기말 프로젝트

Few-shot learning

학번: 20195234

이름: 임지운

소속: 빅데이터 전공

1- 대회 평가지표 분석

mAP(mean Average Precision) 개념을 이해하기 위해선 IOU(Intersectio over Union), Precision(정밀도), Recall(재현율), AP(Average Precision)에 대해 알아야 한다.

- IOU: 12주차에서 Object Detection에서 배운 내용으로 두 개의 Bounding box가 얼마나 일치하는지를 확인한다. (Area of Overlap)/(Area of Union)을 한 값이다.
- Precision: 모델이 True라고 예측한 것 중 진짜 True의 비율 (TP)/(TP + FP)
- Recall: 실제 정답이 True인 것 중 모델이 True라고 예측한 것의 비율 (TP)/(TP + FN)

  + Precision과 Recall은 알고리즘의 평가 지표가 될 수 있지만 어느 한 값으로 성능을 판단하기는 불가능하다. 일반적으로 Precision과 Recall은 반비례 관계이기 때문에 둘 다 좋은 평가를 얻어야 한다. 두 개의 값을 최상으로 얻는 threshold(임계값)을 잘 파악해야 한다. 이를 Precision-Recall 곡선으로 파악할 수 있다. Recall 값의 변화에 따른 Precision값을 알 수 있는 그래프이다.
- AP: Precision-Recall 곡선은 알고리즘의 성능을 평가하기는 좋다. 하지만 서로 다른 두개의 알고리즘을 평가하기에는 나쁘다. 이를 위해 나온 게 AP이다. 이는 Precision-Recall 곡선의 아래 면적으로 해당 값이 높으면 높을수록 해당 알고리즘 성능이 높다고 평가한다.
- mAP: 마지막으로 우리가 알고 싶어하는 대회의 평가지표 mAP이다. mAP는 생각보다 간단하다. 우리가 찾는 클래스가 여러 개라면 AP가 여러 개가 나올 것이다. 각각의 AP를 더해 클래스의 수로 나눈 평균(mean)이 mAP인 것이다.

mAP의 특징은 다중 클래스 문제를 풀기에 적합하다. 그렇기에 Humpback Whale Identification Challenge에서는 mAP를 평가지표로 삼은 것이다.

2- Metric learning에 대한 조사 보고서

개념: Metric Learning에서 Metric은 한국말로 미터법(의)로 쓰이나 수학적인 의미로는 거리, 측정의 기준이라는 뜻을 사용한다. Metric Learning은 거리 학습으로 주어진 데이터의 특징 거리를 학습하는 학습법이다. Metric(거리)를 학습할 때 비슷한 것은 가까이 다른 것을 멀리 학습한다.

Metric Learning에서는 여러가지 Loss(손실 함수)를 사용한다. 어떤 손실함수가 있는지 조사해 보았다.

1-1. Contrastive Loss

Contrastive Loss는 두 벡터의 특징의 유사성을 정량화 하는 함수로 positive loss 와 negative loss를 합친 값이다. 여기서 positive loss는 비슷한 이미지를 표현하고, negative loss는 다른 이미지를 표현하는 손실함수이다.

1-2. Triplet Loss

Triplet Loss는 같은 클래스 또는 특정한 기준에서 유사한 입력 사이의 거리를 가깝게 하고 서로 다른 클래스, 유사하지 않은 입력 사이의 거리를 멀게 만드는데 쓰는 손실함수이다. 여기서 anchor는 입력, positive는 입력과 같은 클래스, negative는 입력과 다른 클래스이다.

1-3. Center Loss

Center Loss는 각 클래스의 중심에 대한 거리를 최소화하는 손실함수이다. 각 클래스의 중심을 학습하여 유사한 샘플들의 거리를 최소로 만들기 위해 이동시킨다. 각 클래스별로 중심을 학습시켜야 하기 때문에 클래스가 많아지면 사용하기 힘들다.

