Journal of The Institute of Electronics and Information Engineers Vol.57, NO.12, December 2020 https://doi.org/10.5573/ieie.2020.57.12.60
ISSN 2287-5026(Print) / ISSN 2288-159X(Online)

논문 2020-57-12-7

정수 연산만을 사용하는 하드웨어 친화적인 양자화된 CNN 구현

(Implementation of a Hardware-friendly Quantized CNN using Integer Arithmetic Only)

김 재 명*, 김 용 우**

(Jaemyung Kim and Yongwoo Kim©)

요 약

최근 컴퓨터 비전 분야에서 CNN을 이용한 연구가 우수한 성능을 보여줌에 따라 보편적인 연구방법으로 자리 잡았다. 하지만 높은 성능을 얻기 위해서는 많은 수의 파라미터와 높은 연산 복잡도를 갖는 CNN 모델이 필요하다. 이러한 모델은 하드웨어 자원 사용량에 제한이 있는 환경에서는 적합하지 않다. 따라서 CNN 모델의 구조를 유지한 채 정밀도를 낮추어 연산 복잡도 및 메모리 사용량을 최적화할 수 있는 양자화 연구가 활발히 진행되고 있다. 하지만 대부분의 양자화 기법은 성능을 유지하기 위해 부동 소수점 스케일 인자를 사용하는 등 하드웨어 친화적이지 않은 방법들이 사용되었다. 따라서 본 논문에서는 하드웨어 친화적인 양자화 기법을 적용하여 양자화 인식 훈련을 수행하였고 양자화 인식 훈련으로 학습된 부동 소수점 파라미터를 정수 파라미터로 변환하였다. 그리고 오직 정수 연산만을 사용하는 기본 연산 블록들을 CNN으로 구성 및 계층 별 정밀도를 검증하는 기법을 제안한다. 8-bit로 양자화된 파라미터를 이용하여 정수 연산만을 사용하는 CNN의 성능을 CIFAR-10 데이터 세트로 평가한 결과, 부동 소수점 대비 파라미터 개수는 1/4로 줄었지만 정확도 하락은 0.71% 이하인 것을 확인하였다. 또한 기존의 양자화 인식 추론 기법과 비교하였을 때 추가적인 부동 소수점 연산기 대신 정수 연산기만을 사용하여 연산 복잡도를 낮출 수 있었으며 정확도 하락은 0.13% 이하인 것을 확인하였다.

Abstract

Recently, in the field of computer vision, research using CNN has established itself as a universal research method as it shows excellent performance. However, to obtain high performance, a CNN model with a large number of parameters and high computational complexity is required. This model is not suitable in an environment where hardware resource usage is limited. Therefore, while maintaining the structure of the CNN, quantization research that can optimize computational complexity and memory usage by lowering the precision is being actively conducted. However, most quantization techniques are not hardware-friendly methods were used, such as using a floating-point scale factor to maintain performance. In this paper, quantization aware training(QAT) was performed by applying a hardware-friendly quantization technique and floating-point parameters learned by QAT were converted into integer parameters. In addition, we proposed a method for constructing a CNN that combines basic operation blocks using integer arithmetic only operation and a method for verifying the precision of each layer. As a result of evaluating the performance of the proposed CNN using 8-bit quantized parameters with the CIFAR-10 dataset, it was confirmed that the number of parameters compared to floating-point was reduced to 1/4, but the accuracy drop was less than 0.71%. Besides, it was confirmed that the computational complexity could be reduced by using only an integer operator instead of an additional floating-point operator, and the accuracy drop was less than 0.13% when compared with the conventional quantization aware inference method.

Keywords : Deep learning, Image classification, Quantization, Integer arithmetic, Hardware-friendly

* 학생회원, 인하대학교 전기컴퓨터공학과(Department of Electrical and Computer Engineering, Inha University)
** 정회원, 상명대학교 시스템반도체공학과(Department of System Semiconductor Engineering, Sangmyung University)
© Corresponding Author(E-mail : yongwoo.kim@smu.ac.kr)

Received ; October 15, 2020 Revised ; October 27, 2020 Accepted ; October 27, 2020

Ⅰ. 서 론

최근 하드웨어 기술의 발전으로 인한 연산능력의 비약적인 향상, IoT 기술의 발달로 인한 데이터의 폭발적인 증가로 인해 기계학습 알고리즘을 이용한 딥러닝 기술이 자율주행자동차, 시각 인식 등 어렵고 복잡한 문제에 대해 뛰어난 성능을 보여줌에 따라 다양한 응용분야에 적용되고 있다. 특히 영상 분류, 객체 인식 등 컴퓨터 비전 분야에서는 CNN(Convolutional neural network)을 이용하는 방식이 우수한 성능을 보여줌에 따라 보편적인 연구방법이 되었다. 높은 성능을 얻기 위해서는 네트워크를 구성할 때 많은 컨벌루션 계층(Convolution layer)이 필요하다. 하지만 이러한 네트워크는 많은 연산량과 메모리가 필요하기 때문에 하드웨어 자원사용량에 제한이 있는 임베디드 환경에서의 응용은 적합하지 않다. 이를 극복하기 위해 연산량, 메모리, 전력소모, 지연시간 측면에서 효율성을 높이기 위한 딥러닝 알고리즘 경량화 연구가 활발히 진행되고 있다.

딥러닝 알고리즘 경량화 연구는 가지치기(Pruning), 지식증류(Knowledge distillation) 양자화(Quantization) 등 크게 3가지 방식으로 나뉜다. 가지치기는 CNN 모델에서 불필요한 파라미터를 제거하여 파라미터 개수를 줄이는 기법이다[1]. 성능에 영향을 주는 파라미터는 한정되어 있다는 이론에서 출발하여 특정 임계치보다 작은 값을 모두 0으로 만들어 절반의 파라미터로 동일한 성능을 달성할 수 있다. 또한, 가지치기 후 재학습 과정을 통해 기존 모델 대비 10%의 파라미터만을 이용하여 성능저하가 거의 없음을 보여주었다. 가지치기 기법은 파라미터를 제거하는 세분성(Granularity)에 따라 2가지로 나뉜다[2]. Fine-grained granularity 방식은 작은 규모의 파라미터를 제거함으로써, 정확도를 유지할 수 있지만 매우 불규칙한 네트워크를 생성하여 추론속도 향상을 기대할 수 없다. Coarse-grained granularity 방식은 파라미터를 그룹으로 묶어 해당 그룹에서 작은 파라미터의 영역을 제거하는 방식인데 이는 추론 속도를 향상시킬 수 있지만 성능 손실이 크다는 단점이 있다.

