Object Detection WrapUP Report

Haneol Kim* , Bohyun Kim† , Sungjoo Kim‡, Suhyun Jung§, Namgyu Yun¶, Minseok Heo‖ [1]

[1] *Naver Boostcamp AI Tech, CV 21조*

1 프로젝트 개요

Figure 1: 프로젝트 개요

다양한 종류의 쓰레기(일반 쓰레기, 플라스틱, 종이, 유리 등 총 10가지)를 포함한 이미지 데이터셋을 활용했다. 데이터셋은 COCO 포맷의 바운딩 박스 정보(좌표, 카테고리)를 포함하고 있어, 학습 시 정확한 쓰레기 위치와 종류를 인식하도록 모델을 훈련시켰다. 모델의 출력값은 바운딩 박스 좌표, 카테고리, 그리고 신뢰도 점수(score)를 포함하며, 이 값을 기반으로 평가가 이루어졌다.

2 프로젝트 팀 구성 및 역할

이름	역할
김한얼	Construct pipeline, Code refactoring, Schedule management, Workload management, Model search
김보현	EDA, Model search, Induce Ultralytics, Induce Diffusion augmentation
김성주	Data feature analysis, Enhanced evaluation accuracy, Implemention data augmentation technique
윤남규	Research tools, Hypothesis test, Data curation & augmentation, Pipeline refatorization
정수현	Model search, Hypothesis test, Hyperparameter tuning, Data augmentation
허민석	Model search, Hypothesis test, Hyperparameter tuning, Data augmentation

Table 1: 팀원 역할

*haneol.kijm@gmail.com
†bhkim4550@gmail.com
‡comaoz1@naver.com
§mandudu6363@gmail.com
¶yynk2012@gmail.com
‖tig0601537@gmail.com

3 프로젝트 수행 절차 및 방법

3.1 Image EDA

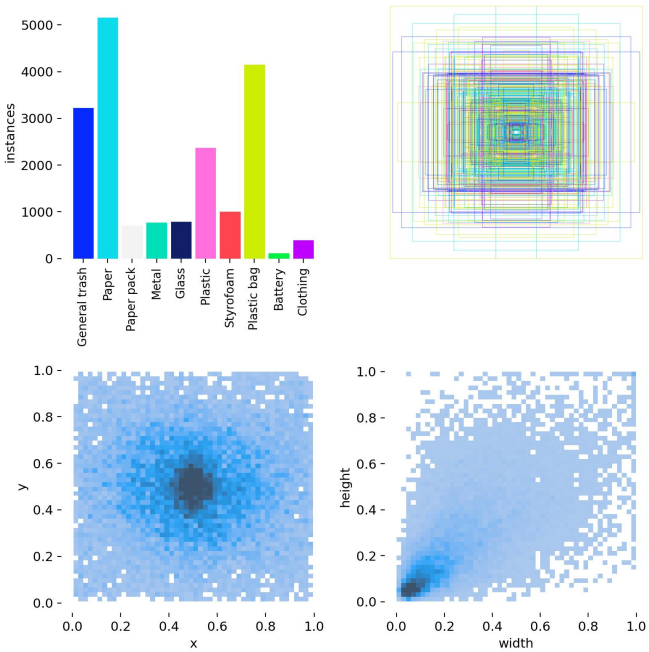

Figure 2: EDA

EDA를 통해 쓰레기 분류 프로젝트의 데이터 분포를 시각화했다. 쓰레기의 카테고리별 인스턴스 수를 보면 "General trash" "Plastic bag" 가장 많이 등장하는 카테고리이다. 각각 약 3500개와 5000개 이상의 인스턴스가 있다. "Glass", "Battery", "Clothing"과 같은 항목들은 비교적 적은 인스턴스 수를 가지고 있다. 다양한 바운딩 박스의 분포를 시각화한 결과, 다양한 크기와 위치의 바운딩 박스가 존재함을 나타낸다. 히트맵 (x, y 좌표 분포)은 바운딩 박스의 중심 좌표 (x, y)가 이미지 상에서 어떻게 분포되어 있는지를 보여준다. 중심은 이미지 중앙에 집중되어 있으며, 이는 대부분의 쓰레기 객체가 이미지의 중심에 위치해 있음을 보여준다. 히트맵 (width, height 분포)는 바운딩 박스의 너비와 높이 분포를 나타낸다. 대부분의 바운딩 박스는 작고, 너비와 높이가 0.2 이하인 경우가 많았다. 이는 객체 크기가 전체 이미지 크기에 비해 작은 경우가 많음을 나타낸다. 이 EDA 결과를 바탕으로 쓰레기 탐지 모델의 학습 과정에서 데이터의 불균형 문제를 해결하거나, 바운딩 박스 크기와 분포를 고려한 데이터 증강 전략 등을 세웠다.

3.2 협업 규칙

프로젝트 협업을 효율적으로 진행하기 위해, GitHub Flow와 커밋 컨벤션, GPU 사용 규칙 등의 협업 가이드를 따릅니다. 이를 통해 팀원 간의 협업을 원활하게 하고, 작업의 명확성을 높입니다.

GitHub Flow 활용 - **main** 브랜치에는 직접 `commit`하지 않습니다. - 필요한 기능 추가나 버그 수정이 있을 경우, GitHub **issue**를 생성합니다. - **feature-N** 브랜치(N: 생성된 이슈의 번호)를 생성하여 해당 브랜치에서 작업을 진행합니다. - 작업이 완료되면 `push`하고, **Pull Request**를 생성하여 **main** 브랜치에 **merge**합니다.

커밋 컨벤션 - Gitmoji 활용 - **Gitmoji**를 사용하여 커밋 메시지를 작성합니다. 이를 위해 **VSCode**의 **Gitmoji** 확장을 설치합니다. - 커밋 컨벤션 규칙을 엄격히 지키기보다는, **Gitmoji**를 사용해 협업의 일관성을 유지하는 데 의의를 둡니다.

GPU 사용 규칙 - 하루에 **GPU 20시간** 사용을 목표로 합니다. - 자기 전에 **GPU 서버**가 놀고 있는지 확인합니다. - 그날의 **GPU 서버** 사용자는 사다리타기를 통해 결정합니다.

공통 VSCode 확장 프로그램 - 프로젝트에서 공통으로 사용하는 **VSCode** 확장 프로그램은 **Git Graph**, **pylint**, **GitHub Pull Requests**입니다. **GitHub Pull Requests** 확장은 선택 사항입니다.

가설 설정과 검증 - 이틀에 하나씩 새로운 가설을 적극적으로 설정하고 검증합니다.

소통과 피드백 - 팀원 중 누구든 문제가 있을 경우, 즉시 말해서 해결합니다.

Issue와 PR 관리 - **Issue**와 **Pull Request**를 생성할 때, 구체적인 **label**을 사용합니다. - **Gitmoji+Name** 형식의 레이블을 사용해 **Issue**의 목적에 따라 구분합니다. 예: : `Bug`. - **Priority** 레이블을 사용해 **Issue**의 중요도를 **High**, **Medium**, **Low**로 구분합니다. - 가설 설정과 검증을 위해 `Hypothesis` 레이블을 추가해 사용합니다. 가능한 모든 작업을 **GitHub** 내에서 진행하는 것을 목표로 합니다.

3.3 Directory Descriptions

최상위 디렉토리: `level2-objectdetection-cv-21` 프로젝트의 루트 디렉토리로, 객체 탐지(Object Detection)와 관련된 다양한 스크립트와 설정 파일들이 포함되어 있습니다. 이 디렉토리는 하위 폴더 및 파일들을 통해 모델 학습, 데이터 전처리, 평가 등의 작업을 관리합니다.

`tools` 데이터 전처리와 평가에 유용한 여러 유틸리티 스크립트가 포함되어 있습니다. `pseudo_labeling` 폴더는 의사 라벨링(Pseudo-labeling)과 관련된 코드나 스크립트를 포함할 수 있으며, `check_image.py`는 데이터의 유효성 검사를 수행하여 손상된 이미지나 잘못된 형식을 확인합니다. `ensemble.py`는 다양한 모델의 예측 결과를 결합하는 앙상블 기법을 구현하며, `mAP_from_csv.py`는 예측 결과의 mAP(Mean Average Precision)를 계산하여 모델의 성능을 평가합니다.

