|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 프로젝트 주제 | 차량 번호판 분석 | 팀명 | 욜로졸로 | |
| 프로젝트 기간 | 2024/07/29 ~ 2024/8/21 | 참여인원 | 4인 | |
| 구성 및 역할 | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **이름** | **역할** | **담당업무** | **학교** | | 1 | 김기훈 | 팀장 | 데이터 정제, 프로젝트 모델 설계 및 구축 | 인하대 | | 2 | 지종훈 | 팀원 | 웹서비스 구축 | 인하대 | | 3 | 박민호 | 팀원 | 프로젝트 모델 구축 | 인하대 | | 4 | 손슬기 | 팀원 | 데이터 수집 및 라벨링 | 인하대 | | | | |
| 활용 기술 | |  |  |  | | --- | --- | --- | | **No** | **구분** | 내용 | | 1 | Languages | Python 3.11 | | 2 | Framework | Ngrok, YOLO | | 3 | Library | PyTorch, OpenCV, PIL , Numpy | | | | |
| 개발환경 | H/W : RTX 4070 Laptop, Tesla T4 (colab)  S/W : Windows 11 64bit  딥러닝 프레임워크 : Pytorch 2.4.0, CUDA 11.8  개발도구 : Visual Studio Code, Jupyter notebook | | | |
| 프로젝트 주제선정 배경 및 기획의도 | **1. 프로젝트 주제 선정 배경**  : 도시 공간에서 CCTV의 활용은 각종 범죄 예방 및 실시간 모니터링에 중요한 역할을 하고 있다. 특히 차량 관련 범죄나 긴급 상황에 빠르게 대응하기 위해서는 효과적인 차량 식별 시스템이 필수적이다. 현재 차량 번호판을 인식하여 데이터베이스와 대조하는 과정은 대부분 수동으로 이루어지고 있어, 시간이 많이 소요되고 인력의 부담이 크다. 이러한 문제를 해결하기 위해 차량 번호판 인식 과정을 자동화하고, 효율성을 극대화하는 시스템을 개발하는 것이 이번 프로젝트의 목표이다.  **2. 배경 및 기획의도**  : 이 프로젝트는 CCTV 영상 속 차량의 번호판 이미지를 자동으로 추출하고 분석하여, 수배차량이나 긴급 상황에 빠르게 대응할 수 있는 시스템을 개발하는 것을 목표로 한다. | | |
| 개발 주요 사항 | **1. 모델 선정 및 성능 테스트**  ① YOLO  프로젝트에 YOLO 모델을 선정한 이유는 전체 이미지를 한 번에 처리함으로써 빠른 속도와 높은 정확도를 제공하기 때문이다. 이러한 특징은 차량 번호판 인식과 같은 실시간 시스템에서 매우 중요하다. 실시간 처리 능력은 이러한 요소들을 신속하게 처리할 수 있음을 의미하며, 특히 빠른 의사결정이 요구되는 본 프로젝트의 핵심 사항이 된다.  ② Paddle OCR  프로젝트에서 Paddle OCR을 선택한 주된 이유는 아시아 언어에 대한 인식이 특화되어 국내 번호판 인식에 적합하다. 반면, Easy OCR은 사용이 간편하지만, 아시아 언어 인식 에서는 Paddle OCR에 비해 성능이 떨어질 수 있다. Easy OCR은 직접 모델을 트레이닝하여 성능을 개선하는 것이 제한적이라는 단점 또한 고려하여 Paddle OCR모델로 선정하여 프로젝트의 목표를 보다 효과적으로 수행할 수 있도록 하였다.  **2. Data set**  주요 데이터셋은 두가지로 나뉜다. YOLO 학습 데이터와 PaddleOCR 학습 데이터이다. YOLO 주요 데이터셋은 컴퓨터 비전 모델을 학습시키는 데 필요한 데이터를 제공하는 공공 데이터 플랫폼인 Roboflow에서 얻은 프린팅 숫자 이미지[[1]](#footnote-1)로 구성되었으나, 프로젝트의 지역적 적용성을 높이기 위해 추가적으로 국내 번호판 이미지를 직접 수집하고 ‘LabelMe’ 를 활용하여 라벨링 작업을 진행하였다. 최종적으로 YOLO 학습 데이터는 1,653개, 검증 데이터는 115개, 추론 데이터는 50개로 구성되었으며 이 데이터셋은 모델의 학습과 정확도 검증에 사용되었다.  