Deformable Convolution 을 활용한 Cost-Efficient Feature 추출

안현석 ○, 박종운, 차재혁†
한양대학교 컴퓨터·소프트웨어학과
pcar530@gmail.com, uxdesign123@hanyang.ac.kr, chajh@hanyang.ac.kr

요 약

그동안 로봇 및 임베디드 시스템 분야에서는 제한된 컴퓨팅 환경에서 computer vision 작업 성능 향상이 지속적인 과제였다. 본 연구는 연산량, 처리 시간, 메모리 사용량과 같은 컴퓨팅 자원이 제한된 환경에서 image classification, object detection 에서 정확도를 개선하는 것을 목표로 한다. 이를 수행하기 위해 특정 조건에서 Deformable Convolution Network (DCN) 과 Transformer 의 Self-attention layer 간의 관계를 확인한다. 이러한 관계를 토대로, 본 연구는 기존의 모델 아키텍처에서 일부 Convolution 과 Self-attention (attention)을 최근 DCN 의 단점을 개선한 Deformable convolution Network version 4 (DCNv4)로 대체하는 방법을 제안한다. 본 논문에서는 제안한 방법의 효과를 검증하기 위해 resnet18 과 tiny-imagenet dataset 을 활용한 image classification 실험, You Only Look Once (YOLO)의 여러 모델과 ms coco2017 dataset 을 활용한 object detection 실험을 진행한다. 그 결과 accuracy 기준 resnet18 에 DCN(resblock)을 적용한 결과 +0.6, resnet18+attention 에 DCN(attention)을 적용한 결과 +1.2, DCN(resblock, attention)을 적용한 결과 +2.7 향상되었다. 또한 mAP 기준 YOLO8n 에 DCN(C2f)를 적용한 결과 +0.5, YOLO10n 에 DCN(c2f)를 적용한 결과 +0.9, DCN(attention)를 적용한 결과 +0.5, DCN(c2f, attention)을 적용한 결과 +0.7 향상되었다. 이 결과로 DCNv4 를 활용한 약간의 변형으로 기존의 모델의 성능을 향상시킬 수 있다는 것을 확인했다.

1. 서론

인공지능의 발전으로 자율 주행 자동차, 및 공장 자동화 시스템이 잇따라 발전하고 있다. 이러한 로봇 및 임베디드 시스템 분야에서는 제한된 컴퓨팅 자원을 활용하여 최대 성능(accuracy, mAP)과 빠른 연산 시간(processing time)을 달성하는 것을 목표로 해왔으며 computer vision 분야에서는 real-time object detection, image classification 이 대표적인 예시이다. 이러한 목표를 달성하기 위해 많은 연구가 진행되어 왔다.

먼저 전통적인 Convolution Neural Network (CNN)을 활용해 모델의 구조 설계한 CNN 기반 object detection 모델로 R-CNN based models[1, 2], YOLO series models[3, 4, 5, 6, 7]이 있으며 당시 state-of-the-art 성능을 달성했다.

한편 Natural Language Processing (NLP) 분야 에서는 transformer[8]를 활용한 연구를 통해 많은 성과를 달성해왔다. Computer vision 에서도 transformer[8]의 self-attention 을 real-time object detector 에 활용해 성능을 향상시키기 위한 연구가 진행된 결과 YOLO series models[9], RT-DETR[10]는 현재 state-of-the-art 성능을 달성했다.

이와 동시에 CNN 의 변형을 통해 성능을 개선하기 위한 연구 또한 활발히 진행되어왔으며, dilated convolution network[11], depth-wise convolution network[12], deformable convolution network (DCN)[13]가 발표되었지만 real-time object detection 에서는 부족한 성능 및 속도로 인해 잘 사용되지 않았다. 하지만 최근 Deformable Convolution Network version 4 (DCNv4)[14]를 통해 DCN 의 성능 및 속도 모두 개선하였다.

