임베디드 보드 환경에서 실시간 객체 탐지를 위한 필터 프루닝 연구

서종웅[1], 안한세[2], 손승욱[3], 정용화[4]
[1]고려대학교 컴퓨터융합소프트웨어학과 석사과정
[2]고려대학교 컴퓨터융합소프트웨어학과 박사과정
[3]인포벨리코리아 기업부설 인공지능 세종연구소 사원
[4]고려대학교 컴퓨터융합소프트웨어학과 교수
seojongwoong@korea.ac.kr, hansahn@korea.ac.kr, sso7199@invako.kr
ychung@korea.ac.kr

A Study on Filter Pruning for Real-Time Object Detection in Embedded Board Environments

Jongwoong Seo*, Hanse Ahn*, Seungwook Son**, Yongwha Chung*
*Dept. of Computer Convergence Software, Korea University
**INFO VALLY KOREA Co., Ltd

요 약

딥러닝 기술은 더 많은 분야와 과제에 적용되기 위해서 네트워크는 더 복잡하고 거대한 형태로 발전해왔다. YOLOv7-tiny과 같은 객체탐지 네트워크는 다양한 객체와 환경에서 활용하기 위해 COCO 데이터 세트를 대상으로 발전해왔다. 그러나 본 논문에서 적용할 모델은 임베디드 보드 환경에서 실시간으로 1개의 Class를 대상으로 객체를 탐지하는 네트워크 모델이 찾고자 프루닝을 적용하였다. 모델의 프루닝을 할 필터를 찾기 위해 본 논문에서는 클러스터링을 통한 필터 프루닝 방법을 제안한다. 본 논문의 제안 방법을 적용했을 때 기준 모델보다 정확도가 7.6% 감소하였으나, 파라미터가 1% 미만으로 남고, 속도는 2.1배 증가함을 확인하였다.

1. 서론

신경망(Neural Network) 모델은 컴퓨터 비전 분야에서 사용되었으며, 시간이 지남에 따라 CNN(Convolution Neural Network)과 같은 구조로 발전되었다[1]. 이에 따라 모델의 정확도는 크게 향상되었지만, 반대로 많은 레이어(Layer)와 필터(Filter)가 사용되게 되었으며, 많은 연산량이 필요로 한다.

네트워크 모델이 발전하면서 다양한 객체와 환경에서 범용적으로 활용하기 위해 80개의 Class를 가진 COCO 데이터 세트[2]가 제안되었다. COCO 데이터 세트[2]를 대상으로 사용하는 YOLOv7-tiny[3]와 같은 모델은 많은 파라미터를 가진다.

그러나 이러한 모델은 1개의 Class를 대상으로 제한된 임베디드 보드 환경에서 실시간 탐지를 수행하는 데 적합하지 않다. 따라서 본 논문에서는 제한된 임베디드 보드 환경에서 실시간 탐지를 수행하는 데 적합한 모델을 찾기 위해서 YOLOv7-tiny[3]에 프루닝을 적용하여 파라미터를 줄이고 실제적인 탐지 속도(FPS)를 개선하고자 한다.

2. 관련 연구

임베디드 보드 환경에서는 자원이 한정되어 있어 낭비되는 계산이나 메모리를 최적화할 필요가 있다[4]. 이를 해결하기 위해 많은 연구자가 모델에서 최종결과에 영향을 적게 미치는 컨볼루션 레이어의 필터를 제거하는 연구를 진행했었으며, 이러한 방법을 프루닝(Pruning)이라고 한다. 프루닝 방법은 크게 비구조적 프루닝(Unstructured Pruning)과 구조적 프루닝(Structured Pruning)으로 나눠진다.

