

[한밭대학교] AIM. LAB.

재현과정 설명

팀원 : 이지상, 김동수, 오서연



목차

1. 재현 환경 설명

2. 이미지 크롭 및 LMDB 생성

3. 모델 훈련

4. 결과 예측

```
on:absolute;z-index:999  
x 5px #ccc}.gbrtl .gbm  
display:block;positio  
acity:1;*top:-2px;*le  
/;top:-4px\0/;left:-6  
e-box;display:inline  
isplay:block;list-s  
e-block;line-height  
pointer;display:bl  
tive;z-index:1000)  
padding-right:9px  
durl111
```

1. 재현 환경 설명

재현환경 설명

모델훈련에 사용했던 환경값들입니다.

GPU : Nvidia RTX A6000 GDDR6 48GB x 1

CUDA Version : v11.6.124

OS : Ubuntu 20.04.4 LTS (GNU/Linux 5.4.0-125-generic x86_64)

라이브러리는 제출파일중 experiment_setting폴더의 torch_requirements.txt, requirements.txt 파일로 첨부해두었습니다.

```
on:absolute;z-index:999  
x 5px #ccc}.gbrtl .gbm  
display:block;positio  
acity:1;*top:-2px;*le  
/;top:-4px\0/;left:-6  
e-box;display:inline  
isplay:block;list-s  
e-block;line-height  
pointer;display:bl  
tive;z-index:1000)  
padding-right:9px  
durl111
```

2. 이미지 크롭 및 LMDb 생성

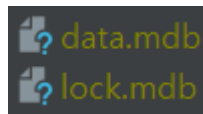
전체 개요

먼저 데이터 생성 전체적인 개요입니다.

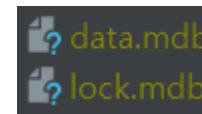
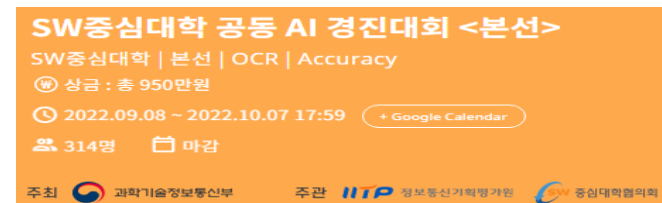
훈련데이터로 AIHub의 '야외 실제 촬영 한글 이미지'의 Training데이터를 크롭해 LMDB파일로 만들어 사용하였고

검증데이터는 본 대회와 훈련데이터를 LMDB파일로 만들어 사용하였습니다.