본 연구는 이러한 DCN 과 self-attention 의 계산 과정이 특정 제약조건에서의 유사함을 확인하고, 이를 바탕으로 제한된 환경에서 모델의 일부 convolution block, self-attention block 을 DCNv4 를 활용한 DCN block 으로 대체하여 성능을 향상하고자 한다. 그리고 제안한 방법의 성능을 Resnet18[15]을 활용한 image classification 실험과, YOLO series models[7, 9]의 nano 크기를 활용해 검증한다.

2. Methods

Self-attention 과 DCN 의 계산 과정을 확인하고, 특정 제약조건에서는 self-attention 이 DCN 과의 관계를 확인한다. 이를 활용하여 기존모델의 일부 convolution block, self-attention block 을 DCN block 으로 대체하여 모델의 성능을 향상시키고자 한다.

2.1 Self-attention

$x \in \mathbb{R}^{H \times W \times C}$, height H, width W, channel C 와 같이 입력이 주어질 때, transformer 의 self-attention 의 계산 과정은 아래와 같다.

$$F = \text{PW\_Conv}(x)$$
$$Q, K, V = \text{Split}(F)$$
$$Attn\_weight = \frac{\text{Matmul}(Q, K^T)}{\sqrt{d}}$$
$$Attn\_score = \text{Softmax}(attn\_weight)$$
$$Attn\_value = \text{Matmul}(atten\_score, V)$$

즉 x 를 point-wise convolution 에 통과시켜 F 를 계산하고 이를 분리해 *query*(Q), *key*(K), *value*(V)를 구한다. 이후 Q 와 K 의 곱을 사용해 *attention weight* 를 계산하며, SoftMax 를 사용한 정규화를 통해 *attention score* 를 계산한다. 최종적으로 *attention score* 와 V 를 곱해 출력 값인 *attention value* 를 계산한다. 이러한 self-attention 방식은 long-range dependence 와 adaptive spatial aggregation 의 장점을 가지지만, 높은 computation/memory 사용의 단점을 가진다.

2.2 Deformable Convolution

$x \in \mathbb{R}^{H \times W \times C}$, height H, width W, channel C 와 같이 입력이 주어질 때, DCN 의 계산 과정은 아래와 같다.

$$F = \text{PW\_Conv}(x)$$
$$V = \text{PW\_Conv}(x)$$
$$X, Y, W = \text{Split}(F)$$
$$I_i = \text{Interp}(V, [X_i, Y_i])$$
$$Value = \text{Matmul}(W, I)$$

즉 x 를 2 개의 point-wise convolution 에 통과시켜 F 와 V 를 계산하고 F 를 분리해 같은 크기의 X, Y, W 를 구한다. 이후 X, Y 를 상대 좌표로 사용해 feature 를 추출하고 보간법을 사용해 값을 보정하여 I 를 계산한다. 최종적으로 앞서 구한 W, I 를 곱해 출력 값인 *Value* 를 계산한다. 이러한 deformable convolution 방식은 long-range dependence 와 adaptive spatial aggregation 과 효율적인 computation/memory 사용의 장점을 가지며, F 의 크기인 i 가 DCN 의 kernel size 와 비례하며, group 내의 channel 은 같은 X, Y, W 를 공유하는 특징이 있다.

2.3 Self-attention 과 Deformable Convolution

Self-attention 에서 N 개의 attention score 중 k개를 제외한 $N - k$개가 0 에 가깝다는 제약조건을 가정한다. 이러한 제약조건 아래에서 self-attention 과 DCN 은 아래의 그림과 수식으로 표현할 수 있다.

