Monday's
Facial Expression Recognition

딥러닝 기술을 이용한 얼굴 표정 인식

2021.04.26

유영현, 안준언, 김화연, 서민하, 김기범

목차

Monday's Facial Expression Recognition

프로젝트 개요

1. 프로젝트 개요

1. 주제 선정 배경

“ 딥러닝 기술을 이용하여 얼굴 표정 인식 기법을 제안 ”

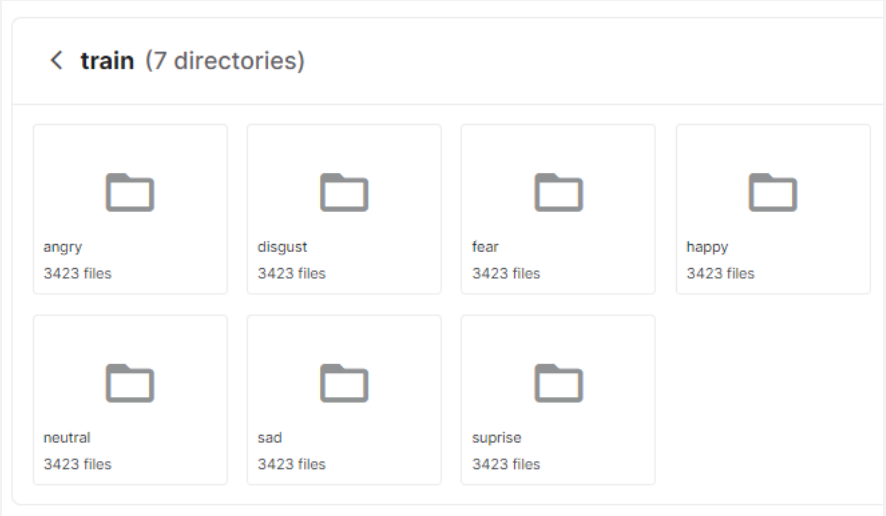
FER 2013 데이터 셋을 사용한 경우 정확도 66.67% 까지 가능

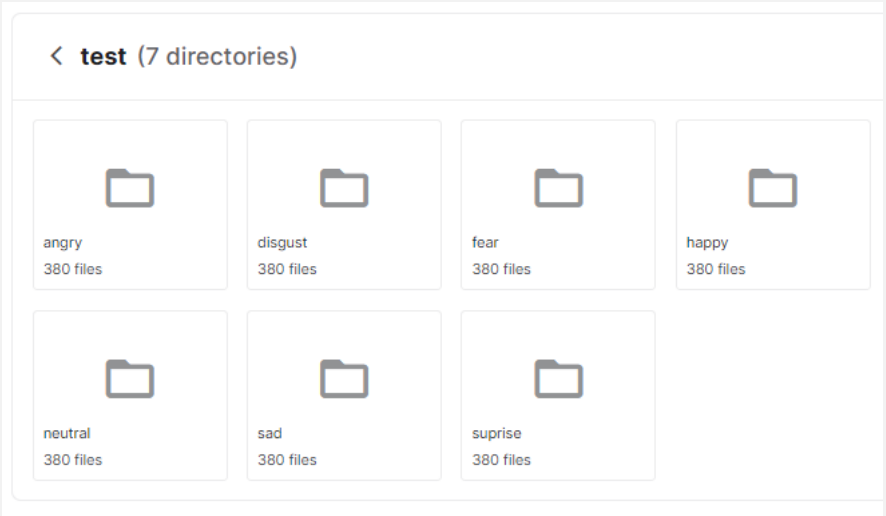
✓ 목표: 다른 데이터 셋 사용하여 비슷한 수준의 정확도 구현하기

1. 프로젝트 개요

2. Raw Data

kaggle

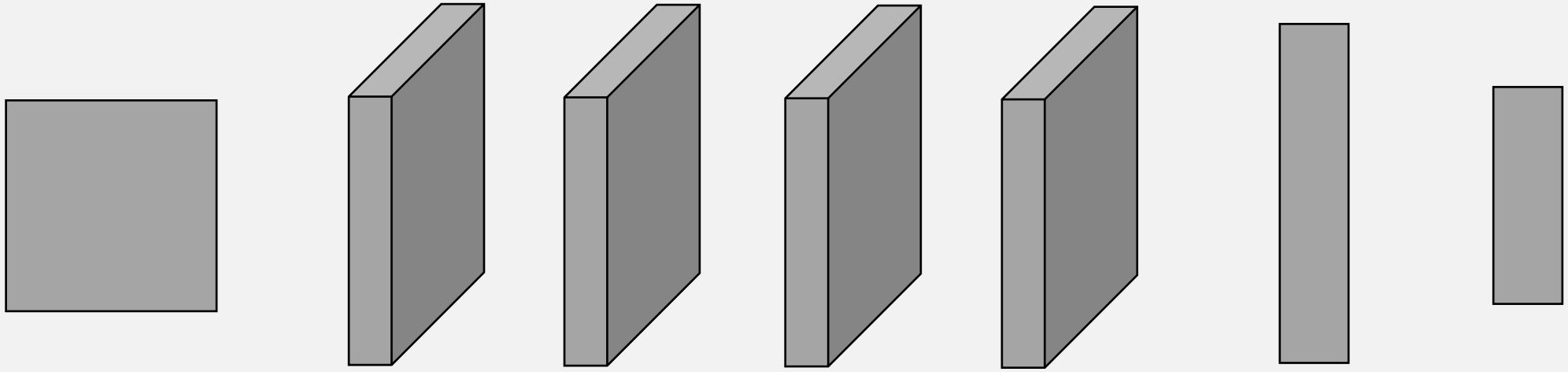




- ✓ Data는 Kaggle 사이트의 'telemoji' 데이터를 이용
- ✓ Angry, Disgust, Fear, Happy, Neutral, Sad, Surprise의 7가지 감정으로 분류
- ✓ Train: 3423*7, Test: 380*7 => 26,621 개로 구성
- ✓ 이미지 사이즈와 데이터 크기는 각각 상이함

1. 프로젝트 개요

3. 전처리 이전 성능



input
150, 150

conv 32,
Maxpooling

conv 64,
Maxpooling

conv 128,
Maxpooling

conv 128,
Maxpooling

FC 7,
Softmax

1. 프로젝트 개요

3. 전처리 이전 성능

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
150,150,3	rescale	1e-4	20	Adam	0.5		1.6102 0.4874
150,150,1	rescale	1e-4	20	Adam	0.5		1.6038 0.4892

전처리 이전의 문제점

1. 데이터의 크기가 달라서 ImageDataGenerator로 가져올때 시간이 많이 걸림
2. 분류가 적절하지 않은 이미지가 존재
3. 얼굴이 아닌 다른 요소가 들어간 이미지가 존재

