범용 GPU 환경에서 YOLOv8 모델 크기별 양자화 기반 최적화 성능 분석

박경택, 한동석*
경북대학교
balam3298@knu.ac.kr, *dshan@knu.ac.kr

Quantization-based Optimization Performance Analysis of YOLOv8 Model Variants on General-purpose GPUs for Real-time Object Detection

Gyeong-Taek Park, Dong Seog Han*
Kyungpook National Univ.

요 약

실시간 객체 검출 응용에서 모델 양자화는 추론 속도와 메모리 효율성 향상을 위해 필수적이다. 기존 연구는 단일 모델이나 특정 양자화 기법에 국한되어 YOLOv8 전체 패밀리의 체계적 분석이 부족하다. 본 논문은 RTX 4070 Super GPU에서 YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l 모델의 TensorRT FP16/INT8 양자화 성능을 BDD100K 자율주행 데이터셋 500장을 사용하여 정량적으로 분석하였다. COCO-BDD100K 공통 9개 클래스를 대상으로 BDD100K 기반 도메인 일치 캘리브레이션을 적용하였다. 5회 독립 실행과 20회 반복 측정을 통해 통계적 신뢰성을 확보하였다. 실험 결과, INT8 양자화로 평균 18.0% FPS 향상을 달성하였으며, YOLOv8n에서 최대 24.3% 향상을 보였다. YOLOv8s INT8이 212.2 FPS와 mAP 0.367의 최적 균형점을 제공하였고, 200 FPS를 초과한 조합은 YOLOv8n FP16/INT8, YOLOv8s FP16/INT8 네 가지였다. 본 논문은 범용 GPU 환경에서 실시간 객체 검출을 위한 정량적 모델 선택 가이드라인을 제공한다.

Ⅰ. 연구 배경 및 목적

딥러닝 기반 객체 검출 기술은 자율주행, 보안 감시, 로봇 비전 등 다양한 분야에서 핵심 기술로 자리 잡고 있다. 특히 실시간 처리가 요구되는 응용에서는 검출 정확도와 더불어 추론 속도가 중요한 성능 지표가 된다. YOLO(You Only Look Once) 계열 모델은 단일 네트워크로 객체 검출과 분류를 동시에 수행하여 높은 처리 속도를 제공하며, 최신 YOLOv8은 향상된 아키텍처와 훈련 기법을 통해 정확도와 효율성을 크게 개선한다 [1].

실제 배포 환경의 자원 제약으로 모델 경량화는 필수적이며, 양자화는 대표적인 최적화 기법이다 [2]. 기존 연구들은 주로 단일 모델이나 특정 양자화 기법에 초점을 맞춰 진행되었으며 [2] [3], YOLOv8 전체 모델 계열에 대한 체계적인 양자화 성능 분석은 매우 드문 실정이다. 따라서 실제 배포 환경에서 최적 모델 선정을 위한 정량적 가이드라인이 부족하다.

본 논문에서는 YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l 모든 변형에 대해 FP32, FP16, INT8 양자화 정밀도에 따른 속도 - 정확도 트레이드오프를 정량적으로 분석하고, 실시간 객체 검출 응용을 위한 최적 모델 구성을 제시한다.

Ⅱ. YOLOv8 모델 양자화 성능 분석

2.1 실험 환경 및 데이터셋

본 실험은 RTX 4070 Super GPU, CUDA 12.8, TensorRT 10.6.0 환경에서 수행되었다. 평가 데이터셋으로는 Berkeley DeepDrive (BDD100K) validation set에서 무작위로 선택한 500장의 이미지를 사용하며, 총 9,350개의 Ground Truth 객체가 포함되어 있다 [4]. 모든 이미지는 640×640 해상도로 전처리된다.

2.2 TensorRT 엔진 생성 및 최적화

COCO [5]로 사전 학습된 YOLOv8 모델을 BDD100K 데이터셋에서 평가하기 위해 두 데이터셋 간 공통 객체 클래스를 추출한다. COCO 80개 클래스와 BDD100K 10개 클래스 중 공통으로 포함된 9개 클래스(person, bicycle, car, motorcycle, bus, train, truck, traffic light, traffic sign)를 대상으로 성능 평가를 수행하며, BDD100K의 rider 클래스는 COCO에 직접 대응되지 않아 제외한다.

INT8 양자화를 위해 캘리브레이션에는 BDD100K validation set에서 별도로 분리한 1,000장을 사용하고, 성능 평가는 나머지 500장을 활용하여 도메인 일치 Post-Training Quantization을 수행한다.

