YOLO 기반 포트홀 감지 시스템

유지환
인하공업전문대학 컴퓨터정보공학과
e-mail: kusatyd@naver.com

YOLO Based pothole **Detection**

ji-hwan yoo
Dept. of Computer Science, Inha Technical College

요 약

본 프로젝트는 도로 파손으로 인한 교통사고 및 유지보수의 비효율성 문제를 해결하고자 YOLO 기반의 객체 검출 시스템을 구축하였다. 방법론으론 KaggleHub의 포트홀 데이터셋 약 665장을 확보하고, XML라벨을 YOLO 형식으로 변환 후 416 x 416크기로 이미지들을 조절하였다. 효율적인 포트홀 탐지 모델을 선정하기 위해 YOLOv11 시리즈의 경량 모델인 n, s, m 버전의 성능을 비교 분석하였다. 비교 결과, YOLOv11-s가 가장 높은 mAP50 0.819를 달성하였으며, 더 큰 모델(m 버전)에서 성능이 하락하는 과적합 징후를 확인하여 YOLOv11-s를 최종 모델로 선정하였다. 최종 모델은 30 Epoch 학습 후 검증 데이터셋에서 mAP50 0.819을 달성하여 높은 탐지 정확도를 확보하였다. 테스트 결과 모델은 크랙이 혼재된 작은 크기의 포트홀도 성공적으로 식별하는 모습을 보여주었다.

▸ Keyword : Pothole Detection, YOLOv11, Object Detection, Road Maintenance, Deep Learning

1. 서론

1.1. 프로젝트 배경

도로의 파손, 특히 포트홀은 차량의 주행안정성을 크게 저하시키며, 교통사고의 원인이 되는 주요 요인으로 지목된다. 포트홀은 계절의 변화, 지반 침하, 배수 불량 등 다양한 환경적 요인으로 인해 지속적으로 발생하며, 보수 인력의 한계로 모든 구간을 실시간으로 관리하기 어렵다. 이러한 문제를 해결하기 위해 자동화된 도로 이상 감지 기술에 대한 수요가 증가하고 있다.

1.2. 도로 포트홀 감지의 필요성

포트홀을 제때 보수하지 못할 경우에는 차량의 파손, 운전자의 안전 위협, 교통 흐름 장애 등 다양한 사회·경제적 손실이 발생한다. 도로 관리 기관은 이를 해결하기 위해서 주기적인 점검을 시행하지만, 인력·시간·비용 측면에서 한계가 뚜렷하다. 따라서 자동화 된 포트홀 감지 시스템은 도로 유지관리의 효율성을 높이고 사고를 예방하는데 필수적인 기술이다. 특히 대규모 도로 네트워크를 관리해야 하는 환경에서는 인공지능 기반 탐지 기술이 기존 방식보다 효율적인 대안이 될 수 있다.

1.3. 프로젝트 목적

이 프로젝트의 목적은 YOLO 객체 검출 모델을 활용하여 도로 이미지에서 포트홀을 정확하게 탐지할 수 있는 모델을 구축하고, 그 성능을 평가하는데 있다. 이를 통해 다음과 같은 의문을 해결하려 한다.

- YOLO 모델을 활용하여 포트홀을 효과적으로 감지할 수 있는가?
- 다양한 학습 조건이 성능에 어떠한 영향을 미치는가?
- 실제 도로 환경에 적용하기 위해 어떤 개선이 필요할까?

1.4. 프로젝트 범위

이 보고서는 YOLO 기반 포트홀 감지 모델 구축 과정을 단계별로 다루며, 데이터 전처리 → 모델 설계 → 학습 → 성능 평가 → 결론의 흐름으로 구성된다.

2. 관련 연구 및 기술적 배경

2.1. 포트홀 탐지 기술 동향

포트홀 감지는 오랫동안 도로 관리 분야에서 중요한 연구 주제로 다루어졌다. 초기에는 인력 점검 또는 단순 센서 기반 진동 분석을 통해 포트홀을 탐지하는 방식이 주류를 이루었다. 스마트폰 장착 가속도 센서를 사용하는 방식도 제안된 바 있으나, 노면 상태와 차량 종류에 따른 변동성이 커서 정확도에 한계가 있었다. 이후 컴퓨터 비전 기술이 발전함에 따라 도로 이미지를 기반으로 포트홀을 자동으로 식별하는 연구가 활발히 진행 되었다. 전통적인 방식인 에지 검출이나 텍스처 분석 기반 기법은 조면, 노면 패턴 변화에 취약하여 성능이 불안정한 문제가 있었다. 최근에는 딥러닝 기반 객체 검출 모델이 높은 성능을 보이면서, 포트홀 감지 분야에서도 CNN이 주류로 자리 잡고 있다.

