KAIT 한국정보통신진흥협회
Korea Association for ICT Promotion

ICT 이노베이션 스퀘어 - 인공지능 고급 (시각)

Chest X-ray Image를 활용한 폐렴 질병진단 (4팀)

프로젝트 보고서

2024년 7월 15일

곽태혁, 김진아, 이지수, 정인화, 최영상

CONTENTS

> “ Kaggle의 흉부 x-ray 데이터를 활용한 폐렴 여부 진단 ”

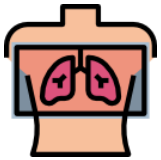
목적	이진분류 (정상/폐렴), 다중분류 (정상/바이러스성폐렴/세균성폐렴)
데이터	Kaggle’s chest_xray dataset
알고리즘	YOLOv8

2. 추진 일정

일	월	화	수	목	금	토
			3	4	5	6
			★ - 팀 구성 및 주제 선정 - 세부 진행사항 논의		- 공유 드라이브 구축 - 데이터 공유 - 프로그램 설계	
7	8	9	10	11	12	13
- 이진 분류 코드 구현 및 테스트		- 이진 분류 성능 개선 - 발표자료 작성		- 다중 분류 코드 구현 및 테스트 - 발표자료 공유 및 검토		
14	15					
	★ 프로젝트 발표					

3. 팀원 및 역할

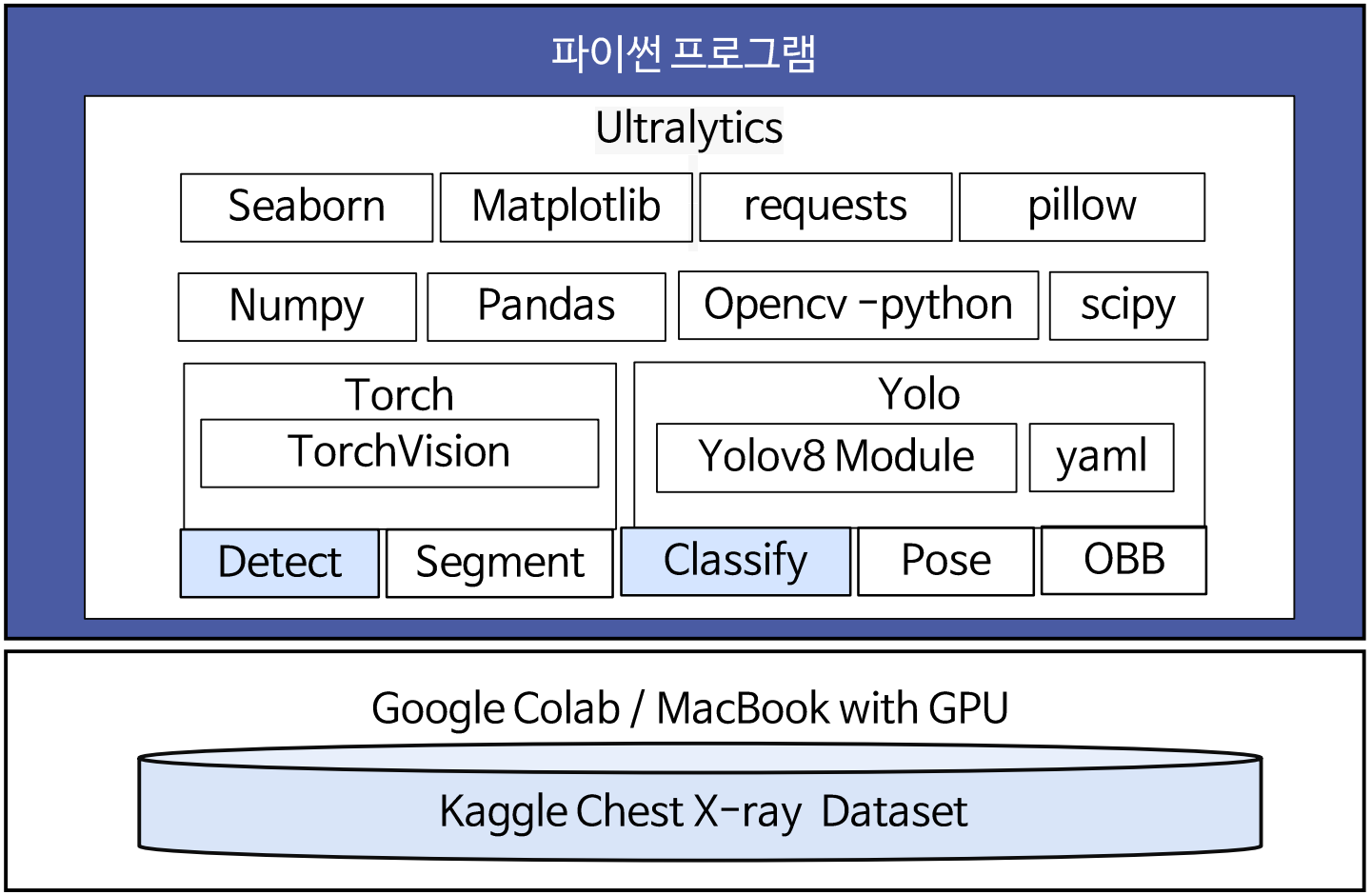
- 4팀의 팀원은 총 5명이며,
  데이터 준비, 프로그램 구현, 보고서 작성 등 3개 파트로 나누어 진행

