영상 압축기술을 통한 가중치 압축방법

김승환[1], 박은수[1], 굴람 무즈타바[2], 류은석[1]
성균관대학교[1], 가천대학교[2]
seunghwankim@skku.edu[1], espark804@skku.edu[1], mujtaba@gc.gachon.ac.kr[2], esryu@skku.edu[1]

Weight Compression Method with Video Codec

Kim, SeungHwan[1] Park, Eun-Soo[1] Ghulam Mujtaba[2] Ryu, Eun-Seok[1]
Sungkyunkwan University[1], Gachon University[2]

요약

최근 모바일 기기에서 딥러닝 모델을 사용하기 위한 경량화 연구가 진행되고 있다. 그중 모델의 가중치 표현 bit를 줄이는 양자화와 사용하기 위한 다양한 압축 알고리즘이 개발되었다. 하지만 대부분의 양자화 및 압축 알고리즘들은 한 번 이상의 Fine-tuning을 거쳐야 하는데 이 과정은 모바일 환경에서 수행하기에는 연산복잡도가 너무 높다. 따라서 본 논문은 양자화된 가중치를 High Efficiency Video Coding(HEVC)을 통해 압축하는 방법을 제안하고 정확도와 압축률을 실험한다. 실험결과는 양자화만 실시한 경우 대비 크기는 25%의 감소했지만, 정확도는 0.7% 감소했다. 따라서 이런 결과는 모바일 기기에 가중치를 전송하는 과정에 적용될 수 있다.

1. 서론

최근 모바일 기기처럼 성능이 제한된 환경에서 딥러닝 모델을 사용하기 위한 연구가 다양하게 진행되고 있다. 딥러닝 모델은 GPU의 통한 병렬처리를 기반으로 다양한 분야에서 뛰어난 성능을 보였지만 대부분 모바일 기기에서 사용하기에는 어려움이 있다. 따라서 다양한 분야의 딥러닝 모델들의 경량화에 대한 요구가 늘어나고 있다.

ImageNet Large Scale Visual Recognition Challenge (ILSVRC)에서 뛰어난 성능을 보였던 다양한 모델들의 연산량과 크기를 줄이는 방법이 다양하게 연구되었다[1]. ResNet은 가중치의 수를 줄이면서 ILSVRC에서 VGGNet보다 뛰어난 정확도를 달성했다[2]. 이후 MobileNet과 같이 모바일 기기에 최적화된 모델 구조들이 다수 발표되었다[3]. 특히 MobileNet은 VGG16의 약 3%의 크기로 1%의 정확도 하락을 기록했다[4].

기존의 모델 구조와 성능을 유지하며 가중치의 크기를 줄이는 연구 역시 활발하게 이루어지고 있다. 본 논문에서는 학습된 가중치의 크기를 줄이는 방법으로 Weight Quantization을 수행한 다음 HEVC를 통해 압축하는 방법을 제안하고 양자화 알고리즘별 성능을 비교하는 실험을 진행한다[5].

본 논문은 총 5개의 장으로 구성된다. 2장에서는 가중치 양자화와 압축에 관한 기존 연구를 소개한다. 3장에서는 Fully-Connected Layer의 가중치 분포를 분석하고 가중치의 통계 수치를 비교한다. 4장에서는 양자화 알고리즘에 따라 HEVC를 통해 압축했을 때 성능과 Quantization Parameter(QP)별 차이에 대한 실험을 진행하고 5장에서 결론과 함께 추후 연구에 대해 정리한다.

2. 관련 연구

본 장에서는 가중치의 압축과 양자화에 관한 연구들을 소개한다. 2.1에서는 표현 bit를 줄이는 가중치 압축의 연구와 양자화 방법에 관해 기술한다. 2.2에서는 다양한 기술을 통해 가중치를 압축하는 방법을 소개한다.

