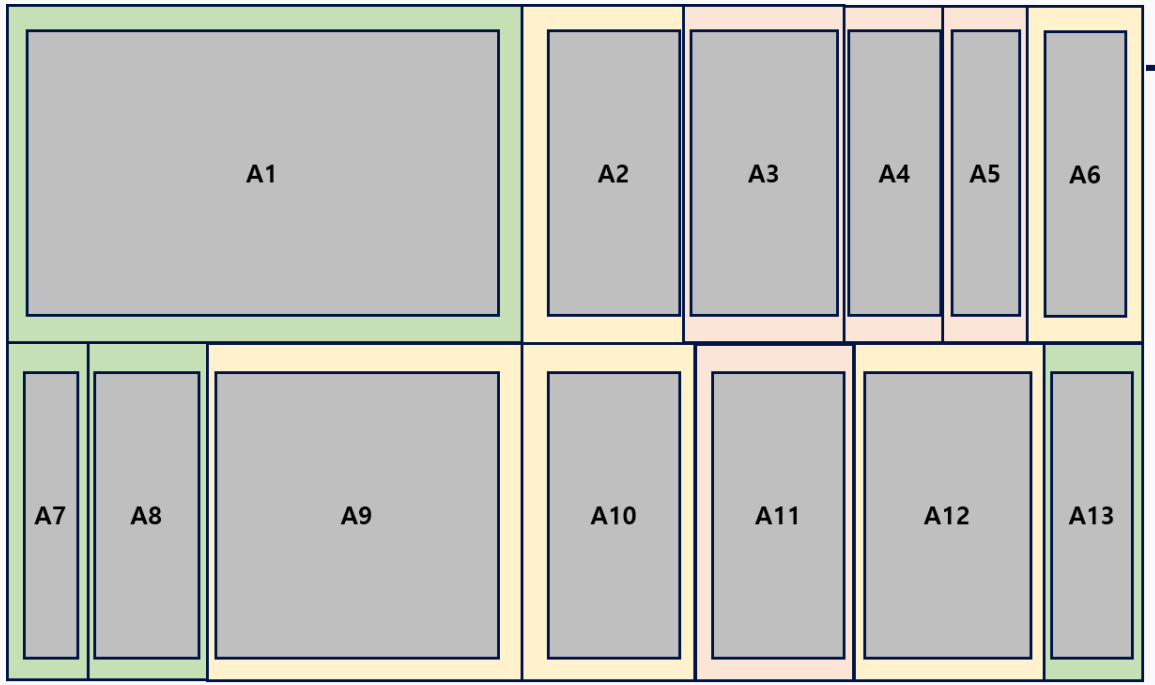
노드 구현



모델 구현

딥러닝 모델 구현

시각화(예정)



도로 히트맵(예)

Heatmap을 통해 도로의 침수정도를 시각화 함

색이 붉을수록 침수되었다는 뜻

주로 침수되는 도로를 기준으로 우수받이 신설