1 데이터 준비	2 프로그램 구현	3 보고서 작성
		
이지수 chest_xray dataset 준비	김진아 Yolov8 프로그램 구현	최영상 보고서 내용 구성 및 작성
최영상 chest_xray dataset 확인	곽태혁 Yolov8 프로그램 테스팅	김진아 보고서 내용 구성 및 작성
	정인화 Yolov8 프로그램 테스팅	이지수 보고서 디자인

1. 추진 범위

소프트웨어공학단계	추진범위
요구 분석	1. Chest X-ray Dataset 조사 및 분석 2. 딥러닝 이미지 처리를 위한 알고리즘 조사 및 분석 3. 개발을 위한 각종 Library 및 Framework 분석 4. 개발 및 운영 환경 분석
설계	1. 개발환경 설계 . Google Colab 및 MacBook with GPU . Yolov8의 yaml 설계 2. Data 설계 . Kaggle Chest X-ray Dataset에서 Train, Test ,Validation 데이터 준비 3. 소프트웨어 아키텍처 및 프로그램 부분 . Yolov8의 Classification 및 Detection Module 설정 . API 설계
구현	1. 분류 프로그램 (이진 분류, 다중 분류) 2. Detection 프로그램
테스트	1. Epoch 회수 (1차 - 5회, 2차 - 10회, 3회 - 30회, 최종 - 100회) 2. Data 수 (1차 클래스당 train 약 80장 / test 10장 / valid 8장, 최종 클래스당 train 약 180장 / test 60장 / valid 60장)

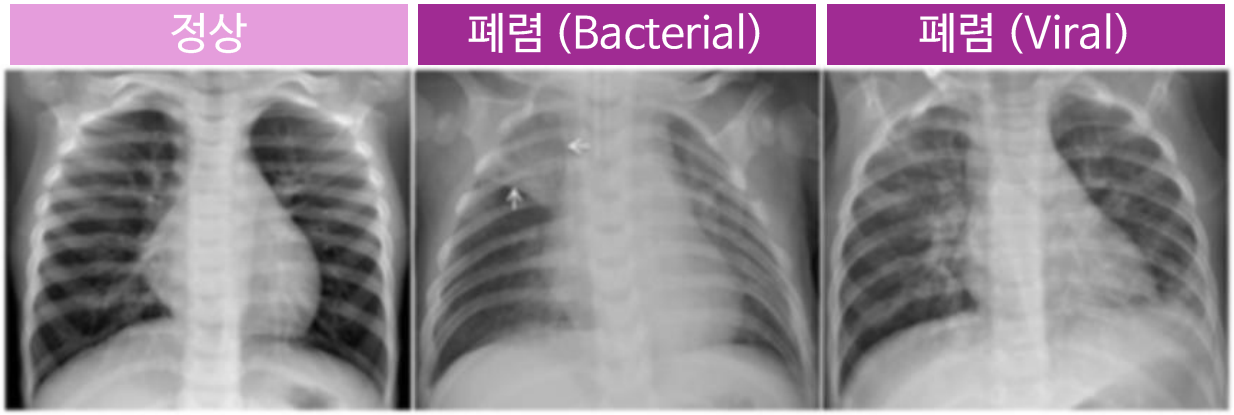
2. 구성도



- GPU 구동 서버 : Google Colab(T4 GPU), MacBook(MPS)
- 협업 툴 : Google drive, G-mail, zoom 소회의실