2.1 Weight Quantization

일반적인 딥러닝 모델의 각 가중치는 32bit float형으로 표현한다. 이는 GPU의 가속을 통해 빠르게 수행할 수 있지만, CPU만을 사용하거나 모바일 환경에서는 사용에 한계가 존재한다. 따라서 표현 bit를 줄여 메모리 사용량과 연산시간을 줄이는 연구가 다양하게 발표되었다. [6]은 기존 모델들의 가중치를 8bit로 줄이는 선형 양자화(Linear Quantization)를 다양한 모델에 적용했을 때 지연시간을 낮추면서 정확도의 손실을 최소화했다. ResNet에 적용했을 때 크기는 25%로 줄었지만, 정확도는 2% 정도 감소했다. [7]은 양자화 과정에서 각 Layer의 연산을 고려하여 표현 범위를 더욱 줄이는 Net-aware Quantization을 제안했다. 예를 들어 Rectified Linear Unit(ReLU) 함수만 사용한다면 음수 범위를 제거하여 더 효율적으로 양자화할 수 있다. 이 방법을 통해 ImageNet 데이터 셋로 학습된 ResNet50을 8bit로 양자화하면서 0.1%의 정확도 손실을 기록했다.

모델을 구성하는 가중치, 활성화 함수를 양자화하는 방법은 대칭과 비대칭 두 가지 방법이 있다. 대칭 양자화는 가중치의 가장 큰 절댓값을 기준으로 대칭으로 범위를 지정한다. 따라서 대칭 양자화는 Zero-point가 0으로 고정되어 있다.

원본 모델의 가중치를 x_f, 양자화된 가중치를 x_q, Scale Factor를 q_x라고 할 때 대칭 양자화는 다음과 같이 계산한다.

$$q_x = \frac{2^{n-1}-1}{\max(|x_f|)} \quad (1)$$

$$x_q = round(q_x x_f) \quad (2)$$

비대칭 양자화는 x_f의 최대, 최소를 정수 범위의 최대 최소에 매핑한다. 이를 위해서 Scale Factor q_x에 추가로 Zero-point zp_x을 사용한다. 수식으로 나타내면 다음과 같다.

$$q_x = \frac{2^n-1}{\max(x_f)-\min(x_f)} \quad (3)$$

$$zp_x = \min(x_f q_x) \quad (4)$$

$$x_q = round(q_x x_f - zp_x) \quad (5)$$

대칭 양자화와 비대칭 양자화는 데이터, 모델에 따라 다르다는 것이 [8]에서 실험을 통해 밝혀졌다.

2.2 Weight Compression

양자화뿐 아니라 다양한 경량화 방법들을 결합하여 모델을 압축하는 방법에 관해서도 많은 연구가 발표되었다. 2015년에 발표된 Deep Compression은 Pruning, Quantization과 허프만 코딩(Huffman Coding)을 통해 정확도의 손실 없이 크기를 크게 줄이는 방법이다[9]. 그 결과 VGGNet의 성능을 유지하며 크기를 552MB에서 11.3MB까지 감소했다.

[10]는 CNN에서 Convolution 필터들을 DCT와 허프만 코딩을 통해 압축하는 CNNpack 모델을 제안했다. CNNPack은 입력 데이터 또한 DCT를 통해 주파수 영역으로 변환하여 입력한다. 이 방법을 통해 VGGNet의 1%의 정확도 감소와 46배의 크기 감소를 달성했다.

본 절에서 소개된 방법들은 압축률이 높고 정확도 감소가 적지만 1회 이상의 Fine-Tuning 혹은 Re-train 과정을 포함한다. 이 과정은 연산량이 많으며 학습 데이터가 필요하므로 모바일 기기에 적합하지 않다. 본 논문에서는 성능이 제한된 모바일 기기에서 수행할 수 있는 양자화와 HEVC Encoder를 통한 압축을 제안한다.

3. HEVC를 통한 가중치 압축

본 장에서는 본 연구에서 실험하는 ResNet50의 FC Layer의 양자화에 관해 기술한다. ResNet50의 FC Layer는 2048x1000 크기의 행렬로 구성되어 있다[2]. FC Layer의 통계수치는 다음과 같다. Qsym는 대칭 양자화를, Qasym를 각각 나타낸다.

	평균	분산	최소값	최대값
Baseline	0	0.001	-0.24	0.73
Qsym	0	33.62	-42	127
Qasym	-65.00	75.92	-128	127

표 1 FC Layer 가중치의 통계

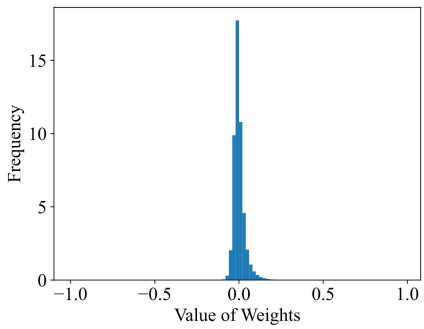

그림 1 사전 학습된 FC Layer의 가중치 분포

그림 1에서 사전 학습된 가중치는 분산이 0.001로 0 주변에 집중되어 있다. 또한, 최댓값은 0.73으로 -0.24인 최솟값보다 절댓값이 크기 때문에 비대칭 양자화의 Zero-point가 음수가 되고 분산은 증가했다.

