Wrap UP 리포트

프로젝트 주제	글자 검출 대회	
	이름	**캠퍼 ID**
팀 장	유승리	T3129
팀 원	이창진	T3169
	심준교	T3124
	전영우	T3192
	송민수	T3113
	김하준	T3066

프로젝트 설명

○ 프로젝트 개요

스마트폰으로 카드를 결제하거나, 카메라로 카드를 인식할 경우 자동으로 카드 번호가 입력되는 경우가 있습니다. 또 주차장에 들어가면 차량 번호가 자동으로 인식되는 경우도 흔히 있습니다. 이처럼 OCR (Optical Character Recognition) 기술은 사람이 직접 쓰거나 이미지 속에 있는 문자를 얻은 다음 이를 컴퓨터가 인식할 수 있도록 하는 기술로, 컴퓨터 비전 분야에서 현재 널리 쓰이는 대표적인 기술 중 하나입니다.

이번 프로젝트에서는 OCR 기술 중 글자 검출 task 만을 진행하게 되며, 모델의 변경 없이, data 만을 변형하거나, 추가하여 검출 성능을 향상시키는 data centric ai 관점으로 프로젝트를 진행합니다.

○ 개발 환경

- CPU : Intel xeon
- GPU : V100
- OS : Ubuntu 18.04
- 개발툴 : Vscode, Jupyerlab
- 협업툴 : Github, Slack, Notion
- 라이브러리 버전 : Python 3.6, Pytorch 1.7.1

○ 베이스라인 모델

EAST (An Efficient and Accurate Scene Text Detector)

프로젝트 수행 내용

○ 프로젝트 구조 및 데이터 셋의 구조도

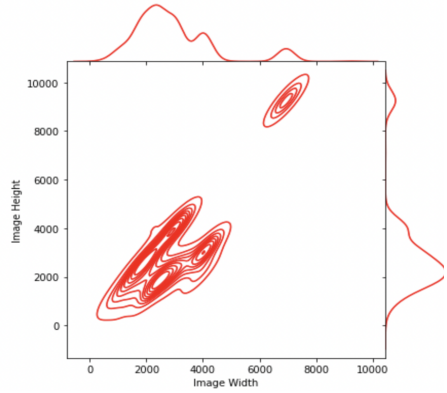
```
.
`-- code
    |-- OCR_EDA.ipynb
    |-- convert_mlt.py
    |-- custom_aug.py
    |-- dataset.py
    |-- detect.py
    |-- deteval.py
    |-- download_ICDAR.sh
    |-- east_dataset.py
    |-- inference.py
    |-- loss.py
    |-- model.py
    |-- pths/
    |-- requirements.txt
    |-- seed.py
    |-- train.py
    |-- trained_models/
    |-- urls.txt
    `-- utils
```

```
.
`-- input
    `-- data
        |-- AIStages_ANN
        |   |-- dataset
        |   `-- ufo
        |-- ICDAR17_Korean
        |   |-- images
        |   `-- ufo
        |-- ICDAR17_MLT
        |   |-- images
        |   |-- raw
        |   `-- ufo
        |-- ICDAR19_MLT
        |   |-- images
        |   |-- raw
        |   `-- ufo
        `-- Modified
            |-- images
            `-- ufo
```

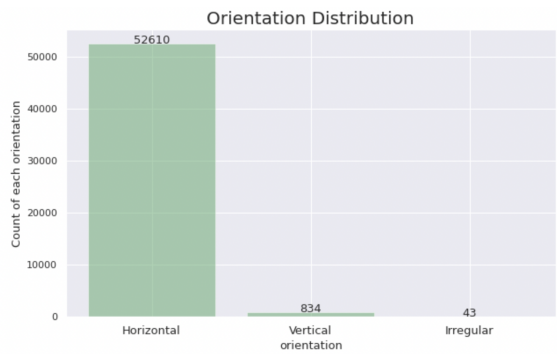
○ 기대 효과

- OCR 객체 검출 모델인 EAST 모델에 대한 이해
- data annotation 에 대한 이해
- 모델의 성능향상에 data 가 미치는 영향
- 추가 data 활용 및 augmentation 에 대한 이해