지식 증류 기법은 모델 앙상블을 통해 학습된 큰 모델(Teacher model)로부터 작은 모델(Student model)을 학습시키는 기법이다[3]. 즉, 큰 모델이 학습한 일반화 능력을 작은 모델에 전달하여 작은 모델을 독립적으로 학습시키는 것보다 더 높은 성능을 얻을 수 있음을 보여주었다. 하지만, 하드웨어로 구현 시 부동 소수점으로 구현이 되기 때문에 연산 속도를 높이기 위해서는 별도의 연산장치를 두어야 한다.

양자화는 CNN의 구조는 유지하면서 32-bit 부동 소수점으로 훈련된 파라미터를 낮은 bit의 고정 소수점 또는 정수로 변환하여 연산을 수행하는 기법이다. 이를 통해 메모리 사용량을 최적화 할 수 있고 보다 단순한 하드웨어를 이용하여 연산량 및 전력 소모를 줄일 수 있으며 연산속도를 향상시킬 수 있다. 그러나 파리미터의 정밀도가 낮아질수록 CNN 모델의 표현력이 감소하여 성능이 감소하는 단점이 있다. 따라서 최신 양자화 연구들[4~14]에서는 성능 감소를 최소화 하려는 시도가 진행되고 있다.

본 논문에서는 하드웨어 친화적인 양자화 기법으로 양자화 인식 훈련(Quantization aware training)을 수행하였고, 양자화 인식 훈련으로 학습된 부동 소수점 파라미터를 정수 파라미터로 변환하였다. 그리고 오직 정수 연산만을 사용하는 기본 연산 블록을 조합하여 CNN을 구성하고 계층 별 정밀도(Layer precision)를 검증하는 기법을 제안한다.

본 논문의 기여는 다음과 같다. 1) Jacob 등[11], Jain 등[14]기법에 영감을 받아 스케일 인자를 2의 거듭제곱(Power of 2)으로 학습되도록 만들어 양자화 과정에서 Shift 연산만 사용하여 스케일 인자의 곱셈과 나눗셈 연산을 대체하였고, 배치 정규화 계층 융합(Batch normalization layer fusion)을 적용하여 배치 정규화 계층에서의 곱셈 및 나눗셈 연산을 제거하여 연산량을 줄일 수 있었다. 2) 오직 정수 연산만을 사용하는 기본 연산 블록의 계층 별 정밀도(Layer precision)를 검증하는 기법을 제안하고 이를 조합하여 정수 연산만을 사용하는 CNN을 구성하였다. 3) 부동 소수점 훈련, 하드웨어 친화적인 양자화 인식 훈련, 정수 연산만을 사용하는 CNN 추론 과정을 하나의 프레임워크로 구현하였다.

본 논문의 구성은 다음과 같다. 2장에서는 양자화 관련 연구에 관해 설명한다. 그리고 3장에서는 하드웨어 친화적인 양자화 기법 및 오직 정수 연산만을 사용하는 기본 연산 블록에 대한 계층 별 정밀도(Layer precision)를 검증하는 방법에 대해 설명한다. 그리고 4장에서는 정수 연산만을 사용하는 기본 연산 블록을 조합하여 구성된 CNN의 실험 결과 및 성능을 분석하고, 마지막으로 5장에서는 본 논문에 대한 결론 및 향후 연구 방향에 대해 논의한다.

Ⅱ. 양자화 관련 연구

이번 장에서는 양자화 기법에 대한 기존 연구들을 소개한다. 양자화 관련 연구는 크게 2가지 기법으로 나누어진다. 이미 학습 된 파라미터를 추론 시에 양자화 하는 훈련 후 양자화(Post training quantization) 기법과 양자화 영향을 고려하여 파라미터를 재학습시키는 양자화 인식 훈련(Quantization aware training)이 있다.

1. 훈련 후 양자화(Post training quantization)

훈련 후 양자화 기법은 부동 소수점으로 학습을 진행하고 훈련된 파라미터에 대해서 양자화를 적용한다. 즉, 추론 시 순방향 경로의 가중치와 활성화 출력을 특정 비트로 표현하는 기법이다. Zhao 등[4]에서는 가중치 및 활성화 출력에서 이상치를 포함하는 채널을 줄이는 OCS(Outlier channel splitting)기법을 도입하였다. Nagel 등[5]에서는 추가 데이터 없이 가중치의 편향과 불균형 문제를 Bias correction 과 Cross-layer equalization 기법을 통해 해결하였다. Choukroun 등[6]에서는 가중치와 활성화 출력의 부동 소수점 값과 양자화된 값 사이의 최소 평균 제곱 오차를 이용하여 양자화 오차(Quantization error)를 최소화 하였다. Banner 등[7]에서는 ACIQ(Analytical clipping for integer quantization)을 이용하여 최적의 클리핑 값 계산을 통해 양자화 오차를 최소화 하였다. Cai 등[8]에서는 배치 정규화 계층의 평균과 분산을 이용하여 원본 데이터와 유사한 증류 데이터(Distilled data)를 이용하여 최적의 양자화 범위를 얻을 수 있었다. 훈련 후 양자화 기법은 재훈련(Re-training)과정이 필요 없으므로 컴퓨팅 리소스 및 최적화 시간을 절약할 수 있으므로 빠른 배포가 가능하지만 하드웨어 구현 시 다음과 같은 단점이 존재한다. [5-7]기법은 정확도 손실을 최소화하기 위해 미세 조정된 값이 필요하고, [4, 5, 6, 8]기법은 스케일 인자(Scaling factor)를 얻기 위해서 양자화 범위(Quantization range)를 구해야 한다. 정확도 손실을 최소화하기 위해 [7-8]기법은 채널 별로 bit수를 다르게 할당해야 하고, [6-7]기법은 채널 별로 스케일 인자를 따로 구해야 한다. 또한 공통적으로 4-bit이하의 정밀도에서는 성능저하가 매우 심각하며 클리핑 값과 스케일 인자 등 양자화를 위해 필요한 파라미터들이 부동소수점으로 구현되기 때문에 하드웨어 구현 시 추가적인 연산기가 필요하다.

