AIB_13_Section4_Project

흉부 X - ray 이미지를 통한
코로나 및 폐질환 판별 딥러닝 모델 개발

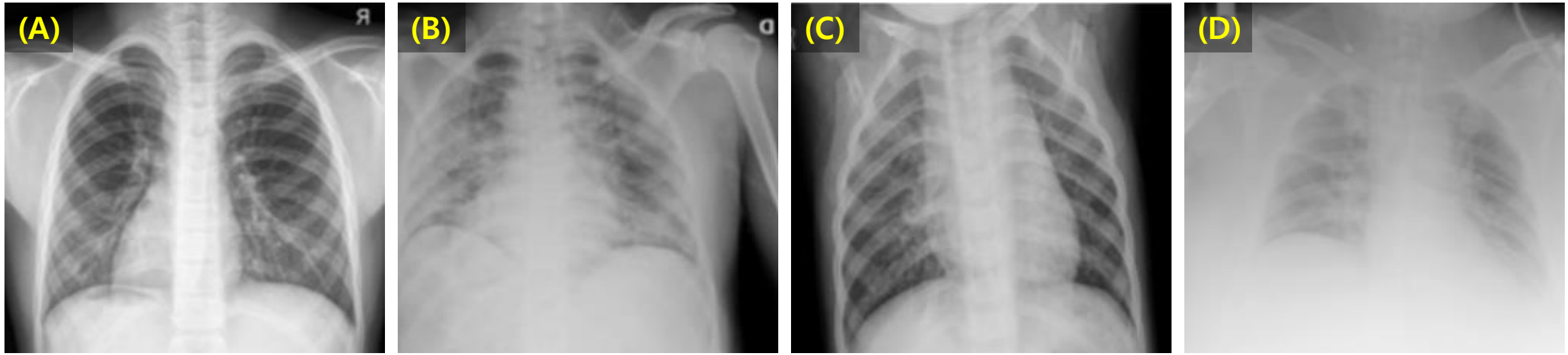
AI_13_김강호

목차

목차

1. 데이터 소개 & 가설

- **데이터 종류** : 4가지 상태의 흉부 X-ray 이미지



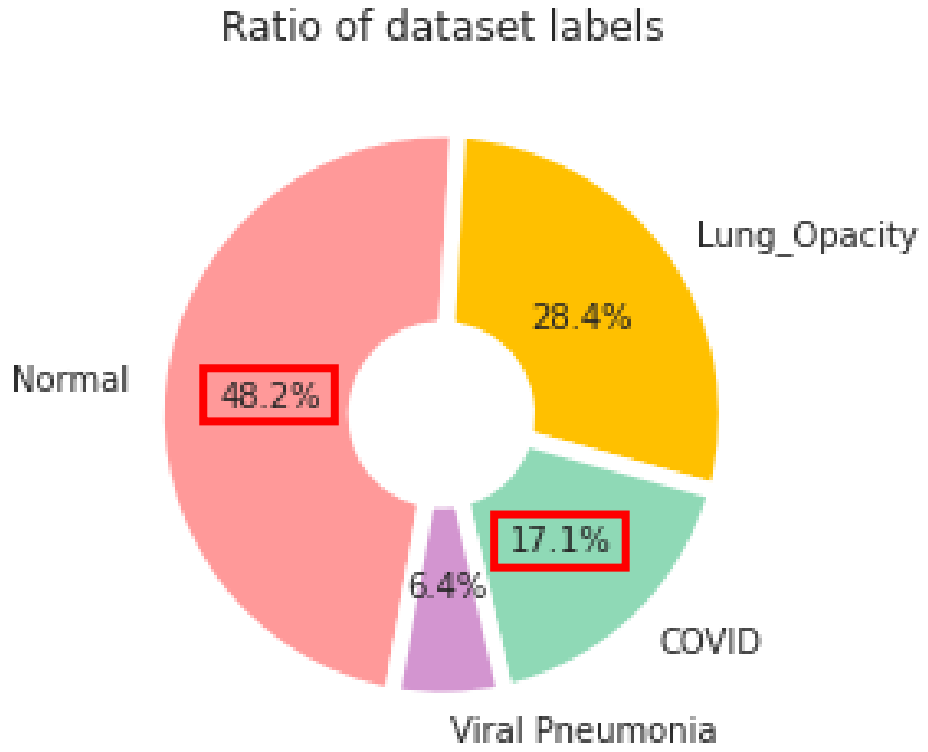
(A). 정상상태 – Normal
(B). 코로나 감염상태 - COVID
(C). 바이러스성 폐렴 - Viral pneumonia
(D). 폐 음영 - Lung Opacity

1. 데이터 소개 & 가설

- **데이터 종류** : 4가지 상태의 흉부 X-ray 이미지

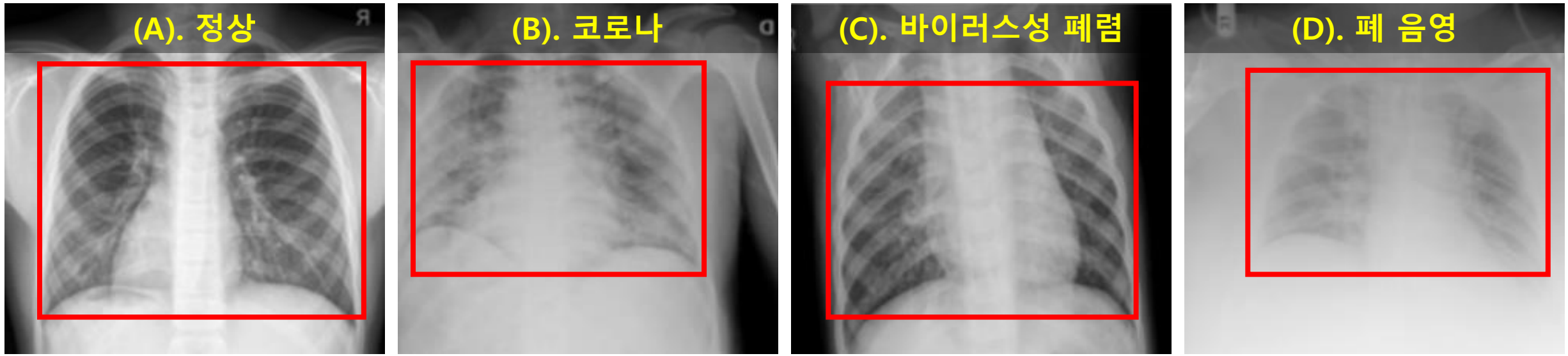
✓ X-ray 이미지의 총 수 : 21165 개

상태	데이터 개수(개)
정상상태(Normal)	10192
코로나 감염상태(COVID)	3616
바이러스성 폐렴(Viral pneumonia)	1345
폐 음영(Lung Opacity)	6012



