2023 SW 중심대학 공동 AI 경진대회

위성 이미지 건물 영역 분할

2023.08.11

TEAM ADED

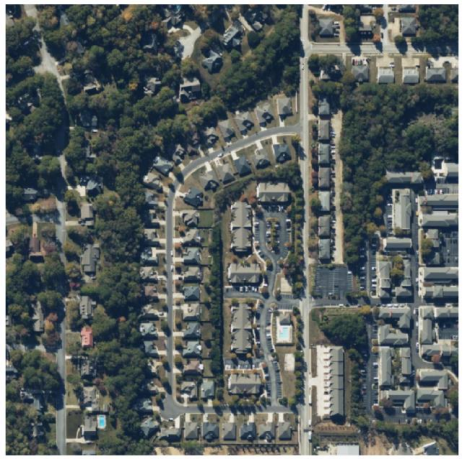
목차 A table of contents

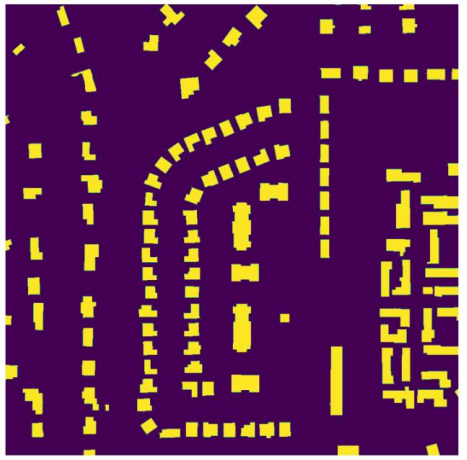
Overview

과제 위성 이미지 건물 영역 분할 AI 모델 개발 ➡Segmentation Task

데이터셋

- **Train_img**
  - 1024x1024
  - 7,140장
- **Test_img**
  - 224x224
  - 60,640장



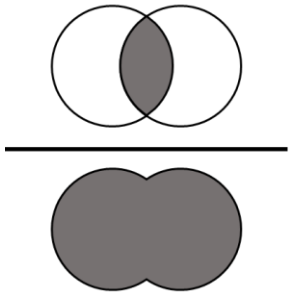


평가 지표

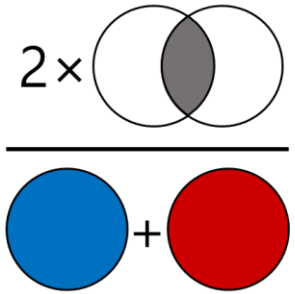
Dice Coefficient(Score)

- Ground-Truth와 Prediction **유사도**
- TP에 가중치 => **겹친 영역**을 더 중요시
- TN의 경우 평균 계산에서 **'제외'**됨
- FP의 경우 **0점 처리**

IoU

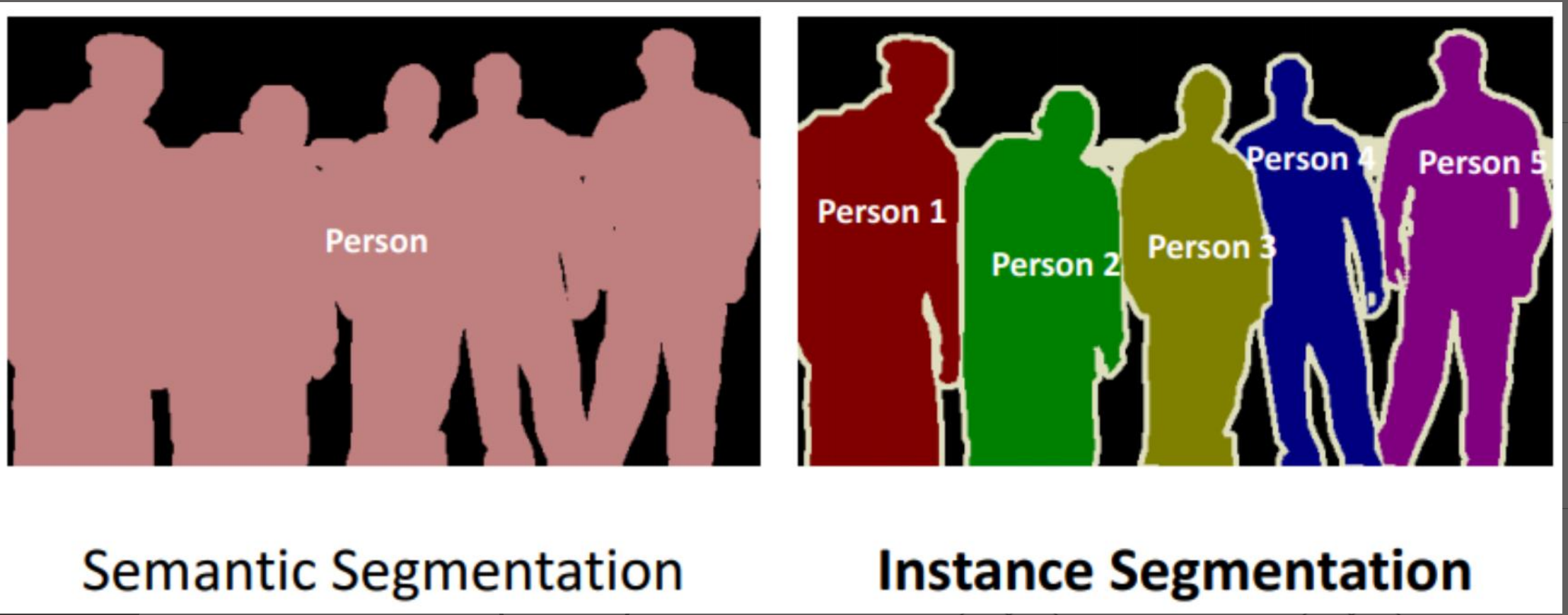


Dice Coefficient



Overview

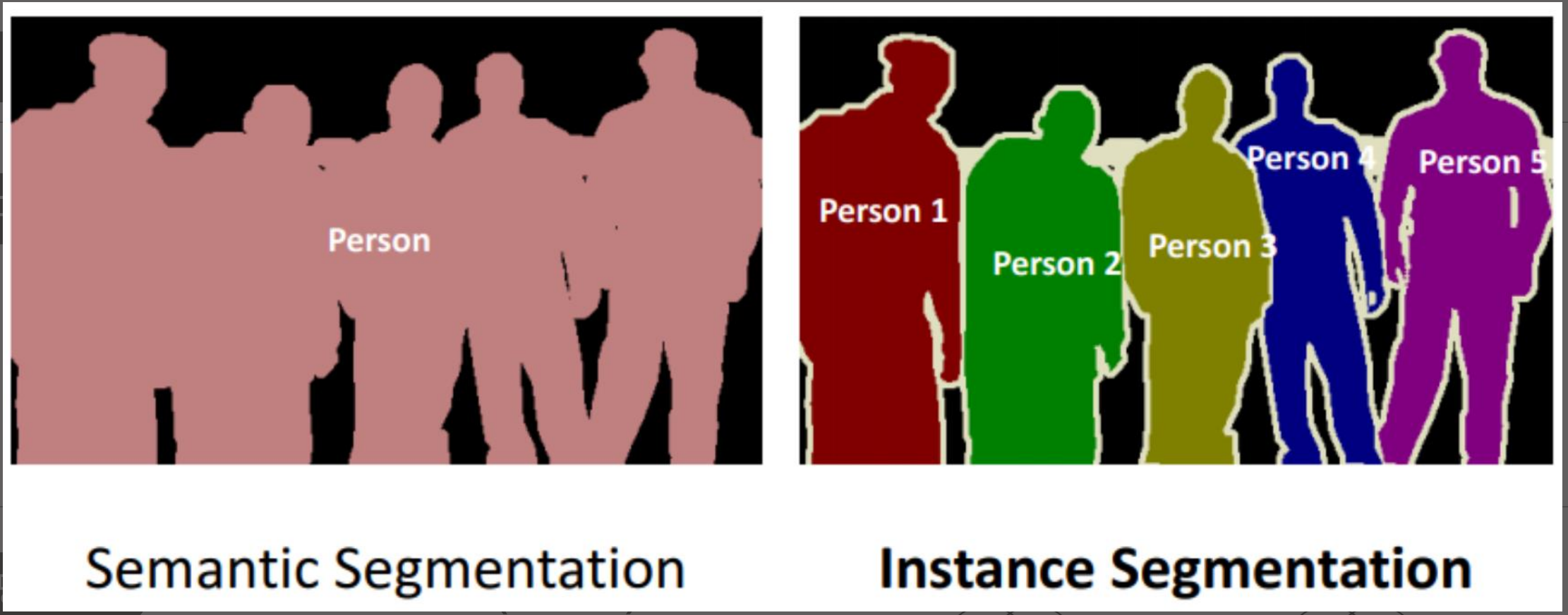
[13] segmentation image



- Ground-Truth와 Prediction **유사도**
- TP에 가중치 => **겹친 영역**을 더 중요시
- TN의 경우 평균 계산에서 **'제외'**됨
- FP의 경우 **0점 처리**

