게임 영상 기반
적 객체 식별 소프트웨어 구현

YOLO11 기반 게임 영상 사람 탐지

컴퓨터소프트웨어학과

2021270626 박태현

목차

01 **파이프라인 및 알고리즘** 파이프라인 개요 및 YOLO11 튜닝 학습

02 **데이터셋 구성 및 전처리** 4개 데이터 셋 통합

03 **개발 환경 및 학습 방법** Colab T4 · 체크포인트 · 조기 종료

04 **결과** Confidence 조정을 통한 게임 영상 적용

05 **기대 효과 및 확장성** 분야 확장 가능성 시사

파이프라인 개요

PIPELINE

데이터 선정 및 다운로드

↓

데이터 preprcessing 및 Train/Validation 데이터 분할

↓

YOLO11기반 튜닝 학습

↓

Confidence 값 조정을 통한 동영상 추출

↓

최종적으로 적합한 결과 동영상 선정

알고리즘

YOLO11 기반 튜닝 학습

- 공개된 YOLO11 사전 학습 모델을 사용
- 선정한 군인 및 게임 캐릭터 이미지로 추가 튜닝 학습
- 사람 객체의 Bounding Box 위치를 추론하도록 학습을 진행
- 적용 영상은 발표자 본인의 녹화영상을 기반으로 함.



게임 영상에서 탐지된 객체 예시

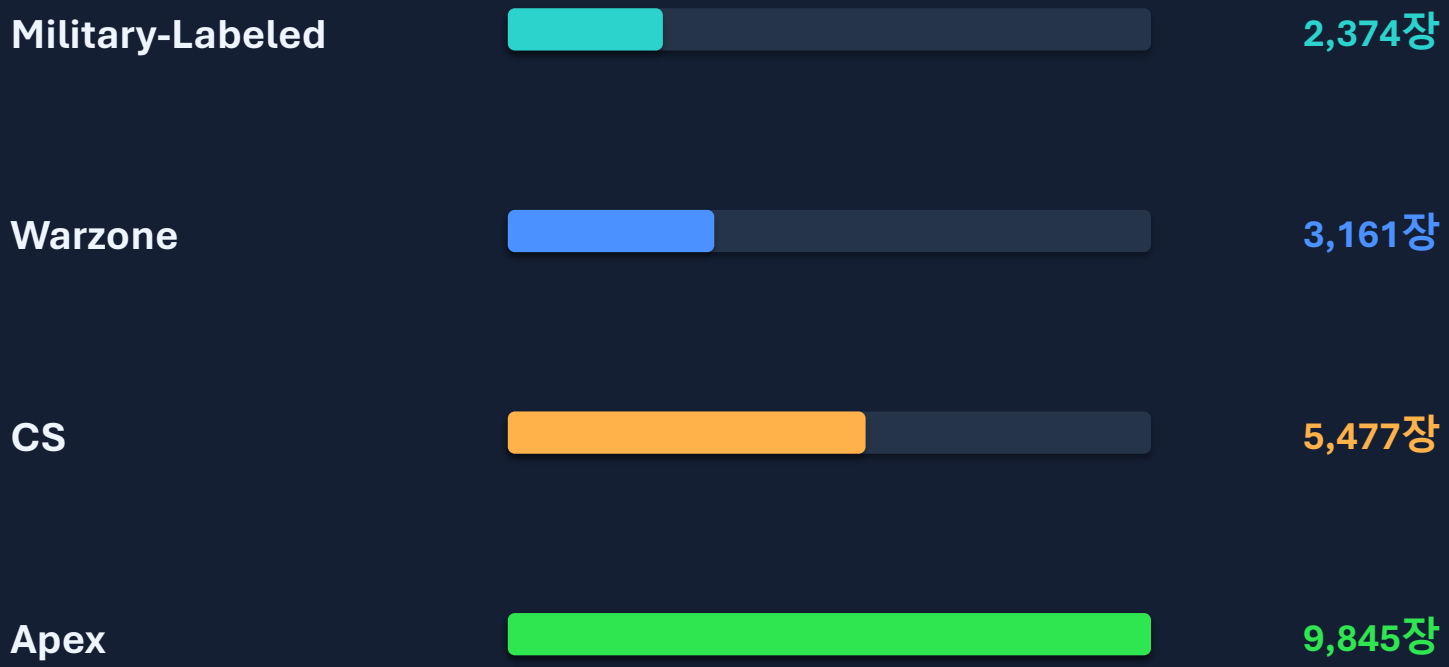
데이터셋 구성

TOTAL IMAGES

20,857 장

4개 데이터셋 통합

학습·검증 데이터로 분할



각 데이터셋의 이미지 수량을 기준으로 구성

데이터셋 예시

Military-Labeled

2,374장



Warzone

3,161장



CS

5,477장



Apex

9,845장



데이터셋별 변환 과정 ①

Military-Labeled와 Warzone의 원본 라벨을 공통 객체 탐지 형식으로 변환

MILITARY-LABELED

원본 형식 · YOLO TXT+ YAML

- 이미지 및 클래스 정보 yaml 파일과 객체의 상대위치 정보 txt파일로 구성
- YAML에서 클래스 이름 정보를 확인 후 military 클래스를 가진 TXT 파일을 선별
- 선택한 객체의 클래스 ID를 0(Person)으로 통일