1. 데이터 소개 & 가설

- 이미지 분석



✓ 정상상태(A)를 제외한 **(B) – (D)**의 X-ray 상태는 **폐가 뿌옇게** 나옴

✓ **코로나**인지 **폐질환**인지 **구별하기 어려운 경우** 존재

✓ **숙달된 의료진**이 아니고서는 구별이 어려울 수 있음

1. 데이터 소개 & 가설

- 가설

이미지 분석

- ✓ 정상상태(A)를 제외한 (B) – (D)의 X-ray 상태는 폐가 뿌옇게 나옴
- ✓ 코로나인지 폐질환인지 구별하기 어려운 경우 존재
- ✓ 숙달된 의료진이 아니고서는 구별이 어려울 수 있음

가설

딥러닝 모델이 4가지 클래스에 대한 X-ray 이미지 패턴을 학습해서

환자 상태를 잘 분류할 수 있다!!

목차

2. 프로젝트 목표 및 기대효과

- **목표**
  - ✓ 흉부 x-ray 이미지의 **코로나 및 폐질환 진단**용 **딥러닝 모델** 개발

2. 프로젝트 목표 및 기대효과

- 딥러닝 모델의 기대효과

  ✓ 의료진의 진단보조 → 휴먼 에러 방지, 판독시간 감소

  ✓ 환자의 입장 : 조기 진단 → 치료결과 ↑

  ✓ 의료진의 입장 : 인력 낭비 감소

목차

3. 모델 개발과정

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

X-ray 이미지의 데이터 프레임 구축

↓

데이터프레임 전처리

↓

이미지 전처리

- X-ray 이미지의 데이터프레임(df)
  - ✓ 구성목적 : 이미지와 해당 이미지의 X-ray 상태(레이블)을 쉽게 불러오기 위함
  - ✓ os 라이브러리로 이미지의 경로 및 이미지의 폴더이름 추출, 저장
  - ✓ file_pathes : 이미지의 경로, labels : 이미지의 실제 X-ray 상태

	file_pathes	labels
0	/content/drive/MyDrive/Colab Notebooks/Section...	COVID
1	/content/drive/MyDrive/Colab Notebooks/Section...	COVID
2	/content/drive/MyDrive/Colab Notebooks/Section...	COVID
3	/content/drive/MyDrive/Colab Notebooks/Section...	COVID
4	/content/drive/MyDrive/Colab Notebooks/Section...	COVID

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

X-ray 이미지의 데이터 프레임 구축

데이터프레임 전처리

이미지 전처리

- 데이터프레임(df) 분할
  - ✓ **모델학습 및 평가**를 위해 **train_df, valid_df, test_df**로 분할
  - ✓ 데이터프레임 분할을 위해 **train_test_split** 사용

```
train_df, dummy_df = train_test_split(df, train_size = 0.9, shuffle = True,
                                      random_state = 123)
valid_df, test_df = train_test_split(dummy_df, train_size = 0.5,
                                     shuffle = True, random_state = 123)
```

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

X-ray 이미지의 데이터 프레임 구축

↓

데이터프레임 전처리

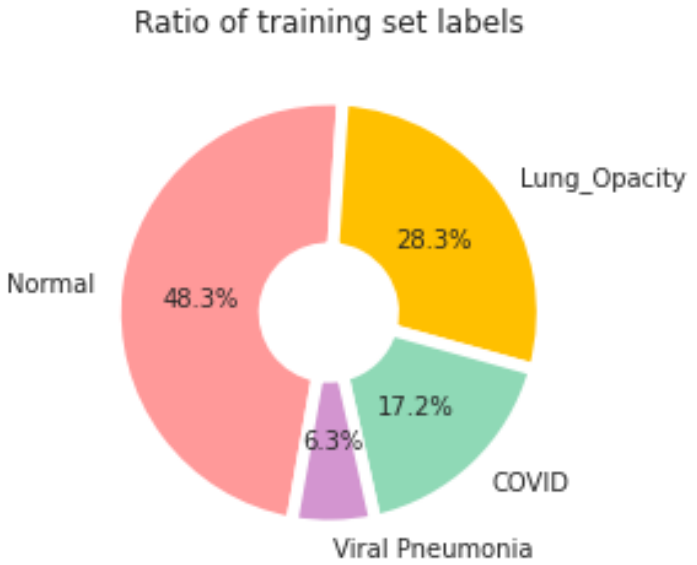
↓

이미지 전처리

- Train_df의 Undersampling
  - ✓ **불균형 데이터**를 그대로 **학습** : 학습모델의 **낮은 성능**을 보임

Train_df 의 레이블 별 이미지 수

```
Normal            9194
Lung_Opacity      5385
COVID             3275
Viral Pneumonia   1194
Name: labels, dtype: int64
```



3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

X-ray 이미지의 데이터 프레임 구축

↓

데이터프레임 전처리

↓

이미지 전처리

- Train_df의 Undersampling
  - ✓ 불균형데이터 처리 : Oversampling, Undersampling
  - ✓ 학습시간 절약을 위해 Undersampling 사용하여 데이터프레임의 레이블 별 정보를 balanced하게 통일 – sample 함수, for, if 문을 사용

Undersampling 전 레이블 counts

```
Normal            9194
Lung_Opacity      5385
COVID             3275
Viral Pneumonia   1194
Name: labels, dtype: int64
```

Undersampling 후 레이블 counts

```
Viral Pneumonia   1194
Normal            1194
COVID             1194
Lung_Opacity      1194
Name: labels, dtype: int64
```

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

X-ray 이미지의 데이터 프레임 구축

↓

데이터프레임 전처리

↓

이미지 전처리

- **ImageDataGenerator**
  - ✓ Data Augmentation(데이터 증강), 데이터 파이프라인 구축
- **Flow_from_dataframe**
  - ✓ 데이터프레임에서 ImageDataGenerator로 데이터를 원활하게 전송

```
trgen = ImageDataGenerator(preprocessing_function = scalar,
                           horizontal_flip = True)
tvgen = ImageDataGenerator(preprocessing_function = scalar)

train_gen = trgen.flow_from_dataframe(train_df, x_col = 'file_pathes', ~)

test_gen = tvgen.flow_from_dataframe(test_df, x_col = 'file_pathes', ~)

valid_gen = tvgen.flow_from_dataframe(valid_df, x_col = 'file_pathes', ~)
```

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

사전학습 모델을 통한 모델구성

Callback 함수 구성

모델학습 진행

- 사전학습 모델 - EfficientNetB1 사용
  - ✓ 이유 : X-ray 이미지 분류 분야에서 다른 사전학습 모델과 비교할 때, 성능이 좋고 참고자료가 많았음

Table 4. Classification results using Strategy II.