3. Chest X-Ray Images Dataset

- 설명 : 5,863개의 흉부 X선 이미지 (폐)
- 출처 : Kaggle
- 클래스: 이미지는 세 가지 클래스로 분류 (정상, 폐렴 - 세균성 / 바이러스성)



- 데이터세트 구조

* 폐렴 : 세균성과 바이러스성의 구체적인 이미지 수는 제공되지 않음

구분	Train	Validation	Test
정상	1,341개	8개	234개
폐렴*	3,875개	8개	390개

4. 아키텍처

- 본 프로젝트에서 활용한 아키텍처는 다음과 같음

4. 아키텍처 - (1) Ultralytics

- Ultralytics의 주요 구성 요소

models/	yolo.py common.py	YOLO 모델의 아키텍처를 정의
data	Dataset.py	데이터셋 로딩 및 전처리 기능
train.py		모델 학습 루프를 구현
val.py		모델 평가 루프를 구현
predict.py		예측 및 추론을 위한 스크립트
export.py		모델 내보내기를 위한 스크립트
utils/	general.py torch_utils.py	일반적인 유틸리티 함수들 PyTorch 관련 유틸리티 함수들
configs/	yolov5.yaml yolov3.yaml	YOLOv5 모델의 구성 파일 YOLOv3 모델의 구성 파일
README.md		

4. 아키텍처 - (1) Ultralytics

- 외부 라이브러리

```
import cv2
import numpy as np
import pandas as pd
import requests
import torch
import torch.nn as nn
from PIL import Image, ImageOps
from torch.cuda import amp
```

- Ultralytics 라이브러리

```
from ultralytics.nn.autobackend import AutoBackend
from ultralytics.yolo.data.augment import LetterBox
from ultralytics.yolo.utils import LOGGER, colorstr
from ultralytics.yolo.utils.files import increment_path
from ultralytics.yolo.utils.ops import Profile, make_divisible,
from ultralytics.yolo.utils.ops import non_max_suppression,
from ultralytics.yolo.utils.ops import  scale_boxes, xyxy2xywh
from ultralytics.yolo.utils.plotting import Annotator, colors, save_one_box
from ultralytics.yolo.utils.tal import dist2bbox, make_anchors
from ultralytics.yolo.utils.torch_utils import copy_attr, smart_inference_mode
```

4. 아키텍처 - (1) Ultralytics

- Ultralytics 클래스

```
def autopad(k, p=None, d=1):
class Conv(nn.Module):
class DWConv(Conv):
class DWConvTranspose2d(nn.ConvTranspose2d):
class ConvTranspose(nn.Module):
class DFL(nn.Module):
class TransformerLayer(nn.Module):
class TransformerBlock(nn.Module):
class Bottleneck(nn.Module):
class BottleneckCSP(nn.Module):
class C3(nn.Module):
class C2(nn.Module):
class C2f(nn.Module):
class ChannelAttention(nn.Module):
class SpatialAttention(nn.Module):
class CBAM(nn.Module):
class C1(nn.Module):
class C3x(C3):
class C3TR(C3):
class C3Ghost(C3):
class SPP(nn.Module):
class SPPF(nn.Module):
class Focus(nn.Module):
class GhostConv(nn.Module):
class GhostBottleneck(nn.Module):
class Concat(nn.Module):
class AutoShape(nn.Module):
class Detections:
class Proto(nn.Module):
class Ensemble(nn.ModuleList):
class Detect(nn.Module):
class Segment(Detect):
class Classify(nn.Module):
```

4. 아키텍처 – (1) Ultralytics

- YAML

chest_xray_data_mini3 > chest_xray.yaml

```yaml
train: /Users/i/Downloads/cv_project/chest_xray_data_mini3/train
valid: /Users/i/Downloads/cv_project/chest_xray_data_mini3/valid
test: /Users/i/Downloads/cv_project/chest_xray_data_mini3/test

# number of classes
nc: 3

# class names
names: ["NORMAL","PNEUMONIA_BACTERIA","PNEUMONIA_VIRUS"]
```

