



인공지능과 응용 팀프로젝트 발표

팀명: TT(오후반 2조)

팀원명: 20201737 백민우,
20227130 장하린,
20222009 진나영,
20222016 황민호

TABLE OF CONTENTS

01 Image classification

02 데이터 분석 및 데이터증대

03 모델 선정

04 하이퍼파라미터 조정

05 결과 분석

06 프로젝트 후기
(진행 과정, 역할, 코드
정보, 배운점)

01

Image classification



Image classification

1. 경진대회 주제

Image Classification 모델을 학습하여 주어진 이미지가
실제 사람의 얼굴 이미지(live)인지 스푸핑 공격 얼굴 이미지(spoof)인지 판별
+) 테스트셋에 일부 unseen domain 데이터가 존재

2. 도출할 결과

△ filename	# label
10001.jpg	0
10002.jpg	0
10003.jpg	0
10004.jpg	0

- A. 제출 양식 구성
 - 특성: filename, label
 - 0(live), 1(spoof)
- B. 목표: test set의 label을 예측
- C. 심사기준: Accuracy

02

데이터 분석 및 데이터 증대



데이터 분석 및 데이터 증대

필요한 데이터 => 얼굴 데이터

따라서, 얼굴을 잘라내는 과정이 필요

A. 기존의 사진 크기 조절 과정

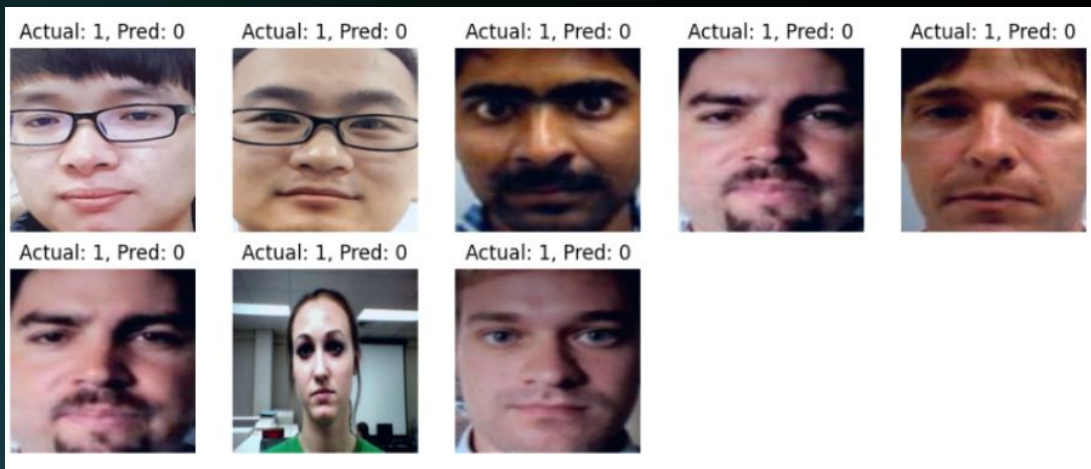
```
transforms.Resize((256, 256)),
```

B. 얼굴 위주로 잘라내기 위해 CenterCrop(224) 추가

```
transforms.Resize(256, 256)  
transforms.CenterCrop(224),
```

데이터 분석 및 데이터 증대

C. 검증 데이터셋의 오류 이미지 탐색



공통점

- 화질이 낮다.
- 얼굴 전체가 나오지 않는다

데이터 분석 및 데이터 증대

D. 오류 이미지를 바탕으로 이미지 데이터 증대

```
transform = transforms.Compose([
    transforms.Resize((320, 320)),
    transforms.CenterCrop(300),
    transforms.RandomApply([transforms.RandomHorizontalFlip()], p=0.2),
    transforms.Lambda(lambda img: jpeg_compression(img, quality=55)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
    1)
```


- Resize와 CenterCrop을 늘려서 사람 얼굴 전체가 나오도록 한다,
- transforms.RandomHorizontalFlip()를 통해 얼굴 이미지의 좌우 반전함으로써 얼굴 이미지의 방향성을 다양화
- 낮은 품질로 압축하는 전처리를 통해 낮은 품질 이미지에 대한 학습을 강화
(quality 값 조절해가며 test를 진행, 45, 65 -> 모두 낮은 결과를 도출, 55를 최종적으로 선택)

