이미지 피라미드를 이용한 큰 객체 실시간 탐지

주권일*, 손승욱*, 안한세*, 정용화*, 박대희*
고려대학교 컴퓨터융합소프트웨어학과
{wnrnjsdlf, sso7199, hansahn, ychungy, dhpark}@korea.ac.kr

Real-Time Detection of Large Objects using Image Pyramid

Gwonil Joo*, Seungwook Son*, Hanse Ahn*, Yongwha Chung*, and Daihee Park*
*Dept. of Computer Science, Korea University

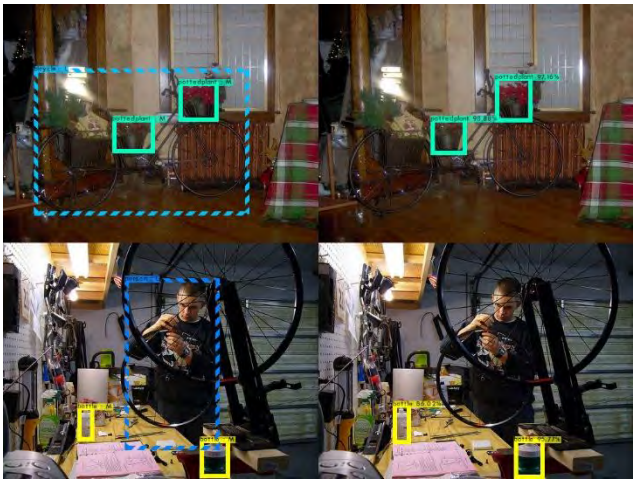
요 약

영상 처리 응용을 위해 개발된 대부분의 CNN 기반 객체 탐지 기법은 mAP를 올리기 위해 작은 객체 탐지에 더 주력하는 경향이 있다. 본 연구에서는 이미지 피라미드를 통한 서로 다른 해상도의 탐지 결과를 앙상블을 하여 작은 객체의 탐지 성능은 유지하면서 큰 객체의 탐지 성능을 향상시키고자 한다. 또한, 기존 NMS 방식의 문제점을 파악하고 새로운 NMS 방식인 G-NMS를 제안한다. COCO 데이터로 실험 결과 서로 다른 해상도의 탐지 결과 앙상블을 통하여 30fps 이상의 실시간 탐지를 만족하면서 큰 객체에 대한 AP가 0.5~1.5% 상승되었음을 확인하였다. 제안한 G-NMS 방식 적용시 큰 객체에 대한 AR이 2.6~3.8% 상승되었으며, 작은 객체를 포함한 전체 mAP가 0.7~0.9% 상승되었음을 확인하였다.

1. 서론

최근 CNN(Convolutional Neural Network) 기술의 발전으로 다양한 영상 처리 응용에 딥러닝 기법이 적용되고 있다. 객체 탐지 알고리즘의 성능 평가 방법으로 널리 쓰이는 방식인 mAP(mean average precision)는 얼마나 많은 객체를, 얼마나 정확하게 탐지했는지를 평가해준다. 특히, COCO 나 ImageNet 에는 작은 객체들이 더 많기 때문에, 성능지표인 mAP 를 높이기 위하여 작은 객체를 더 많이, 정확하게 탐지하는 방향으로 발전하고 있는 추세이다[1,2]. 특히, 작은 크기의 객체를 잘 탐지하기 위해서 사용되는 방법 중 입력 이미지의 해상도를 증가시키는 이미지 피라미드(image pyramid) 방법이 있다[1]. 그러나, 그림 1과 같이 이미지의 해상도를 증가시킬 경우 반대로 큰 크기의 객체를 잘 탐지하지 못하게 되는 trade-off 가 발생한다.