먼저 프루닝 연구 중 비구조적 프루닝은 특정 기준에 따라 가중치를 0으로 하는 방법이다. 절댓값이 낮은 가중치들은 최종 결과물에 적게 반영된다. Han et al. [4]은 이러한 가중치를 제거하기 위해서 L1-norm을 적용하였다. 또한 반복적인 프루닝을 통해 네트워크 구조를 찾고, 프루닝을 통해 떨어진 정확도를 재학습해 정확도를 보완하는 방법을 제안했다. 그러나, He et al. [5]는 비구조적 프루닝은 이론

적으로는 파라미터는 줄어들지만, 실질적으로는 탐지 속도 향상에 많은 도움을 줄 수 없음을 확인하였다. 반면 구조적 프루닝은 레이어나 필터를 제거함으로써 제거된 레이어나 필터에 대해 연산하지 않기 때문에 실제적인 모델의 탐지 속도를 효과적으로 향상될 수 있음을 확인했다. 또한, Li et al. [6]은 구조적 프루닝 방법을 제안하였으며, 레이어의 필터들에 L1-norm을 적용해 가중치의 절댓값의 합을 계산해서 필터의 중요도를 평가해 프루닝을 적용하였다. 본 논문에서는 특징맵의 정답 영역과 정답 영역이 아닌 부분의 정보를 활용하여 클러스터링을 통해 대표 특징맵을 제외한 나머지 특징맵을 생성하는 필터를 프루닝하는 방법을 제안하고자 한다.

3. 제안 방법

본 논문의 제안 방법은 학습, 프루닝, 그리고 재학습 단계로 진행된다. 먼저 기준 모델을 설정하기 위해 네트워크에 미리 학습된 가중치를 이용해 학습한다. 다음 단계로 프루닝을 적용할 레이어를 선택하는 방법으로 로컬 프루닝과 선택된 레이어에서 프루닝을 적용해 제거하는 필터 개수를 선택하는 방법으로 레벨 프루닝을 적용한다. 또한 프루닝 할 필터를 선택하기 위해 제안 방법인 특징맵 클러스터링을 적용한다. 마지막으로 재학습을 통해 프루닝으로 손실된 정확도를 보완한다.

3.1 로컬 프루닝과 레벨 프루닝

구조적 프루닝은 어느 레이어를 얼마나 프루닝 할지 정해야 할 필요가 있다. 이를 선택하기 위해 로컬 프루닝과 레벨 프루닝을 사용한다. 로컬 프루닝은 모든 레이어에 같은 프루닝 비율이 각각 적용되며, 레벨 프루닝은 레이어의 필터 개수를 희소성 레벨로 설정해 레이어의 희소성이 특정 값이 되게 프루닝하는 방법이다.

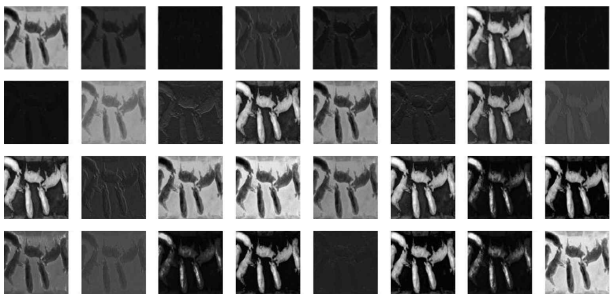
L^i는 i번째 레이어라고 하고, F_l^i를 레이어 L^i의 l번째 필터라고 하고, L^i에서 프루닝 할 필터 개수를 P^l 라고 하고, 희소성을 S라고 할 때 아래 식으로 각각의 레이어마다 독립적으로 계산한다.

$$P^l = \left\lfloor \sum_{l=0}^{n} F_l^i - \sum_{l=0}^{n} F_l^i \times S \right\rfloor$$

3.2 특징맵 클러스터링

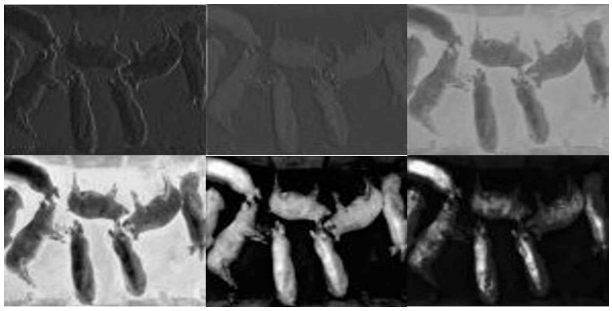
각각의 레이어마다 몇 개의 필터를 프루닝 할지 결정한 후에는 구체적으로 어떤 필터를 프루닝 할지 결정해야 할 필요가 있다. 본 논문에서는 필터를 통해 추출되는 특징맵의 유사도를 이용해서 어떤 필터를 프루닝 할지 결정하는 방법을 제안하고자 한다.

