Jetson 임베디드 플랫폼에서의 YOLOv7 추론 속도 개선에 관한 연구

강보찬[1], 유동영[2]
홍익대학교 소프트웨어융합학과 [1]학부생,[2]교수
gillco00@naver.com, ydy@hongik.ac.kr

A Study on the Improvement of YOLOv7 Inference Speed in Jetson Embedded Platform

Bo-Chan Kang[1], Dong-Young Yoo[2]
Dept. of Software and Communications Engineering, Hongik University

요 약

오픈 소스인 YOLO(You Only Look Once) 객체 탐지 알고리즘이 공개된 이후, 산업 현장에서는 고성능 컴퓨터에서 벗어나 효율과 특수한 환경에 사용하기 위해 임베디드 시스템에 도입하고 있다. 그러나, NVIDIA의 Jetson nano의 경우, Pytorch의 YOLOv7 딥러닝 모델에 대한 추론이 진행되지 않는다. 따라서 제한적인 전력과 메모리, 연산능력 최적화 과정은 필수적이다. 본 논문은 NVIDIA의 임베디드 플랫폼 Jetson 계열의 Xavier NX, Orin AGX, Nano에서 딥러닝 모델을 적용하기 위한 최적화 과정과 플랫폼에서 다양한 크기의 YOLOv7의 PyTorch 모델들을 Tensor RT로 변환하여 FPS(Frames Per Second)를 측정 및 비교한다. 측정 결과를 통해, 각 임베디드 플랫폼에서 YOLOv7 모델의 추론은 Tensor RT는 Pytorch에서 약 4.1배 적은 FPS 변동성과 약 2.25배 정도의 FPS 속도 향상을 보였다.

1. 소개

최근 산업 현장에서 기존의 고성능 대형 컴퓨터가 아니라 저전력, 낮은 메모리 소모량 등 제약사항[1]을 갖는 임베디드 플랫폼에서 딥러닝 모델을 통해 임무를 수행한다. 임베디드 플랫폼은 자율주행 분야, 지능형 CCTV에서의 거동 수상자 추적 등에 사용되고, One-stage-detection 방법을 고안한 YOLO라는 객체 탐지 알고리즘이 사용된다. 특히 YOLOv7은 YOLOv5에 보다 최대 45% 매개 변수를 줄이고 최대 65%의 계산과정을 줄여 47%의 더 빠른 추론 속도 및 개선된 정확도를 달성하였다. [2]

본 논문에서는 NVIDIA의 Jetson 임베디드 플랫폼[3]인 Xavier NX, Orin AGX, Nano를 사용한다. Jetson플랫폼의 추론 속도를 개선하기 위해 기존 Pytorch의 YOLOv7 모델들을 Tensor RT [3]로 해당 플랫폼에 맞게 변환하고 추론 테스트를 진행한다. Jetson Nano의 4GB RAM에서 Workspace 옵션을 통해 YOLO를 구동하는 방법을 기술하고, Jetson 플랫폼마다 Pytorch 및 Tensor RT의 추론 FPS 측정 후, YOLO 모델의 FPS를 비교 분석한다.

2. TensorRT를 활용한 FPS 추론 속도 개선 기법

YOLOv7 모델은 Pytorch 모델로 제공되며 Nomal(이하 'N'), X, W6, E6, D6, E6E 그리고 tiny (이하 'T')로 나뉜다. YOLOv7-N은 일반 GPU, YOLOv7-T는 Edge GPU, YOLOv7-W6는 클라우드 GPU에 최적 사용을 위해 설계[4]되었다.

본 실험에서는 임베디드 플랫폼의 성능을 측정하기 위해 NVIDIA의 Jetson Orin AGX, Jetson Xavier NX, Jetson Nano 을 사용한다. YOLOv7의 Pytorch 모델(총 7종)에 대해 동일한 사진을 추론하고 걸린 시간을 통해 FPS를 10회 측정하여 기록한다. ONNX로 변환 후, TensorRT의 모델로 변환하여 FPS를 측정한다. 모델별 Pytorch 와 Tensor RT의 최댓값, 최솟값 그리고 평균을 산출한 뒤, 결과를 통해 가장 적은 FPS 범위의 모델을 선정하고 다른 모델과 비교한다.

I. Jetson AGX Orin

Pytorch	N	X	W6	E6	E6E	D6	T
Max	39.19	22.81	33.22	24.07	12.46	22.17	59.64
Min	27.75	11.06	21.71	23.02	10.90	20.04	55.20
Max-Min	11.44	11.75	11.51	1.05	1.56	2.13	4.44
Avg	36.49	18.50	18.05	23.75	12.06	21.30	57.84

<표 2> Jetson AGX Orin의 Pytorch 모델 FPS 측정값

TensorRT	N	X	W6	E6	E6E	D6	T
Max	38.30	23.18	42.16	28.80	19.27	23.69	104.8
Min	37.92	22.54	41.49	27.57	19.08	22.99	101.1
Max-Min	0.38	0.64	0.67	1.23	0.19	0.70	3.96
Avg	38.21	22.98	42.00	28.51	19.17	23.62	104.0

<표 3> Jetson AGX Orin의 TensorRT 모델 FPS 측정값

<표 2>와 <표 3>에서 YOLOv7-T 모델의 FPS 측정 결과, 1.8배 향상되어 나타났다. 또한, 해당 모델의 두 딥러닝 모델 플랫폼 간 최대 최소의 값의 차이는 약 1.1배 정도 차이를 나타냈다.

