**□ 최종 결과보고서**

1. 산출물 기준

1) 분량 : 자유롭게 작성하나 10매 내를 권장 [내용이 길어지면 부록\_형식으로 분리 권장]

2) 기한 : **세미 프로젝트 종료일 (3/23)** 까지

3) 워드(docx) 파일로 작성

(예 : [4차 6기] 딥러닝A\_결과보고서\_팀명.docx)

4) 제출 : 1. **결과보고서 워드 파일(또는 pdf), 발표PPT 파일 등** 제출 및 NCS평가시스템 업로드

2. 매니저 이메일로 전송 (ym320.shin@multicampus.com)

2. 프로젝트 목차

1) 표지 - 1매

2) 목차 - 1매

3) 팀 구성원 및 역할 소개 : 개인별 업무 분장 - 1매

4) 기획 배경 및 목표 - 1~2매

5) 추진 계획(일정) : 팀 전체 일정 및 개인별 업무 진행 일정 - 1매

6) 결과

- 데이터 수집 및 분석 : 자유롭게 [3매 내외 권장]

- 분석 결과 : 분석, 설계자료, 핵심적 알고리즘, 주요 동작 등 그림, 표 등

팀별로 자유롭게 기술 - 자유롭게 [3매 내외 권장]

7) 기대 효과 - 1매

8) 개발 후기 : 프로젝트 진행 팀원 단체 사진(1~2장) 및 개인별 후기 - 2매

※ 본 목차의 **대목차**는 **유지**하되 산출물 결과에 따라

세부목차는 강사님과 협의하여 각 팀 프로젝트에 맞게 수정 가능함

3. 작성 유의사항

1) 현재 페이지는 삭제(페이지 삭제)하여 표지가 첫번째 페이지가 되도록 함

2) 페이지 번호 삽입(삽입 > 페이지 번호 > 아래쪽 > 굵게 표시된 번호2)

3) 목차 이후 프로젝트 개요부터 페이지 목차 설정(바닥글 1페이지 시작)

4) **대제목 [16pt, Bold]** / 소제목 [12pt] / 본문 [10pt]

4차 6기 세미프로젝트

|  |
| --- |
| Steel Defect Detection |

**2020년 3월 23일**

**딥러닝 기반 AI 엔지니어링 (A)**

STS**(팀명)**

권윤옥

김수연

이선주

장경석

전현진

정혁진

**목 차**

**1. 프로젝트 개요**  **1**

1.1 프로젝트 기획 배경 및 목표 1

1.2 구성원 및 역할 2

1.3 프로젝트 추진 일정 3

**2. 프로젝트 결과**  **7**

2.1 데이터 수집 7

2.2 데이터 분석 8

2.3 데이터 분석 결과 9

**3. 기대 효과**  **10**

3.1 향후 개선 사항 10

3.2 기대 효과 11

**4. 개발 후기**  **12**

**1. 프로젝트**

1.1 프로젝트 기획 배경 및 목표

* **기획 배경**

철강은 현대에서 가장 중요한 건축 자재 중 하나입니다. 철강 결함 식별을 자동화 하는 것은 생산성에 큰 도움이 됩니다. 철강 회사들은 자동화를 개선하고 효율성을 높이며 생산에서 고품질을 유지하기 위해 기계 학습을 찾고 있습니다. 이를 해결해보고자 프로젝트를 진행할 예정입니다.

* **프로젝트 목표**

과정을 통해 배운 Python 및 Flask 를 이용하여 Mask R-CNN을 사용한 철강 결함 감지 및 분류

1.2 구성원 및 역할

|  |  |  |  |
| --- | --- | --- | --- |
| 이름 | 전공 | 역할 | 구현 부분 |
| 권윤옥 |  | 팀장 | 프로젝트 관리  데이터 분석 |
| 김수연 |  | 팀원 | 데이터 분석 |
| 이선주 |  | 팀원 | 데이터 분석 |
| 장경석 |  | 팀원 | Back-end |
| 전현진 |  | 팀원 | Back-end |
| 정혁진 |  | 팀원 | Front-end |