Overview

[13] segmentation image



- Ground-Truth와 Prediction 유사도

2×

=>Semantic Segmentation Task

1. 데이터 분석

고해상도

Training Dataset 해상도 : **1024x1024**

높은 해상도로 인해 원본 이미지를 바로 처리하기 어려움

다양한 환경/시간대/화질

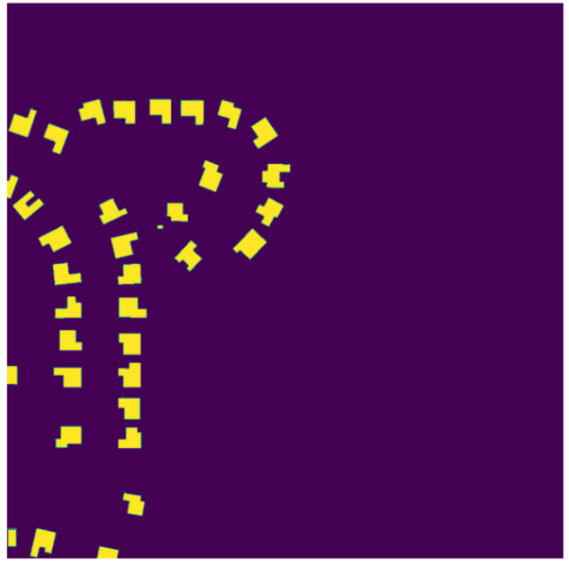
일반 사진에 비해 사진이 찍힌 **시간대, 계절, 환경, 안개** 정도에 따라 사진 별 차이 존재

데이터 수 부족

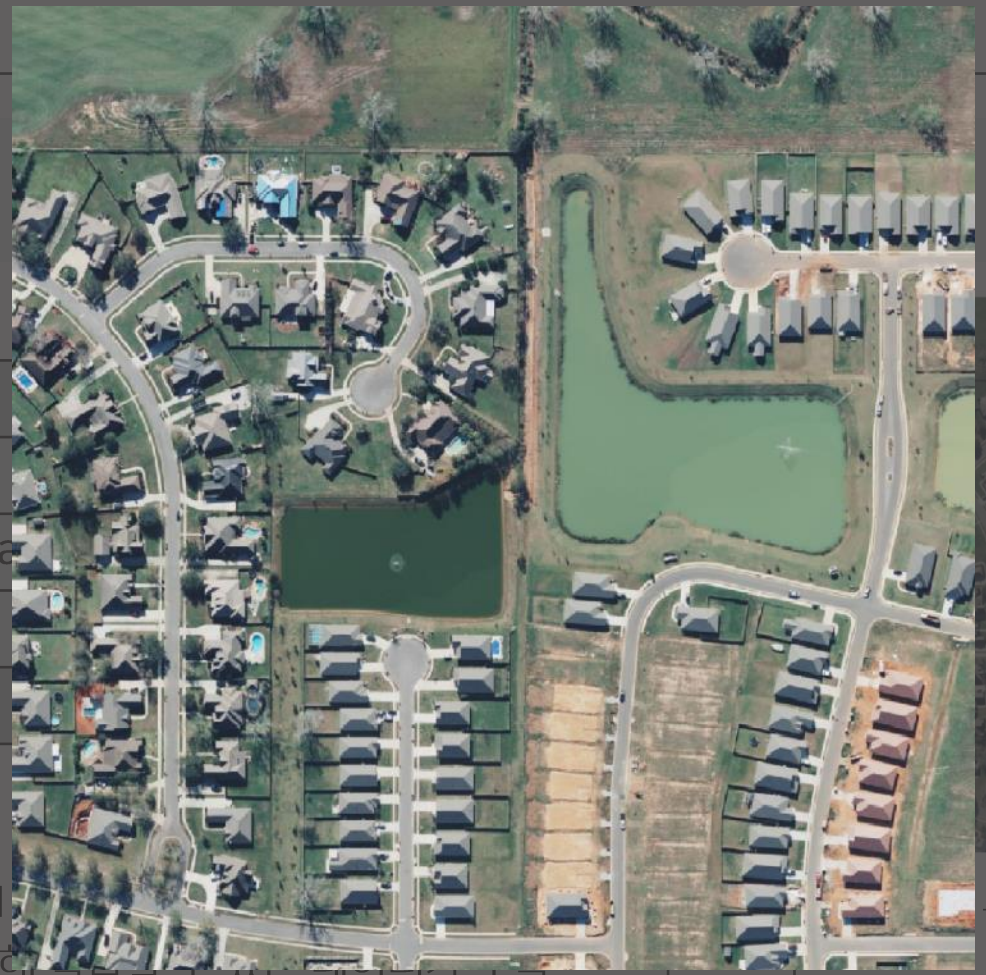
DATASET	VOLUME
ADE20K	25,574
Mapillary Vistas	25,000
Cityscapes	25,000
Dacon	7,140

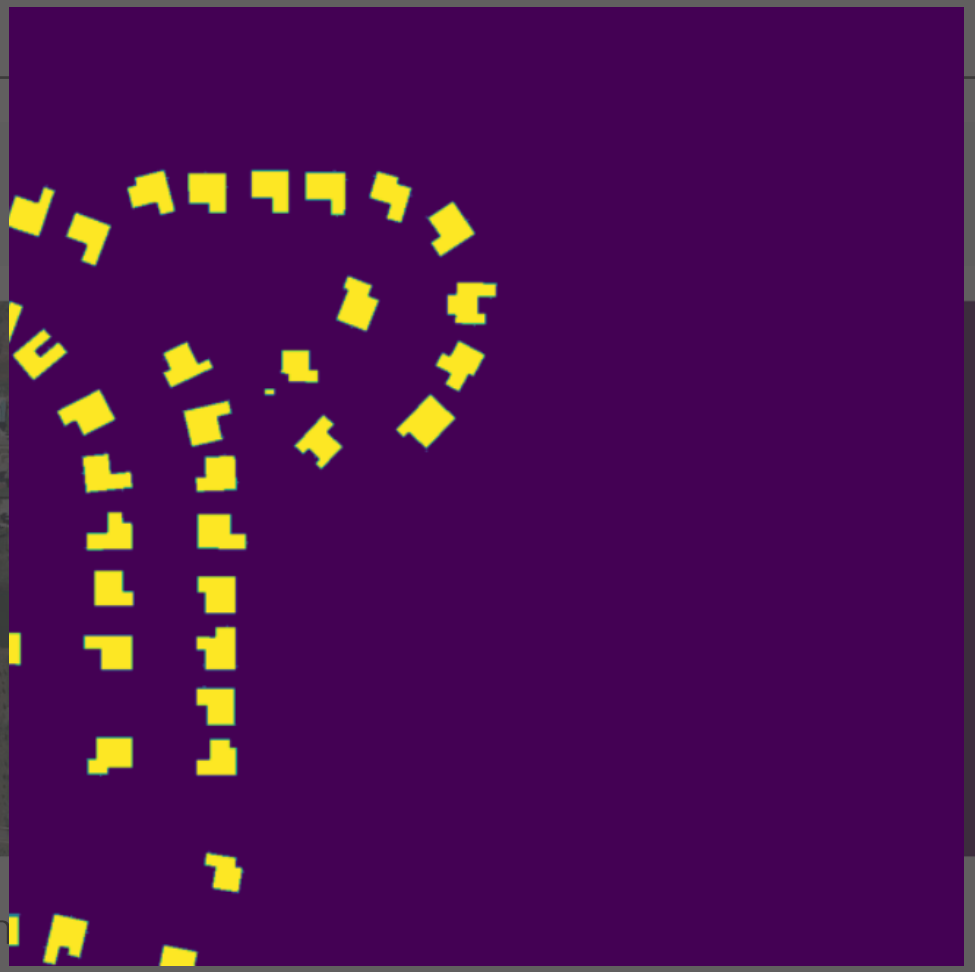
총 **7140장**으로, 다른 이미지 데이터셋에 비해 학습할 수 있는 데이터가 **부족**