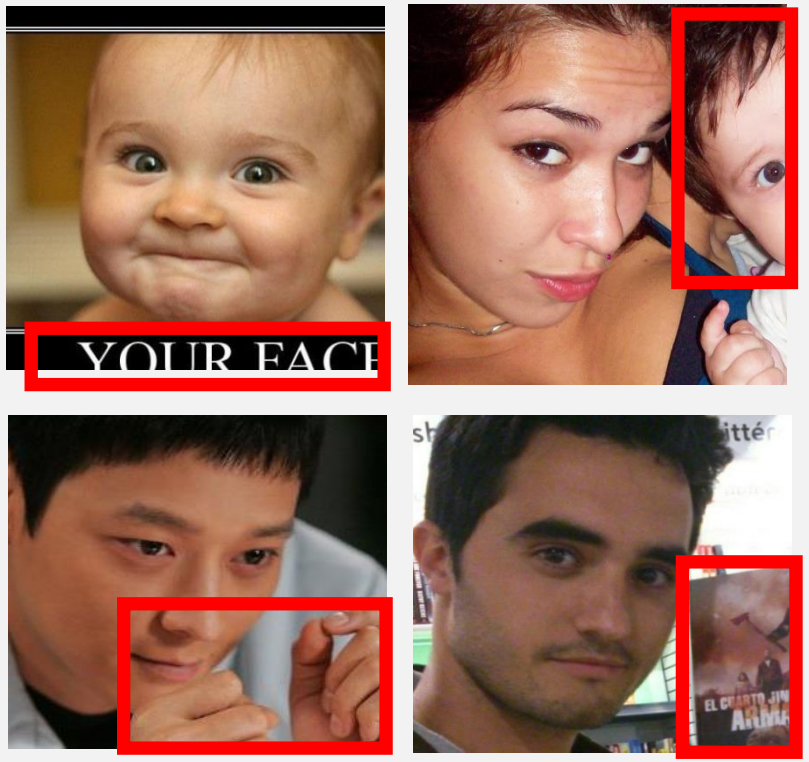
-> 따라서 모델 탐색 이전에 데이터 전처리가 필요하다고 판단

Monday's Facial Expression Recognition

데이터 전처리

2. 데이터 전처리

1. Raw Data의 문제점



얼굴 주변 글자, 신체 일부분, 사물 등 존재

↓

모델의 학습 정확도 저해요소가 됨

↓

얼굴만 정확히 Extraction이 필요

2. 데이터 전처리

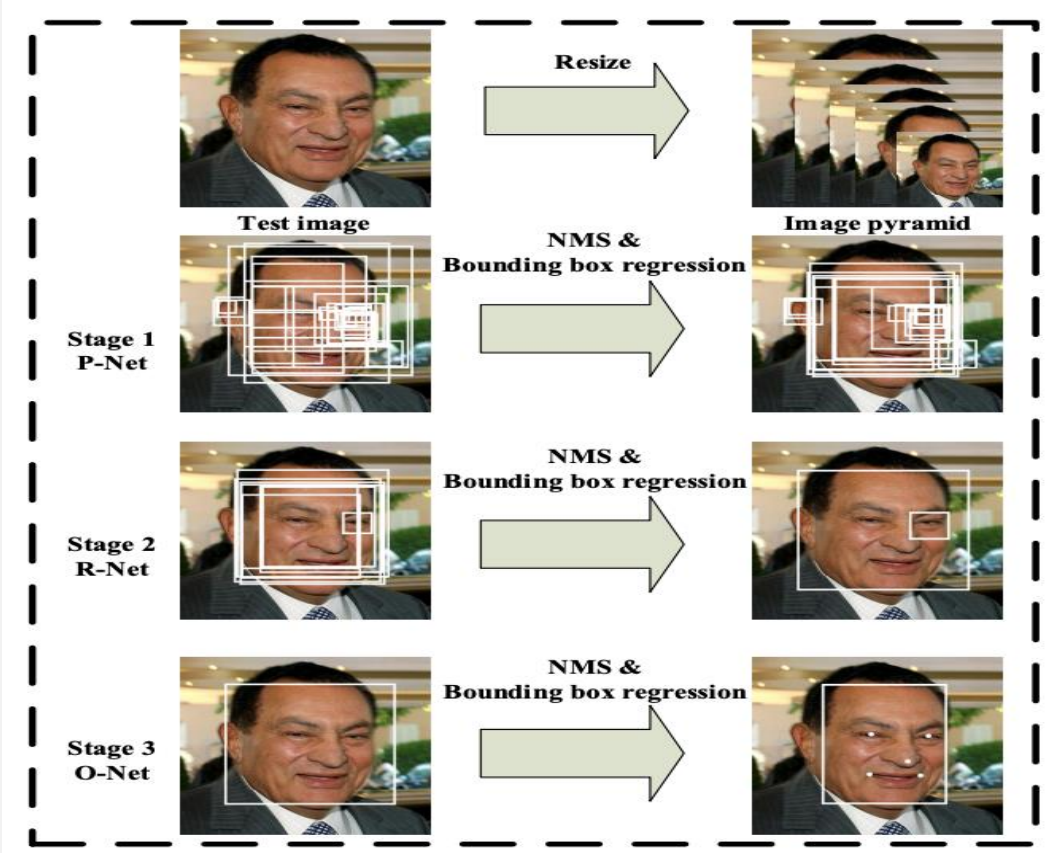
2. MTCNN

얼굴을 검출하는 딥러닝 모델

- 3개의 Neural network(P-Net, R-Net, O-Net) 구성
- Face detection, Bounding box regression, Face alignment 세 가지 테스크를 동시에 학습시키는 Joint learning 방식

✓ 기존의 다른 방법에 비해 정확도가 높고 속도가 빠름

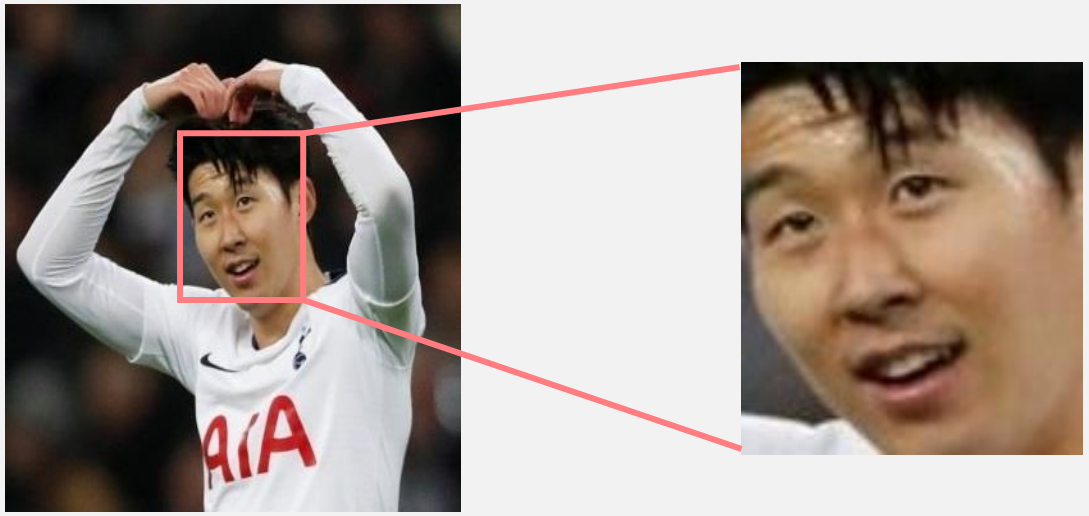
✓ 얼굴 검출 정확도 95%, 약 1300회 인용



2. 데이터 전처리

2. MTCNN

얼굴만 추출한 데이터로 전처리

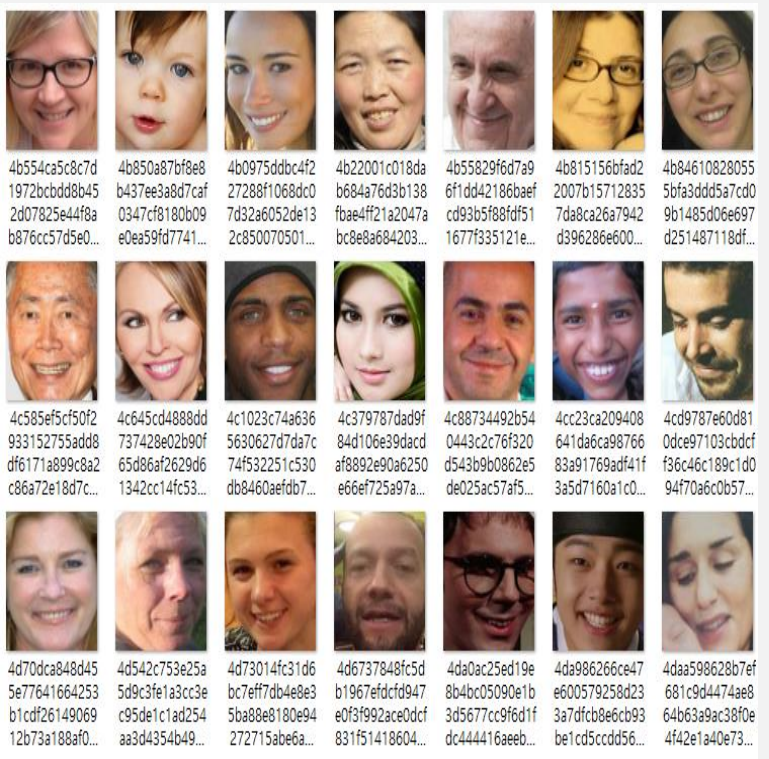


Raw Data

이미지 내에 얼굴 외에
학습 방해 요소가 존재

MTCNN

MTCNN이 얼굴만을 인식,
추출한 이미지를 Return



AFTER

2. 데이터 전처리

3. OCR

OCR (Optical character recognition) API를 사용하여 글자 인식되는 그림 제거

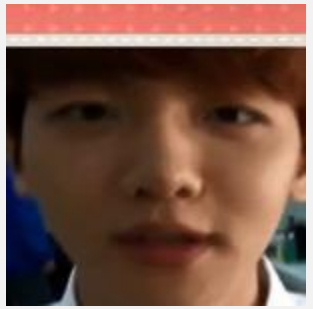
2. 데이터 전처리

3. 재분류

수동 분류 기준

- ✓ OCR로 걸러지지 않은 사진
- ✓ 측면 사진
- ✓ 손, 담배, 음식, 선글라스 등으로 눈 또는 입 주변을 가린 사진
- ✓ 얼굴 이외 그림이 있는 사진
- ✓ 화질이 낮은 사진

