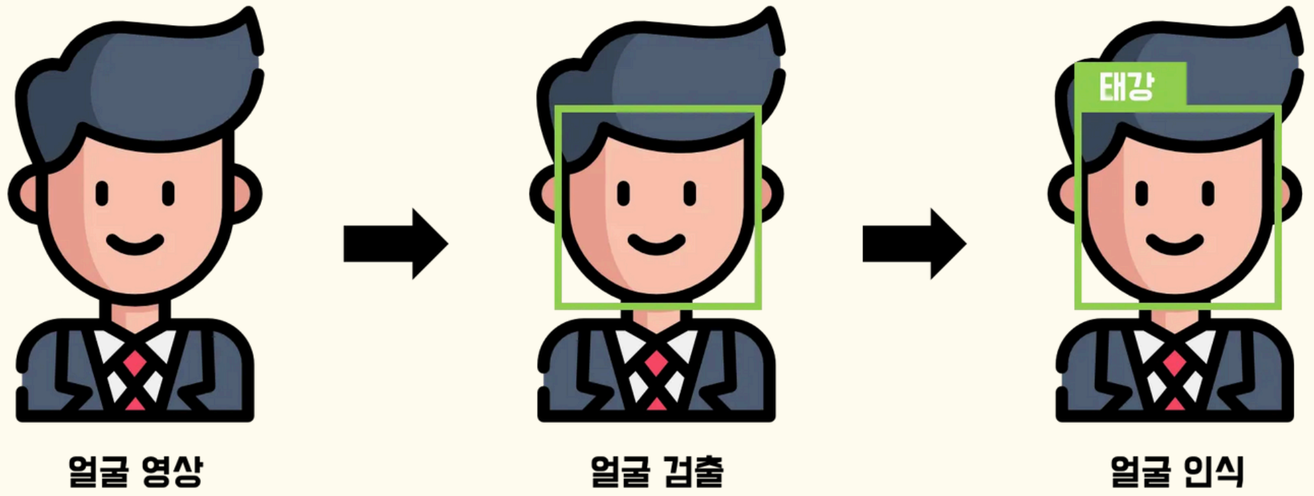
얼굴인식 모델 분석

LOCKERS

Team Minions

김문선, 김세찬, 김기현, 이찬혁

01 얼굴 인식 순서



[Detection] 영상 내 얼굴의 위치를 찾기
[Embedding] 얼굴을 계산할 수 있는 숫자(벡터)로 표현
[Identification] 벡터값을 이용해 타인과 나를 구분

02 모델 비교

↓선정↓

	FaceNet-Pytorch	Insight-Face
소개	특징 벡터(임베딩)를 추출하여 얼굴 간의 유사도를 비교하는 데 많이 사용되는 모델이다. TensorFlow의 FaceNet 모델을 PyTorch로 재구현한 모델이다.	얼굴 임베딩을 512차원 공간에 매핑하여 유클리드 거리를 기반으로 얼굴 간의 유사성을 평가한다. 주로 아크페이스(ArcFace) 손실 함수를 사용하여 높은 정확도를 제공한다.
Inference-time	약 0.2s ~ 0.37s	약 0.20 ~ 0.25s
정확도	약 98.16%	약 98.2%
Fine-Tuning 여부	O 추가 학습을 통해 다양한 데이터셋으로 적용 가능	△ 다양한 설정과 손실 함수 조정 가능하나 Github에 정확한 fine-tuning 정보가 없어서 어려움

FaceNet 이란?	• 안면 인식을 위해 개발된 Deep Learning 모델로, 얼굴 이미지에서 특징 벡터(=embedding)를 추출하여 얼굴 간의 유사도를 비교하는 데 사용됨. • 얼굴 이미지 사이의 거리를 학습하여 얼굴 인식 기능을 수행함.
정확도	• 이미지를 얼굴 크기에 맞게 crop하여 진행했을 때 : **98.87%** • 추가적인 alignment를 했을 때 : **99.63%**
fine-tuning	• PyTorch 기반 구현체에서 fine-tuning이 가능하며, 사용자 정의 데이터셋에 맞게 모델을 **재학습** 가능 • Facenet-pytorch를 사용해서 fine-tuning이 가능함.