그림 1은 0번째 레이어에서 생성된 32개의 특징맵이다. 그림 1의 특징맵과 같이 하나의 레이어에서 유사한 특징맵이 관찰됨을 확인하였다. 유사한 특징맵을 K-mean++ 클러스터링을 통해 그룹화하고 그 중 그룹을 대표할 수 있는 특징맵을 선택하기 위해 물체가 존재하는 정답 영역과 정답 영역을 제외한 영역의 픽셀값 평균과 분산을 구해 정답 영역의 분산이 가장 높은 것을 대표 특징맵으로 선택해 대표 특징맵을 만드는 필터를 제외한 나머지 필터에 프루닝을 적용한다. 분산이 가장 높은 것을 대표 특징맵으로 선택하는 이유는 정답 영역이 박스로 표현된다. 이때 영역 내의 물체와 물체가 아닌 부분의 경계가 분산이 큰 경우에 더 잘 나타낼 수 있다.



(그림 1) 기준 모델의 특징맵.

K-mean++ 클러스터링은 기계 학습의 비지도 학습에서 주로 사용되는 알고리즘 중 하나로, 데이터의 특징을 통해 패턴을 발견할 수 있고, 같은 특성을 가진 필터의 특징맵을 그룹으로 나누는데 사용할 수 있다. 기존의 K-mean 알고리즘은 중심점을 선택하는 기준이 무작위로 선택되기 때문에 이 알고리즘을 사용할 때마다 다른 그룹화가 이루어진다. 그러나 K-mean++ 알고리즘은 데이터 간의 거리를 고려해 중심점이 될 확률이 높은 것을 선택하기 때문에 대부분 같은 그룹화가 이루어진다. 그림 2는 제안 방법으로 기준 모델에 적용해 프루닝 하였을 때 살아남은 특징맵이다. 제안 방법을 적용하면 그림 2와 같이 제안 방법으로 그림 1에 존재하는 유사한 특징맵을 그룹화하고, 그룹화된 특징맵 중에서 대표 특징맵을 선택하게 되고, 이에 대응하는 필터 번호를 제외한 필터를 프루닝을 적용할 필터로 선택한다.



(그림 2) 제안 방법으로 기준 모델에서 남겨진 특징맵.

Feature-map clustering Algorithm

Input : Feature-map

Output : Useless Filter Index

Variable:

F=filters

S = Pruning Level

MG=Mean of Pixels Values on Ground-truth

VG=Variance of Pixels Values on Ground-truth

MB=Mean of Pixels Values on Background

VB=Variance of Pixels Values on Background

K=The Number of Centers

$$K = \left\lfloor \sum_{i=0}^{n} F_i - \sum_{i=0}^{n} F_i \times S \right\rfloor$$

Feature = (MG, VG, MB, VB)

do K-mean++

for 0 to Labeled Groups

if Max.VG == F_i.VG

Pruning except F_i

4. 실험

이 실험은 YOLOv7-tiny[3]의 네트워크를 사용해 본 논문은 커스텀 데이터 세트를 사용해 실험을 진행하였다. 특징맵을 생성하는 이미지는 검증 데이터 세트 중 하나를 사용하였다. 또한, 본 논문에서는 Nvidia Jetson TX2[7] 임베디드 보드 환경에서 실험을 진행하였다. 재학습할 때 사용한 LR(Learning Rate)는 학습에 사용된 LR의 1/10로 조정하여 실험을 진행하였다.

<표 1> 비교 모델[6]의 성능.

