Dynamic Sparse Training을 응용한 점진적 그룹단위 프루닝

이광한 ○, 신동군

성균관대학교 인공지능학과 ○, 성균관대학교 전자전기컴퓨터공학과

{ican0016, dongkun}@skku.edu

Gradual Group-level Pruning with Dynamic Sparse Training

Gwanghan Lee○, Dongkun Shin

Department of Artificial Intelligence, Sungkyunkwan University○, Department of Electrical and Computer Engineering, Sungkyunkwan University

요 약

Pruning 기법은 제한된 하드웨어 리소스를 가진 임베디드 장치에서 딥러닝 모델을 구동하기 위한 효율적인 방법이다. 그중 group-level pruning은 하드웨어에서 가속이 가능하고 높은 비율의 파라미터들을 pruning하면서 정확도를 유지할 수 있는 모델 압축 방법이다. 기존의 group-level pruning에서는 single global threshold를 기준으로 pruning 했기에 레이어마다 중요도가 다르다는 점을 고려하지 못했다. 본 논문에서는 학습가능한 threshold를 통해 레이어별 중요도를 고려할 수 있는 Dynamic Sparse Training(DST)에 group-level pruning을 적용했다. 또한 네트워크를 효과적으로 pruning 하기위해 pruning scheduler를 적용하여 gradual하게 pruning한 Gradual Group-level Pruning with DST(GGP-DST)를 제안한다. ResNet-20 모델에서 CIFAR10 데이터셋에 대해 검증하였을 때, GP-DST는 single global threshold를 사용하는 기존 group-level pruning 모델과 비교하여 85%, 90%, 95% 근처의 sparsity에서 정확도가 각각 0.51%, 0.86%, 2.53% 상승했고 GGP-DST는 기존 모델과 비교하여 85%, 90%, 95% 근처의 sparsity에서 정확도가 각각 1.53%, 1.61%, 2.74% 상승했다.

1. 서 론

DNN(Deep Neural Network)은 오늘날 컴퓨터 비전, 자연어처리, 음성인식 등의 분야에서 좋은 성능을 보인다. 더 높은 성능을 달성하기 위해 DNN 모델의 사이즈는 커져가고 그에 따라 많은 연산량과 에너지 비용을 필요로 하는데, 제한된 하드웨어 리소스를 가진 모바일과 IoT 기기에서는 이러한 모델을 사용하기 어렵다.

이러한 문제를 해결하기 위해 모델의 가중치 중에서 중요도가 낮은 가중치 연결을 제거하고 남은 가중치를 재학습하는 pruning 기법[1]이 제안되었다. pruning은 주로 fine-grained granularity를 가지는 element-level pruning과 coarse-grained granularity를 가지는 filter-level pruning으로 나눌 수 있다. Element-level pruning은 정확도의 손실 없이 대부분의 가중치와 연산량을 줄일 수 있지만 불규칙한 가중치 값 접근으로 인해 pruning된 네트워크가 하드웨어에서 가속되기 어렵다. 반면에 filter-level pruning은 filter 단위로 가중치를 제거하기 때문에 가속이 쉽지만 element-level pruning에 비해 정확도 손실이 크다. 하지만 element-level pruning과 filter-level pruning의 중간형태를 가지는 group-level pruning은 SIMD(single instruction multiple data)연산이 가능한 형태로 가중치를 그룹화하기 때문에 가속이 가능하면서 filter-level pruning에 비해 정확도 손실을 최소화하며 모델을 압축할 수 있다. 하지만 기존의 group-level pruning[2]은 각 레이어의 중요도를 고려하지 않고 single global threshold를 기준으로 전체 레이어의 가중치를 pruning하기 때문에 최적의 성능을 낼 수 없다.

본 논문에서는 필터마다 학습가능한 threshold를 통해 레이어별 중요도를 고려할 수 있는 DST[3]에 group-level pruning을 적용한 Group-level pruning with DST(GP-DST)를 제안한다. 또한 네트워크의 정확도 손실을 줄이기 위해 pruning scheduler를 추가로 적용하여 gradual하게 pruning한 Gradual Group-level Pruning with DST(GGP-DST)를 제안한다. 그 결과 기존 single global threshold를 사용한 group-level pruning 방법에 비해 같은 sparsity 대비 더 높은 정확도를 가질 수 있었다.

