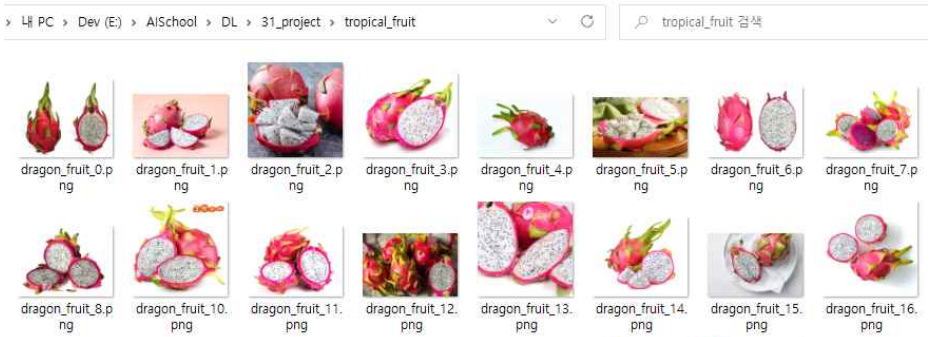
데이터 수집, 전처리 보고서

이나겸

1. 크롤링을 통한 데이터 수집 (코드 : crawling.py)

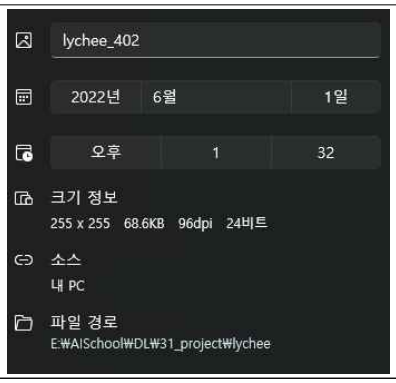
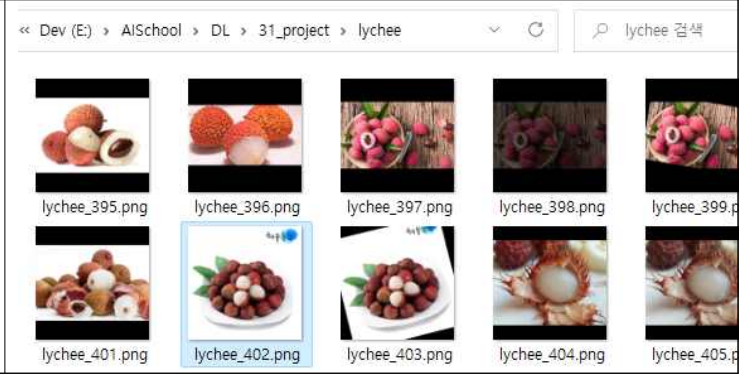
웹 크롤링을 통해 구글 이미지 검색에서 '망고, 용과, 리치, 두리안' 이미지를 수집한다.
urllib.request의 urlretrieve를 사용해서 이미지 url 주소를 저장한다.
dirname : 작업 위치 하위에 생성한 tropical_fruit 폴더에 .png 파일로 저장되도록 만든다.

```
urlretrieve(url, dirname + "₩₩" + image_name + "_" + str(t) + ".png")
# 파일이름 : dragon_fruit_0.png
```



2. 데이터 전처리 (코드 : data_preprocessing.py)

- 우선 tropical_fruit 폴더에 저장된 이미지 중에서 불필요한 이미지를 제거했다.
- 파일 개수 확인을 위해 tropical_fruit 폴더에서 mango, dragon, lychee, durian을 따로 분류했다.
  파일의 개수가 150개 미만이면 image augmentation을 이용해서 데이터를 증강시켰다.
  random 라이브러리를 사용해서 file의 개수만큼의 범위에서 랜덤값을 사용해서 무작위로 파일을 선택하고,
  밑에서 random.randint를 사용해서 정수 난수를 출력해서 랜덤으로 기법을 사용하도록 했다.
  (좌우반전, 밝기, 회전, 색상 균형조정)
- 코드 상의 padding 함수를 통해 이미지에 padding을 추가하고 255 * 255 사이즈가 되게 변형시킨다.
- 변형된 이미지는 각 mango, dragon_fruit, lychee, durian 폴더를 새로 만들어서 저장했다.
  # 파일이름 : {category}_{idx}.png ▶ dragon_48.png

폴더 생성	전저리 후 파일 속성	lychee 폴더
		

3. 데이터 후처리 (코드 : data_postprocessing.py)

- **CustomData : MyCustomDataset(Dataset) 클래스**

  __init__

  파일경로 리스트와 transform 전달한다.

