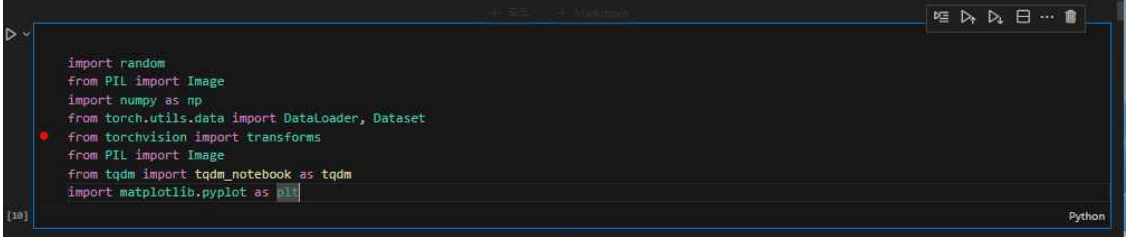


## 2024-1학기 창의학기제 주간학습보고서 (5주차)

창의과제	세종대학교 집현캠퍼스를 개선시킨 웹서비스 개발				
이름	신찬영	학습기간	5월 1일 ~ 5월 8일		
학번	23012094	학습주차	3	학습시간	3
학과(전공)	인공지능	과목명	자기주도창의전공1	수강학점	3
※ 수강학점에 따른 회차별 학습시간 및 10주차 이상 학습 준수					
금주 학습목표	본격적으로 수집한 데이터를 모델에 사용하기위해 harcascade로 짜르고 pytorch 라이브러리를 사용해 데이터를 전처리 후 MobileNet과 삼네트워드를 결합한 신경망에 학습시킨다..(기존에 한번 시도했던 것(3주차의 이지민 학생의 보고서 참고)이지만 지금은 다른 데이터 셋을 달라졌고 코드도 일부 달라 졌기 때문에 다시 시도한다.)				
학습내용	<p>삼네트워드 구조의 신경망을 학습시키기 위해 나의 얼굴 사진과 Ifw데이터 중 일부를 사용하였다. Ifw데이터는 여러 유명인의 얼굴이 모여있는 데이터 셋이다.</p>  <p>먼저 학습과 데이터 전처리를 위한 라이브러리들을 불러 와준다.</p> <p>random은 데이터를 랜덤하게 썬을 때사용할 라이브러리이다.  Image는 이미지 처리를 위한 라이브러리이다.  numpy는 수치해서, 통계 관련기능을 구현할 때 사용하는 라이브러리이다.  Dataset 라이브러리는 우리가 파이썬에서 모델을 학습시킬 때 이미지에 쉽게 접근할 수 있도록 하는 라이브러리이다.  Dataloader라이브러리는 딥러닝을 위해 미니 배치로 만들어 주는 라이브러리이다.  transform라이브러리는 모델이 학습할 수 있게 데이터를 전처리해주는 라이브러리이다.  matplotlib은 다양한 그래프를 그려주는 라이브러리이다. 이미지를 보여줄때도 사용된다.</p>				



```
import cv2
import os
from pathlib import Path

def save_detect_image(folder_path, padding_factor=0.1):
    if folder_path[-1] == '/':
        save_folder_path = folder_path[:-1] + '_face'
    else:
        save_folder_path = folder_path + '_face'

    Path(save_folder_path).mkdir(parents=True, exist_ok=True)

    cascade_path = cv2.data.harcascades + "haarcascade_frontalface_default.xml"
    face_cascade = cv2.CascadeClassifier(cascade_path)

    for filename in os.listdir(folder_path):
        image_path = os.path.join(folder_path, filename)
        image = cv2.imread(image_path)
        if image is None:
            continue
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(gray, 1.1, 4)

        for (x, y, w, h) in faces:
            padding_w = int(w * padding_factor)
            padding_h = int(h * padding_factor)
            x_pad = max(x - padding_w, 0)
            y_pad = max(y - padding_h, 0)
            w_pad = min(w + 2 * padding_w, image.shape[1] - x_pad)
            h_pad = min(h + 2 * padding_h, image.shape[0] - y_pad)

            face_img_with_padding = image[y_pad:y_pad+h_pad, x_pad:x_pad+w_pad]

            save_path = os.path.join(save_folder_path, filename)
            cv2.imwrite(save_path, face_img_with_padding)

    return save_folder_path
```