1.3 프로젝트 추진 일정

|  |  |  |  |
| --- | --- | --- | --- |
| 구분 | 기간 | 활동 | 비고 |
| 사전  기획 | 3/17 | 프로젝트 기획 및 팀 구성 |  |
| 3/17 | 프로젝트 주제 선정 | 3~5 인/팀 |
| 프로젝트  수행  /  완료 | 3/18 | 프로젝트 수행 - 데이터 분석 및 딥러닝 모델 선정 - 웹 어플리케이션 구상 |  |
| 3/18 ~ 3/21 | 프로젝트 설계 - 모델 분석 및 시각화 - UI 구상 및 기능 설계 - 웹 어플리케이션 프레임워크 설계 |  |
| 3/22 | 구현 및 테스트 - 모델 성능 테스트 - UI 구현 및 수정 - 웹 어플리케이션 구현 및 테스트 |  |
| 3/23 | 최종 발표 - 구축 완료 보고 | 최우수 한 팀 선발 |

**2. 프로젝트 개발 결과 [대제목 : 16pt / 맑은 고딕]**

2.1 데이터 수집 [소제목 : 12pt / 맑은 고딕]

* Kaggle Steel Defect Detection 제공



(출처 : <https://www.kaggle.com/c/severstal-steel-defect-detection>)

* 데이터 구성

1. train\_images

- 학습에 필요한 철강 image set.

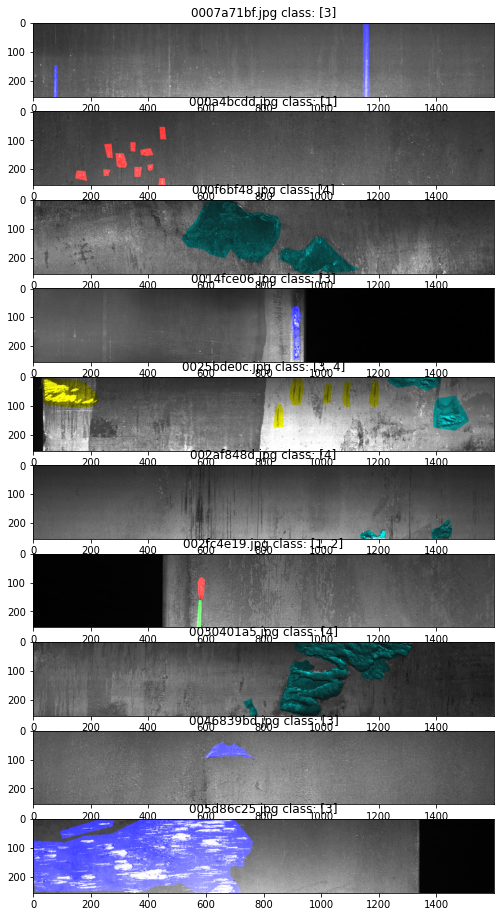
2) test\_images

- 학습 완료 후 결함을 예측할 철강 image set.

3) train.csv

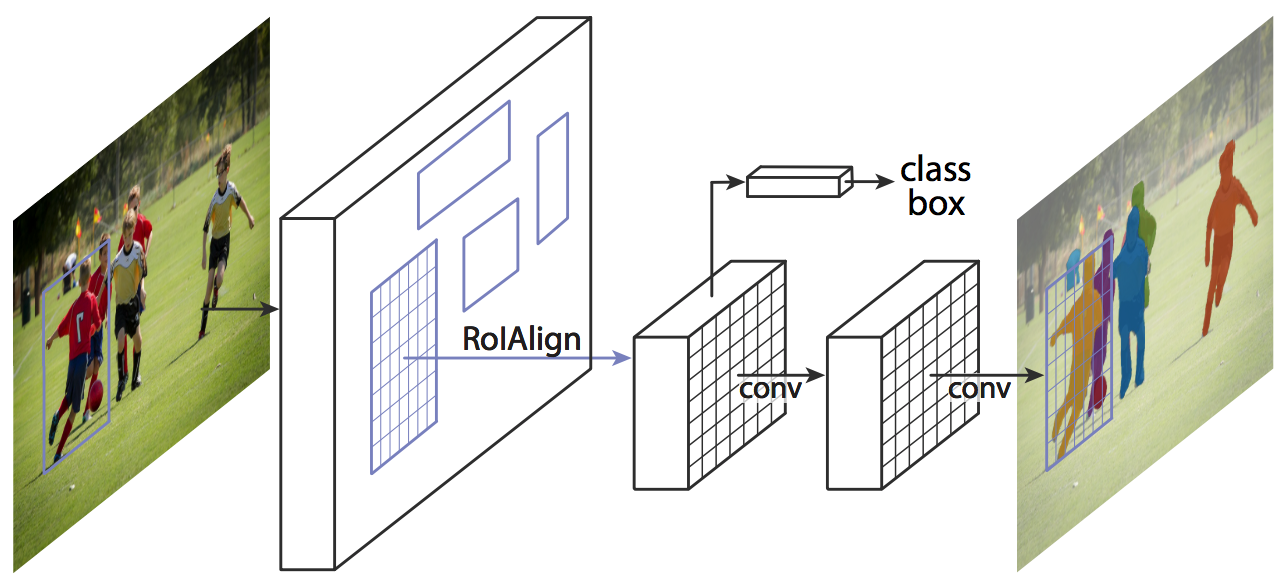
- 학습에 필요한 철강 defect encoded pixels, class data.