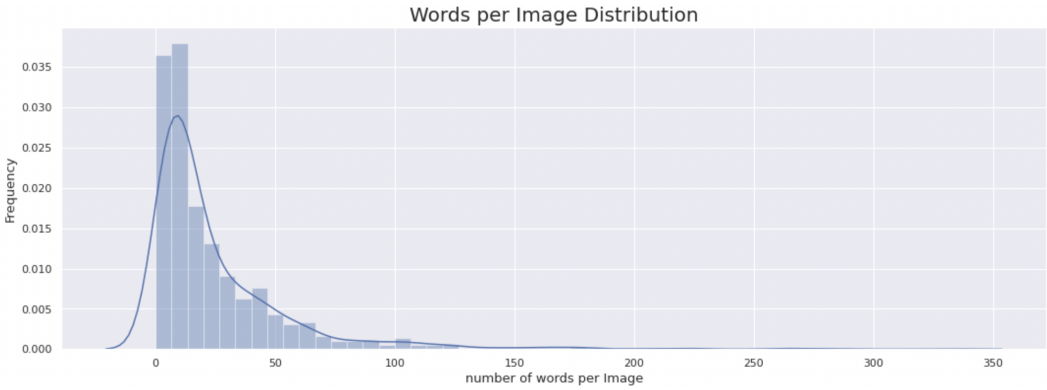
○ 탐색적 분석 및 데이터 전처리(EDA) - 학습 데이터 소개



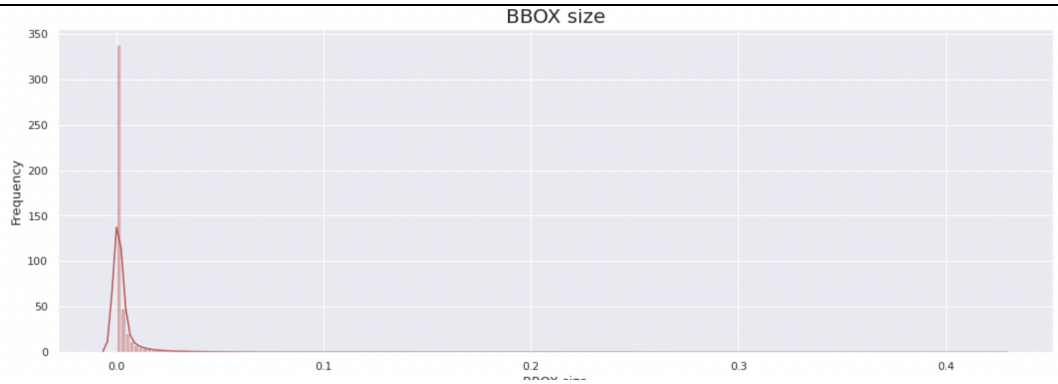
<이미지 높이, 넓이에 따른 분포도>



<Word 의 방향에 따른 개수 분포>



<이미지 당 word 개수 분포>



<bbox size 에 따른 분포>

○ **데이터 전처리**

- Dataset 추가

  i. MLT 2017 추가

  text 객체 검출의 성능향상을 위해선, 단순히 한글 image 뿐만 아니라, 다른 언어들에 대한 data 가 주어질수록 robust 하게 글자를 검출할 수 있게 된다. 그에 따라서, MLT_Korean 이 아닌 MLT 2017 dataset 전체를 사용하기로 하였고, 그 중 일부를 선택하기로 하였다.

  ii. MLT 2019 추가

  이미지 당 단어의 개수를 10 개 이상으로 기준을 정하게 되면 약 85% 의 이미지가 학습에서 배제 된다. 또한 test set 의 분포 또한 정확히 알 수 없는 상황에서 단어를 무조건 제거하기 보다는 모두 학습에 사용해보았다. 또한 MLT 2019 는 2017 에 비해 굴곡진 단어가 많다고 알려져 있기에 선택하였다. 해당 방법만으로도 대상 이미지가 10,000 개이기에 추가적인 데이터는 사용하지 않았다.

  iii. AIStages Annotation 추가

  기본적으로 존재하였던 MLT_Korean 은 총 536 개 sample 로, 이는 training 시 좋은 성능을 보장할 수 없기 때문에, 추가적인 한글 dataset 이 필요하였다. 이에 따라, ai stage 에서 제공한 annotation tool 을 활용하여 직접 한글 data 를 annotation 을 진행하였고, 이를 추가적인 dataset 으로 활용하였다.