훈련데이터



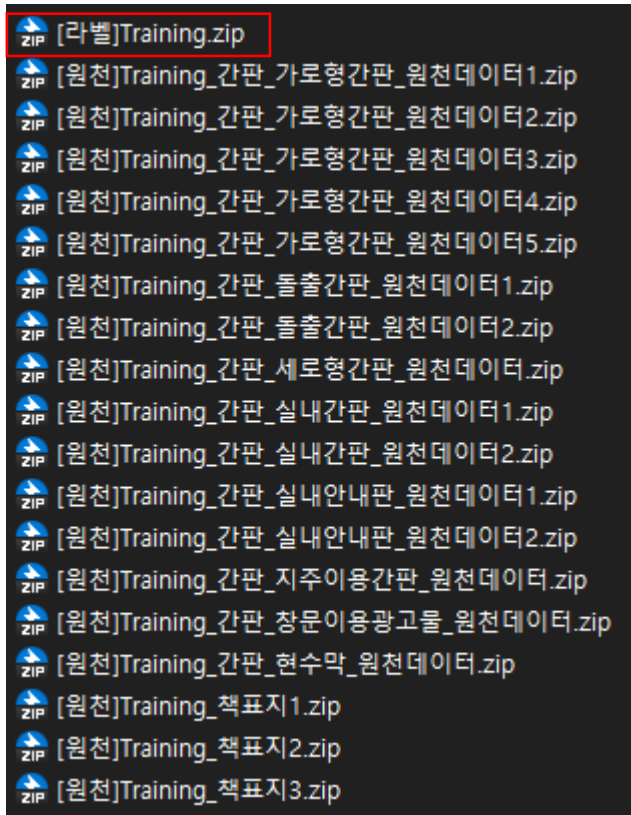
검증데이터



<https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=105>

<https://dacon.io/competitions/official/235970/data>

이미지 크롭

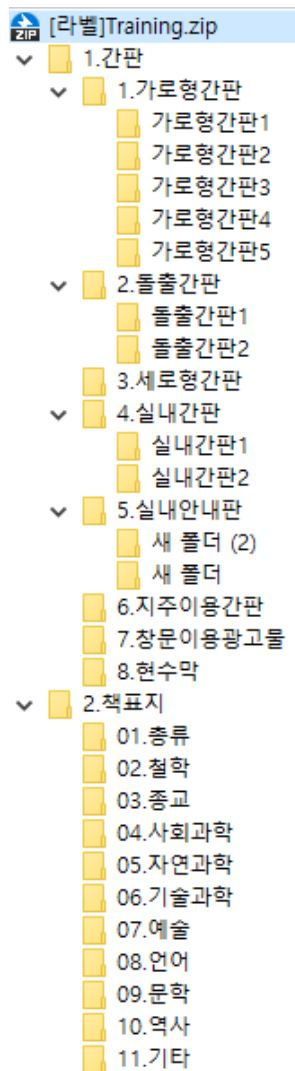


훈련LMDB를 생성하기 앞서 이미지크롭을 진행하겠습니다.

AI Hub의 야외 실제 촬영 한글 이미지 Training파일을 다운받으면 들어있는 [라벨], [원천]데이터 압축파일들입니다.

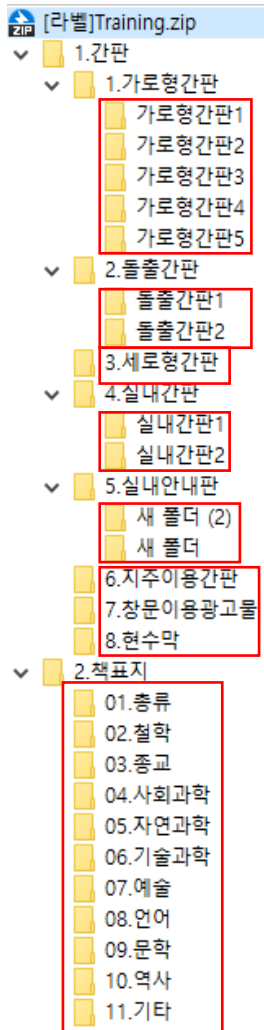
먼저 [라벨]Training파일부터 가공을 하겠습니다.

이미지 크롭



[라벨]Training 파일의 압축을 풀면 좌측의 이미지와 같은 구조로 폴더들이 이루어져있습니다.

이미지 크롭



이때 좌측에 붉은색 네모로 표시한 폴더들의 이름을 아래와같이 수정해주시면됩니다.

(아래와 이름이 모두 동일하게 하지 않아도 폴더의 **순서**가 동일하면 문제없습니다.)

가로형간판{1~5} => **간판0{1~5}.가로형간판{1~5}**

돌출간판{1~2} => **간판0{6~7}.돌출간판{1~2}**

3.세로형간판 => **간판08.세로형간판**

새 폴더 => **간판09.실내 안내판1**

새 폴더 (2) => **간판10.실내 안내판2**

실내간판{1~2} => **간판{11~12}.실내간판{1~2}**

6.지주이용간판 => **간판13.지주이용간판**

7.창문이용광고물 => **간판14.창문이용광고물**

8.현수막 => **간판15.현수막**

01.총류 => **책01.총류**

02.철학 => **책02.철학**

03.종교 => **책03.종교**

04.사회과학 => **책04.사회과학**

05.자연과학 => **책05.자연과학**

:

11.기타 => **책11.기타**

이미지 크롭

최종적으로 좌측과 같은 폴더들이 생성되게됩니다.
이 폴더들을 오른쪽 이미지같이 [라벨]Training 폴더를 생성해 넣어주시면됩니다.

