JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 29 No. 8, pp. 85-92, August 2024
https://doi.org/10.9708/jksci.2024.29.08.085

Comparison analysis of YOLOv10 and existing object detection model performance

Joon-Yong Kim*

*Professor, Dept. of IT Convergence Software, Seoul Theological University, Gyeonggi-do, Korea

[Abstract]

In this paper presents a comparative analysis of the performance between the latest object detection model, YOLOv10, and its previous versions. YOLOv10 introduces NMS-Free training, an enhanced model architecture, and an efficiency-centric design, resulting in outstanding performance. Experimental results using the COCO dataset demonstrate that YOLOv10-N maintains high accuracy of 39.5% and low latency of 1.84ms, despite having only 2.3M parameters and 6.7G floating-point operations (FLOPs). The key performance metrics used include the number of model parameters, FLOPs, average precision (AP), and latency. The analysis confirms the effectiveness of YOLOv10 as a real-time object detection model across various applications. Future research directions include testing on diverse datasets, further model optimization, and expanding application scenarios. These efforts aim to further enhance YOLOv10's versatility and efficiency.

▸ **Key words**: YOLOv10, Object Detection, NMS-Free Training, Model Efficiency, Real-Time Processing

[요 약]

본 논문에서는 최신 객체 탐지 모델인 YOLOv10과 이전 버전들 간의 성능을 비교 분석하였다. YOLOv10은 NMS-Free 훈련, 향상된 모델 아키텍처, 효율성 중심의 설계 등을 도입하여 뛰어난 성능을 보인다. COCO 데이터셋을 사용한 실험 결과, 특히 YOLOv10-N은 2.3M의 적은 파라미터 수와 6.7G의 부동 소수점 연산(FLOPs)으로도 39.5%의 높은 정확도와 1.84ms의 낮은 지연 시간을 유지하였다. 주요 성능 지표로는 모델 파라미터 수, FLOPs, 평균 정확도(AP), 지연 시간을 사용하였다. 분석 결과, YOLOv10은 다양한 응용 분야에서 실시간 객체 탐지 모델로서의 효과성을 확인하였다. 향후 연구로는 다양한 데이터셋 테스트와 모델 최적화, 응용 사례 확대 등을 제안하였다. 이를 통해 YOLOv10의 범용성과 효율성을 더욱 높일 수 있을 것이다.

▸ **주제어**: YOLOv10, 객체탐지, NMS-Free훈련, 모델 효율성, 실시간 처리

• First Author: Joon-Yong Kim, Corresponding Author: Joon-Yong Kim
*Joon-Yong Kim (musimk@stu.ac.kr), Dept. of IT Convergence Software, Seoul Theological University
• Received: 2024. 06. 20, Revised: 2024. 07. 19, Accepted: 2024. 07. 30.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

객체 탐지(Object Detection)는 이미지나 비디오에서 객체의 존재를 식별하고 그 위치를 찾아내는 중요한 컴퓨터 비전 작업이다. 객체 탐지 기술은 자율 주행, 보안 감시, 로봇 공학, 의료 영상 분석 등 다양한 분야에서 핵심적인 역할을 하고 있다. 이러한 기술의 발전은 인공지능(AI)과 딥러닝(Deep Learning)의 발전에 크게 의존하고 있다.

YOLO(You Only Look Once)는 실시간 객체 탐지를 위한 대표적인 모델로, 단일 패스에서 객체를 감지하는 방식으로 높은 속도와 정확도를 제공한다. YOLO는 2016년 Joseph Redmon과 그의 동료들에 의해 처음 소개되었으며, 이후 여러 버전으로 발전해왔다[1]. YOLO 시리즈는 단일 신경망을 사용하여 이미지 전체를 동시에 처리함으로써 빠른 속도와 높은 정확도를 달성하는 것이 특징이다.

YOLOv1부터 YOLOv8까지의 버전들은 각각 다양한 개선 사항을 도입하여 성능을 향상시켜왔다. 예를 들어, YOLOv2는 배치 정규화(Batch Normalization)와 고급 앵커 박스(Anchor Box)를 도입하여 객체 탐지 정확도를 높였으며, YOLOv3는 다중 스케일(FPN)과 SPP(Spatial Pyramid Pooling)를 통해 작은 객체 탐지 성능을 개선하였다[12]. YOLOv4는 CSPNet과 PANet을 사용하여 모델의 효율성을 높였고, YOLOv5는 PyTorch 기반으로 구현되어 사용성과 성능을 더욱 향상시켰다[1].