`mmdetection(v2)` `mmdetection` 프레임워크를 활용한 객체 탐지 모델의 학습과 평가를 위한 코드를 포함하고 있습니다. `configs` 폴더는 다양한 실험 설정 파일을 저장하며, `inference.py`와 `inference_on_val_set.py`는 학습된 모델을 사용해 새로운 데이터나 검증 데이터셋에서 추론을 수행합니다. `testTTA.py`는 TTA(Test Time Augmentation)를 적용한 테스트를 실행하며, `trainer.py`는 모델의 학습을 관리하는 트레이너 스크립트로 학습 로직을 정의합니다.

`yolov11` YOLOv11 객체 탐지 모델과 관련된 코드가 포함된 폴더로, 다양한 파일들이 각각의 역할을 수행합니다. `cfg` 폴더는 모델의 하이퍼파라미터와 구조를 설정하는 파일들이 저장되어 있으며, `augmentation.py`는 데이터 증강을 위한 스크립트로 학습 데이터에 다양한 변환을 적용합니다. `convert.py`는 데이터 포맷 변환이나 모델 가중치 변환을 처리하며, `inference.py`는 YOLOv11 모델을 사용해 이미지 또는 비디오에서 객체를 탐지합니다. `split.py`는 데이터셋을 학습, 검증, 테스트 세트로 분할하고, `streamlit.py`는 Streamlit을 활용해 모델의 추론 결과를 시각화하는 웹 애플리케이션을 제공합니다. 마지막으로, `train.py`는 YOLOv11 모델의 학습을 수행하는 메인 스크립트로, 전체 학습 파이프라인을 실행합니다.

`README.md` 프로젝트의 개요와 사용법, 설치 방법 등을 설명하는 문서입니다. 개발자나 사용자가 프로젝트의 구조와 기능을 쉽게 이해할 수 있도록 안내하며, 프로젝트의 시작점이 되는 중요한 파일입니다.

```
level2-objectdetection-cv-21
|
|─── tools
|    |- pseudo_labeling
|    |- check_image.py
|    |- ensemble.py
|    └── mAP_from_csv.py
|─── mmdetection(v2)
|    |- configs
|    |- inference.py
|    |- inference_on_val_set.py
|    |- testTTA.py
```

```
|     └── trainer.py
|── yolov11
|     |- cfg
|     |- augmentation.py
|     |- convert.py
|     |- inference.py
|     |- split.py
|     |- streamlit.py
|     └── train.py
└──── README.md
```

4 Data Preprocessing and Augmentation

Diffusion Model을 활용한 합성 이미지 생성을 통한 Data Augmentation 기법을 사용했습니다. Object detection은 image classification보다 복잡한 주석 작업이 필요하며, 정확한 바운딩 박스 정보를 포함해야 한다. 새로운 데이터를 주석하는 대신, 기존 데이터를 변형하여 더 많은 학습 데이터를 생성하는 데이터 증강을 사용했다.

4.1 Diffusion Models

생성 기반 데이터 증강의 주요 이점은 다양성과 현실성이 높다는 점이다. 새로운 시각적 특징을 학습 데이터에 추가할 수 있으며, 이는 모델의 일반화 능력을 향상시킨다. 생성된 이미지에서 정확한 바운딩 박스 주석을 얻는 것은 어렵다. 이를 해결하기 위해 바운딩 박스를 미리 지정하고 해당 영역에 객체를 생성하는 접근 방식을 사용했다. 텍스트 기반 이미지 생성과 시각적 사전 정보(visual priors)를 결합해 다양한 스타일과 조건에서도 고품질의 바운딩 박스를 유지하며 이미지를 생성했다.

4.1.1 Contrallable Diffusion Model

Visual Prior Generator (비주얼 프라이어 생성기) 데이터셋에서 샘플링한 이미지에 HED 엣지 검출기(HED edge detector)를 사용하여 "비주얼 프라이어-주석 쌍"을 생성합니다. 이 과정에서는 원본 이미지의 중요한 구조적 특징을 유지하면서도, 모델 학습에 유용한 시각적 사전 정보(visual priors)를 제공합니다. 다른 엣지 검출 방법이나 세그멘테이션 마스크도 사용할 수 있으나, HED 엣지 검출기가 가장 효과적임이 입증되었습니다.

Prompt Constructor (프롬프트 생성기) 주석 정보에 따라 프롬프트를 생성하며, 기본 전략은 주석에 포함된 모든 카테고리 라벨을 결합하여 하나의 문장으로 구성하는 것입니다. 예를 들어, 이미지에 포함된 객체가 '플라스틱 병'과 '종이 박스'인 경우, "플라스틱 병과 종이 박스"라는 프롬프트를 생성합니다. 이를 통해 모델이 이미지 생성을 위한 명확한 조건을 설정할 수 있으며, 생성된 이미지의 품질과 주석 일관성을 높입니다.

Controllable Diffusion Model (제어 가능한 확산 모델) 합성 이미지와 주석 데이터를 생성하기 위해 확산 모델을 사용합니다. 미리 훈련된 확산 모델의 일부 파라미터는 고정된 상태로 유지되며, 나머지 파라미터는 주어진 프롬프트와 시각적 사전 정보에 따라 업데이트됩니다. 이 과정에서는 고품질의 바운딩 박스를 포함한 합성 이미지-주석 쌍이 생성됩니다. 또한, 다양한 스타일과 조명 조건에서도 일관성 있는 객체를 생성할 수 있도록 제어 가능한 확산 모델을 활용합니다.

Figure 3: 85번 이미지 - Original

Figure 4: 85번 이미지 - Augmented

Figure 5: 85번 이미지 - Visprior

Figure 6: 190번 이미지 - Original

Figure 7: 190번 이미지 - Augmented

Figure 8: 190번 이미지 - Visprior

Figure 9: 295번 이미지 - Original

Figure 10: 295번 이미지 - Augmented

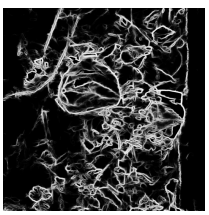

Figure 11: 295번 이미지 - Visprior

Figure 12: Diffusion Augmentation 예시 이미지들

Controllable Diffusion을 통해 생성한 Streamlit에서 사용할 수 있는 예시 이미지입니다. 테스트 용도로 3개의 이미지씩 배치했으며, 10개 이상의 이미지로 구성할 수도 있습니다. 1. 프롬프트 생성 방식의 문제점 발견 이번 실험에서는 blip_large 모델을 사용하여 Diffusion 프롬프트를 생성했습니다. 그러나 프롬프트가 제대로 동작하지 않는 듯한 문제가 발생했습니다. 프롬프트가 적용된 부분을 직접 확인했을 때, 지난번의 cat 방식에 비해 생성된 이미지의 퀄리티가 낮아졌다는 점이 관찰되었습니다. 이로 인해 기존의 방식으로 다시 회귀해야 할지에 대한 고민이 생겼습니다.

추가로, 10개의 클래스가 blip_large 모델이 생성할 수 있는 COCO 데이터셋의 80개 클래스에 포함되지 않는 것 같다는 의문이 들었습니다. 만약 이 가설이 맞다면, blip_large 모델이 실제로

생성할 수 있는 범위를 벗어난 프롬프트를 사용했을 가능성이 있습니다. 이 문제를 해결하기 위해서는 blip_large 모델에 대해 더 깊이 살펴볼 필요가 있겠지만, 현재 우선순위는 Diffusion 생성을 완료하는 것이기 때문에 당장은 원래의 cat 방식으로 다시 시도해보기로 결정했습니다.

2. a_prompt와 n_prompt 활용 여부 현재 실험에서 a_prompt와 n_prompt라는 추가적인 프롬프트 옵션이 제공되고 있습니다. 이 옵션들이 이미지 생성에 미칠 수 있는 잠재적 영향을 고려하여, 이를 활용할지 여부를 고민하고 있습니다. 따라서, 두 가지 방법을 모두 시도해보기로 했습니다. 이를 통해 생성된 이미지의 질과 다양성에 어떤 차이가 나타나는지 비교해볼 예정입니다.