■ 간판01.가로형간판1	2022-09-21 오후 10:04	파일 폴더
■ 간판02.가로형간판2	2022-09-21 오후 9:49	파일 폴더
■ 간판03.가로형간판3	2022-09-21 오후 9:49	파일 폴더
■ 간판04.가로형간판4	2022-09-21 오후 9:49	파일 폴더
■ 간판05.가로형간판5	2022-09-21 오후 9:49	파일 폴더
■ 간판06.돌출간판1	2022-09-21 오후 9:49	파일 폴더
■ 간판07.돌출간판2	2022-09-21 오후 9:49	파일 폴더
■ 간판08.세로형간판	2022-09-21 오후 9:44	파일 폴더
■ 간판09.실내 안내판1	2022-09-21 오후 9:45	파일 폴더
■ 간판10.실내 안내판2	2022-09-21 오후 9:49	파일 폴더
■ 간판11.실내간판1	2022-09-21 오후 9:45	파일 폴더
■ 간판12.실내간판2	2022-09-21 오후 9:45	파일 폴더
■ 간판13.지주이용간판	2022-09-21 오후 9:45	파일 폴더
■ 간판14.창문이용광고물	2022-09-21 오후 9:45	파일 폴더
■ 간판15.현수막	2022-09-21 오후 9:46	파일 폴더
■ 책01.총류	2022-09-21 오후 9:46	파일 폴더
■ 책02.철학	2022-09-21 오후 9:46	파일 폴더
■ 책03.종교	2022-09-21 오후 9:46	파일 폴더
■ 책04.사회과학	2022-09-21 오후 9:46	파일 폴더
■ 책05.자연과학	2022-09-21 오후 9:46	파일 폴더
■ 책06.기술과학	2022-09-21 오후 9:46	파일 폴더
■ 책07.예술	2022-09-21 오후 9:46	파일 폴더
■ 책08.언어	2022-09-21 오후 9:46	파일 폴더
■ 책09.문학	2022-09-21 오후 9:46	파일 폴더
■ 책10.역사	2022-09-21 오후 9:46	파일 폴더
■ 책11.기타	2022-09-21 오후 9:46	파일 폴더

```
[라벨]Training
├── 간판01.가로형간판1
├── 간판02.가로형간판2
├── 간판03.가로형간판3
├── 간판04.가로형간판4
├── 간판05.가로형간판5
├── 간판06.돌출간판1
├── 간판07.돌출간판2
├── 간판08.세로형간판
├── 간판09.실내 안내판1
├── 간판10.실내 안내판2
├── 간판11.실내간판1
├── 간판12.실내간판2
├── 간판13.지주이용간판
├── 간판14.창문이용광고물
├── 간판15.현수막
├── 책01.총류
├── 책02.철학
├── 책03.종교
├── 책04.사회과학
├── 책05.자연과학
├── 책06.기술과학
├── 책07.예술
├── 책08.언어
├── 책09.문학
├── 책10.역사
└── 책11.기타
```

이미지 크롭



















마지막으로 모든 폴더 내에 있는 **.json** 파일을 제외한 다른 파일들을 제거해주시면 됩니다.

PC 간판_가로형간판_029994.json	470	817	JSON 파일
PC 간판_가로형간판_029995.json	416	647	JSON 파일
PC 간판_가로형간판_029996.json	410	644	JSON 파일
PC 간판_가로형간판_029997.json	424	691	JSON 파일
PC 간판_가로형간판_029998.json	425	692	JSON 파일
PC 간판_가로형간판_029999.json	514	947	JSON 파일
ZIP 간판_가로형간판_라벨링데이터1.zip	15,123,304	21,269,803	압축(ZIP) 파일

ZIP Training_책표지_기타_라벨링데이터.zip	1,546,011	2,172,683	압축(ZIP) 파일
PC 책표지_기타_000001.json	417	658	JSON 파일
PC 책표지_기타_000002.json	374	579	JSON 파일
PC 책표지_기타_000003.json	386	589	JSON 파일
PC 책표지_기타_000004.json	394	634	JSON 파일
PC 책표지_기타_000005.json	451	732	JSON 파일
PC 책표지_기타_000006.json	398	637	JSON 파일
PC 책표지_기타_000007.json	374	577	JSON 파일
PC 책표지_기타_000008.json	396	635	JSON 파일

이미지 크롭

다음은 [원천]데이터입니다. [라벨]데이터와 동일하게 아래 폴더들 모두 압축을 해제해주시면됩니다.

 [원천]Training_간판_가로형간판_원천데이터1.zip	2022-09-21 오후 5:18	압축(ZIP) 파일
 [원천]Training_간판_가로형간판_원천데이터2.zip	2022-09-21 오후 5:38	압축(ZIP) 파일
 [원천]Training_간판_가로형간판_원천데이터3.zip	2022-09-21 오후 5:31	압축(ZIP) 파일
 [원천]Training_간판_가로형간판_원천데이터4.zip	2022-09-21 오후 5:45	압축(ZIP) 파일
 [원천]Training_간판_가로형간판_원천데이터5.zip	2022-09-21 오후 5:59	압축(ZIP) 파일
 [원천]Training_간판_돌출간판_원천데이터1.zip	2022-09-21 오후 4:52	압축(ZIP) 파일
 [원천]Training_간판_돌출간판_원천데이터2.zip	2022-09-21 오후 4:23	압축(ZIP) 파일
 [원천]Training_간판_세로형간판_원천데이터.zip	2022-09-21 오후 4:58	압축(ZIP) 파일
 [원천]Training_간판_실내간판_원천데이터1.zip	2022-09-21 오후 4:46	압축(ZIP) 파일
 [원천]Training_간판_실내간판_원천데이터2.zip	2022-09-21 오후 4:18	압축(ZIP) 파일
 [원천]Training_간판_실내안내판_원천데이터1.zip	2022-09-21 오후 5:11	압축(ZIP) 파일
 [원천]Training_간판_실내안내판_원천데이터2.zip	2022-09-21 오후 4:08	압축(ZIP) 파일
 [원천]Training_간판_지주이용간판_원천데이터.zip	2022-09-21 오후 4:34	압축(ZIP) 파일
 [원천]Training_간판_창문이용광고물_원천데이터.zip	2022-09-21 오후 5:52	압축(ZIP) 파일
 [원천]Training_간판_현수막_원천데이터.zip	2022-09-21 오후 4:28	압축(ZIP) 파일
 [원천]Training_체크표지1.zip	2022-09-21 오후 5:24	압축(ZIP) 파일
 [원천]Training_체크표지2.zip	2022-09-21 오후 5:05	압축(ZIP) 파일
 [원천]Training_체크표지3.zip	2022-09-21 오후 4:40	압축(ZIP) 파일