Deep learning Models	Evaluation Parameters			
	Accuracy	Precision	Sensitivity	F1 Score
EfficientNetB1	**96.13%**	97.25%	96.50%	97.50%
NasNetMobile	94.81%	95.50%	95%	95.25%
MobileNetV2	93.96%	94.50%	95%	94.50%

참고논문 :
1. Khan, E.; Rehman, M.Z.U.; Ahmed, F.; Alfouzan, F.A.; Alzahrani, N.M.; Ahmad, J. Chest X-ray Classification for the Detection of COVID-19 Using Deep Learning Techniques. *Sensors* 2022, *22*, 1211. https://doi.org/10.3390/s22031211
2. Ravi, V., Acharya, V. & Alazab, M. A multichannel EfficientNet deep learning-based stacking ensemble approach for lung disease detection using chest X-ray images. *Cluster Comput* (2022). https://doi.org/10.1007/s10586-022-03664-6

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

사전학습 모델을 통한 모델구성

Callback 함수 구성

모델학습 진행

- 학습모델 구성

Base_model : EfficientNetB1

Batchnormalization - 과적합, gradient vanishing 문제 해결

Regularizer 적용(가중치 감소) – 과적합 예방

Dropout – 과적합 예방

Output : activation = 'softmax'

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

사전학습 모델을 통한 모델구성

Callback 함수 구성

모델학습 진행

- 학습모델 구성

Base_model : EfficientNetB1

Batchnormalization - 과적합, gradient vanishing 문제 해결

Regularizer 적용(가중치 감소) – 과적합 예방

Dropout – 과적합 예방

Output : activation = ‘softmax’

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

사전학습 모델을 통한 모델구성

Callback 함수 구성

모델학습 진행

- 학습모델 구성

Base_model : EfficientNetB1

Batchnormalization - 과적합, gradient vanishing 문제 해결

Regularizer 적용(가중치 감소) – 과적합 예방

Dropout – 과적합 예방

Output : activation = ‘softmax’

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

사전학습 모델을 통한 모델구성

Callback 함수 구성

모델학습 진행

- 학습모델 구성

Base_model : EfficientNetB1

Batchnormalization - 과적합, gradient vanishing 문제 해결

Regularizer 적용(가중치 감소) – 과적합 예방

Dropout – 과적합 예방

Output : activation = ‘softmax’

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

사전학습 모델을 통한 모델구성

Callback 함수 구성

모델학습 진행

- 학습모델 구성

Base_model : EfficientNetB1

Batchnormalization - 과적합, gradient vanishing 문제 해결

Regularizer 적용(가중치 감소) – 과적합 예방

Dropout – 과적합 예방

Output : activation = 'softmax'

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

사전학습 모델을 통한 모델구성

Callback 함수 구성

모델학습 진행

- 학습모델 구성

Base_model : EfficientNetB1

Batchnormalization - 과적합, gradient vanishing 문제 해결

Regularizer 적용(가중치 감소) – 과적합 예방

Dropout – 과적합 예방

Output : activation = 'softmax'

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

사전학습 모델을 통한 모델구성

↓

Callback 함수 구성

↓

모델학습 진행

- **Callback 함수**
  - ✓ 딥러닝 모델의 **학습을 효율적으로 수행**하기 위해 구성된 함수
  - ✓ 딥러닝 모델이 **실행의 분기점**(train 시작과 끝, train batch의 시작과 끝, train epoch의 시작과 끝)**마다 모델의 학습을 효율적으로 수행하도록** 하는 **여러 기능들**로 구성.
    - ➢ 성능 개선 여지 X -> 학습중단
    - ➢ 성능 개선 여지 X -> Learning Rate를 조절해 모델의 개선 유도
    - ➢ 특정 Epochs 마다 학습을 계속할 것인지 여부를 묻는 기능

```
class LRA(keras.callbacks.Callback):
    def __init__(self, model, base_model, patience, stop_patience, threshold, factor, dwell, batches, initial_epoch, epochs, ask_epoch):
        super(LRA, self).__init__()
        self.model = model
        self.base_model = base_model
        self.patience = patience # 학습률이 조정되기 전에 개선되지 않은 Epoch의 수를 지정한다.
        self.stop_patience = stop_patience # 훈련을 중지하기 위해 개선 없이 얼마나 lr을 조정할 것인지 지정
```

이하 생략

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

사전학습 모델을 통한 모델구성

Callback 함수 구성

모델학습 진행

- 모델학습 진행
  - ✓콜백함수를 객체화 하고 모델학습을 진행

```
epochs =40
patience= 1 # 모니터링 되는 값이 개선되지 않을 경우 학습률을 조절하기 위해 기다릴 epoch수
stop_patience =3 # 모니터링된 값이 개선되지 않는 경우 훈련을 중지하기 전에 기다릴 에포크 수
threshold=.9 # if train accuracy is < threshhold adjust monitor accuracy, else monitor validation loss
factor=.5 # 학습률 감소 factor
dwell=True # experimental, if True and monitored metric does not improve on current epoch set  modelweights back to we

ask_epoch=2 # 훈련을 중단할 것인지 묻기 전에 실행할 에포크 수
batches=train_steps
callbacks=[LRA(model=model,base_model= base_model,patience=patience,stop_patience=stop_patience, threshold=threshold,
                    factor=factor,dwell=dwell, batches=batches,initial_epoch=0,epochs=epochs, ask_epoch=ask_epoch )]

history=model.fit(x=train_gen, epochs=epochs, verbose=0, callbacks=callbacks,  validation_data=valid_gen,
               validation_steps=None,  shuffle=False,  initial_epoch=0)
```

