1. 프로젝트 개요

📌 프로젝트 개요

Problem Type. 사진에서 쓰레기 Detection 및 10 종류(`General trash`, `Paper`, `Paper pack`, `Metal`, `Glass`, `Plastic`, `Styrofoam`, `Plastic bag`, `Battery`, `Clothing`) 쓰레기 분류
Metric. mAP 50

Data. 총 9,754장의 이미지 (1024 x 1024), annotation file (coco format)

📌 개발환경 & 협업툴

- **개발환경**

개발환경	버전
VSCode	1.60.0
PyTorch (GPU)	1.7.1
mmdetection	

- **협업 Tool**
  GitHub, Wandb, Notion

📌 GitHub file & directory

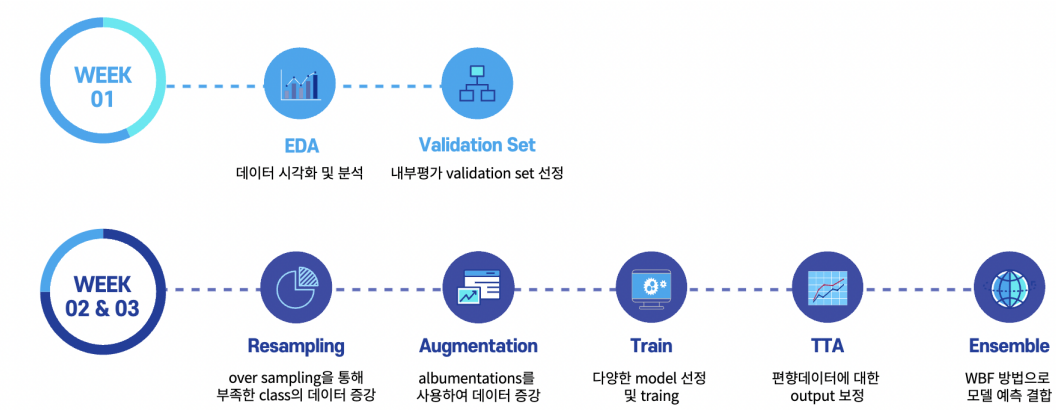
```
baseline
├── mmdetection
|   ├── config
|   └── ...
├── requirements.txt
|
└── yolo
    ├── dataset
    |   ├──images
    |   ├──labels
    └── cocotrash.yaml
```

2. 팀 구성 및 역할

📌 팀 구성 및 역할

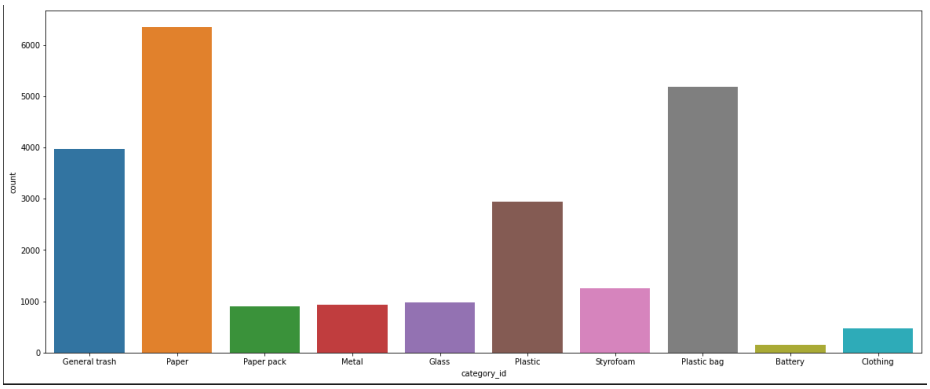
- ***Dataset Part*** → ***김창현***
- ***Model Part***
  - 1-Stage → ***강소망, 김기태***
  - 2-Stage → ***박민수, 박기련, 김창현***

