기계학습 기말 과제 결과 보고서

컴퓨터공학부
201901208
김현동

1. 프로젝트 개요

- 목표 : 데이터 전처리 및 사용 모델 성능을 비교하고 성능 차이의 원인을 분석할 수 있다.
- 데이터 : 기본적인 전처리 과정은 동일. 증강의 유무에 따른 성능 차이 비교
- 모델 : 직접 정의한 CNN, CNN 기반 EfficientNet 사용 및 모델 분석

2. 문제 정의

- 문제 설명 : 계절을 맞추는 다중 클래스 분류
- 데이터 셋 : Weather Image Recognition + Multi-class Weather Dataset
  - 두 데이터 병합
  - 일반적으로 불리는 날씨(눈,비,안개,구름 etc..)등을 분류하는 것을 목표. 일부 label 제거['Sunrise','dew','frost','glaze','hail','rainbow','rime']
  - 훈련, 검증, 테스트 데이터 분리. 비율(0.8 : 0.08 : 0.12)

3. 데이터 전처리

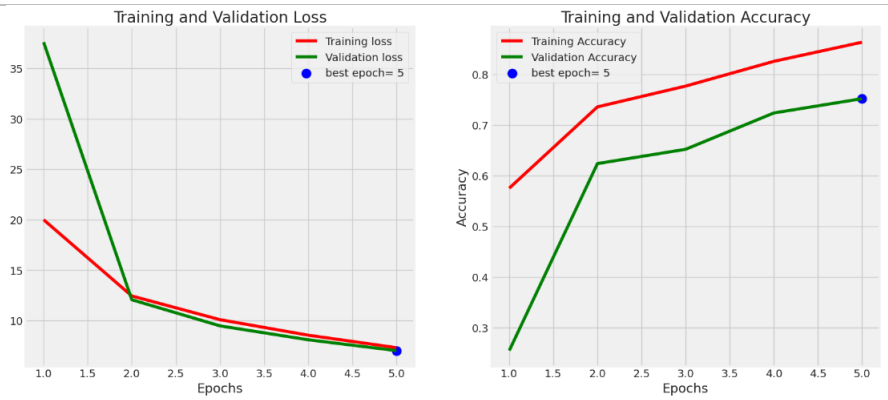
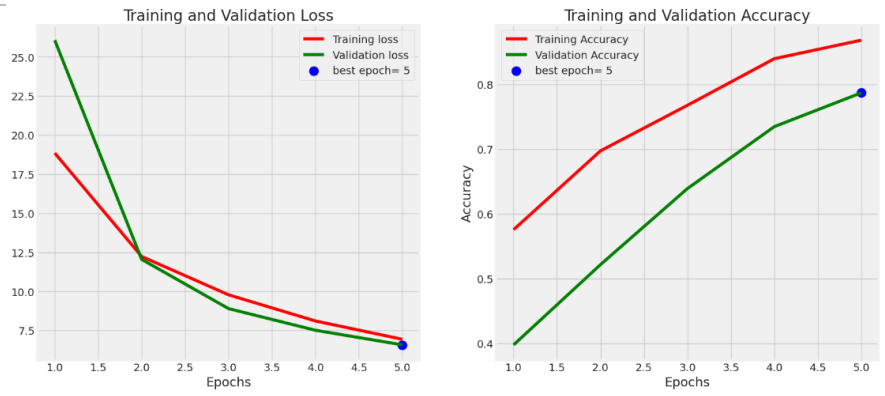
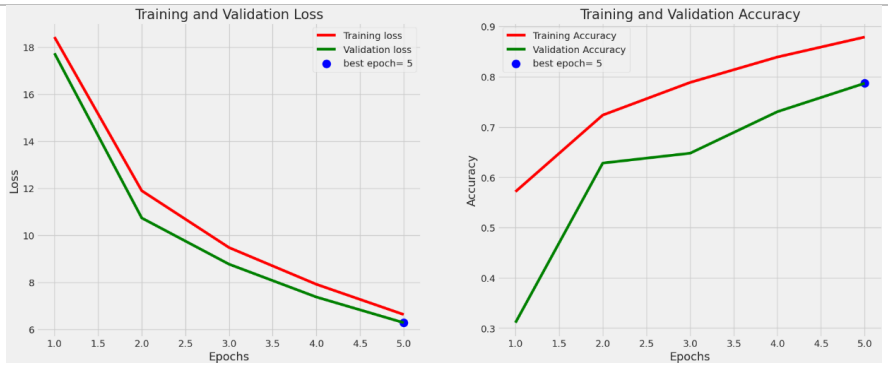
- **공통되는 전처리 :**
  1) 이미지 크기 (224, 224)로 일치
  2) 클래스 라벨을 원-핫 인코딩 형태로 변경
  3) 훈련 검증 데이터 에포크마다 섞기
  4) color mode 'rgb' 설정