2주차에 했던 harcasade를 이용해 사진을 잘라주는 코드이다.  
(자세한 코드 설명은 2주차 이지민 보고서에 있다.)



```
class SiameseDataset(Dataset):
    def __init__(self, others_directory, my_directory, transform=None):
        self.others_images = [os.path.join(others_directory, img) for img in os.listdir(others_directory)]
        self.my_images = [os.path.join(my_directory, img) for img in os.listdir(my_directory)]
        random.shuffle(self.others_images)
        self.count=0
        if len(self.others_images) < len(self.my_images):
            for i in range(0, len(self.my_images)-len(self.others_images)):
                if self.count==len(self.others_images):
                    self.count=0
                    self.others_images.append(self.others_images[self.count])
                    self.count+=1
        elif len(self.others_images) > len(self.my_images):
            for i in range(0, len(self.others_images)-len(self.my_images)):
                if self.count==len(self.my_images):
                    self.count=0
                    self.my_images.append(self.my_images[self.count])
                    self.count+=1
        self.transform = transform
        # 그림자는 0, 강아지는 1로 레이블 지정
        self.labels = []
        for i in self.others_images:
            if 'chan' in i:
                self.labels.append(1)
            else:
                self.labels.append(0)
        self.cat_len=len(self.others_images)
        self.my_len=len(self.my_images)

    def __getitem__(self, index):
        # 첫 번째 이미지 선택
        img1_path = self.my_images[index]
        label = self.labels[index]

        img2_path = self.others_images[index]

        img1 = Image.open(img1_path)
        img2 = Image.open(img2_path)

        if self.transform is not None:
            img1 = self.transform(img1)
            img2 = self.transform(img2)

        return img1, img2, torch.from_numpy(np.array([int(label)], dtype=np.float32))

    def __len__(self):
        return len(self.others_images)
```

내가 준비한 데이터를 파이썬에서 쉽게 접근할 수 있게 Dataset클래스를 정의해 주었다. 먼저 데이터셋에서 사진이 저장되어 있는 경로를 받는다. 그리고 사진의 경로들을 리스트에 저장해 준다.

삼네트워크 모델을 학습시킬때 두 폴더의 데이터의 수가 다르면 안된다. 왜냐하면 삼네트워크의 원리가 신경망을 통과시킨 두 결과 값의 벡터값을 이용하여 가중치를 업데이트하는 방식이기 때문이다. 따라서 두 폴더의 데이터수를 맞추기 위해 수가 부족한 폴더의 경로 리스트에다 개수가 같아질 때까지 똑같은 경로를 복사해주었다.

ex)

[1,2,3],[4,5,6,7,8,9]-->[1,2,3,1,2,3],[4,5,6,7,8,9]

이렇게 만들어진 경로를 이용해 모델이 이미지를 요청하면 경로를 이용해 이미지를 불러온 다음 transform을 이용하여 전처리를 하고 이미지를 반환해준다.



```
# 데이터 전처리 정의
transform = transforms.Compose([
    transforms.Resize((120, 120)),
    transforms.ToTensor(),
])

# 데이터셋 인스턴스 생성
dataset = SiameseDataset(cat_directory=r'./others_face', my_directory=r'./my_face', transform=transform)
val_dataset=SiameseDataset(cat_directory=r'./others_v_face',my_directory=r'./my_v_face', transform=transform)
# 데이터로더 설정
dataloader = DataLoader(dataset, batch_size=32, shuffle=True)
val_dataloader=DataLoader(val_dataset,batch_size=32,shuffle=True)
dataloader_dict={'train':dataloader,'val':val_dataloader}
```

준비된 이미지 데이터 셋을 모델에 학습 시키기 위해 120\*120의 해상도로 맞추고 텐서 형태로 바꿔 준다. 또한 미니 배치를 사용하기위해 DataLoader로 32개씩 묶어준다.

```
class ContrastiveLoss(torch.nn.Module):
    """
    Contrastive loss function.
    Based on: http://yann.lecun.com/exdb/publis/pdf/hadsell-chopra-lecun-06.pdf
    """
    def __init__(self, margin=2.0):
        super(ContrastiveLoss, self).__init__()
        self.margin = margin

    def forward(self, output1, output2, label):
        euclidean_distance = F.pairwise_distance(output1, output2, keepdim = True)
        loss_contrastive = torch.mean((1-label) * torch.pow(euclidean_distance, 2) +
                                       (label) * torch.pow(torch.clamp(self.margin - euclidean_distance, min=0.0), 2))
```

학습에 쓸 손실함수를 정의해준다.(자세한 설명은 이지민 3주차 보고서에 있다.)