3. 프로젝트 수행 절차 및 방법

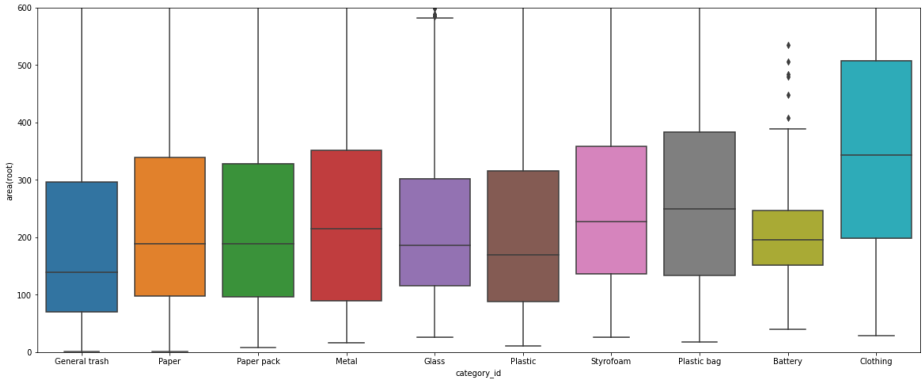


1 EDA

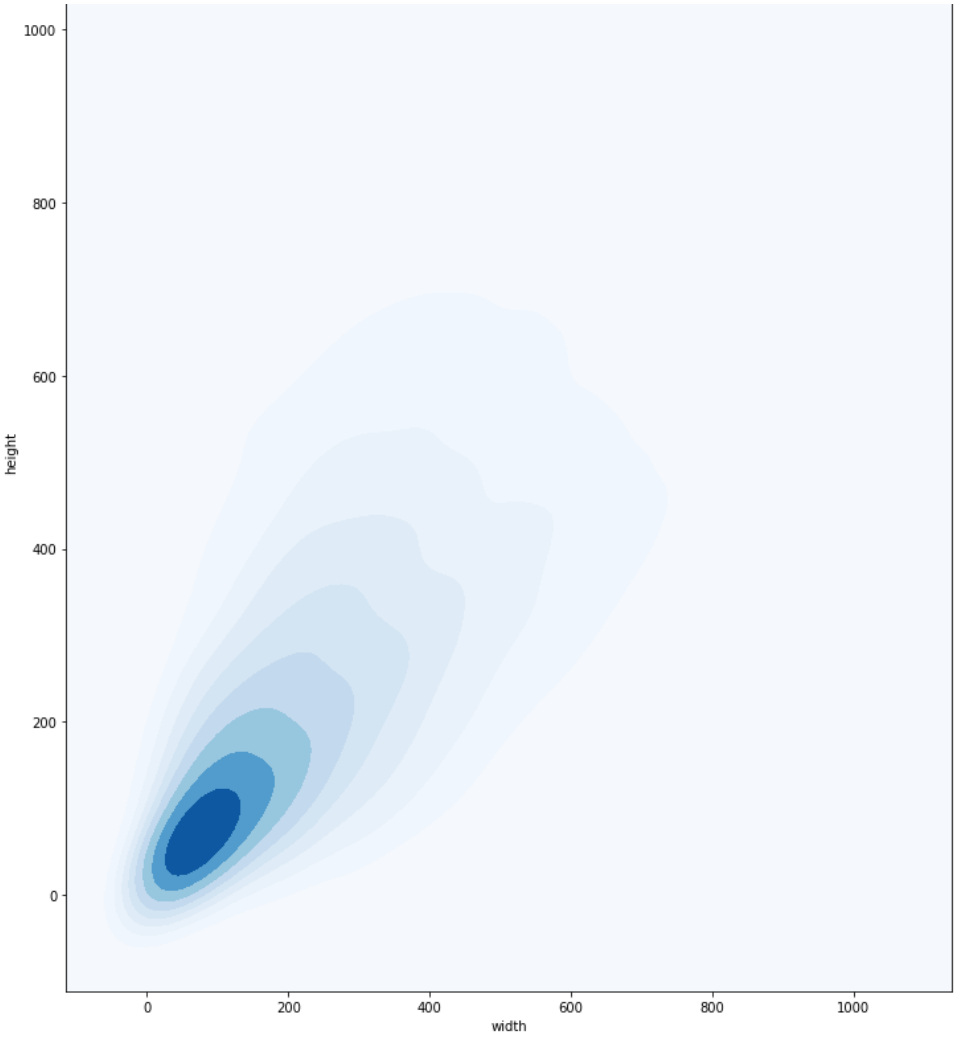
- Class별 분포 : 불균형으로 인한 문제가 발생할 여지가 있음
- Augmentation, Sampling, Loss measurement 를 통해 해결할 필요가 있음



- Class별 box area 통계치
- General Trash가 평균적으로 작은 Bbox, clothing이 큰 Bbox



- 큰 Bbox보다 작은 Bbox 분포가 높다는 것을 확인할 수 있음



2️⃣ *내부 평가 지표 설정*

📌 Validation Set 통일

- 하루 10회로 한정된 제출 기회를 효율적으로 사용하기 위해, 내부 평가 지표 설정
- Stratified K-Fold 를 이용하여 Class 를 균등하게 나눈 후, Fold 별로 동일한 모델 학습
- LB Score 와 가장 근사한 Validation mAP50 를 가지는 Dataset을 공통 평가 지표로 설정

3️⃣ *불균형 클래스 문제*

📌 *Over Sampling*

- 가설
  - EDA결과 2,3,4,8,9번 카테고리의 데이터가 다른 클래스에 비해 현저히 적은 것을 확인하고 해당 카테고리의 데이터를 증강
  - 클래스별 분포 : `[3165, 4939, 710, 758, 765, 2372, 1036, 4073, 139, 386]`
- 결과
  - 기존 데이터셋에서 위의 카테고리를 Over Sampling한 결과 LB상 0.02 상승
  - 2,3,4번 카테고리를 3배 8,9는 2배 0.0008 상승(8, 9번 카테고리는 데이터가 제일 적긴하지만 제일 높은 정확도로 예측해 더 이상 늘리지 않음)
- 정리
  - Over sampling을 적용한 후 LB의 mAP50의 유의미한 상승을 확인. 그러나 데이터 증강의 수가 특정 수준을 넘어가게 되면 미미한 영향을 미치거나 오히려 하락
  - Over sampling을 한 데이터에 Augmentation상에서는 들어가지 않는 옵션들로 미리 적용해 더 다양한 데이터로 만들었으면 좋았을거라는 아쉬움이 듦