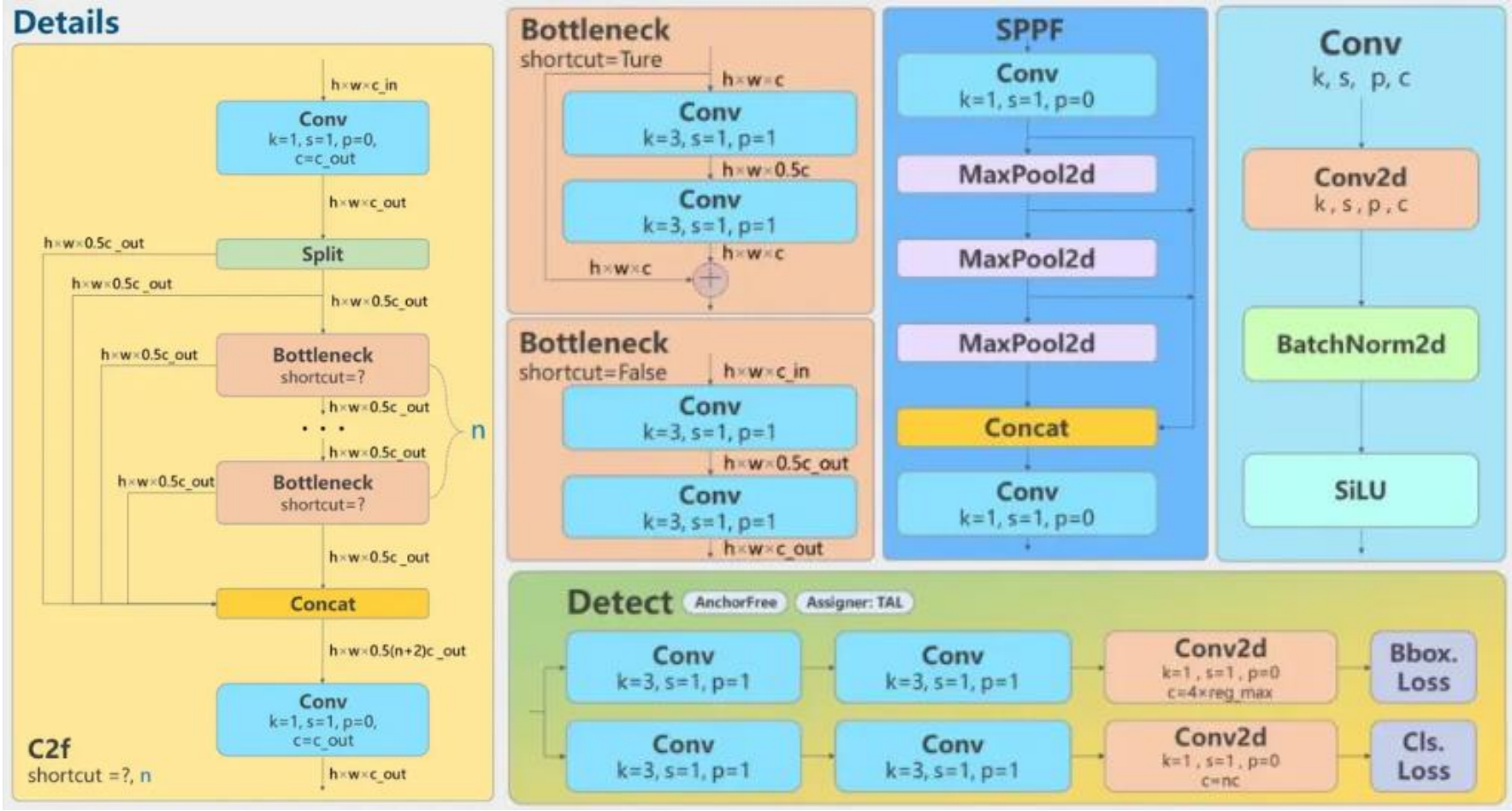
4. 아키텍처 - (2) YOLOv8

- YOLOv8의 주요 구성 요소

(1) Backbone	• 특징 추출을 위한 네트워크 • CSPNet (Cross Stage Partial Network)과 같은 효율적인 네트워크 설계를 포함
(2) Neck	• 다양한 해상도의 특징을 결합하여 정보를 강화 • PANet (Path Aggregation Network) 또는 FPN (Feature Pyramid Network) 구조를 사용
(3) Head	• 객체 탐지 및 분류를 위한 최종 레이어 • YOLOv8의 Anchor-free 구조가 사용