Train Ground-Truth 마스킹 오류

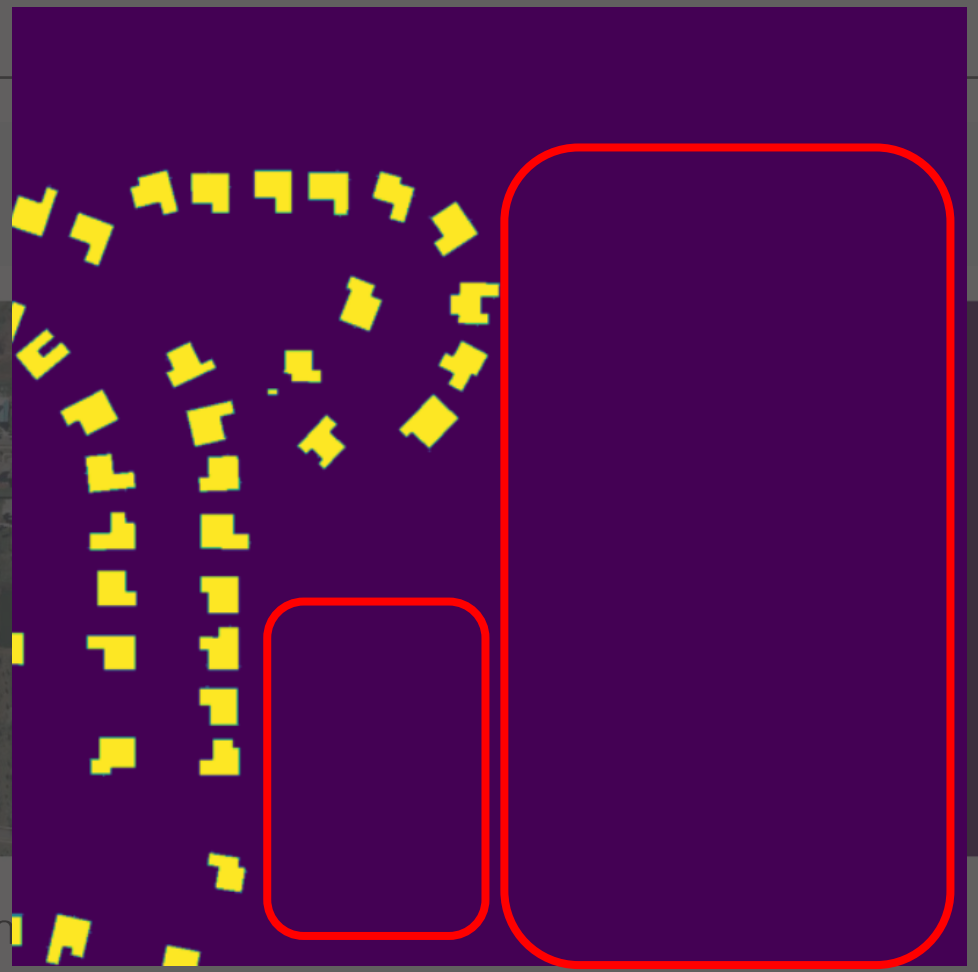


Training dataset 중 건물이 존재하나 마스크엔 표시되지 않은 이미지 **951장** 검출





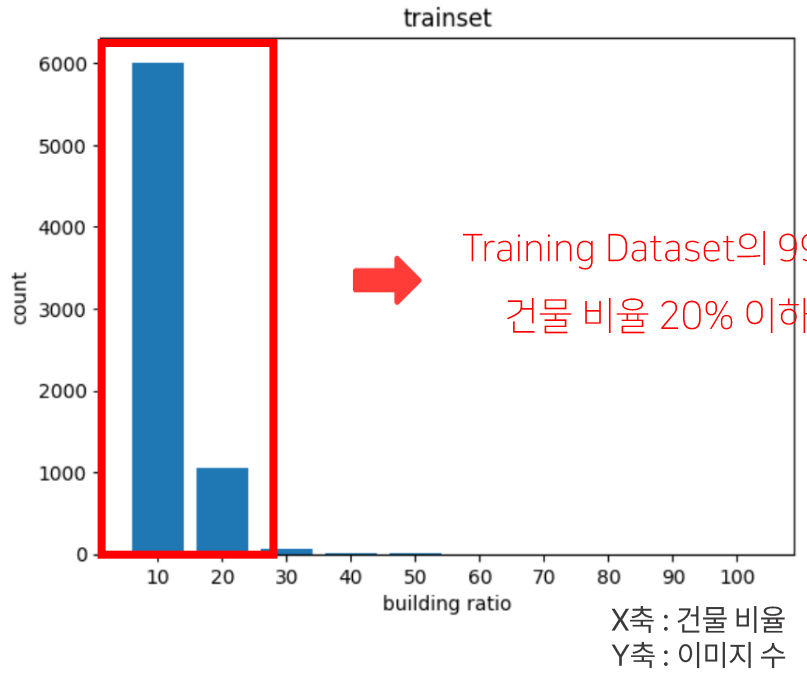
약 951장 검출



약 951장 검출

건물이 존재하는 부분의 마스크가 없음

A 건물 비율

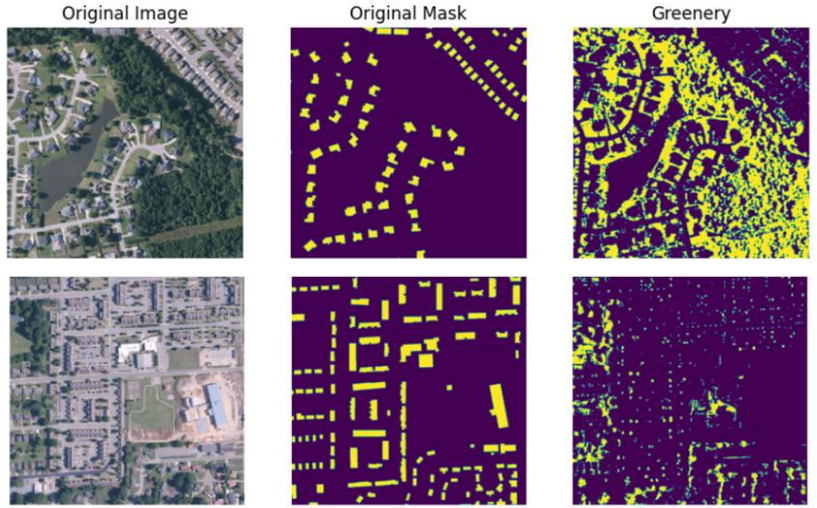


Training Dataset의 99%가
건물 비율 20% 이하임

X축 : 건물 비율
Y축 : 이미지 수

⇒ Task에 비해 **건물의 수가 부족함**

B 녹지 비율



- Training Dataset 중 배경에 **녹지가 포함된 데이터가 대부분임**
- HSV 기반 Greenmask로 녹지 비율 계산
- 사용된 HSV 값 : cv2.inRange(HSV,(32,23,23),(90,210,210))

(Greenmask의 HSV값은 Trainset에 직접 마스크를 씌워가며 생성)

- 건물 : 전체 데이터셋의 84%가 건물 비율10% 이하

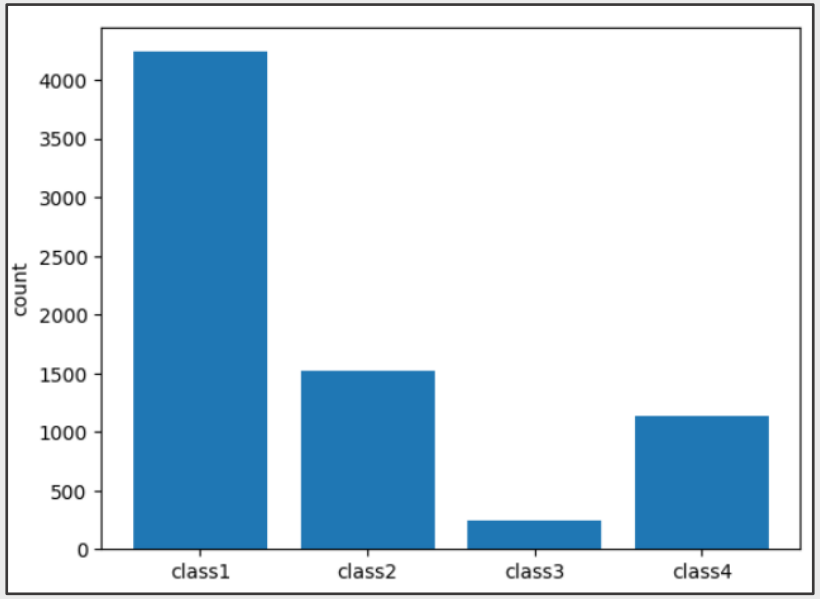
  ⇒ 건물 비율 10%이하, 10% 초과로 나눔

- 녹지 : HSV 분석 결과 데이터를 녹지 비율이 20%,80%을 기점으로 그룹화 가능

  ⇒ 크게 녹지 비율 0~20%, 20~80%, 80~100%로 나눔

- 앞서 분석한 결과를 토대로 나눈 임의의 클래스

  - class1 : 건물 ≤ 10% and 20% ≤ 녹지 ≤ 80%
  - class2 : 건물 ≤ 10% and 녹지 < 20%
  - class3 : 건물 ≤ 10% and 80% < 녹지
  - class4 : 10% < 건물



각 클래스별 데이터 수

- class1 : 4239장
- class2 : 1523장
- class3 : 247장
- class4 : 1131장

1. 마스킹 오류	Labelme Tool
2. 데이터 수 부족	RICAP, Mirror, Flip, Rotate, OBA(증강)
3. 데이터별 편차	Blur, RandomShadow, ColorJitter CLAHE… (증강)
4. 클래스 불균형	부족한 클래스 증강
5. 고해상도 이미지	Image Patch

2. 데이터 전처리

2-1 Data Cleaning

2-2 데이터 증강

2-3 Data Balancing

2-4 Image Patch

1 Data Cleaning

데이터 분석을 통해 Training Dataset에서 **비정상적인 Ground-Truth를 발견**하였고, 이것이 모델의 학습에 악영향을 미친다고 판단하여 다음 **두가지 방법을 시도**하였음.

I. 비정상적인 Ground Truth을 가진 이미지들을 모두 제거 후 재 학습

⇒ 데이터셋이 줄어들어 기존보다 **성능이 저하**된 것으로 보임.