최신 버전인 YOLOv10은 이전 버전들에 비해 더욱 혁신적인 기능을 도입하였다. NMS-Free 훈련 방식을 통해 비최대 억제(NMS) 과정을 제거함으로써 추론 속도를 크게 개선하였고, 일관된 이중 할당 방식을 도입하여 높은 정확도를 유지하면서도 효율적인 모델 학습을 가능하게 하였다[2]. 또한, YOLOv10은 CSPNet 기반의 향상된 백본과 PAN 계층을 사용하여 다양한 스케일에서의 특징 추출 및 결합을 더욱 효율적으로 수행한다[3-5].

본 논문에서는 YOLOv10과 이전 YOLO 버전들 간의 성능을 공인된 COCO 데이터셋을 사용하여 비교 분석한다. 주요 성능 지표로는 모델 파라미터 수, FLOPs, 평균 정확도(AP), 지연시간(Latency)을 사용하며, 이를 통해 YOLOv10의 성능 향상을 객관적으로 평가한다. 또한, YOLOv10의 주요 개선 사항과 이를 통해 얻어진 성능 향상의 이유를 상세히 분석한다. 본 연구는 실시간 객체 탐지 모델의 발전 방향을 제시하고, 다양한 응용 분야에서 YOLOv10의 활용 가능성을 검토하는데 기여할 것이다.

본 논문의 구성은 다음과 같다. II장에서는 관련 연구를 다루고, III장에서는 연구 방법을 설명한다. IV장에서는 실험 결과를 제시하며, V장에서는 결과 분석을 수행한다. 마지막으로 VI장에서는 결론 및 향후 연구 방향에 대해 논의한다.

II. Preliminaries

1. Related works

1.1 YOLO

객체 탐지 모델은 전통적인 HOG와 DPM부터 딥러닝 기반의 CNN 모델로 발전해왔다. YOLO 시리즈는 단일 패스에서 객체를 탐지하는 One Stage Detector 방식으로 높은 성능을 보이며, 다양한 버전으로 발전해왔다[6-10]. YOLOv1부터 v8까지의 특징을 비교하면 다음의 표 1과 같다. 표 1은 본 연구자의 기존연구 결과로 발표한 논문을 인용하였다[11].

Table 1. Characteristics of YOLO Models[11]

Ver	Lunch	Feature
1	2016	- Feature map extraction based on CNN - Application of Bounding Box - Application of Darknet-19 model - Single size object detection
2	2017	- Improvement of shortcomings of v1 - Introduction of Batch Normalization technique - Application of Darknet-53 model - Detection of various object sizes - Issues in recognizing some objects
3	2018	- Application of FPN to improve small object detection performance - Application of 3 Bounding Boxes - Application of SPP-back to improve positioning - Improved processing speed using GPU (real-time object detection possible)
4	2020	- Improvement of v3's performance and processing speed (Application of CSPNet) - Improved small object detection (Application of PANet) - Increased accuracy using GPU - Application of Mosaic Argument technique for data augmentation and model regularization
5	2020	- Announced by Ultralytics based on PyTorch - 2 to 4 times improvement in speed and accuracy compared to v4 - Support for lightweight models (mobile, edge) - Support for both CPU and GPU - Application of AutoML functionality
6	2022	- Increased speed and accuracy of v5 - Added pose estimation functionality - Applied RepPAN to enhance PAN
7	2022	- Expanded based on v4 - Proposed expansion and scaling for real-time object detection
8	2023	- Application of state-of-the-art SOTA model - Advanced detection, segmentation, pose estimation, and tracking functionalities - Support for Vision AI tasks

표 1에서 비교한 YOLO의 주요 특징은 다음과 같다.