3. 모델 개발과정

1. 데이터 전처리 → 2. 모델학습 → 3. 모델평가

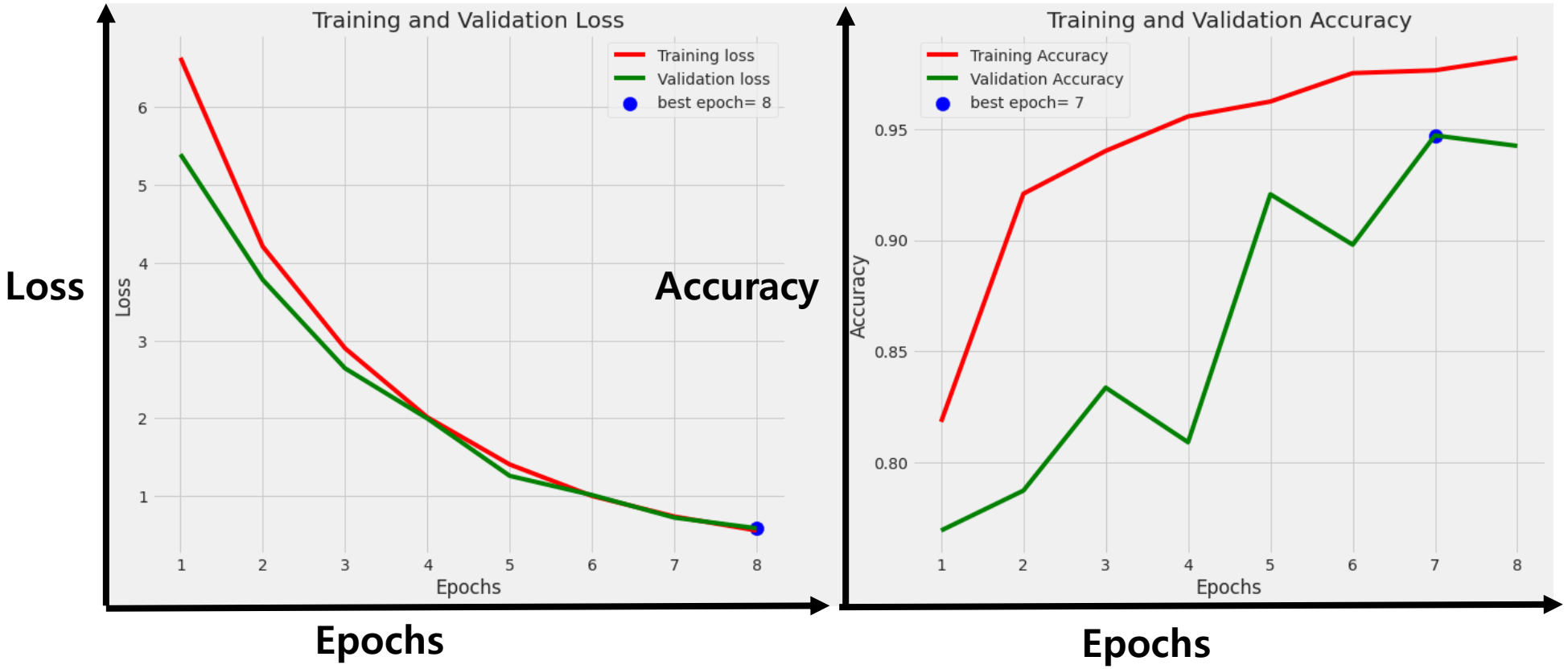
- 모델평가
  - ✓ 테스트 셋으로 학습된 모델을 평가한 뒤 모델 저장

```
tr_plot(history,0)
save_dir=r'./'
subject='covid'
acc=model.evaluate( test_gen, batch_size=test_batch_size, verbose=1, steps=test_steps, return_dict=False)[1]*100
msg=f'accuracy on the test set is {acc:5.2f} %'
print_in_color(msg, (0,255,0),(55,65,80))
save_id=str (model_name +  '-' + subject +'-'+ str(acc)[:str(acc).rfind('.')+3] + '.h5')
save_loc=os.path.join(save_dir, save_id)
model.save(save_loc)
```

목차

4. 학습모델 해석

- Epochs에 따른 Loss와 Accuracy의 변화



Training set

Validation set

@ Epoch 8 기준,

- Train acc : 98.19%
- Validation acc : 94.23%

4. 학습모델 해석

- Classification Report (Test set 예측결과)

Classification Report:

	precision	recall	f1-score	support
COVID	0.98	0.99	0.98	164
Lung_Opacity	0.89	0.94	0.91	317
Normal	0.96	0.92	0.94	501
Viral Pneumonia	0.93	1.00	0.96	77
accuracy			0.94	1059
macro avg	0.94	0.96	0.95	1059
weighted avg	0.94	0.94	0.94	1059

- ✓ Test set 정확도 : 94%
  - ➢ Chance Level : 25%
- ✓ 테스트 셋 : 데이터 개수의 불균형
- ✓ 불균형 데이터 >> f1-score중요
- ✓ 모든 범주의 F1 score : 0.9 이상
- ✓ 희귀 케이스에서 분류성능 양호

목차

5. 가설 검증

- **가설**
  - ✓ 딥러닝 모델이 **X-ray 이미지 패턴**을 **학습**해서 **환자 상태를 잘 분류**할 수 있다!!

- **결론**
  - ✓ 이번 테스트셋에 한정해서는, **잘 학습된 딥러닝 모델**이 X-ray 이미지의 **정상상태와 코로나, 폐 질환 상태** 등을 **잘 분류**할 수 있다.

목차

6. 프로젝트 회고

- **긍정적인 점**
  - ✓ **모델 성능**이 **잘** 나왔으며, 의료분야에서의 **실용적인 문제를 해결**한 것 같은 **느낌**이 들어 **뿌듯**했다.
  - ✓ **프로젝트를 진행**하면서, 기존에 알고 있던 **딥러닝 관련 지식**이 전체적으로 좀 더 **정리되는 느낌**을 받아서 기분이 좋았다.

    ➢ 딥러닝이 **여전히 어렵긴** 하지만 **흥미가 점점 더** 생기는 느낌이 든다.