2.2. YOLO 모델 구조 및 특징

YOLO는 입력된 영상을 하나의 그리드로 분할하고, 각 그리드 셀에서 객체의 위치와 클래스 확률을 예측하는 방식으로 작동한다. 이 구조는 전체 이미지를 단 한번의 Forward Pass로 처리할 수 있도록 설계되어 매우 빠른 속도를 자랑한다. YOLO 모델은 다양한 버전이 존재하며, 이 프로젝트에서는 YOLOv11 시리즈의 모델을 기반으로 학습과 평가하였다.

3. 데이터셋 구성 및 전처리

3.1. 데이터셋 구성

본 프로젝트에서는 KaggleHub를 통해 제공되는 Fully Annotated Potholes Dataset을 사용하였다. 이 데이터셋은 도로 표면을 촬영한 이미지와 함께 각 이미지에 대응되는 XML형식의 어노테이션 파일(바운딩 박스 정보)로 구성되어 있다.

- 출처: KaggleHub ([chitholian/annotated-potholes-dataset])
- 규모: 총 665장의 이미지와 대응되는 라벨 파일
- 클래스: 모든 객체는 단일 클래스('pothole')로 지정되었으며, YOLO 형식으로 변환 시 클래스 인덱스 '0'으로 통일하여 사용하였다.

다만, 이 데이터셋은 대부분 주간 환경에서 포트홀 위에서 가까이 다가가 찍은 사진이 주를 이루는 데이터 편향성을 가진다. 이러한 다양성 부족은 모델의 강건성 확보에 한계점으로 작용하며, 이에 대한 상세 분석은 5.2절에서 다룬다.

3.2. XML 라벨을 YOLO형식으로 변환

원본 라벨은 Pascal VOC형식(XML)으로 제공되기 때문에 YOLO학습에 적합한 방식으로 변환하는 과정이 필요하였다. 이를 위해 다음과 같은 절차를 수행하였다.

- XML파일에서 xmin, ymin, xmax, ymax 좌표 추출
- 원본 이미지 기반 정규화 수행
- YOLO포맷(label.txt)으로 변환
- 단일 클래스(pothole)로 고정하여 0으로 기록

해당 절차는 직접 구현한 convert_xml_to_yolo()함수를 사용하여 자동화하였다.

3.3. 이미지 리사이즈

YOLO 모델 학습에 일관된 입력 크기를 사용하기 위해 모든 이미지를 416 x 416으로 리사이즈하였다. 리사이즈 작업은 pillow를 이용하여 처리하였으며 리사이즈 이미지와 변환된 YOLO라벨은 별도의 폴더에 저장하였다.

3.4. 학습/검증 데이터 분할

전체 이미지를 셔플 후 다음과 같은 비율로 분할하였다.

- Train: 80%

- Validation: 20%

분할 과정 중 라벨 누락을 검사하여 누락된 경우 경고를 출력 하도록 하였다.

3.5. data.yaml 생성

YOLO학습을 위한 설정 파일을 아래의 항목으로 생성하였다.

- train 이미지 경로
- val 이미지 경로
- 클래스 개수(nc): 1
- names: ['pothole']

이 설정을 기반으로 YOLO 모델 학습을 진행 하였다.

3.6. 모델 선정

학습에 사용할 모델을 선정하기 위해 다음과 같은 동일한 조건으로 3종류의 모델의 성능을 비교하였다. 비교 결과는 표 [Table 3.6.1]와 같다.