1) YOLOv1: CNN 기반, 단일 크기의 객체 감지, Darknet-19 모델 적용

2) YOLOv2: Batch Normalization 도입, 다양한 크기의 객체 감지, Darknet-53 모델 적용

3) YOLOv3: FPN 적용, 작은 객체 감지 성능 개선, 3개의 Bounding Box 적용

4) YOLOv4: CSPNet 적용, 작은 객체 감지 개선, Mosaic Data Augmentation 도입[12][13]

5) YOLOv5: PyTorch 기반, 경량화 모델 지원, AutoML 기능 적용

6) YOLOv6: 포즈 추정 기능 추가, RepPAN 적용

7) YOLOv7: 실시간 객체 검출을 위한 확장

8) YOLOv8: SOTA 모델 적용, 객체 추적 및 비전 AI 작업 지원

연구결과는 YOLOv1, YOLOv2는 정확도가 필요로하는 모델에, YOLOv3, YOLOv4는 처리속도가 우선인 모델, YOLOv5, YOLOv6은 정확도와 속도 모두를 필요로하는 모델과 실시간 객체탐지에 적합함을 확인하였다. YOLOv7, YOLOv8은 메모리와 컴퓨팅 성능에 한계가 있는 모델에 적용하여 객체탐지, 포즈 추정 그리고 객체 추적 등을 적용할 모델에 적합하다는 연구결과를 얻었다[11].

III. The Proposed Scheme

본 연구에서는 YOLOv10과 이전 YOLO 버전들의 성능을 공인된 COCO 데이터셋을 사용하여 비교하였다[13].

이전 발표모델 중 YOLOv6~v8을 비교 기준으로 선택한 이유는 YOLOv6부터 정확도와 속도를 모두 반영하고 소형디바이스나 메모리에 한계가 있는 모델에서도 이전 YOLOv1~v5모델의 성능을 상회하는 것으로 나타나 YOLOv1~v5까지의 모델은 비교대상에서 제외하였다.

성능을 확인할 모델의 주요 성능 지표로는 모델 파라미터 수, FLOPs, 평균 정확도(AP), 지연시간(Latency)을 사용하였다. YOLOv10의 주요 개선 사항은 다음과 같다.

1) NMS-Free Training: NMS 제거, 일관된 이중 할당 방식 도입[2][14]

2) 향상된 모델 아키텍처: CSPNet 개선, PAN 계층 포함, one-to-one 및 one-to-many 헤드 도입

3) 효율성 중심의 설계: 깊이별 분리 합성곱, 공간-채널 분리 다운샘플링, 랭크 가이드 블록 설계[14]

실험은 COCO 데이터셋을 사용하여 진행하였다. 각 모델은 동일한 환경에서 훈련 및 테스트되었으며, 실험 절차는 다음과 같다.

1) 데이터셋 준비

COCO 데이터셋을 사용하여 훈련 및 테스트 데이터를 준비한다. COCO 데이터셋은 다양한 객체 클래스와 실제 환경을 반영한 이미지들로 구성되어 있어 모델의 성능 평가에 적합하다.

2) 모델 설정

YOLOv6-3.0-N, Gold-YOLO-N, YOLOv8-N, YOLOv10-N, YOLOv10-S, YOLOv10-M, YOLOv10-B, YOLOv10-L, YOLOv10-X 모델을 설정하였다. 각 모델은 사전 훈련된 가중치를 사용하거나 동일한 조건에서 재 훈련한다.

YOLOv10 모델 각각의 주요 특성은 다음과 같다.

1. YOLOv10-N: 가장 경량화된 모델로, YOLOv6~v8보다 작은 파라미터 수(2.3M)와 FLOPs(6.7G)로 구성되어 있어 모바일 및 엣지 디바이스와 같은 자원 제약이 있는 환경에 적합하다.

2. YOLOv10-S: 속도와 정확도의 균형을 맞춘 모델로, 중간 크기의 응용 분야에서 높은 성능을 제공한다.

3. YOLOv10-M: 중간 크기의 모델로, 더 높은 성능을 필요로 하는 응용 분야에 적합하다.

4. YOLOv10-B: 대형 모델로, 더 높은 정확도와 성능을 제공하여 자율 주행 및 보안 감시와 같은 고성능 응용 분야에 적합하다.

5. YOLOv10-L: 추가적으로 성능을 높인 대형 모델로, 높은 성능과 정확도를 요구하는 복잡한 응용 분야에 적합하다.

6. YOLOv10-X: 최대 성능을 위한 최상위 모델로, 매우 높은 정확도와 성능을 제공하여 실시간 객체 탐지가 중요한 응용 분야에 적합하다.

이상과 같이 YOLOv10 모델들을 세분화하여 비교한 이유는 다양한 응용 분야에서 최적의 성능을 확인하기 위함이다.

3) 훈련

각 모델을 동일한 하드웨어 환경(TensorRT FP16 on T4 GPU)에서 훈련시킨다. 하이퍼파라미터는 모델 간의 공정한 비교를 위해 동일하게 설정한다.