3. 이미지 해상도 결정: 512x512 vs 1024x1024 현재 설정된 이미지 해상도는 512x512이며, 이를 1024x1024로 변경하는 것이 더 나을지에 대한 고민이 있습니다. 1024x1024 해상도는 더 높은 퀄리티의 이미지를 생성할 수 있지만, 생성 시간과 자원 소모가 증가하게 됩니다. 본 실험에서는 최종 결과의 품질이 중요한 만큼, 1024x1024로 생성하는 것이 바람직할 수 있다고 판단되지만, 실험의 효율성도 고려해야 합니다.

서버 접속 및 작업 준비 서버에 접속한 후, 앞으로 모든 명령어(Command)는 터미널(Terminal)에서 진행합니다. 이 과정에서 필요한 라이브러리 설치 및 데이터 다운로드, 압축 해제 등의 작업을 수행합니다.

gdown을 이용한 데이터셋 다운로드 `gdown` 라이브러리를 사용하여 "diffusion 처리된 데이터셋"을 Google Drive에서 다운로드합니다. `gdown`은 Google Drive의 "링크 공유"를 통해 파일에 접근하여 빠르게 다운로드할 수 있는 라이브러리입니다. - 다운로드 명령어: `gdown 1_BhqVp4UWXKnjzH8CYaRBm9oz4ZIDNmM` - 만약 `gdown` 명령어를 실행할 수 없다는 오류 메시지가 나타나면, `pip install gdown` 명령어를 통해 `gdown` 라이브러리를 설치합니다. - 정상적으로 다운로드가 완료되면, ".tar" 형식의 압축 파일이 서버에 저장됩니다.

서버에서 압축 풀기 작업 수행 다운로드한 ".tar" 파일의 압축을 풀기 위해 `tar` 명령어를 사용합니다. - 명령어: `tar -xvf diffusion_1172.tar -C ./diffusion_data` - `tar -xvf`: ".tar" 파일의 압축을 해제하는 명령어입니다. - `-C ./diffusion_data`: 압축을 해제한 파일들을 저장할 폴더 경로입니다. 사용자가 원하는 경로로 변경할 수 있습니다.

5 Ultralytics

YOLO-World Model YOLO-World는 Ultralytics YOLOv8을 기반으로 한 실시간 Open-Vocabulary Detection 모델로, 이미지 내에서 설명 텍스트를 기반으로 임의의 객체를 탐지할 수 있는 고급 접근 방식을 제공합니다. 기존 모델들이 가지고 있는 높은 연산 요구 사항을 크게 낮추면서도 경쟁력 있는 성능을 유지하여, 다양한 비전 기반 응용 프로그램에 적합한 다재다능한 도구로 자리매김하고 있습니다.

모델 개요 YOLO-World는 전통적인 Open-Vocabulary 탐지 모델들이 직면한 문제들을 해결합니다. 이러한 모델들은 주로 방대한 연산 자원을 필요로 하는 Transformer 기반의 접근 방식을 사용하며, 사전 정의된 객체 카테고리에 의존하는 경우가 많아 동적인 시나리오에서 유용성이 제한됩니다. YOLO-World는 YOLOv8 프레임워크를 확장하여 Open-Vocabulary Detection 기능을 통합하고, 비전-언어 모델링과 대규모 데이터셋 사전 학습을 통해 제로샷(zero-shot) 상황에서도 다양한 객체를 높은 효율로 탐지할 수 있도록 설계되었습니다.

주요 기능 1. **실시간 솔루션**: YOLO-World는 CNN의 빠른 연산 속도를 활용하여 실시간 Open-Vocabulary Detection을 구현하며, 즉각적인 결과가 필요한 산업 분야에 적합합니다. 2. **효율성과 성능**: 모델은 연산 및 자원 요구 사항을 줄이면서도 우수한 성능을 유지하며, SAM(Segment Anything Model)과 같은 모델 대비 실시간 응용에 적합한 대안을 제공합니다. 3. **오프라인 Vocabulary를 사용한 추론**: "prompt-then-detect" 전략을 도입하여, 사용자 정의 프롬프트를 사전에 계산하고 저장된 오프라인 임베딩을 사용하여 효율을 높입니다. 4. **YOLOv8 기반 성능**: YOLO-World는 최신 실시간 객체 탐지 기술을 활용하여 YOLOv8의 정확성과 속도를 기반으로 Open-Vocabulary Detection을 수행합니다. 5. **벤치마크 우수성**: YOLO-World는 MDETR, GLIP 시리즈와 같은 기존 Open-Vocabulary 탐지기보다 속도와 효율성 면에서 우수한 성능을 보여주며, NVIDIA V100 GPU 단일 환경에서도 뛰어난 성능을 발휘합니다.

사용 가능한 모델, 지원되는 작업 및 작동 모드 YOLO-World는 다양한 사전 학습된 가중치를 제공하며, 객체 탐지 작업을 지원합니다. 각 모델은 추론, 검증, 학습, 내보내기 등 다양한 작동 모드와 호환되며, 이를 통해 유연한 적용이 가능합니다.

Zero-shot Transfer 성능 COCO 데이터셋에서의 zero-shot 성능은 다음과 같습니다. 작은 모델(yolov8s-world)에서 큰 모델(yolov8x-world)로 갈수록 성능이 향상되며, mAP와 mAP50, mAP75 모두에서 일관된 성능 개선을 보여줍니다.

프롬프트 설정 기능 YOLO-World는 사용자 정의 프롬프트를 통해 동적으로 클래스를 지정할 수 있습니다. 이를 통해 모델을 새로운 도메인이나 특정 작업에 적응시킬 수 있으며, 재학습 없이 탐지 정확도와 효율성을 높일 수 있습니다. 예를 들어, 'person'과 'bus'만 탐지해야 하는 경우, 해당 클래스를 프롬프트로 지정하여 모델의 탐지 성능을 향상시킬 수 있습니다.

모델 사용자화와 효율성 모델에 사용자 정의 클래스를 설정하고 저장함으로써, 특정 응용에 최적화된 YOLO-World 모델을 생성할 수 있습니다. 이렇게 저장된 모델은 추가 조정 없이도 지정된 클래스만 탐지하도록 최적화되어 있으며, 탐지 성능과 효율성을 더욱 향상시킵니다.

YOLO-World 학습 YOLO-World 모델은 간편한 Python API와 CLI 명령을 제공하여 사용자 정의 데이터셋에서 손쉽게 학습할 수 있습니다. 모델 학습 시 사전 학습된 YOLOv8 기반의 다양한 가중치를 사용할 수 있으며, 효율적인 학습과 추론을 지원합니다.

RT-DETR 개요 RT-DETR(Real-Time Detection Transformer)는 Baidu에서 개발한 최첨단 실시간 객체 탐지 모델로, Vision Transformer 기반의 아키텍처를 사용합니다. 이 모델은 DETR(Non-Maximum Suppression이 필요 없는 프레임워크)의 개념을 기반으로 하며, conv 기반의 백본과 효율적인 하이브리드 인코더를 도입하여 실시간 성능을 유지하면서도 높은 정확도를 제공합니다. RT-DETR은 intra-scale 상호작용과 cross-scale 융합을 분리하여 멀티스케일 특징을 효율적으로 처리합니다.

주요 특징 1. **효율적인 하이브리드 인코더**: RT-DETR의 하이브리드 인코더는 멀티스케일 특징을 처리할 때 intra-scale 상호작용과 cross-scale 융합을 분리하여 연산 비용을 줄이면서도 실시간 객체 탐지를 가능하게 합니다. 2. **IoU-기반 쿼리 선택**: 이 모델은 IoU-기반 쿼리 선택을 통해 객체 쿼리 초기화를 개선합니다. 이를 통해 장면에서 가장 관련성이 높은 객체에 집중할 수 있어 탐지 정확도가 향상됩니다. 3. **적응형 추론 속도**: RT-DETR은 다양한 디코더 레이어를 사용해 추론 속도를 조절할 수 있으며, 이를 위해 재학습이 필요하지 않습니다. 이러한 유연성 덕분에 다양한 실시간 객체 탐지 시나리오에 맞게 성능을 조정할 수 있습니다.