II. 비정상적인 Ground Truth을 가진 이미지들을 Labelme Tool을 이용해 모두 수정 후 재 학습

⇒ 0.819 -> 0.820 로 **성능 향상됨**

Lableme Tool을 이용해
마스킹 되지 않은 부분 수정 예시

2 데이터 증강 – 데이터 부족 해결

- Albumentation 라이브러리: Rotate transform / Flip transform

<Origin>

<Rotate>

<VerticalFlip>

<HorizontalFlip>

2 커스텀 데이터 증강 – 데이터 부족 해결

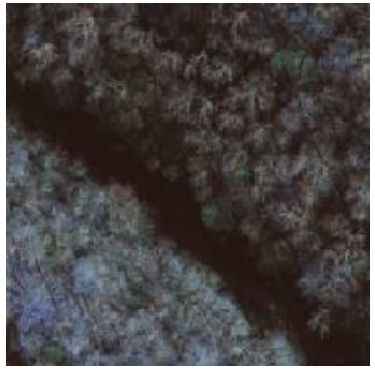
- Mirror 증강기법

HorizontalMirrorDown

VerticalMirrorUp

- RICAP 증강기법

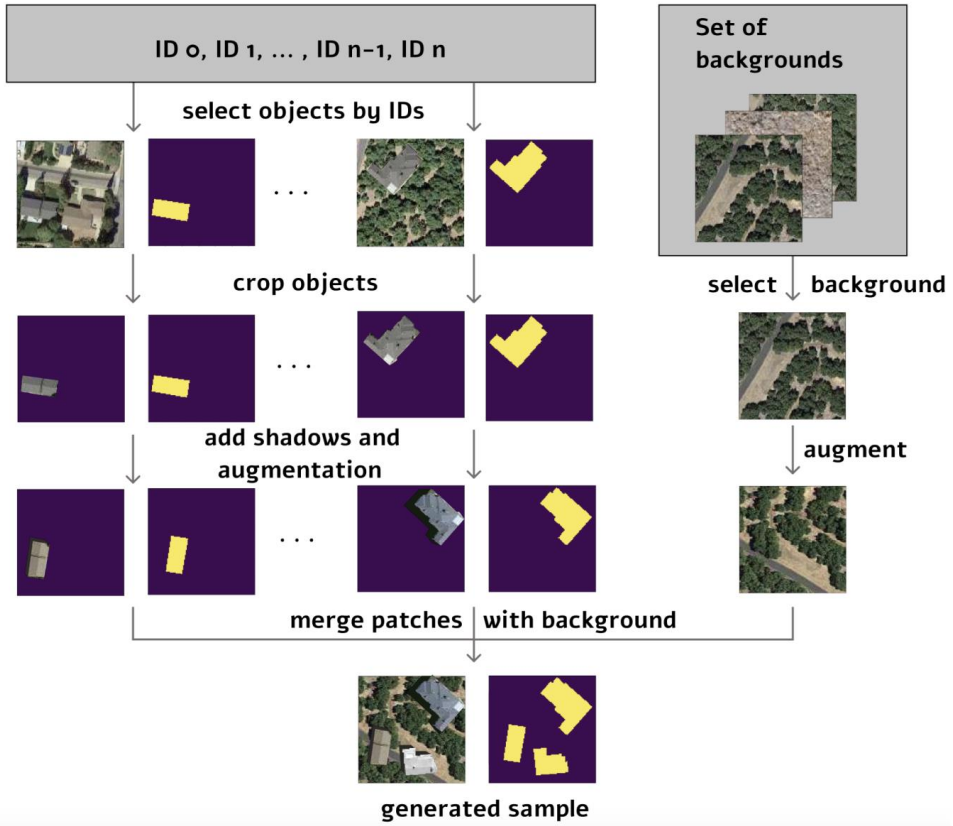
Original Samples



Generated Samples

2 커스텀 데이터 증강 – 데이터 부족 해결

- OBA 증강기법



[3] OBA

Original Samples

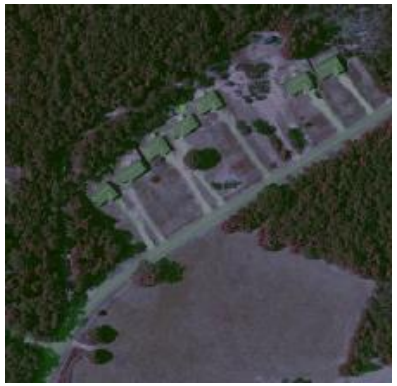
Generated Samples

3 데이터 증강 – 시간/환경별 편차 완화

- Albumentation 라이브러리: Color, Blur transform

<CLAHE>

<ColorJitter>



<Brightness Contrast>

<Blur>

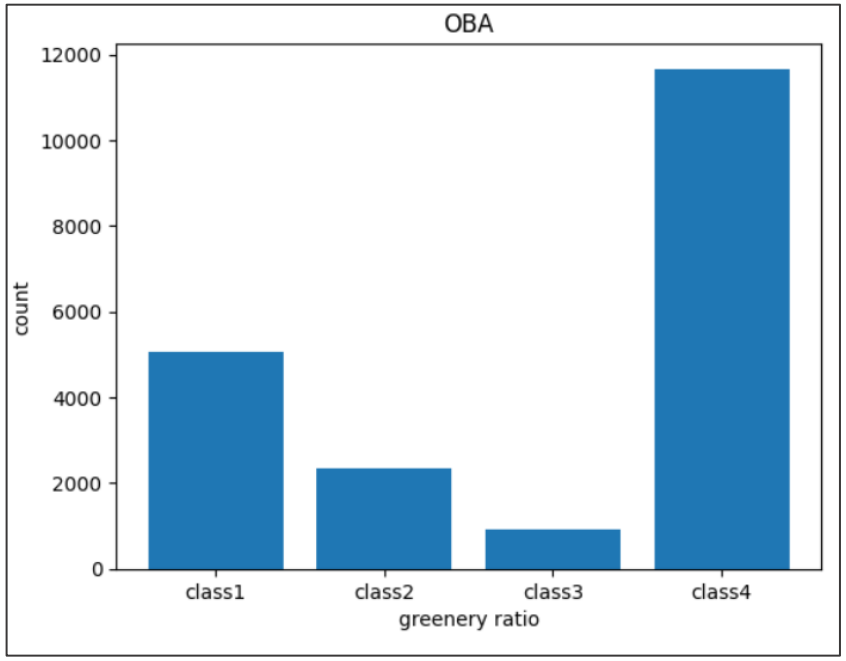
<HueSaturationValue>

<Equalize>

4 Data Balancing

데이터 분석을 통해 Training Dataset의
클래스 불균형 문제를 발견.

Training Dataset의 부족한 클래스 보완을 위해 OBA 수행

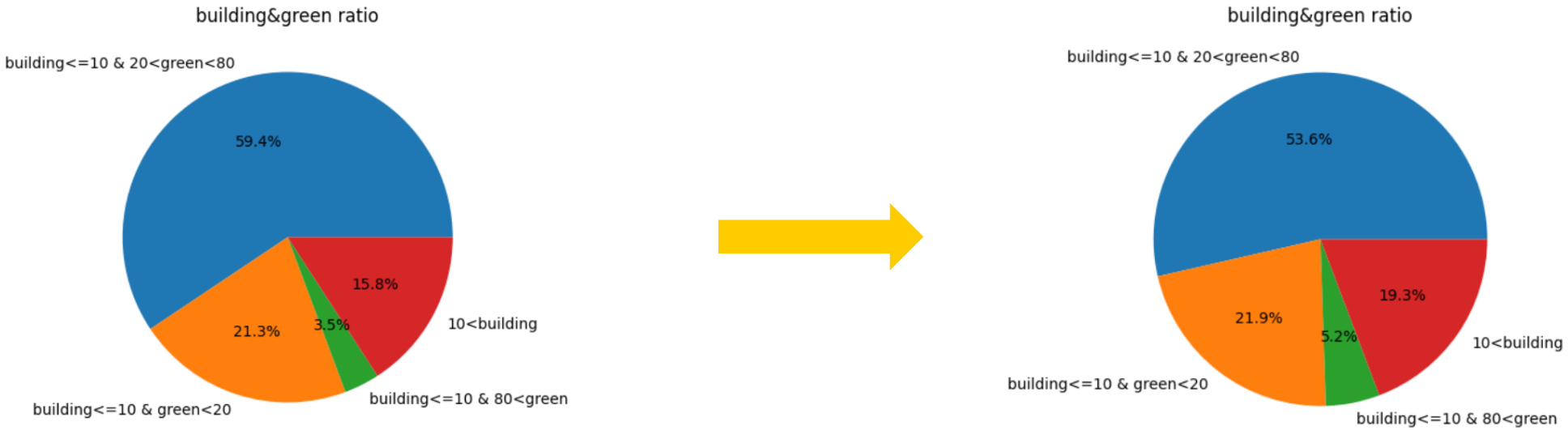


- OBA를 통해 데이터 20000장 증강
- class1 : +5060장
- class2 : + 2347장
- class3 : + 926장
- class4 : + 11667장

4 Data Balancing

데이터 분석을 통해 Training Dataset의
클래스 불균형 문제를 발견.