4) 성능 평가

훈련된 모델을 테스트 데이터셋에 적용하여 성능을 평가한다. 주요 성능 지표로는 모델 파라미터 수, FLOPs, AP, Latency를 측정한다.

IV. Experience

실험의 구체적인 과정과 설정은 다음과 같다.

1) 하드웨어 및 소프트웨어 환경

실험은 NVIDIA T4 GPU에서 TensorRT FP16을 사용하여 수행되었다. 모든 모델은 동일한 하드웨어에서 동일한 조건으로 훈련 및 테스트되었다.

2) 데이터 전처리:

COCO 데이터셋의 이미지는 모델 입력 크기에 맞게 조정되었다. 데이터 증강(Data Augmentation) 기법을 적용하여 훈련 데이터의 다양성을 높였다.

3) 훈련 프로세스

각 모델은 사전 훈련된 가중치를 사용하거나, 동일한 데이터셋에서 재 훈련되었다. 훈련 파라미터는 동일하게 설정되었으며, 학습률(Learning Rate), 배치 크기(Batch Size), 에폭(Epoch)수 등을 포함한다.

4) 평가 지표

4-1) 평균 정확도(AP) : COCO 데이터셋의 검증 세트를 기준으로 측정.

4-2) 지연 시간(Latency) : 모델의 추론 속도를 측정하기 위해 TensorRT FP16을 사용하여 계산.

5) 비교 분석

YOLOv10과 이전 YOLO 모델들의 성능을 비교하기 위해 파라미터 수, FLOPs, AP, Latency를 분석하였다.

V. Analysis

실험결과를 분석하기 위한 사전단계로, 다음의 그림 1과 같이 데이터셋을 사용하여 각 모델을 훈련하고, 성능을 확인하기 위해 바운딩 박스를 사용하여 객체의 경계를 표시하고 레이블과 신뢰도 정보를 얻는 코드를 설계하였다.

```
# 라이브러리 불러오기
- Import OpenCV 라이브러리
- Import NumPy 라이브러리
- Import YOLO 라이브러리 from ultralytics

# 모델 불러오기
- Load YOLOv10 모델 from 'yolov10.pt'

# 비디오 파일 열기
- Set 비디오 파일 경로 to 'myDrive/video.mp4'
- Open 비디오 파일 using OpenCV

# 비디오 프레임 처리
- While 비디오 파일이 열려있는 동안:
    - Read a frame from the 비디오 파일
    - If no frame is read, break the loop

    # 객체 탐지 수행
    - Perform object detection using YOLOv10 모델 on the frame

    # 탐지된 객체의 바운딩 박스 그리기
    - For each detection result:
        - Get the bounding boxes from the detection result
        - For each bounding box:
            - Get 좌표 (x1, y1, x2, y2) from the bounding box
            - Get 레이블 from the bounding box
            - Get 신뢰도 from the bounding box
            - Draw a rectangle on the frame using the 좌표
            - Write the 레이블 and 신뢰도 on the frame near the rectangle

    # 프레임 표시
    - Show the frame with the drawn bounding boxes and labels
    - If 'q' key is pressed, break the loop

# 비디오 파일 닫기 및 창 닫기
- Release the 비디오 파일
- Close all OpenCV windows
```

Fig. 1. Train Model Pesudo Code

```
1. Import necessary libraries
   - Import matplotlib.pyplot as plt
   - Import seaborn as sns

2. Create a dictionary to hold sample data

3. Convert the dictionary to a pandas DataFrame

4. Set up a 2x2 grid for plotting the graphs

5. Plot Params vs AP
   - Create a scatter plot with Params (M) on the x-axis
     and AP (val, %) on the y-axis
   - Add a title and labels to the axes

6. Plot FLOPs vs AP
   - Create a scatter plot with FLOPs (G) on the x-axis
     and AP (val, %) on the y-axis
   - Add a title and labels to the axes

7. Plot Params vs Latency
   - Create a scatter plot with Params (M) on the x-axis
     and Latency (ms) on the y-axis
   - Add a title and labels to the axes

8. Plot FLOPs vs Latency
   - Create a scatter plot with FLOPs (G) on the x-axis
     and Latency (ms) on the y-axis
   - Add a title and labels to the axes

9. Adjust the layout of the plots for better visualization

10. Display the plots
```

Fig. 2. Train Model Pesudo Code

그림 2의 과정은 다음과 같이 설계하였다.