03

모델 선정



모델 선정

baseline(resnet18)	0.8787
efficientnet	0.9575
MLPMixer  열기	0.8237
Mobilenet_v3	0.9075
Inception-v3	0.9437

5가지 모델 성능 측정

resnet18

→resnet의 한 종류, 18개의 레이어로 구성

→네트워크가 깊어질수록 기울기 소실 문제를 완화

→기존의 CNN 구조의 문제 해결을 위해 잔차 학습 개념을 도입한 모델

5가지 모델 성능 측정

<u>basisline(resnet18)</u>	0.8787
<u>efficientnet</u>	0.9575
<u>MLPMixer</u>  열기	0.8237
<u>Mobilenet_v3</u>	0.9075
<u>Inception-v3</u>	0.9437

Efficientnet-B0

→ 모델의 크기, 깊이, 너비를 효율적으로 조정하여 성능과 효율성의 균형을 맞추는데 중점을 둔 모델

→ imagenet 데이터셋의 성능을 기준으로 선택된 최적의 네트워크 구조를 가지고 있다.

→ b0 부터 b7까지의 버전이 존재.

5가지 모델 성능 측정

baseline(resnet18)	0.8787
efficientnet	0.9575
MLPMixer  열기	0.8237
Mobilenet_v3	0.9075
Inception-v3	0.9437

MLPMixer

→합성곱 연산을 사용하지 않고도 이미지 분류 작업을 수행할 수 있는 모델.

→합성곱 연산이 없다는 것은 모델의 설계와 계산 비용 측면에서 큰 장점

5가지 모델 성능 측정

<u>baseline(resnet18)</u>	0.8787
<u>efficientnet</u>	0.9575
<u>MLPMixer</u>  열기	0.8237
<u>Mobilenet_v3</u>	0.9075
<u>Inception-v3</u>	0.9437

MobileNetV3

→ 모바일 및 사물인터넷에서 효율적으로 이미지 처리를 위한 모델

→ 여러 가지 버전이 존재하며 모델의 크기와 성능에 따라 트레이드 오프를 조정할 수 있도록 한 모델.

5가지 모델 성능 측정

<u>basisline(resnet18)</u>	0.8787
<u>efficientnet</u>	0.9575
<u>MLPMixer</u>  열기	0.8237
<u>Mobilenet_v3</u>	0.9075
<u>Inception-v3</u>	0.9437

inception -v3

→ 네트워크의 깊이와 너비를 효율적으로 확장한 모델.

→ CNN 아키텍처로, inception모듈을 사용하는 모델

→ inception 모듈은 다양한 크기의 커널을 동시에 사용하여 네트워크가 다양한 크기의 특징을 동시에 학습할 수 있도록 기여.

5가지 모델 성능 측정

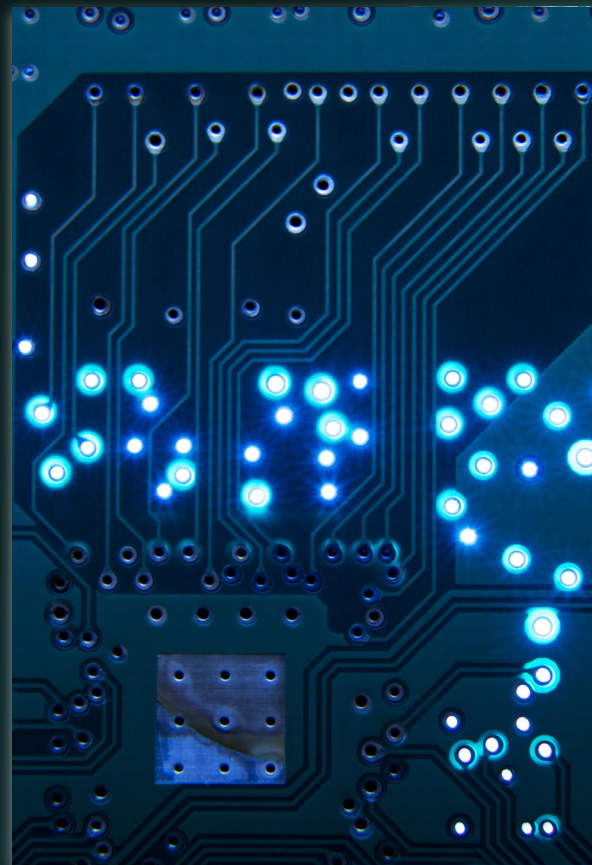
<u>baseline(resnet18)</u>	0.8787
<u>efficientnet</u>	0.9575
<u>MLPMixer</u>  열기	0.8237
<u>Mobilenet_v3</u>	0.9075
<u>Inception-v3</u>	0.9437