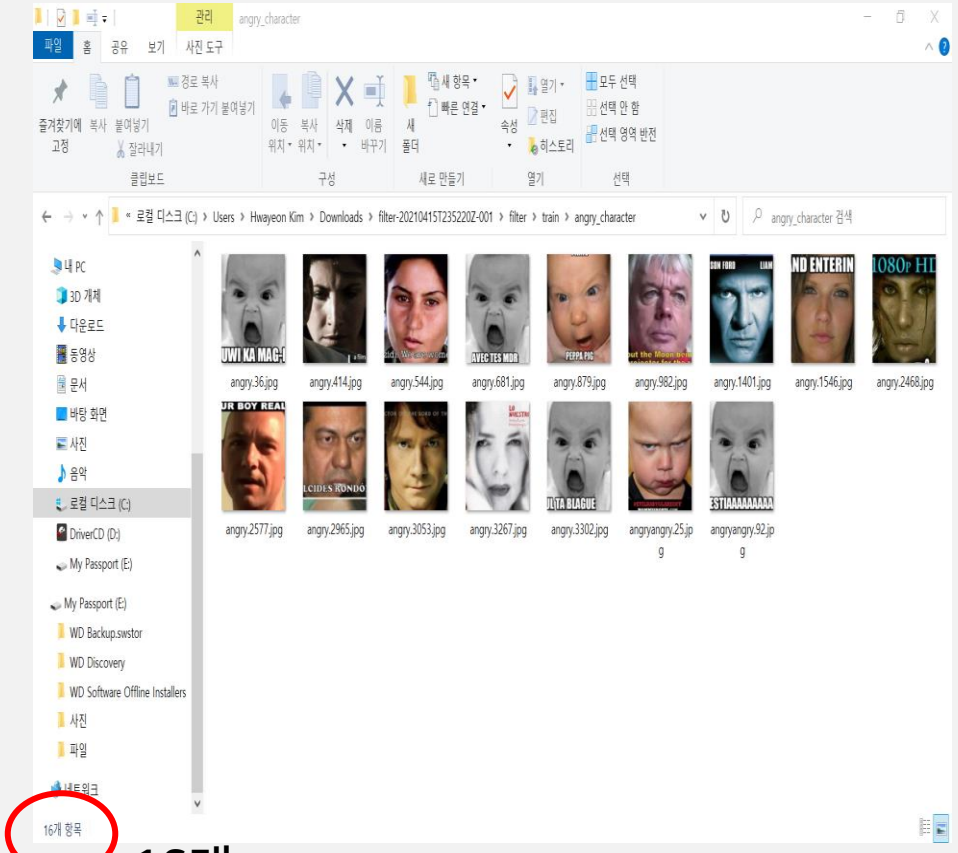


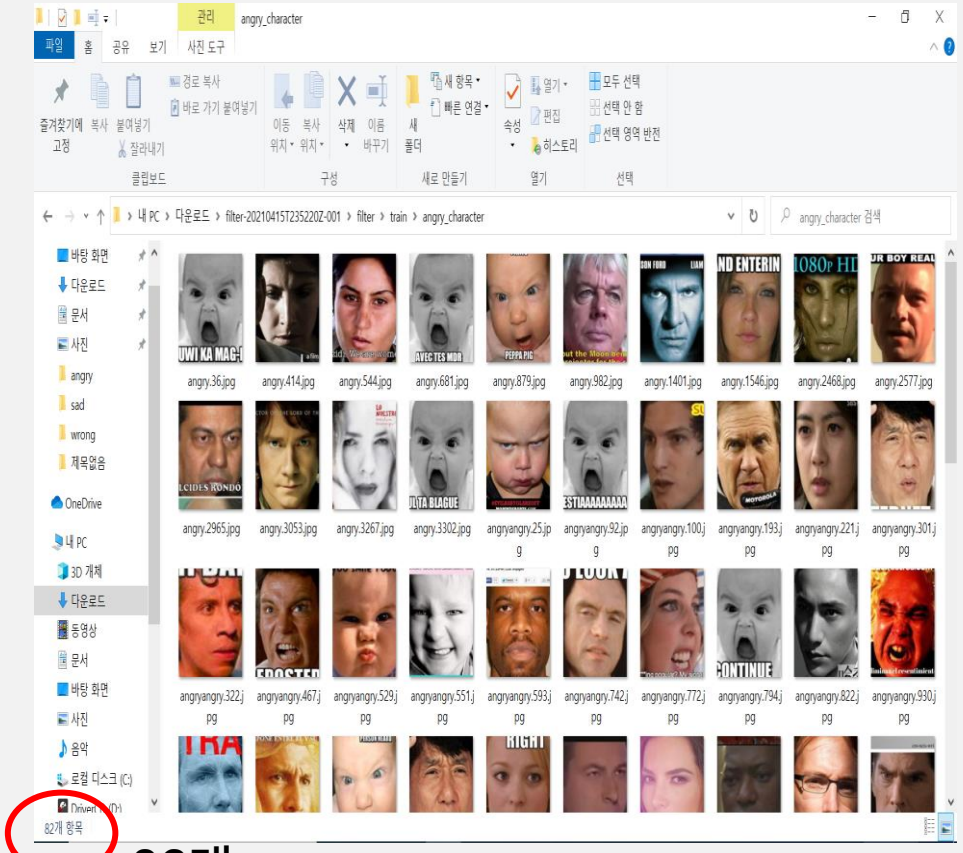

2. 데이터 전처리

4. 재분류

분류 결과 : Angry 폴더 예시



16개



82개

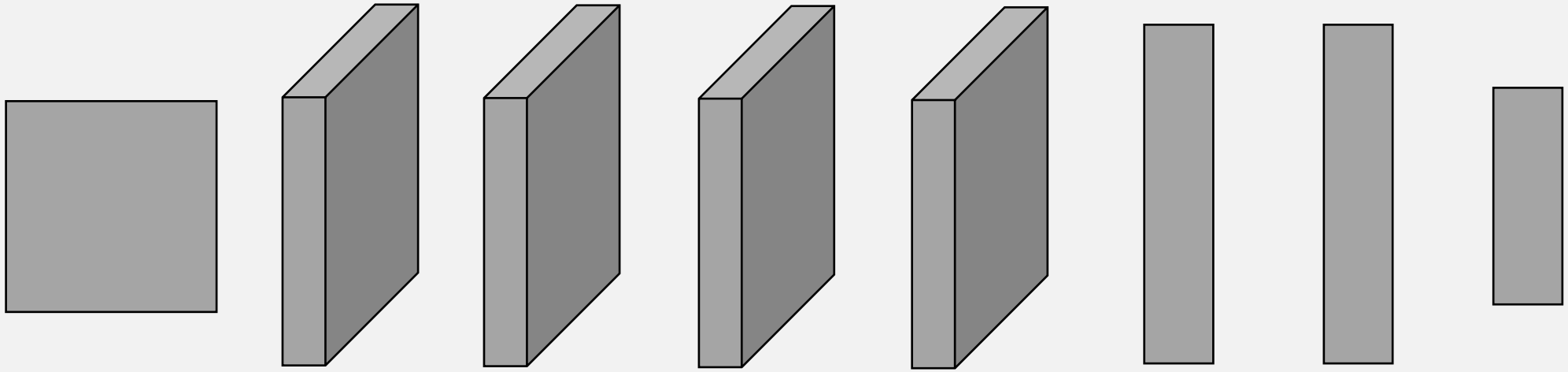
Monday's Facial Expression Recognition

프로젝트 진행과정

3. 프로젝트 진행과정

1. 논문 모델 1

“Convolutional Neural Networks for Facial Expression Recognition”



input
150, 150

conv 64, BN,
Maxpooling,
Dropout

conv 128, BN,
Maxpooling,
Dropout

conv 512, BN,
Maxpooling,
Dropout

conv 512, BN,
Maxpooling,
Dropout

FC 256,
BN, ReLU,
Dropout

FC 512,
BN, ReLU,
Dropout

FC 7,
Softmax

3. 프로젝트 진행과정

1. 논문 모델 1

Batch Normalization

- ✓ Batch의 평균과 분산을 이용해서 이전 layer의 출력값을 Normalization
- ✓ scale과 shift를 위한 γ와 β를 추가

- ✓ 정규화 시켰던 부분을 원래대로 돌리는 identity mapping 가능
- ✓ 학습을 통해 γ와 β를 정할 수 있어 훨씬 강력한 학습 가능

: Convolutional layer에서 나온 결과를 정규화하여 vanishing gradient를 예방하는 방법

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

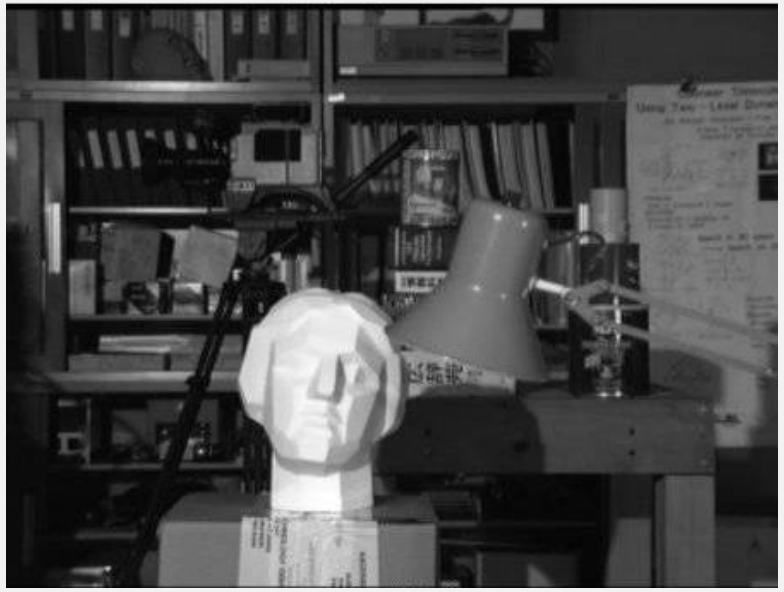
$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

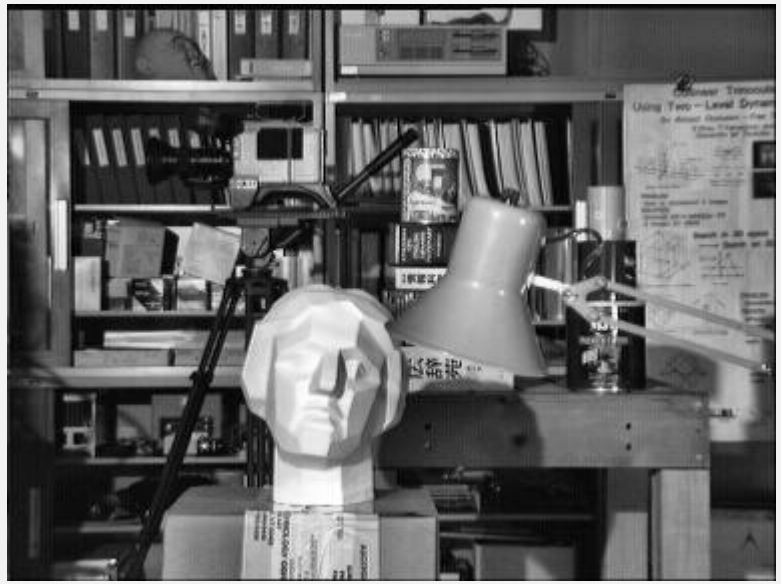
3. 프로젝트 진행과정

1. 논문 모델 1 – 특징 추출

1) CLAHE (Contrast Limited Adaptive Histogram Equalization)

이미지의 대조를 조절하여 선명한 이미지를 얻는 방법



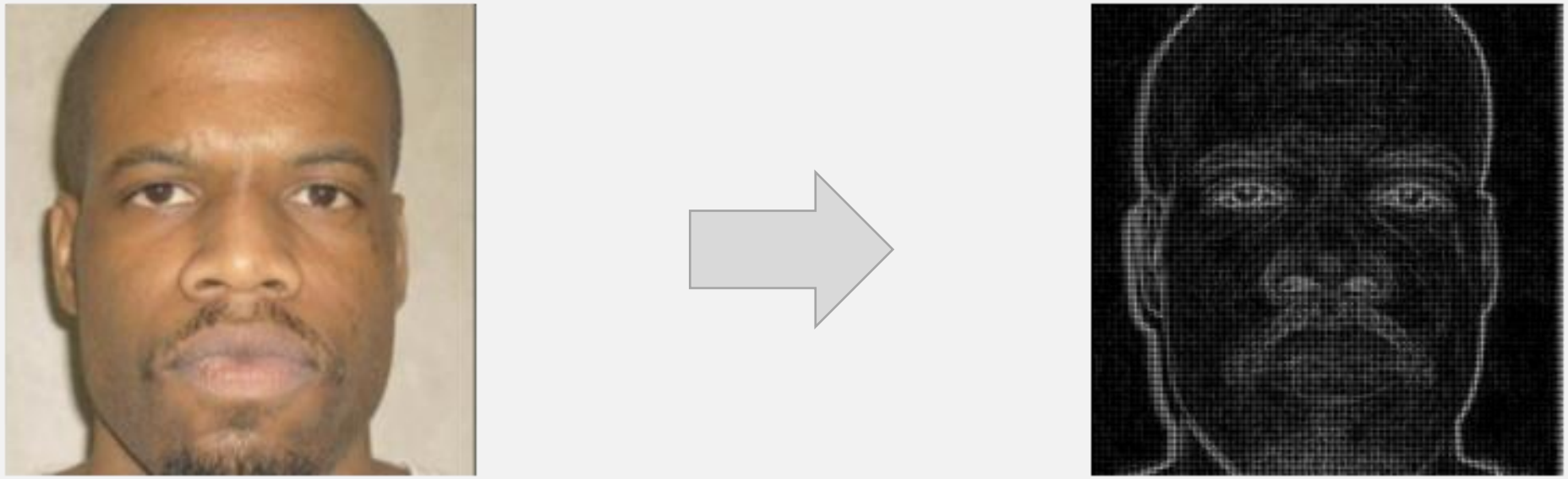


3. 프로젝트 진행과정

1. 논문 모델 1 – 특징 추출

2) HOG (Histogram of Oriented Gradient)

대상 영역을 일정 크기의 셀로 분할하고, 각 셀마다 edge 픽셀(gradient magnitude가 일정 값 이상인 픽셀)들의 방향에 대한 히스토그램을 구한 후 이들 히스토그램 bin 값들을 일렬로 연결한 벡터