이미지 크롭

압축해제한 폴더들의 이름을 [라벨]데이터와 동일하게 아래와 같이 수정해주시면됩니다.

[원천]Training_간판_가로형간판_원천데이터1 => [원천]Training01_간판_가로형간판_원천데이터1

[원천]Training_간판_가로형간판_원천데이터2 => [원천]Training02_간판_가로형간판_원천데이터2

[원천]Training_간판_가로형간판_원천데이터3 => [원천]Training03_간판_가로형간판_원천데이터3

:

[원천]Training_간판_현수막_원천데이터 => [원천]Training15_간판_현수막_원천데이터

책표지{1~3} 내부의 폴더들은 하위에 있는 폴더들의 이름을 아래와 같이 수정해주시면됩니다.

01.총류 => [원천]Training_16.총류

02.철학 => [원천]Training_17.철학

03.종교 => [원천]Training_18.종교

04.사회과학 => [원천]Training_19.사회과학

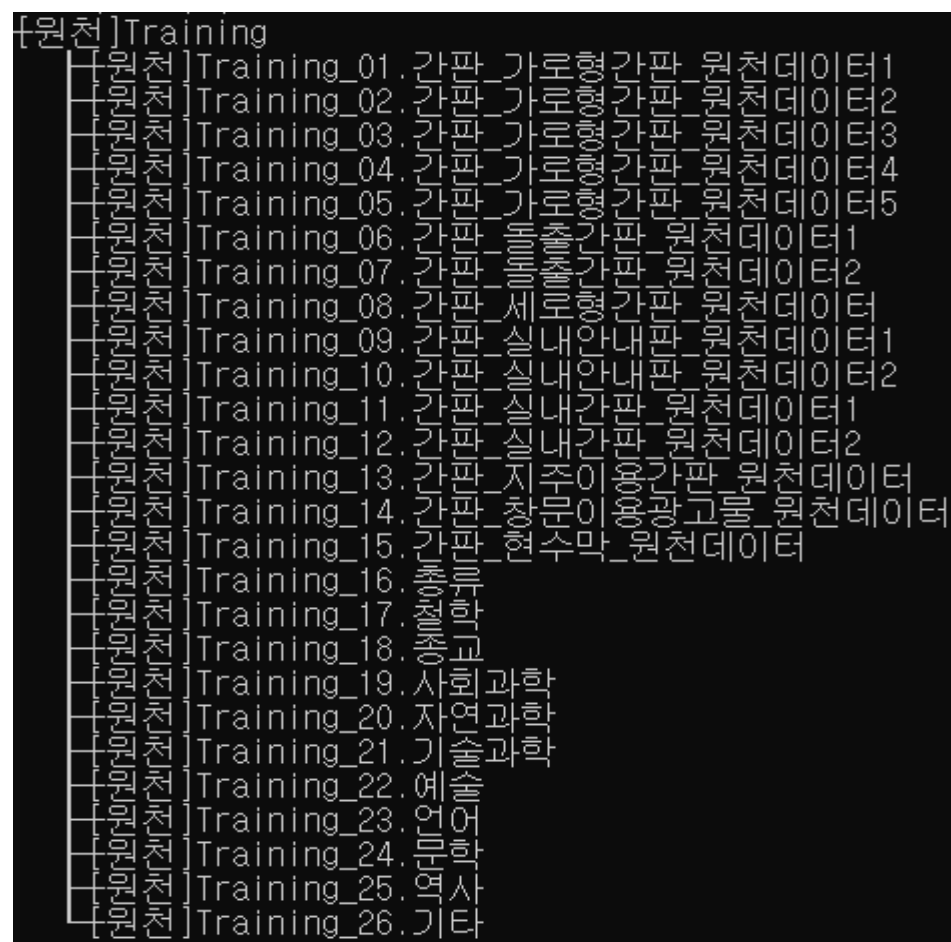
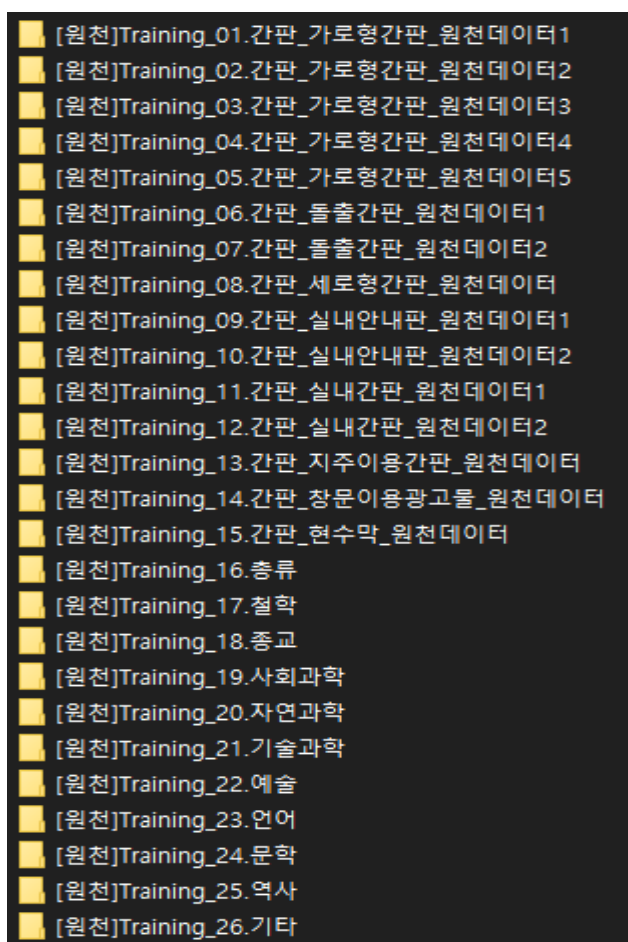
:

