YOLOv5 기반의 딥러닝 성능 개선 연구

박진원, 김영진
아주대학교
jinwon0310@ajou.ac.kr, youngkim@ajou.ac.kr

A Study on Deep Learning Performance Improvement Based on YOLOv5

Park Jin Won, Young-Jin Kim
Ajou Univ.

요 약

최근 딥러닝 분야에서 우수한 성능을 나타내고 있는 CNN(Convolution Neural Network)가 주목받고 있다. GPU 성능 향상으로 인해 딥러닝이 가능해졌지만, 여전히 높은 성능을 내기 위해서는 많은 layer와 parameter 연산이 필요하다. 따라서 딥러닝을 모바일에서 적용하기 위해서는 제한적인 전력과 메모리와 연산능력 때문에 성능을 개선하는 과정이 필요하다. 본 논문에서는 이러한 문제를 해결하기 위해서 오픈소스인 Pytorch YOLO v5[1]를 기반으로 layer와 parameter 연산 수를 줄이는 과정을 통해 성능을 평가하였다. width_multiple과 depth_multiple을 조정하여 layer와 convolution 필터 수를 감소시켰으며, 이 방법을 이용해 YOLOv5n 모델에 비해 inference time 측면에서 약 13% 향상을 보였으며, mAP 측면에서는 약 3.4%의 성능 감소를 보였다. 또한 pre-trained Weight를 이용한 custom pre-trained weight를 이용해서 mAP에서 약 6%의 성능향상을 보였다.

Ⅰ. 서 론

최근 object detection이 자율주행 자동차 등에 활용되면서 정확도뿐만 아니라 신속성 또한 중요해지고 있다. 하지만 기본적으로 연산량이 많아 높은 성능의 GPU 성능을 요구한다. 이전에 RCNN계열의 모델들은 일정수준 이상의 정확성은 확보하였으나, 속도가 너무 떨어져 실시간 object detection에 활용하기에는 문제가 있다.

본 논문에서는 속도가 빠른 CNN계열의 YOLO 모델을 이용하여 성능 개선을 연구하였으며, YOLOv5의 구조분석을 진행하였다. 오픈소스인 YOLOv5의 n 모델을 기반으로 더 빠르고 정확도가 높은 새로운 모델을 만들기 위한 과정과 프로그램의 성능을 높이기 위해 시도한 방법 및 모델에 대한 설명을 서술한다.

Ⅱ. 본 론

1. YOLOv5의 구조

YOLO모델은 object detection을 수행하기 위해 고안된 심층 신경망으로, bounding box coordinate와 classification을 동일 신경망 구조를 통해 실행하는 one stage detector이다.

본 논문에서는 one stage detector인 YOLOv5를 이용하여 성능 개선을 평가하였다. YOLOv5의 구조는 그림 1과 같이 backbone(0~9)와 head(10~23)으로 나눠진다. backbone과 head는 Focus 모듈, Conv 모듈, C3 모듈, SPPF 모듈 등으로 이루어져있으며, v6.0이후 Focus 모듈은 GPU에서 더 빠른 Conv 모듈로 대체되었다. YOLOv5는 네트워크의 크기에 따라 n(nano), s(small), m(medium), l(large), x(xlarge)로 여러 가지 버전이 제공되어 있다. n 모델이 빠르지만 정확도는 가장 낮으며 반대로 x 모델이 느리지만 정확도는 제일 높다[1][2].

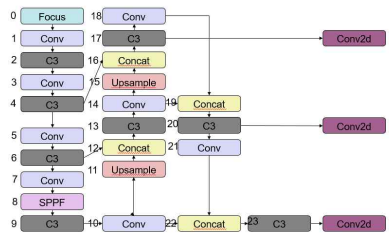

그림 1 YOLOV5의 구조

2. 경량화된 YOLOv5 모델

실시간 object detection을 수행하기 위해서 layer 수와 convolution 필터 수를 n 모델보다 더욱 줄인 새로운 모델(이하 p 모델)을 만들어 성능(그림 2)을 평가해 보았다. n 모델에 비해 정확도가 약 3.4%가 감소하였지만, 속도도 약 13% 감소하였다. 정확도가 감소한 문제를 해결하기

위해 custom pre-trained weight를 새로 생성해주었다. 이를 만들기 위해서는 많은 데이터와 반복이 필요하기 때문에 효율을 높이기 위해서 기존에 주어진 pre-trained weight를 이용해 학습하여 나온 weight를 사용해주었다. inference time에서의 감소는 없었으며, 정확도만 증가하는 결과를 보였다.