- **데이터 증강 (불균형 해소)**
  - 위 사진을 통해 같은 라벨을 가지더라도 다양한 밝기, 그림자 등 다양한 특성을 가지고 있음을 알 수 있다. 따라서 데이터 증강은 flip, rotation, shift, zoom 기법을 사용했다. 또한 split 후 증강을 진행했다. 테스트 데이터는 모델이 실전에서 어떻게 동작하는지 평가하기 위한 역할을 하기 때문에, 훈련 데이터와 테스트 데이터 간의 일관성을 유지하는 것이 중요하다고 여겨지기 때문이다. 이러한 이유로 증강은 test 데이터를 제외하고 진행했다. 또한 증강을 통해 과적합 방지, 일반화 성능

향상을 기대할 수 있다.

Train 데이터 불균형 확인	Max_sampling 개수 500 설정 후 불균형 해소
labels fogsmog 690 rain 589 sandstorm 548 snow 489 lightning 313 Cloudy 243 Shine 196	Original Number of classes in dataframe: 7 [500, 500, 500, 489, 313, 243, 196] Found 313 validated image filenames. Found 196 validated image filenames. Found 489 validated image filenames. Found 243 validated image filenames. Total Augmented images created= 759 [500, 500, 500, 500, 500, 500, 500]

*증강 기법에 따른 learning rate with CNN

증강 방법	train loss & Validation loss
증강 x	
증강 (Shift , Zoom)	
증강 (Shift , Zoom Flip , Rotation)	

해석 : 다양한 증강기법을 사용할수록 loss curve의 시작점이 낮아지고 마지막 epoch에서도 더

낮은 loss를 보인다. 감소 기울기도 더 큰 것으로 보아 Epoch 수를 늘려서 학습을 진행했을 때, loss가 더 낮아질 가능성이 높다.
Accuracy curve에서는 세 증강 기법의 차이가 크지는 않지만 증강을 많이 할 수록 정확도가 높아진다는 점도 확인 할 수 있다. 제가 사용한 날씨 데이터는 증강 기법이 성능 향상에 도움이 되었고 일반화에도 도움이 되므로 사용하는 것이 옳다고 판단했다.
5 epoch까지는 세 기법 모두 train loss, validation loss가 동시에 감소하고 accuracy 또한 증가하고 있어서 추가 학습을 진행해도 될 것이라 판단된다.