성능 측정 결과에 따라,

-> Efficientnet을 최종 모델로 선정

04

하이퍼파라미터 조정



1. 손실함수

ArcFace : 각 클래스의 중심점(centroid)으로부터 샘플의 각도를 조정하여 더 뚜렷한 경계를 만든다

- 얼굴 인식과 같은 인식 문제에서 강력한 성능을 보이는 손실함수



trial_3 - Version 2

Complete · 백민우_20201737 · 16d ago · 데이터 증대

0.9558

0.9525



trial_3_1 - Version 3

Complete · 백민우_20201737 · 15d ago · ArcFace 수정

0.9641

0.9575

```
from efficientnet_pytorch import EfficientNet
from arcface import ArcFace
from torchvision import models

model = EfficientNet.from_pretrained('efficientnet-b0')
num_fters = model._fc.in_features
model.fc = ArcFace(num_fters, 2, m=0.5) #추가됨
model = model.to(device)
```

→ Arcface 적용 후 성능 점수가 더 향상함에 따라 efficientnet에 손실함수 Arcface를 적용했다.

2. 스케줄러

1. <u>multistepLR</u>	 열기	특정 에포크에서 학습률 저하 (2번째 4번째 에포크)	0.9575로 기존 코드와의 점수 차이 x
2. <u>ExponentialLR</u>		매 에포크마다 0.9비율만큼 학습률 감소	0.9575로 기존 코드와의 점수 차이 x
3. <u>cosineAnnealingLR</u>		주기마다 학습률을 코사인 함수 형태로 조정하여 감소시킴	0.9575로 기존 코드와의 점수 차이 x
4. <u>cyclicLR</u>		주기마다 학습률을 최대 최소로 왔다갔다	0.9575로 기존 코드와의 점수 차이 x
5. <u>onecycleLR</u>		하나의 학습 주기에서 학습률을 증가시키고 이후에 감소시키기	0.9575로 기존 코드와의 점수 차이 x

1. 특정 에포크마다 과적합 방지를 위해 학습률을 저해시키는 스케줄러를 사용.
2. 비교 대상은 베이스 라인이었던 stepLR을 기준으로 비교.
3. 모든 스케줄러 함수가 베이스 라인인 stepLR과 스코어 차이가 없었다

→ 학습에 사용된 에포크 수가 적기 때문에 학습률 조정의 효과를 발휘하지 못했다

3. 옵티마이저

옵티마이저란?

→ 기계 학습에서 모델의 가중치나 파라미터를 업데이트하여 손실 함수를 최소화하는 알고리즘.

Adam(Adaptive Moment Estimation)

→ Adam은 1차 모멘트(기울기)의 이동 평균과 2차 모멘트(기울기 제곱)의 이동 평균을 사용하여 학습률 조정.

AdamW(baseline)

→ Adam의 단점이었던 기울기 기반 가중치 감소를 순수 가중치 감소로 바꿔 L2정규화가 더 잘 수행되도록 업데이트된 옵티마이저.

→ **Adam**을 옵티마이저로 설정했을 때 점수가 **0.5037**로 현저히 떨어짐에 따라 **AdamW**를 채택하였다

4. 앙상블

(1) Efficientnet b0, b1, b2 ensemble → 결과들의 평균으로 최종 결정



trial_Ensemble_Model - Version 10

Complete · 백민우_20201737 · 1d ago

0.9425

0.9475

(2) resnet18, inception-v3, efficientnet-b0 ensemble



ensemble_1.csv

Complete · HARIN JANG · 21h ago · 'resnet18', 'inception_v3', 'efficientnet_b0' 세 모델로 앙상블 진행