Training Dataset의 부족한 클래스 보완을 위해 OBA 수행
⇒ 0.815 -> 0.820(Public) 로 성능 향상됨



클래스별 비율 17 : 6 : 1 :4.5 -> 15 : 6.2 : 1 : 5.5

5 Image Patch

데이터 분석을 통해 고해상도 위성 이미지를 모델 학습에 적합한 크기로 조정해야 할 필요성 인지

1024x1024 이미지를 512x512로 나눔,

stride = 256

⇒ 총 9장의 Train_img Patch 생성

1024x1024 Train_img 1장당

512x512 Train_img 9장

2. 데이터 전처리

1. 마스킹 오류	Labelme Tool
2. 데이터 수 부족	RICAP, Mirror, Flip, Rotate, OBA(증강)
3. 데이터별 편차	Blur, RandomShadow, ColorJitter CLAHE… (증강)
4. 클래스 불균형	부족한 클래스 증강
5. 고해상도 이미지	Image Patch

3. 모델 검증

3-1 Validation Set 구축 전략

a. 랜덤 8:2 분할

b. 랜덤 K-Fold 분할

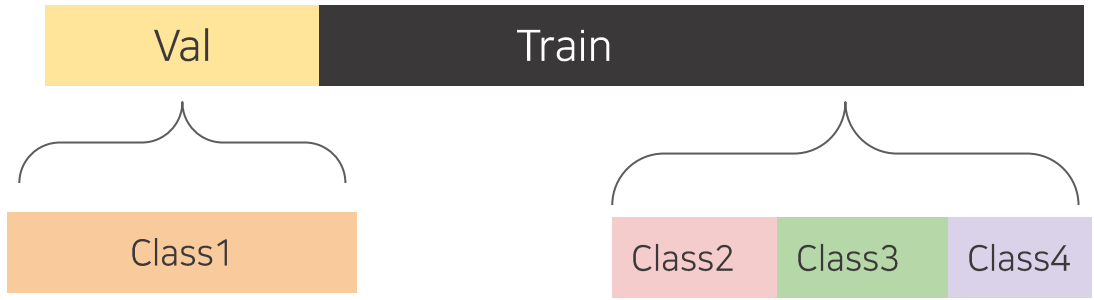
c. Custom Stratified K-fold

3-2 자체 모델 결과 분석

a. 건물 o 이미지

b. 건물 x 이미지

1 Train : Val = 8 : 2 => 랜덤 분할

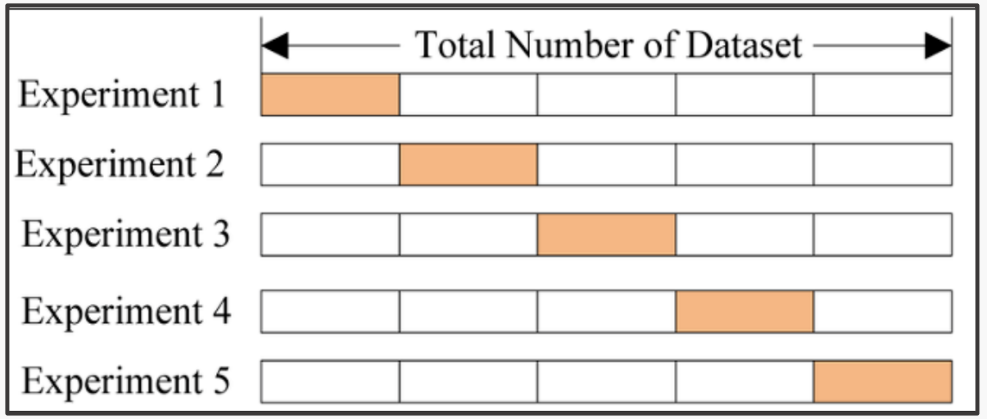


전체 Training Dataset을 **랜덤**하게 나눌시

=> **데이터가 편향**되어 클래스별 학습이 어려움

=> Validation Set을 학습에 사용하지 않음

2 K-Fold => 랜덤 분할



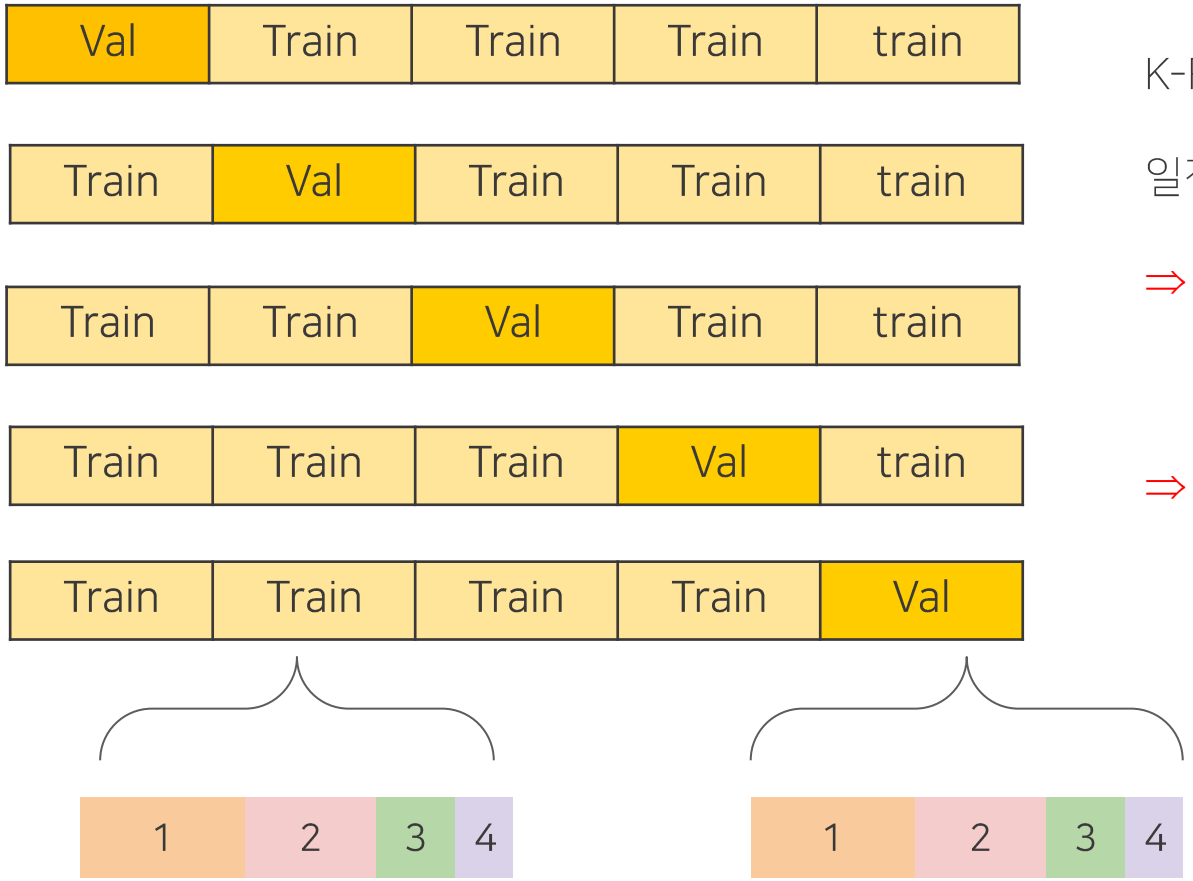
[11] k-fold

Fold가 5인 K-Fold 방식으로 랜덤하게 분할시

⇒ Validation Set도 학습에 사용

⇒ **데이터가 편향**되어 클래스별 학습이 어려움

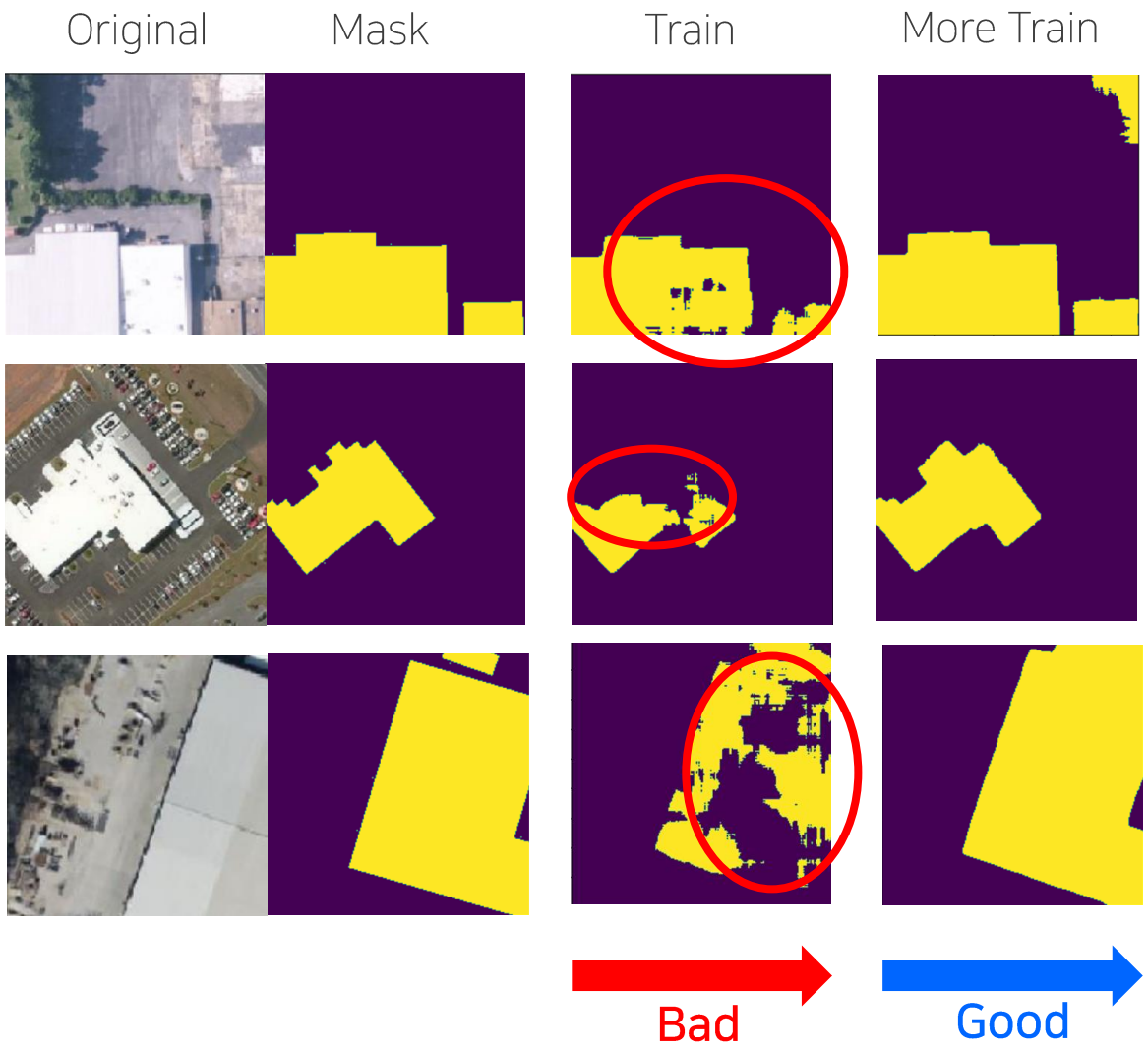
3 K-Fold + Class 균형(Custom Stratified K-fold)



K-Fold로 Dataset을 나누면서 각 Fold의 데이터가 클래스별 일정 비율을 유지하도록 구성함.