1) 라이브러리 불러오기 :

matplotlib.pyplot과 seaborn을 불러와서 그래프를 생성하고 커스터마이징한다.

2) 샘플 데이터 생성:

다양한 YOLO 모델 버전에 대한 샘플 데이터를 담은 딕셔너리 data를 생성합니다. 여기에는 각 모델의 파라미터 수, FLOPs, AP(평균 정밀도), 지연 시간이 포함된다.

3) DataFrame 생성 :

딕셔너리를 판다스 DataFrame df로 변환하여 데이터를 더 쉽게 조작하고 그래프를 그릴 수 있게 한다.

4) 그래프 설정 :

plt.subplots(2, 2, figsize=(15, 12))를 사용하여 2x2 그리드의 서브플롯을 생성한다. 이는 여러 그래프를 위한 캔버스 설정이다.

5) Params vs AP 그래프 :

Params (M)를 x축에, AP (val, %)를 y축에 사용하여 산점도를 생성한다. 각 점은 모델 버전에 따라 색상을 구분한다. 그래프에 제목과 축 레이블을 추가하여 정보를 더한다.

6) FLOPs vs AP 그래프 :

첫 번째 그래프와 유사하게, FLOPs (G)를 x축에, AP (val, %)를 y축에 사용하여 산점도를 생성하고 제목과 레이블을 추가한다.

7) Params vs Latency 그래프 :

Params (M)를 x축에, Latency (ms)를 y축에 사용하여 산점도를 생성하고 제목과 레이블을 추가한다.

8) FLOPs vs Latency 그래프:

마지막으로, FLOPs (G)를 x축에, Latency (ms)를 y축에 사용하여 산점도를 생성하고 제목과 레이블을 추가한다.

9) 레이아웃 조정 :

plt.tight_layout() 함수를 호출하여 그래프들 사이의 간격을 조정하고, 그래프들이 피규어 안에 잘 맞도록 조정한다.

10) 그래프 표시 :

plt.show()를 호출하여 그래프를 표시한다.

위의 코드에 의해 얻은 파라미터 수, FLOPs, AP, Latency는 다음의 그림 3, 그림 4, 그림 5와 그림 6과 같다.