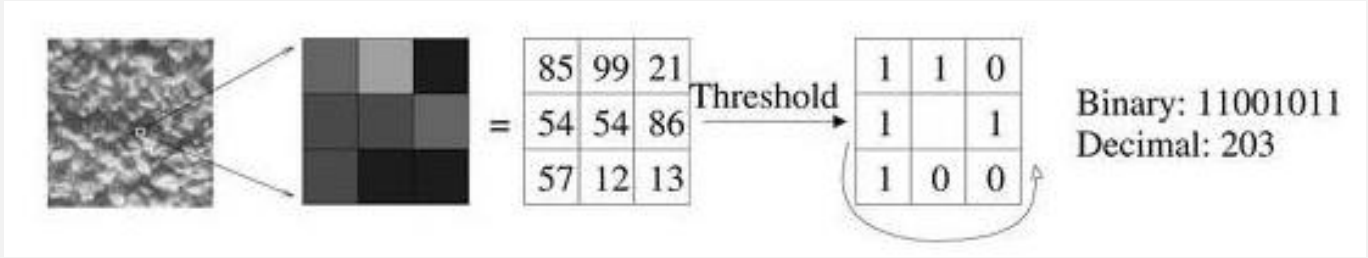
3. 프로젝트 진행과정

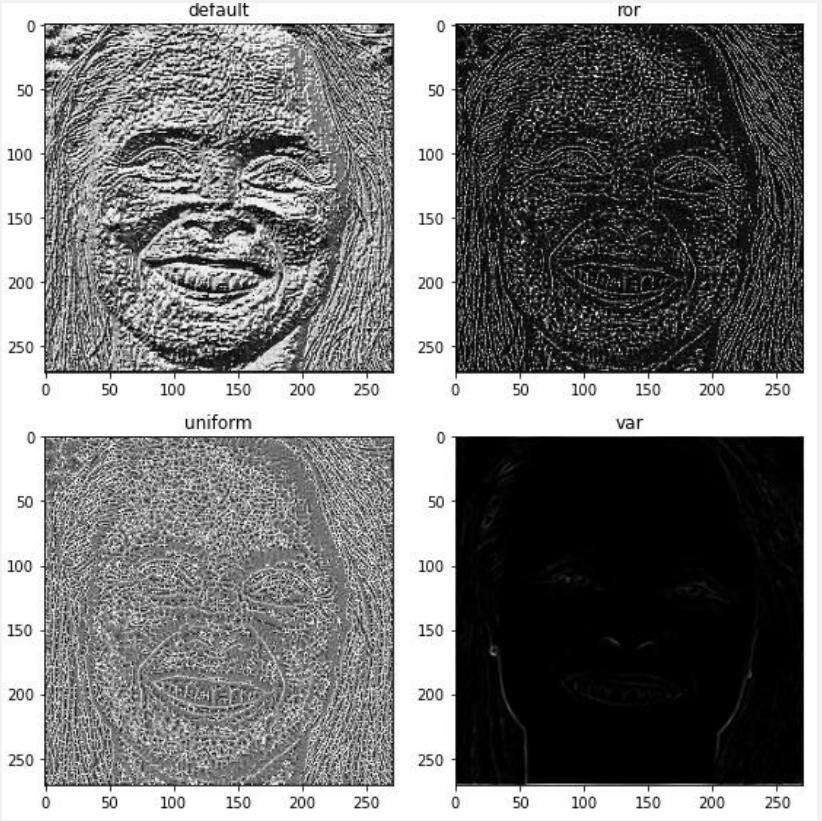
1. 논문 모델 1 – 특징 추출

3) LBP (Local Binary Pattern)

LBP는 원래 영상의 텍스쳐(texture)를 분류하기 위한 용도로 개발된 feature인데
이후 얼굴인식(face recognition)과 같은 다른 영상인식 응용에도 활용되고 있다.

LBP는 영상의 모든 픽셀에 대해 계산되는 값으로서 각 픽셀의 주변 3 x 3 영역의
상대적인 밝기 변화를 2진수로 코딩한 인덱스 값이다





3. 프로젝트 진행과정

1. 논문 모델 1

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
150, 150, 1	rescale, 증식	1e-4	50	RMSprop	0.2/ 0.5		1.2673 0.5453
150, 150, 1	rescale, HE(5,4)	5e-5	30	RMSprop	0.2/ 0.5		2.7941 0.5424
150, 150, 1	rescale, HOG	1e-4	20	RMSprop	0.2/ 0.5		3.1272 0.5164
150, 150, 1	rescale, LBP(ror)	1e-4	20	RMSprop	0.2/ 0.5		2.6229 0.4701
150, 150, 1	rescale, LBP(var)	1e-4	20	RMSprop	0.2/ 0.5		2.6998 0.2715
150, 150, 1	rescale, HE(5,8)	1e-4	20	RMSprop	0.2/ 0.5		2.5209 0.5521

3. 프로젝트 진행과정

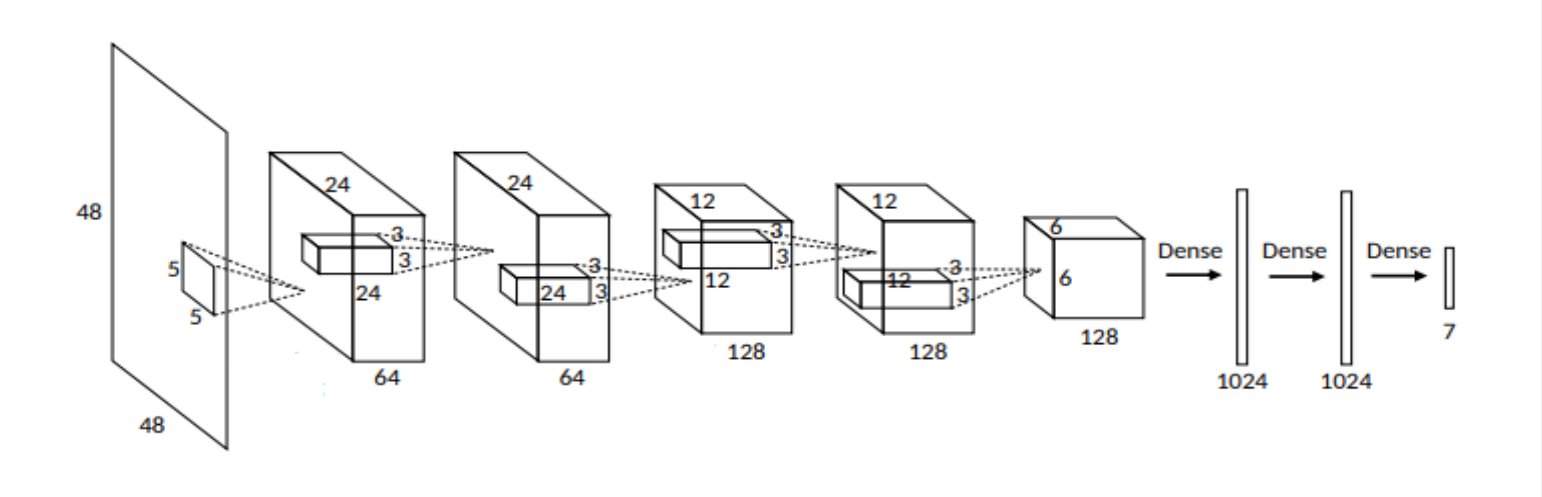
1. 논문 모델 1

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
150, 150, 1	rescale	1e-4	30	Adam	0.2/0.5	relu	1.8634 0.5603
150, 150, 1	rescale	1e-4	30	RMSprop	0.2/0.5	relu	2.6098 0.5718
150, 150, 3	rescale	1e-4	30	Adam	0.2/0.5	relu	1.9483 0.5477
150, 150, 3	rescale	1e-4	30	RMSprop	0.2/0.5	relu	2.5655 0.5516

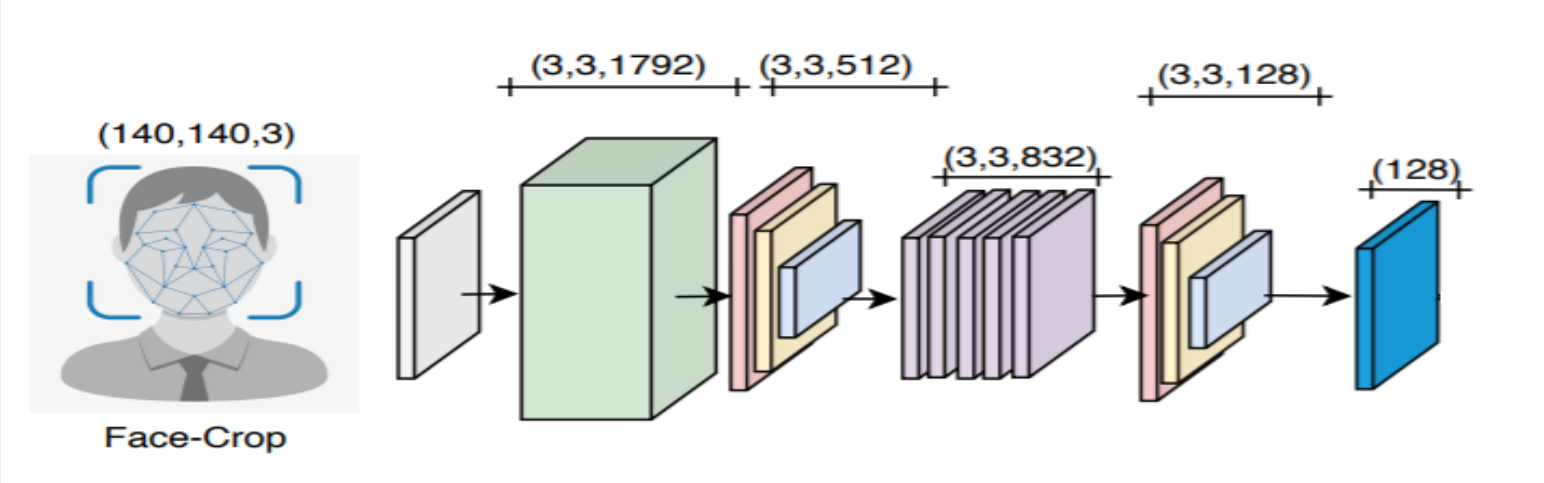
- ✓ 흑백처리
- ✓ RMSprop 사용
- ✓ 특징 추출 x
- ✓ 최종 ACC : 약 57%