위와 같은 제약조건 내에서는 [그림 2], [그림 3] 과 같이 attention 의 파라미터가 sparse matrix 의

Self − Attention

$$score_i = \text{Softmax}(\frac{q_i^T K}{\sqrt{d}})$$
$$f_i = [score_i \times V_j]_{j=1,\dots,gc}$$

$N = H \times W$
$k = (\text{kernel\_size})^2$
$d =$ key dimention
$gc =$ group_channel size

DCN

$$I = [\text{Interp}(p_0 + p_j + p_{x_j, y_j})]_{j=1,\dots,k}$$
$$f_i = [w_i \times I_m]_{m=1,\dots,gc}$$

그림 2. Self-attention 과 DCN 의 계산식 일부

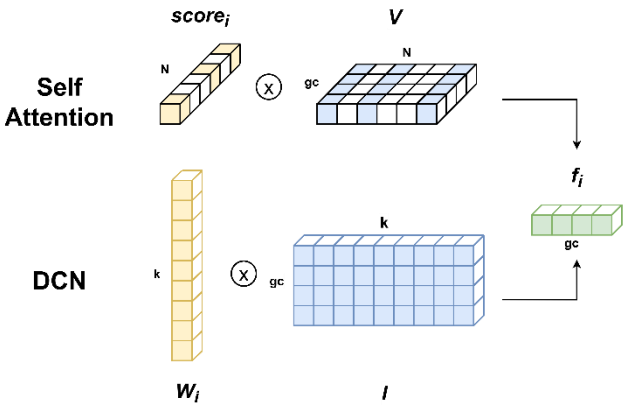

그림 3. Self-attention 과 DCN 의 계산식 일부 그림

dense representation 방식으로 sparse matrix 를 학습하는 것을 확인할 수 있으며, DCN 의 파라미터가 sparse matrix 의 sparse representation 방식으로 sparse matrix 를 학습한다는 것을 알 수 있다. 결과적으로 attention 과 DCN 모두 다른 표현 방법으로 sparse matrix 를 학습하는 것을 알 수 있다.

3. 실험 및 분석

본 실험에서는 제안한 방법의 성능을 image classification 과 object detection 작업을 통해 실험한다. 두 실험 모두 backbone 의 마지막 convolution block, attention block 을 DCNv4 를 기반으로 한 DCN block 으로 대체해 비교 실험한다. 실험 간 사용 장비는 CPU: 17-13700K, GPU: RTX 4090 1-way 를 사용하여 실험하며, 실험 코드는 Ultralytics, DCNv4 의 코드를 사용해 실험한다.

3.1 Image Classification

Image classification 실험은 resnet18 와 Tiny-ImageNet dataset 을 활용해 실험한다. 다만 기존 resnet18 모델의 크기를 이후 실험의 모델 크기와 비슷하게 맞추기 위해 64 channel size 기반 resnet18 을 32 channel size 기반 resnet18 로 변형하였다. 실험 간 backbone 의 마지막 convolution block 을 DCN block 으로 대체하였으며, attention 을 backbone 뒤에 추가하여 실험하고, DCN block 으로 대체하여 비교 실험한다.

표 1. Image Classification 성능 비교

Models	Modules	Params (M)	Accuracy	Process-time (ms)	FPS
Resnet18		1.454	46.2	1.261	793
DeCo-Resnet18	+DCN(Resblock)	1.435(-0.019)	**46.8**(+0.6)	1.405(+0.144)	711
PSA-Resnet18	+Attention	1.704	47.3	1.639	610
PSD-Resnet18	+DCN(Attention)	1.742(+0.038)	**48.5**(+1.2)	1.583(-0.056)	631
DeCo-PSA-Resnet18	+DCN(Resblock), Attention	1.685	48.6	1.880	531
DeCo-PSD-Resnet18	+DCN(Resblock, Attention)	1.723(+0.038)	**48.9**(+0.3)	1.795(-0.085)	557