InsightFace란?	• InsightFace는 얼굴 인식과 관련된 최첨단 기술을 구현한 오픈 소스 프로젝트. 주로 ArcFace, CosFace, SphereFace 등과 같은 다양한 얼굴 인식 모델을 제공. MS1M, VGG2, CASIA-WebFace 등 다양한 데이터셋을 지원. • **ArcFace** : 얼굴 인식을 위한 대표적인 모델로, 대규모 얼굴 데이터셋과 **Additive Angular Margin Loss(추가 각도 마진 손실) 함수**를 사용하여 학습. 얼굴 임베딩 벡터를 512차원 공간으로 매핑하며, 임베딩 벡터 간의 유사성을 **유클리드 거리**로 평가.
정확도	• ArcFace, CosFace, SphereFace 3가지 모델 중 **ArcFace 모델**은 얼굴 인식에서 **높은 정확도**를 제공하며, Labeled Faces in the Wild (LFW) 데이터셋에서 **99.83%**의 정확도를 기록
fine-tuning	• 사용자 정의 데이터셋으로 모델을 PyTorch 및 MXNet 기반 **fine tuning 가능**. • 그러나 Git에 정확한 fine tuning 코드가 존재하지 않아서 fine tuning 을 하기에는 어려움이 있음

03 Loss 함수

CrossEntropy Loss

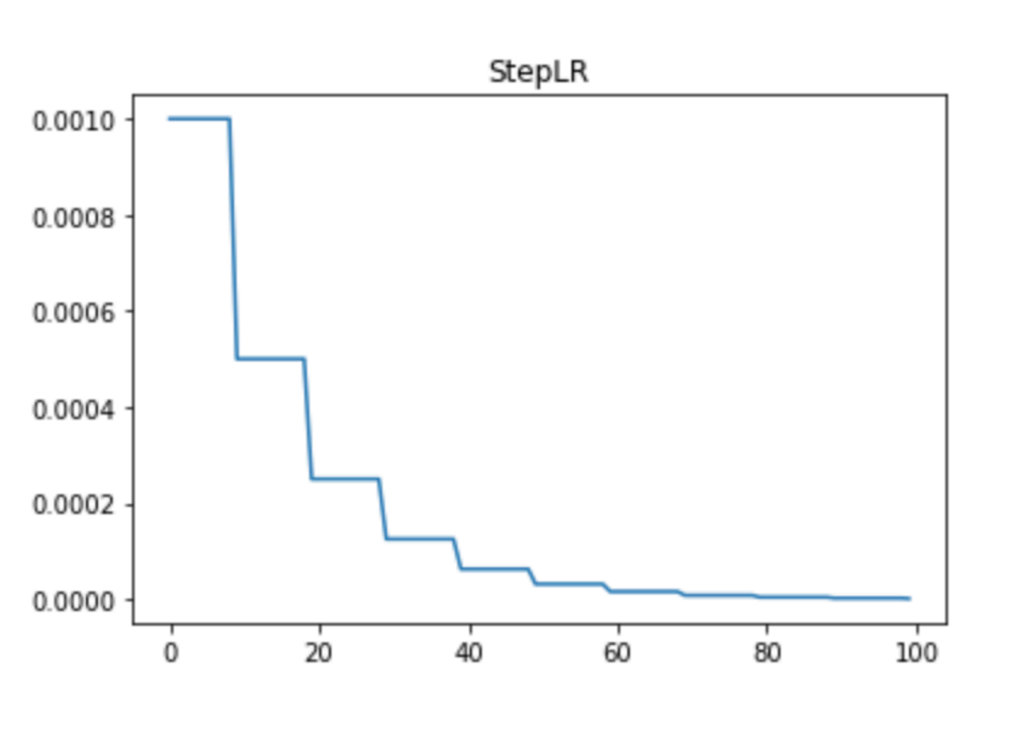
$$L = -\sum_{i=1}^{C} y_i \log(p_i)$$

- C는 클래스의 수입니다.
- y_i는 실제 레이블에 해당하는 원-핫 인코딩 벡터입니다. (정답인 클래스는 1, 나머지는 0
- p_i는 모델이 예측한 해당 클래스의 확률입니다.

- **nn.CrossEntropyLoss() 계산 순서**
  - 1. 정답을 one hot encoding 처리해서 OneHot Vector로 만든다.
    - 정답은 Label Encoding 상태의 값을 사용한다.
  - 2. 모델이 예측한 결과에 Softmax를 적용해 확률값으로 바꿔준다.
    - Softmax 적용 전 값을 입력한다.
  - 3. Categorical Crossentropy 공식을 이용해 오차를 계산한다.

04 Learning Rate Scheduler

Step LR



StepLR는 **지정된 스텝 간격마다 학습률을 감소**시키는 간단하면서도 효과적인 방법. 이 스케줄러는 특정 에폭의 수마다 학습률을 미리 정해진 비율로 줄임. 이러한 방식은 학습 과정에서 중요한 지점에서만 학습률을 조정하고 싶을 때 유용함.