3. 프로젝트 진행과정

2. 논문 모델 2



- ✓ Stochastic pooling 대신 max pooling 사용
- ✓ 앙상블 학습 X
- ✓ 논문에서는 두 가지 데이터 셋 사용



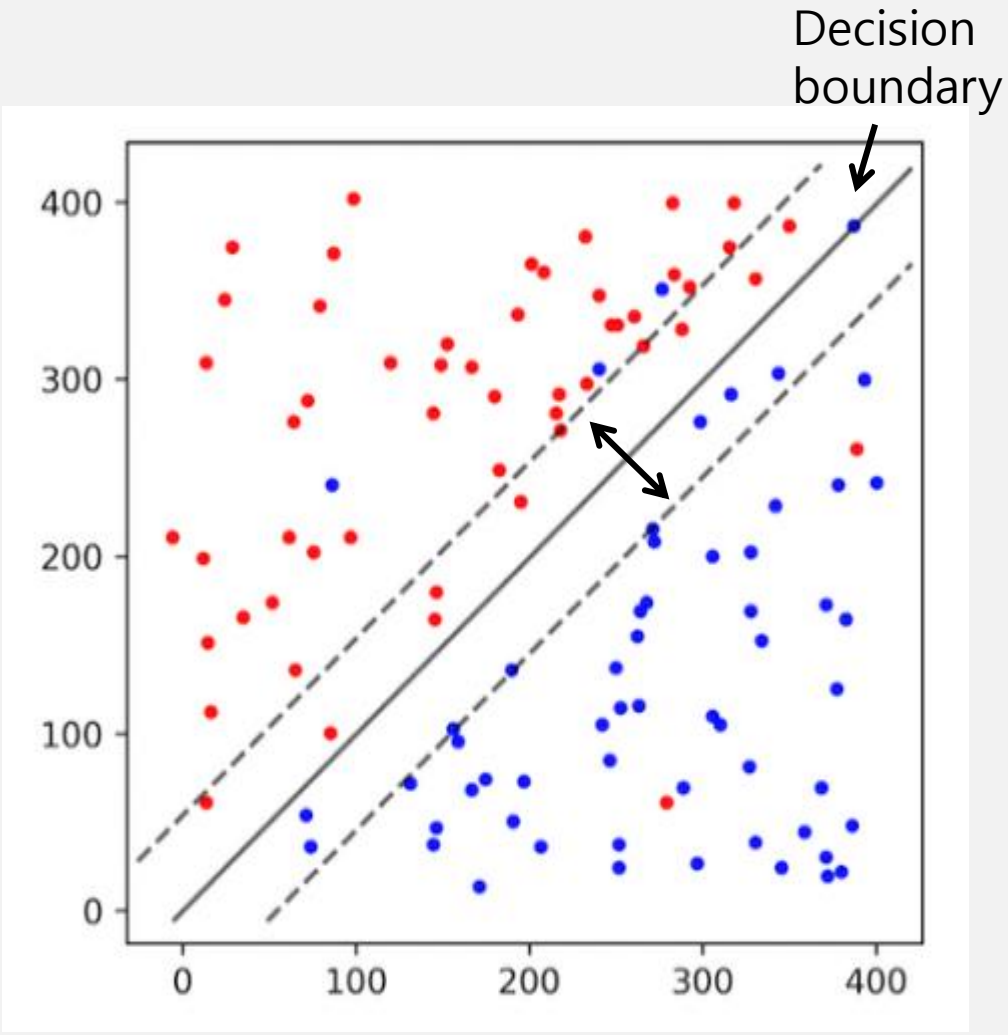
- ✓ Knowledge distillation의 teacher 모델만 사용
- ✓ InceptionResNetV2 + Fine Tuning + DNN + Convolution Layers
- ✓ 논문에서는 세 가지 데이터 셋 사용

3. 프로젝트 진행과정

2. 논문 모델 2

Hinge Loss

- ✓ 예측값과 실제값이 일치하면 loss=0
- ✓ 예측값과 실제값이 다르면 decision boundary와 예측값의 거리가 멀수록 loss 값 증가
- ✓ decision boundary 안에 있는 일정 범위 안의 값 = 불확실한 예측
  → 예측값과 실제값이 같아도 loss값 계산



3. 프로젝트 진행과정

2. 논문 모델 2

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
48, 48, 3	rescale	1e-3	20	RMSprop	X	Categorical Crossentropy	1.5499 0.5029
48, 48, 3	rescale	2e-3	20	Adam	X	Categorical Hinge	1.0033 0.2329
48, 48, 1	rescale	1e-3	20	RMSprop	X	Categorical Crossentropy	1.5831 0.5048
48, 48, 1	rescale	2e-3	20	Adam	X	Categorical Hinge	1.0041 0.2498
140, 140, 3	rescale	1e-3 1e-5	20	Adam	X	Categorical Crossentropy	1.7854 0.4870
140, 140, 3	rescale	1e-3 1e-5	20	Adam	0.5	Categorical Crossentropy	1.3977 0.5000

- ✓ 흑백처리 , RMSprop 사용
- ✓ Hinge loss x
- ✓ 최종 ACC : 약 50%

3. 프로젝트 진행과정

3. 전이학습 모델 - MobileNet V1

특징

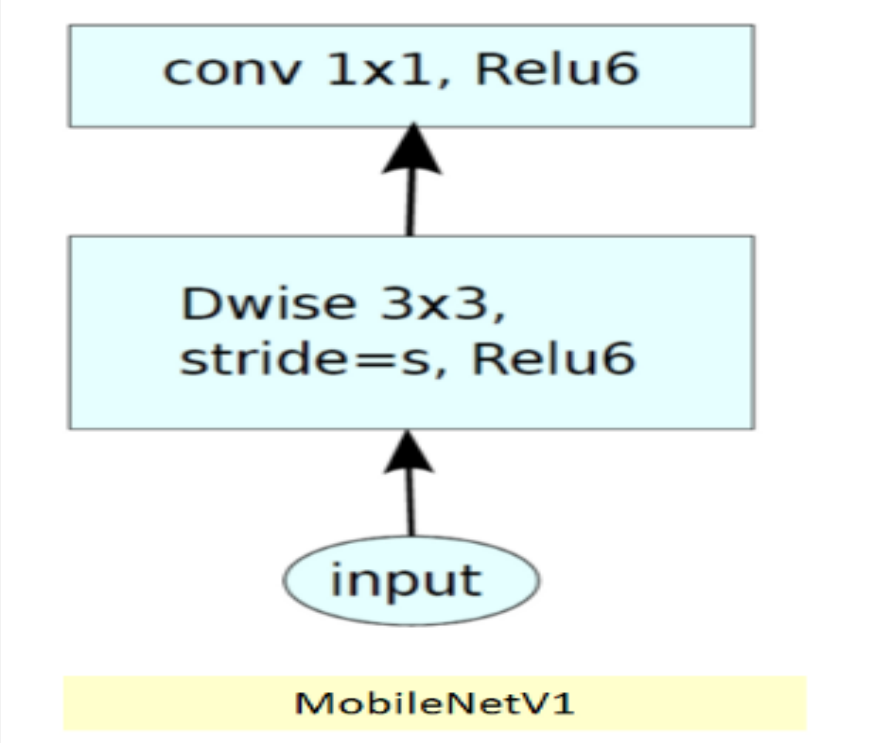
✓ 모바일 환경에서 구동시킬 수 있도록 파라미터량을 줄인 모델

Depthwise Convolution

✓ 채널별로 분리하여 각 채널을 각각의 커널로 convolution

Pointwise Convolution

✓ 출력의 채널을 바꿀 수 있으며, 1x1로 convolution



3. 프로젝트 진행과정

3. 전이학습 모델 - MobileNet V2

특징

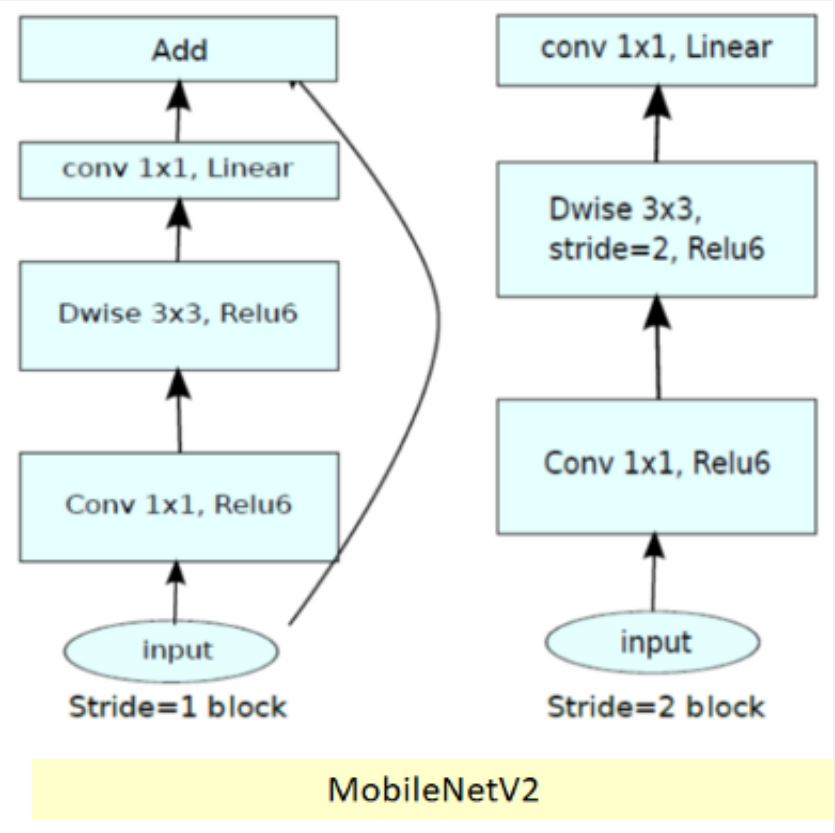
- ✓ V1보다 파라미터 수와 연산량을 줄인 모델
- ✓ 2가지 종류의 block 존재