프루닝 레벨(%)	파라미터	정확도 (%)	속도	통합성능
0	6	93.2	20.2	1882.6
50	1.5M	89.0	39.5	3515.5
60	956.1K	88.2	39.9	3519.1
70	535.4K	88.0	41.2	3625.6
80	238.3K	86.4	42.1	3637.4
90	58.6K	86.5	43.7	3780

<표 2> 제안 방법의 성능.

프루닝 레벨(%)	파라미터	정확도 (%)	속도	통합성능
0	6M	**93.2**	20.2	**1882.6**
50	1.5M	**89.1**	39.5	**3519.4**
60	956.1K	**90.5**	39.9	**3610.9**
70	535.4K	**89.2**	41.2	**3675.0**
80	238.3K	**88.2**	42.1	**3713.2**
90	58.6K	**87.7**	43.7	**3832.4**

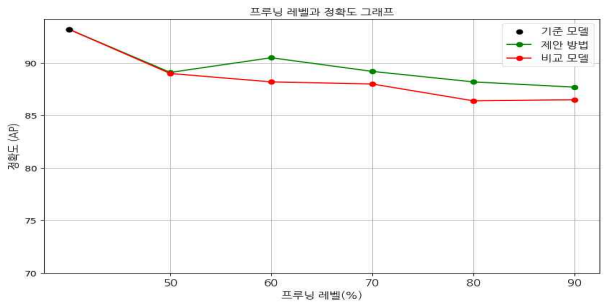
표 1은 비교 모델로서 Li et al. [6]의 방법을 적용하여 진행한 실험 결과를 의미하며 필터에 L1-norm을 적용해 필터의 중요도를 평가해 프루닝을 적용하여 실험을 진행하였다. 표 2는 본 논문의 제안 방법을 따라 프루닝이 적용된 모델의 실험 결과를 보여준다. 표 1과 표 2에 사용된 파라미터는 학습에 사용된 변수들의 총량을 의미하며, 파라미터가 많은 모델은 일반적으로 더 많은 데이터를 학습할 수 있지만 모델의 계산량이 많아져 탐지에 더 많은 시간이 소모된다. 또한, 표 1과 표 2에 사용된 정확도는 평균 정밀도(Average Precision)를 의미한다. 평균 정밀도는 정밀도(Precision)가 y축이고, 재현율(Recall)이 x축인, PR 곡선의 면적을 계산한 값을 의미한다. 이때 정밀도와 재현율은 다음과 같은 수식으로 표현된다.

$$\mathrm{Pr}escision = \frac{TP}{TP+FP},\ Recall = \frac{TP}{TP+FN}$$

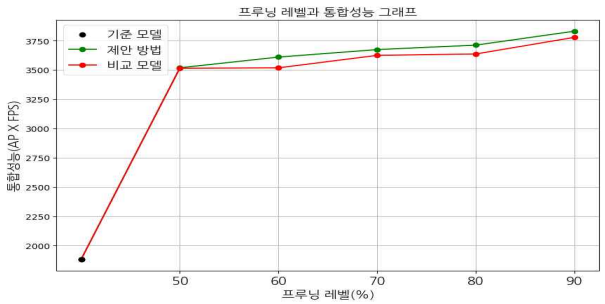
TP(True Positive)는 실제 True일 때, 모델에서 예측이 True라고 판단한 경우이다. FP(False Positive)는 실제 False일 때, 모델이 True라고 예측한 경우이다. FN(False Negative)은 실제 True일 때, 모델에서 예측이 False라고 판단된 경우이다. 속도의 경우 FPS(Frames Per Second)를 사용했는데, 이는 1초에 얼마나 많은 이미지를 처리할 수 있는가를 나타내는 지표로 일반적으로 실시간 탐지가 가능하기 위해서 1초에 30개의 이미지를 처리할 수 있어야 한다.