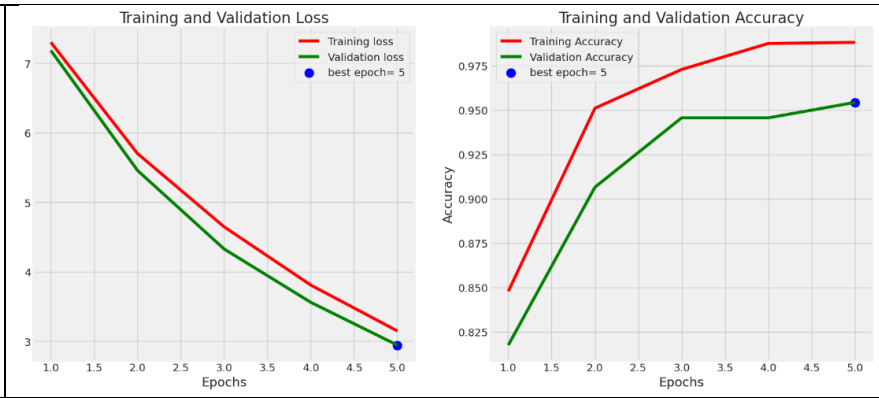
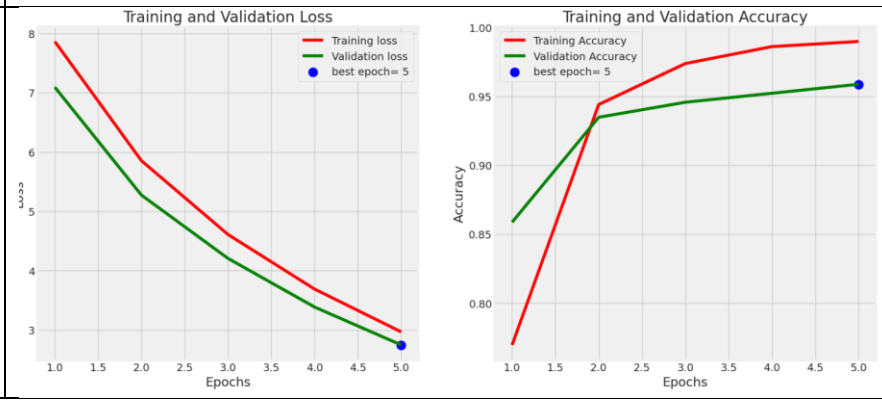
4. 모델

CNN

- 구조 : convolution layer 3개, dense layer 1개, 출력 layer 1
- 특징 :
    - 활성화 함수 (ReLU) : 비선형성을 추가하고, 계산 효율성이 증가한다.
    - MaxPooling : 가장 중요한 특징을 강조하고 작은 변화에 덜 민감하다는 특징이 있다.
    - 배치 정규화 : 각 배치에 대한 정규화를 통한 학습 속도 증가, 과적합 방지, 초기 가중치 민감성 감소를 기대할 수 있다.
    - DropOut layer : 과적합, 특정 뉴런에 의존하는 것을 방지하고 일반화 성능 향상시킨다.
    - Flatten layer : Convolution layer 출력을 1차원 벡터로 변환하여 Dense layer 입력으로 사용할 수 있도록 변환한다.
    - Softmax : 다중 클래스 분류 문제를 해결하기 위한 확률 분포를 제공한다. Output layer에서 최종 날씨 예측을 위해 사용된다.

EfficientNet

- 특징 : Compound Scaling. 네트워크의 깊이(depth), 필터 수(width), 이미지 해상도(resolution)를 최적으로 조합하여 모델 성능을 극대화한다.
- 효율성 : 다른 모델에 비해 적은 파라미터, 연산으로도 높은 성능을 보인다. 이미지의 특징을 잘 파악하고 있는 것을 의미한다.

증강 x	
증강 (Shift , Zoom Flip , Rotation)	

해석 : CNN 모델과 비교했을 때, loss, accuracy의 시작지점과 마무리 지점에서 성능차이가 많이 난다는 것을 알 수 있다. 그리고 이러한 차이가 나는 이유를 다음과 같은 특징 때문이라고 유추할 수 있다.

- **네트워크 아키텍처의 효율성:**

  네트워크의 깊이, 너비, 해상도를 조정하는 방식으로 네트워크의 복잡성을 조절한다. 이러한 효율적인 구조는 특정 문제에 맞는 학습 가능한 파라미터를 생성할 수 있도록 도와준다.

- **Feature Extractor의 풍부성:**

  특정한 이미지 분류나 감지 작업에 더 효과적인 특징을 추출하는 데 중점을 둔 효율적인 네트워크 구조를 가지고 있다. 이는 더 적은 파라미터로도 더 많은 정보를 효과적으로 캡처할 수 있게 해준다.

- **전이 학습의 효과:**

  사전 학습된 가중치와 함께 제공되는 경우가 많다. 따라서, 더 적은 파라미터로 시작해도 높은 성능을 달성할 수 있는 전이 학습의 이점을 얻을 수 있다.