⇒ Class가 0과 1밖에 없기때문에 데이터 편향을 **방지할 수 없음.**

⇒ **HSV와 건물 비율을 도입**해서 임의로 Class를 늘려 비율별로 구축



1 건물이 있는 이미지

단일 모델 학습 결과, 예측 마스크 테두리가

일그러지거나, 건물이 제대로 검출되지 않는 등

Segmentation이 깔끔하지 않은 경우 발생

⇒ **학습이 부족하다고 판단**하여 추가적인 훈련 진행한 결과 훨씬 GT에 가깝게 Segmentation됨을 확인

2 건물이 없는 이미지

단일 모델 학습 결과, Segmentation이 잘 된 것을 확인.

하지만 훈련을 추가적으로 진행했을 경우 오히려

건물이 있다고 판단하는 경우 가 많아짐.

⇒ 건물이 거의 없는 이미지에 대해 단일모델이 취약해서 다양한 모델들로 앙상블 기법을 적용하기로 함.
이때 **건물의 존재 여부**에 대해서만 앙상블을 적용.

4. 모델 알고리즘

4-1. 모델 구조

a. Segmentation 모델 비교

b. Backbone 선정

4-2. 손실함수

a. 손실함수 비교

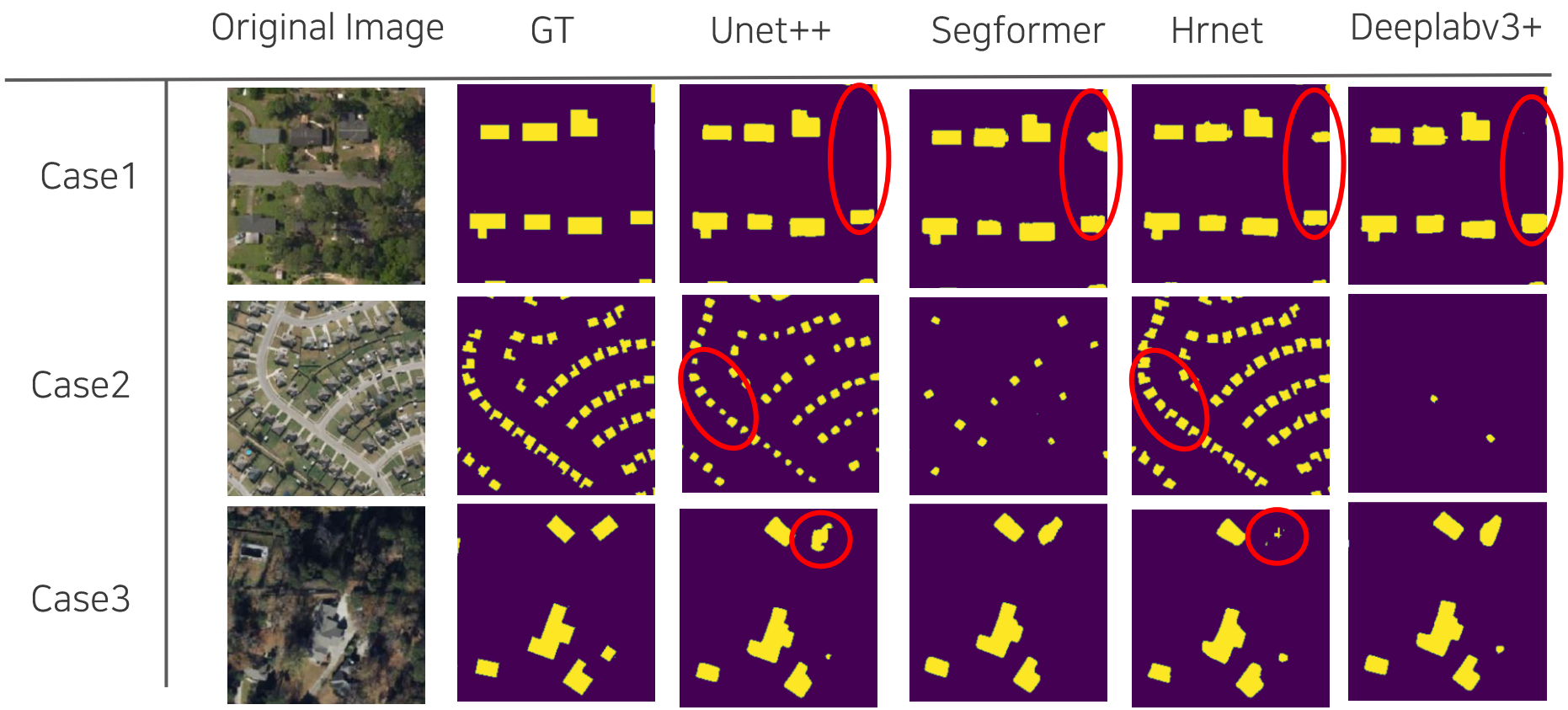
b. 실험

4-3. 성능 개선 방법

a. 앙상블

4. 모델 알고리즘

1 Segmentation 모델 비교



4. 모델 알고리즘

1 Segmentation 모델 비교

	대회 데이터셋 성능(종류)	복잡도
Unet++ & RegNet_320	0.820	높음
Segformer & MIT-B5	0.812	높음
Deeplabv3+ & RegNet_320	0.798	중간
HRnet & RegNet_320	0.816	중간

실험/논문 으로 확인해본 결과 **Unet++**가

위성 이미지 Segmentation에 적합하다고 판단

2 왜 RegNet을 Backbone으로 사용했는가

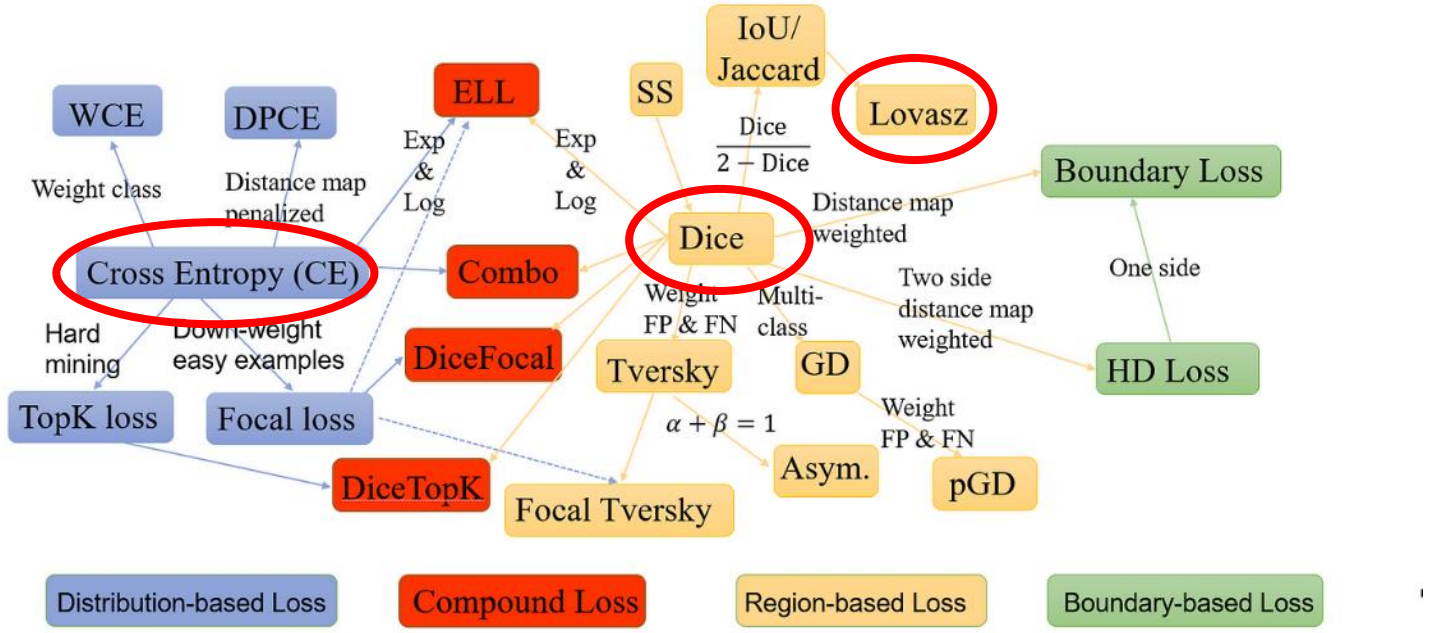
- RegNet은 ResNet과 비교하여 동일한 성능 수준을 유지하면서 더 작은 모델 크기를 가짐. **즉 더 적은 파라미터**를 사용해도 성능 하락이 없고 **Over Fitting 문제 완화** => **메모리 사용량, 계산 비용, 속도** 면에서 효율적임

	flops (B)	params (M)	acts (M)	infer (ms)	train (hr)	top-1 error ours±std [orig]
RESNET-101	7.8	44.6	16.2	90	20.4	21.4±0.11 [22.0]
RESNEXT-101	8.0	44.2	21.2	137	31.8	**20.7**±0.08 [21.2]
REGNETX-8.0GF	8.0	39.6	14.1	94	22.6	**20.7**±0.07
RESNET-152	11.5	60.2	22.6	130	29.2	20.9±0.12 [21.6]
RESNEXT-152	11.7	60.0	29.7	197	45.7	20.4±0.06 [21.1]
REGNETX-12GF	12.1	46.1	21.4	137	32.9	**20.3**±0.04