본 논문에서는 이미지의 해상도를 달리 하여 객체 탐지 결과를 앙상블 함으로써, 30fps 이상으로 실시간 탐지 만족 및 작은 객체에 대한 탐지 성능은 유지하면서 큰 객체에 대한 탐지 성능을 향상시키고자 한다. 또한, 중복된 탐지 결과들을 제거하는 기존 Non-Maximum Suppression(NMS) 방식의 문제점을 개선하기 위하여, bounding box 의 기하학적인 중심거리와 종횡비를 고려하는 새로운 NMS 방식을 제안한다. 마지막으로 객체 탐지 실험에서 가장 널리 활용되는 MS COCO 데이터[3]로 실험하여, 실시간 탐지를 만족시키면서 제안 방법의 타당성을 확인한다.



(그림 1) 608×608 해상도일 때, 큰 객체(파란색 점선 박스로 표현)를 탐지하지 못한 경우(왼쪽은 정답 레이블, 오른쪽은 탐지 결과)

2. 서로 다른 해상도의 탐지 앙상블

본 연구에서는 실시간 객체 탐지를 위하여 one-stage 딥러닝 구조로 빠른 객체 탐지에서 널리 이용되는 YOLOv4[4]를 사용하였으며, Ubuntu 18.04 LTS OS, Intel(R) Core(TM) i7-9700 CPU, NVIDIA RTX

2080TI GPU, 16GB RAM 환경에서 실험을 진행하였다.

먼저 표 1 과 같이 COCO TEST 데이터에 대한 YOLOv4 의 입력 해상도 별 정확도 및 처리속도를 측정하였다. 입력 해상도가 커질수록 작은 객체에 대한 탐지 정확도는 감소하였다. 본 논문에서는 전체 객체의 탐지 성능을 유지하면서 큰 객체의 탐지 성능을 향상시키기 위해, mAP 가 최대인 608×608 해상도의 bounding box 들을 큰 객체 탐지가 가장 정확한 384×384 해상도의 bounding box 들과 앙상블하였다. 그 후에 각각의 Greedy-NMS[1], Diou-NMS[5], WBF[6]을 거쳐서 나온 탐지 결과의 성능을 측정하는 실험을 진행하였으며, 이에 대한 결과는 표 2 와 같다.

<표 1> Greedy-NMS 를 사용하였을 때 해상도별 YOLOv4 의 정확도(mAP 및 small/medium/large object 에 대한 average precision) 및 처리속도(frames per second)

해상도	mAP	AP(S)	AP(M)	AP(L)	FPS
384	40.3%	19.0%	43.6%	**56.1%**	**91.1**
416	41.2%	20.4%	44.4%	56.0%	86.0
512	43.0%	24.3%	46.1%	55.2%	74.4
608	**43.5%**	26.7%	46.7%	53.3%	58.6
704	43.1%	28.0%	**46.8%**	49.9%	48.9
800	42.0%	**28.8%**	46.3%	32.9%	39.1

<표 2> 해상도 384×384 와 608×608 의 bounding box 들을 앙상블 했을 경우의 정확도

NMS	mAP	AP(S)	AP(M)	AP(L)	FPS
Greedy [1]	44.2%	25.9%	47.0%	56.6%	**36.3**
Diou [5]	44.5%	26.2%	47.5%	56.9%	36.2
Diou + WBF[6]	**45.1%**	**26.7%**	**48.2%**	**57.6%**	35.5