II. Jetson Xavier NX

Pytorch	N	X	W6	E6	E6E	D6	T
Max	16.85	13.46	18.07	13.54	8.52	12.10	40.21
Min	12.68	5.77	11.79	9.90	8.34	9.20	26.87
Max-Min	4.17	7.69	6.28	3.64	0.18	2.90	13.34
Avg	14.15	12.35	13.67	11.39	8.46	10.58	35.35

<표 4> Jetson Xavier NX의 Pytorch 모델 FPS 측정값

TensorRT	N	X	W6	E6	E6E	D6	T
Max	20.95	12.28	24.25	15.95	10.66	12.88	80.73
Min	20.82	12.11	22.54	15.03	10.62	12.59	77.48
Max-Min	0.13	0.17	1.71	0.92	0.04	0.29	3.25
Avg	20.89	12.25	24.15	15.88	10.67	12.82	79.55

<표 5> Jetson Xavier NX의 TensorRT FPS 측정값

<표 4>와 <표 5>에서의 YOLOv7-T 모델의 FPS 측정 결과, 2.25배 향상되어 나타났다. 또한, 해당 모델의 두 딥러닝 모델 플랫폼 간 최대 최소의 값의 차이는 최대 4.1배 정도 차이를 나타냈다.

III. Jetson Nano

Pytorch	N	X	W6	E6	E6E	D6	T
Max							12.21
Min							10.98
Max-Min							1.23
Avg							11.63

<표 6> Jetson Nano의 Pytorch 모델 FPS 측정값

<표 6>의 경우, Jetson Nano의 Pytorch 작동을 위한 포팅을 진행하였다. 한정된 4GB RAM의 이유로 YOLOv7-T 모델을 제외한 나머지 Pytorch 모델이 작동하지 않는 문제가 발생하였다. 이 문제를 해결하기 위해 기존 모델을 TensorRT로 변환 기법을 시도하였으나 기존의 방식대로 변환 시, 메모리 부족의 이유로 변환이 되지 않았다. 모델의 최적 메모리 사용을 위해 workspace 값을 3200 이상 3955 이하의 범위로 직접 찾아 TensorRT 모델로 변환하여 진행하였다.

TensorRT	N	X	W6	E6	E6E	D6	T
Max	2.85	1.649	3.390	2.272		1.770	12.74
Min	2.79	1.635	3.367	2.257		1.740	12.52
Max-Min	0.06	0.014	0.023	0.015		0.03	3.96
Avg	2.820	1.643	3.377	2.263		1.761	12.68
workspace	3200	3950	3900	3900		3950	3200

<표 7> Jetson Nano의 TensorRT 모델 FPS 측정값

<표 7>의 경우 E6E 모델을 제외하고 Jetson Nano에서 TensorRT-Workspace 변환을 통해 추론이 이루어지는 것을 확인하였다. <표 6> 와 <표 7>에서 YOLOv7-T 모델의 FPS 측정 결과, 1.1배 향상되어 나타났다. 또한, 해당 모델의 두 딥러닝 모델 플랫폼 간 최대 최소의 값의 차이는 최대 3.2배 정도 차이를 나타냈다.

3. 결론

Jetson 임베디드 플랫폼에서 TensorRT 변환을 통해 YOLOv7 전반적인 적은 FPS 변동성과 추론 속도 개선이 이루어지는 것을 확인하였다. 또한, 3가지 플랫폼 공통으로 YOLOv7-T 모델이 빠른 추론에 적합하다는 것을 확인하였다. 이를 통해 자율주행 분야에서 높은 추론 속도를 활용할 수 있을 것이라 기대한다.

참고문헌

[1] 이민학, & 강우철. (2017). 통합메모리를 이용한 임베디드 환경에서의 딥러닝 프레임워크 성능 개선과 평가. 정보과학회 컴퓨팅의 실제 논문지, 23(7), 417-423.

[2] Wang, Chien-Yao, Alexey Bochkovskiy, and Hong-Yuan Mark Liao, “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.”, arXiv preprint arXiv:2207.02696, 2022

[3] NVIDIA Corporation, TENSORRT Developer’s Guide, 2019.

[4] Wang, Chien-Yao, Alexey Bochkovskiy, and Hong-Yuan Mark Liao, “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.”, arXiv preprint arXiv:2207.02696, 2022

[4] 박천수. "다양한 컴퓨팅 환경에서 YOLOv7 모델의 추론 시간 복잡도 분석." 반도체디스플레이기술학회지 21.3 (2022): 7-11.