PaddleOCR 주요 데이터셋은 AI 학습용 데이터와 국내외 기관/기업에서 보유한 AI 학습용 데이터를 제공하는 플랫폼인 AI-hub에서 얻은 번호판 이미지이다. 데이터셋은 학습 데이터 30,000개, 검증데이터 3,000개로 구성되어 있다.  **3. Object Detection and OCR 학습**  YOLOv10s 모델을 학습 후 활용하여 1차적으로 번호판만 찾아내 Crop하여 주변 방해요소를 제거 후 2차로 번호판의 숫자들만 각 바운딩박스 찾아낸다. 커스텀 학습된 OCR(광학 문자 인식)모델을 통해 차량 번호판의 문자를 추출하는 과정을 진행하였다. 이 모델은 전체평균 mAP50-95가 평균 0.8253  가 나오며 차량 이미지에서 번호판 위치를 정확하게 식별하고, 안정적으로 텍스트변환을 했다.  **4. Ngrok을 이용한 웹서비스 구축**  VScode 환경에서 Ngrok(외부 네트워크에서 로컬 환경으로 접속하게 해주는 터널링 프로그램)을 이용해 로컬 개발 환경에서 인터넷을 통해 웹 애플리케이션에 접근하였고, 프로젝트 모델을 웹에서 원하는 사진에 학습된 모델을 적용하여 결과물을 볼 수 있는 모델 활용 서비스를 구현했다. | | |
| 프로젝트 수행절차 및 방법 |  | | |
| 다이어그램 |  | | |
| 프로젝트 수행내용 | **1. 데이터 수집 및 준비**  ① 데이터 수집: 도로, 주차장, 교차로 등 다양한 환경과 맑음, 흐림 등의  날씨 조건에서 촬영된 차량 번호판 이미지 데이터셋을 수집한다. 이때 데이터는 다양한 각도, 해상도, 거리에서 촬영한 이미지들을 포함한다. 데이터의 다양성을 확보하기 위해 차량의 종류, 색상, 번호판의 위치 등 다양한 변수를 반영한 데이터셋을 구축한다. 이를 통해 모델이 다양한 상황에 대해 일반화할 수 있도록 한다.  ② 데이터 정제: 수집된 이미지 데이터에서 사람이 식별할 수 없을 정도로 해상도가 떨어지거나, 중복된 이미지, 불필요한 배경이 많이 포함된 이미지를 제거하여 데이터의 품질을 향상시킨다.  ③ 데이터 증강: 이미지 회전, 크기 및 각도 조절, 밝기 조정 등 다양한 기법을 사용하여 데이터셋을 증강한다.  ④ 데이터 라벨링: 각 이미지에서 번호판과 번호를 바운딩 박스로 라벨링한다.  ⑤ 데이터셋 분리: 데이터를 Train, Valid, Test 세가지를 8:1:1로 분류한다.  **2. YOLO 모델 Fine-tune**  ① 모델 선정: 데이터셋을 가지고 YOLO 모델(V8n, V8s, V8m, V9s, V9m, V10s, V10m)을 실험하여 각 모델의 성능을 비교한다. 검증과 추론을 통한 수치 비교해석을 통해 YOLOv10s로 선정하였다. (figure1, figure2, figure3)  ② 모델 학습: 실험 데이터를 활용하여 최적의(YOLOv10s)로 Train 데이터셋을 사용해 300 epochs로 학습을 진행한다.  ③ 모델 검증: 학습이 끝난 모델을 Valid 데이터셋을 사용하여 모델 성능을 평가한다.  ④ 모델 최적화: 모델 최적화 함수는 AdamW 메소드를 사용하여 최적화를 진행한다. 이는 Yolov8에서 공식적으로 제안하는 최적화함수 중 하나이며, 본 과제는 보편적인 최적화 함수를 채택하여 모델 안전성을 보완한다.  **3. PaddleOCR Fine-tune**  ① 모델 선정 : PaddleOCR 모델의 최신버전인 PP-OCRv4는 한국어 지원을  안하는 이유로 PP-OCRv3를 선정하였다.  ② 모델 학습 : Train 데이터 30,000개를 100 epochs로 학습하였다.  ③ 모델 검증 : 학습이 끝난 모델을 Test 데이터셋을 사용하여 모델 성능을 평가한다.  **4. Prediction**  ① Fine-tuning 되어 있는 YOLOv10s와 PP-OCRv3을 활용하여 Test 데이터를  예측한다. 이때 YOLOv10s는 1차적으로 번호판을 검출하고 Crop한 후  Crop된 번호판의 번호를 검출한다. 검출된 번호와 검출되지 않은 한글을 PP-OCRv3가 인지하고 텍스트로 변환하여 출력한다.  **5. 서비스 구축**  ① 번호판 인식 시스템 API 개발: 학습된 YOLO 모델을 기반으로 번호판 인식 API를 개발한다. 이때 API는 입력 이미지에서 번호판을 검출하고, 번호판을 분석하는 기능을 포함한다.  ② 사용자 인터페이스 설계 및 구현: 최종 사용자가 쉽게 접근하고 사용할 수 있도록 직관적이고 사용하기 쉬운 웹 인터페이스를 설계한다.  ③ 성능 최적화 및 실시간 처리: 실시간 처리가 가능하도록 시스템의 추론 속도를 최적화한다.  **6. 배포**  시스템 배포: 개발된 시스템을 실제 운영 환경에 배포한다. | | |