사전 학습된 모델 RT-DETR은 Ultralytics Python API를 통해 제공되는 PaddlePaddle 기반의 사전 학습된 모델을 지원합니다. RT-DETR-L은 COCO val2017에서 53.0% AP를 달성하며 T4 GPU에서 114 FPS의 속도를 제공하고, RT-DETR-X는 54.8

RT-DETR 사용 예제 RT-DETR 모델을 사용하여 학습 및 추론을 쉽게 수행할 수 있습니다. 예를 들어, Python 코드를 사용해 COCO 데이터셋으로 사전 학습된 RT-DETR-L 모델을 불러와 학습하고, 특정 이미지에서 추론을 실행할 수 있습니다. Ultralytics API를 통해 간단한 명령으로 학습, 검증, 예측 및 내보내기 작업을 수행할 수 있습니다.

RT-DETR의 장점 RT-DETR은 효율적인 하이브리드 인코더와 IoU-기반 쿼리 선택 기능을 통해 높은 정확도를 유지하면서도 연산 비용을 크게 줄일 수 있습니다. 디코더 레이어를 사용해 추론 속도를 조절할 수 있는 유일한 모델로, 재학습 없이도 유연한 추론 속도 조정이 가능하여 다양한 실시간 객체 탐지 응용에 유리합니다. 특히, CUDA 및 TensorRT와 같은 가속화된 백엔드에서 우수한 성능을 발휘합니다.

적응형 추론 속도 지원 RT-DETR은 디코더 레이어를 사용하여 다양한 실시간 객체 탐지 작업에 맞게 추론 속도를 유연하게 조정할 수 있습니다. 이는 낮은 정밀도 요구사항을 위해 빠른 처리를 필요로 하거나, 더 높은 정확도를 위해 느리지만 정확한 탐지를 요구할 때 성능을 조절할 수 있게 해줍니다.

Ultralytics 모드와의 호환성 RT-DETR 모델은 Ultralytics의 학습, 검증, 예측, 내보내기 모드와 모두 호환됩니다. 이를 통해 객체 탐지 솔루션을 개발하고 배포하는 데 필요한 모든 워크플로우를 포괄적으로 지원합니다.

6 Pseudo Labeling

Background 이 코드는 Object Detection에서 생성된 예측 파일의 데이터를 처리하는 작업을 수행합니다. 예측 데이터는 "class", "p(가능성, 확률)", "BBox 꼭짓점들의 좌표"를 포함한 형식으로 제공됩니다. 예측의 신뢰도를 나타내는 p 값이 높을수록 모델은 해당 BBox의 존재 가능성이 높다고 판단합니다.

Paper key point: Basic Pseudo Labeling 기본적인 Pseudo Labeling 접근 방식에서는 모델이 생성한 다수의 BBox 중 가장 신뢰할 수 있는 값(Argmax)을 True BBox로 정의하고, 나머지 BBox는 False BBox로 분류합니다. 이러한 True BBox를 반복적으로 재학습하여 모델의 정확도를 점진적으로 향상시키는 것이 논문의 핵심 아이디어입니다.

Paper key point: Confidence Pseudo Labeling 기본 Pseudo Labeling의 한계를 극복하기 위해 Confidence(모델의 신뢰도)를 도입하여 True BBox 중 신뢰도가 설정된 임계값(threshold) 이상인 경우만을 사용합니다. 이를 통해 더 높은 정확도를 달성할 수 있으며, 모델이 올바르다고 판단된 예측에 집중할 수 있습니다.

PredictionStringProcessor 클래스 이 클래스는 모델의 예측 데이터를 CSV 파일에서 읽어들여 Confidence 기준에 따라 필터링하고, 결과를 저장하는 역할을 수행합니다. - `csv_path`: 처리할 CSV 파일의 경로를 지정합니다. - `save_path`: 처리된 결과를 저장할 경로를 설정합니다. - `conf_thr`: 예측을 필터링할 때 사용할 신뢰도 임계값을 설정합니다.

기능 설명 1. **예측 데이터 파싱** - `parse_prediction_string()` 메서드는 예측 문자열을 개별 BBox 정보로 변환합니다. 각 BBox는 "category", "confidence", "bbox" 정보로 구성됩니다.

2. **필터링 및 형식 변환** - `process_single_row()` 메서드는 각 예측 데이터에서 신뢰도가 `conf_thr` 이상인 BBox만 남기고,

이를 필터링된 문자열로 변환합니다. 필터링을 통과하지 못한 예측은 빈 문자열로 반환됩니다.

3. **CSV 파일 처리** - `run()`메서드는 CSV 파일의 각 행을 순회하면서 예측 문자열을 처리하고, 필터링된 결과를 새로운 CSV 파일로 저장합니다. 처리 완료 후, 전체 행의 수, 예측이 있는 행의 수, 필터링된 행의 수 등의 통계를 제공합니다.

실행 방식 1. **명령줄 인자 처리** - `parse_args()`함수는 명령줄에서 입력받은 인자들을 처리합니다. `--input_path`, `--output_path`, `--confidence_threshold` 등을 설정할 수 있습니다.

2. **처리 작업 수행** - `main()`함수는 인자를 파싱하고, `PredictionStringProcessor` 인스턴스를 생성하여 파일을 처리합니다. 신뢰도 임계값을 사용하여 데이터를 필터링하고, 결과를 CSV 파일로 저장합니다.

3. **결과 출력** - 처리 과정이 완료되면 총 처리된 행의 수, 예측이 남아 있는 행의 수, 필터링된 행의 수를 출력하며, 결과 파일의 저장 경로를 안내합니다.

7 Ensemble

앙상블 코드 개요 이 코드는 다양한 모델의 예측 결과를 결합하여 성능을 개선하는 앙상블 기법을 구현한 것입니다. 여러 개의 submission 파일을 입력받아, 이미지별로 예측된 바운딩 박스(BBox), 점수, 라벨을 추출하여 NMS(Non-Maximum Suppression), Soft-NMS, WBF(Weighted Boxes Fusion) 등의 앙상블 기법을 적용해 최종 결과를 생성합니다.

앙상블 진행 방식 1. **데이터 준비** - 여러 모델의 예측 결과가 담긴 CSV 파일을 `--csv-files` 인자를 통해 입력받습니다. 각 CSV 파일에는 이미지 ID와 예측된 BBox 정보가 포함되어 있습니다. - CSV 파일들을 하나의 데이터프레임으로 병합하여, 이미지 ID를 기준으로 그룹화하여 처리합니다.

2. **예측 데이터 파싱** - `parse_prediction_string()`함수는 각 예측 문자열을 파싱하여 BBox, 점수, 라벨 리스트로 변환합니다. 이를 통해 각 모델의 예측 결과를 통일된 형식으로 정리합니다. - BBox 좌표는 0에서 1 사이의 값으로 정규화하여 처리합니다.

3. **NMS(Non-Maximum Suppression)**- NMS는 겹치는 BBox를 제거하여 중복을 줄이는 기법입니다. `apply_nms` 인자가 활성화된 경우 NMS를 수행하며, `nms_thr` 인자는 NMS 임계값을 설정합니다. - 모든 모델의 예측 결과를 바탕으로 NMS를 적용하여 겹치는 BBox를 제거하고 최적의 예측을 남깁니다.

4. **Soft-NMS** - NMS 대신 Soft-NMS를 적용할 수도 있습니다. `apply_soft_nms` 인자가 활성화되면 Soft-NMS가 사용됩니다. - Soft-NMS는 겹치는 BBox의 점수를 감소시켜 예측을 조정하는 방식으로, 임계값을 넘지 않는 BBox는 제거되지 않습니다.

5. **WBF(Weighted Boxes Fusion)**- `apply_wbf` 인자가 활성화되면 WBF를 수행합니다. WBF는 각 모델의 예측을 가중치로 결합하여 새로운 BBox를 생성하는 기법입니다. - WBF는 NMS 또는 Soft-NMS와 함께 사용될 수 있으며, IoU 임계값 `wbf_thr`를 통해 적용 범위를 조정할 수 있습니다.

6. **결과 생성 및 저장** - 최종 BBox, 점수, 라벨 정보를 `format_prediction_string()`함수로 예측 문자열 형식으로 변환합니다. - 변환된 결과를 데이터프레임에 저장하고, 최종적으로 `to_csv()`함수를 통해 CSV 파일로 출력합니다. - 결과 파일명은 NMS와 WBF의 사용 여부 및 임계값에 따라 자동으로 지정되며, 앙상블 완료 후 저장된 파일의 경로를 출력합니다.