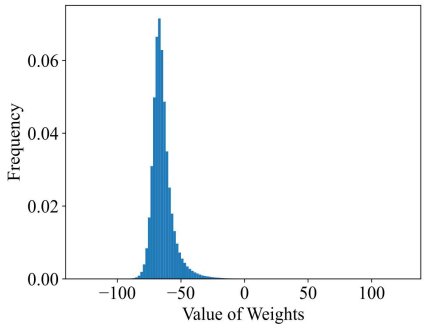

그림 2 비대칭 양자화한 FC Layer의 가중치

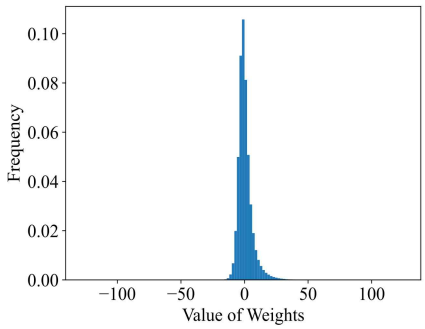

그림 3 대칭 양자화한 FC Layer의 가중치

대칭 양자화는 그림 3과 같이 Zero-point가 0으로 원본 모델과 비슷하게 최솟값의 절댓값이 더 작았다. 비대칭 양자화의 경우 -128에서 127의 범위를 모두 사용하며 분산도 높게 나타났다. 비대칭과 대칭 양자화 가중치는 8bit로 표현되어 양자화로 인한 압축률은 같다.

4. 실험 및 결과

본 장은 ResNet50 전체를 양자화 후 FC Layer를 HEVC의 Intra-picture모드로 압축했을 때 결과를 나타낸다. 본 실험에 사용한 데이터 셋은 ImageNet 데이터 셋의 검증 데이터로 1000가지 클래스 이미지 50000개로 구성되어 있다. 4.1에서는 양자화 방법에 따라 정확도의 변화를 비교하여 나타내며 4.2에서는 압축률과 정확도에 관해 기술한다.

	Size [kB]	Compression	Top -1 Acc	Top -5 Acc	Loss
Baseline	8,196,256	-	76.13	92.86	0.962
Qsym	2052498	4.00X	74.57	92.01	1.023
Qasym	2052498	4.00X	74.33	92.08	1.028
Qsym + HEVC	487353	16.82X	73.91	91.88	1.032
Qasym + HEVC	655094	12.51X	74.17	91.98	1.031

표 2 비대칭 양자화(Qasym)와 대칭 양자화(Qsym) 및 HEVC(QP:23)으로 이를 압축한 결과.

4.1 대칭 양자화와 비대칭 양자화

이 절에서는 양자화 방법과 가중치의 평균, 분산, 범위가 HEVC를 통한 압축률과 정확도에 미치는 영향에 대해 실험한다. Baseline은 pytorch에서 제공하는 사전 학습된 가중치를 통해 실험한 내용이다. Compression 는 압축률을 나타낸 것으로 (원본 크기) / (압축된 크기)로 계산된다.

표 1에서 볼 수 있듯이 비대칭 양자화의 분산이 더 크게 나타났다. 이는 정확도를 조금 더 보존하기도 하지만 중복성이 줄어들어 압축률이 크게 낮아지는 것이 표 2에서 나타났다. 양자화 후 HEVC로 압축했을 때 비대칭 양자화를 실시한 것이 정확도는 0.25% 정도 높았지만 압축된 크기는 25% 정도 컸다. 따라서 HEVC를 통해 압축을 진행할 경우 대칭 양자화를 통해 압축하는 것이 정확도 손실은 더 크지만, 압축효율을 높일 수 있다.

4.2 QP에 따른 정확도 변화

HEVC는 양자화 계수 QP를 통해 압축의 손실률을 조절할 수 있다. 본 절에서는 QP값에 따라 정확도와 압축률의 변화를 실험하여 그래프로 나타낸다. 다음은 대칭 양자화와 비대칭 양자화 후 압축했을 때 각각 QP에 따른 정확도의 변화를 나타낸 것이다.