(철강 이미지 예시)

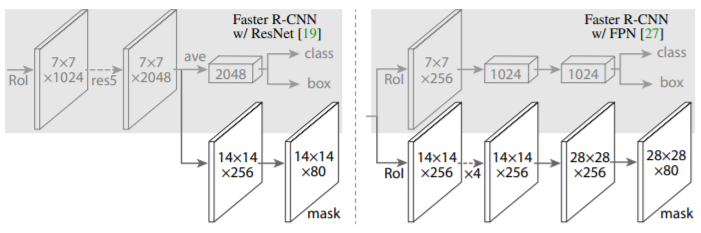


2.2 데이터 분석

* **모델 선정: Mask R-CNN 모델**
  + Mask R-CNN은 Faster R-CNN(Object Detection)을 확장하여 Instance Segmentation에 적용하고자 하는 모델입니다. Mask R-CNN을 간략하게 요약한다면, Faster R-CNN에서 detect한 각각의 box에 mask를 씌워주는 모델입니다.

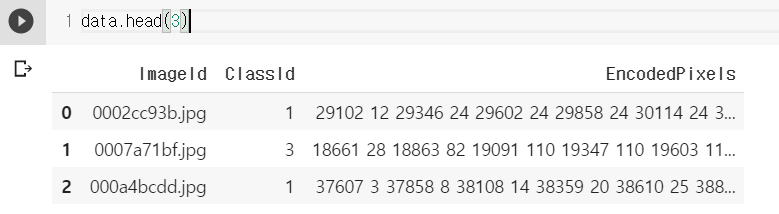


(이미지: 인스턴스 분할을 위한 Mask R-CNN 프레임워크)

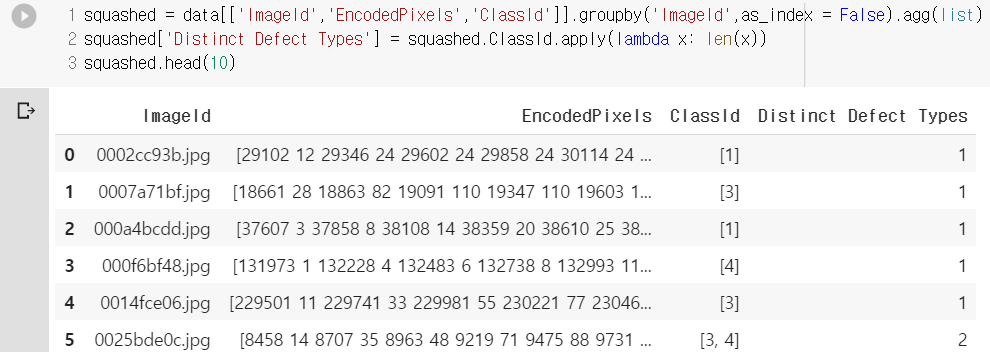


(Mask R-CNN Head Architecture)