코드의 주요 기능 - **NMS, Soft-NMS, WBF를 통한 앙상블 선택적 수행**: 사용자는 NMS, Soft-NMS, WBF 중 하나 또는 복합적으로 선택하여 앙상블을 수행할 수 있습니다. - **BBox 정규화 및 역정규화**: 이미지 크기에 맞게 BBox 좌표를 정규화하여 모델 간의 일관성을 유지하며, 최종 결과에서는 원래의 이미지 크기로 변환하여 저장합니다. - **다양한 하이퍼파라미터 설정 가능**: `nms_thr`, `wbf_thr`와 같은 하이퍼파라미터를 사용자 설정으로 조정할 수 있어 유연한 앙상블 적용이 가능합니다. - **앙상블 결과의 자동 저장 및 로그 출력**: 결과 파일명은 설정된 앙상블 기법과 임계값을 기반으로 자동 생성되며, 로그를 통해 적용된 앙상블 기법 및 파라미터를 확인할 수 있습니다.

8 Multi-Crop

Multi-Crop 방식 개요 Multi-Crop 방식은 이미지를 여러 부분으로 나누어, 각 부분에 대해 개별적으로 Object Detection을 수행하는 기법입니다. 원본 이미지를 다양한 크기로 잘라서 데이터셋을 확장하고, 작은 객체의 탐지 성능을 향상시키는 데 목적이 있습니다. 이 방식은 이미지를 4등분하여 각 영역에 대해 별도로 Bounding Box(BBox) 주석을 업데이트하고, 새로운 데이터셋을 생성하는 과정으로 이루어집니다.

Multi-Crop 과정 1. **이미지와 주석 파일 불러오기** - 원본 이미지와 주석(annotation) 파일을 읽어옵니다. 주석 파일은 COCO 형식으로 제공되며, 각 이미지에 대한 BBox 정보를 포함하고 있습니다.

2. **이미지 분할** - 각 이미지를 4등분하여, 상단 왼쪽, 상단 오른쪽, 하단 왼쪽, 하단 오른쪽의 4개의 영역으로 나눕니다. - 분할된 각 영역에 대해 새로운 이미지 ID와 함께 별도의 이미지를 생성합니다.

3. **Bounding Box 업데이트** - 각 분할된 영역에 대해 기존 BBox가 겹치는지를 확인하여, 겹치는 BBox만을 새로운 주석 정보로 업데이트합니다. - BBox 좌표는 각 분할된 영역의 기준으로 조정됩니다. 예를 들어, 상단 왼쪽 영역의 경우, BBox 좌표는 원래의 좌표에서 왼쪽 상단 모서리의 위치를 기준으로 이동합니다.

4. **새로운 이미지와 주석 데이터 생성** - 분할된 이미지와 업데이트된 BBox 정보를 사용하여 새로운 이미지 데이터와 주석

데이터를 생성합니다. - 새로운 주석 정보는 각 부분 이미지에 대해 고유한 ID를 할당하며, 원본 데이터와의 혼동을 방지합니다.

Multi-Crop 방식의 장점 - **작은 객체 탐지 성능 향상**: 이미지를 분할하여 개별 영역에 대해 BBox를 세밀하게 조정함으로써, 작은 객체의 탐지 성능을 높일 수 있습니다. - **데이터 확장**: 하나의 이미지에서 여러 부분 이미지를 생성함으로써, 데이터셋의 크기를 인위적으로 확장할 수 있습니다. - **효과적인 주석 관리**: 각 분할된 이미지에 대해 별도의 주석을 생성함으로써, 원본 데이터와 분할된 데이터의 혼동을 최소화할 수 있습니다.

최종 데이터셋 저장 - 모든 분할 이미지와 주석 정보를 JSON 형식으로 저장하며, 원본 주석 파일을 기반으로 새롭게 생성된 데이터를 포함합니다. - 생성된 데이터셋은 기존 COCO 형식과 호환되며, 학습 시 동일한 형식으로 사용할 수 있습니다.

9 프로젝트 수행 결과

프로젝트 수행의 성과 이번 프로젝트에서는 쓰레기 분류를 위한 Object Detection 모델을 개발하면서, 여러 가지 데이터 증강 기법과 앙상블 방법을 적용하였습니다. 특히, Diffusion 모델을 활용한 합성 이미지 생성을 통해 데이터의 다양성을 높였으며, Multi-Crop 기법을 적용해 작은 객체의 탐지 성능을 개선했습니다. 그 결과, 모델의 성능이 꾸준히 향상되었으며, 최종적으로 평균 정확도(mAP) 측면에서 유의미한 성능 향상을 달성할 수 있었습니다.

개선된 협업과 워크플로우의 효과 GitHub Flow를 기반으로 한 협업 규칙을 적용하여 팀 내 협업의 일관성을 유지할 수 있었습니다. 각 팀원이 담당한 역할에 맞게 GitHub Issue와 Pull Request를 적극적으로 활용했고, 이를 통해 실험 결과와 코드 변경 사항을 명확히 관리할 수 있었습니다. 또한, 팀원들이 스스로 새로운 가설을 설정하고 검증하는 문화를 형성하여 협업 효율성을 극대화했습니다. 슬랙과 노션을 연동하여 실시간 피드백과 실험 기록을 관리한 점도 프로젝트 성공에 큰 기여를 했습니다.

Diffusion 모델과 데이터 증강 실험의 결과 Diffusion 모델을 이용한 합성 이미지 생성은 데이터 증강의 새로운 시도로, 모델의 일반화 성능을 향상시키기 위한 실험이었습니다. 이 과정에서 프롬프트 생성 방식의 한계를 발견하고, blip_large 모델의 적용 범위에 대한 문제를 해결하려는 노력이 있었습니다. Diffusion 모델로 생성된 이미지가 작은 객체를 효과적으로 표현하지 못하거나 테스트 이미지와 일관성이 부족한 문제가 발생했지만, 이를 통해 기존 데이터 증강 방식의 개선 방향을 모색할 수 있었습니다.

Multi-Crop 기법의 장점과 효과 Multi-Crop 기법을 통해 이미지를 4등분하여 데이터셋을 확장함으로써, 작은 객체 탐지 성능을 크게 향상시킬 수 있었습니다. 원본 이미지에서 여러 부분 이미지를 생성하고, 개별적으로 Bounding Box를 조정함으로써 작은 객체에 대한 모델의 인식 능력을 높였습니다. 이를 통해 데이터의 다양성과 학습 효과를 극대화할 수 있었습니다.

Pseudo Labeling 기법 적용의 이점 Pseudo Labeling 기법은 모델의 신뢰도에 따라 예측된 BBox를 필터링하는 방식으로, 예측의 정확도를 높이는 데 기여했습니다. Confidence Pseudo Labeling을 도입함으로써 신뢰도가 높은 BBox만을 True BBox로 사용하여 모델을 학습시켰으며, 반복적인 학습을 통해 점진적인 성능 향상을 이룰 수 있었습니다. 이를 통해 모델의 예측 안정성과 신뢰도를 크게 개선했습니다.

앙상블 기법을 통한 최종 성능 향상 여러 모델의 예측 결과를 결합하는 NMS, Soft-NMS, WBF 앙상블 기법을 적용하여, 단일 모델보다 향상된 성능을 얻을 수 있었습니다. 특히, 다양한 모델의 결과를 가중치로 결합해 새로운 BBox를 생성하는 WBF 기법은 최종 성능 향상에 큰 기여를 했습니다. 이 과정을 통해 모델의 일반화 성능을 높이고, 예측의 안정성을 강화했습니다.

한계와 교훈 프로젝트 진행 중에 여러 가지 한계도 경험했습니다. Diffusion 모델을 활용한 데이터 증강이 기대만큼 성능을 향상시키지 못한 점, 그리고 실험 시간과 자원 부족으로 인해 모든 계획을 실행하지 못한 점 등이 있었습니다. 이러한 한계를 통해, 실험의 우선순위를 명확히 하고, 가설 설정과 EDA를 더 신중하게 접근해야 한다는 교훈을 얻었습니다.