2. 관련 연구

Scalpel[2]에서는 하드웨어 구조를 활용하기 위해 SIMD 유닛에 알맞은 크기로 가중치 group을 설정하여 pruning하는 방법을 제안했다. 각 group의 크기는 SIMD 폭과 같고 SIMD연산을 활용하여 성능을 향상시킬 수 있었다. 하지만 이러한 기존 group-level pruning 방법은 single global threshold를 기준으로 pruning하는데 이는 레이어마다 중요도가 다르다는 점을 고려하지 못했기에 최적의 성능을 내지 못한다. 또한 sparsity가 커질수록 레이어 마다 중요한

이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.IITP-2017-0-00914, 지능형 IoT 장치용 소프트웨어 프레임워크)

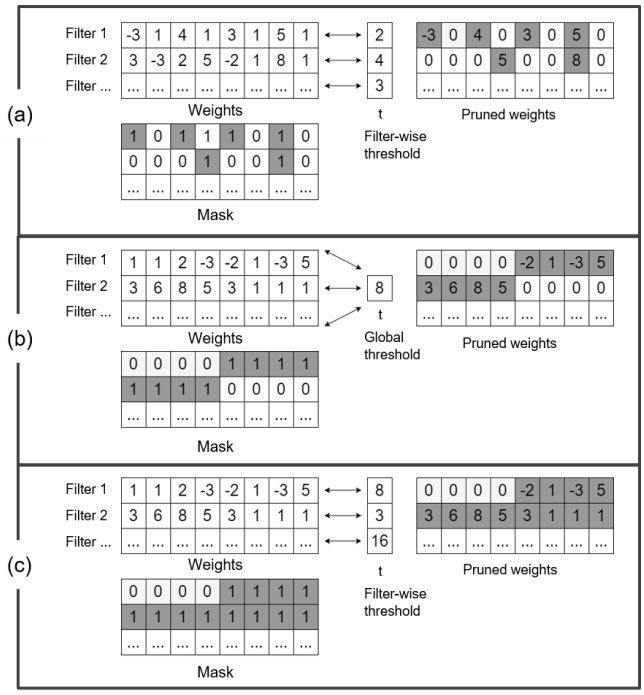

그림 1 (a) filter-wise threshold로 모든 레이어의 가중치를 pruning 하는 DST의 element-level pruning. (b)는 global threshold로 모든 레이어의 가중치를 pruning 하는 기존의 group-level pruning. (c) filter-wise threshold로 각 필터의 가중치를 pruning하는 본 논문의 GP-DST.

가중치를 남기지 못하기에 정확도의 손실은 더 커진다.

이러한 문제를 해결하기 위해서는 중요한 레이어의 가중치들은 많이 남기고 중요하지 않은 레이어의 가중치는 더 많이 제거하는 방법이 필요하다. 최근에는 각 특정 계층별 중요도를 고려하여 pruning하기 위해 레이어나 필터마다 학습가능한 threshold를 두고 가중치와 함께 학습하며 threshold를 기준으로 가중치를 pruning하는 기법[3,4] 이 연구되고 있다. 본 논문에서는 이러한 방법 중 하나인 DST를 group-level pruning에 적용하였고 기존방법인 single global threshold에 비해 더 나은 layer-wise sparsity를 찾았다.

기존 연구에서는 한번에 모든 가중치를 pruning하는 one-shot pruning에 비해 sparsity를 단계적으로 높여가며 가중치를 제거하는 gradual pruning[5,6]이 안정적으로 학습되며 성능이 더 개선됨을 보였다. 기존 DST에서는 α 라는 penalty를 통해 pruning ratio를 조절하는데, 학습 처음부터 끝까지 일정한 α 값을 통해 학습하기 때문에 pruning이 급격하게 이루어진다. 따라서 학습이 안정적으로 진행되지 않는데 본 논문에서는 α 값을 작은 값에서부터 목표 값까지 천천히 상승시킴으로써 gradual pruning하는 효과를 보였다.

3. Gradual Group-level Pruning with DST

DST는 filter 마다 threshold를 두고 pruning과 recover를 반복하며 sparse한 모델로 만드는 방법이다. 이러한 DST를 group-level pruning에 적용하여 filter마다 서로 다른 중요도로 group의 pruning 여부를 결정하도록 했다.

Convolution layer의 가중치 행렬은 $W \in \mathbb{R}^{c_o \times c_i \times w \times h}$로 나타낼 수 있다. 여기서 c_i는 입력 채널 수, c_o는 출력 채널 수, w와 h 는 각각 convolution kernel의 너비와 높이를 뜻한다. 가중치를 pruning 하는 기준인 threshold $t \in \mathbb{R}^{c_o}$ 는 가중치 행렬 W_{ij}의 L1 norm과의 차이를 통해 pruning mask M_{ik}를 수식 (1)과 같이 생성하고 가중치와 mask의 Hadamard product를 통해 가중치를 pruning한다. 여기서 i 는 i번째 레이어, j 는 j번째 커널, k 는 k번째 group, G 는 group 사이즈를 뜻하는데 본 논문에서는 4의 값을 사용했다.