4️⃣ *Augmentation*

📌 Albumentation

1. **RandomRotate90 , RandomFlip (Horizion, Vertical)**
   - Rotate 를 시켜도 Object 의 형태는 동일하므로 데이터 증강의 목적으로 사용
2. **HueSaturationValue, RandomGamma, CLAHE [One of]**
   - 여러가지 밝기에 따른 일반화 성능 향상 도모
3. **Blur, GaussianNoise, MotionBlur [One of]**
   - 초점이 흐린 Image가 들어올 것을 대비

Result

Augmentation 적용 전 : Swin_T Cascade R-CNN **(0.5419)**
Augmentation 적용 후 : Swin_T Cascade R-CNN **(0.5949)** → **약 0.05 상승**

5️⃣ *WBF (Ensemble)*

📌 WBF

- 가장 정확한 Bbox 하나만을 남기는 NMS나 NMW와 달리, WBF는 예측된 Bbox의 정보를 모두 사용하기 때문에 여러개 모델 앙상블 시 더 유리하다 판단

4. 프로젝트 수행 결과 및 분석

📌 Result

	BackBone	Model	Val mAP50	LB Score
1	**Resnet101**	Cascade R-CNN	0.5290	
2		Grid R-CNN	0.5054	
3		ATSS + DyHead	0.4934	
4		UniverseNet	0.6070	0.6134
5		Centernet2	0.5860	0.5927
6	**Swin_T**	Cascade R-CNN	0.5810	0.5949
7		HTC	0.5899	0.6003
8	**Swin_L (1K)**	Cascade R-CNN	0.6520	0.6849
9		HTC	0.6660	0.6819
10		Centernet2	0.6160	0.6184
11		ATSS + DyHead	0.6450	0.6660
12	**Swin_L(22K)**	HTC	0.6510	0.6722
13	**YOLO**	YOLOx	0.4280	
14		YOLOv5x6	0.5600	
15		YOLOv5l6	0.5900	

📌 Ensemble

- **선정 기준**
  - val_mAP50 0.6 ↑ (YOLO의 경우 0.5)
  - val_mAP_s 0.001 ↑
- **선정 모델 (mAP50)**
  - 1. Resnet101 - UniverseNet (0.6070)
  - 8. Swin_L(1K) - Cascade R-CNN (0.6520)
  - 9. Swin_L(1K) - HTC (0.6660)
  - 10. Swin_L(1K) - Centernet2 (0.6160)
  - 12. Swin_L(22K) - HTC (0.6510)
  - 14. YOLO - YOLOv5x6 (0.5600)
  - 15. YOLO - YOLOv5l6 (0.5900)
- **최종 LB Score**

  Public : 0.7207 **Private : 0.7059 (5등 / 19 team)**

📌 **분석**

- **TTA (Flip)** 와 **Augmentation** 을 모든 실험에 적용
- **BackBone** 을 ResNet → Swin Transformer 로 키워나감으로써 **성능** ↑
- 전체적으로 **Swin_L** 기반 Model 성능이 압도적
- Small Size 객체 탐지율을 높이기 위하여 YOLO 를 사용하여 Ensemble

5. 자체 평가 의견

⚪ **잘한 점들**

- Wandb 시각화 , 결과 공유
- 실험 설계 및 분석
- Notion으로 체계적인 실험 관리
- 각기 다른 특징의 모델로 실험
- 협업

⚪ **아쉬웠던 점들**

- Inference 결과를 체계적으로 관리하지 못함
- Github을 활용하지 못함
- 실험 description을 자세히 적지 못함
- 디테일하지 못한 EDA

⚪ **시도했으나 잘 되지 않았던 것들**

- EfficientDet을 시도하려 했으나 에러가 발생하거나 결과가 제대로 출력이 안되었음
- Instaboost를 시도하려 했으나 Segment 정보가 없어서 하지 못함
- Convnext 를 BackBone으로 사용하려 했으나, OOM 문제로 실패
- Model Soup 를 시도하려 했으나 체크포인트 파일 백업부재로 실패

⚪ **프로젝트를 통해 배운 점 또는 시사점**

- mmDetection을 활용한 object detection 전반적인 과정 학습을 경험할 수 있었음
- 단일 모델의 성능도 중요하지만, 모델 간 시너지를 높이는 Ensemble의 중요성을 깨달음
- Detection Task 의 모델에서 Feautre Map 추출하는 Backbone 이 성능에 가장 많은 영향을 미친다는 것을 알게됨