0.9316

0.9337

결과 분석

05



결과 분석

A. Train set, Val set의 accuracy

100%|██████████| 219/219 [04:36<00:00, 1.26s/it]

train_accuracy: 0.9969285714285714

100%|██████████| 94/94 [01:28<00:00, 1.06it/s]

val_accuracy: 0.9991666666666666

B. Test set의 accuracy

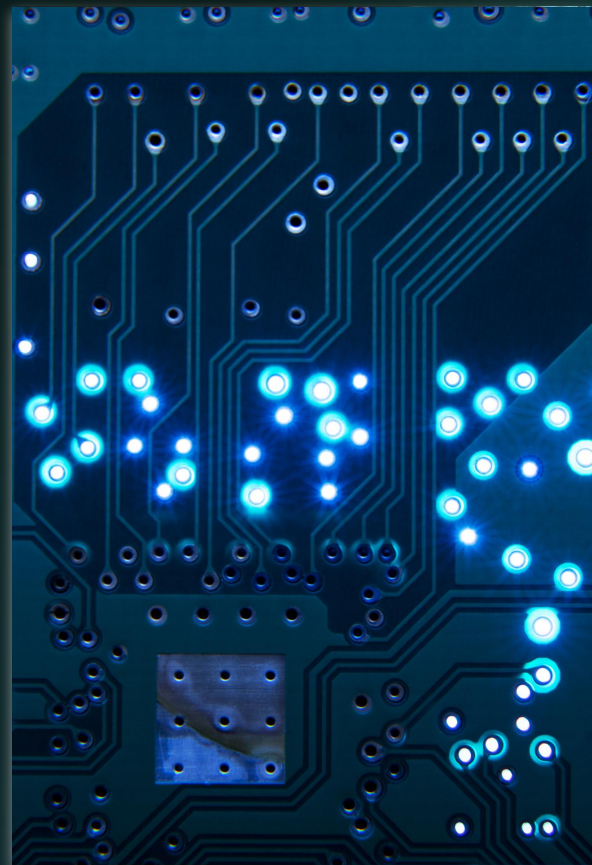
0.9712

- Train set과 Val set의 label을 예측하는 성능이 좋은 걸로 보아 주어진 데이터를 제대로 학습한 것으로 보인다.
- 그러나, test set에선 결과가 비교적 낮은 걸로 보아 unseen domain 처리가 더 필요해 보인다.

06

프로젝트 후기

(진행 과정, 역할, 코드 정보, 느낀점)



프로젝트 후기 - 진행 과정

1. 데이터 분석 각자 주어진 데이터를 분석하고 분석한 결과를 회의를 통해 공유
2. 모델 탐색 공유한 데이터 분석 결과를 바탕으로 적합한 모델을 각자 탐색
3. 모델 선정 탐색한 모델과 적용 결과를 공유한 후 가장 성능이 좋았던 모델을 선정
4. 이미지 증대
및
하이퍼파라미
터 조절 이미지 증대, 하이퍼 파라미터 조절 등으로 역할을 나눠
모델의 성능을 개선

최종적으로 모델의 앙상블 시도
5. 앙상블

프로젝트 후기 - 역할



프로젝트 자료 정리

20201737 백민우



PPT 제작

20222009 진나영,
20222016 황민호



발표

20227130 장하린

프로젝트 후기 - 코드 정보

```
from efficientnet_pytorch import EfficientNet
from arcface import ArcFace
from torchvision import models

model = EfficientNet.from_pretrained('efficientnet-b0')
num_fts = model._fc.in_features
model.fc = ArcFace(num_fts, 2, m=0.5)
model = model.to(device)
```

- **efficientnet b0**

```
pip install efficientnet_pytorch
```

<https://github.com/lukemelas/EfficientNet-PyTorch>

- **ArcFace**

```
!git clone https://github.com/shyhyawJou/ArcFace-Pytorch.git
%cd ArcFace-Pytorch
```

<https://github.com/shyhyawJou/ArcFace-Pytorch>

Permissions

- ✓ Commercial use
- ✓ Modification
- ✓ Distribution
- ✓ Patent use
- ✓ Private use

프로젝트 후기-느낀점

적합한 모델을 탐색하며 여러 모델에 대해 공부할 수 있었고
다양한 성능 개선 방법을 직접 적용해볼 수 있어
인공지능 공부에 도움이 되는 시간이었습니다.

수업과 프로젝트를 통해 인공지능과 캐글 활용에 익숙해질 수
있었습니다. 수업이 끝나고도 캐글을 활용하여 인공지능 공부를
계속 이어나가려 합니다.

THANKS!

Do you have any questions?



CREDITS: This presentation template was created by [Slidesgo](#), and includes icons by [Flaticon](#), and infographics & images by [Freepik](#)

Please keep this slide for attribution