$$M_{ik} = step\left(\sum_{j=Gk-G+1}^{Gk} |W_{ij}| - t_i\right), 0 < k < \frac{c_i}{G} \qquad (1)$$

$step(\cdot)$ 은 step function으로, $|W_{ij}|$ 의 값이 t 값보다 크면 pruning 하지 않고 t 값보다 작을 경우 pruning하도록 하는 mask를 생성한다. 그림 1 (a)는 DST에서 mask가 생성되어 가중치가 pruning되는 과정으로 mask는 element-level로 생성되며 가중치 또한 element-level로 선택된다. 하지만 이는 불규칙한 가중치 값 접근으로 가속이 어렵기에, 본 논문에서는 그림 1 (b)처럼 SIMD연산이 가능한 형태로 가중치를 그룹화하여 pruning한 기존의 group-level pruning을 그림 1 (c)와 같이 변형하였다. 이때 mask는 학습가능한 threshold를 통해 생성되며, 가중치와 같은 형태의 그룹단위로 생성된다.

네트워크의 pruning ratio는 threshold regularize term을 조절하는 penalty α 값을 통해서 조절할 수 있고 threshold regularize term은 다음과 같이 쓸 수 있다.

$$L_t = \sum_{i=1}^{c_o} \exp(-t_i) \qquad (2)$$

하지만 기존 DST는 위의 regularize term을 일정한 α 값으로 조절하기 때문에 학습 초반부터 pruning이 급격하게 이루어져서 기존의 연구[5,6] 내용처럼 최적의 성능을 달성하지 못한다.

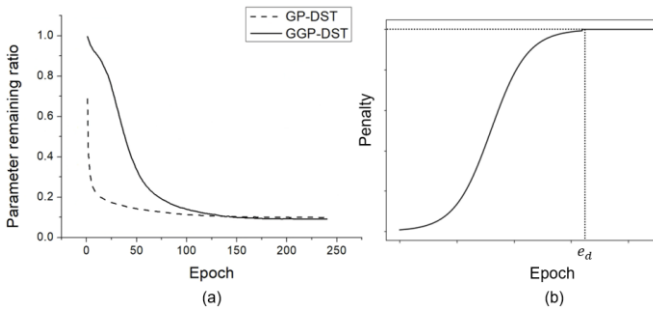

그림 2 (a) epoch에 따른 pruning 속도 차이를 나타냄. (b) epoch에 따른 α값의 변화를 나타내고 e_d는 learning rate decay 가 이루어지는 epoch를 뜻함.

따라서 본 논문에서는 gradual pruning을 통한 성능 향상을 위해 기존 α 값을 작은 값에서 목표 값까지 상승시키는 α

scheduler를 적용한 GGP-DST를 제안한다. α 값을 작은 값에서부터 상승시킬 경우 threshold도 점차 큰 값을 가지며 점점 더 많은 가중치를 제거하게 된다. 그림 2 (a)는 α scheduler 적용 여부에 따른 pruning 속도차이를 나타낸 그림으로, 일정한 α 값을 사용한 GP-DST에 비해 gradual α 값을 사용하는 GGP-DST가 gradual하게 pruning됨을 보인다.

$$\lambda(epoch) = \begin{cases} \alpha \cdot Sigmoid(-5 + \frac{10 \cdot epoch}{e_d}), & epoch < e_d \\ \alpha, & epoch \geq e_d \end{cases} \quad (3)$$

그림 2 (b)는 수식 (3)을 도식화한 것으로, sigmoid 함수는 saturate되는 특성이 있기에 학습 초반에는 낮은 sparsity에서 안정적으로 가중치를 학습하다 점차 목표 α 값과 자연스럽게 연결되며 pruning할 수 있다. e_d 는 learning rate decay가 이루어지는 epoch를 뜻한다. 그래서 최종적으로 사용하는 전체 loss L은 수식 (4)와 같이 분류손실(classification loss) L_{cls}와 pruning ratio를 조절하는 L_t의 합으로 이루어진다.

$$L = L_{cls} + \lambda L_t \quad (4)$$

4. 실험

4.1 실험 환경

본 논문의 실험은 AMD Ryzen Threadripper 1950X, DRAM 64GB, Nvidia Titan RTX x 1 로 구성된 pc 환경에서 진행했다.

본 논문의 기법을 평가하기 위해 ResNet-20 모델과 CIFAR10 데이터 셋을 사용하여 실험을 진행했다. 또한 pruning할 group의 차원은 1차원이며 group 크기는 4로 설정하였다.