다음 프로젝트에 대한 방향성 이번 프로젝트를 통해 얻은 교훈을 바탕으로, 다음 프로젝트에서는 데이터 분석과 가설 설정을 더 체계적으로 수행할 계획입니다. 또한, 효율적인 협업을 위해 GitHub Issue와 PR을 더 적극적으로 활용하고, 노션과 슬랙을 연동하여 실험 기록을 체계적으로 관리할 예정입니다. 향후 프로젝트에서도 모델의 성능 향상과 협업의 효율성을 동시에 달성할 수 있는 방안을 지속적으로 모색할 것입니다.

개인 회고록

10 개인회고

10.1 김한얼

저는 기본적으로 이 팀에서 팀장 역할을 맡아서 진행했습니다.

나는 내 학습 목표 달성을 위해 무엇을 어떻게 했는가? 저는 학습 목표 달성을 위해 팀을 구성하고 팀장 역할을 맡아 프로젝트를 진행했습니다. 전체적인 워크플로우와 목표를 설정하고, 매일 팀원들에게 필요한 작업을 할당하였으며, 팀원들이 맡은 일을 제대로 수행하는지 점검하는 역할을 맡아 4주간 프로젝트를 진

행했습니다. 또한, 기본적인 뼈대가 되는 코드는 제가 작성하여 프로젝트의 기초를 마련했습니다.

모델 개선 방식 가설을 설정하고 이를 검증 및 개선하는 방식으로 실험을 진행했습니다. 특히 빠른 검증을 위해 작은 모델과 큰 모델을 모두 실험에 적용하였습니다. - 온라인 데이터 증강으로 heavy한 증강 기법을 적용하여, 작은 모델과 큰 모델 모두에서 성능 향상을 얻었습니다. - 오프라인 데이터 증강을 위해 diffusion 모델을 사용하여 이미지를 새롭게 생성했지만, 작은 객체나 중간 크기 객체를 잘 생성하지 못해 성능 하락을 초래했습니다. - 모델의 다양한 파라미터 실험을 위해 Bayesian Hyperband 기법을 사용하여 하이퍼파라미터 튜닝을 진행했습니다. - 다양한 모델 앙상블을 위해 mmdetection 라이브러리와 ultralytics 라이브러리를 활용했습니다. - Transformer 기반 모델과 CNN 기반 모델을 모두 사용하여 성능을 비교했습니다. - 최종 단계에서 TTA(Test Time Augmentation)를 통해 성능 향상을 얻었습니다. - 마지막으로, 앙상블을 통해 성능을 약간 개선했으나, 두 모델의 성능 차이가 클 경우 앙상블 적용이 어려운 점을 깨달았습니다.

행동의 결과로 달성한 지점 및 얻은 깨달음 이번 프로젝트에서는 diffusion 모델이 주요 주제였으며, 이를 통해 다양한 새로운 결과들을 실험하고 적용해 볼 수 있는 기회가 되었습니다. 덕분에 최신 논문에서 다루는 기술과 개념을 이해하는 데 큰 도움이 되었습니다. 또한, 이런 모델을 사용하는 데 있어서는 긴 시간과 많은 리소스가 필요하다는 점도 깨달았습니다.

새롭게 시도한 변화와 그 효과 이번 프로젝트에서는 이전 프로젝트보다 GitHub의 이슈와 PR을 중점적으로 활용하였습니다. 팀원들도 이를 잘 따르면서, 다른 팀들에 비해 더 효율적으로 협업을 진행할 수 있었습니다. 이를 통해 협업의 효율성을 높였고, 팀의 전체적인 진행 상황을 명확하게 파악할 수 있었습니다.

마주한 한계와 아쉬웠던 점 모델 측면에서는 diffusion 모델이 많은 시간과 서버 리소스를 소모하는 한계가 있었습니다. 더 넉넉한 기간을 확보했을 때 이 기법을 더 시도해 보고 싶습니다. 팀적으로는 세 가지 아쉬움이 있었습니다. 1. 체계적인 워크플로우의 부족: 즉흥적으로 매일 필요한 작업을 인식하고 분배하다 보니, 작업에 명확한 데드라인이 설정되지 않았습니다. 이로 인해 작업 진행 속도가 빠르지 않았습니다. 2. 이슈와 PR 활용 부족: GitHub의 이슈와 PR을 적극적으로 활용했으나, 일부 팀원은 이슈를 작성한 후 팀원들에게 알리지 않아 이슈가 묻히는 경우가 발생했습니다. 3. 내용 정리의 비체계성: Google Docs를 사용해 실험 결과와 회의록을 정리했지만, 정리가 체계적이지 않았습니다.

한계와 교훈을 바탕으로 다음 프로젝트에서 시도해볼 것 다음 프로젝트에서는 체계적인 워크플로우와 기한 설정을 토대로 작업을 진행하려고 합니다. 특히 다음 프로젝트는 기간이 2주밖에 되지 않아 이러한 방법론을 적용하기에 좋은 기회라고 생각합니다. 1. 자료 조사 (1-2일) 2. 조사한 자료에 대한 논의 (자료 조사 이후 저녁 피어 세션) 3. 자료를 토대로 한 코딩 (2일) 4. 코딩을 기반으로 한 실험 (1-2일) 5. 1-4번 반복

또한, 팀원들이 작업을 명확하게 알 수 있도록 Slack과 GitHub를 연동해, 이슈와 PR을 Slack에서 확인할 수 있도록 변경했습니다. 프로젝트 2 말미에 이 기능을 적용했으며, 이를 통해 토론도 GitHub 이슈에서 진행할 수 있어 매우 만족도가 높습니다. 추가로, Notion도 Slack이나 GitHub와 연동하여 실험 내용을 체계적으로 정리하고자 합니다.

10.2 김보현

나는 내 학습 목표 달성을 위해 무엇을 어떻게 했는가? Ultralytics의 공식 문서를 읽고 라이브러리를 공부하여 프로젝트에 적용했습니다. Object Detection 작업에서 라이브러리 학습의 중요성을 깨닫게 되었으며, Ultralytics 라이브러리가 제공하는 다양한 기능, 예를 들어 streamlit, wandb 연동 등을 최대한 활용하고자 노력했습니다. 주요 모델로는 one-stage detector를 주로 사용하여 프로젝트를 진행했습니다.

나는 어떤 방식으로 모델을 개선했는가? YOLO 시리즈의 yolov11x, yolov11n, 그리고 YOLOWorld의 yolov8x 모델과 DETR 기반의 rt-detrx, rt-detrn 등 다양한 one-stage detector를 사용했습니다. YOLO 모델은 평균적으로 약 4시간, rt-detr 모델은 약 12시간 정도의 학습 시간이 필요했습니다. 효율적인 모델 학습과 검증을 위해 에폭 수를 조정하고 체크포인트에서 성능이 가장 좋은 `best.pt` 파일을 활용하는 방법을 도입했습니다. 이를 통해 최적의 모델 성능을 달성할 수 있었습니다.

내가 한 행동의 결과로 어떤 지점을 달성하고, 어떤 깨달음을 얻었는가? Ultralytics 라이브러리를 통해 다양한 모델을 시도하면서 Object Detection 작업에서 라이브러리의 기능을 최대한 활용하는 것이 얼마나 중요한지 깨닫게 되었습니다. 또한, 각 모델의 학습 시간과 에폭 설정이 성능에 미치는 영향을 깊이 이해하게 되었으며, 체크포인트 활용을 통해 학습 과정을 최적화할 수 있었습니다. 이를 통해 효율적인 모델 개발과 성능 개선에 대한 중요한 인사이트를 얻을 수 있었습니다.

전과 비교하여 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가? 이번 프로젝트에서는 이전보다 딥러닝 기반 데이터 증강 기법을 적극적으로 시도했습니다. 특히, diffusion 모델을 활용한 오프라인 데이터 증강을 통해 새로운 이미지를 생성해보았으나, 생성된 이미지가 작은 객체를 효과적으로 표현하지 못했고, 테스트 데이터와의 일관성이 부족하여 성능이 향상되지 않았습니다. 협업 측면에서는 Slack과 GitHub를 연동하여, 팀원 간의 가설과 생각을 공유하는 데 있어 더 용이하게 하였습니다. GitHub에 이

슈를 적극적으로 기록함으로써, 팀원들이 서로의 작업을 명확히 이해할 수 있도록 개선했습니다.