2. 양자화 인식 훈련(Quantization aware training)

양자화 인식 훈련 기법은 CNN모델의 연산 블록(컨벌루션 계층, 완전 연결 계층)의 입력에 양자화기(Quantizer)를 추가하여 부동 소수점으로 훈련된 파라미터를 재훈련 시키는 방법이다. 이를 통해 순방향 경로에서의 양자화 효과를 시뮬레이션 할 수 있고 추론 시 CNN 연산 블록을 고정 소수점 또는 정수로 동작시킬 수 있다. Choi 등[9]에서는 PACT(Parameterized clipping activation)함수를 정의하여 양자화 인식 훈련 과정에서 최적화된 활성화 클리핑 파라미터를 구하고 이를 이용하여 적절한 스케일 인자를 찾아 양자화 오류를 최소화 하였다. Zhang 등[10]에서는 양자화 오류를 최소화하는 방향으로 양자화기를 최적화 하는 방법을 도입하였다. Jacob 등[11]에서는 정수 연산만을 이용하여 행렬 곱셈을 수행할 수 있도록 양자화기의 동작을 정의하고 계층 융합(Layer fusion)을 도입하였다. Jung 등[12]에서는 매개 변수 화된 양자화 간격(Quantization interval)을 훈련 가능한 양자화기를 통해 최적의 양자화 간격을 찾아 작업 손실(Task loss)을 최소화 하였다. Esser 등[13]에서는 스케일 인자를 학습 가능한 파라미터로 만들어 최적의 양자화 매핑(Quantization mapping)을 학습할 수 있도록 하였다. 양자화 인식 훈련은 재훈련을 통해 스케일 인자, 클리핑 임계값 등의 양자화 파라미터를 다른 파라미터와 함께 학습을 진행함으로써 학습 후 양자화 보다는 높은 정확도를 얻을 수 있다. 하지만, [10, 12, 13]기법은 배치 정규화 계층에 대해 양자화를 하지 않았고, [9-13]기법은 양자화를 위해 필요한 파라미터가 부동 소수점으로 구현되어 하드웨어 구현 시 추가적인 연산기가 필요하다.

따라서 본 논문에서는 하드웨어 친화적인 양자화 기법으로 양자화 인식 훈련을 수행하여 스케일 인자에 대한 곱셈 및 나눗셈 연산을 Shift 연산으로 대체할 수 있도록 하였고 배치 정규화 계층 융합을 적용하여 배치 정규화 계층에서 필수적인 곱셈 및 나눗셈 연산을 제거하여 연산량을 줄였다. 그리고 오직 정수 연산만을 사용하는 기본 연산 블록에 대한 계층 별 정밀도(Layer precision) 검증을 위해 기본 연산 블록으로 조합된 CNN을 구성하여 정수 연산만으로 모든 연산이 같은 정밀도에서 수행되도록 하였다.

Ⅲ. 연구 방법

이번 장에서는 제안하는 연구 방법에 대해 설명한다.

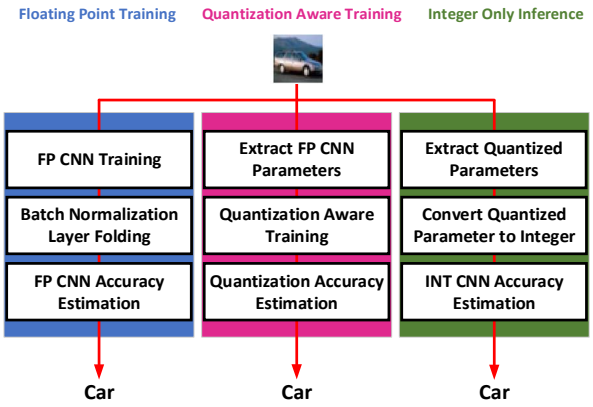


그림 1. 제안하는 기법의 전체 알고리즘 개념도
Fig. 1. Block diagram of the overall algorithm of the proposed method.

1절에서는 제안하는 기법의 전체 알고리즘 개념도에 대해 설명하고 2절에서는 하드웨어 친화적인 양자화 기법에 대해 설명을 하고 3절에서는 양자화 인식 훈련 알고리즘에 대해 설명한다. 그리고 4절에서는 오직 정수 연산만을 사용하는 기본 연산 블록에 대한 계층 별 정밀도(Layer precision)를 검증하는 방법을 설명한다.

1. 제안하는 기법의 전체 알고리즘 개념도

제안하는 기법의 전체 알고리즘 개념도는 그림 1에서 확인할 수 있다. 제안하는 알고리즘은 부동 소수점 훈련, 하드웨어 친화적인 양자화 인식 훈련, 정수 추론 등 3가지 부분으로 이루어져 있다. 부동 소수점 훈련 부분은 다음과 같이 동작한다. 부동 소수점으로 CNN을 학습 시킨 뒤 배치 정규화 계층 융합(Batch normalization layer fusion)을 수행한다. 그리고 부동 소수점으로 훈련된 파라미터를 이용하여 CNN의 성능을 평가한다. 하드웨어 친화적인 양자화 인식 훈련(Quantization aware training) 부분은 다음과 같이 동작한다. 먼저 부동 소수점으로 훈련된 파라미터를 이용하여 양자화 인식 훈련을 진행한다. 만약 부동 소수점으로 기 훈련된 파라미터가 있을 경우 부동 소수점 훈련과정을 생략할 수 있다. 양자화 인식 훈련을 통해 양자화된 부동 소수점 파라미터 및 스케일 인자를 얻을 수 있다. 이를 이용하여 양자화된 CNN의 성능을 평가한다. 정수 추론부분은 다음과 같이 동작한다. 양자화 인식 훈련으로 학습된 파라미터를 정수 파라미터로 변환 후에 정수 연산만을 사용하는 CNN을 구성하여 추론을 수행하고 성능을 평가한다.

2. 하드웨어 친화적인 양자화 기법

하드웨어 친화적인 양자화를 위해 다음과 같은 기법들을 적용하였다.

1) 균일 양자화(Uniform quantization) : 하드웨어 상에서 산술 연산을 쉽게 구현하기 위해 양자화 간격을 균일하게 설정하는 균일 양자화를 이용하였다. 균일 양자화는 실수 도메인에 존재하는 값 r을 양자화 도메인에 존재하는 값 q로 변환하기 위해 아핀 매핑(Affine mapping)을 사용하였고 수식 (1)에서 확인 할 수 있다.

$$r = s(q - z) \tag{1}$$

아핀 매핑을 통해 실수 값을 양자화된 값으로 변환하기 위해서는 양자화 매개변수 s, z가 필요하다. 스케일 인자 s는 실수 범위의 값을 정수 범위로 변환하기 위해 필요한 매개변수 이고, 영점 z는 실수 도메인에서의 0과 정수 도메인에서의 0을 맞춰주기 위해 필요한 매개변수 이며 양자화된 값들의 편향 정도를 나타낸다.