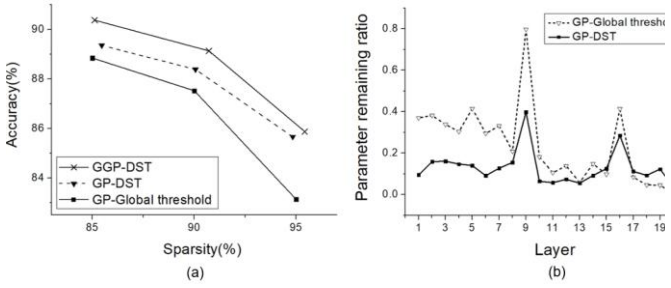

그림 3 (a) Sparsity 85%, 90%, 95%에서 각 모델의 정확도. (b) Layer-wise sparsity.

4.2 Layer-wise sparsity

그림 3 (b)는 CIFAR10에서 학습시킨 ResNet-20의 레이어마다 pruning되지 않고 남아있는 가중치 비율을 나타낸 것으로, 기존의 global threshold 방법은 레이어의 중요도에 따라 효과적으로 pruning하지 못하고 단순히 대부분의 가중치가 집중된 후반부 레이어로 갈수록 pruning이 많이 이루어지는 양상을 보인다. 그림 3 (a)는 CIFAR10에서 학습시킨 ResNet-20 모델들을 sparsity 85%, 90%, 95%에서 서로 비교한 것으로, 이때 sparsity는 전체 가중치 중에서 pruning된 가중치의 비율을 뜻한다. 그림 3 (a)를 보면 기존 방법이 sparsity 85%, 90%, 95%일 때 정확도는 88.84%, 87.52%, 83.13%이고 GP-DST는 sparsity 85.46%, 90.06%, 94.83%일 때 정확도는 89.35%, 88.38%, 85.66%로 기존 방법에 비해 정확도가 각각 0.51%, 0.86%, 2.53% 상승했다. 특히 sparsity가 커질수록 single global threshold와 GP-DST의 정확도 차이가 커지는 것으로 보아 전체 레이어에 걸쳐 적절한 가중치만 남기고 pruning한 GP-DST가 기존 방법보다 더 나은 layer-wise sparsity를 찾았다고 할 수 있다.

4.3 Gradual Group-level Pruning

그림 3 (b)에서 GP-DST와 GGP-DST를 비교했을 때 GGP-DST는 sparsity가 85.11%, 90.72%, 95.42% 일 때 정확도는 90.37%, 89.13%, 85.87%로 비슷한 sparsity에서 정확도가 각각 1.02%, 0.75%, 0.21% 상승했고 global threshold 방법과 비교해서 1.53%, 1.61%, 2.74% 상승했다.

5. 결론 및 향후 연구

본 논문에서는 GP-DST를 통해 기존 group-level pruning에 비해 더 나은 layer-wise sparsity를 찾음으로써 비슷한 sparsity에서 더 높은 정확도를 달성했다. 또한 학습 초반부터 급격하게 pruning하는 GP-DST에 gradual pruning을 적용해 정확도를 더욱 상승시켰다.

본 논문에서 SIMD연산에 활용할 가중치 그룹이 항상 규칙적으로 4의 배수(group size) 위치에서 선택되기에 높은 중요도의 가중치이지만 해당 그룹이 pruning되어 선택되지 않을 수 있다. 따라서 최적의 그룹을 선택하기 위해 불규칙적인 위치에서 가중치 그룹을 선택할 수 있다면 기존의 가중치 그룹이 선택할 수 없었던 가중치를 포함시킨 가중치 그룹을 만들어 정확도를 더 향상시킬 수 있을 것으로 기대된다.

참고문헌

[1] Han, Song, Huizi Mao, and William J. Dally. "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding." arXiv preprint arXiv:1510.00149 (2015).

[2] Yu, Jiecao, et al. "Scalpel: Customizing dnn pruning to the underlying hardware parallelism." ACM SIGARCH Computer Architecture News. Vol. 45. No. 2. ACM, 2017.

[3] Liu, J., Xu, Z., Shi, R., Cheung, R. C. C., and So, H. K. Dynamic sparse training: Find efficient sparse network from scratch with trainable masked layers. In International Conference on Learning Representations, 2020.

[4] Kusupati, A., Ramanujan, V., Somani, R., Wortsman, M., Jain, P., Kakade, S., and Farhadi, A. Soft threshold weight reparameterization for learnable sparsity. arXiv preprint arXiv:2002.03231, 2020.

[5] Li, H., Kadav, A., Durdanovic, I., Samet, H., and Graf, H. P. Pruning filters for efficient convnets. arXiv preprint arXiv:1608.08710, 2016.

[6] Zhu, M. and Gupta, S. To prune, or not to prune: exploring the efficacy of pruning for model compression. arXiv preprint arXiv:1710.01878, 2017.