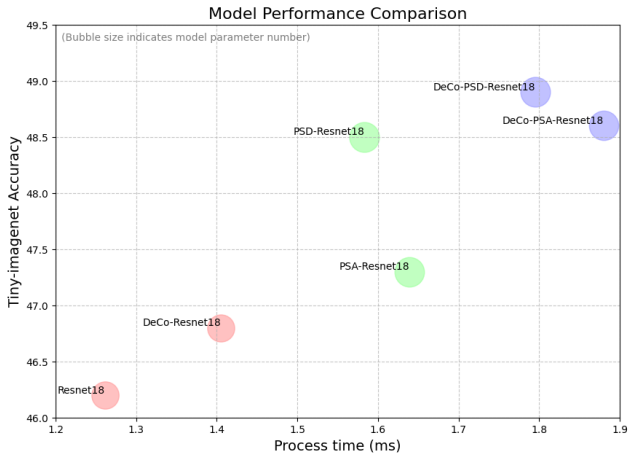

그림 4. Imgae Classification 성능 비교

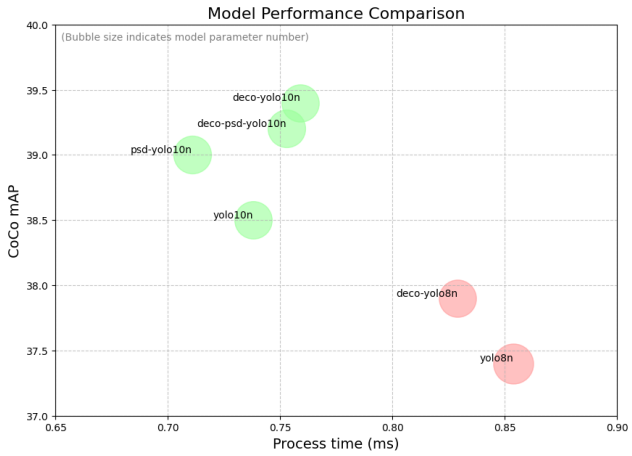

그림 5. Object Detection 성능 비교

[표 1], [그림 4]에서 확인할 수 있듯이 PSA-Resnet18 은 Resnet18 대비 process time 이 +0.378ms, parameter 수는 +0.25M, accuracy 는 +1.1 변화하였다. 이를 통해 attention block 의 성능 향상 효과를 확인할 수 있다. DeCo-Resnet18 은 Resnet18 대비 process time 이 +0.144ms, parameter 수는 -0.019M, accuracy 는 +0.6 변화하였고. DeCo-PSA-Resnet18 은 PSA-Resnet18 대비 process time 이 +0.241ms, parameter 수는 -0.019M, accuracy 는 +1.3 변화하였다. 이를 통해 convolution block 을 대체한 DCN block 의 성능 향상 효과를 확인할 수 있다. PSD-Resnet18 는 PSA-Resnet18 대비 process time 이 -0.056ms, parameter 수는 +0.038M, accuracy 는 +1.2 변화하였다. DeCo-PSD-Resnet18 는 DeCo-PSA-Resnet18 대비 process time 이 -0.085ms, parameter 수는 +0.038M, accuracy 는 +0.3 변화하였다. 이를 통해 attention block 을 대체한 DCN block 의 성능 향상 효과를 확인할 수 있으며, convolution block 과 attention block 을 모두 DCN block 으로 교체한 DeCo-PSD-Resnet18 이 빠른 속도로 가장 높은 정확도를 달성한 것을 알 수 있다.

3.2 Object Detection

Object Detection 실험은 YOLO8n, YOLO10n 모델과 ms coco2017 dataset 을 활용해 실험한다. 마찬가지로 backbone 의 마지막 convolution block 을 DCNv4 block 으로 대체해 비교 실험하며, attention block 을 DCN block 으로 대체하여 비교 실험한다.

[표 2], [그림 5]에서 확인할 수 있듯이 YOLO8n 에 DCN block 을 적용한 DeCo-YOLO8n 는 process time 이 -0.025ms, parameter 수는 -0.427M, mAP 는 +0.5 변화하였다. YOLO10n 에 DCN block 을 적용한 DeCo-YOLO10n 은 process time 이 +0.028ms, parameter 수는 -0.021M, mAP 는 +0.9 변화하고, PSD-YOLO10n 은 process time 이 -0.027, parameter 수는 +0.039M, mAP 는 +0.5 변화하였고, DeCo-PSD-YOLO10n 은 process time 이 +0.015, parameter 의 수가 0.018M, mAP 가 +0.7 변화하였다. 이를 통해 YOLO8n, YOLO10n 의 convolution block 과 attention block 을 DCN block 으 대체함으로써 여전히 빠른 속도와 향상된 정확도를 달성한 것을 확인할 수 있다.