- Epoch: 30
- 이미지 사이즈: 416
- 배치 사이즈: 16

항목	YOLOv11-n	YOLOv11-s	YOLOv11-m
Precision (P)	0.77	0.828	0.82
Recall (R)	0.717	0.703	0.667
mAP50	0.802	0.819	0.773
mAP50-95	0.517	0.534	0.49

[Table 3.6.1]

위 표와 같이 YOLOv11 시리즈의 경량 모델들을 비교한 결과, YOLOv11-s 모델이 모든 항목에서 가장 우수한 성능을 보였다. 특히 mAP50은 0.819로, YOLOv11-n(0.802) 대비 약 1.7% 향상되었다.
주목할 점은 모델의 크기가 커진 YOLOv11-m 버전의 경우, 파라미터 수가 증가했음에도 불구하고 mAP50이 0.773으로 오히려 YOLOv11-s보다 성능이 하락했다는 점이다. 이는 본 프로젝트의 데이터셋 규모가 비교적 작기 때문에, 모델 용량이 과도하게 큰 m 버전은 데이터의 일반적인 특징 대신 노이즈까지 외우는 과적합 징후를 보인 것으로 해석된다. 따라서 본 프로젝트는 성능(mAP 0.819)과 모델 용량의 효율성 측면에서 가장 균형 잡힌 YOLOv11-s를 최종 모델로 선정하였다.

3.7. 데이터 증강 및 하이퍼파라미터 최적화

앞선 3.6절의 모델 성능 비교 결과에 따라, 선정된 YOLOv11-s 모델에 대해 그리드 서치 대신 랜덤 서치를 사용하여 데이터 증강 비율, 학습률 등 핵심 하이퍼파라미터 탐색을 다음과 같이 수행하였다.

- 학습 및 검증에 data.yaml파일 사용
- 탐색 반복 횟수는 30회로 설정
- 학습 epoch는 10으로 설정
- 파라미터 탐색 완료 후 최적의 파라미터를 별도의 파일로 저장

이 과정을 통해 모델의 오차를 최소화 할 수 있는 방향의 하이퍼파라미터 설정을 확보하였다.

3.7.1. 하이퍼파라미터 탐색 결과

본 학습에 적용될 최적의 하이퍼파라미터를 도출하기 위해, 10 Epoch 동안 30회에 걸쳐 랜덤 서치를 기반으로 탐색을 진행하였다. 탐색 과정에서 시도 된 하이퍼파라미터 조합중 fitness 점수가 가장 높게 나온 상위 5개 조합의 결과는 [Table 3,7,1,1]과 같다.

fitness	lr0	momentum	box
0.46121	0.01199	0.937	7.83165
0.46024	0.00894	0.93783	7.85901
0.45693	0.00641	0.93893	8.33169
0.45466	0.01176	0.92439	7.32363
0.4543	0.00993	0.97107	7.25639

[Table 3.7.1.1]

4. 실험 결과 및 성능 평가

4.1. 모델 학습

학습은 YOLOv11-s 모델을 사용하여 최적 하이퍼파라미터를 적용하였다. 학습에 사용된 설정은 다음과 같다.

- 데이터셋: data.yaml
- 이미지 크기: 416 x 416
- 배치 사이즈: 16
- 학습 epoch: 30
- 하이퍼파라미터: 자동 탐색 된 파라미터 적용

4.2. 학습 결과 및 분석

4.1절에서 설명한 방법으로 모델의 학습을 진행하였다. 학습 과정에서 주요 성능 지표인 총 손실(Train_Loss)과 검증 정확도(Val-mAP)의 변화 추이는 다음과 같으며, 이는 모델의 안정적인 수렴을 보여준다.

1) 학습 손실 분석

모델의 학습 손실(Train_Loss)은 초기 5.940에서 시작하여 최종 3.224까지 꾸준히 감소하며 수렴하는 안정적인 경향을 보였다.

- **Box Loss (객체 위치 오차):** 초기 1.737에서 시작하여 최종 1.121으로 감소하였다. 이는 모델이 포트홀의 바운딩 박스(Bounding Box) 위치를 정확하게 예측하도록 학습되었음을 나타낸다.
- **Cls Loss (클래스 분류 오차):** 초기 2.324에서 최종 0.732으로 가장 큰 폭인 1.592 감소하였으며, 이는 모델이 포트홀을 배경 또는 다른 객체와 명확하게 분류하는 능력을 확보했음을 보여준다.
- **DfL Loss (바운딩 박스 품질 오차):** 초기 1.879에서 최종 1.371으로 감소하여, 바운딩 박스 예측 품질(Quality of Bounding Box Prediction)이 향상되었음을 확인하였다.

2) 검증 정확도 분석

모델의 탐지 능력을 평가하는 핵심 지표인 mAP은 학습이 진행됨에 따라 크게 향상되었다.