  __getitem__

  index와 split을 사용해서 self.all_data를 특정 category로 분류될 수 있게 파일명에서 추출한다.

  (mango, dragon, lychee, durian)

mango의 label값	0	dragon의 label값	1
lychee의 label값	2	durian의 label값	3

  return은 transform이 적용된 image, data_path, data_label을 반환받도록 했다.

- **Dataset**

  path : 전처리된 mango, dragon_fruit, lychee, durian 폴더 하위의 파일들을 리스트 형태로 넘겨주었다.

  train, valid, test로 나누기 위해 sklearn의 train_test_split을 사용해서

- tv_data(=train+valid) 와 test로 나누기 위해 test_size를 0.1로 주고,

  tv_data를 train과 valid로 나누기위해 train_test_split을 한번 더 사용해서 test_size를 0.1로 적용시켰다.

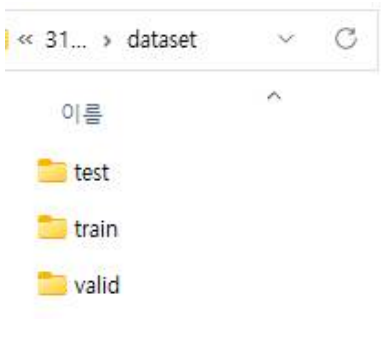
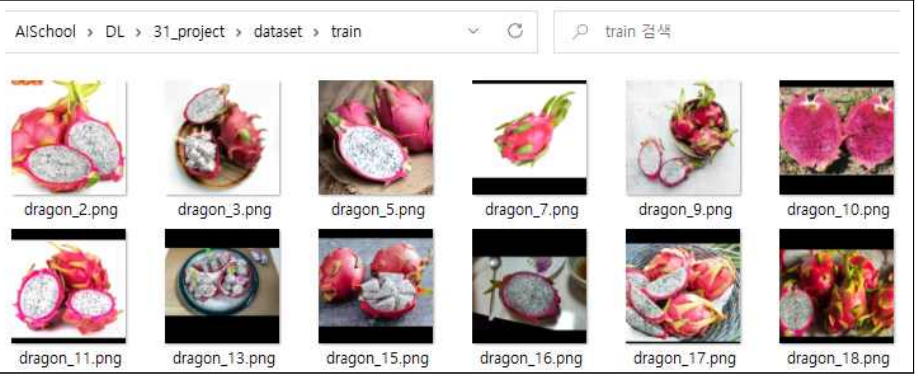
- **DataLoader**

  train, valid test 따로 DataLoader를 만들었다.

  train_dataloader, valid_dataloader, test_dataloader를 for문을 사용해서 이미지 경로, label 확인가능.

```
image >> ('./lychee/lychee_456.png',)   label >> tensor([2])
image >> ('./durian/durian_291.png',)   label >> tensor([3])
image >> ('./dragon_fruit/dragon_176.png',)     label >> tensor([1])
image >> ('./mango/mango_618.png',)     label >> tensor([0])
```

  - os.makedirs로 이미지 저장을 위해 dataset 폴더 하위에 train, valid, test 폴더 만듬
  - image는 torch tensor 형식이라 저장하기 위해서 torchvision의 save_image를 사용
  - 이미지의 경로를 split해서 저장할 이미지의 파일이름을 추출

dataset > test, train, valid	train 폴더
	

- **torchvision transforms**

  256, 256 사이즈로 Resize, 224, 224 사이즈로 RandomCrop, 80% 확률로 좌우반전

```
torchvision_transform = transforms.Compose([
    transforms.Resize((256, 256)),
    transforms.RandomCrop((224, 224)),
    transforms.RandomHorizontalFlip(p=0.8),
    transforms.ToTensor()]
)
```