6. 프로젝트 회고

- **아쉬운 점**

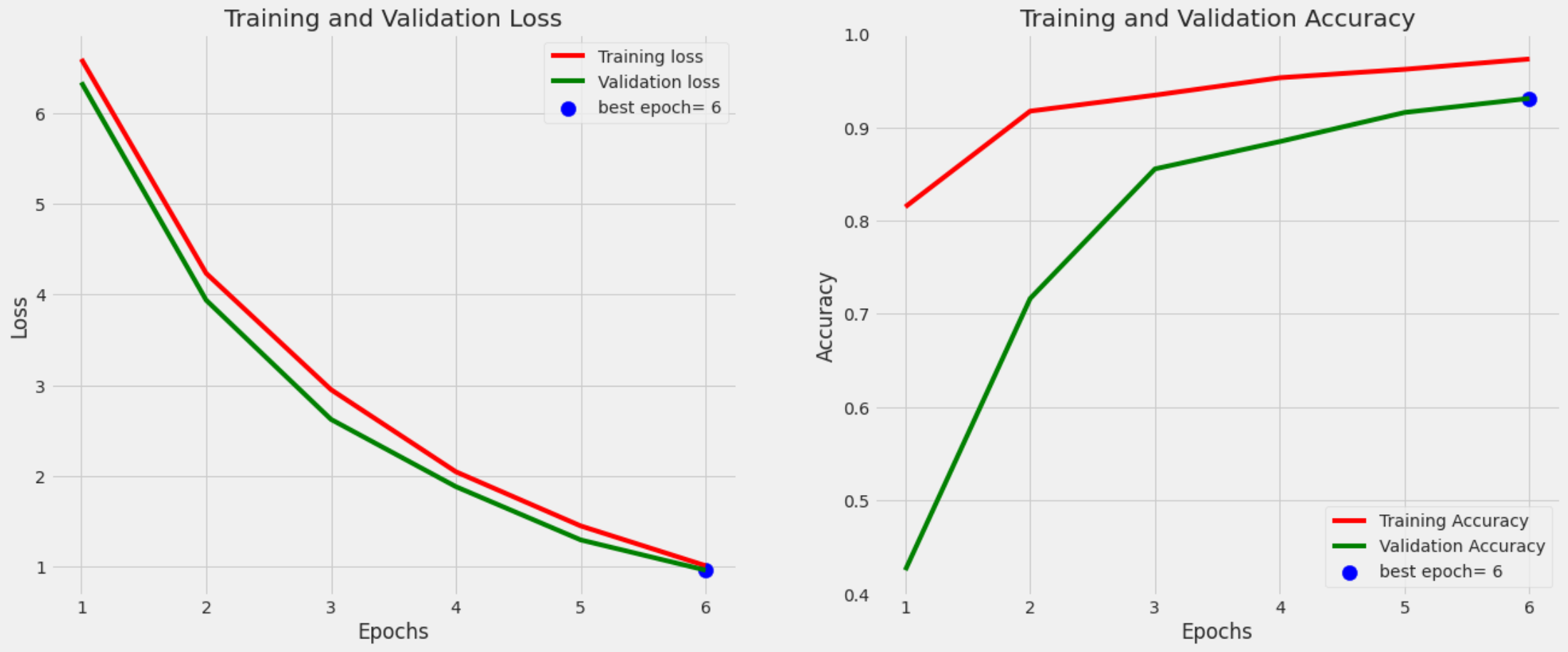
  ✓ **사전학습 모델과 다른 사람들의 코드를 상당수 인용**
  - **코드**에 대한 **이해**를 완전히 하지 못함
  - 내가 **직접 학습모델**을 **최적화** 할 기회가 적었음

  ✓ **데이터셋에 대한 아쉬운 점**
  - **데이터의 개수**가 그렇게 **많지 않았음**
  - **인종**이나 **나이, 성별** 등에 대한 **데이터**가 **없었음**