- **Val_mAP50:** 가장 일반적으로 사용되는 정확도 지표인 mAP_50은 초기 0.356에서 시작하여 Epoch 30에 최고치인 0.815을 기록하며 높은 탐지 성능을 보였다. 이는 테스트 이미지에서 포트홀을 80% 이상의 확률로 검출할 수 있음을 의미한다.
- **Val_mAP50-95:** 보다 엄격한 기준인 mAP50-95 역시 0.192에서 최종 0.52로 모델의 정밀 탐지 능력이 향상된것을 확인할 수 있었다.

주요 결과	Epoch 1	최종/최고치
Train_Loss	5.94	3.224
Val_mAP50	0.356	0.815
Val_mAP50-95	0.192	0.52

[Table 4.2.1]

4.3. 시각적인 탐지 성능 평가

학습을 완료한 모델의 실질적인 탐지 능력을 검증하기 위해, 별도로 분리된 테스트셋을 대상으로 추론을 수행하고 그 결과를 시각적으로 분석하였다. 모델의 정확도와 정밀도를 입증하는 대표적인 탐지 사례는 다음과 같다.

1) 표준 조건에서의 고신뢰도 탐지

[Fig 4.3.1] 일반 단일 포트홀 감지

모델은 배경 노이즈가 적고 포트홀의 형태가 명확한 표준적인 이미지에서 매우 높은 신뢰도를 보였다. [Fig 4.3.1]에서 확인할 수 있듯이, 단일 포트홀에 대해 90%에 달하는 신뢰도 점수를 부여하며 정확하게 바운딩 박스를 설정하였다.

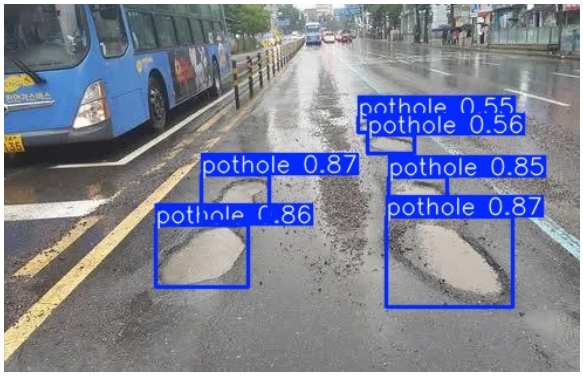
2) 높은 난이도에서의 탐지

[Fig 4.3.2] 크랙과 혼재된 포트홀

모델은 단순한 배경뿐만 아니라 복잡한 노면 환경에서도 높은 탐지 능력을 입증하였다. [Fig 4.3.2]는 일반적인 포트홀 이미지와 달리, 아스팔트의 미세한

크랙들 사이에 위치한 작은 크기의 포트홀을 보여준다. 크랙은 포트홀과 유사하게 어둡고 비연속적인 특징을 가지므로, 모델이 크랙을 포트홀로 오분류 하거나, 크랙의 노이즈 때문에 포트홀 객체를 놓칠 가능성이 높다. 또한, 포트홀의 크기가 작기 때문에 특징 정보가 쉽게 손실된다는 이중적인 어려움이 존재한다. 이러함 조건에서도 모델은 72%의 신뢰도로 포트홀을 성공적으로 식별하였다.

3) 다수 객체 동시 탐지



[Fig 4.3.3] 여러 포트홀 감지

YOLO모델의 핵심 장점인 다중 객체 탐지 능력 또한 확인되었다. [Fig 4.3.3]의 결과는 하나의 이미지 내에 존재하는 여러 개의 포트홀을 동시에 식별하였다. 대부분의 탐지 객체에서 80% 이상의 신뢰도를 유지하였으며, 각 포트홀의 위치와 크기에 맞는 바운딩 박스를 정확하게 예측하여, 넓은 도로 구간에 산재된 포트홀을 효율적으로 관리할 수 있음을 확인하였다.

5. 결론 및 향후 과제

5.1. 프로젝트 요약 및 결론

본 프로젝트는 도로 안전 문제를 해결하기 위해 YOLO 기반의 객체 검출 모델을 활용한 포트홀 자동 탐지 시스템을 구축하였다. 학습 데이터 전처리부터 모델 최적화까지 일련의 과정을 체계적으로 수행하였으며, 그 결과는 다음과 같이 요약할 수 있다.