2) 균등 양자화(Symmetric quantization) : 실수 값을 양자화된 값으로 변환할 때 영점에 대한 추가적인 계산 오버헤드를 줄이고자 균등 양자화를 이용하였다. 실수 r_1, r_2을 곱셈한 결과를 r_3라고 할 때 양자화 매개 변수를 이용하여 r_3를 표현하면 수식 (2)와 같다.

$$\begin{aligned} r_3 &= s_3(q_3 - z_3) = r_1 \cdot r_2 \\ &= s_1(q_1 - z_1) \cdot s_2(q_2 - z_2) \end{aligned} \tag{2}$$

위의 식을 양자화된 값인 q_3에 대해서 정리를 하면 수식 (3)과 같은 식이 나오게 된다.

$$q_3 = z_3 + \frac{s_1 s_2}{s_3}(q_1 q_2 - q_1 z_2 - q_2 z_1 + z_1 z_2) \tag{3}$$

위의 식에서 균등 양자화를 적용하여 영점 z_1, z_2 ,z_3을 제거하면 영점에 의해 발생하는 항들이 사라지게 되어 수식 (4)와 같은 식이 나오게 되고 덧셈에 대한 계산을 줄일 수 있다.

$$q_3 = \frac{s_1 s_2}{s_3}(q_1 q_2) \tag{4}$$

3) 2의 거듭제곱 스케일 인자(Power of two scaling factor) : 수식 (4)에서 스케일 인자 s_1, s_2 ,s_3를 처리할 때 곱셈기와 나눗셈기가 필요하므로 하드웨어적인 부담이 발생하게 된다. 이러한 부담을 줄이고자 스케일 인자를 2의 거듭제곱으로 학습되도록 만들어 스케일 인자를 처리할 때 모두 Shift 연산으로만 수행될 수 있도록 하였다.

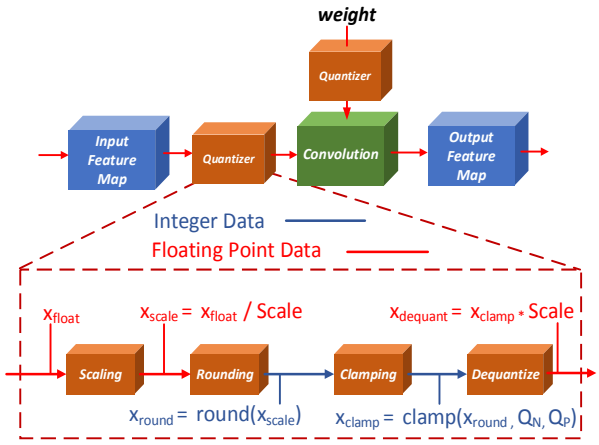


그림 2. 양자화기의 동작
Fig. 2. The operation of the quantizer.

4) 텐서 당 양자화(Per-tensor scaling) : 파라미터 및 활성화 출력에 대해서 1개의 스케일 인자로 양자화를 할 수 있도록 하였다. 이를 통해 하드웨어로 구현 시 메모리 공간을 절약할 수 있고 일관성을 유지할 수 있도록 하였다.

5) 배치 정규화 계층 융합(Batch normalization layer fusion) : 배치 정규화 계층 융합 기법을 적용하여 컨벌루션 계층의 파라미터와 배치 정규화 계층의 파라미터를 하나의 컨벌루션 계층의 파라미터로 합쳐 배치 정규화 계층의 파라미터에 대해서 따로 양자화를 진행하지 않아도 된다. 또한 배치 정규화 계층에서의 곱셈 및 나눗셈 연산을 제거하여 연산량을 줄일 수 있다. 배치 정규화 계층 융합 후 만들어진 컨벌루션 계층의 파라미터 (w_{fold}, b_{fold})는 수식 (5), (6)에서 확인할 수 있다.

$$w_{fold} = \frac{\gamma w}{\sqrt{\sigma_B^2 + \epsilon}} \tag{5}$$

$$b_{fold} = \frac{\gamma(b - \mu_B)}{\sqrt{\sigma_B^2 + \epsilon}} + \beta \tag{6}$$

위의 수식에서 γ, β, σ_B, μ_B 은 배치 정규화 계층의 파라미터이고 w, b는 배치 정규화 층 융합 이전의 컨벌루션 계층의 파라미터이다.

3. 양자화 인식 훈련 알고리즘

위에서 설명한 5가지 하드웨어 친화적인 양자화 기법을 적용한 양자화 인식 훈련(Quantization aware training) 알고리즘은 다음과 같다. 양자화기는 연산 블록(컨벌루션 계층, 완전 연결층)의 입력 쪽에 존재하여 가중치와 이전 계층의 활성화 출력에 대해 양자화를 진행한다. 첫 번째 과정은 'Scale'이다. 부동소수점 범위를 정수 범위로 매핑을 하는 역할 한다. 두 번째 과정은 'Round'이다. 이를 통해 정수 범위로 매핑이 된 부동소수점 값을 모두 정수로 변환해주는 연산을 수행한다. 세 번째 과정은 'Clamp'이다. 이 과정은 양자화 범위를 초과하는 요소를 잘라주어 양자화 수준에 맞게 값을 표현할 수 있도록 한다. 양자화 비트를 n이라고 하였을 때, 부호가 있는 수에 대해서는 $[-2^{n-1}, 2^{n-1}-1]$의 범위 안에 들어오도록 하고, 부호가 없는 수에 대해서는 $[0, 2^n - 1]$의 범위 안에 들어오도록 잘라준다.