- **Regularization 및 정규화 기법:**
- EfficientNet 구조는 규제와 정규화 기법을 통합적으로 적용하여 과적합을 방지하고, 효과적인 학습을 가능하게 한다. 따라서, 훈련 데이터의 과적합을 피하면서 높은 성능을 달성할 수 있다.

*모델 구조

CNN

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 222, 222, 32)      896

max_pooling2d (MaxPooling2D  (None, 111, 111, 32)      0
)

batch_normalization (BatchN  (None, 111, 111, 32)      128
ormalization)

conv2d_1 (Conv2D)            (None, 109, 109, 64)      18496

max_pooling2d_1 (MaxPooling  (None, 54, 54, 64)        0
2D)

batch_normalization_1 (Batc  (None, 54, 54, 64)        256
hNormalization)

conv2d_2 (Conv2D)            (None, 52, 52, 128)       73856

max_pooling2d_2 (MaxPooling  (None, 26, 26, 128)       0
2D)

batch_normalization_2 (Batc  (None, 26, 26, 128)       512
hNormalization)

flatten (Flatten)            (None, 86528)             0

dense (Dense)                (None, 256)               22151424

dropout (Dropout)            (None, 256)               0

dense_1 (Dense)              (None, 7)                 1799

=================================================================
Total params: 22,247,367
Trainable params: 22,246,919
Non-trainable params: 448
_________________________________________________________________
```

EfficientNet

```
Layer (type)                 Output Shape              Param #
=================================================================
efficientnetb3 (Functional)  (None, 1536)              10783535

batch_normalization (BatchN  (None, 1536)              6144
ormalization)

dense (Dense)                (None, 256)               393472

dropout (Dropout)            (None, 256)               0

dense_1 (Dense)              (None, 7)                 1799

=================================================================
Total params: 11,184,950
Trainable params: 11,094,575
Non-trainable params: 90,375
_________________________________________________________________
```

*테스트 결과 (Accuracy)

	CNN	EfficientNet
증강 x	0.75	0.94
증강 (Shift , Zoom)	0.76	NuLL
증강 (Shift , Zoom Flip , Rotation)	0.76	0.93

테스트 결과 CNN에서 조금이나마 유의미한 결과가 있었지만 EfficientNet에서는 오히려 정확도가 감소했다. EfficientNet은 특정 범위의 이미지 해상도에 최적화 되어있는데 증강기법의 노이즈가 학습을 방해할 가능성이 있을 수 있다.

느낀점 : 컴퓨터 비전은 처음 다뤄봐서 간단한 증강 기법과 CNN 모델 구현, 사전학습 모델 사용하는 것에 초점을 두고 진행했다. 사전학습된 모델은 이미 최적화되어 있어서 성능이 좋은 것은 어쩔 수 없으나 직접 한 증강 기법에서 유의미한 결과를 얻은 것 같지 않아서 아쉬운 점이 있다. 이 텀프로젝트에 이어서 데이터, 모델 특성을 고려한 전처리에 대해 더욱 고민하고, 모델 설계도 집중해서 해봐야 겠다는 생각이 들었다.

참고자료 :

날씨 데이터 특징 추출

https://patents.google.com/patent/KR20190095714A/ko

kaggle

https://www.kaggle.com/code/gpiosenka/weather-f1-test-set-score-92

https://www.kaggle.com/code/hadeerismail/weather-image-recognition-acc-94

CNN

https://sdc-james.gitbook.io/onebook/4.-and/5.4.-tensorflow/5.4.2.-cnn-convolutional-neural-network

EfficientNet

https://blog.kubwa.co.kr/%EB%85%BC%EB%AC%B8%EB%A6%AC%EB%B7%B0-efficientnet-rethinking-model-scaling-for-convolutional-neural-networks-2019-%EA%B0%84%EB%8B%A8%ED%95%9C-%EC%8B%A4%EC%8A%B5%EC%BD%94%EB%93%9C-cbe0e9963ffc