4. 결론

본 논문에서는 특정 조건에서 self-attention 와 DCN 의 계산 과정이 유사함을 보인다. 이를 활용한 image classification, object detection 실험에서 일부 convolution block 과 attention block 을 DCNv4 를 기반으로 한 DCN block 으로 대체하여 성능을 확인한다. 이는 제한된 환경에서 기존의 모델의 일부에

표 2. Object Detection 성능 비교

Models	Modules	Params (M)	mAP	Process-time (ms)	FPS
YOLO8n		3.152	37.4	0.854	1170
DeCo-YOLO8n	+DCN(C2f)	2.725(-0.427)	**37.9**(+0.5)	0.829(-0.025)	1206
YOLO10n	+Attention	2.762	38.5	0.738	1355
DeCo-YOLO10n	+DCN(C2f)	2.741(-0.021)	**39.4**(+0.9)	0.759(+0.021)	1317
PSD-YOLO10n	+DCN(Attention)	2.802(+0.039)	39.0(+0.5)	0.711(-0.027)	1406
DeCo-PSD-YOLO10n	+DCN(C2f, Attention)	2.781(+0.018)	39.2(+0.7)	0.753(+0.015)	1329

DCNv4 를 적용해 성능 개선에 기여할 수 있음을 시사한다.

5. 한계

본 논문에서는 DCNv4 의 코드를 활용하여 실험을 진행했다. 하지만 DCNv4 에서는 feature 에서 몇 개의 feature 를 사용하는지를 의미하는 kernel size 가 3 일 경우만 지원하는 점과 DCN 의 같은 X, Y, W 를 공유하는 channel 의 수를 의미하는 group channel size 가 8 이상만 지원한다는 점에서 한계가 있다. 이러한 부분이 해결된다면 보다 성능을 개선할 수 있을 것으로 기대된다.

감사의 글

본 연구는 2018 년도 한국연구재단의 지원에 의하여 이루어진 연구로서, 관계부처에 감사드립니다 (2018R1A5A7059549).

참고문헌

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," In Advances in Neural Information Processing Systems (NIPS), pp. 91-99, 2015.

[2] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 2980-2988, 2017.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779-788, 2016.

[4] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7263-7271, 2017

[5] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv preprint arXiv:1804.02767, 2018.

[6] C. Wang, A. Bochkovskiy, and H. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7464-7475, 2023.

[7] Glenn Jocher, YOLOv8, https://github.com/ultralytics/ultralytics, 2023.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS), pp. 6000-6010, 2017.

[9] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, and G. Ding, "YOLOv10: Real-Time End-to-End Object Detection," In Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS), 2024

[10] Y. Zhao, B. Zhang, W. Wang, J. Xu, Y. Xiong, Y. Bai, W. Ouyang, and D. Xu, "DETRs Beat YOLOs on Real-time Object Detection," In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 16965-16974, 2024.

[11] F. Yu and V. Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions," In Proceedings of the International Conference on Learning Representations (ICLR), 2016.

[12] L. Sifre and S. Mallat, "Rigid-Motion Scattering for Image Classification," Ph.D. thesis, École Polytechnique, 2014.

[13] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable Convolutional Networks," In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 764-773, 2017.

[14] Y. Xiong, Z. Li, Y. Chen, F. Wang, X. Zhu, J. Luo, W. Wang, T. Lu, H. Li, Y. Qiao, L. Lu, J. Zhou, and J. Dai, "Efficient Deformable ConvNets: Rethinking Dynamic and Sparse Operator for Vision Applications," In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5652-5661, 2024.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778, 2016.