

딤러닝 기반 건물 침수 대응 프로그램

EXIT

21827847 김남주

21827889 안중보

21828040 박수영

목차

1. 목적

2. 과제 세부 내용

- 사용한 모델
- 데이터셋 수집
- 이미지 전처리
- 성능 개선

3. 결과물 형태

- 동작 과정
- 실시간 침수 상황 인식 결과

목적

행정안전부가 집계한 바에 따르면, 최근 10년간(2011~2020년) 발생한 자연재해로 총 4조 4,193억 원의 재산피해가 발생했다. 그런데 전체 피해액의 대부분인 4조 2,776억 원(96.8%)이 태풍, 호우, 대설 등 물과 관련된 재해로 인해 발생했으며, 그 밖의 재해로 인한 피해액은 1,417억 원(3.2%)에 불과하다. 같은 기간 자연재해로 인한 인명피해는 태풍, 호우, 폭염으로 총 290명이 사망했는데, 전체 인명피해 중 절반이 넘는 183명(63.1%)이 물 관련 재해로 인해 발생했다.

[표 1] 최근 10년간(2011~2020년) 자연재해 피해액 현황

(단위: 백만 원, 당해연도 가격)

구분	합계	2011년	2012년	2013년	2014년	2015년	2016년	2017년	2018년	2019년	2020년
합계	4,419,280	794,200	1,089,210	172,137	180,019	31,862	288,862	187,302	141,284	216,226	1,318,177
물 관련 재해	4,277,597	793,901	1,062,498	171,161	179,924	27,637	269,540	101,675	138,447	215,100	1,317,713
기타 재해	141,683	299	26,712	976	95	4,225	19,322	85,627	2,837	1,126	464

※ 주: 물 관련 재해는 태풍, 호우, 대설이 있으며, 기타 재해는 강풍, 풍랑, 낙뢰, 한파, 지진, 폭염이 있다.

※ 자료: 행정안전부, 『2020 재해연보』, 2021. p.26. 저자 수정.

목적

침수가 되기 쉬운 지역의 건물 등에서 cctv를 통해 주변 지역의 침수를 감지하고 건물 내의 인원들에게 경보를 하고 차수막을 자동으로 올려주어 침수에 대응할 수 있도록 하는 프로그램을 만들 예정이다.



침수가 된 이미지 감지



경보 및 차수막 가동

과제 세부 내용 (사용한 모델)

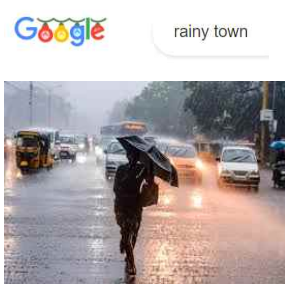
이미지 분류를 위해 2015년 ILSVRC (ImageNet Large Scale Visual Recognition Challenge)에서 우승을 차지한 ResNet을 사용해 이미지 분류를 할 것이다.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

resnet의 구조

과제 세부 내용 (데이터셋 수집)

flood 이미지, not-flood 이미지, rainy 이미지 3가지 class로 이미지 분류를 하였다.
구글 이미지 크롤링을 통해 데이터셋을 구하였다.



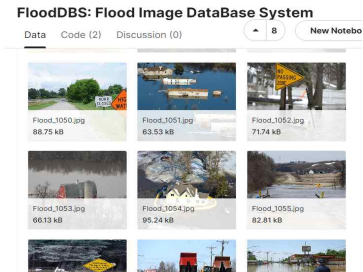
구글 크롤링 검색어와 이미지

과제 세부 내용 (데이터셋 수집)

flood 이미지는 크롤링과 kaggle의 FloodDBS: Flood Image DataBase System에서 400개의 이미지를 다운받아 사용하였다.