결과: Soldier → Person (Class 0)

WARZONE

원본 형식 · COCO JSON

- 이미지 메타데이터(images), 객체 어노테이션(annotations), 카테고리 정보(categories)가 JSON 형태로 통합 저장
- Category ID(keep_ids) 이름을 기준으로 사람 객체 선별
- COCO의 (x, y, w, h) 절대 좌표를 불러옴
- 이미지 경계에 맞게 박스를 보정한 뒤 YOLO의 상대좌표로 변환
- 중심점 기준 좌표로 바꾸고 이미지 크기에 따라 정규화 진행

결과: keep_ids→ Person (Class 0) + 정규화 좌표

데이터셋별 변환 과정 ②

CS:GO metadata와 Apex YOLO 원본 라벨을 공통 객체 탐지 형식으로 변환

CS:GO PLAYER DETECTION

원본 형식 · Hugging Face metadata.jsonl

- metadata.jsonl에서 이미지 파일명과 바운딩 박스 정보 추출
- ct_body, ct_head, t_body, t_head 로 구성된 정보를 확인
- 몸체(Body) 클래스만 사용하고 머리(Head)는 제외
- 좌표를 이미지 크기 기준으로 정규화해 YOLO 형식 변환

결과: Body → Person (Class 0) + 정규화 좌표

APEX

원본 형식 · YOLO TXT

- TXT 라벨에서 객체별 Class ID와 박스 좌표 확인
- Class ID 0인 enemy만 학습 대상으로 선택
- Class ID 1인 mate는 학습 대상에서 제외
- 선택한 적 객체의 Class ID를 0으로 통일
- 기존 YOLO 정규화 좌표 형식을 유지

결과: Enemy → Person (Class 0)

공통 YOLO 형식 통합 및 데이터 분할

서로 다른 라벨 구조를 하나의 클래스 체계와 좌표 규칙으로 맞춰 학습 데이터 구성

Military

원본 라벨

Warzone

원본 라벨

CS:GO

원본 라벨

Apex

원본 라벨

공통 클래스 ID = 0 | Person

YOLO 라벨: class_id x_center y_center width height 형식 / 이미지 크기를 기준으로 0~1 사이로 정규화

+ 분할 정보 처리

infer_split()으로 원본 경로·파일에서 train/val 구분을 추정하고, 구분되지 않으면 train으로 처리

+ 제외 및 통합

test 데이터는 변환에서 제외하고, 이미지와 라벨을 짝지어 공통 데이터 폴더로 복사

※ 코드상 별도 무작위 분할 비율은 확인되지 않음. 원본 split을 추정해 유지하며, 미확인 데이터는 train으로 처리

개발 환경

실행 환경

Google Colab

클라우드 기반 Python 노트북

GPU

NVIDIA T4

모델 학습 및 영상 추론 연산

개발 도구

Python · YOLO11

데이터 처리와 모델 학습 파이프라인

일반적인 모델 학습 소요 시간

약 4~8시간

데이터 규모와 학습 설정에 따라 변동

모델 학습 과정 및 문제 해결

01

데이터 분할

학습용 / 검증용 데이터 구성

→

02

YOLO11 튜닝

사전 학습 가중치에서 추가 학습

→

03

주기적 저장

Epoch별 체크포인트 저장

→

04

조기 종료

검증 성능 정체 시 학습 중단

→

05

최종 가중치

학습된 모델을 영상 추론에 사용

해결한 구현상 문제

Colab 런타임이 예기치 않게 종료되는 경우가 빈번하게 발생
이를 해결하기 위해 학습 가중치를 Epoch주기 5마다 .pt파일로 저장하고,
검증 성능이 0.1이상으로 개선되지 않는 경우가 Epoch 주기 10 이상 지속되면
조기 종료하도록 구성함.

Hyperparameter 최적화

최적화 절차

- Confidence 임계값을 단계적으로 변경
- 객체를 연속적으로 식별하는지에 대한 Frame_count 변수 사용
- 탐지 수와 오탐지 발생 양상을 비교
- 식별률을 올리기 위해 Confidence는 줄이고, 오탐률을 줄이기 위해 Frame_count는 증가시킴.
- Confidence는 0.4, Frame_count는 3으로 설정 됨.



탐지 결과 예시 · Person 0.48

실제 구현 결과



게임 영상 결과 ①

15.5초 · MP4



게임 영상 결과 ②

18.6초 · MP4



게임 영상 결과 ③

34.0초 · MP4

실제 화면에서 탐지된 객체를 Bounding Box와 Confidence 값으로 표시

결과 및 확장성







결과

- 게임 영상에서 적 객체를 자동 식별하는 기술 구현
- 픽셀이 작은 게임 캐릭터를 식별하는데 유의미한 식별률을 보임

확장성

- 고품질의 영상 데이터를 학습시키면 실제 군사 목적으로 사용 가능한 학장성
- 드론, CCTV, 열화상 탐지 같은 응용분야에 적용가능.

CONCLUSION

프로젝트 요약

01 4개 데이터셋을 통합하여 총 20,857장 규모의 학습 데이터 구성

02 Google Colab T4 GPU 환경에서 YOLO11 튜닝 학습 수행

03 개발 환경 설정 및 Hyperparameter 최적화 작업 수행

04 결과를 바탕으로 유의미한 확장 가능성을 식별

감사합니다.