- **데이터 처리 및 모델 준비:** KaggleHub의 포트홀 데이터셋을 사용하여 Pascal VOC방식의 XML라벨을 YOLO형식으로 변환하고, 모델 입력에 최적화 된 416 x 416사이즈로 리사이즈 하는 과정을 자동화 하였다.
- **하이퍼파라미터 최적화:** model.tune 기능을 사용하여 데이터 증강 비율과 학습률 등 핵심 파라미터를 탐색하여 모델의 성능을 극대화 할 수 있는 최적 환경을 구축하였다.

성능 입증: 총 30 Epoch의 학습 결과, 모델은 총 손실을 3.224까지 안정적으로 감소하였으며, 검증 정확도(Val_mAP50)는 0.815를 달성하여 높은 탐지 성능을 보였다. 특히, 낮은 시야각, 작은 크기, 그리고 아스팔트 크랙이 혼재된 복잡한 환경에서도 포트홀을 성공적으로 탐지하여 구축된 모델의 강건성과 실효성을 입증하였다. 결론적으로, 이 프로젝트는 인공지능 기반의 자동화된 포트홀 감지 시스템이 기존의 인력 중심 점검 방식의 한계를 극복하고 도로 유지보수 효율성을 높일 수 있다는 것을 입증하였다.

5.2. 한계점 및 개선 방안

이 프로젝트의 결과를 실제 도로 관리 시스템으로 사용하기 위해서는 다음과 같은 한계점을 개선하고 개발할 필요성이 있다.

1. 실험 환경 및 자원 제약으로 인한 성능 한계

실제 도로 환경에서 요구하는 높은 수준의 정확도를 달성하지 못한 근본적인 원인 중 하나는 데이터셋의 규모와 다양성 부족에 기인한다.

- **데이터셋 규모 부족:** 학습에 사용된 데이터셋은 665장의 이미지로 규모가 상당히 작았다. 그로 인해 파라미터의 개수가 많은 YOLOv11-m, YOLOv11-l 모델은 충분한 학습을 하지 못하고 과적합되는 모습을 보이며 낮은 성능을 기록하였다. 또한, 최종 선정 모델인 YOLOv11-s 역시 이 규모의 한계로 인해 30 Epoch 이후의 추가 학습에서 성능 향상을 기대하기 어려웠다.
- **데이터셋 이미지의 다양성 부족:** 모델의 학습에 사용된 데이터셋은 대부분 주간의 환경이며, 포트홀 위에서 가까이 다가가 찍은 사진이 주를 이루었다. 이는 모델이 먼 거리, 낮은 시야각 등 다양한 원근감의 데이터가 부족하고 야간, 우천시 등 조도 및 날씨의 변화에 따른 특징 정보 부족으로 이어지고, 이러한 데이터 편향은 모델의 강건성을 저해하여, 학습되지 않은 환경에서의 오탐지율을 높일 수 있다.
- **입력 해상도 및 모델 용량의 한계:** 이미지의 입력 사이즈를 416 x 416의 낮은 해상도로 설정한 것은 미세한 포트홀 탐지에 한계로 작용하였다. 또한, 이 해상도 제약과 작은 데이터셋 규모를 고려했을 때, 더 많은 파라미터(L, M 버전)는 과적합을 유발하여 성능을 저하시켰다. 최종 선정 모델인 YOLOv11-s는 성능과 용량의 균형을 찾은 최적의 모델이었으나, 더 높은 해상도와 복잡한 모델 구조를 활용하지 못한 점은 잠재적인 mAP 개

선에 한계로 작용하였다.

2. 실용적인 적용을 위한 기능적 개선 방향

- **실시간 처리속도 최적화:** 차량 탑재를 위해 동영상 스트림에서의 실시간 처리 성능을 극대화 하여야 한다. 이를 위해 모델 경량화 기술을 적용하는 방안을 고려할 수 있다.
- **피해 등급 분류 기능 도입:** 포트홀의 크기, 깊이, 심각도 등을 기준으로 피해 등급을 분류하는 기능을 추가하여 시스템의 실용적 가치를 높여야 한다.

References

〔1〕 https://www.kaggle.com/datasets/chitholian/annotated-potholes-dataset/data
〔2〕 https://catch-sin.tistory.com/26

GitHub

https://github.com/yjh-311/2025_2_machine-learning