1-4. N-pair Loss

N-pair Loss는 Contrastive Loss와 Triplet Loss의 단점인 느린 수렴과 부적절한 로컬 최적화로 인해 발생하는 손실을 줄이기 위해 만들어진 손실 함수이다. 앞에서 Triplet Loss는 하나의 anchor, 하나의 positive, 하나의 negative를 추출하여 비교를 했다면 N-pair Loss는 하나의 anchor, 하나의 positive, N개의 negative를 추출하여 비교한 것이다. 하지만 Triplet Loss보다 불균형한 데이터 처리에 나쁘며 다중 클래스 분류 문제에 특화되어 있다.

1-5. Margin Loss(Negative pair의 거리를 Margin이라 한다.)

Margin Loss는 Triplet Loss(유연성)와 Contrastive Loss(계산 효율성)의 장점을 합친 Loss로 Triplet Loss에서 세 개의 클래스 간의 거리를 학습했다면 Margin Loss는

학습할 때 추가로 Margin도 사용한다.

1-6. ArcFace Loss(Additive Angular Margin Loss)

각 클래스에 각도 Margin을 추가하여 특정 각도의 Margin을 유지하면서 각 클래스의 특징을 추출한다. 즉, 해당 anchor과 같은 클래스라면 각도가 작아지게, 다른 클래스라면 각도가 멀어지게 한다는 것이다. 해당 Loss는 각도로 분류를 되도록 학습시킨다.

1-7. Online Triplet Loss

Triplet Loss의 변형으로 Triplet Loss는 모든 가능한 Triplet을 사용하여 학습하지만 Online Triplet은 매 학습 반복에서 동적으로 Triplet을 선택하여 보다 효과적으로 학습하는 방법이다.

적용 사례 1: 얼굴 인식(Face Recognition)

➔ 사용 Loss: Contrastive Loss, Triplet Loss, Center Loss, N-pair Loss, Margin Loss, ArcFace Loss

➔ 얼굴 인식에서 얼굴의 특징을 학습하여 같은 사람의 다른 모습의 얼굴 간의 거리를 최소화하고 다른 사람의 얼굴 간의 거리를 최대화한다. 이를 Metric Learning으로 학습시킬 수 있다.

적용 사례 2: 자연어 처리(Natural Language Processing - NLP)

➔ 사용 Loss: Contrastive Loss, Triplet Loss, Center Loss, N-pair Loss, Margin Loss, Online Triplet Loss

➔ 텍스트 데이터를 고차원의 벡터로 변환시켜 텍스트 간의 유사성을 파악하기 위해 Metric Learning으로 비슷한 의미의 텍스트들과 다른 의미의 텍스트를 학습하여 텍스트 분류기로 사용할 수 있다.

적용 사례 3: 이미지 분류(Image Classification)

➔ 사용 Loss: Contrastive Loss, Triplet Loss, Center Loss, ArcFace Loss

➔ 이미지 분류를 위해 이미지 간 유사한 이미지의 특징과 다른 이미지의 특징의 거리를 Metric Learning으로 학습시켜 특징이 유사한 이미지끼리 분류할 수 있다.

3- 본인의 문제 해결 과정에 대한 핵심 내용 작성

3-1. 데이터 증강 적용

이전 과제05에서 사용했던 데이터 증강을 사용하기로 했다.

```
#데이터 증강을 통해 데이터를 늘린다.
data_transforms_plus = transforms.Compose([
                                        transforms.Resize((128, 128)),
                                        transforms.RandomRotation(degrees=30),   # Random
                                        transforms.RandomHorizontalFlip(),        # Random
                                        transforms.ToTensor(),
                                        transforms.Normalize(mean=[0.485, 0.456, 0.406],
                                                             std=[0.229, 0.224, 0.225])
```

랜덤하게 30도 회전과 랜덤하게 좌우로 반전시킬지를 적용해 데이터를 늘렸다.

Crop으로 데이터를 자르기에는 고래의 꼬리 모양까지 잘라 이도 저도 아닌 데이터가 생길 수 있어 사용하지 않았다.