4. 아키텍처 - (2) YOLOv8

- YOLOv8의 주요 구성 요소들간의 흐름도



4. 아키텍처 - (3) Torch Vision

- Torch Vision 의 주요 구성 요소

(1) Datasets	다양한 이미지 데이터셋을 로드하는 기능
(2) Transforms	이미지 전처리를 위한 다양한 변환 기능을 제공 . Transforms on PIL Image . Transforms on torch.*Tensor . Conversion Transforms . Generic Transforms . Functional Transforms
(3) Models	사전 훈련된 다양한 딥러닝 모델들을 제공 . Alexnet . VGG . ResNet . SqueezeNet . DenseNet . Inception v3 . GoogLeNet
(4) Utils	기타 유틸리티 함수들로 구성

5. 구현 프로그램

이진분류 (Binary Classification)

- 정상
- 폐렴

- 클래스 수 : 2 (정상, 폐렴)
- 초기 사용 가중치 : YOLOv8x-cls
- 데이터셋 비율
  - Train : test : valid = 6 : 2 : 2 (360장, 120장, 120장)
- Epoch : 100

```
|-- chest_xray
|   |-- test
|   |   |-- NORMAL
|   |   |   `-- IM-0001-0001.jpeg
|   |   `-- PNEUMONIA
|   |       `-- person78_bacteria_378.jpeg
|   |-- train
|   |   |-- NORMAL
|   |   |   `-- IM-0115-0001.jpeg
|   |   `-- PNEUMONIA
|   |       `-- person1_bacteria_1.jpeg
|   |-- train.cache
|   |-- val
|   |   |-- NORMAL
|   |   |   `-- IM-0099-0001.jpeg
|   |   `-- PNEUMONIA
|   |       `-- person139_bacteria_665.jpeg
|   `-- val.cache
|-- chest_xray.yaml
|-- train.cache
`-- val.cache
```

그림1. 데이터 파일 구조

```
|-- models
|   `-- yolov8x-cls.pt
|-- predict
|   `-- person1_virus_6.jpeg
|-- train
|   |-- args.yaml
|   |-- confusion_matrix.png
|   |-- confusion_matrix_normalized.png
|   |-- results.csv
|   |-- results.png
|   |-- train_batch0.jpg
|   |-- train_batch1.jpg
|   |-- train_batch2.jpg
|   |-- val_batch0_labels.jpg
|   |-- val_batch0_pred.jpg
|   `-- weights
|       |-- best.pt
|       `-- last.pt
`-- val
    |-- confusion_matrix.png
    |-- confusion_matrix_normalized.png
    |-- val_batch0_labels.jpg
    `-- val_batch0_pred.jpg
```

그림2. 코드 실행 결과 저장 파일 구조

구분	Train	Validation	Test
정상	180개	60개	60개
폐렴*	180개	60개	60개

* 폐렴은 bacteria, virus 중 bacteria 데이터 사용

표1. 데이터셋 비율 표

5. 구현 프로그램

다중분류 (Multi Classification)

- 정상
- 폐렴 (세균성)
- 폐렴 (바이러스성)