그림 3은 비교 모델[6]과 제안 방법의 프루닝 레벨과 정확도 그래프를 의미한다. 그림 3과 같이 비교 모델[6]은 프루닝 레벨 90%에서 정확도가 6.7% 감소하였고, 제안 방법은 5.5% 감소하였다. 그림 4는 비교 모델[6]과 제안 방법의 프루닝 레벨과 통합성능 그래프를 의미한다. 그림 4와 같이 비교 모델[6]와 제안 방법 모두 레벨이 단계가 증가하면서 통합성능이 향상되었다. 표 1과 표 2와 함께 비교해 보면 로컬 프루닝과 레벨 프루닝을 적용했기 때문에 프루닝 레벨 별 두 표의 파라미터와 속도가 같은 것

을 확인할 수 있었다. 90% 프루닝을 했을 때, 파라미터는 99% 감소하였고 속도는 2.1배 향상됨을 확인하였다.



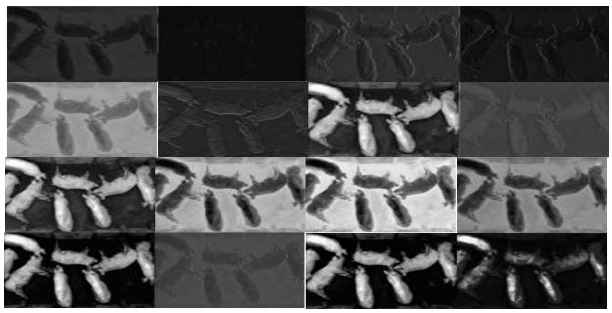
(그림 3) 비교 모델[6]과 제안 방법의 프루닝 레벨과 정확도 그래프.



(그림 4) 비교 모델[6]과 제안 방법의 프루닝 레벨과 통합성능 그래프.

5. 논의

그림 5은 프루닝 레벨 50%에서 선택된 특징맵을 나타낸다. 그림 5와 같이 프루닝 레벨 50%의 경우 선택된 특징맵들 중 탐지에 영향이 적을 것으로 추측되는 필터들도 선택되었다. 이에 따라 표 2의 프루닝 레벨 50%의 정확도가 프루닝 레벨 60%의 정확도와 비교하였을 때 1.4% 가 낮은 것으로 추측된다.



(그림 5) 프루닝 레벨 50%에서 선택된 특징맵.

6. 결론

YOLOv7-tiny[3] 네트워크 모델에 프루닝을 적용해 임베디드 보드 환경에서 실시간 객체 탐지가 가능한 특징맵 클러스터링 알고리즘을 이용해 프루닝을 적용하는 방법을 제안한다. 그러나 로컬 프루닝과 레벨 프루닝을 적용하여 모든 레이어들의 관계나 민감도를 고려할 수 없었다. 향후 연구로는 글로벌 프루닝을 사용하여 레이어 각각 프루닝을 적용하는 비율을 다르게 설정해 정확도를 보완하고자 한다.

감사의 글

이 논문은 2023년도 정부(산업통상자원부)의 재원으로 한국산업기술진흥원의 지원을 받아 수행된 연구임

(PP00241770, 2023년 지역혁신클러스터육성)

참고문헌

[1] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," Proceedings of the CVPR, 2015, pp. 1150-1210.

[2] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár and C Lawrence Zitnick, "Microsoft coco: Common objects in context," Proceedings of the ECCV, 2014, pp. 740-755.

[3] Chien-Yao 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,", Proceedings of the CVPR, 2023, pp. 7464-7475.

[4] Song Han, Jeff Pool, John Tran, and William Dally, "Learning both weights and connections for efficient neural network,", Proceedings of the NIPS, 28, 2015, pp. 1135-1143.

[5] Yang He and Lingao Xiao, "Structured Pruning for Deep Convolutional Neural Networks: A survey," arXiv preprint arXiv:2303.00566.

[6] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf, "Pruning filters for efficient convnets," Proceedings of the ICLR, 2017, pp. 1-13.

[7] NVIDIA Jetson TX2 4GB Module, Available online:https://developer.nvidia.com/embedded/jetson-tx2-4gb (accessed on February 2020).