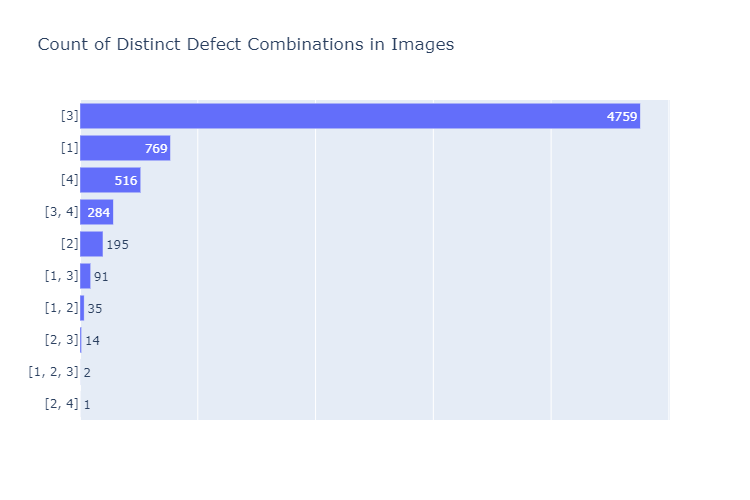
* **데이터 전처리**
  + train.csv은 다음과 같다.



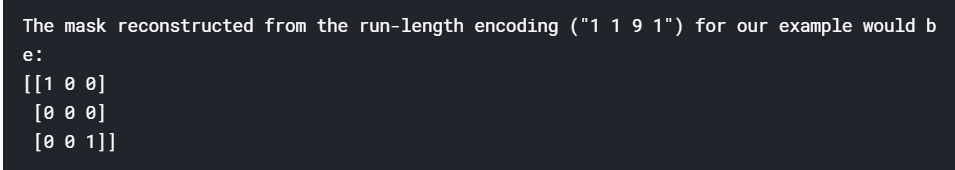
* + ImageId로 그룹화하여 한 이미지에 여러가지 클래스의 결함이 존재할 경우를 하나의 row로 줄이고, Distinct Defect Types column을 추가하여 총 클래스 개수 정보를 나타낸다.



* + 데이터 프레임 결합 후 결함 클래스 별 이미지 수는 다음과 같다.



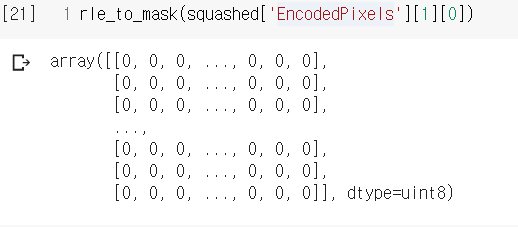
* + EncodedPixels는 결함 위치 정보를 Run-Length Encoding한 것이다. 따라서 이를 Decoding해주는 작업이 필요하다. Run-Length Encoding이란 간단한 비 손실 압축 방법으로, 데이터에서 같은 값이 연속해서 나타나는 것을 그 개수와 값으로 표현한다. 즉 중복되는 문자를 한 문자로 치환하는 것이다. 예를 들면, 1 1 9 1로 RLE 된 것을 Decoding하면 [1 0 0][0 0 0][0 0 1]이 된다.



* + 본 프로젝트에서 EncodedPixels column은 run-length 인코딩 되어있다.



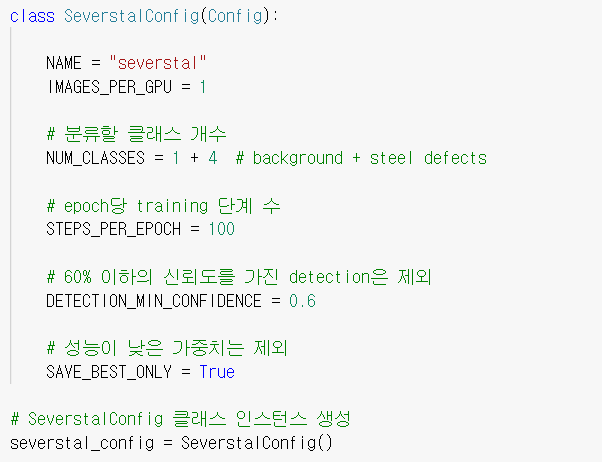
* + 이를 사용자 정의한 rle\_to\_mask 함수를 통하면 다음과 같이 디코딩 된다.



