**Assignment**

**허재호**

**데이터 전처리 코드**

import json

import os

import cv2

from PIL import Image

import glob

from tqdm import tqdm

import shutil

from sklearn.model\_selection import train\_test\_split

def bbox\_crop\_images(json\_data\_path, images\_path) :

    with open(json\_data\_path, 'r', encoding='utf-8') as j :

        json\_data = json.load(j)

    files = os.listdir(images\_path)

    for file in tqdm(files) :

        json\_infos = json\_data[file]

        annotations = json\_infos['anno']

        for i, annots in enumerate(annotations) :

            label = annots['label']

            bbox = annots['bbox']

            os.makedirs(os.path.join("./data/metal\_damged", 'data', label), exist\_ok=True)

            os.makedirs(os.path.join("./data/metal\_damged", 'train', label), exist\_ok=True)

            os.makedirs(os.path.join("./data/metal\_damged", 'val', label), exist\_ok=True)

            x, y, w, h = bbox

            img = cv2.imread(os.path.join(images\_path, file))

            cropped\_img = img[y:y+h, x:x+w]

            file\_name = file.split(".")[0]

            new\_file\_path = os.path.join("./data/metal\_damaged", "data", label, f"{file\_name}\_cropped\_{str(i).zfill(1)}.png")

            print(new\_file\_path)

            cv2.imwrite(new\_file\_path, cropped\_img)

def expend2square(pil\_image, background\_color):

    width, height = pil\_image.size

    if width == height:

        return pil\_image

    elif width > height :

        result = Image.new(pil\_image.mode, (width, width), background\_color)

        result.paste(pil\_image, (0, (width-height) // 2))

        return result

    else:

        result = Image.new(pil\_image.mode, (height, height), background\_color)

        result.paste(pil\_image, ((height-width) // 2, 0))

        return result

def resize\_with\_padding(pil\_image, new\_size, background\_color):

    img = expend2square(pil\_image, background\_color)

    img = img.resize((new\_size[0], new\_size[1]), Image.ANTIALIAS)

    return img

def resize\_images(data\_path) :

    img\_path\_list = glob.glob(os.path.join(data\_path, "\*","\*.png"))

    for img\_path in tqdm(img\_path\_list):

        try:

            dir, file = os.path.split(img\_path)

            folder\_name = dir.rsplit('\\')[1]

            os.makedirs(os.path.join("./data/metal\_damaged/resized", folder\_name), exist\_ok=True)

            name = os.path.basename(img\_path).rsplit('.png')[0]

            img = Image.open(img\_path).convert('RGB')

            img\_new = resize\_with\_padding(img, (256,256), (0,0,0))

            save\_file\_name = f"./data/metal\_damaged/resized/{folder\_name}/{name}.png"

            img\_new.save(save\_file\_name, "png")

        except Exception as ex:

            print(f"Error occurs on : {file} with the reason of {ex}")

def file\_split(src, dst) :

    files = glob.glob(os.path.join(src, "\*", "\*"))

    train\_list, val\_list = train\_test\_split(files, test\_size=0.1)

    for file in train\_list :

        folder\_path = file.split("\\")[1]

        shutil.copy2(file, os.path.join(dst, "train", folder\_path))

    for file in val\_list :

        folder\_path = file.split("\\")[1]

        shutil.copy2(file, os.path.join(dst, "val", folder\_path))

    print("File Split Completed")

json\_data\_path = "./data/data/anno/annotation.json"

images\_path = "./data/data/images"

bbox\_crop\_images(json\_data\_path, images\_path)

resized\_data\_path = "./data/metal\_damaged/data"

resize\_images(resized\_data\_path)

src = "./data/metal\_damaged/resized"

dst = "./data/metal\_damaged/"

file\_split(src, dst)