- K Fold Cross Validation

  MLT 2017 과 AIStages 데이터를 대상으로 k fold cross validation 을 수행하기 위해 sklearn 에서 제공하는 StratifiedGroupKFold 를 사용하였다. 우선, EDA 및 시각화를 통해 각 annotation 의 bbox size 와 분포를 분석한 후, 그 값이 0.0005 미만이거나 0.27 이상인 bbox 는 삭제하였다. 그리고 bbox size 를 오름차순 정렬 후 30%, 60%, 80%, 95%를 기준으로 총 5 개의 class(A~E)로 나누었고, group 을 image name 으로 지정하여 StratifiedGroupKFold(K=5)를 진행하였다. 따라서 각 fold 의 train 과 validation set 에서 5 개 class 의 비율이 유지되면서도, 그 사이에 중복되는 image 가 없도록 cross validation set 을 생성하였다.

bbox_size	count	mean	std	min	max
(-0.0005, 0.0011]	10477	0.000760	0.000171	0.000500	0.001104
(0.0011, 0.0027]	10477	0.001741	0.000454	0.001104	0.002703
(0.0027, 0.00721]	6984	0.004383	0.001255	0.002704	0.007207
(0.00721, 0.0345]	5238	0.015322	0.007122	0.007207	0.034474
(0.0345, 0.27]	1747	0.078088	0.048079	0.034489	0.269680

	A	B	C	D	E
training set	30.00%	30.00%	20.00%	15.00%	5.00%
train - fold0	29.98%	30.19%	19.97%	14.96%	4.91%
val - fold0	30.11%	29.18%	20.14%	15.17%	5.40%
train - fold1	29.73%	29.99%	20.05%	15.25%	4.98%
val - fold1	31.03%	30.05%	19.79%	14.03%	5.10%
train - fold2	30.37%	30.05%	19.86%	14.56%	5.16%
val - fold2	28.70%	29.82%	20.48%	16.54%	4.46%
train - fold3	29.83%	29.94%	20.03%	15.15%	5.05%
val - fold3	30.69%	30.23%	19.87%	14.38%	4.83%
train - fold4	30.10%	29.83%	20.08%	15.06%	4.93%
val - fold4	29.56%	30.73%	19.66%	14.72%	5.33%

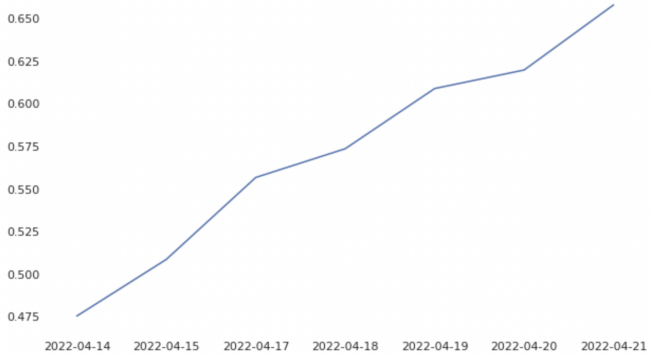
```
[Epoch 13]: 100%| | 51/51 [05:04<00:00,  5.98s/it, Cls loss=0.442, Angle loss=0.277, IoU loss=1, Learning Rate=0.001]
 Train 13/200 - Mean loss: 1.5857, Cls loss: 0.3941, Angle loss: 0.2053, IoU loss: 0.9863 | Elapsed time: 0:05:04.756539
Calculating validation results...
[Epoch 13]: 100%| | 13/13 [01:15<00:00,  5.81s/it, Cls loss=0.319, Angle loss=0.185, IoU loss=0.908]
 Validation 13/200 - Mean loss: 1.6237, Best val loss: 1.5837 | Elapsed time: 0:01:15.527474
[Epoch 14]: 100%| | 51/51 [04:45<00:00,  5.59s/it, Cls loss=0.35, Angle loss=0.0325, IoU loss=0.918, Learning Rate=0.001]
 Train 14/200 - Mean loss: 1.5419, Cls loss: 0.3801, Angle loss: 0.2012, IoU loss: 0.9606 | Elapsed time: 0:04:45.099670
Calculating validation results...
[Epoch 14]: 100%| | 13/13 [01:25<00:00,  6.59s/it, Cls loss=0.411, Angle loss=0.256, IoU loss=1.07]
 Validation 14/200 - Mean loss: 1.6513, Best val loss: 1.5837 | Elapsed time: 0:01:25.613516
[Epoch 15]: 100%| | 51/51 [04:43<00:00,  5.55s/it, Cls loss=0.473, Angle loss=0.457, IoU loss=0.98, Learning Rate=0.001]
 Train 15/200 - Mean loss: 1.4922, Cls loss: 0.3738, Angle loss: 0.1871, IoU loss: 0.9313 | Elapsed time: 0:04:43.294680
Calculating validation results...
[Epoch 15]: 100%| | 13/13 [01:14<00:00,  5.69s/it, Cls loss=0.317, Angle loss=0.0388, IoU loss=0.78]
 Validation 15/200 - Mean loss: 1.5812, Best val loss: 1.5837 | Elapsed time: 0:01:14.018938
Best val loss at epoch 15! Saving the model to trained_models/best_val_loss.pth...
```