[7]RegNet

- 실제 Submit에서도 ResNet을 Backbone으로 사용한 모델보다 **RegNet을 사용한 모델의 성능이 더 좋은 것을 확인**

1 손실함수 비교



[6] Loss

- **Distribution-based Loss** : 속도가 빠르고 안정적.(ex. CE, BCE)
- **Region-based Loss** : 영역을 기반으로 계산, 배경과 건물의 대비가 클 경우 좋음.(ex, Dice, Lovasz)
- **Hybrid Loss** : 손실함수의 다양함과 안정성을 가지지만 속도가 느림.(ex. BCE + Dice)

4. 모델 알고리즘

2 손실함수별 Dice Score 비교 실험

- Base model : RegNet320(Encoder) + Unet++(Decoder)
- Epoch : 50 Epoch(실험) / 140 Epoch(메인)

	BCE_logistic	Dice	BCE + Dice	BCE + Lovaz
Dice Score	0.641(std ± 0.02)	0.62(std ± 0.05)	0.643(std ± 0.04)	0.638(std ± 0.04)

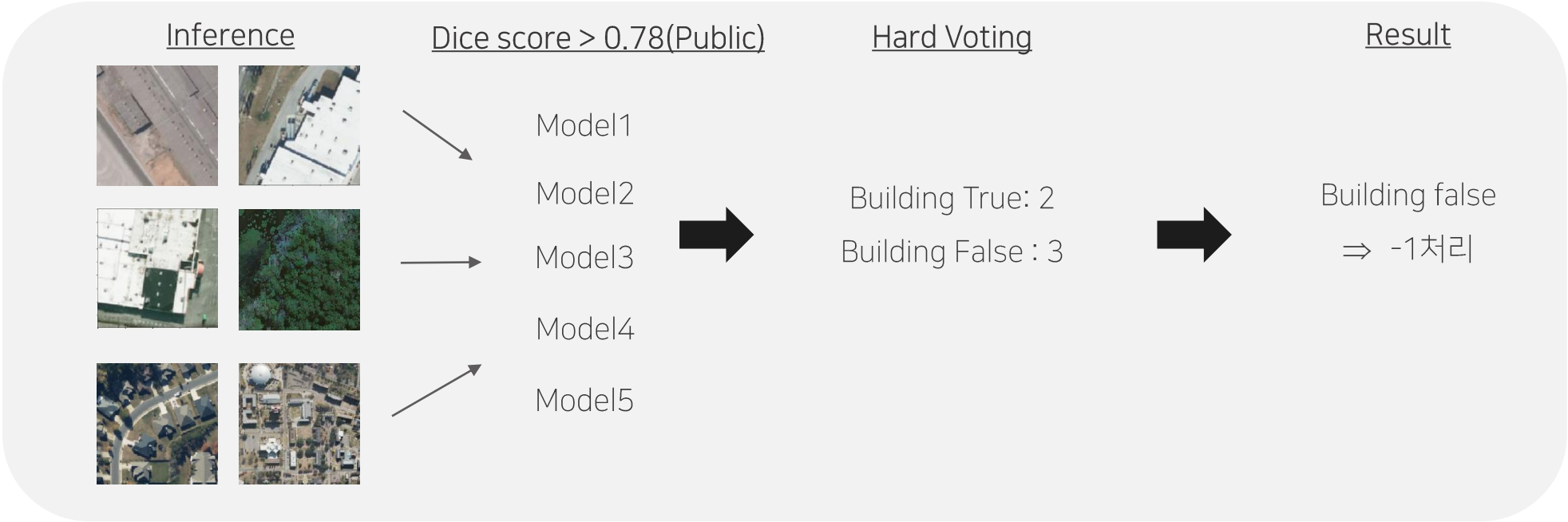
⇒ BCE+DICE가 제일 좋은 성능을 보여줬지만, **표준 편차가 제일 작은 BCE logistic**이 **안정적일 것이라고 판단**해서 선택

● 앙상블 기법(Ensemble Method) – 건물 유무 판단

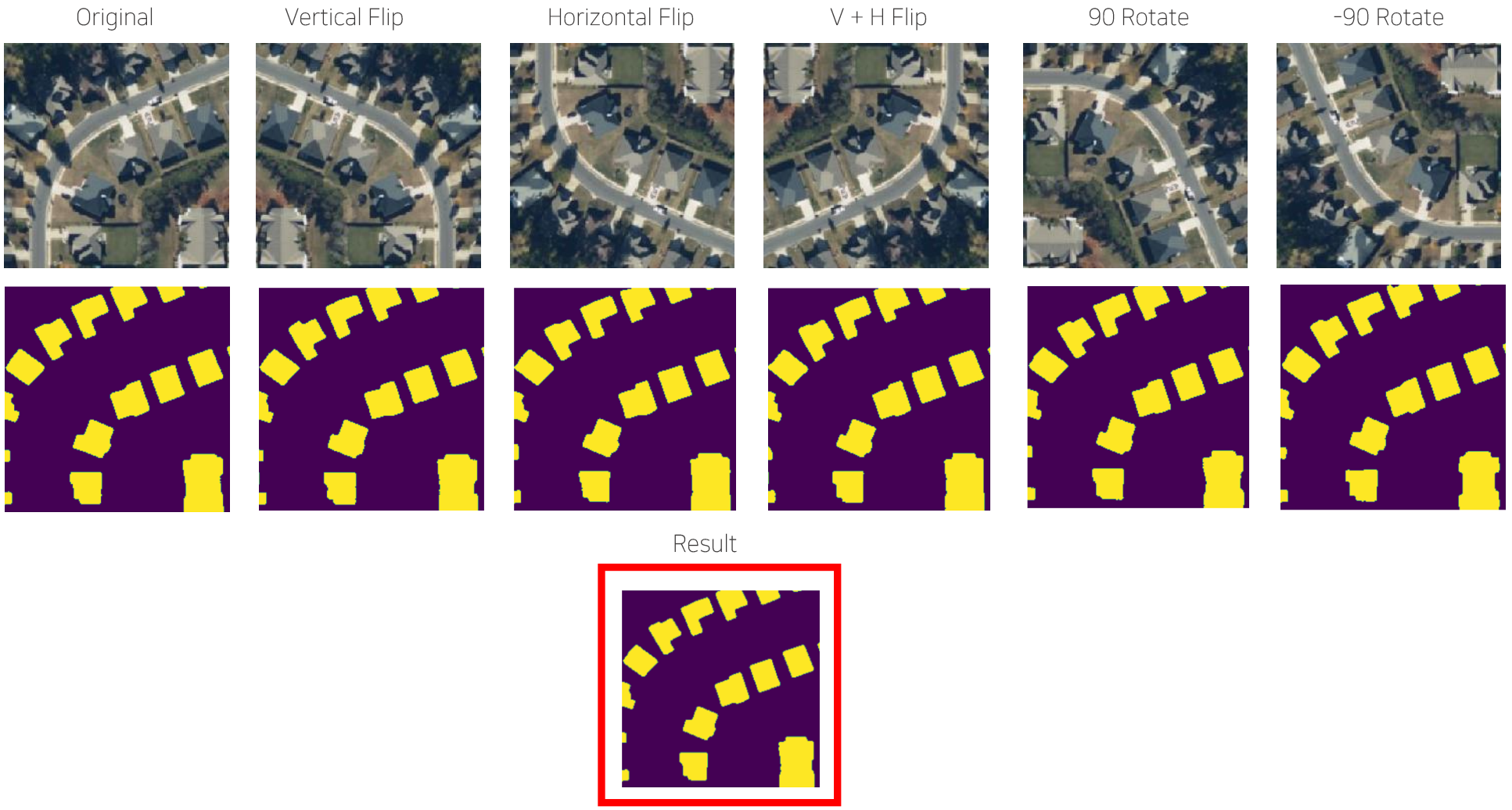
건물 여부에 대해 정확히 판단을 하기 위해 서로 다른 5개의 모델에서 건물이 없다고 판단한 이미지들 중 투표를 통해 최종적으로 건물이 없다고(-1) 판단.