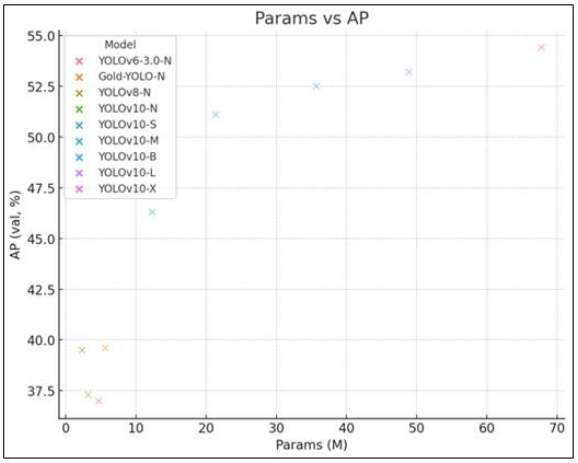

Fig. 3. Parama vs AP

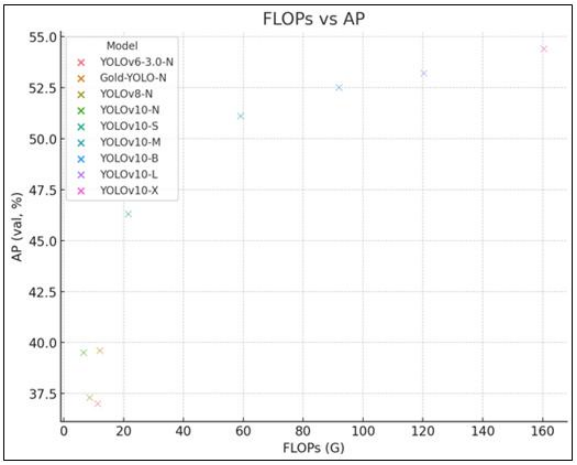

Fig. 4. FLOPs vs AP

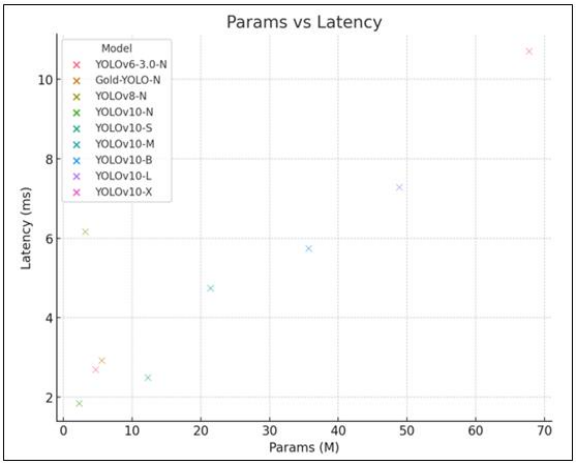

Fig. 5. Parama vs Latency

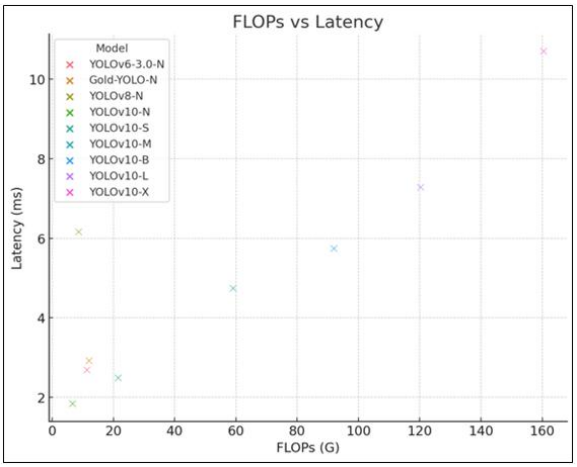

Fig. 6. FLOPs vs AP

위의 그래프는 YOLOv10과 이전 YOLO 버전들 간의 성능 비교를 나타낸다. 각 그래프의 의미는 다음과 같다.

1) Params vs AP(그림 2)

모델 파라미터 수와 평균 정확도(AP) 간의 관계를 보여준다. 파라미터 수가 증가할수록 AP가 높아지는 경향을 확인할 수 있다. YOLOv10-N은 적은 파라미터 수로도 높은 AP를 기록한다.

2) FLOPs vs AP(그림 3)

부동 소수점 연산 수(FLOPs)와 평균 정확도(AP) 간의 관계를 보여준다. FLOPs가 증가함에 따라 AP가 높아지는 경향을 보인다. YOLOv10 모델들은 높은 효율성을 보여준다.

3) Params vs Latency(그림 4)

모델 파라미터 수와 지연 시간(Latency) 간의 관계를 나타낸다. 파라미터 수가 적을수록 지연 시간이 낮아지는 경향을 보인다. YOLOv10-N은 적은 파라미터 수로 1.84ms의 낮은 지연 시간을 유지한다.

4) FLOPs vs Latency(그림 5)

부동 소수점 연산 수(FLOPs)와 지연 시간(Latency) 간의 관계를 보여준다. FLOPs가 적을수록 지연 시간이 낮아지는 경향을 확인할 수 있다. YOLOv10 모델들은 높은 FLOPs에도 불구하고 1.84ms의 낮은 지연 시간을 유지한다.

다음의 표 2는 그래프로 확인한 결과를 정량적으로 비교하기 위한 것이다.

Table 2. Performance comparison table

Model	Parama (M)	FLOPs (G)	AP (val,%)	Latency (ms)
v6-N	4.7	11.4	37.0	2.69
Gold-N	5.6	12.1	39.6	2.92
v8-N	3.2	8.7	37.3	6.16
v10-N	2.3	6.7	39.5	1.84
v10-s	12.3	21.6	46.3	2.49
v10-M	21.4	59.1	51.1	4.74
v10-B	35.7	92.0	52.5	5.74
v10-L	48.9	120.3	53.2	7.28
v10-X	67.8	160.4	54.0	10.70

1) Params vs AP는 모델 파라미터 수와 평균 정확도(AP) 간의 관계를 분석한 것이다. 경량화된 모델인 YOLOv10-N은 적은 파라미터 수(2.3M)로도 높은 정확도(39.5val,%)를 유지한다. 모델 크기가 커짐에 따라 정확도(AP)가 증가하는 경향을 보인다.

2) FLOPs vs AP는 부동 소수점 연산 수와 평균 정확도(AP)간의 관계분석이다. FLOPs가 적은 모델(YOLOv10-N, YOLOv8-N)도(6.7G, 8.7G) 높은 정확도(39.5val,%, 37.3val,%)를 보이며, FLOPs가 증가함에 따라 정확도도 증가하는 경향을 보인다.