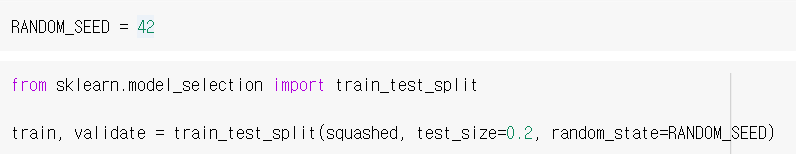
* + 디코딩 된 정보는 철강의 결함위치를 알려주며, 이미지로 출력하면 아래와 같다.

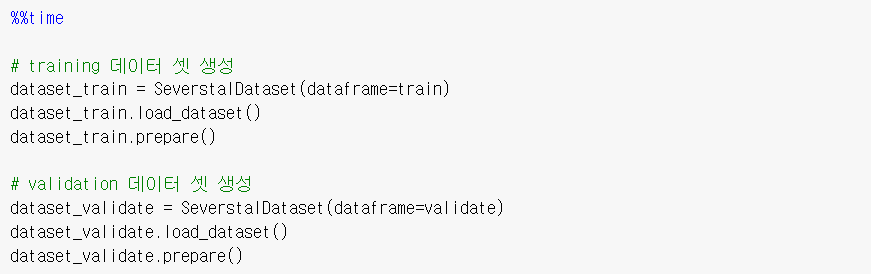


* **모델 구성하기**
  + Mask R-CNN 모델에 이미지를 학습시키기 위해 학습시킬 이미지와 사용환경에 맞게 모델을 설정해준다.



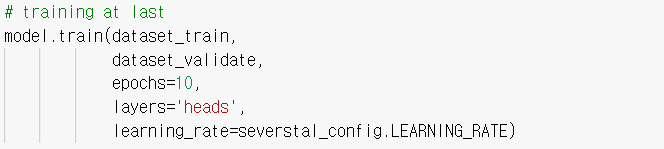
* **모델 학습 및 테스트**
  + 학습시킬 데이터와 테스트 할 데이터를 분리한 후 Mask R-CNN 모델에 맞게 데이터 셋을 생성한다.



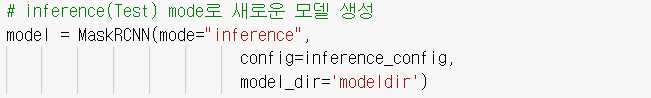


* + 모델을 생성하여 학습 시킨다.