Residual block

- ✓ Stride 1

Downsizing block

- ✓ Stride 2



3. 프로젝트 진행과정

3. 전이학습 모델 - MobileNet V3 Large, Small

특징

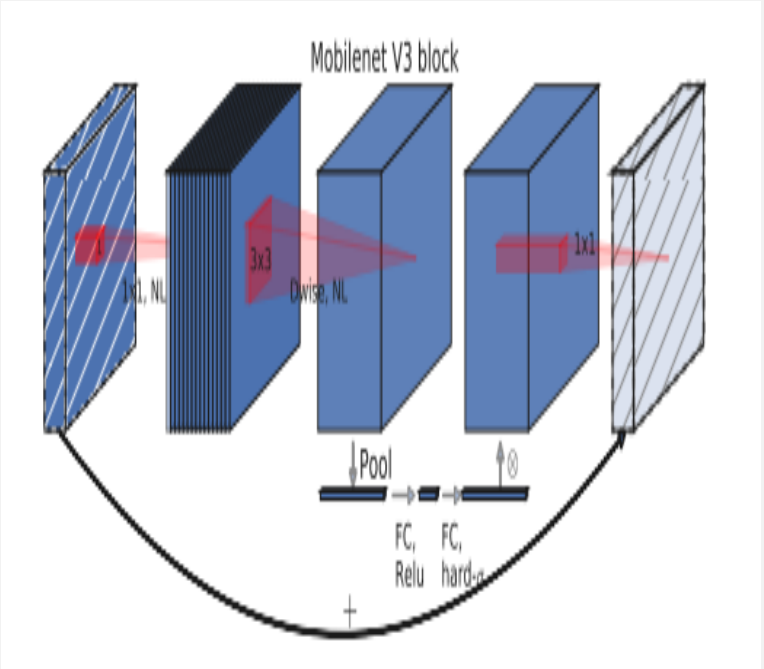
- ✓ mobile phone CPU에 최적화
- ✓ NetAdapt와 NAS를 조합한 구조
- ✓ Large와 Small 모델로 나뉨

Large

- ✓ High resource
- ✓ 정확도 3.2% 상승
- ✓ 시간 20% 감소

Small

- ✓ Low resource
- ✓ 정확도 6.6% 상승
- ✓ 시간은 동일



본 프로젝트에서는 MobileNet V2와 MobileNet V3 사용

3. 프로젝트 진행과정

3. 전이학습 모델 - MobileNet V2

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
150, 150, 3	rescale	1e-4	20	Adam	0.5		2.8986 0.5636
150, 150, 3	rescale	1e-4	20	RMSprop	0.5		9.3962 0.5347
150, 150, 3	rescale	1e-4 1e-5	20	Adam	0.55	Finetuning	1.6514 0.4841
150, 150, 3	rescale	1e-4	20	Adam	0.55	DNN	2.624 0.5612
150, 150, 1	Rescale 데이터증식	1e-4	20	Adam	0.5		1.1813 0.5911

3. 프로젝트 진행과정

3. 전이학습 모델 - MobileNet V3 Large

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
150, 150, 3	x	1e-4	20	Adam	0.5		2.3471 0.5897
150, 150, 3	x	1e-4	20	RMSprop	0.5		2.9252 0.5805
150, 150, 3	x	1e-4 1e-5	20	Adam	0.55	Fine Tuning 적용	1.4318 0.5309
150, 150, 1	데이터 증식	1e-4	20	Adam	0.5		1.2277 0.6162
150, 150, 1	데이터 증식	1e-4	20	Adam	0.5	DNN 추가	1.2139 0.6095
150, 150, 1	데이터 증식	1e-4	20	SGD	0.5		학습중단
150, 150, 1	HE	1e-4	20	Adam	0.5		1.2506 0.5419

3. 프로젝트 진행과정

3. 전이학습 모델 - MobileNet V3 Small

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
150, 150, 1	데이터 증식	1e-4	20	Adam	0.5		1.1608 0.5569
150, 150, 1	데이터 증식	1e-3	30	Adam	0.5	DNN 추가	학습중단
150, 150, 1	HE	1e-4	20	Adam	0.5		1.2506 0.5419

MobileNet V3 Large

- ✓ 흑백처리 및 데이터 증식
- ✓ Adam 사용
- ✓ Fine Tuning, DNN x
- ✓ 최종 ACC : 약 62%

3. 프로젝트 진행과정

4. 전이학습 모델 - EfficientNet

2012년 이후 AlexNet 이후 기존 다른 CNN 모델들은 Depth를 증가시켜 성능을 향상시키는 방식으로 발전했지만,

EfficientNet은 다음 3가지 요소를 조화롭게 늘리는 방식으로 모델 구현

- ✓ Layer 개수(Depth)
- ✓ 필터 개수(Width)
- ✓ 입력 이미지 크기(Resolution)

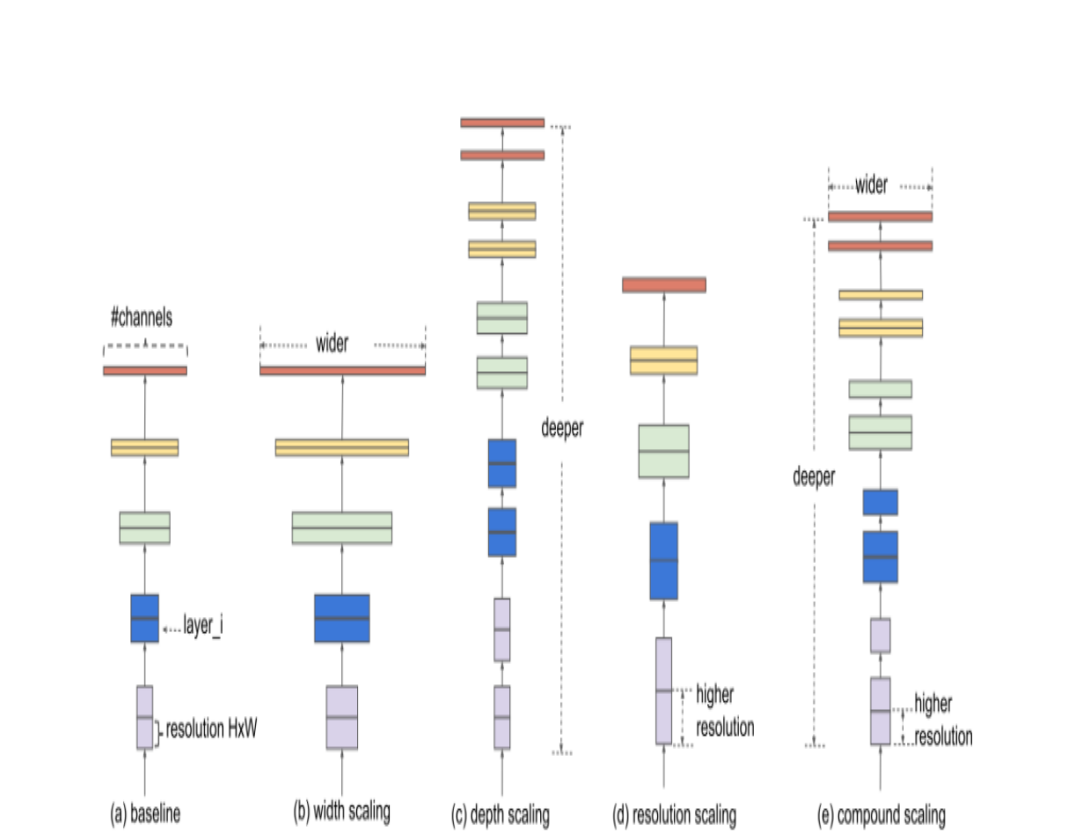


Figure 2. **Model Scaling.** (a) is a baseline network example; (b)-(d) are conventional scaling that only increases one dimension of network width, depth, or resolution. (e) is our proposed compound scaling method that uniformly scales all three dimensions with a fixed ratio.

3. 프로젝트 진행과정

4. 전이학습 모델 - EfficientNet

B0-B7로 갈수록 Number of Parameters가 증가하고, 정확도 또한 증가한다

EfficientNet B0-B7에 따라 최적화된 입력 이미지 크기 다르다

본 프로젝트에서는 150 X 150 크기로 데이터 전처리

→ B0 모델이 B6모델보다 좋은 결과 도출

Base model	resolution
EfficientNetB0	224
EfficientNetB1	240
EfficientNetB2	260
EfficientNetB3	300
EfficientNetB4	380
EfficientNetB5	456
EfficientNetB6	528
EfficientNetB7	600

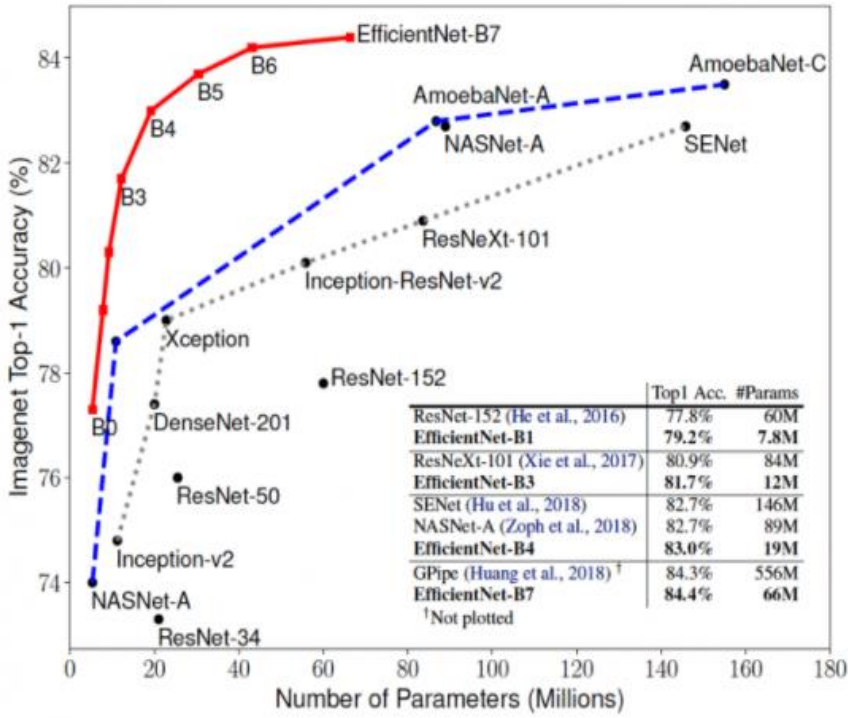


Figure 1. **Model Size vs. ImageNet Accuracy.** All numbers are for single-crop, single-model. Our EfficientNets significantly outperform other ConvNets. In particular, EfficientNet-B7 achieves new state-of-the-art 84.4% top-1 accuracy but being 8.4x smaller and 6.1x faster than GPipe. EfficientNet-B1 is 7.6x smaller and 5.7x faster than ResNet-152. Details are in Table 2 and 4.