침수가 된 이미지



kaggle - FloodDBS: Flood Image DataBase System

과제 세부 내용 (데이터셋 수집)

이미지 검수 후 학습할 최종 데이터

Flood Image : 1089

Not Flood Image : 1004

Rainy Image : 979

각각8:2의 비율로 나누어 train, test set을 나누었다

1. train/test split
8:2의 비율로 split

train 폴더 이미지 개수 : 871

test 폴더 이미지 개수 : 218

train 폴더 이미지 개수 : 803

test 폴더 이미지 개수 : 201

train 폴더 이미지 개수 : 783

test 폴더 이미지 개수 : 196

2. 이미지를 랜덤 추출하였기
때문에 다시 이미지 넘버링

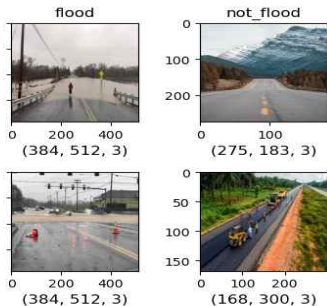
각각 마지막 파일 이름 : flood_870.jpg not_flood_802.jpg rainy_782.jpg

각각 마지막 파일 이름 : flood_217.jpg not_flood_200.jpg rainy_195.jpg

과제 세부 내용 (이미지 전처리)

직접 수집한 이미지들에 모델이 학습을 더 잘 하기 위해서 여러 전처리를 진행했다.

1. 이미지 사이즈가 모두 달라 resize 진행



resize 되지 않은 이미지들



128 x 128 변환한 이미지

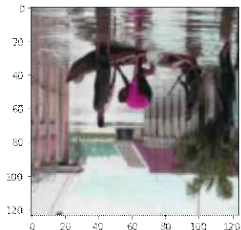
과제 세부 내용 (이미지 전처리)

2. albumentations로 Image augmentation를 하였다.

```
albumentations.HorizontalFlip(p=0.8), # p확률로 이미지 좌우 반전  
albumentations.RandomRotate90(p=0.8), # p확률로 90도 회전  
albumentations.VerticalFlip(p=0.8) # p확률로 이미지 상하 반전
```

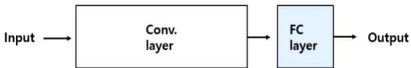
([



```
albumentations.MotionBlur(p=0.8), # p확률로 이미지를 흐리게(?) 만들어 줌  
albumentations.OpticalDistortion(p=0.8), # p확률로 이미지 왜곡  
albumentations.GaussNoise(p=0.8) # 임의의 noise를 삽입
```



과제 세부 내용 (성능 개선)

fine tuning 을 하여 성능을 높였다.
Data의 양이 적어 transfer learning을 하였다.
100epoch로 학습결과 86%의 정확도가 나왔다.



 : Train (LR = original LR / 10)
 : Frozen (LR = 0)

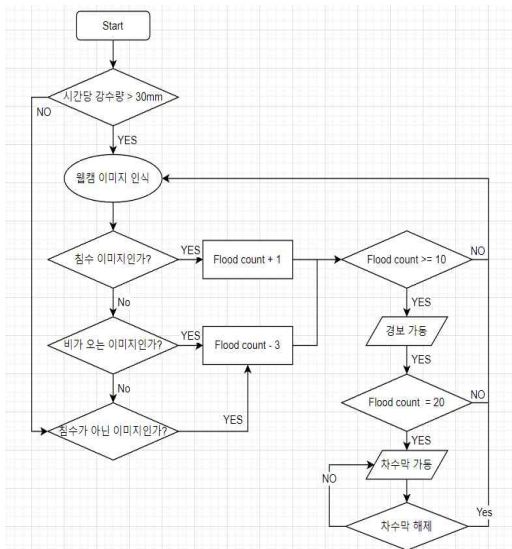
사전 학습한 FC layer를
연결하여 transfer learning



86% 정확도

결과물 형태 (순서도)

1. 입력한 도시의 시간당 강수량이 30mm 이상이면 이미지 인식 (30분마다 강수량 측정)
2. 실시간 웹캠 이미지 인식
 - (1). 침수면 Flood count + 1
 - (2). 침수가 아니거나, 비가 내리는 이미지 이면 Flood count - 3
3. Flood count가 10이상이면 경보
4. Flood count가 20이상이면 차수막 가동
5. 사용자가 차수막 직접 작동 중지



결과물 형태 (동작 과정)