마지막 과정은 'De-Quant' 과정이다. 이 과정은 정수로 매핑 된 값을 다시 부동소수점 범위로 변환하여 양자화 영향을 받은 부동소수점 값으로 연산이 수행된다. 양자화기의 순방향 경로에 대한 전체적인 공식은 수식 (7)과 같고, 양자화기의 동작은 그림 2에서 확인할 수 있다.

$$q(x;s) = clamp(round(\frac{x}{s}); Q_N, Q_P) \cdot s \tag{7}$$

위의 수식에서 $clamp(x; Q_N, Q_P)$함수는 Q_N보다 작은 x값은 Q_N으로 설정하고, Q_P보다 큰 x값은 Q_P로 설정한다. 그리고 $round(\cdot)$함수는 앞에서 설명한 'Round' 과정을 수행한다. 앞에서 설명한 양자화기의 동작에는 Round 연산을 통해 값들이 이산화 된다. 이는 기울기를 기반으로 파라미터를 최적화하는 경사하강법(Gradient descent) 알고리즘을 사용할 수 없으므로, STE(Straight through estimator)[15]를 이용하여 최적화를 수행한다. STE 알고리즘은 수식 (8)에서 확인할 수 있다.

$$\frac{\partial}{\partial x} round(x) = 1, \qquad round(x) \neq x \tag{8}$$

위의 수식에서 STE알고리즘을 사용하면 순방향 경로에서는 $round(\cdot)$ 함수는 Round-to-nearest-even 연산을 수행하고, 역방향 경로에서는 입력으로 받은 값을 이전 계층으로 전달한다.

4. 기본 연산 블록에 대한 계층 별 정밀도 검증

정수 연산만을 사용하는 기본 연산 블록에 대해서 계층 별 정밀도(Layer precision)를 검증하기 위해 CNN에서 자주 사용되는 연산 블록 3가지를 TypeA, TypeB, TypeC라고 명명하였고 구조는 그림 3에서 확인할 수 있다. TypeA블록은 2개의 컨벌루션 계층을 연속적으로

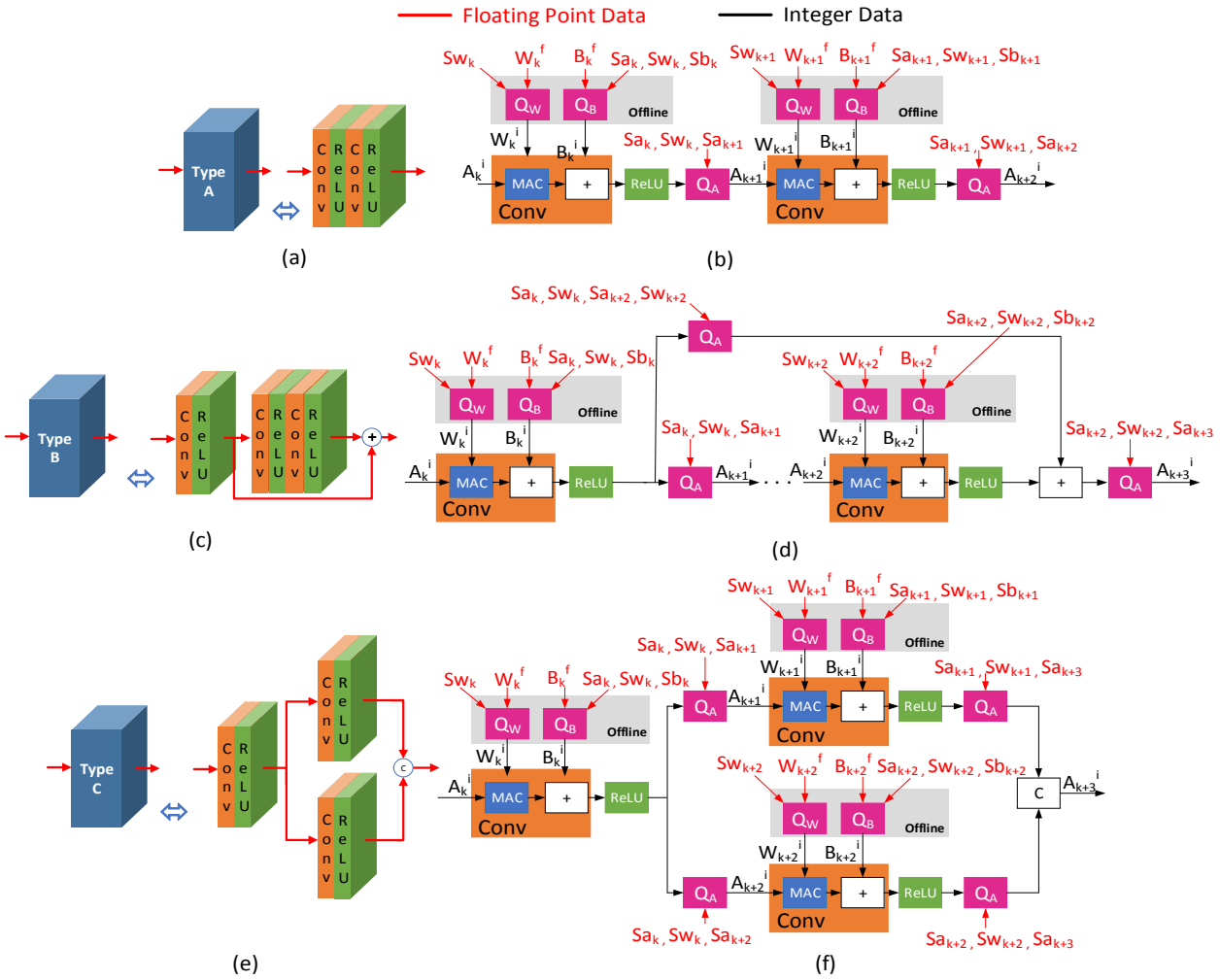


그림 3. 부동 소수점 연산 기본 블록과 정수 연산만을 사용하는 기본 블록 ((a) 부동 소수점 연산 TypeA 블록, (b) 정수 연산 TypeA 블록, (c) 부동 소수점 연산 TypeB 블록, (d) 정수 연산 TypeB 블록, (e) 부동 소수점 연산 TypeC 블록, (f) 정수 연산 TypeC 블록)

Fig. 3. Basic blocks for floating-point operations and basic blocks using only integer operations ((a) Floating-point operation TypeA block, (b) Integer operation TypeA block, (c) Floating-point operation TypeB block, (d) Integer operation TypeB block, (e) Floating-point operation TypeC block, (f) Integer operation TypeC block).

쌓은 구조이고 VGGNet[16]에서 제안되었다. TypeB블록은 3개의 컨벌루션 계층을 사용하는데, 앞쪽의 컨벌루션 계층의 출력을 2가지 경로로 나누어 첫 번째 경로는 중간, 뒤쪽의 컨벌루션 계층을 통과하지 않고 우회하는 경로 이며 이를 우회경로(Bypass connection)이라고 한다. 두 번째 경로는 중간, 뒤쪽의 컨벌루션 계층을 통과한다. 그리고 2가지 경로의 값을 더하는 연산을 수행하는 구조이고 ResNet[17]에서 제안되었다. TypeC블록은 3개의 컨벌루션 계층을 사용하는데, 앞쪽의 컨벌루션 계층의 출력을 2가지 경로로 나누어 각각 다른 컨벌루션 계층을 통과한다. 그리고 2가지 경로의 출력을 채널 단위로 쌓는 연산(Channel-wise concatentation)을 수행하는 구조이고 SqueezeNet[18]에서 제안되었다.