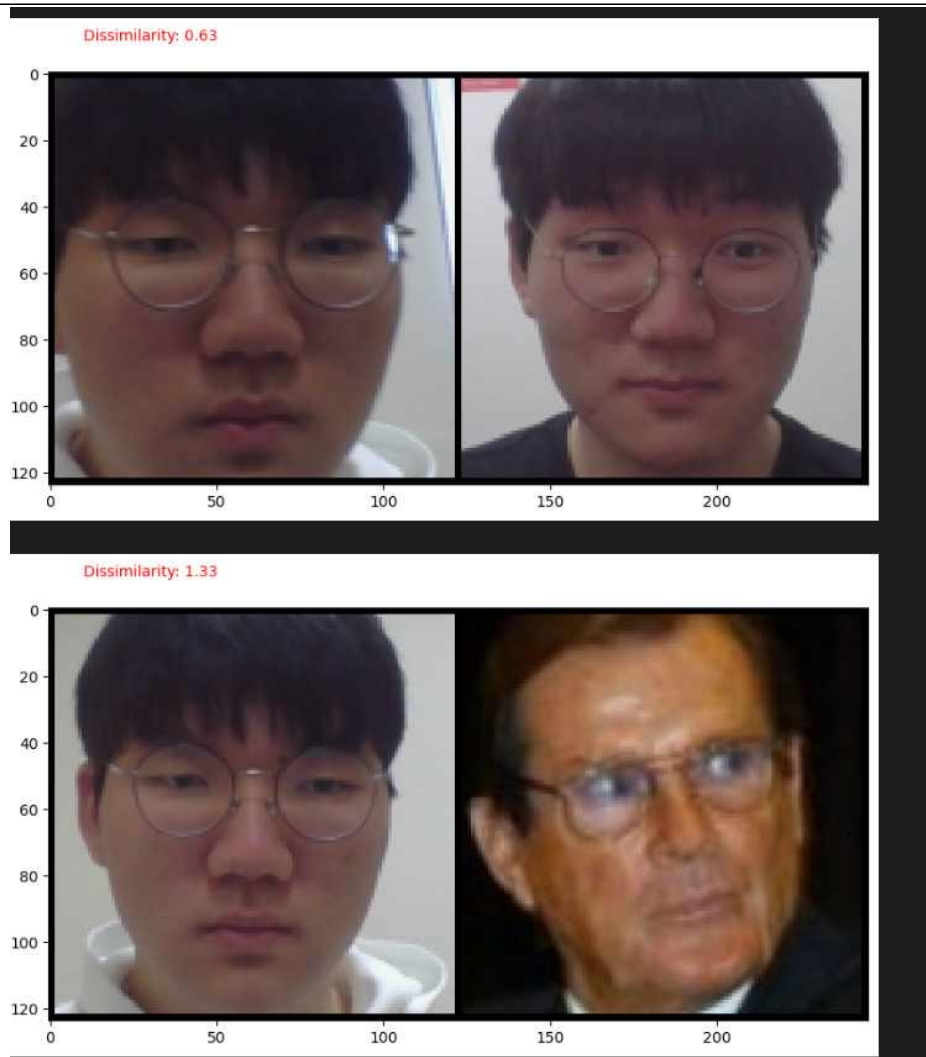
```
from torchvision.models import mobilenet_v3_large
class SiameseNetwork(nn.Module):
    def __init__(self):
        super(SiameseNetwork, self).__init__()
        self.model=mobilenet_v3_large(pretrained=True)

        self.fc = nn.Sequential(
            nn.Linear(960, 512),
            nn.ReLU(inplace=True),

            nn.Linear(512, 512),
            nn.ReLU(inplace=True),

            nn.Linear(512, 5))
        self.model.classifier=self.fc
    def forward(self, input1, input2):
        output1 = self.model(input1)
        output2 = self.model(input2)
        return output1, output2
```

모바일 넷과 삼네트워크를 결합한 신경망이다. 우리는 컴퓨터 리소스 자원이 부족하여 완전 연결층만 학습시키는 전이학습을 하였다.



학습 시킨 결과물이다. 위의 사진처럼 잘 구별하는 모습을 보여준다.





	그러나 위의 사진같이 치명적인 오류가 일부 존재한다.
학습방법	위의 코드들은 공식 파이토치사이트의 튜토리얼을 보고 작성하였다.
학습성과 및 목표달성도	<p>성공적으로 데이터를 자르고 파이썬에서 데이터셋 형태로 바꿔주었다. 그러나 이미지 데이터를 얼굴만 자르는 과정에서 harcasade가 잘 못 자르는 경우가 있어서 잘 못자른 데이터는 걸러주었다.</p> <p>모델의 학습 결과는 성곡적이지만 일부 치명적인 오류가 있었다. 이것은 우리가 평가기준 없이 임의의 하이퍼 파라미터를 설정해 생긴 오류로 추정된다.</p>
참고자료 및 문헌	<p><a href="https://tutorials.pytorch.kr/">https://tutorials.pytorch.kr/</a></p> <p><a href="https://vis-www.cs.umass.edu/lfw/">https://vis-www.cs.umass.edu/lfw/</a></p>
내주 계획	하이퍼파라미터를 수정하기 위해 인공지능에 사용되는 평가 기준에 대해 조사할 것이다.

2024 년 5월 2 일

지도교수

(인)