11.기타 => [원천]Training_26.기타

이미지 크롭

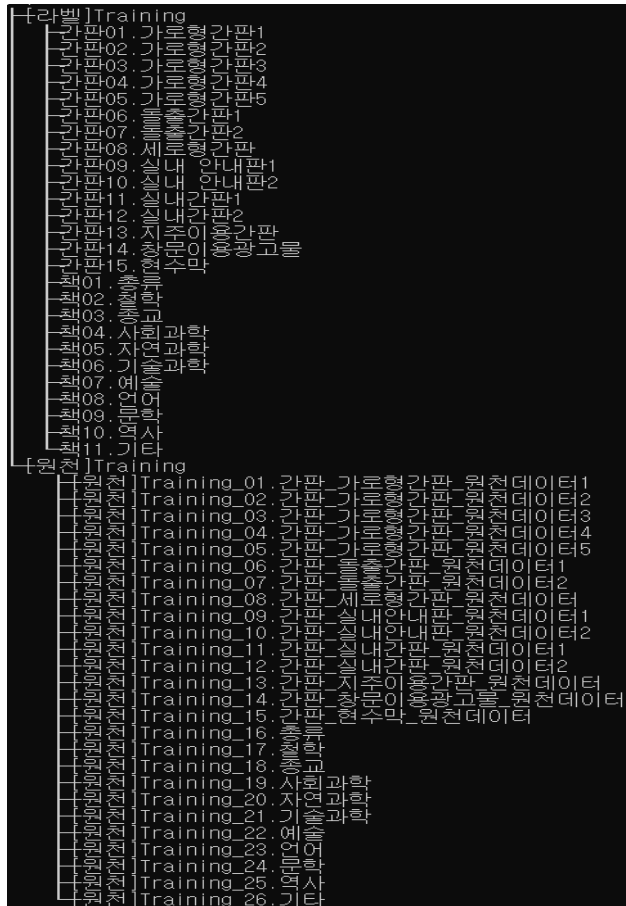
최종적으로 좌측과 같은 폴더들이 생성되게됩니다.

이 폴더들을 오른쪽 이미지같이 [원천]Training 폴더를 생성해 넣어주시면됩니다.



이미지 크롭

마지막으로 [라벨]데이터와 동일하게 모든 폴더 내에 있는 **이미지파일을 제외한** 다른 파일들이 있다면 제거해주시면 됩니다. 아래는 모든 과정을 마쳤을때의 폴더 구조입니다.