- 클래스 수 : 3 (정상, 세균성 폐렴, 바이러스성 폐렴)
- 초기 사용 가중치 : YOLOv8x-cls
- 데이터셋 비율
  - Train : test : valid = 6 : 2 : 2 (540장, 180장, 180장)
- Epoch : 100

```
|-- chest_xray
|   |-- test
|   |   |-- NORMAL
|   |   |   `-- IM-0001-0001.jpeg
|   |   |-- PNEUMONIA_BACTERIA
|   |   |   `-- person78_bacteria_378.jpeg
|   |   `-- PNEUMONIA_VIRUS
|   |       `-- person1_virus_6.jpeg
|   |-- train
|   |   |-- NORMAL
|   |   |   `-- IM-0115-0001.jpeg
|   |   |-- PNEUMONIA_BACTERIA
|   |   |   `-- person1_bacteria_1.jpeg
|   |   `-- PNEUMONIA_VIRUS
|   |       `-- person1389_virus_2387.jpeg
|   |-- train.cache
|   |-- val
|   |   |-- NORMAL
|   |   |   `-- IM-0099-0001.jpeg
|   |   |-- PNEUMONIA_BACTERIA
|   |   |   `-- person139_bacteria_665.jpeg
|   |   `-- PNEUMONIA_VIRUS
|   |       `-- person55_virus_110.jpeg
|   `-- val.cache
|-- chest_xray.yaml
|-- train.cache
`-- val.cache
```

그림1. 데이터 파일 구조

```
|-- models
|   `-- yolov8x-cls.pt
|-- predict
|   `-- person1_virus_6.jpeg
|-- train
|   |-- args.yaml
|   |-- confusion_matrix.png
|   |-- confusion_matrix_normalized.png
|   |-- results.csv
|   |-- results.png
|   |-- train_batch0.jpg
|   |-- train_batch1.jpg
|   |-- train_batch2.jpg
|   |-- val_batch0_labels.jpg
|   |-- val_batch0_pred.jpg
|   `-- weights
|       |-- best.pt
|       `-- last.pt
`-- val
    |-- confusion_matrix.png
    |-- confusion_matrix_normalized.png
    |-- val_batch0_labels.jpg
    `-- val_batch0_pred.jpg
```

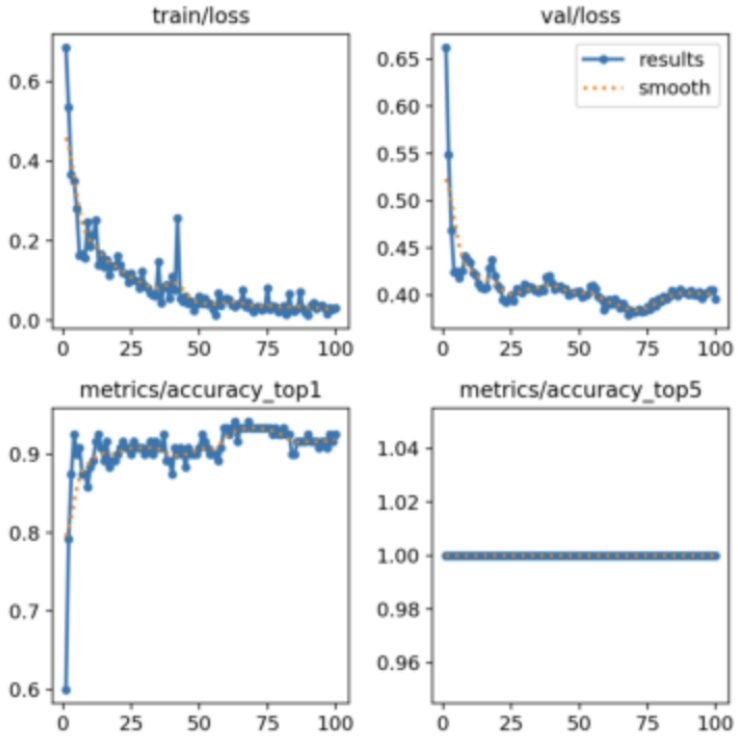
그림2. 코드 실행 결과 저장 파일 구조

구분	Train	Validation	Test
정상	180개	60개	60개
폐렴 (세균성)	180개	60개	60개
폐렴 (바이러스성)	180개	60개	60개