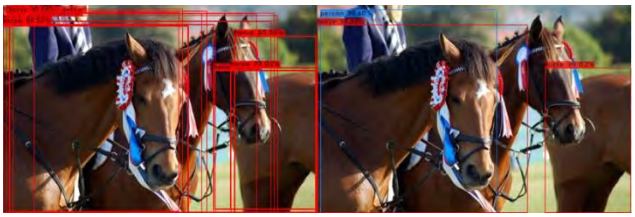
실험 결과, mAP 는 608×608 기준으로 0.7~1.6% 상승하였으며, 96×96 보다 큰 객체에 대한 AP(L)도 3.3~4.3% 상승하였고, 처리속도 또한 30fps 이상으로 실시간 탐지를 만족하였다. 그러나, COCO 데이터에서 작은 객체에 해당하는 32×32 이하의 크기의 객체에 대한 AP(S)는 0~0.8% 감소하였다. 이는 384×384 의 해상도의 작은 객체에 대한 탐지 성능이 608×608 보다 7.7% 좋지 않기 때문에, 384×384 의 작은 객체 탐지 결과가 앙상블한 작은 객체에 대한 AP(S) 감소에 영향을 준다. 따라서 384×384 의 bounding box 들 중 크기가 32×32 보다 작은 bounding box 들을 제거하는 추가 실험을 수행하였고, 32×32 보다 작은 객체에 대한 AP 가 608×608 과 비교하였을 때 0.3~0.7% 증가함을 확인하였다(표 3 참조). 또한 처리속도가 유사한 800×800 과 비교하였을 때, 앙상블 방법은 mAP 와 AP(L)을 각각 2.4~3.1%, 23.7~24.7% 개선할 수 있음을 확인하였다.

<표 3> 표 2 에서 384×384 의 32×32 보다 작은 bounding box 들을 추가로 제거한 결과

NMS	mAP	AP(S)	AP(M)	AP(L)	FPS
Greedy [1]	44.4%	27.0%	47.0%	56.6%	**36.3**
Diou [5]	44.8%	**27.4%**	47.5%	56.9%	36.2
Diou + WBF[6]	**45.1%**	**27.4%**	**48.2%**	**57.6%**	35.5

3. Geometric NMS

본 연구에서는 탐지 앙상블의 정확도를 더욱 개선하기 위하여, 새로운 NMS 방법인 G-NMS(Geometric NMS)을 제안한다. 기존 NMS 방식은 동일한 class 에 대하여 confidence score 가 높은 순서대로 bounding box 들을 정렬한 후에, 가장 confidence 가 높은 bounding box 와 IOU 가 일정 이상인 bounding box 들은 동일한 물체를 탐지한 것으로 판단하고 제거한다. 예를 들어 그림 2 와 같이, NMS 전에는 중간의 말을 탐지했지만, NMS 이후에는 중간의 말을 탐지한 bounding box 들이 전부 사라지는 경우가 발생한다. 이는 IOU 가 두 bounding box 간의 중복 영역의 비율을 나타낼 뿐, 두 bounding box 의 기하학적인 중심거리와 종횡비는 고려하지 못하기 때문이다. 즉, 객체에 해당하는 bounding box 들을 생성했지만, NMS 과정에서 bounding box 를 제거함으로써 TP(True-Positive)가 감소되고 전체 mAP 가 감소되는 문제가 발생한다.



(그림 2) NMS 전후 비교(기존 NMS 방식을 사용하면 동일한 클래스의 물체들이 겹친 경우 중복으로 간주하여 에러 발생)

이러한 기존 방식의 문제를 해결하기 위하여 IOU 를 쓰지 않는 G-NMS 방식을 제안한다. G-NMS 방식은 기하학적 중심 거리와 bounding box 의 종횡비를 고려한다. 먼저, bonding box 들간의

기하학적 중심거리는 기존 Diou-NMS[5]에서 사용된 penalty term 을 사용하였다. Diou-NMS 의 penalty term 은 두 bounding box 간의 중심 거리와 두 bounding box 를 감싸는 큰 box 의 대각선의 비율을 사용하기 때문에 중심 거리를 0~1 사이로 표준화가 가능하다. 종횡비는 두 bounding box 간의 너비와 높이의 비율을 각각 작은 값을 큰 값으로 나눔으로써 0~1 사이로 표준화한 값을 사용하였다. 기하학적 중심 거리와 종횡비 정보를 동시에 사용함으로써 bounding box 들이 같은 객체를 탐지한 것인지에 대한 평가를 할 수 있다. 표 4 는 기존의 NMS 방식과 G-NMS 방식의 결과를 비교한 것이며, 표 5 는 G-NMS 의 세부적인 실험 결과를 나타낸다.