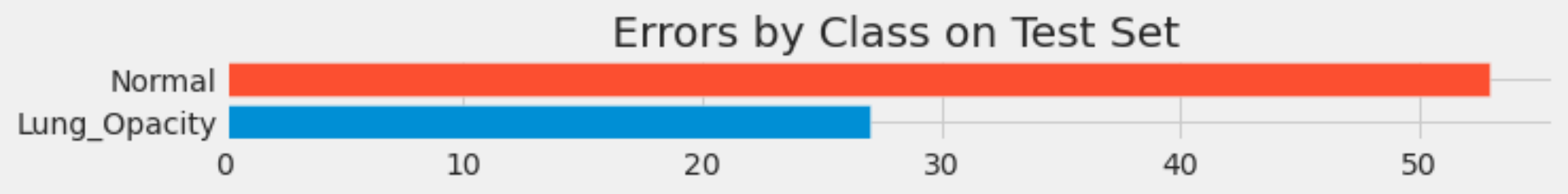
  ∴ 이번 딥러닝 학습모델의 일반화는 무리

부록
Test 1 결과

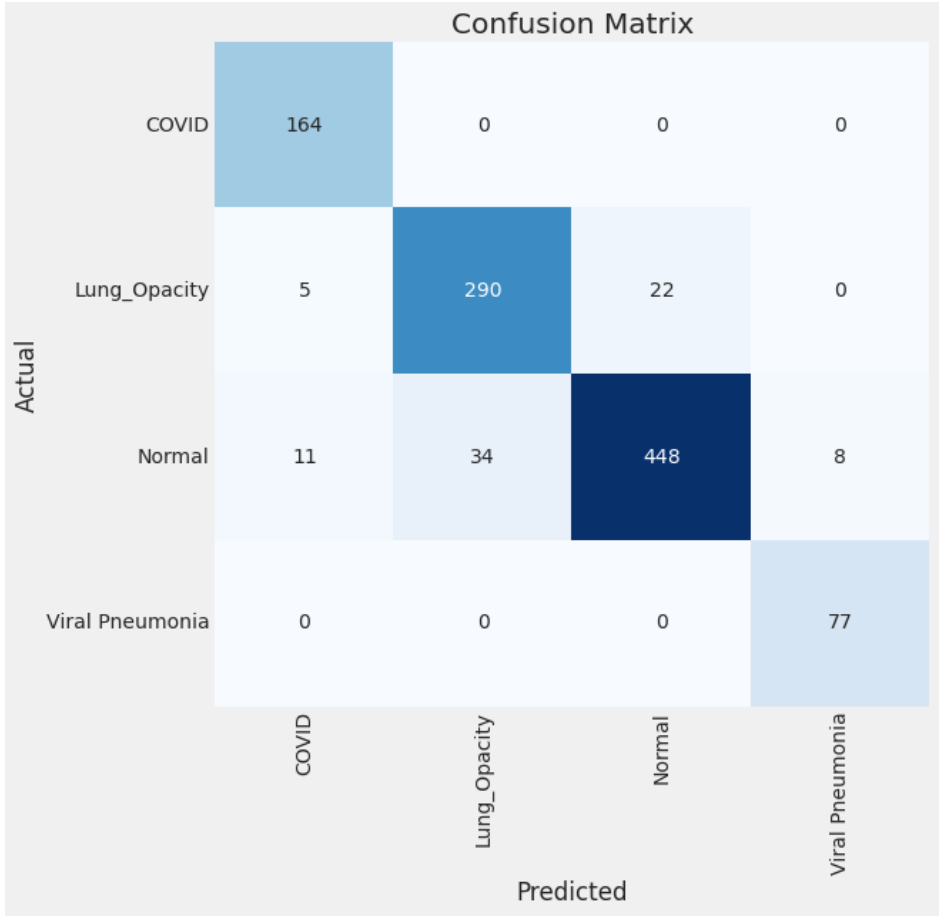
accuracy on the test set is 92.45 %



Errors by Class on Test Set

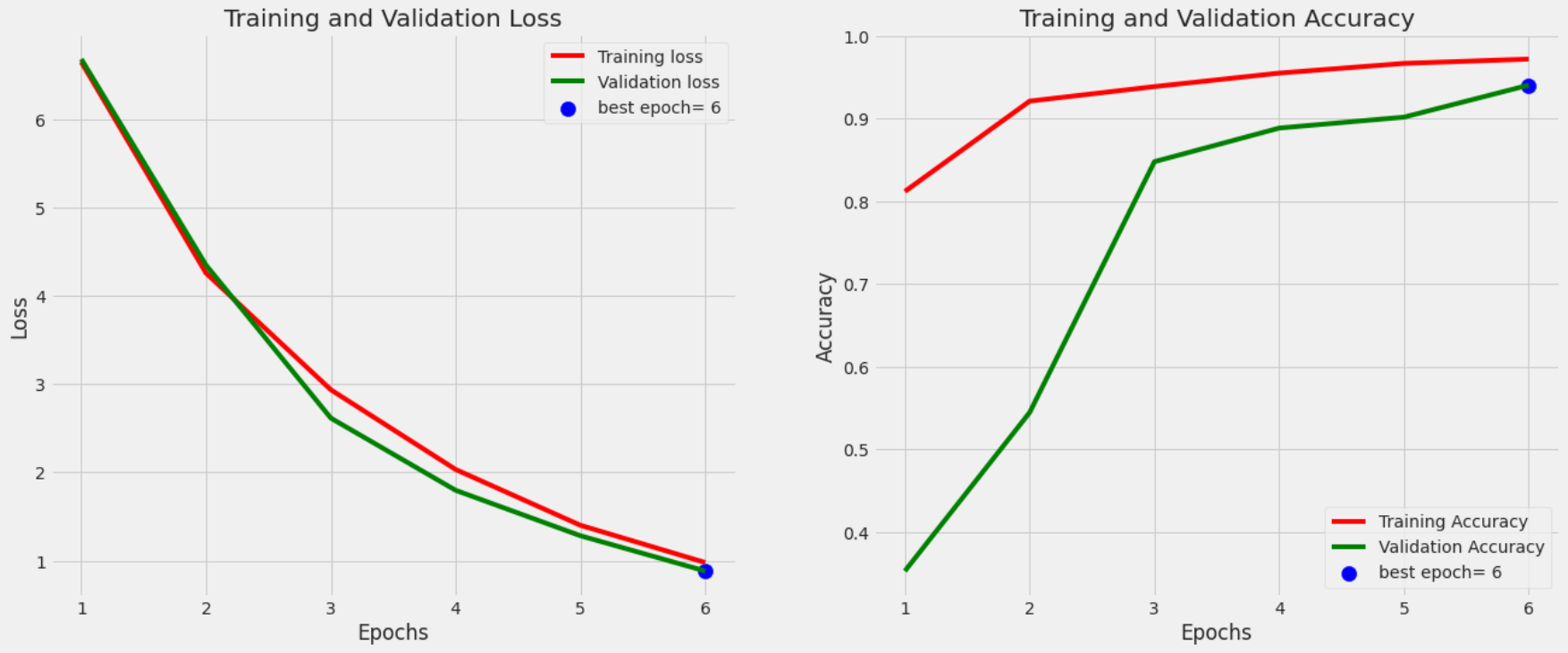


Confusion matrix



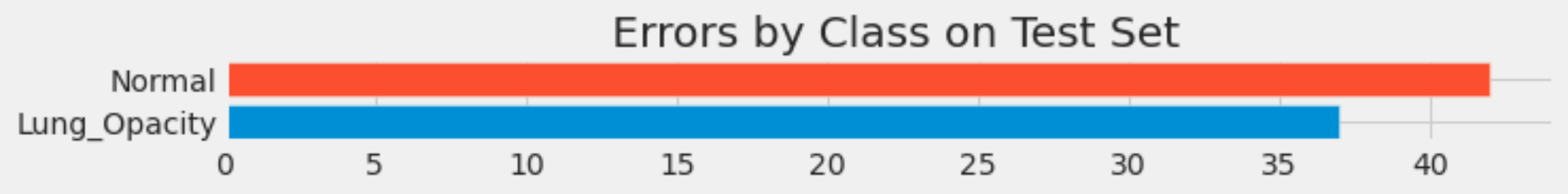
부록
Test 2 결과

accuracy on the test set is 92.54 %

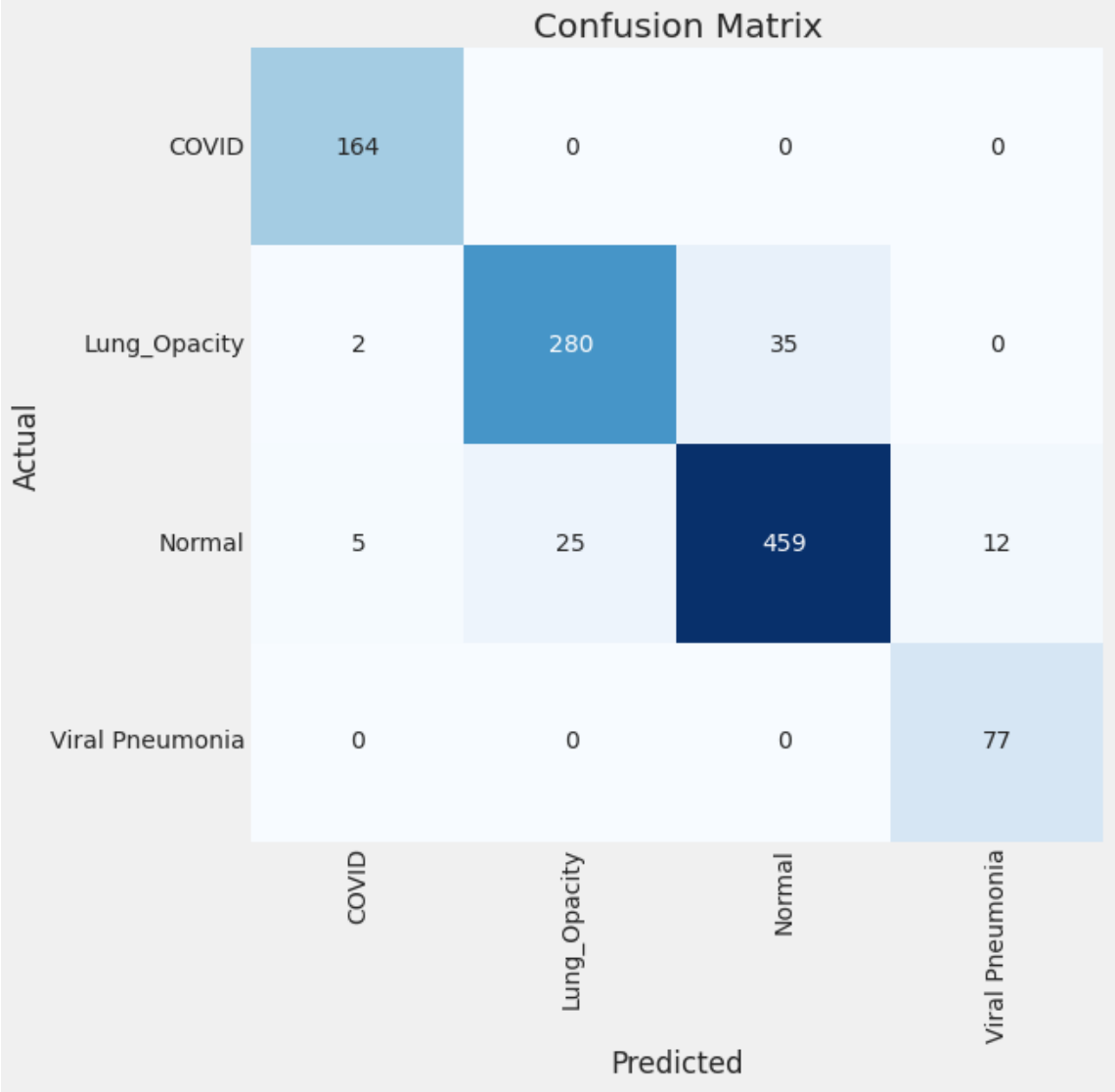


training accuracy : 97.194