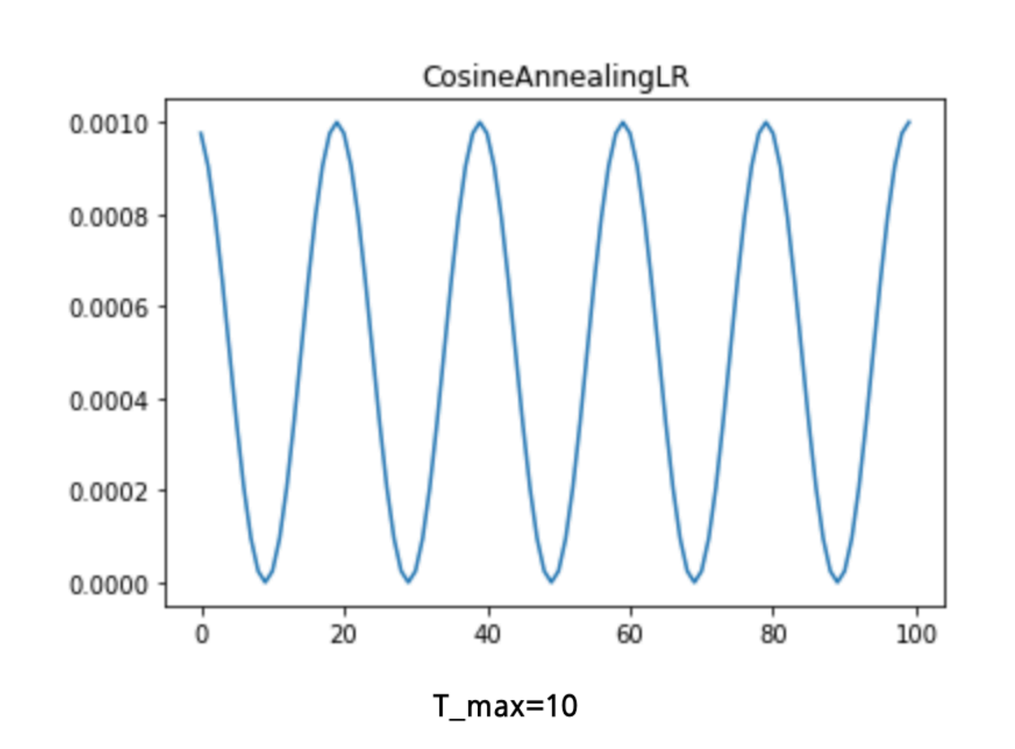
Parameters

- optimizer: 이전에 정의한 optimizer 변수명을 넣어준다.
- step_size: 몇 epoch마다 lr을 감소시킬지가 step_size를 의미한다.
- gamma: gamma 비율로 lr을 감소시킨다.

```
optimizer = torch.optim.SGD(model.parameters(), lr=0.001)
scheduler = optim.lr_scheduler.StepLR(optimizer,
step_size=10, gamma=0.5)
```

04 Learning Rate Scheduler

CosineAnnealingLR



cosine 그래프를 그리면서 learning rate를 변경 하는 방식.
최근에는 learning rate를 단순히 감소시키기 보다는 **감소와 증가를 반복하여 진동하는 방식**으로 최적점을 찾아가는 알고리즘을 많이 사용한다. 이러한 방법 중 가장 간단하면서도 많이 사용되는 방법이 CosineAnnealingLR이다.

Parameters

- optimizer: 이전에 정의한 optimizer 변수명을 넣어준다.
- T_max: 최대 iteration 횟수
- eta_min: 최소로 떨어질 수있는 learning rate default=0

```
optimizer = torch.optim.SGD(model.parameters(), lr=0.001)
scheduler = optim.lr_scheduler.CosineAnnealingLR(optimizer,
T_max=50, eta_min=0)
```

최종 선정 모델

- 선정 모델 : Facenet_pytorch

선정 이유

- ‘얼굴 인식 무인 보관함’이라는 타이틀에 맞는 서비스를 구현하기 위해서 Real-Time Face Recognition이 중요하다고 생각함.
- 또한 두 모델 전부 외국인 얼굴 데이터셋으로 pretrained된 모델이므로 AI HUB에서 수집한 한국인 얼굴 데이터셋으로 Fine Tuning의 필요성이 존재함.
- Facenet_pytorch 모델과 InsightFace 모델의 구조상 Fine tuning 하여 재학습하기에 Facenet_pytorch 모델의 구조가 상대적으로 편리함.
- 그러므로 비교적 가벼운 모델인 Facenet_pytorch를 채택함.