<표 4> 표 3 의 결과를 G-NMS 의 성능과 비교한 결과

NMS	Greedy[1]	Diou[5]	Diou+WBF[6]	G-NMS
mAP	44.4%	44.8%	45.1%	**45.3%**
mAP(50)	66.4%	**66.5%**	66.4%	65.4%
mAP(75)	48.3%	48.8%	49.5%	**50.5%**
AP(S)	27.0%	27.4%	27.4%	**27.6%**
AP(M)	47.0%	47.5%	48.2%	**48.6%**
AP(L)	56.6%	56.9%	57.6%	**57.8%**
AR(1)	34.9%	34.9%	35.0%	**35.1%**
AR(10)	55.7%	56.5%	56.4%	**57.2%**
AR(100)	58.9%	60.1%	59.9%	**61.2%**
AR(S)	39.6%	**40.6%**	39.6%	40.3%
AR(M)	62.4%	63.9%	64.6%	**66.2%**
AR(L)	74.3%	75.6%	75.4%	**76.9%**
FPS	**36.3**	36.2	35.5	35.5

<표 5> 종횡비는 0.6 으로 고정하고 Diou 값을 달리 했을 때의 G-NMS 성능

Diou	0.1	0.07	0.05
mAP	45.3%	45.3%	45.1%
mAP(50)	65.4%	64.7%	63.9%
mAP(75)	50.5%	50.7%	50.7%
AP(S)	27.6%	27.6%	27.6%
AP(M)	48.6%	48.5%	48.4%
AP(L)	57.8%	57.8%	57.6%
AR(1)	35.1%	35.0%	35.0%
AR(10)	57.2%	57.5%	57.5%
AR(100)	61.2%	61.7%	62.0%
AR(S)	40.3%	40.4%	40.4%
AR(M)	66.2%	66.7%	67.1%
AR(L)	76.9%	77.8%	78.1%

실험 결과, G-NMS 는 전체적으로 mAP 가 0~0.2% 향상됨을 확인하였다. 또한, IOU 가 0.5 일 때 mAP(50)이 1% 감소되었으나, IOU 가 0.75 일 때는 mAP(75)가 1% 증가함을 확인하였다. 또한, 큰 객체에 대한 AP(L)는 0~0.2% 내외로 큰 변동이 없었으나, recall AR 은 1.5~2.7%가 향상됨을 확인하였다. 이는 G-NMS 가 기존의 NMS 방식보다 bounding box 간의 상관관계를 더 잘 파악하여 겹친 물체들의 탐지 bounding box 의 제거를 방지해줌을 의미한다. G-NMS 의 알고리즘은 다음과 같다.

Algorithm 1. G-NMS

Input
B: Detected boxes with different resolutions
(Each box has x,y,w,h and confidence score)
W: Threshold for width ratio of two boxes
H: Threshold for height ratio of two boxes
Output
Detected boxes processed by G-NMS

```
Sort B in order of high confidence score
for i = 0 to number of B do
    if B[i]'s confidence score is 0 do
        continue
    for j = i + 1 to number of B do
        if B[i].w > B[i].w do
            ratio_w = B[i].w/B[j].w
        else do
            ratio_w = B[j].w/B[i].w
        if B[j].h > B[i].h do
            ratio_h = B[i].h/B[j].h
        else do
            ratio_h = B[j].h/B[i].h
        if (diou(B[i], B[j]) < D && ratio_w > W
          && ratio_h > H) do
            change confidence score of B[j] to zero
```

<표 6> COCO 데이터로 측정한 기존 방법(YOLOv4[4] 및 SNIPER[2])과 제안 방법의 성능 비교

방법	해상도	mAP	AP(S)	AP(M)	AP(L)	FPS
YOLOv4 [4]	384×384	40.3%	19.0%	43.6%	56.1%	**91**
	608×608	43.5%	26.7%	46.7%	53.5%	59
	800×800	42.0%	28.8%	46.3%	32.9%	39
	제안방법 384×384 & 608×608 앙상블 + G-NMS	45.3%	27.6%	48.6%	57.8%	36 (2080Ti)
SNIPER [2]	512×408 & 1280×800 & 2000×1400 앙상블	**46.1%**	**29.6%**	**48.9%**	**58.1%**	5 (V100)