* + 학습시킨 모델에 대한 테스트를 수행하기 위해 inference 모드로 모델을 재생성





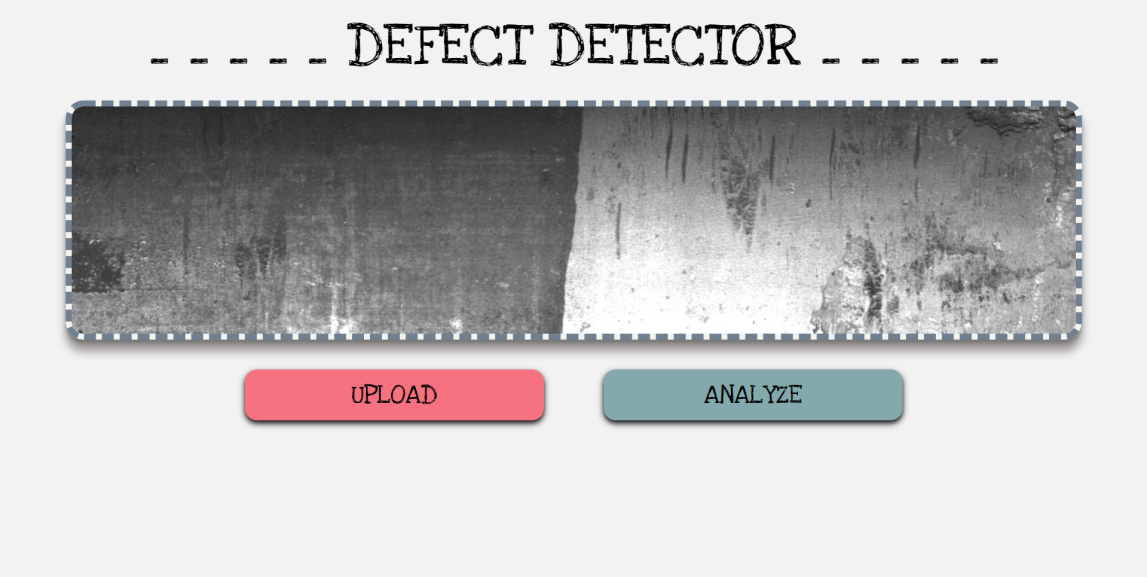
* + 임의의 이미지로 모델 테스트 수행



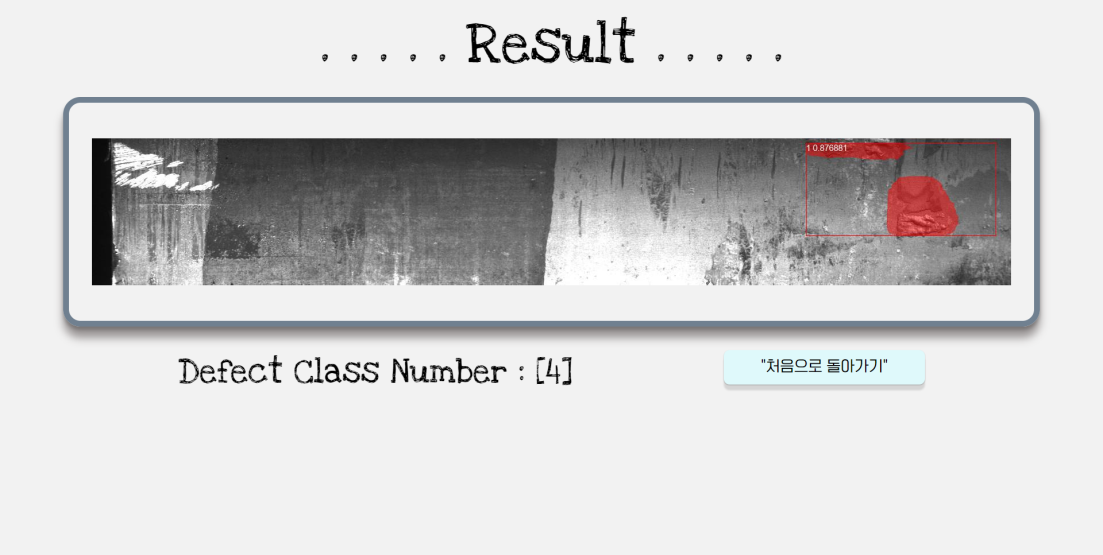


2.3 데이터 분석 결과

* Flask를 이용해 웹 페이지를 제작하여 서비스 구현



* 학습시킨 모델을 통해 철강 이미지를 분석했을 때 나타나는 결과 웹 페이지



**3. 기대 효과**

3.1 향후 개선 사항

Epoch를 횟수를 늘려 정확도를 높인다. 7000여개의 데이터를 generate를 통해 증강시켜 더 많은 데이터로 학습시킬 수 있게 한다. Mask RCNN이 아닌 다른 모델을 사용하여 학습해본다. 특히 의료영상 분석에 많이 사용하는 U-Net 모델을 사용해, 기존 모델과 비교하여 성능차이가 있는지 알아본다. 본 데이터와 비교 가능하게 하여 얼마나 예측이 잘 되었는지 육안으로 확인해 볼 수 있게 개선한다. 분석 시 소요되는 시간을 줄여 실제 철강 공정에 활용할 수 있게 한다.

3.2 기대 효과

고주파 카메라의 이미지를 사용한 결함 감지 알고리즘을 강화할 수 있다. 알고리즘 강화로 하여금 철강생산 과정의 자동화를 개선할 수 있다. 또한 철강 결함을 식별하는 데 도움을 주어 철강 생산 효율을 높힐 수 있을 것이며 고품질의 생산이 가능해질 것이다. 이를 통해 철강의 제조 표준을 높게 유지하고 철강 채굴 및 생산 분야의 혁신을 이룩할 수 있다.