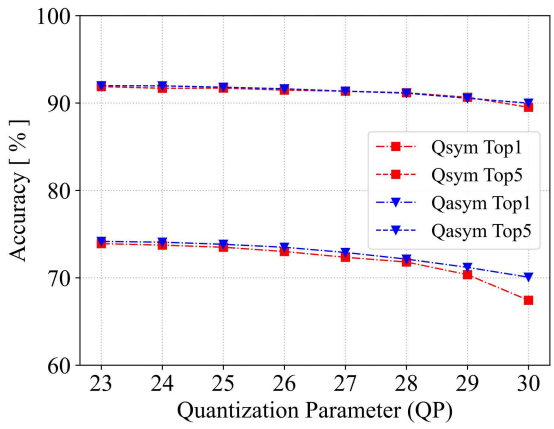

그림 4 QP에 다른 정확도

그림 4에서 Top-5 정확도의 경우 두 양자화 알고리즘 간에 차이가 1%이하로 나타났다. Top-1 정확도는 QP가 30일 때 2.6%의 차이가 있었다. 하지만 4.1절의 실험에서 비대칭 양자화의 경우 정확도는 높지만, 압축효율은 낮아졌다. 다음은 양자화 알고리즘 간의 압축효율을 나타낸 것이다.

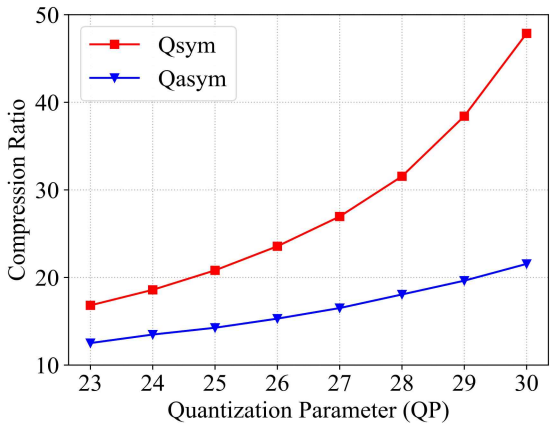

그림 5 QP에 따른 압축률

그림 4에서 나타난 정확도의 차이에 비해 그림 5의 압축률의 차이는 높은 폭으로 증가했다. 특히 QP가 30일 때 비대칭 양자화의 압축률과 QP가 25일 때 대칭 양자화의 압축률은 같지만, Top -1 정확도의 경우 3.5% 이상 차이가 났다.

5. 결론

본 논문은 8bit 정수로 양자화한 가중치 행렬을 HEVC를 통해 압축하는 방법을 제안하고 그 과정에서 양자화 알고리즘이 압축률과 정확도에 미치는 영향을 실험했다. 양자화만 적용하는 경우 비대칭 양자화가 같은 크기에서 더 높은 정확도를 기록했다. 하지만 본 논문이 제시하는 압축과정에서 비대칭 양자화는 효율이 크게 떨어진다는 것을 실험을 통해 밝혔다. 후속 연구로는 CNN 모델이 아닌 다른 모델에 양자화와 HEVC를 통한 압축에 대한 실험을 진행할 예정이다.

Acknowledgement

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT 연구센터육성지원사업의 연구결과로 수행되었음 (IITP-2020-2017-0-01630)

참고문헌

[1] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Berg, A. C. 1(2015). Imagenet large scale visual recognition challenge. International journal of computer vision, 115(3), 211-252.

[2] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[3] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[4] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.-

[5] Sullivan, G. J., Ohm, J. R., Han, W. J., & Wiegand, T. (2012). Overview of the high efficiency video coding (HEVC) standard. IEEE Transactions on circuits and systems for video technology, 22(12), 1649-1668.

[6] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., ... & Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2704-2713).

[7] Park, J., Naumov, M., Basu, P., Deng, S., Kalaiah, A., Khudia, D., ... & Pino, J. (2018). Deep learning inference in facebook data centers: Characterization, performance optimizations and hardware implications. arXiv preprint arXiv:1811.09886.

[8] Krishnamoorthi, R. (2018). Quantizing deep convolutional networks for efficient inference: A whitepaper. arXiv preprint arXiv:1806.08342.

[9] Han, S., Mao, H., & Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149.

[10] Wang, Y., Xu, C., You, S., Tao, D., & Xu, C. (2016). Cnnpack: Packing convolutional neural networks in the frequency domain. In Advances in neural information processing systems (pp. 253-261).