훈련 LMDB 생성

다음으로는 이미지를 크롭하기 위해 첨부된 Image_Crop 프로젝트의 image_crop.py 코드에서 폴더 경로를 설정해주시면 됩니다.

순서대로 {} 내부에

[원천] Training 폴더의 경로, [라벨] Training 폴더의 경로, 크롭한 이미지를 저장할 폴더의 경로를 입력해주시면 됩니다.

```
13 image_path = '{[원천] Training 폴더 경로}'  
14 json_path = '{[라벨] Training 폴더 경로}'
```

```
87 croppedImage.save('{크롭 이미지를 저장할 폴더 경로}/대회 훈련 데이터_크롭' + str(crop_index) + '.jpg')
```

예시 ↓

```
15 image_path = 'data/[원천] Training'  
16 json_path = 'data/[라벨] Training'
```

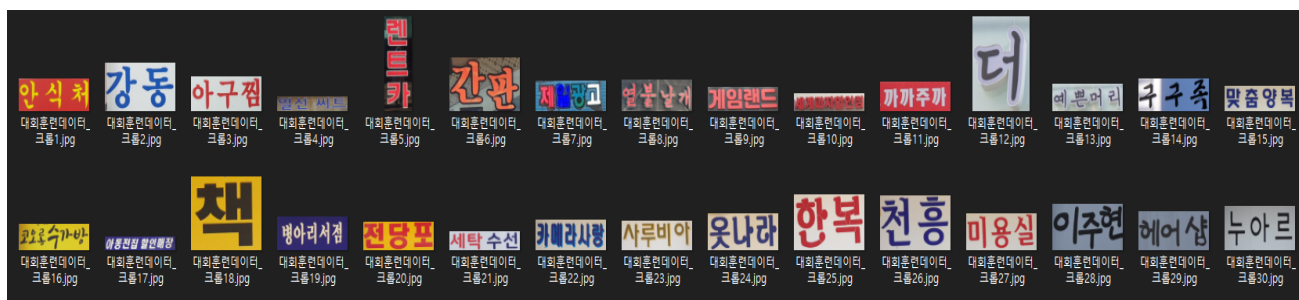
```
89 croppedImage.save('asset/crop/images/대회 훈련 데이터_크롭' + str(crop_index) + '.jpg')
```

훈련 LMDB 생성

이후 image_crop.py를 실행하면 아래와 같이 크롭된 이미지들과 타겟파일이 저장되게 됩니다.

타겟 : Image_Crop/asset/label_data/text_in_wild_target.txt

크롭이미지 : 이전에 설정한 경로에 저장



text_in_wild_target.txt - Windows 메모장

대회훈련데이터_크롭1.jpg	안식처
대회훈련데이터_크롭2.jpg	강동
대회훈련데이터_크롭3.jpg	아구찜
대회훈련데이터_크롭4.jpg	열선 씨트
대회훈련데이터_크롭5.jpg	렌트카
대회훈련데이터_크롭6.jpg	간판
대회훈련데이터_크롭7.jpg	제일광고
대회훈련데이터_크롭8.jpg	열불날개
대회훈련데이터_크롭9.jpg	게임랜드
대회훈련데이터_크롭10.jpg	세계과자할인점
대회훈련데이터_크롭11.jpg	까까주까
대회훈련데이터_크롭12.jpg	더
대회훈련데이터_크롭13.jpg	예쁜머리
대회훈련데이터_크롭14.jpg	구구족
대회훈련데이터_크롭15.jpg	맞춤양복
대회훈련데이터_크롭16.jpg	코오롱수가방
대회훈련데이터_크롭17.jpg	아동전집할인매장
대회훈련데이터_크롭18.jpg	책
대회훈련데이터_크롭19.jpg	병아리서점
대회훈련데이터_크롭20.jpg	전당포

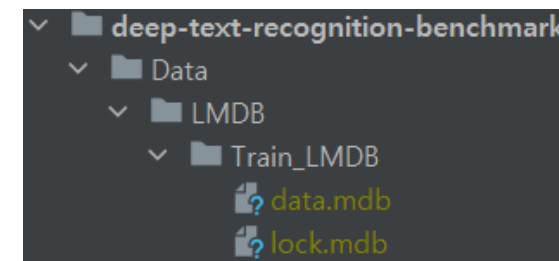
훈련 LMDB생성

다음으로 LMDB파일을 만들기 앞서 dee-text-recognition-benchmark 프로젝트의 create_lmdb_dataset.py내부의 map_size 변수를 25148518400으로 변경해주시면됩니다. (이후 과정에서 메모리사이즈 오류가난다면 map_size를 조금 더 키워주시면됩니다.)

```
40 # 메모리사이즈 설정
41 # Train = 25148518400
42 # Valid = 2588743168
43 env = lmdb.open(outputPath, map_size=25148518400)
```

다음으로 deep-text-recognition-benchmark 프로젝트내부에서 터미널 입력으로 `python create_lmdb_dataset.py --inputPath {크롭한이미지폴더의 경로} --gtFile {타겟데이터의 경로} --outputPath Data/LMDB/Train_LMDB` 실행해주시면됩니다. 실행이 완료되면 아래와 같이 .mdb파일이 생성되게됩니다.