3) Params vs Latency는 모델 파라미터 수와 지연 시간(Latency) 간의 관계를 분석한 결과로, 파라미터 수가 적은 모델들이 상대적으로 낮은 지연 시간을 보인다.

YOLOv10-N은 가장 적은 파라미터 수(2.3M)와 가장 낮은 지연 시간(1.84ms)을 기록하여, 경량화된 환경에서 매우 효율적임을 보여준다. 또한, 모델 크기가 증가함에 따라 지연 시간이 증가하지만, YOLOv10 모델들은 파라미터 수 대비 낮은 지연 시간을 유지하여 효율적인 성능을 보인다.

4) FLOPs vs Latency는 부동 소수점 연산 수(FLOPs)와 지연 시간(Latency) 간의 관계를 분석한 결과로 FLOPs가 적은 모델들은 낮은 지연 시간을 보이며, 이는 실시간 응용에 유리하다. YOLOv10-N은 FLOPs가 6.7G로 가장 적으면서도 낮은 지연 시간을 유지하여 높은 효율성을 보인다. FLOPs가 증가함에 따라 지연 시간이 증가하지만, YOLOv10 모델들은 높은 FLOPs에도 불구하고 의 낮은 지연 시간을 유지한다.

이상과 같은 분석결과로 YOLOv10은 낮은 파라미터 수와 FLOPs로도 높은 정확도와 낮은 지연 시간을 유지하여, 다양한 응용 분야에서 효과적으로 활용될 수 있는 것을 확인하였다. YOLOv10-S는 일반적인 용도에서 속도와 정확성의 균형을 잘 맞추고 있으며, YOLOv10-X는 최대 성능이 필요한 응용 분야에서 뛰어난 성능을 발휘한다.

VI. Conclusions

본 연구에서는 YOLOv10과 이전 YOLO 버전들 간의 성능을 비교 분석하였다. YOLOv10은 NMS-Free 훈련, 향상된 모델 아키텍처, 효율성 중심의 설계 등을 통해 높은 정확도와 낮은 지연 시간을 제공한다. 이는 다양한 응용 분야에서 실시간 객체 탐지 모델로서의 효과성을 높이는 결과를 가져왔다.

본 연구에서 YOLOv10 모델 6개(Ov10-N, YOLOv10-S, YOLOv10-M, YOLOv10-B, YOLOv10-L, YOLOv10-X)와 나머지 모델 3개(v6-N, Gold-N, v8-N)를 비교함으로써 다음과 같은 의미를 얻었다.

1. 성능 및 효율성의 진화:

YOLOv10의 다양한 버전들은 각각 다른 파라미터 수와 FLOPs를 가지며, 이를 통해 모델의 경량화와 성능 향상의 균형을 맞출 수 있다. 이는 다양한 응용 분야에서 최적의 모델을 선택하는 데 중요한 정보를 제공한다.

YOLOv6-N, Gold-YOLO-N, YOLOv8-N과의 비교를 통해 YOLOv10의 성능 향상을 객관적으로 평가하였고, 이를 통해 모델의 발전 과정과 그에 따른 성능 변화를 명확히 이해할 수 있다.

2. 응용 분야별 최적 모델 선택:

YOLOv10의 각 버전은 서로 다른 응용 분야에 적합한 성능과 효율성을 제공한다. 예를 들어, YOLOv10-N은 모바일 및 엣지 디바이스와 같은 자원 제약이 있는 환경에 적합하고, YOLOv10-X는 자율 주행 및 보안 감시와 같은 고성능 응용 분야에 적합하다.

이러한 비교 연구는 사용자들이 특정 응용 분야에 최적화된 모델을 선택할 수 있도록 도와준다. 이는 실시간 객체 탐지 모델의 실제 적용에 있어 중요한 지침을 제공한다.

3. 실시간 응용에서의 효과성 검증:

YOLOv10 모델들의 짧은 지연시간과 높은 정확도는 다양한 실시간 응용 분야에서의 효과성을 검증한다. 이는 자율 주행, 드론, 보안 감시 등 실시간 데이터 처리와 분석이 중요한 분야에서 모델의 적용 가능성을 높인다.

4. 모델 발전의 방향성 제시:

YOLOv10의 성능 분석을 통해 모델의 발전 방향성을 제시할 수 있다. 이는 향후 연구에서 모델의 추가적인 개선과 최적화를 위한 기초 자료로 활용될 수 있다.