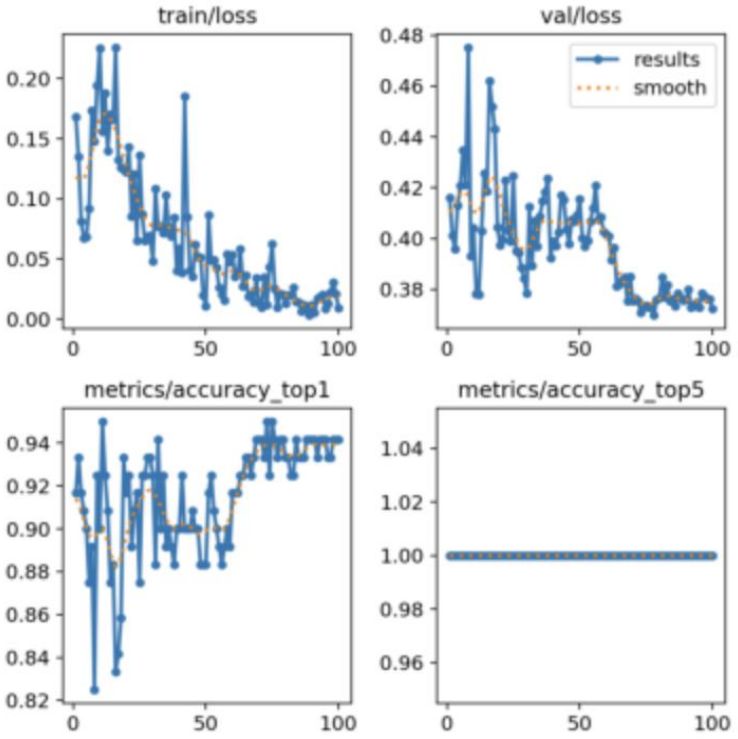
표1. 데이터셋 비율 표

5. 구현 프로그램

이진분류 손실 및 성능 지표 결과



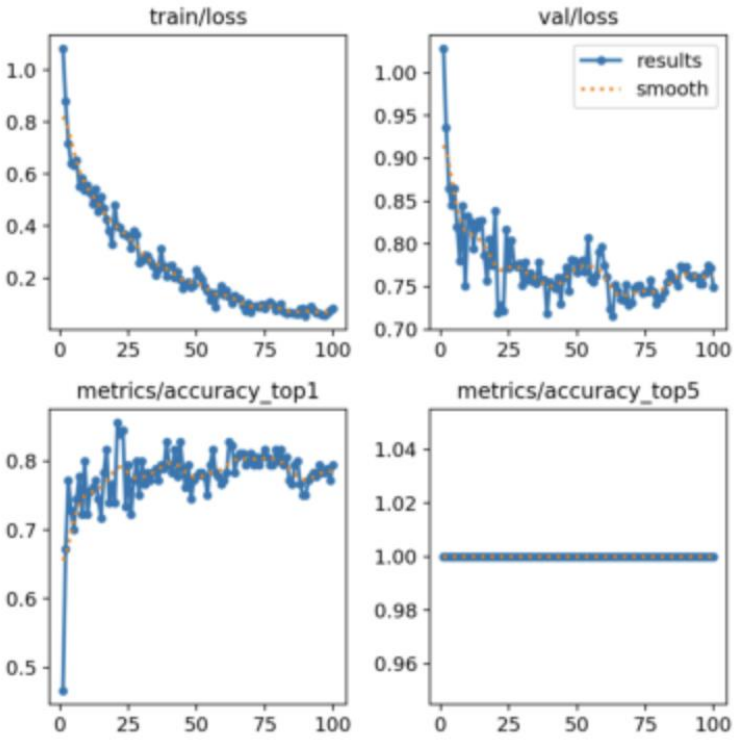
yolov8**m**-cls



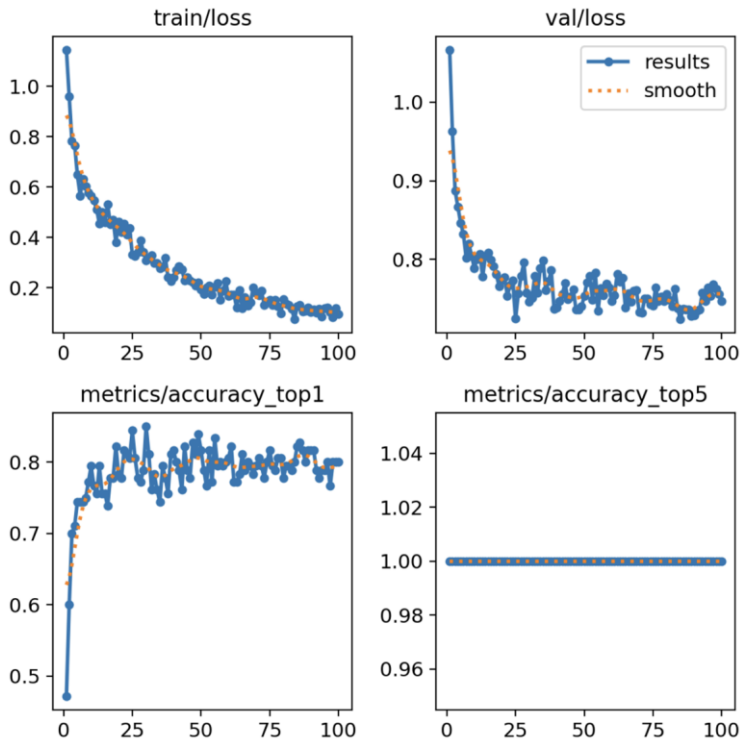
yolov8**x**-cls

5. 구현 프로그램

다중분류 손실 및 성능 지표 결과



yolov8**m**-cls



yolov8**x**-cls

5. 구현 프로그램

- Ultratics 공식 레포지토리에 공표된 YOLOv8-cls 가중치별 성능 결과를 토대로 가중치 별 성능 테스트 후 사용할 가중치를 설정함.

Model	size (pixels)	acc top1	acc top5	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B) at 640
YOLOv8n-cls	224	69.0	88.3	12.9	0.31	2.7	4.3
YOLOv8s-cls	224	73.8	91.7	23.4	0.35	6.4	13.5
YOLOv8m-cls	224	76.8	93.5	85.4	0.62	17.0	42.7
YOLOv8l-cls	224	76.8	93.5	163.0	0.87	37.5	99.7
YOLOv8x-cls	224	79.0	94.6	232.0	1.01	57.4	154.8

- 분석 결과, Yolov8x-cls가 Yolov8m-cls보다 분류 성능이 더 높음을 확인하였음.

	YOLOv8m-cls	YOLOv8x-cls
이진분류	94.2	95.0
다중분류	85.0	85.6

6. 시연(in macbook)

이중분류 시연

6. 시연(in macbook)