기본 연산 블록에 대한 계층 별 정밀도(Layer precision)를 검증하기 위해서 먼저 그림 3에서 사용된 용어를 정리한다. 그림 3의 (b), (d), (f)에서 볼 수 있는 붉은 선은 부동 소수점 데이터를 의미하고, 검은 선은 정수 데이터를 의미한다. 각 영문자의 윗 첨자는 데이터의 유형을 나타내는데 f는 부동 소수점 데이터를 의미하고 i는 정수를 의미한다. 또한 아래 첨자 $k, k+1, k+2, k+3$는 각 계층의 번호를 나타낸다. 그리고 Q_A, Q_W, Q_B는 각각 활성화 출력, 가중치, 편향에 대한 양자화기이다. Sw_k, Sw_{k+1}, Sw_{k+2}는 가중치의 스케일 인자이고 Sb_k, Sb_{k+1}, Sb_{k+2}는 편향의 스케일 인자이고 Sa_k, Sa_{k+1}, Sa_{k+2}는 활성화 출력의 스케일 인자이다. 그리고 A, W, B는 각각 활성화 출력, 가

중치, 편향을 의미한다. 회색 바탕에 $Offline$으로 적혀 있는 부분은 양자화된 부동 소수점 파라미터들을 정수로 변환하는 과정인데, 이는 추론 과정 이전에 수행되어 메모리에 정수 파라미터가 저장되며 가중치와 편향에 대한 변환공식은 수식 (9), 수식 (10)에서 확인할 수 있다.

$$W_k^i = clamp(round(W_k^f \ll Sw_k);\ Q_N, Q_P) \quad (9)$$

$$B_k^i = clamp(round(B_k^f \ll Sb_k);\ Q_N, Q_P) \gg (Sb_k) \ll (Sa_k + Sw_k) \quad (10)$$

TypeA 블록은 다음과 같이 검증할 수 있다. 정수로 변환된 가중치와 컨벌루션 계층의 입력에 대해서 MAC(Multiply - accumulate operation)연산을 수행하고 그 결과에 대해서 정수로 변환된 편향을 더해준다. 이 때 편향은 수식 (10)에서 볼 수 있듯이 MAC연산의 결과와 동일한 정밀도로 스케일링이 되어야 한다. 연산결과는 수식 (11)에서처럼 활성화 함수를 거치고, 활성화 출력 양자화기의 입력으로 들어가 수식 (12)에서 볼 수 있는 Down-scaling 및 Re-scaling과정이 진행된다.

$$C_k^i = ReLU((A_k^i * W_k^i) + B_k^i) \quad (11)$$

$$A_{k+1}^i = clamp(round((C_k^i \gg (Sa_k + Sw_k)) \ll Sa_{k+1});\ Q_N, Q_P) \quad (12)$$

수식 (11)에서 ReLU는 활성화 함수인 ReLU를 뜻하고, C_k^i 는 MAC 연산을 수행하고 편향을 더한 값에 ReLU를 적용한 결과이다. 수식 (11)의 값이 활성화 출력의 양자화기의 입력으로 들어가, 현재 계층의 스케일 인자의 영향을 취소하고 다음 계층의 입력 정밀도에 맞게 Down-scaling 및 Re-scaling 과정이 진행된다. 위의 연산은 모두 정수로 수행이 되고, 스케일 인자는 모두 Shift 연산으로 수행이 된다.

TypeB 블록을 검증하기 위해서 다음과 같은 과정이 수행된다. 먼저 k번째 계층에 대해서는 수식 (9)-(11)의 과정을 수행한다. 그리고 C_k^i을 2개의 경로로 분주한다. 우회 경로 쪽은 k번째 계층의 스케일 인자의 영향을 취소하고 덧셈 연산을 수행하게 되는 다른 계층 출력의 스케일과 동일하게 맞추는 Re-scaling을 수행하는 활성화 출력 양자화기를 통과한다. 이에 해당하는 연산은 수식 (13)에서 볼 수 있다.

$$C_{k,bp}^i = clamp(round((C_k^i \gg (Sa_k + Sw_k)) \ll (Sa_{k+2} + Sw_{k+2}));\ Q_N, Q_P) \quad (13)$$

위의 수식에서 $C_{k,bp}^i$는 우회 경로 활성화 출력 양자화기의 출력이다. 다른 경로는 TypeA 블록에서 사용된 수식 (12) 과정에 해당하는 활성화 출력 양자화기를 통과한 뒤, k+1번째 계층을 통과한다. 그리고 k+2번째 계층에서는 수식 (11)까지만 수행하고, 우회경로의 데이터와 덧셈을 수행한 뒤 k+2번째 계층의 스케일 인자의 영향을 취소하고 k+3번째 계층의 입력 정밀도에 맞게 Down-scaling 및 Re-scaling과정을 수행하는 활성화 출력 양자화기를 통과한다. 이에 해당하는 연산은 수식 (14)-(15)에서 확인할 수 있다.

$$C_{k+2,add}^i = C_{k+2}^i + C_{k,bp}^i \quad (14)$$

$$A_{k+3}^i = clamp(round((C_{k+2,add}^i \gg (Sa_{k+2} + Sw_{k+2})) \ll Sa_{k+3});\ Q_N, Q_P) \quad (15)$$

마지막으로, TypeC 블록은 다음과 같이 검증할 수 있다. TypeB블록과 동일하게 k번째 계층에 대한 연산을 수행한다. 그리고 2가지 경로에 서로 다른 컨벌루션 계층이 존재하므로 각각의 컨벌루션 계층의 입력이 고려된 활성화 출력 양자화기를 만들어야 한다. k+1번째, k+2번째의 활성화 출력 양자화기에 대한 동작은 각각 수식 (16), (17)에서 확인할 수 있다.

$$A_{k+1}^i = clamp(round((C_k^i \gg (Sa_k + Sw_k)) \ll Sa_{k+1});\ Q_N, Q_P) \quad (16)$$