**데이터 전처리 결과**

텍스트, 스크린샷, 폰트, 블랙이(가) 표시된 사진

자동 생성된 설명

Figure Outputs

텍스트, 폰트, 화이트, 영수증이(가) 표시된 사진

자동 생성된 설명

Figure Cropped data

텍스트, 폰트, 화이트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

Figure Resized with padding data

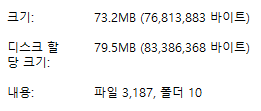


Figure Train data

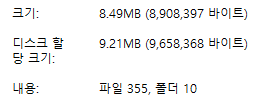


Figure Test data

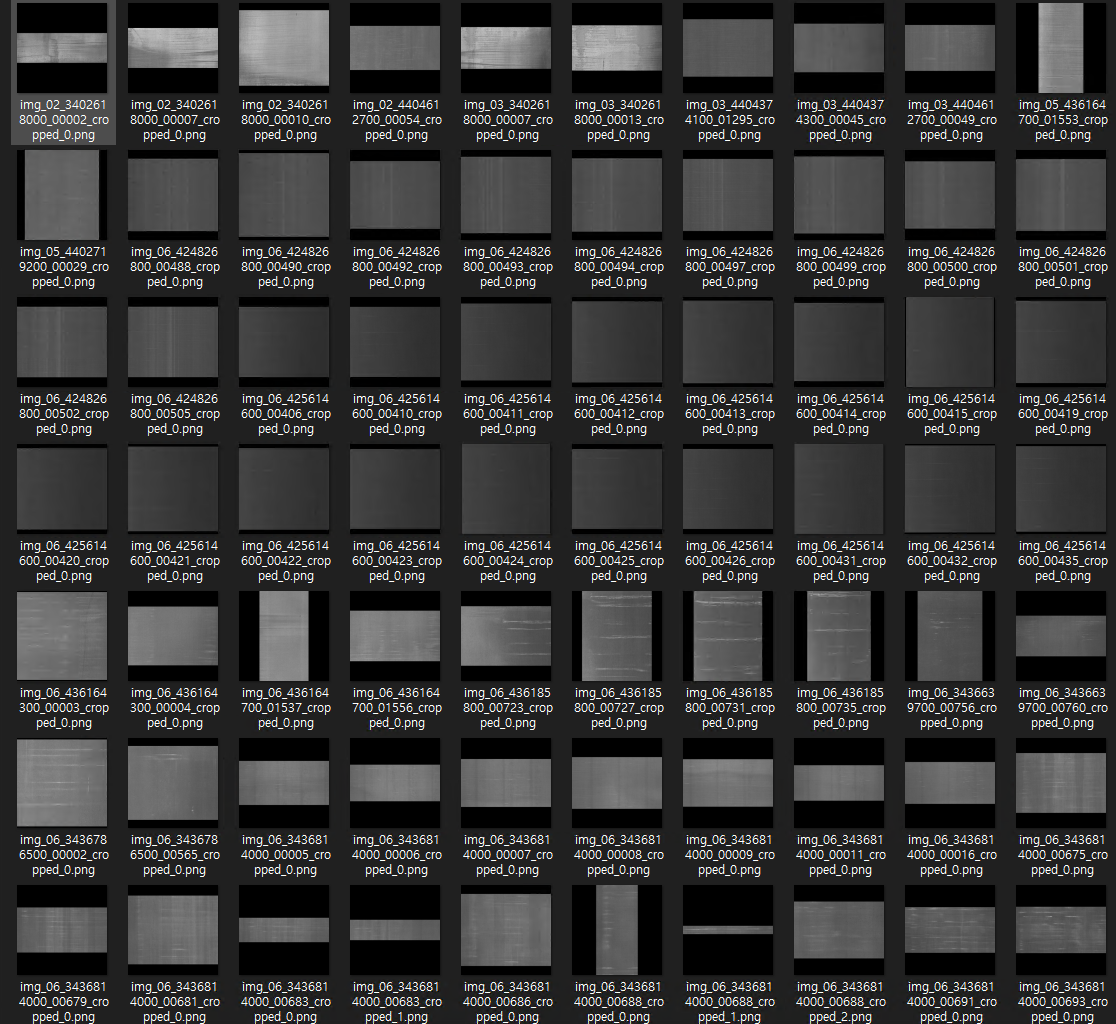


Figure Images

**학습 결과**

Epoch [1/20], Train Loss : 1.4055, Train Acc : 0.5723, Val Loss : 0.4810, Val Acc : 0.8507

Epoch [2/20], Train Loss : 0.5005, Train Acc : 0.8431, Val Loss : 0.4443, Val Acc : 0.8676

Epoch [3/20], Train Loss : 0.3955, Train Acc : 0.8798, Val Loss : 0.3422, Val Acc : 0.8873

Epoch [4/20], Train Loss : 0.3266, Train Acc : 0.8987, Val Loss : 0.2809, Val Acc : 0.9070

Epoch [5/20], Train Loss : 0.3325, Train Acc : 0.8968, Val Loss : 0.3390, Val Acc : 0.8958

Epoch [6/20], Train Loss : 0.2797, Train Acc : 0.9112, Val Loss : 0.3260, Val Acc : 0.9155

Epoch [7/20], Train Loss : 0.2768, Train Acc : 0.9118, Val Loss : 0.2991, Val Acc : 0.9127

Epoch [8/20], Train Loss : 0.2678, Train Acc : 0.9156, Val Loss : 0.2359, Val Acc : 0.9211

Epoch [9/20], Train Loss : 0.2422, Train Acc : 0.9247, Val Loss : 0.2757, Val Acc : 0.9239

Epoch [10/20], Train Loss : 0.2405, Train Acc : 0.9200, Val Loss : 0.3190, Val Acc : 0.9183

Epoch [11/20], Train Loss : 0.2332, Train Acc : 0.9216, Val Loss : 0.2994, Val Acc : 0.9070

Epoch [12/20], Train Loss : 0.2131, Train Acc : 0.9278, Val Loss : 0.4208, Val Acc : 0.9099

Epoch [13/20], Train Loss : 0.2243, Train Acc : 0.9272, Val Loss : 0.3137, Val Acc : 0.9155

Epoch [14/20], Train Loss : 0.1893, Train Acc : 0.9426, Val Loss : 0.3080, Val Acc : 0.9155

Epoch [15/20], Train Loss : 0.1893, Train Acc : 0.9366, Val Loss : 0.2732, Val Acc : 0.9239

Epoch [16/20], Train Loss : 0.1808, Train Acc : 0.9432, Val Loss : 0.3007, Val Acc : 0.9042

Epoch [17/20], Train Loss : 0.1699, Train Acc : 0.9435, Val Loss : 0.3201, Val Acc : 0.8986

Epoch [18/20], Train Loss : 0.1726, Train Acc : 0.9426, Val Loss : 0.2417, Val Acc : 0.9268

Epoch [19/20], Train Loss : 0.1743, Train Acc : 0.9404, Val Loss : 0.2570, Val Acc : 0.9155

Epoch [20/20], Train Loss : 0.1649, Train Acc : 0.9463, Val Loss : 0.2431, Val Acc : 0.9380

**테스트 결과**

Test Set : Acc [333/355] 94%

**힉습에 사용한 하이퍼 파라미터**

epochs = 20

criterion = CrossEntropyLoss().to(device)

optimizer = Lion(model.parameters(), lr=1e-4, weight\_decay=1e-2)