본 연구는 YOLOv10과 이전 모델들의 비교를 통해 모델의 성능 향상과 응용 분야별 최적화를 위한 중요한 학술적 기여를 하고자 한다. 이는 실시간 객체 탐지 기술의 발전과 적용 가능성을 높이는 데 기여할 것이다.

향후 연구에서는 YOLOv10의 추가적인 응용 분야에 대한 검토와 더불어 모델의 지속적인 개선이 필요하다.

YOLOv10의 성능을 더욱 향상시키기 위해 다음과 같은 향후 연구가 필요하다.

1) 다양한 데이터셋 테스트

COCO 이외의 다양한 데이터셋에서 YOLOv10의 성능을 테스트하여 범용성을 검토한다.

2) 추가적 디바이스 최적화

모바일 및 엣지 디바이스에서의 성능을 더욱 최적화하여 경량화된 모델의 효율성을 극대화한다.

3) 응용 사례 확대

자율 주행, 드론, 보안 감시 등 다양한 실시간 응용 분야에서 YOLOv10의 적용 가능성을 검토한다.

4) 모델 해석성

모델의 예측 결과를 해석하고 설명할 수 있는 방법을 개발하여 신뢰성을 높인다.

ACKNOWLEDGEMENT

This study was supported by the Research Program funded by the Seoul Theological University.

REFERENCES

[1] YOLOv10 object detection Better, Faster and Smaller, visionplatform, https://visionplatform.ai

[2] YOLOv10 - Ultralytics YOLO Docs, https://docs.ultralytics.com

[3] YOLOv10 vs. YOLOv9, Augmented Startups, https://www.augmentedstartups.com

[4] Unlocking the Power of YOLOv10, DEV Community, https://dev.to

[5] YOLOv10: Real-Time Object Detection Evolved, viso.ai, https://viso.ai

[6] Sultana, M., Ahmed, T., Chakraborty, P., Khatun, M., Hasan, M. R., & Uddin, M. S., “Object detection using template and HOG feature matching”, International Journal of Advanced Computer Science and Applications, 11(7), pp233-238, 2020. DOI: 10.14569/IJACSA.2020.0110730

[7] Ali, A., Olaleye, O. G., & Bayoumi, M., "Fast region-based DPM object detection for autonomous vehicles", 2016 IEEE 59th International Midwest Symposium on Circuits and Systems (MWSCAS). IEEE, pp1-4, Oct. 2016. DOI: 10.1109/MWSCAS.

2016.7870113

[8] Arya, S., & Singh, R., "A Comparative Study of CNN and AlexNet for Detection of Disease in Potato and Mango leaf", 2019 International conference on issues and challenges in intelligent computing techniques (ICICT), Vol. 1. IEEE, pp1-6, 2019. DOI: 10.1109/ICICT46931.2019.8977648

[9] Ik-Su, K., Moon Gu, L., & Yongho, J., “Comparative Analysis of Defect Detection Using YOLO of Deep Learning”, Journal of the Korean Society of Manufacturing Technology Engineers, 30(6), pp514-519, 2021. DOI: 10.7735/ksmte.2021.30.6.514

[10] Soviany, P., & Ionescu, R. T., “Optimizing the Trade-off between Single-Stage and Two-stage Deep Object Detectors using Image Difficulty Prediction”, 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), pp209-214, 2018. DOI: 10.1109/SYNASC.2018.00039

[11] Joon-Yong Kim, “A comparative study on the characteristics of each version of object detection model YOLO”. Proceedings of the Korean Society of Computer Information Conference , V.31, No. 2, pp.75-78, July.2023.

[12] Comparison of accuracy and speed by model, https://learnopencv.com/ultralytics-yolov8/

[13] Performance comparison by object size, https://learnopencv.com/ultralytics-yolov8/

[14] YOLOv10: Real-Time Object Detection Evolved, viso.ai, https://viso.ai

Authors

Joon-Yong Kim received the B.S.,degrees in Civil Engeneering from SungKyunKwan University, Korea, in 1985. Then received the M.S. and Ph.D. degrees in Computer Science and Engineering from KongJu National University, Korea, in 2013 and 2018, respectively. Dr. Kim joined the faculty of the Department of IT Convergence Software at Seoul Theological University, Gyeonggi-do, Korea, in 2020. He is currently a Professor in the Department of IT Convergence Software, Seoul Theological University. He is interested in Machine Learning, Auto ML, and AI.