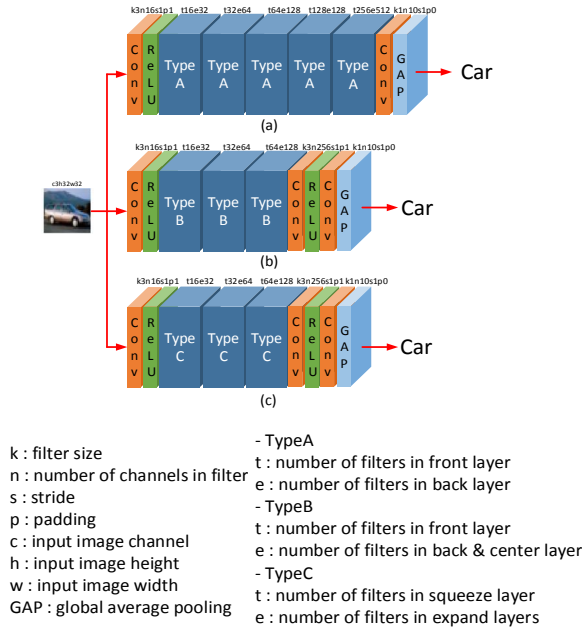
$$A_{k+2}^i = clamp(round((C_k^i \gg (Sa_k + Sw_k)) \ll Sa_{k+2});\ Q_N, Q_P) \quad (17)$$

양자화기를 통과한 각각의 활성화 출력은 컨벌루션 계층을 통과한 뒤 다음 계층의 입력 정밀도가 고려되는 활성화 출력 양자화기를 통과한다. 다음 계층은 2가지 경로 모두 동일하기 때문에 동일한 스케일 인자를 사용하여 Down-scaling 및 Re-scaling 과정이 진행된다. 그리고 채널 단위로 쌓는 연산을 수행한다. 채널 단위 쌓는 연산은 값에 영향을 주지 않고 채널을 서로 합치기 때문에 해당 연산을 위한 양자화기는 따로 필요 없다.

Ⅳ. 실 험

1. 실험 환경 설정

하드웨어 친화적인 양자화 인식 훈련(Quantization aware training) 및 정수 연산만을 사용하는 CNN의 추론 성능을 확인하고자 다음과 같은 실험 환경을 설정하였다. Intel(R) Xeon(R) Gold 5120 CPU@2.2GHz,



k : filter size
n : number of channels in filter
s : stride
p : padding
c : input image channel
h : input image height
w : input image width
GAP : global average pooling

- TypeA
t : number of filters in front layer
e : number of filters in back layer
- TypeB
t : number of filters in front layer
e : number of filters in back & center layer
- TypeC
t : number of filters in squeeze layer
e : number of filters in expand layers

그림 4. 기본 연산 블록으로 구성된 CNN 구조 ((a) TypeA (b) TypeB (c) TypeC)
Fig. 4. CNN structure composed of basic operation block ((a) TypeA (b) TypeB (c) TypeC).

180Gb RAM과 NVIDIA Tesla V100(SXM2) GPU로 구성된 하드웨어를 사용하였고, 소프트웨어는 Python에서 Pytorch 라이브러리를 이용하여 부동 소수점 훈련 및 추론, 양자화 인식 훈련 및 추론, 정수 추론을 구현하였다. 데이터 세트는 이미지 분류에서 많이 사용되는 CIFAR-10을 사용하였으며, 50000개의 훈련 데이터 세트, 10000개의 검증 데이터 세트로 구성되어있다. 그리고 각 이미지는 3개의 채널을 가지고 있으며 가로, 세로 각각 32픽셀이다. 정수 연산만을 사용하는 CNN의 추론 결과를 검증하기 위해서 그림 4와 같은 TypeA, TypeB, TypeC를 기본 연산 블록으로 사용하는 영상 분류를 위한 CNN을 구성하였다. 부동 소수점 훈련에서는 정수 추론 시의 입력을 부호 없는 8-bit로 가정을 하였기 때문에 일반적으로 데이터 전처리에 사용되는 정규화(Normalization)을 수행하지 않았다. 양자화 인식 훈련에서도 정수 추론 시의 입력을 부호 없는 8-bit로 가정하였기 때문에 입력 계층은 8-bit로 양자화 되도록 설정하였으며 일관성을 위해 출력 계층 역시 8-bit로 양자화 되도록 하였다.

2. 실험 결과

기본 연산 블록(TypeA, TypeB, TypeC)으로 구성된 CNN에 대해서 부동 소수점 훈련, 양자화 인식 훈련(Quantization aware training), 정수 추론의 결과는 표 1에서 확인할 수 있으며, CNN 추론과정에서 발생하는 연산량 단위로 부동 소수점 연산량인 FLOPs와 정수 연산량인 IOPs로 나타낼 수 있다. TypeA, TypeB, TypeC를 기본 연산 블록으로 구성한 CNN의 파라미터 개수는 부동 소수점 훈련에서 각각 1.96M, 0.8M, 1.76M 개로 구현하였다. 그리고 [14]기법 및 정수 추론에서는 8-bit로 연산이 수행되었고 스케일 인자가 추가되었지만 무시할 수 있을 만큼의 작은 크기를 갖기 때문에 부동 소수점 대비 1/4만큼의 파라미터 개수로 근사화 할 수 있다. 정수 추론의 분류 정확도는 부동 소수점 정확도와 비교하였을 때 약 0.71% 이하로 확인되었고 [14] 기법 대비 정확도 하락이 약 0.13% 이하로 확인되었다. [14]기법에서는 부동 소수점 대비 정확도 하락이 약 0.1% 이하로 확인되었다. [14]기법은 추론 과정에서 'De-Quant' 과정을 거치기 때문에 본 논문에서 제안한 방법 보다는 정확도가 높을 수는 있지만 부동 소수점 연산을 위한 부동소수점 연산기가 추가적으로 필요하

표 1. 부동소수점 추론 결과, 양자화 인식 추론 결과, 정수 추론 결과 성능 비교
Table1. Performance comparison of floating-point inference results, quantization aware inference results, and integer inference results.