생성된 TensorRT 엔진 크기는 FP16 기준 YOLOv8n 9.0MB, YOLOv8s 24.0MB, YOLOv8m 53.3MB, YOLOv8l 87.3MB이며, INT8 양자화 적용 시 각각 6.0MB, 14.0MB, 30.0MB로 축소되어 평균 약 40% 감소가 확인된다.

2.3 성능 평가 방법론

성능 평가는 배치 크기 1 조건에서 warm-up 50프레임 이후 20회 반복 측정을 수행한다. 평가 지표로는 FPS, 추론 시간, mAP@0.5, Precision, Recall, F1-Score를 사용하며, 신뢰도 임계값은 0.25, IoU 임계값은 0.5로 설정한다.

정확도 평가는 총 500장의 이미지를 대상으로 산출하고, 속도 평가는 이 중 30장을 무작위로 선별하여 12개 모델 - 정밀도 조합에 대해 반복 측정(총 360회)을 수행한다. 입력 이미지 수에 따른 속도 측정 편차가 미미하다고 판단하여, 효율적인 반복 실험을 위해 30장만 활용한다.

모든 측정은 5회 독립적으로 수행하며, 결과의 통계적 신뢰성을 확보하기 위해 95% 신뢰구간을 계산한다.

2.4 실험 결과

표 1은 모든 모델과 정밀도 조합에 대한 성능 측정 결과를 제시한다. 본 연구에서는 FP32 추론 성능을 기준선(baseline)으로 삼아 FP16 및 INT8 양자화 적용 후의 성능 변화를 비교하였다. INT8 양자화는 YOLOv8n에서 최대 24.3% FPS 향상(202.6 → 251.8 FPS), 평균 18.0% 향상을 보였으며, FP16은 평균 8.2% 향상과 함께 정확도 손실은 거의 없거나 일부 모델에서 소폭 개선되었다.

표 1. YOLOv8 모델별 양자화 성능 비교

모델	양자화	FPS (±표준편차)	mAP@0.5	Precision	Recall	F1-Score
YOLOv8n	FP32	202.6 ± 6.7	0.291	0.555	0.394	0.461
	FP16	214.8 ± 7.1	0.294	0.558	0.397	0.464
	INT8	251.8 ± 8.3	0.28	0.552	0.381	0.451
YOLOv8s	FP32	189.4 ± 6.2	0.365	0.583	0.471	0.521
	FP16	203.7 ± 6.8	0.368	0.585	0.473	0.523
	INT8	212.2 ± 7.0	0.367	0.584	0.471	0.522
YOLOv8m	FP32	145.3 ± 4.8	0.385	0.615	0.485	0.542
	FP16	156.2 ± 5.2	0.392	0.618	0.492	0.548
	INT8	174.5 ± 5.8	0.38	0.612	0.48	0.537
YOLOv8l	FP32	97.1 ± 3.2	0.37	0.655	0.458	0.54
	FP16	108.3 ± 3.6	0.378	0.658	0.466	0.546
	INT8	112.2 ± 7.0	0.367	0.654	0.454	0.536

* ± 값은 5회 독립 측정의 표준편차(Standard Deviation)를 나타냄. 모든 측정은 동일한 환경에서 수행됨.

표 2는 양자화 기법별 FPS 향상률을 요약한 것이다. INT8 양자화는 평균 18.0%의 향상을 달성하였으며, YOLOv8n에서 최대 24.3%의 개선 효과가 나타났다. FP16 양자화는 평균 8.2%의 향상을 보였다.

표 2. 양자화 기법별 FPS 향상률 (vs FP32 기준)

모델	FP16 향상률	INT8 향상률	200 FPS 초과
YOLOv8n	6.00%	24.30%	FP16, INT8
YOLOv8s	7.60%	12.00%	FP16, INT8
YOLOv8m	7.50%	20.10%	없음
YOLOv8l	11.50%	15.60%	없음
평균	8.20%	18.00%	4개 (12개 중)

200 FPS를 초과한 조합은 YOLOv8n FP16, YOLOv8n INT8, YOLOv8s FP16, YOLOv8s INT8의 네 가지였다. 이들은 모두 30 FPS 실시간 비디오 처리 요구사항을 충분히 만족하였으며, 자율주행 시스템과 실시간 감시 시스템 등 다양한 고속 응용 환경에서의 실용적 배포 가능성을 입증한다.