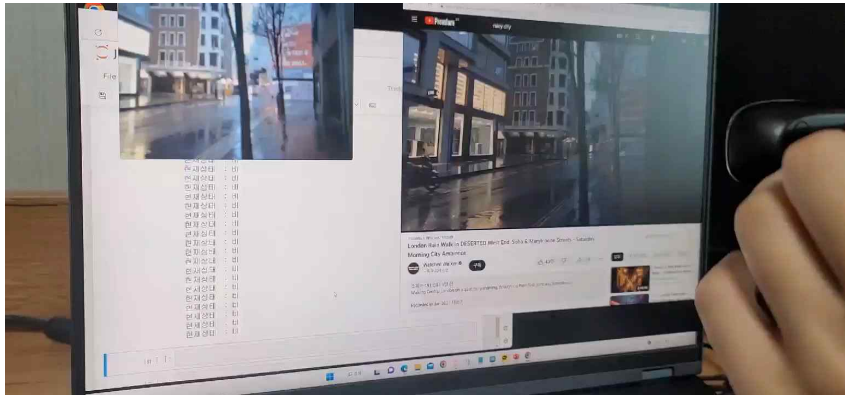
1. 오작동을 방지하기 위해 도시를 입력받고 기상청 날씨누리 홈페이지에서 실시간 강수량을 측정해 시간당 강수량이 30mm 이상일 때부터 비, 침수를 감지한다.

날씨누리 홈페이지에서 데이터를 크롤링하는 모습

The screenshot shows a Jupyter Notebook with a Python script for webcam monitoring. The script includes a loop that checks for a key press (cv2.waitKey) and updates a count. A red box highlights the line '현재 지역 : 인천' and '현재 관측량 : 0.7'. To the right, a table displays weather data for various locations, with a red box highlighting the row for '인천 구름조금' (Incheon, Partly Cloudy).

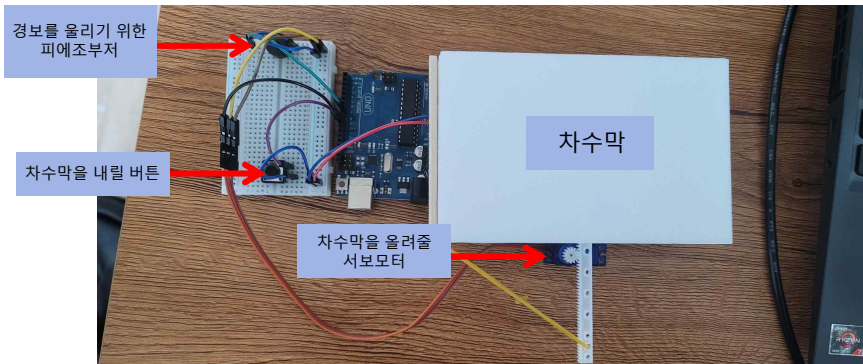
결과물 형태 (동작 과정)

3. 비가 오는 이미지를 인식하여 단순히 비가 와도 침수로 인식하지 않는다.

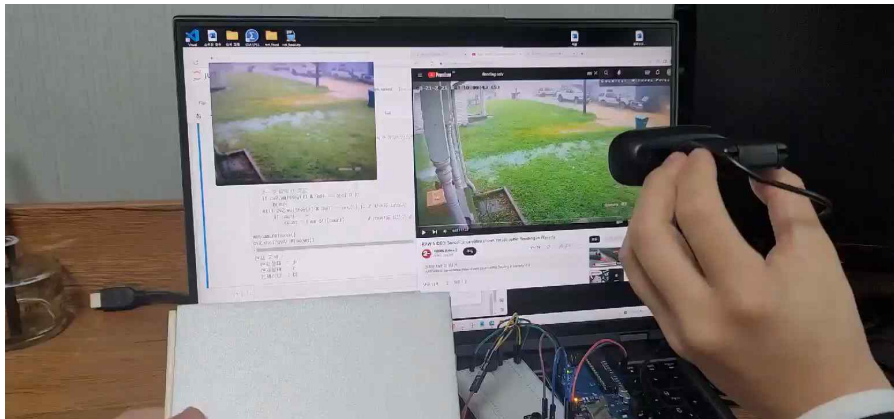


결과물 형태 (아두이노 설계)

파이썬 시리얼 통신으로 값을 받아 피에조부저와 서보모터가 작동하도록 최종 형태를 만들었다.



결과물 형태 (실시간 침수 상황 인식 결과)



감사합니다.