3. 프로젝트 진행과정

4. 전이학습 모델 – EfficientNetB0

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
150, 150, 3	데이터 증식	1e-4	20	Adam	0.55	EfficientNetB6	2.3074 0.3924
150, 150, 3	데이터 증식	2e-4	20	Adam	0.3		1.6437 0.4687
150, 150, 3	데이터 증식	1e-4	20	Adam	0.55		1.4347 0.4754

3. 프로젝트 진행과정

4. 전이학습 모델 – EfficientNetB0

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
224, 224, 3	x	3e-4	20	Adam	0.5		1.7843 0.4316
150, 150, 3	x	2e-4	20	Adam	0.2		1.3976 0.5294
150, 150, 3	x	3e-4	20	Adam	0.2		1.5624 0.4952
150, 150, 3	x	3e-4	20	Adam	0.2		1.4774 0.5289
150, 150, 3	x	3e-4	20	RMSprop	0.2		1.3088 0.5381

- ✓ 흑백처리, RMSprop 사용
- ✓ 데이터 증식 x
- ✓ 최종 ACC : 약 54%

3. 프로젝트 진행과정

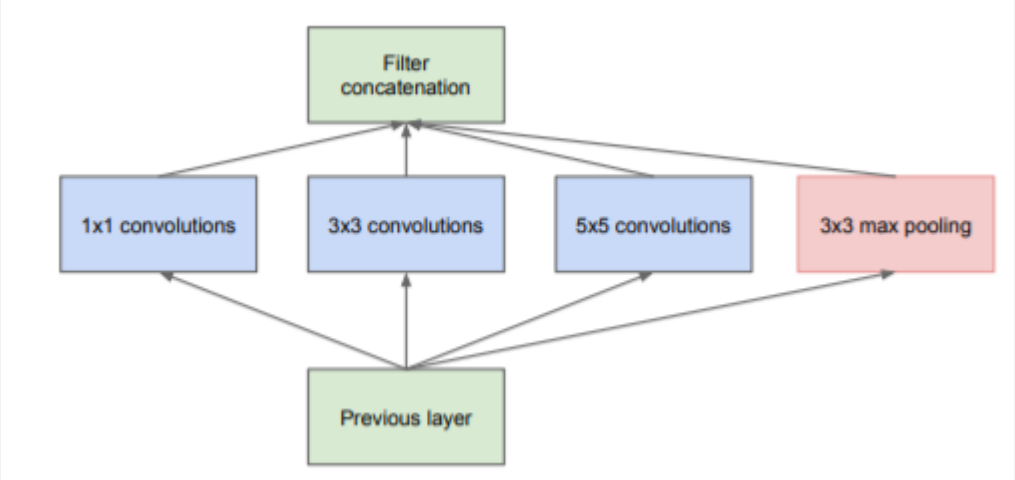
5. 전이학습 모델 - InceptionResNetV2

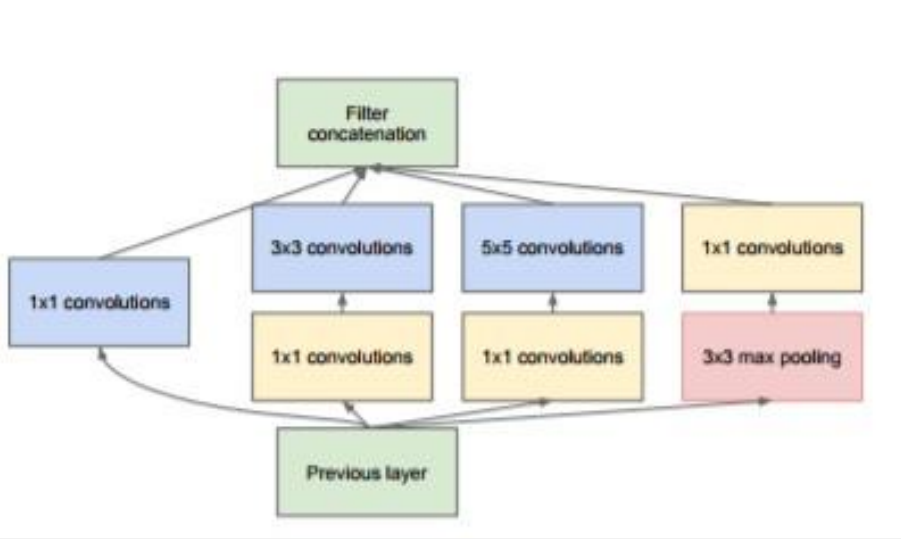
Inception

✓ feature를 효율적으로 추출하기 위해 1x1, 3x3, 5x5의 Convolution 연산을 각각 수행

✓ 입력과 출력의 H, W가 같도록 Pooling 연산에서 Padding을 추가

✓ 연산량을 조절하기 위해 1x1 Conv를 사용해 dimension reduction 수행





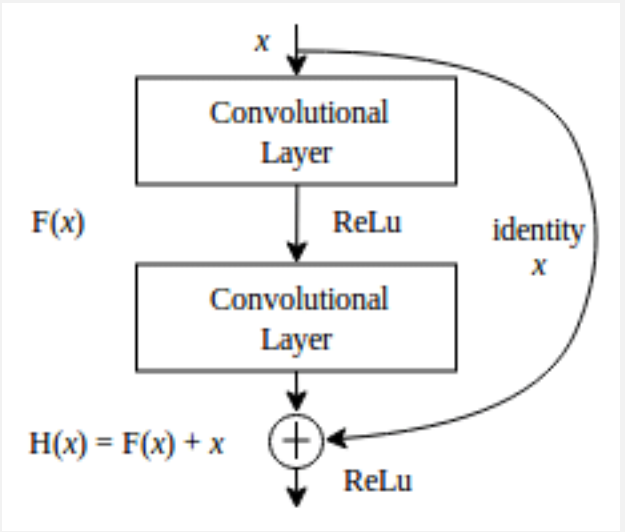
3. 프로젝트 진행과정

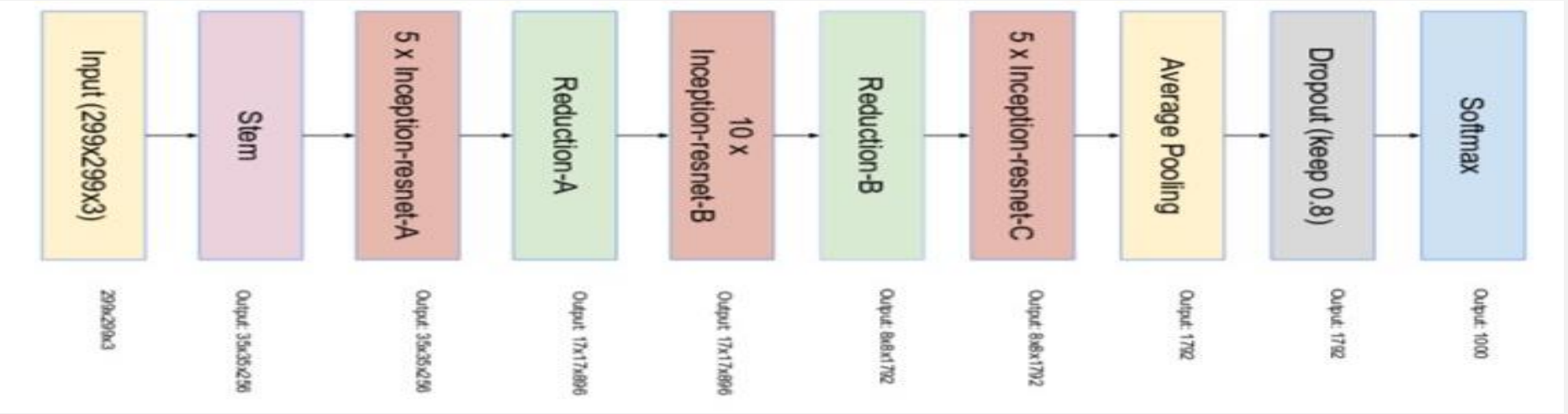
5. 전이학습 모델 - InceptionResNetV2

인셉션 네트워크에 Residual Connection을 적용한 네트워크

Residual Connection (잔차 연결) :

gradient vanishing 문제를 해결하기 위해 레이어1의 출력을 레이어2의 입력으로도 사용하면서 레이어 2의 출력과 연결





3. 프로젝트 진행과정

5. 전이학습 모델 - InceptionResNetV2

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
150, 150, 3	rescale	1e-4	20	Adam	0.5		1.332 0.5145
150, 150, 3	rescale	1e-4 1e-5	20	Adam	0.5	Finetuning	1.3436 0.5034
150, 150, 3	rescale	1e-4	20	Adam	0.25 0.5	DNN	1.4671 0.5077
150, 150, 3	rescale	1e-4 1e-5	20	Adam	0.25 0.5	Finetuning, DNN	1.2651 0.5357
150, 150, 3	rescale	1e-4 1e-5	30	Adam	0.25 0.5	Finetuning, DNN	1.2706 0.5448
150, 150, 3	데이터 증식	1e-4 1e-5	30	Adam	0.25 0.5	Finetuning, DNN	1.4484 0.4629

3. 프로젝트 진행과정

5. 전이학습 모델 - InceptionResNetV2

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
150, 150, 3	rescale	1e-4	20	RMSprop	0.5		1.424 0.5077
150, 150, 3	rescale	1e-4 1e-5	20	RMSprop	0.5	Finetuning	1.3278 0.513
150, 150, 3	rescale	1e-4	20	RMSprop	0.25 0.5	DNN	1.3951 0.5164
150, 150, 3	rescale	1e-4 1e-5	20	RMSprop	0.25 0.5	Finetuning, DNN	1.3951 0.5164
150, 150, 3	rescale	1e-4 1e-5	30	RMSprop	0.25 0.5	Finetuning, DNN	1.2741 0.5468
150, 150, 3	데이터 증식	1e-4 1e-5	30	RMSprop	0.25 0.5	Finetuning, DNN	1.4147 0.4682

- ✓ RMSprop 사용
- ✓ 데이터 증식 x
- ✓ Finetuning, DNN 사용
- ✓ 최종 ACC : 약 55%

Monday's Facial Expression Recognition

프로젝트 결과

4. 프로젝트 결과

모델 비교 최종 결과 - MobileNet V3 Large

형태	전처리	Learning rate	epochs	optimizer	dropout	비고	결과
150, 150, 1	데이터 증식	1e-4	20	Adam	0.5	-	1.2277 0.6162

결과

- ✓ 흑백처리 및 데이터 증식
- ✓ Adam 및 dropout 사용
- ✓ 최종 ACC : 약 62%

한계점

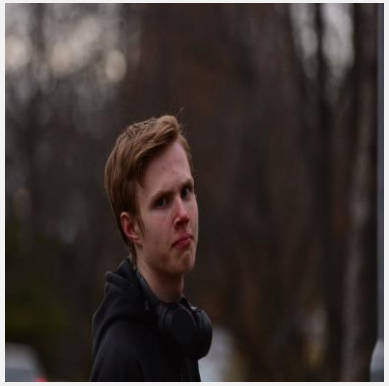
다양한 데이터 전처리 방식, Optimizer, DNN구성 변경하며 학습하였지만,

비슷한 결과를 보여 모델 수정에 판단의 어려움.