validation accuracy : 94.045
test accuracy : 92.54

Errors by Class on Test Set



Confusion matrix

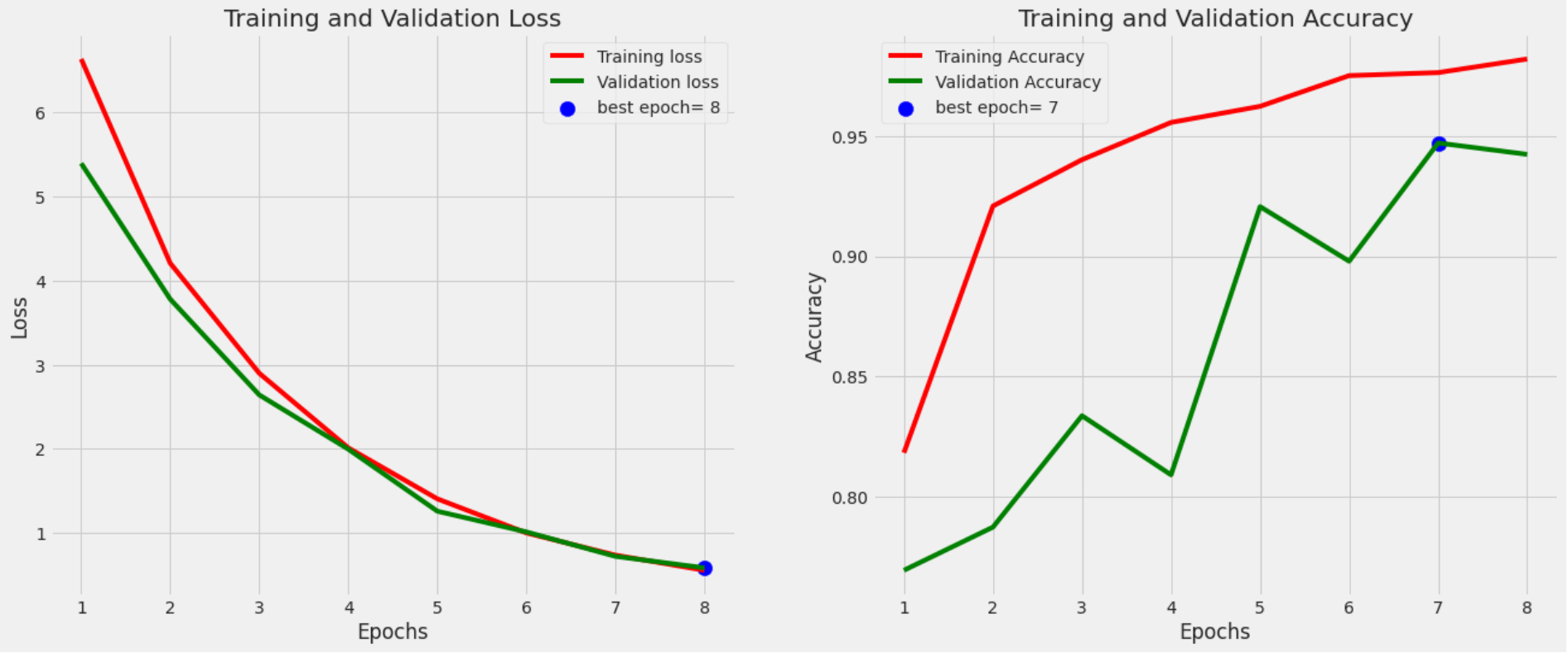


Classification Report:

	precision	recall	f1-score	support
COVID	0.96	1.00	0.98	164
Lung_Opacity	0.92	0.88	0.90	317
Normal	0.93	0.92	0.92	501
Viral Pneumonia	0.87	1.00	0.93	77
accuracy			0.93	1059
macro avg	0.92	0.95	0.93	1059
weighted avg	0.93	0.93	0.93	1059