마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가? 시간 부족 문제는 여전히 큰 한계로 다가왔습니다. 특히, diffusion 모델과 같은 복잡한 기법을 적용하는 데 필요한 충분한 시간을 확보하지 못해 성능 개선에 한계가 있었습니다. 또한, 코딩 실력 부족을 실감했으며, 특히 inference 코드 작성이나 데이터 증강과 같은 베이스라인 코드를 구현하는 데 시간이 예상보다 더 걸렸습니다. 이에 따라 초반에 시간 분배의 중요성을 더욱 절실히 느꼈습니다.

한계/교훈을 바탕으로 다음 프로젝트에서 시도해볼 것은 무엇인가? 다음 프로젝트에서는 강의를 통해 베이스라인 코드의 리뷰를 빠르게 진행하여 팀의 베이스라인 코드 작성 시간을 단축하고, 코딩 실력을 향상시키는 데 집중할 것입니다. 또한, 협업 환경을 최적화하기 위해 노력할 예정입니다. 예를 들어, 노션, GitHub, Slack을 연동하여 협업의 효율성을 높이고, 가설과 지식 공유를 체계적으로 관리할 계획입니다. 이를 통해 각 팀원이 자신의 역할을 명확히 이해하고, 더욱 효율적인 협업을 달성할 수 있을 것으로 기대됩니다.

10.3 허민석

나는 내 학습 목표 달성을 위해 무엇을 어떻게 했는가? 학습 목표를 달성하기 위해 여러 가지 가설을 세우고, 이를 검증하기 위한 다양한 실험을 진행하였습니다. 실험을 통해 가설의 타당성을 평가하고, 모델의 성능을 향상시키기 위해 꾸준히 노력했습니다.

나는 어떤 방식으로 모델을 개선했는가? 모델 개선을 위해 다양한 데이터 증강(augmentation) 기법을 추가하거나 새로운 모델을 도입하는 방식으로 접근했습니다. 각기 다른 증강 기법이 모델의 성능에 미치는 영향을 평가하면서, 성능 향상을 위한 최적의 방법을 찾고자 했습니다.

내가 한 행동의 결과로 어떤 지점을 달성하고, 어떤 깨달음을 얻었는가? 모델을 점진적으로 개선하면서 성능이 오르거나 내리는 변화를 경험했습니다. 이를 통해 단순히 실험 결과에 의존하기보다는, 좀 더 근거 있는 가설을 세우고 추가적인 실험을 진행할 필요가 있음을 깨달았습니다. 예를 들어, 구체적인 EDA를 통해 작은 객체(small object)를 탐지하지 못하는 문제를 파악하고, 이를 해결하기 위한 수정 실험을 진행했어야 했지만, 단순히 wandb에서 보이는 성능 변화만을 기준으로 판단한 점이 아쉬웠습니다. 추가적인 실험에 대한 근거가 부족했던 것을 반성하게 되었습니다.

전과 비교하여 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가? 이전과 달리, 이번 프로젝트에서는 단순히 주어진 실험을 반복하는 대신, 문제 해결에 대해 스스로 고민하고 분석할 수 있는 능력이 향상되었습니다. 어떤 부분에서 문제가 발생했는지, 이를 해결하기 위해 어떤 방법을 사용할 수 있는지에 대한 지식이 넓어졌으며, 이러한 경험은 향후 프로젝트에서도 큰 도움이 될 것이라고 생각합니다.

마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가? 무겁고 큰 모델이 더 나은 성능을 보일 것이라는 가정, 특정 증강 기법을 수정하면 성능이 향상될 것이라는 생각 등 모델 구조와 특성에 대한 깊이 있는 이해 없이 실험을 진행한 점이 아쉬웠습니다. 모델의 내부 구조와 동작 원리를 충분히 파악하지 못하고 실험을 진행한 것이 성능 향상에 제한적인 결과를 초래했습니다.

한계/교훈을 바탕으로 다음 프로젝트에서 시도해볼 것은 무엇인가? 다음 프로젝트에서는 좀 더 진취적인 마인드를 가지고 접근하고자 합니다. 단순히 실험을 돌리는 것에 그치지 않고, 전체적인 파이프라인을 구성해 보거나, 의미 있는 가설을 세워 꼬리에 꼬리를 무는 형태의 실험을 진행해 보고 싶습니다. 이를 통해 더 깊이 있는 분석과 실험을 수행할 수 있을 것입니다.

10.4 윤남규

나는 내 학습 목표 달성을 위해 무엇을 했는가? 프로젝트 진행 중 이미지에 대해 멀티 크롭(Multi-crop)을 수행하는 방식을 제안하였고, 작은 바운딩 박스(Small BBox)에 대해서는 클렌징(Cleansing)과 초해상도(SR, Super Resolution)를 적용하는 방법을 시도했습니다. 자료 조사 과정에서 다양한 오픈 소스(open source)를 읽어보는 것에 흥미가 있음을 깨달았으며, "이 기능을 왜 이렇게 구현했을까? 다른 방법은 없을까?"와 같은 질문을 스스로 던지고 해답을 찾아가는 과정이 즐거웠습니다.

나는 어떤 방식으로 모델을 개선했는가? 이전 프로젝트에서 부족했던 점을 보완하기 위해, 관련된 배경 지식을 확장하는 데 집중했습니다. 이전 기수의 프로젝트 결과물을 읽고, 필요하다고 생각되는 내용을 선별하여 팀원들에게 공유함으로써 팀의 협업 경험을 향상시키고자 노력했습니다. 또한, GitHub의 이슈(Issue)와 PR(Pull Request)의 목적과 중요도를 표시하는 라벨(Label) 기능을 개선하여 협업 과정에서의 명확성을 높였습니다. 이를 통해 다른 팀원들이 생성한 이슈의 목적이 분명해져 긍정적인 피드백을 받을 수 있었습니다.

내가 한 행동의 결과로 어떤 지점을 달성하고, 어떤 깨달음을 얻었는가? 자료 조사와 오픈 소스 분석을 통해 배경 지식의 중요성을 깨닫고, 이를 팀원들과 공유함으로써 협업의 질을 높였습니다. GitHub 라벨 개선 작업을 통해 팀 내 커뮤니케이션의 명확성을 향상시켰고, 긍정적인 반응을 얻었습니다. 하지만 Pseudo Labeling과 같은 특정 기법에 대한 후속 연구를 충분히 학습하지 못한 점이 아쉬웠습니다. 논문 출판 시기 이후의 다양한 후속

연구들이 있었음에도 불구하고, 이를 깊이 탐구하지 못한 것이 한계로 남았습니다.

전과 비교하여 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가? 프로젝트 진행 과정에서 GitHub의 협업 기능을 개선하여, 이슈와 PR의 목적을 명확히 표시하고 중요도를 구분할 수 있도록 라벨링 기능을 개선했습니다. 이를 적용한 후 팀원들로부터 협업의 명확성이 향상되었다는 피드백을 받았습니다. 또한, 이전 기수의 프로젝트 결과물을 읽고 정리하는 시간을 가지면서, 팀 전체의 학습 효율성을 높이는 데 기여했습니다.

마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가? Pseudo Labeling과 같은 기법의 후속 연구에 대해 충분히 탐구하지 않은 점이 아쉬웠습니다. 논문의 출판 이후 다양한 후속 연구들이 있었지만, 이를 깊이 있게 학습해보지 않아 다양한 접근 방법을 시도하지 못했습니다. 또한, 전체적인 배경 지식이 부족하여 팀의 진행 상황을 따라가는 데 어려움이 있었고, 이를 보완하기 위해 추가적인 학습이 필요했습니다.

한계/교훈을 바탕으로 다음 프로젝트에서 시도해볼 것은 무엇인가? 다음 프로젝트에서는 필요한 배경 지식을 우선 학습하고, 관련된 후속 연구들에 대해서도 공부함으로써 보다 깊은 사고 방식을 경험하고자 합니다. 이러한 과정에서 논문을 읽고 분석하는 능력을 향상시킬 수 있을 것으로 기대됩니다. 이를 통해 새로운 아이디어를 시도하고, 구체적인 개선 방안을 마련하는 데 더욱 집중할 계획입니다.