4. 프로젝트 결과

모델을 이용한 최종 이미지 분석

ANGRY : 0



DISGUST : 1

FEAR : 2

HAPPY : 3

NEUTAL : 4

SAD : 5

SURPRISE : 6

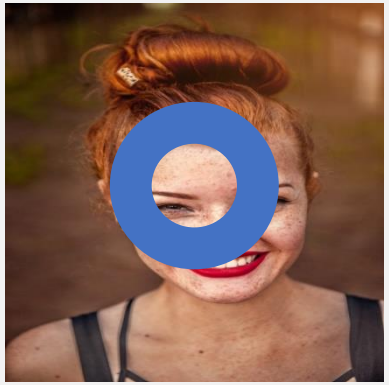
4. 프로젝트 결과

모델을 이용한 최종 이미지 분석

ANGRY : 0

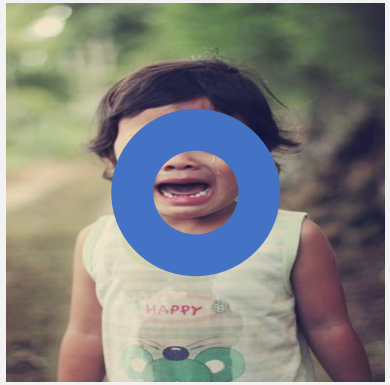
DISGUST : 4

FEAR : 0



HAPPY : 3

NEUTAL : 4



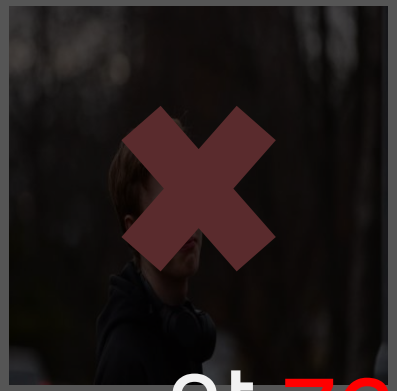
SAD : 5

SURPRISE : 6

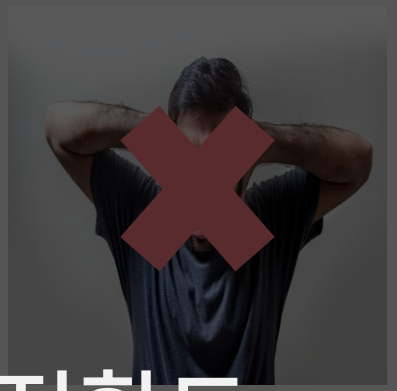
4. 프로젝트 결과

모델을 이용한 최종 이미지 분석

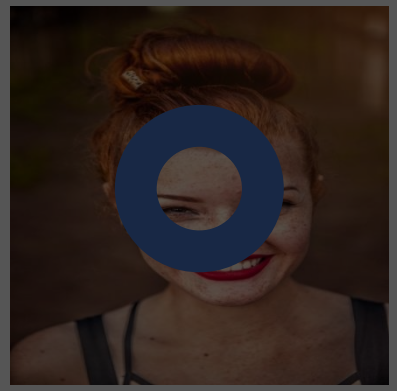
ANGRY : 0



DISGUST : 4

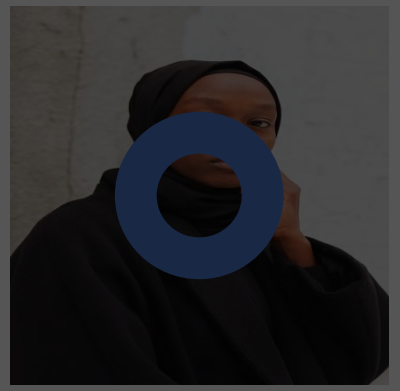


FEAR : 0

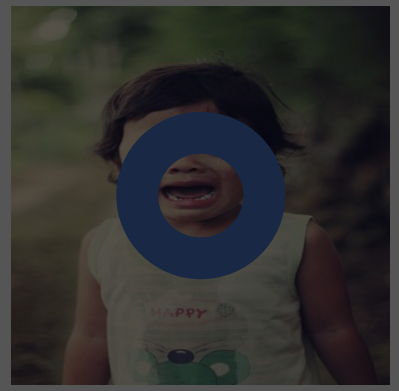


HAPPY : 3

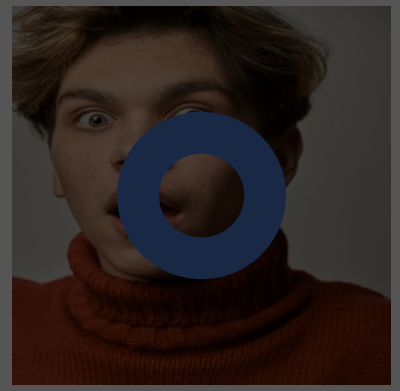
약 72% 정확도



NEUTAL : 4



SAD : 5



SURPRISE : 6

월요일

Monday's Facial Expression Recognition

한계 및 발전방향

5. 한계 및 발전방향

1. 한계

MLCNN 모델을 사용하는 논문을 찾았으나 해당 모델의 구현이 힘들어서 포기했음
컴퓨팅 파워의 문제로 팀원 4명이 AWS를 사용했음
특징 추출 방법(HE, HOG, LBP)을 시간이 없어서 최적의 변환 방법을 찾지 못한 것 같음

논문과 정확히 똑같은 모델 재현 어려워 논문에서 언급한 만큼의 accuracy값 얻지 못함

사람은 기계가 아니라서 표정이 개성이 있어 데이터의 한계가 있다. 감정인식을 위한 표정을 규격화한 전용 데이터의 필요성을 느꼈다.
MTCNN 등 새로운 개념들을 이해하고 코드를 구현하는데 생각보다 많은 시간이 할애되어 모델링을 하는 시간이 부족함을 느껴 아쉬웠다.

얼굴 표정을 분류해 놓은 데이터 자체가 사람이 분류한 것이어서 개인에 따라 분류하는 데에 오류가 발생할 수 밖에 없다. 실제로 우리 조에서 사용한 데이터는 대부분 서양 사람들의 사진이었는데 중간에 섞여있는 동양인 사진의 분류는 잘못되어 있는 것들이 많았다.

로컬 컴퓨터의 성능이 좋지 않아 AWS를 나누어 쓴 점이 아쉬웠고, 기존 사용했던 데이터의 분류가 정확하게 되어 있지 않아 어려움이 많았다.

5. 한계 및 발전방향

2. 발전방향

MLCNN 모델에 도전하기

4차원 ndarray 처리 방법을 이해해서 stochastic pooling 구현해보고 싶다

모델 마케팅이나 의학 분야로 더 발전시키고 싶다. 사람들의 표정을 통해 감정을 읽어서 기분에 맞춰 상품들을 추천하거나 우울증과 같은 표정에 드러나는 질병들을 조기에 발견하는데 쓰이는 모델로 발전시키고 싶다.

얼굴 표정을 읽는 것은 범죄 수사나 협상, 인터뷰 등에서 어떠한 사람이 진실을 얘기하는지 거짓말을 하고 있는지를 판별하는 지표가 될 수 있다. 그러므로 사진을 분석하기 보다는 소프트웨어 개발을 통해 실시간으로 표정을 분석하는 기술이 필요하다. 감정 인식과 더불어 동공의 움직임 등을 실시간으로 분석하여 범죄자나 재판에서의 위증 등을 판별하는 데에 사용한다면 인간에게 유익한 기술이 될 것이라고 생각한다.

정적인 사진를 분석하는 모델을 발전시켜 동적인 영상을 분석하는 모델을 구현해보고 싶다고 느꼈다. 또한 파라미터를 좀 더 다양하게 변화시켜 더 좋은 성능을 가진 모델을 구현해 보고 싶다.

부록

[1] https://arxiv.org/abs/1905.11946
[2] https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet
[3] https://ai.googleblog.com/2019/05/efficientnet-improving-accuracy-and.html
[4] https://hoya012.github.io/blog/EfficientNet-review/
[5] https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_histograms/py_histogram_equalization/py_histogram_equalization.html
[6] https://ocr.space/ocrapi
[7] https://pysource.com/2019/10/14/ocr-text-recognition-with-python-and-api-ocr-space
[8] https://hwangtoemat.github.io/paper-review/2020-03-28-MTCNN-%EB%82%B4%EC%9A%A9/
[9]https://jkisaaclee.kro.kr/keras/facenet/deep%20learning/computer%20vision/2019/10/01/how_to_develop_a_face_recognition_system_using_facenet_in_keras_ko/
[10] https://ichi.pro/ko/mtcnneul-sayonghan-eolgul-gamji-sogdo-e-chojeom-eul-majchun-eolgul-chuchul-gaideu-13663800405321
[11] https://yeomko.tistory.com/16
[12] https://soobarkbar.tistory.com/62
[13] https://seongkyun.github.io/papers/2019/12/03/mbv3/
[14] https://minimin2.tistory.com/43
[15] https://minimin2.tistory.com/42
[16] https://eehoeskrap.tistory.com/431
[17] https://gaussian37.github.io/dl-concept-mobilenet_v2/
[18] Image based Static Facial Expression Recognition with Multiple Deep Network Learning(Zhiding Yu, Cha Zhang)
[19] Leveraging Recent Advances in Deep Learning for Audio-Visual Emotion Recognition(Liam Schoneveld, Alice Othmani, Hazem Abdelkawy)
[20] Convolutional Neural Networks for Facial Expression Recognition (Shima Alizadeh, Azar Fazel)

부록

[21] Raw data : https://www.kaggle.com/kreinisalex/telemoji
[22] angry : https://unsplash.com/photos/AjMXxHwxW_k
[23] disgust : https://unsplash.com/photos/zaK5x8InoaU
[24] fear : https://unsplash.com/photos/8dvyPDYa35Q
[25] happy : https://unsplash.com/photos/u3WmDyKGsrY
[26] neutral : https://unsplash.com/photos/2eC_-0yRrGU
[27] sad : https://unsplash.com/photos/H566W24FyL8
[28] surprise : https://unsplash.com/photos/_J2fMaieLSM
[29] 손흥민 : https://m.post.naver.com/viewer/postView.nhn?volumeNo=18181897&memberNo=45090529
[30] Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning
[31] http://datacrew.tech/inception-v4-2016/
[32] https://starter-a.tistory.com/6
[33] https://a292run.tistory.com/entry/A-Simple-Guide-to-the-Versions-of-the-Inception-Network-1

감사합니다

Monday's Facial Expression Recognition

2021.04.26

유영현, 안준언, 김화연, 서민하, 김기범