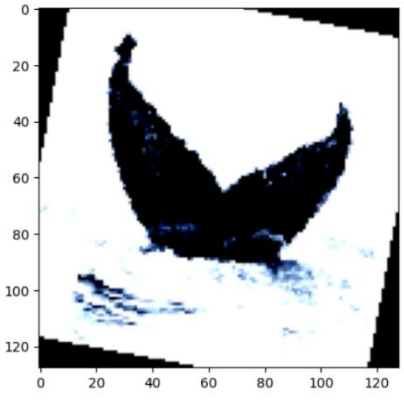
```
from torch.utils.data import ConcatDataset

train_dataset_list = [train_dataset]

for i in range(1):
    new_train_dataset = WhaleDataset(datafolder='/kagg
    train_dataset_list.append(new_train_dataset)

# train_dataset_list 주위의 대괄호를 제거합니다.
train_dataset = ConcatDataset(train_dataset_list)
```

ConcatDataset으로 리스트 안에 있는 데이터를 하나의 데이터로 통합하는 torch의 메서드를 사용하여 데이터 증강을 적용한 사진을 하나의 데이터셋으로 합쳤다. 여기서 주의할 점은 ConcatDataset은 데이터 열(col)의 모양이 동일해야 한다는 점이다.



옆의 사진은 데이터 증강이 잘 적용됐는지 확인하기 위해 출력한 것이다.

3-2. 모델 구현하기

먼저 모델은 kaggle의 다른 사람의 모델과 ResNet18을 참고했다.

출처: https://www.kaggle.com/code/martinpiotte/whale-recognition-model-with-score-0-78563

위 출처는 해당 챌린지에서 고득점을 받은 사람이 적어 놓은 설명글로 코드가 포함되어 있지만 keras로 구현했고 데이터 전처리 방식이 달라 해당 코드를 다시 작성해야 했다. 여기서 새로 안 사실은 모델의 구조에 sub-block 구조를 만들어 데이터의 feature을 중간중간에 가다듬을 수 있다는 것을 배웠다.

해당 글에서는 Metric Learning의 중 하나인 Siamese Network를 사용했다. Siamese Network는 두개의 이미지를 같은 모델에 넣어 추출된 특징을 비교하는 네트워크로 이미지 분류에 사용되는 네트워크다. 해당 네트워크는 Triplet Loss와 같은 손실함수를 사용해 학습한다. 손실함수를 구현하려고 했는데 우연히 pytorch에서 제공하는 라이브러리 pytorch_metric_learning를 찾아 해당 라이브러리를 사용할 예정이다.

3-3. pytorch_metric_learning 사용하기

설치 명령어: !pip install pytorch-metric-learning

참고 사이트: https://kevinmusgrave.github.io/pytorch-metric-learning/

해당 라이브러리는 위 명령문을 실행해 설치하여 사용할 수 있다. metric learning에 필요한 다양한 모듈을 제공한다. 내가 여기서 사용할 것은 학습을 위한 데이터 쌍을 만들어주는 miner과 loss 함수를 제공해주는 losses를 사용할 예정이다.

하지만 크나큰 문제가 발생하였다. Metric Learning을 위해 해당 라이브러리에서 miner라는 모듈을 사용하여 데이터 쌍을 만들었으나 그것이 대체로 negative쌍이라는 것이 문제다. Positive 쌍이 있어야 하는데 해당 쌍이 생길 확률이 너무 희박하다. 그래서 학습이 제대로 되지 않는 것 같다.

3-4. 결과

시간이 모자라서 결국 3-3은 적용시키지 못한 채로 3-2에서 one-hot encoding 라벨에 모델을 학습하여 binary-crossentropy loss로 학습을 한 어중간한 모델이자, 조교 선생님이 제공하신 모델과 크게 다를바가 없는 모델이 만들어졌다.

그나마 데이터 증강을 적용하여 점수가 미세하게 올랐지만 내가 원하는 결과는 아니였다.

4- 리더보드 순위/점수 스크린샷