**4. 개발 후기**

|  |
| --- |
|  |

|  |  |
| --- | --- |
| 성명 | 후기 |
| 권윤옥 | 이번 세미 프로젝트를 통해 Mask R-CNN 모델 구조를 이해할 수 있는 계기가 되었습니다. 모델을 학습시키기 위해 모델을 사용목적과 데이터에 맞게 구성하는 법을 알 수 있었습니다. 앞으로 있을 파이널 프로젝트를 대비하여 어떻게 공부해야 하는지 대략적으로 방향을 잡을 수 있게 되었습니다. 그리고 데이터 분석에 사용한 코드를 Flask 프레임워크에 적용하는 과정에서 코드를 공유하고 합칠 때 소통의 중요성을 느낄 수 있는 좋은 기회였습니다. |
| 김수연 | 영상분석에 쓰이는 여러가지 모델에 대해 배울 수 있었다. 그 중 Mask R-CNN이라는 모델을 사용하며 object detection뿐만 아니라 segmentation까지 다뤄보았다. Deep learning을 활용한 영상처리 프로세스에 대해 더 잘 알게 되었다. 또한 run-length encoding이라는 데이터 압축 방식을 익혔으며, 해당 방식으로 압축된 데이터를 처리할 수 있게 되었다. 향후 과제로 예측 성능향상을 위해 U-Net이라는 모델에 대해 추가적으로 공부해도 좋을 것이다. |
| 이선주 | 철강 이미지에서 결함을 더 정확하게 감지하기 위해 이미지 보정 작업을 진행했지만 예상했던 것과는 달리 결함을 잘 찾아내지 못했다. 원인을 분석해본 결과 첫째, 학습용 레이블-이미지 데이터의 수가 많지 않았고 둘째, 데이터를 학습하는 동안 이미지를 변형시켜 더 다양한 종류의 결함을 감지할 수 있도록 학습시켜야 모델의 성능을 더 향상시킬 수 있을 것 같은데 시간 부족으로 이 작업까지 완료하지 못해 아쉽다. |
| 장경석 | 현재 교육과정에 목표로 했던 '데이터 활용 또는 데이터 분석모델을 활용한 웹 서비스 구현 방법'에 대해 한걸음 다가간 느낌입니다. 시행착오가 상당히 많았기 때문에 더욱 배울 점이 많았습니다. flask를 통해 tensorflow 분석 모델을 웹 서비스에 적용하기 위해 tensorflow.session과 tensorflow.graph에 대해 공부하고 웹 캐시메모리의 역할, 웹 서비스에서의 정적 데이터 등에 대한 공부를 하게 되었습니다. semi final 프로젝트답게 final 프로젝트에 도움이 될만한 내용들을 얻게 되어 기분 좋게 공부했습니다. 또한 팀 구성원과 분담한 파트를 결합할 때에 진행할 방법에 대해서도 체득하게 되었습니다. 현재 결과물에 그치지 않고 final project에서 사용하게 될 '분석모델 API서버'에 대해 공부하고 싶습니다. |
| 전현진 | Flask 프레임워크를 사용해 모델을 서버에 올려보고 predict 해보는 경험을 해서 좋았습니다. 그 중 문제점을 해결한 것이 좋았는데, 모델을 불러오는 과정에서 특정 URL을 호출 할 때 모델을 불러오는 방법보단 Flask 서버가 init 될 때 모델을 불러오는 방법이 속도가 빨랐습니다. 대규모 프로젝트는 처음이라 데이터 파트 프런트 파트와 코드를 합칠 때 어려움이 있었지만, 각자의 코드와 생각을 맞춰가면서 완성하는 과정이 의미 있고 좋았습니다. |
| 정혁진 | 프론트엔드 개발자로 진로를 바꾼 후 작업한 첫 프로젝트인 만큼, 공부한 것들을 충분히 활용하려 노력했고, UI 작업을 하며 JS의 중요성에 대해서 다시 한 번 느꼈습니다. 백엔드 개발 동료들과 데이터를 주고받으며 전반적인 데이터 흐름도 배울 수 있었습니다. 또한, 이미지 업로드와 버튼 태그에서 생긴 문제점들이 있었지만, 구글링을 통해 충분히 커버할 수 있었고, 이미 알고 있던 태그들의 숨겨진 기능도 공부할 수 있는 좋은 기회가 되었습니다. |