마지막으로 앙상블 방식과 G-NMS 방식을 결합한 제안 방법과 기존 방법(YOLOv4[4] 및 SNIPER[2])을 비교하면 표 6 과 같다. 특히 다양한 크기의 객체를 모두 탐지하기 위해 제안된 SNIPER 와 비교하여도, 제안 방법의 정확도가 크게 차이 나지 않음을 확인할 수 있다. 반면 V100([2]에서 측정된 GPU) 보다 성능이 크게 떨어지는 2080Ti 에서 측정된 처리속도임을

감안할 때, 제안 방법은 SNIPER 에 비하여 수십 배 빠른 처리속도를 제공할 수 있는 매우 효과적인 방법이라 판단된다.

4. 결론

현재 객체 탐지 분야는 mAP 를 향상시키기 위해 작은 객체 탐지에 주력하고 있다. 그러나 작은 크기의 객체를 잘 잡기 위해서 이미지 해상도를 증가시킬 경우, 큰 객체를 잘 탐지하지 못하는 trade-off 가 발생한다. 본 연구에서는 이러한 문제를 해결하기 위하여 실시간 처리를 만족하는 효과적인 이미지 피라미드 방식의 앙상블 방법을 제안하였다. COCO 데이터로 YOLOv4 에 적용 결과, 384 및 608 해상도의 객체 탐지 bounding box 들을 앙상블함으로써 작은 객체의 탐지 성능은 유지되면서 큰 객체의 탐지 성능은 향상됨을 확인하였다.

또한, 기존 NMS 방식은 두 bounding box 간의 기하학적인 거리와 종횡비는 고려되지 않은 채, 중복 bounding box 들을 제거하였다. 이는 두 bounding box 가 탐지한 물체가 다름에도 불구하고 하나의 bounding box 를 제거함으로써, True-Positive 를 감소시켰고, 따라서 mAP 가 감소되는 문제가 있었다. 본 연구에서는 이러한 문제를 해결하기 위하여 IOU 가 아닌 Diou penalty term 과 종횡비를 고려하는 새로운 NMS 방식인 G-NMS 를 제안하였다. 제안된 G-NMS 방식은 대부분의 AP 및 AR 정확도가 개선되는 효과가 있음을 확인하였다.

감사의 글

이 논문은 2018년도 교육부의 재원으로 한국연구재단의 기초연구사업(2018R1D1A1A09081924)과 2019년도 과학기술정보통신부의 재원으로 한국연구재단-현장맞춤형 이공계 인재양성 지원사업(No. 2019H1D8A1109907)의 지원을 받아 수행된 연구임.

참고문헌

[1] L. Liu, et al., “Deep Learning for Generic Object Detection: A Survey,” *International Journal of Computer Vision*, Vol. 128, pp. 261-318, 2020.

[2] B. Singh, et al., “SNIPER: efficient multi-scale training,” *In Proceedings of the NeurIPS*, pp. 9333–9343, 2018.

[3] T. Lin, et al., “Microsoft COCO: Common Objects in Context,” *In Proceedings of the European Conference on Computer Vision*, pp. 740–755, 2014.

[4] A. Bochkovskiy, et al., “YOLOv4: Optimal Speed and Accuracy of Object Detection,” *arXiv preprint arXiv:2004.10934*, 2020.

[5] Z. Zheng, et al., “Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression,” *In Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[6] R. Solovyev, et al., “Weighted Boxes Fusion: Ensembling Boxes for Object Detection Models,” *In Proceedings of Computer Vision and Pattern Recognition*, 2019.