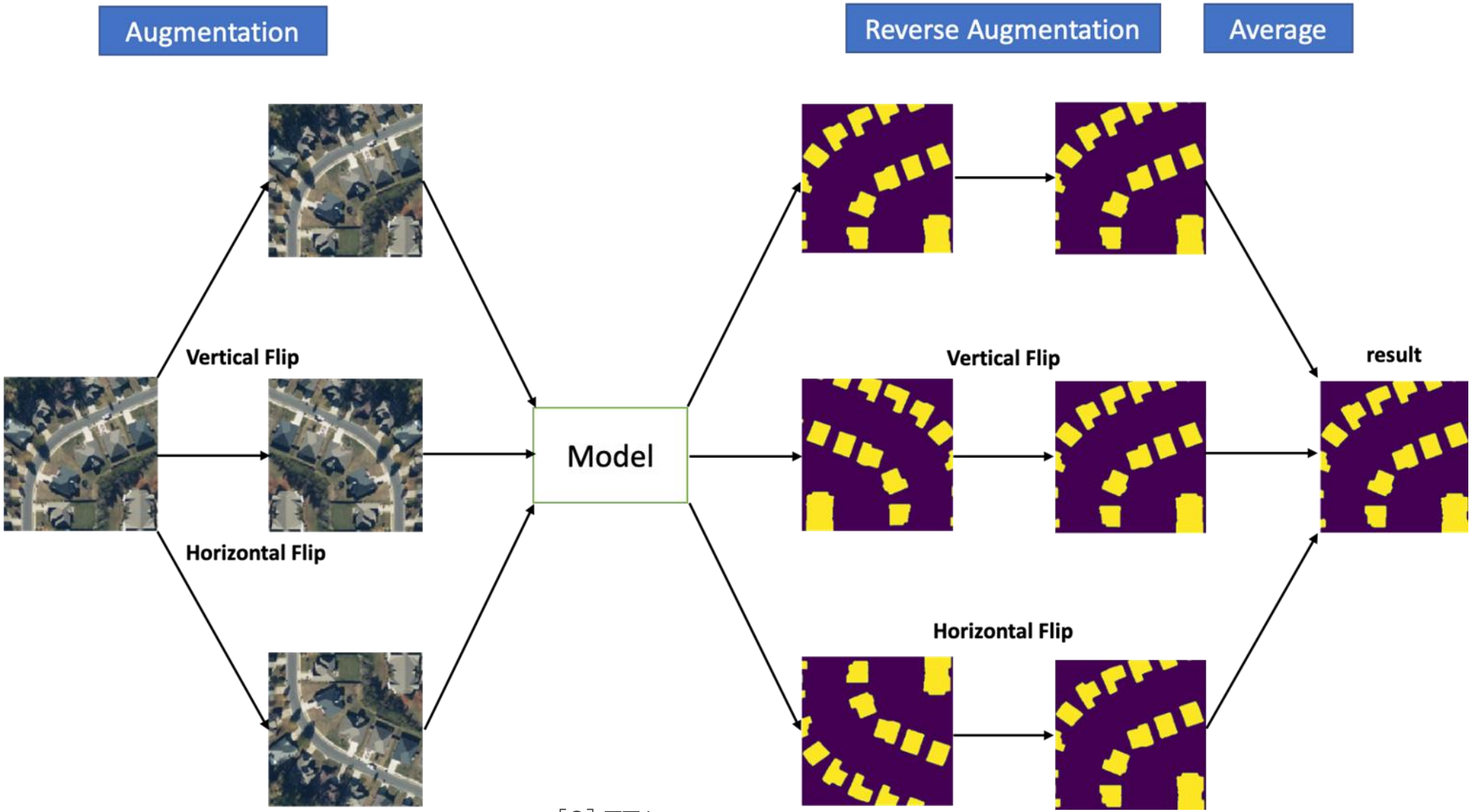
⇒ 가장 성능 좋은 모델과 다른 네트워크를 가진 5개 모델을 **Hard Voting 앙상블**

⇒ 0.822 에서 0.826 으로 성능 향상



● 앙상블 기법(Ensemble Method) - TTA (Test Time Augmentation)




Augmentation
Reverse Augmentation
Average
Vertical Flip
Horizontal Flip
Model
Vertical Flip
Horizontal Flip
result

[2] TTA

5. 결론

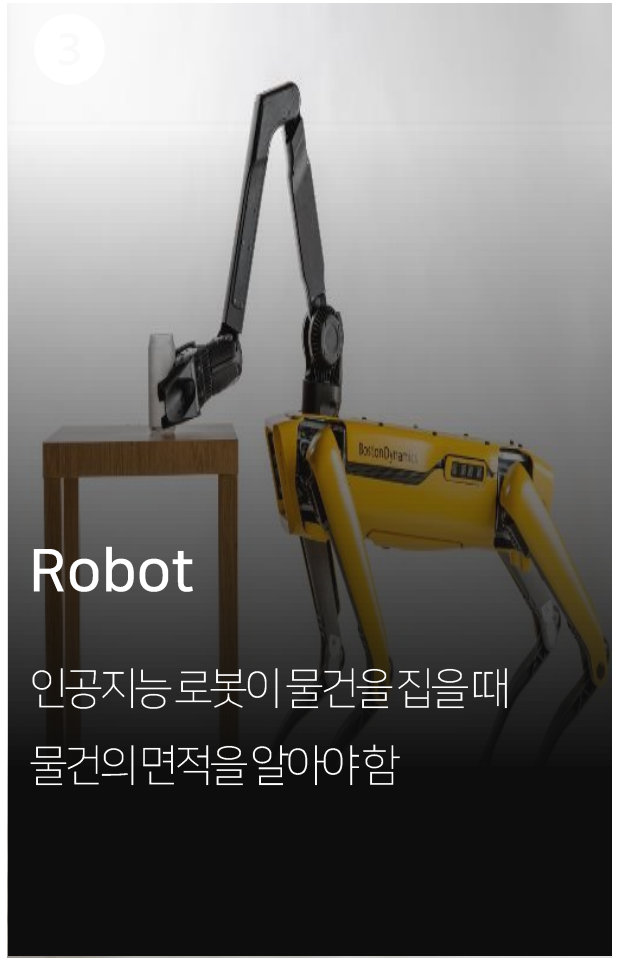
5-1. 모델 활용

5-2. 현업에서의 적용 가능성

5-3. 결론 및 향후 계획

[17] model_uses_1

[18] model_uses_2



[19] model_uses_3

[8] application_1

[9] application_2

[10] application_3

5. 결론

● 최종 모델

- **모델** : Unet++ & timm-regnety_320(backbone)
- **데이터셋** : 512x512 (84260장)
- **손실함수** : BCEwithLogitsLoss
- **옵티마이저** : Adam
- **하이퍼 파라미터**
  - **에포크 수** : 140
  - **훈련 시간** : 36시간
  - **학습률** : 1e-03
  - **스케줄러** : CosineAnnealingLR
  - **배치 크기** : 64
- **최종 성능 결과 : 0.82541(Private)**

● 결론

전처리 단계에서 **OBA기법**을 사용하여 건물이 있는 데이터를 추가적으로 늘릴 수 있었음.

추론 단계에서 **TTA(Test Time Augmentation)**를 사용하여 단일 모델의 성능을 최대화 할 수 있었음.

건물의 유무를 판단하는 단계에서 **Ensemble 기법**을 사용하였고, 이 덕분에 FP(False Positive)를 감소시킬 수 있었음.

● 향후 계획

1. 현재 모델은 초록색 지붕을 반듯하게 Segmentation 하지 못하는데, Training Dataset에서 HSV를 이용하여 **초록색 지붕과 같은 취약한 이미지들을 추출한 후 데이터셋을 늘려서 학습을 진행(Fine Tunning)** 할 것.

2. 여러 모델들을 사용해서 **Segmentation Mask에도 앙상블 기법을 적용**하면 성능 향상을 기대 할 수 있음.

3. **건물 윤곽선 클래스**를 새로 만들어 학습에 적용하면 건물을 더욱 반듯하게 Segmentation할 수 있을 것.

Reference

[1][TTA]Divya Shanmugam, Davis Blalock, Guha Balakrishnan, John Guttag, Better Aggregation in Test-Time Augmentation, Oct 2021, https://arxiv.org/abs/2011.11156

[2][TTA] https://www.kaggle.com/code/joshi98kishan/let-s-understand-tta-in-segmentation

[3][OBA]Svetlana Illarionova, Sergey Nesteruk, Dmitrii Shadrin, Vladimir Ignatiev, Mariia Pukalchik, Ivan Oseledets, Object-Based Augmentation Improves Quality of Remote Sensing Semantic Segmentation, Nov 2022, https://arxiv.org/abs/2105.05516

[4][RICAP]Ryo Takahashi, Takashi Matsubara, Kuniaki Uehara, RICAP: Random Image Cropping and Patching Data Augmentation for Deep CNNs, 2018, https://proceedings.mlr.press/v95/takahashi18a.html

[5][LABELME] https://github.com/wkentaro/labelme

[6][Region based loss]

[7][Regnet] Jing Xu, Yu Pan, Xinglin Pan, Steven Hoi, Zhang Yi, Zenglin Xu, RegNet: Self-Regulated Network for Image Classification, Jan 2021, https://arxiv.org/abs/2101.00590

[8][application_1] https://m.news1.kr/articles/?5109572

[9] [application_2] https://m.yonhapnewstv.co.kr/news/MYH20230314010600641

[10][application_3] http://www.jbnews.com/news/articleView.html?idxno=1366405

[11] [Loss] https://www.kaggle.com/code/sungjunghwan/loss-function-of-image-segmentation
[12][ppt template] Saebyeol Yu. Saebyeol's Powerpoint
[13] [dice score image] https://fakecan.tistory.com/64
[14] [segmentation image] Daniel Bolya, Chong Zhou, Fanyi Xiao, Yong Jae Lee, YOLACT: Real-time Instance Segmentatio, Oct 2019, https://arxiv.org/abs/1904.02689
[15][Loss]https://www.geeksforgeeks.org/region-and-edge-based-segmentaion/

t

Reference

[16][model_uses_1] https://hsun-100hsun-100.tistory.com/7983473
[17][model_uses_2] https://flpan.tistory.com/275
[18][model_uses_3] https://www.techm.kr/news/articleView.html?idxno=5358

t