모델	mAP@.5	mAP@.5:.95	Inference Time	FPS
Yolov5s	0.985	0.796	25.3ms	39
Yolov5n	0.944	0.691	19.5ms (22%감소)	51 (29%증가)
Yolov5p	0.912	0.652	16.9ms (13%감소)	59 (15%증가)
Yolov5p (best.pt)	0.977	0.742	16.7ms (1%감소)	60 (1%증가)

그림 2 s 모델, n 모델, p 모델의 성능 평가

3. prune 기법을 이용한 YOLOv5 모델

YOLOv5에서 parameter의 수를 가장 많이 차지 하고 있는 것은 단연 Conv 모듈이다. 반면 Upsampling과 Concat 모듈은 parameter가 존재하지 않으며, Focus와 Detect 모듈은 모듈의 수가 적을뿐더러 parameter의 수가 적다. 따라서 해당 모듈을 가지치기를 진행하게 되면 정확도가 현저히 저하된다[3]. 따라서 Conv 모듈과, SPPF 모듈, C3 모듈만 선택적으로 가지치기 기법을 진행하였다.

현재 수행중인 가지치기 기법은 L1 norm을 이용한 가지치기 기법[1]으로 Pytorch 라이브러리에서 제공하는 함수이다. 구성요소중 가장 낮은 L1 norm을 가지는 단위를 가지치기 하는 방법이다. Conv, SPPF, C3 모듈을 동일하게 30%로 가지치기해서 진행해 보았으며, 가장 많은 parameter 수를 차지하는 Conv 모듈만 30%의 가지치기를 진행하고, SPPF, C3 모듈은 10%의 가지치기를 진행한 결과를 그림 3에 정리해놓았다.

동일한 비율로 모듈을 가지치기한 것에 비해 모듈별로 비율을 다르게 하여 가지치기한 모델이 정확도가 약 3%p 좋게 결과가 나왔다.

모델	mAP@.5	mAP@.5:.95	Inference time	FPS
Yolov5p (best.pt)	0.967	0.732	16.7ms	60
Yolov5p (L1 norm prune, 0.3)	0.896	0.587	16.7ms	60
Yolov5p (L1 norm prune, 0.3, 0.1, 0.1)	0.926	0.614	16.7ms	60

그림 3 Prune 비율별 YOLOv5 성능 평가

L1 unstructured prune을 이용한 최적의 가지치기 비율을 찾아보았으며, 성능은 그림 4와 같다. 성능은 모델의 sparsity가 0.0994인 경우 가장 높지만 제거되는 parameter의 수가 작아서 sparsity가 0.0209 혹은 0.0309의 비율로 prune하는 비율을 선택하는 것이 바람직하다.

Conv	C3	SPPF	mAP.5	mAP.5:.95	Sparsity
0.3	0.3	0.1	0.945	0.659	0.298
0.2	0.2	0.1	0.969	0.732	0.199
0.1	0.1	0.1	0.971	0.748	0.0995

그림 4 L1 unstructured prune 성능 평가

또한, 본 연구에서는 L1 structured prune을 이용해 성능을 평가해 보았다. 성능 평가 결과는 그림 5와 같다. 일정 수준이상의 성능을 보이기 시작하는 비율이 너무 작은 값이여서 L1 structured prune을 사용하기에는 부적합하다.

Conv	C3	SPPF	mAP.5	mAP.5:.95	Sparsity
0.03	0.01	0.01	0.829	0.512	0.0309
0.02	0.01	0.01	0.863	0.586	0.0209
0.01	0.01	0.01	0.966	0.732	0.0994

그림 5 L1 structured prune 성능 평가

Ⅲ. 결 론 및 향후 연구 계획

YOLOv5의 backbone을 개량하여 만든 p 모델은 이미 배포된 n 모델에 비해 inference time이 약 13%정도 개선되었으며, custom pre-trained weight를 이용해 학습시킨 p 모델은 n 모델보다 높은 정확도를 보였다.

보다 빠른 속도를 얻기 위해 가지치기 기법을 적용해 보았지만 inference time에서 성능 향상이 없었다. 이는 Pytorch에서는 희소텐서를 지원하는 범위가 너무 제한적이라 가지치기를 통해 0으로 채워진 텐서를 희소텐서로 변환할 수가 없다. 하지만 나중에 희소텐서가 Pytorch에서 지원이 된다면, 성능향상에 도움이 될 수 있을 것으로 생각된다.

본 논문에서 사용한 prune 기법에 비해 더 높은 sparsity를 가지면서 일정수준 이상의 성능을 보일것으로 기대되는 L1 global unstructured prune 기법은 더 좋은 성능을 낼 것으로 기대된다.

ACKNOWLEDGMENT

이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No..2021R1F1A1063706)

참 고 문 헌

[1] YOLOv5, https://pytorch.org/hub/ultralytics_yolov5/.
[2] 전용호, 김익수, 이문구, YOLOv5 모델에 따른 결과 분석, 한국정밀공학회 2021년도 추계학술대회논문집.
[3] 전지훈, 김재명, 강진구, 김용우, “임베디드 시스템에서의 객체 탐지 네트워크 추론 가속을 위한 필터 가지치기 기법 연구”, Journal of The Insititue of Electronics and Information Engineers, 2022.