10.5 김성주

나는 내 학습 목표 달성을 위해 무엇을 어떻게 했는가? 프로젝트 초반에 부스트 캠프 강의들을 빠르게 수강하며, 프로젝트에 적용할 수 있는 유용한 요소들을 찾아 적용하였습니다. 또한, 이전 프로젝트에서 시도하고자 했지만 구현하지 못했던 부분들을 초기에 적용하려고 노력했습니다. 이러한 접근을 통해 학습 목표를 보다 효과적으로 달성하고자 했습니다.

나는 어떤 방식으로 모델을 개선했는가? 클래스별 데이터 불균형 문제와 이미지별 라벨 수 차이로 인한 학습 문제를 해결하기 위해 데이터셋을 재정의하려고 시도했습니다. 또한, 이미지 불균형 문제를 해소하고 부족한 데이터를 보충하기 위해 Controllable Diffusion을 활용하여 추가적인 이미지를 생성했습니다. 학습 단계에서의 전처리뿐만 아니라 테스트 단계에서도 TTA(Test Time Augmentation)를 적용하여 대회 막바지에 성능을 높이기 위해 노력했습니다.

내가 한 행동의 결과로 어떤 지점을 달성하고, 어떤 깨달음을 얻었는가? 프로젝트 초기에는 단순히 클래스별 Ground Truth의 개수만으로 클래스 데이터의 불균형을 파악하고 이를 해결하려고 했습니다. 그러나 데이터를 자세히 살펴보는 과정에서, 데이터의 양이 적어도 질이 좋은 경우 분류가 잘 이루어진다는 점을 확인할 수 있었습니다. 이를 통해 데이터 전처리와 불균형 문제에 대해 더 깊게 고민하는 계기가 되었습니다.

전과 비교하여 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가? 이번 프로젝트에서는 오프라인 데이터 증강에 딥러닝 기법을 적용했습니다. Diffusion 모델을 사용하여 데이터를 증강하고 이미지를 성공적으로 생성했으나, 생성된 이미지가 작은 객체를 효과적으로 표현하지 못했고, 테스트 이미지와의 일관성이 부족했습니다. 이로 인해 생성된 이미지를 사용했음에도 성능 향상이 이루어지지 않았습니다. 또한, 협업을 개선하기 위해 Slack과 GitHub를 연동하고 GitHub에 더 많은 이슈를 기록함으로써, 팀원 간의 가설과 생각을 공유하는 데 더 용이해졌습니다.

마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가? 프로젝트의 마지막 단계에서 시간 부족을 다시 한번 절실히 느꼈습니다. 이를 통해 프로젝트 초반부터 시간 분배가 얼마나 중요한지를 깨닫게 되었습니다. 제 역할을 명확히 이해하고, 자료 조사와 논문 리뷰에 더 많은 시간을 투자해야겠다고 느꼈습니다. 또한, 코딩 실력의 부족을 실감했습니다. Inference 코드 작성이나 데이터 증강과 같은 베이스라인 코드 구현이 예상보다 오래 걸렸습니다.

한계/교훈을 바탕으로 다음 프로젝트에서 시도해볼 것은 무엇인가? 다음 프로젝트에서는 강의를 들으면서 베이스라인 코드 리뷰를 빠르게 시도해 볼 계획입니다. 이를 통해 팀의 베이스라인 코드 작성 시간을 단축하고, 코딩 실력을 향상시키기 위해 노력하겠습니다. 또한, 협업 환경을 최적화하려고 합니다. 가설이나 지식 공유가 다소 부족하다고 느꼈으며, 다음 프로젝트에서는 Notion, GitHub, Slack의 연동을 통해 협업 효율성을 극대화해 보겠습니다.

10.6 정수현

나는 내 학습 목표 달성을 위해 무엇을 어떻게 했는가? 학습 목표를 달성하기 위해 프로젝트 주제를 탐구하고 가설을 세워 실험을 진행하였습니다. 또한, 팀원으로서 1인분 이상의 역할을 수행하기 위해 노력하였습니다. 다양한 실험을 통해 학습 데이터의 활용과 모델 성능 개선을 목표로 삼았습니다.

나는 어떤 방식으로 모델을 개선했는가? 가설을 설정하고, 결과를 확인한 뒤 개선하는 방식으로 실험을 진행했습니다. 학습 데이터에 다양한 데이터 증강(augmentation)을 추가하여 일부 모델의 성능을 개선할 수 있었습니다. CNN 계열의 모델뿐만 아니라 입력 이미지의 크기에 영향을 받지 않는 Transformer 계열 모델도 도입하여 성능 개선을 시도했습니다. 오프라인 데이터 증강을 위해 Diffusion 모델을 사용하여 학습 이미지를 새롭게 생성해 보았으나, 성능이 떨어졌습니다. 성능 저하의 원인에 대해 충분히

탐구하지 못한 점은 아쉽습니다. TTA(Test Time Augmentation)를 적용하여 Swin 모델의 성능을 5% 정도 개선하는 성과를 얻었습니다.

내가 한 행동의 결과로 어떤 지점을 달성하고, 어떤 깨달음을 얻었는가? EDA를 통해 데이터에 대한 가설을 세우기보다는 모델에만 집중하여 가설을 설정하고 실험을 진행한 것이 아쉬웠습니다. 특히, 중간 크기(medium)와 작은 크기(small) 객체를 잘 탐지할 수 있는 방법에 대해 더 깊이 탐구하고 나만의 가설을 세워 실험을 진행했어야 했는데, 시간 부족으로 그러지 못한 점이 후회됩니다. EDA를 통해 클래스별 탐지 성능을 파악하고, 개별 클래스에 대한 가설 설정 및 실험을 진행하지 못한 점도 아쉬움으로 남습니다.

전과 비교하여 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가? CUDA 버전과 라이브러리 종속성을 고려하여 서로 호환되는 버전들로 필요한 패키지를 설치함으로써 프로젝트 진행을 원활하게 할 수 있었습니다. 이번 프로젝트의 라이브러리 구성을 분석하며, 설정 파일을 객체 지향적으로 수정하고, 우리의 베이스라인 코드에 맞게 파이프라인을 재구성하거나 리팩토링했습니다. 이를 팀원들에게 공유하여 프로젝트 협업의 효율성을 높였습니다. 이전 프로젝트의 피드백을 반영하여, 협업 레포지토리에 이슈(issue)나 PR(Pull Request)을 만들 때 이슈를 제기하는 이유와 수행할 실험, 실험 결과 정리와 원인 분석, 개선 실험 보고 등을 철저히 기록했습니다. 이를 통해 기록의 중요성을 느끼고 회의록과 가설 정리 문서(docs)를 만들어 능률적인 협업을 시도했습니다.

마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가? 학습 데이터의 클래스가 10개인 상황에서, 무겁고 큰 모델이 성능이 더 좋을 것이라는 가설이 항상 맞지 않을 수 있다는 점을 고려하지 못했습니다. Transformer 기반 모델 대신 비교적 가벼운 여러 모델을 사용하여 실험하지 못한 점도 아쉬웠습니다. Diffusion 모델을 사용해 생성한 학습 이미지가 성능을 향상시킬 것이라고 기대했으나, 그렇지 못했고 이에 대한 원인 분석도 하지 못했습니다. Swin 모델에 추가로 적용한 증강 기법 중 일부가 성능을 하락시키는 결과를 초래했는데, 이는 EDA를 통해 해당 태스크와 Swin 모델에 맞지 않는 증강 기법이 무엇인지 탐구하지 않은 탓이 컸습니다. 모델의 구조와 특성을 제대로 파악하지 못한 채 무작정 실험을 진행한 점도 아쉬운 부분입니다.

한계/교훈을 바탕으로 다음 프로젝트에서 시도해볼 것은 무엇인가? 다음 프로젝트에서는 모델에 국한된 가설을 세우는 것뿐만 아니라, 구체적인 EDA를 통해 가설을 설정하고 실험을 진행할 것입니다. 모든 실험에는 명확한 이유가 있으며, 그 결과를 명확히 제시할 수 있도록 노력할 것입니다. 또한, 협업의 효율성을 높일 수 있는 기록 방법을 고민하고 이를 시도해볼 계획입니다.