| 성과 및 기대효과 | **성과 - Yolo**  YOLOv10s의 mAP 평가는 다음과 같다.   |  |  |  | | --- | --- | --- | | Label | mAP50 | mAP50-95 | | 0 | 0.993 | 0.822 | | 1 | 0.992 | 0.813 | | 2 | 0.994 | 0.81 | | 3 | 0.995 | 0.83 | | 4 | 0.995 | 0.83 | | 5 | 0.995 | 0.825 | | 6 | 0.994 | 0.87 | | 7 | 0.993 | 0.82 | | 8 | 0.995 | 0.842 | | 9 | 0.988 | 0.817 | | License\_Plate | 0.991 | 0.8 |   mAP50 평균 0.9932 ,mAP50-95 평균 0.8253이 나오며 준수한 검출 수행능력을 볼 수 있다.  다음은 프로젝트 모델의 결과물이다.  텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진  자동 생성된 설명    **성과 – PaddleOCR**  Pre-trained 모델을 AI-hub의 데이터셋으로 학습을 진행하면서 Valid 데이터셋으로 평가하였다.  Best epochs 기준 acc: 0.9375, CTCLoss: 0.2941을 기록했다.  **결과 분석 및 추후 개발사항**  YOLO를 활용한 숫자 검출은 잘 되었으나, PaddleOCR을 활용한 한글 검출은  다소 부정확한 결과가 나왔다.  본 과제는 이러한 결과에 대한 원인으로 다음과 이유를 분석함.  검증단계에서 확인했던 Accuracy는 0.9375를 기록했지만 Test 데이터를 가지고 한 예측에서는 낮은 인식률을 보였다.  - 이러한 결과를 나타낸 주된 원인으로는 과적합으로 보임  - Pre-trained 모델의 기본 시퀀스가 960인 것 반해 본 프로젝트의 모델의 필요 시퀀스는 8 이하였기 때문에 편향적인 학습이 이루어진 것으로 예측됨  **활용방안 및 기대효과**  차량의 번호판 이미지를 자동으로 추출하고 분석함으로써 이를 활용할 수 있는 기대 효과는 다음과 같다.  **교통 법규 위반 차량 감시의 효율화**  : 자동화된 번호판 인식 시스템은 과속, 신호 위반 등 교통 법규를 위반하는 차량을 자동으로 감지하여 교통 단속의 효율성을 높이고, 단속 인력을 절감할 수 있다.  **주차 및 출입 관리의 자동화**  : 주차장에서 차량의 번호판을 인식하여 출입 시간을 기록하고, 주차 요금을 계산하는 시스템을 구축할 수 있다.  이처럼 차량 번호판 인식 프로그램의 OCR 기술은 여러 산업 및 분야에 응용 가능하며, 다양한 방식으로 활용될 수 있다. | | |
| 자체 평가 의견 | 저해상도 또는 불명확한 번호판에 대한 데이터셋으로 학습하지 못하여 현재는 고해상도의 이미지한정 높은 인식률을 형성하고 있다.  PaddleOCR의 Fine-Tuning은 기대에 미치지못한 결과를 보여주었다.  YOLO 모델 Fine-tuning 당시 번호판의 한글도 라벨링하여 학습을 하였다면 매우 높은 인식률을 기대할 수 있다고 본다. | | |
| Github | https://github.com/kkihoo/KEB\_teamproj.git | | |
| figure1 | YOLO 모델 성능 테스트 비교    (학습 후 YOLOv8s)    (학습 후 YOLOv10s) | | |
| figure2 | YOLO model train result  v10s.png  (YOLOv10s)  v8s.png  (YOLOv8s) | | |

1. <https://universe.roboflow.com/university-of-toronto-xho85/numberdetection-eppfj> [↑](#footnote-ref-1)