예시 (deep-text-recognition-benchmark 프로젝트 내부에 1_create_train_dataset.sh 스크립트파일 실행하셔도됩니다.)
`python create_lmdb_dataset.py --inputPath asset/crops/images --gtFile asset/crops/text_in_wild_target.txt --outputPath Data/LMDB/Train_LMDB`



훈련 LMDB생성

LMDB생성 시

libGL.so.1 import error 발생 시 아래 명령어로 라이브러리 업데이트해주시면 됩니다.

apt-get update && apt-get install libgl1

apt-get install libgtk2.0-dev

검증LMDB 생성

검증데이터의 라벨을 생성하기 위해 첨부된 Image_Crop 프로젝트 내부의 create_valid_target.py를 사용합니다.
{ }내부에 대회 데이터 csv파일의 경로를 넣어주고 실행하면
asset/label_data/competition_train_data/dacon_train_target.txt 경로에 검증데이터의 라벨이 생성됩니다.

```
19 csv_to_target(R'{대회훈련데이터.csv파일의 경로}', 'asset/label_data/competition_train_data/dacon_train_target.txt')
```

검증LMDB 생성

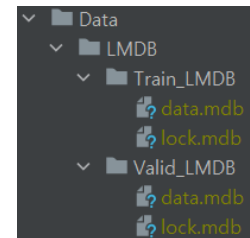
다음으로 LMDB파일을 만들기 앞서 dee-text-recognition-benchmark 프로젝트의 create_lmdb_dataset.py내부의 map_size 변수를 5588743168으로 변경해주시면됩니다.(이후 과정에서 메모리사이즈오류가 난다면 map_size를 조금 더 키워주시면됩니다.)

```
40 # 메모리사이즈 설정
41 # Train = 25148518400
42 # Valid = 5588743168
43 env = lmdb.open(outputPath, map_size=5588743168)
```

다음으로 deep-text-recognition-benchmark 프로젝트내부에서 터미널 입력으로 `python create_lmdb_dataset.py --inputPath {대회훈련데이터이미지 경로} --gtFile {타겟데이터의 경로} --outputPath Data/LMDB/Valid_LMDB` 실행해주시면됩니다. {}내부에는 폴더, 파일 경로로 입력
실행이 완료되면 오른쪽 이미지와 같이 .mdb파일이 생성되게됩니다.

예시 (deep-text-recognition-benchmark 프로젝트 내부에 2_create_val_dataset.sh 스크립트파일 실행하셔도됩니다.)

`python create_lmdb_dataset.py --inputPath asset/compete/train/images --gtFile asset/compete/train/dacon_train_label.txt --outputPath Data/LMDB/Valid_LMDB`



```
on:absolute;z-index:999  
x 5px #ccc}.gbtrl .gbm  
display:block;positio  
acity:1;*top:-2px;*le  
/;top:-4px\0/;left:-6  
e-box;display:inline  
isplay:block;list-s  
e-block;line-height  
pointer;display:bl  
tive;z-index:1000)  
padding-right:9px  
durl111
```

3. 모델 훈련

모델 훈련



clovaai/deep-text-recognition-benchmark
Text recognition (optical character recognition) with deep learning methods.

github.com

모델은 clova-ai의 deep-text-recognition-benchmark 모델을 수정하여 사용하였습니다.

Apache 2.0 라이선스, 수정/배포/상업적 이용 가능

수정한 모델은 제출한 파일의 deep-text-recognition-benchmark로 첨부하였습니다.

<https://github.com/clovaai/deep-text-recognition-benchmark>

모델 훈련

모델 훈련은 deep-text-recognition-benchmark 내부의 train.py를 이용합니다.
터미널 입력으로

```
python train.py --train_data Data/LMDB/Train_LMDB --valid_data Data/LMDB/Valid_LMDB --Transformation TPS --  
FeatureExtraction ResNet --SequenceModeling BiLSTM --Prediction CTC --rgb --adam
```

실행해주시면됩니다.

(deep-text-recognition-benchmark 프로젝트 내부에 3_train.sh 스크립트파일 실행하셔도됩니다.)