CNN Structure	Method	Accuracy (%)	Parameters (M)	Operations	Remark
TypeA	FP32	90.34	1.96	33M FLOPs	Floating-point
	[14]	89.93	0.49	33M FLOPs	Need 'De-Quant'
	Ours	89.99	0.49	33M IOPs	Integer only
TypeB	FP32	89.45	0.8	19M FLOPs	Floating-point
	[14]	88.86	0.2	19M FLOPs	Need 'De-Quant'
	Ours	88.74	0.2	19M IOPs	Integer only
TypeC	FP32	83.14	1.76	5M FLOPs	Floating-point
	[14]	83.60	0.44	5M FLOPs	Need 'De-Quant'
	Ours	83.47	0.44	5M IOPs	Integer only

다. 하지만 본 논문에서 제안한 방법은 CNN을 구성하는 모든 파라미터가 정수이고 추론과정에서의 연산 역시 모두 정수로 수행되기 때문에 정확도 하락이 [14]기법 보다는 크지만 정수 연산기만을 사용하고 별도의 부동 소수점 연산기가 필요 없다는 장점이 있다. 따라서 제안한 기법으로 CNN을 하드웨어로 구현 시 연산량, 전력 소모, 하드웨어 면적 등의 측면에서 성능이 우수할 것으로 예상된다.

V. 결론 및 향후 연구 방향

본 논문에서는 하드웨어 친화적인 양자화 기법을 적용하여 양자화 인식 훈련을 수행하였고 스케일 인자에 대한 곱셈 및 나눗셈 연산을 Shift 연산으로 수행하도록 만들었다. 또한 배치 정규화 계층 융합을 적용하여 배치 정규화 계층에서의 곱셈 및 나눗셈 연산을 제거하여 연산량을 줄일 수 있었다. 그리고 오직 정수 연산만을 사용하는 기본 연산 블록에 대한 계층 별 정밀도(Layer precision)를 검증하는 방법을 제안하고 기본 연산 블록을 조합하여 정수 연산만을 사용하는 CNN을 구성하여 모든 연산이 같은 정밀도에서 수행되도록 하였다. 실험 결과 부동 소수점 대비 약 1/4만큼의 파라미터를 사용하여 정확도 하락이 0.71%이하로 확인되었고 [14]기법과 비교하여 정수 연산만을 사용하여 연산량을 최적화 하였으며 정확도 하락이 0.13%이하로 확인되었다. 본 연구의 후속 연구로서 향후 4bit로 양자화 인식 훈련(Quantization aware training) 수행 및 정수 연산만을 사용하는 CNN 구현 및 그에 대한 성능을 평가하고 제안된 CNN을 FPGA에 구현 및 성능 검증을 진행할 예정이다.

ACKNOWLEDGMENT

이 논문은 과학기술정보통신부 및 정보통신산업진흥원의 '고성능 컴퓨팅 지원' 사업으로부터 지원과 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원 지원(No.2020-0-02221, 인공지능산업원천기술개발 사업)을 받아 수행된 연구임.

REFERENCES

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

[2] K. B. Lee, D. K. Shin, "Unaligned Pruning for Fast DNN Inference on Mobile Devices", Korea Computer Congress, pp. 1412-1414, 2019.

[3] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network", arXiv preprint arXiv:1503.02531, 2015.

[4] R. Zhao, Y. Hu, J. Dotzel, C. De Sa, and Z. Zhang, "Improving neural network quantization without retraining using outlier channel splitting", arXiv preprint arXiv:1901.09504, 2019.

[5] M. Nagel, M. V. Baalen, T. Blankevoort, and M. Welling, "Data-free quantization through weight equalization and bias correction", In Proceedings of the IEEE International Conference on Computer Vision, pp. 1325-1334, 2019.

[6] Y. Choukroun, E. Kravchik, F. Yang, and P. Kisilev, "Low-bit quantization of neural networks for efficient inference" In 2019 IEEE/CVF International Conference on Computer Vision Workshop, pp. 3009-3018, 2019.

[7] R. Banner, Y. Nahshan, and D. Soudry, "Post training 4-bit quantization of convolutional networks for rapid-deployment", In Advances in Neural Information Processing Systems, pp. 7950-7958, 2019.

[8] Y. Cai, Z. Yao, Z. Dong, A. Gholami, M. W. Mahoney, and K. Keutzer, "Zeroq: A novel zero shot quantization framework", In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13169-13178, 2020.

[9] J. Choi, Z. Wang, S. Venkataramani, P. I. J. Chuang, V. Srinivasan, and K. Gopalakrishnan, "Pact: Parameterized clipping activation for quantized neural networks", arXiv preprint arXiv:1805.06085, 2018.

[10] D. Zhang, J. Yang, D. Ye, and G. Hua, "Lq-nets: Learned quantization for highly accurate and compact deep neural networks", In Proceedings of the European conference on computer vision, pp. 365-382, 2018.

[11] B. Jacob, S. Kligys, B. Chen, M.Zhu, M. Tang, A. Howard, and D. Kalenichenko, "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, 2018.

[12] S. Jung, C. Son, S. Lee, J. Son, J. J. Han, Y. Kwak, and C. Choi, "Learning to quantize deep networks by optimizing quantization intervals with task loss", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,pp. 4350-4359, 2019.
[13] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned step size quantization", arXiv:1902.08153, 2019.
[14] S. R. Jain, A. Gural, M. Wu, and C. H. Dick, "Trained quantization thresholds for accurate and efficient fixed-point inference of deep neural networks", arXiv preprint arXiv:1903.08066, 2019.
[15] P. Yin, J. Lyu, S. Zhang, S. Osher, Y. Qi, and J. Xin, "Understanding straight-through estimator in training activation quantized neural nets", arXiv preprint arXiv:1903.05662, 2019.
[16] K. Simonyan, A. Zisserman, "Very deep convolutional networks for large-scale image recognition", arXiv:1409.1556, 2014.
[17] 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, pp. 770-778, 2016.
[18] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet:AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size", arXiv:1602.07360, 2016.

저 자 소 개

김 재 명(학생회원)
2020년 인하대학교 전자공학과 학사 졸업.
2020년～현재 인하대학교 전기컴퓨터공학과 석사 과정.
<주관심분야 : SoC 설계, 딥러닝, 영상처리>

김 용 우(정회원)
2007년 인하대학교 전자공학과 졸업 (학사).
2009년 인하대학교 전자공학과 졸업 (석사).
2019년 KAIST 전기 및 전자공학과 졸업 (공학박사).
2009년～2017년 ㈜실리콘웍스 선임연구원
2019년～2020년 한국항공우주연구원 선임연구원
2020년 3월～현재 상명대학교 시스템반도체공학과 교수
<주관심분야 : SoC 설계, 영상 신호처리, 딥러닝, 고속인터페이스/디스플레이 IC>