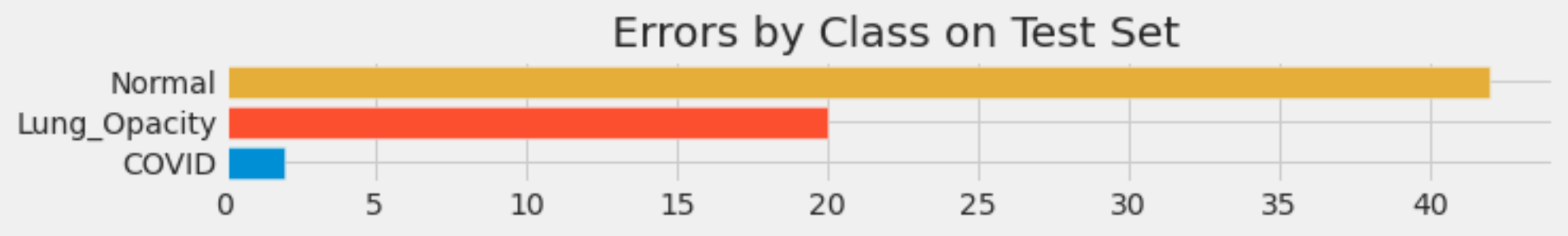
부록
Test 3 결과

accuracy on the test set is 93.96 %

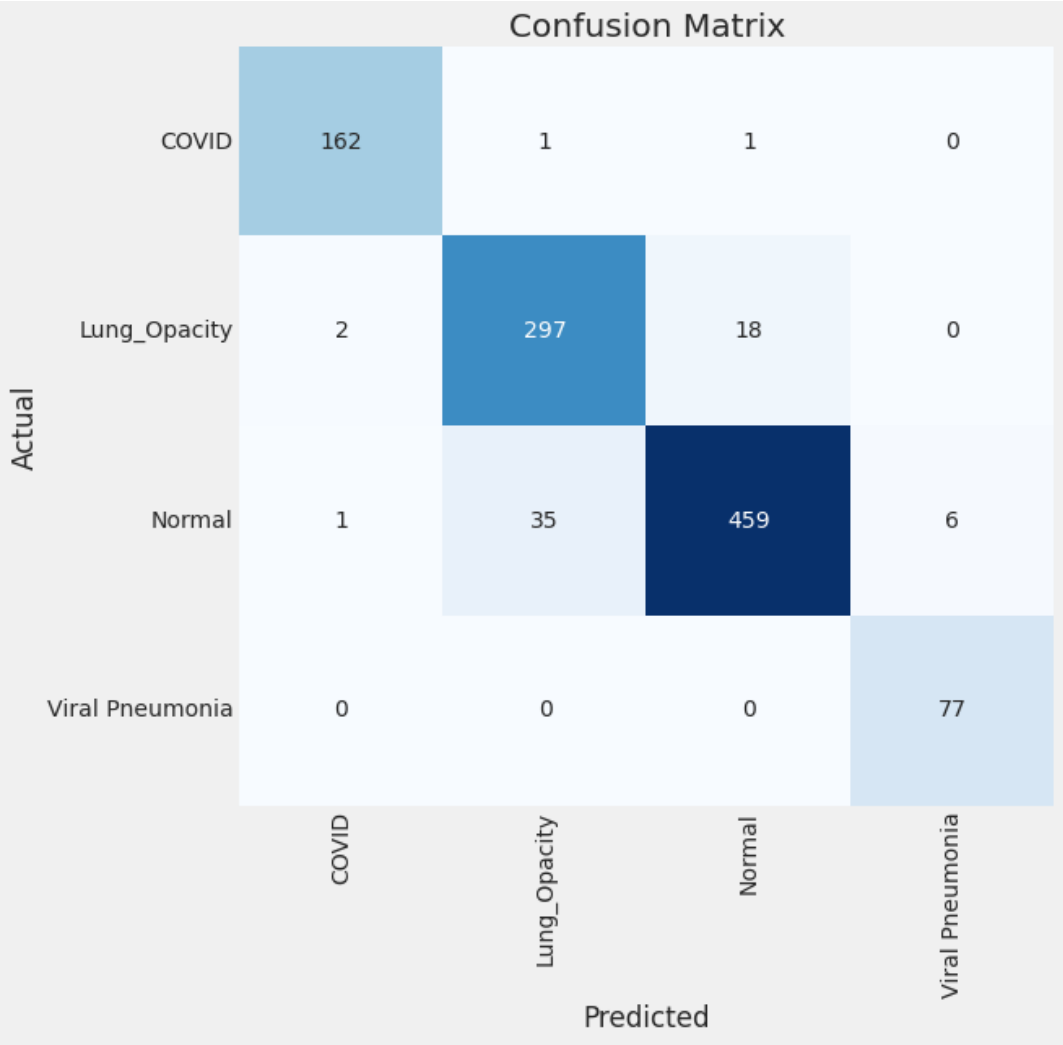


training accuracy : 98.199
validation accuracy : 94.234
test accuracy : 93.96

Errors by Class on Test Set



Confusion matrix



Classification Report:

	precision	recall	f1-score	support
COVID	0.98	0.99	0.98	164
Lung_Opacity	0.89	0.94	0.91	317
Normal	0.96	0.92	0.94	501
Viral Pneumonia	0.93	1.00	0.96	77
accuracy			0.94	1059
macro avg	0.94	0.96	0.95	1059
weighted avg	0.94	0.94	0.94	1059