만약 훈련LMDB, 검증LMDB파일을 다른 경로에 저장하셨다면 붉은색 글씨부분을 해당 경로로
설정해주시면됩니다.

```
on:absolute;z-index:999  
x 5px #ccc}.gbrtl .gbm  
display:block;positio  
acity:1;*top:-2px;*le  
/;top:-4px\0/;left:-6  
e-box;display:inline  
isplay:block;list-s  
e-block;line-height  
pointer;display:bl  
tive;z-index:1000)  
padding-right:9px  
durl111
```

4 . 결 과 예 측

결과 예측

훈련이 종료되면 deep-text-recognition-benchmark/saved_models/TPS-ResNet-BiLSTM-CTC-Seed1111/ 경로에 iter_28700.pth 모델이 생성되게 됩니다.

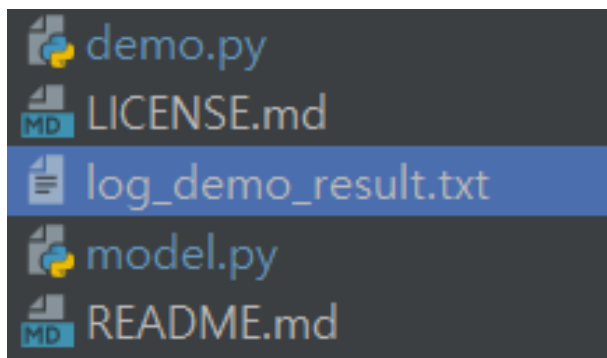
다음으로는 deep-text-recognition-benchmark 프로젝트 내부의 demo.py의 실행을 위해 터미널 입력으로 `python demo.py --image_folder {대회 테스트 이미지 폴더의 경로} --saved_model {iter_28700.pth의 경로} --Transformation TPS --FeatureExtraction ResNet --SequenceModeling BiLSTM --Prediction CTC --rgb`를 실행해주시면 됩니다.

예시(deep-text-recognition-benchmark 프로젝트 내부에 4_demo.sh 스크립트파일 실행하셔도 됩니다.)

```
python demo.py --image_folder asset/images/test_images --saved_model saved_models/TPS-ResNet-BiLSTM-CTC-Seed1111/iter_28700.pth --Transformation TPS --FeatureExtraction ResNet --SequenceModeling BiLSTM --Prediction CTC --rgb
```

결과 예측

demo.py의 실행이 끝나면 아래와같이 예측결과들이 log_demo_result.txt파일에 입력되게됩니다.
이 입력값들을 복사하여 대회sample_submission.csv의 text 하단에 붙여넣어주시면됩니다.



1	img_path	text
2	./test/test_00001.png	
3	./test/test_00002.png	
4	./test/test_00003.png	
5	./test/test_00004.png	
6	./test/test_00005.png	
7	./test/test_00006.png	
8	./test/test_00007.png	
9	./test/test_00008.png	
10	./test/test_00009.png	
11	./test/test_00010.png	
12	./test/test_00011.png	
13	./test/test_00012.png	
14	./test/test_00013.png	
15	./test/test_00014.png	
16	./test/test_00015.png	
17	./test/test_00016.png	
18	./test/test_00017.png	
19	./test/test_00018.png	
20	./test/test_00019.png	
21	./test/test_00020.png	
22	./test/test_00021.png	

결과 예측

마지막으로 저장한 csv파일의 인코딩을 변경하기 위해 deep-text-recognition-benchmark 프로젝트 내부의 make_csv.py 의 코드를 수정해주어야 합니다.

순서대로 앞서 생성한 sample_submission.csv의 경로, 인코딩을 변경해 csv파일을 저장할 경로를 입력해주시면 됩니다.

예시

```
test_predicts = 'Desktop/sample_submission.csv'
```

```
submit.to_csv('asset/result/final_submission.csv', index=False, encoding='utf-8-sig')
```

```
1 import pandas as pd
2 test_predicts = '{인코딩을 변경할 csv파일의 경로}'
3 |
4 submit = pd.read_csv(test_predicts, encoding='cp949')
5 submit.to_csv('{변경된 인코딩의 csv파일을 저장할 경로}', index=False, encoding="utf-8-sig")
```

결과 예측

`make_csv.py`를 실행하면 최종적으로 제출할 수 있는 `submission.csv`파일이 생성되고 이 `csv`파일을 이용해 재현성 확인해주시면 될 것 같습니다.

문의사항이 혹시 생기시면 아래 연락처로 연락주시면 감사하겠습니다.



김동수



010-9067-5828



20191766@edu.hanbat.ac.kr





이상입니다.