○ 실험 과정

- 기본적으로 주어진 536 장의 데이터셋으로 베이스라인 코드 실행 및 제출 결과 precision 에 비해 recall 값이 매우 낮게 나왔다. 또한 AIStages 데이터와 MLT 2017 데이터를 전부 추가하였을 때는 학습 시간이 너무 오래 걸리는 문제가 생겼다. 따라서 데이터셋을 선택적으로 구성하여 최소한의 데이터로 recall 을 높이기 위한 목적으로 여러 실험을 진행하였다.
- MLT 2017 데이터셋의 경우 한글이 포함된 이미지는 전부 사용했으나, 영어와 다른 언어로만 구성된 이미지의 경우 이미지 내 단어 개수가 n 개 이상인 데이터만 가져오도록 하여 n 을 40,30,20 등으로 바꿔가며 결과를 비교하였다. AIStages 데이터의 경우 캠퍼들이 직접 어노테이션을 수행했기 때문에 실수가 굉장히 많아 잘못 수행된 데이터를 제거한 뒤, 마찬가지로 이미지 내 단어 개수가 n 개 이상인 데이터만 가져오도록 하여 n 을 15, 10 등으로 바꿔가며 결과를 비교하였다.

프로젝트 수행 결과

○ 날짜 별 LB score 변화도



○ 최종 점수

- [Public] f1 : 0.6583, recall : 0.5724, precision : 0.7745
- [Private] f1 : 0.6377, recall : 0.5630, precision : 0.7352

자체 평가 의견	**○ 잘한 점** 최소한의 데이터셋으로 recall 을 높이기 위한 실험을 진행했는데 실제 제출 결과 precision 은 낮아졌지만 recall 값이 크게 올라 f1 score 가 상승하는 결과가 나왔다. **○ 시도했으나 잘 되지 않았던 점** custom augmentation 구현 시, 오피스아워에서 제공되었던 augmentation 코드를 토대로 모든 cropped patch 에 온전한 bbox 가 무조건 1 개 이상 들어가도록 해보았으나 Polygon 관련 에러가 발생하였고 해결하지 못하였다. 그 후, albumentations 에서 제공하는 여러 augmentation 을 적용해보았으나 예상과 다르게 큰 도움이 되지 않았다. 그 이유는 crop 시 잘린 bbox 에 대해서 masking 처리를 하지 못했기 때문일 것이라고 추측된다. **○ 아쉬운 점** 최대한 많은 데이터를 긴 기간 학습을 통해 결과를 보는 실험을 해봤어야 했는데, 못해본 것이 아쉽다. **○ 프로젝트를 통하여 배운 점, 시사 점** ● 고품질 데이터 확보의 어려움 실제로 데이터 어노테이션을 진행해 보았는데, 시간도 오래 걸릴 뿐더러 라벨링 가이드가 주어졌음에도 잘못 라벨링하는 경우가 매우 많이 발생하였다. 고품질의 데이터셋을 구성하기 위해 많은 인력과 시간이 필요하다는 것을 직접 체험해 볼 수 있었다. ● 데이터의 양 vs 질 실험을 진행한 결과 데이터의 절대적인 양이 줄었음에도 제출 시 점수가 높은 결과가 나오는 경우가 있었다. 데이터의 절대적인 양도 중요하지만, 데이터의 품질과 다양성 또한 매우 중요함을 알 수 있었다.