다중분류 시연

1. Learn & Lesson

이지수

이미지 분석에서 활용되는 고전 모델부터 최신 기법까지 차근히 배울 수 있어 너무 좋았습니다. 그 동안 여러 강의들을 많이 접했지만, 강사님께서 자세히 설명해주셔서 가장 이해도 잘되고 배울 게 많은 수업이었다고 생각합니다.
좋은 팀원분들과 만나 프로젝트까지 잘 마무리할 수 있어, 강사님과 팀원 분들께 감사의 말씀을 전하고 싶습니다.

최영상

본 과정에서 딥러닝 이미지 처리 최신기술을 배울 수 있어 좋았습니다. 프로젝트에 참여하면서 Yolov8로 classification을 해보았고, 이중/삼중 분류는 물론, Ultralytics, yolov8의 아키텍처를 심도 있게 공부할 수 있어서 뜻 깊은 시간이었습니다.

김진아

yolov8을 사용해서 task를 할 때 경로 설정이 매우 중요하다는 점을 다시 한번 인지하게 되었고, 데이터 증강을 시도해보지 못한 점이 아쉬웠습니다. 또한, github 레포지토리를 참고하면서 classification을 하는데 필요한 함수의 파라미터들에 대해 공부할 수 있어 좋았습니다.

곽태혁

프로젝트에 참여하면서 객체 인식에 대해 공부할 수 있어 좋았습니다. 이미지 합성곱(CONV)과 필터 등을 구현하여 실험을 해보았고, Pix2pix 등 다양한 이미지 분석 모델의 기본 개념을 알 수 있는 계기가 되었습니다.

정인화

폐렴 진단 AI모델 중 CheXNet과 VUNO-Med-Chest X-ray 모델이 가장 성능이 좋다는 것을 알게 되었습니다.
다음 기회에는 YoLoV8과 성능 비교를 시도해 보고자 합니다.

Chest X-ray Image를 활용한 폐렴 질병진단 (4팀)

Thanks for your listening