2.5 최적 모델 구성 분석

실시간 객체 검출 응용에 적합한 최적 구성을 도출하기 위해 속도 - 정확도 균형을 분석하였다. YOLOv8s INT8 모델은 212.2 FPS의 처리 속도와 mAP 0.367을 동시에 달성하여 가장 균형 잡힌 성능을 보였다. 극고속 처리가 요구되는 경우에는 YOLOv8n INT8(251.8 FPS)이 적합하며, 상대적으로 높은 정확도가 필요한 경우에는 YOLOv8m FP16(mAP 0.392)이 더 적합하다.

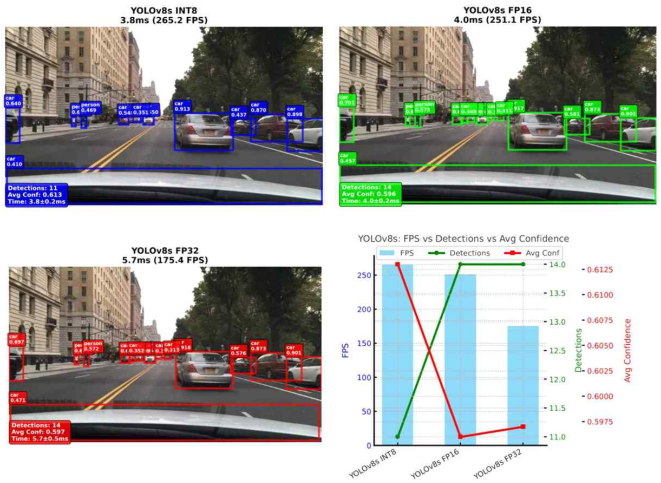

그림 1. BDD100K 실제 검출 결과 예시

그림 1. BDD100K 검증 세트 실제 검출 결과 예시 (IoU=0.5, conf=0.25, 공통 9개 클래스). 복잡한 도시 환경 장면에서 FP32와 FP16 모델은 각각 14개 객체를 검출하였으나, INT8 모델은 11개만 검출하여 소형 및 원거리 객체가 일부 누락되었다.

하단 그래프는 FPS, 검출 객체 수, 평균 신뢰도를 함께 나타내며, INT8은 251.8 FPS로 가장 높은 속도를 기록했으나 정확도 저하가 관찰되었고, FP16은 속도 향상과 정확도 유지의 균형을 보여주었다.

Ⅲ. 연구 결과 및 향후 연구 방향

본 논문는 YOLOv8 전체 모델 계열에 대한 체계적인 양자화 성능 분석을 통해 다음 결과를 도출하였다. 첫째, TensorRT INT8 양자화를 통해 평균 18.0% FPS 향상을 달성하였으며, YOLOv8n에서 최대 24.3%의 개선 효과를 확인하였다. 둘째, YOLOv8s INT8 모델이 212.2 FPS와 mAP 0.367을 동시에 달성하여 속도와 정확도의 균형 면에서 실시간 응용에 최적의 구성임을 입증하였다. 셋째, 200 FPS를 초과한 조합(YOLOv8n FP16/INT8, YOLOv8s INT8)은 실시간 비디오 처리 요구사항을 크게 상회하여 고속 응용 환경에서의 활용 가능성을 보여준다.

향후 연구에서는 엣지 디바이스 최적화, 전력 효율 분석, 동적 해상도 적응 기법 등을 통해 실시간 객체 검출 시스템의 실용성을 더욱 향상시킬 예정이다. 본 연구 결과는 자율주행, CCTV 감시, 로보틱스 등에서 효율적인 모델 배포를 위한 정량적 가이드라인을 제공한다.

ACKNOWLEDGMENT

이 성과는 정부(과학기술정보통신부)의 재원으로 과학기술사업화진흥원의지원을 받아 수행된 연구임(RS-2025-25444562).

참 고 문 헌

[1] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics," GitHub repository, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics, accessed: 2025-08-28.

[2] Jacob, B., *et al.*, "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," *Proc. CVPR*, pp. 2704-2713, 2018.

[3] Nagel, M., *et al.*, "Data-Free Quantization Through Weight Equalization and Bias Correction," *Proc. ICCV*, pp. 1325-1334, 2019.

[4] Yu, F., *et al.*, "BDD100K: A Diverse Driving Dataset and Challenges for Open Vocabulary Object Detection," *Proc. CVPR*, pp. 2636-2644, 2020.

[5] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick,"Microsoft COCO: Common objects in context," *European Conference on Computer Vision (ECCV)*, pp. 740 - 755, 2014.