CNN 훈련 속도 개선을 위한 합성곱 후처리 데이터 흐름 최적화

우고은, 김형원*
충북대학교, *충북대학교

woogoeun@cbnu.ac.kr, *hwkim@cbnu.ac.kr

Optimization of Convolutional Post-processing Data Flow for Accelerating CNN Training

Goeun Woo, Hyungwon Kim *
Chungbuk Univ., *Chungbuk Univ.

요 약

본 논문은 신경망 훈련에서 배치 정규화 시에 요구되는 후처리(평균, 분산 계산 이후의 배치 정규화, 활성화 함수 적용, 풀링 등)의 외부 메모리(e.g., Dynamic Random Access Memory, DRAM) 접근을 효율적으로 처리하기 위한 해결 방안을 제시한다. 본 연구는 후처리용 입력 버퍼로 두 개의 FIFO를 도입하고, 풀링 과정을 고려한 데이터 흐름을 구성하여 각 버퍼에서 두 개의 column을 순차적으로 처리한다. You Only Look Once version 2-Tiny (YOLOv2-Tiny)에 해당 입력 버퍼를 적용하여 Verilog 시뮬레이션을 검증한 결과, 기존 방법 대비 속도는 29.9% 향상되었고, 버퍼 크기는 4/[타일 width] 로 감소하였다.

Ⅰ. 서 론

딥러닝 기술의 발전은 지속적으로 가속화되고 있으며, 그 중에서도 모바일 기기에서의 실시간 데이터 처리와 학습 능력이 중요해지고 있다. 특히, Neural Processing Unit (NPU)와 같은 전용 신경망 연산 하드웨어는 대량의 데이터를 처리하며 복잡한 모델을 학습할 수 있는 능력이 요구된다[1].

합성곱 신경망(Convolutional Neural Network, CNN) 훈련에서는 배치 정규화(Batch Normalization, BN)가 필수적인데, 이 기술은 내부 공변량 변화를 줄이고 솔루션 스페이스를 부드럽게 만들어 신경망의 수렴 속도를 빠르게 하도록 돕는다[2-3]. 하지만, 이 과정에서 미니 배치의 합성곱 결과로부터 평균과 분산을 계산하여 이를 기반으로 정규화를 수행해야 하므로, 계산된 값에 대한 재접근이 필요하다. 또한, 뒤따르는 활성화 함수 적용과 풀링 레이어에도 영향을 주기 때문에, 효율적으로 데이터를 가져오는 것이 중요하다.

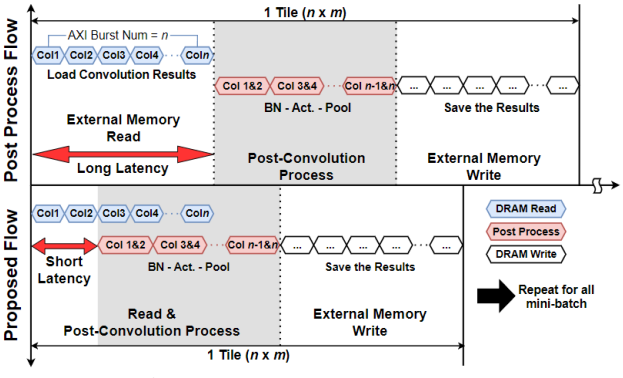

그림 1. 후처리 단계의 구성 및 제안하는 흐름

이전 연구[4]에서는 내부 메모리에 모든 값을 저장하여 사용하고 있지만, 본 논문에서는 면적 소모를 최소화하기 위해 외부 메모리인 Dynamic Random Access Memory (DRAM)을 사용하였다. 본 연구의 주요 목표는 외부 메모리 접근으로 인한 처리 지연을 최소화하고, 이와 동시에 추가되는 입력 버퍼로 인한 회로 면적 증가를 줄이는 것이다. 이를 달성하기 위해, forward propagation 과정에서 배치 정규화를 포함한 레이어에 대해 효율적인 입력 버퍼와 데이터 흐름을 구현하였다.

Ⅱ. 본론

A. 배경 설명

현재 대부분의 CNN 아키텍처, 예를 들어 이미지 분류에 사용되는 VGG 와 ResNet, 그리고 객체 탐지 모델인 You Only Look Once (YOLO)는 훈련의 forward propagation 과정에서 배치 정규화를 포함한 레이어가 다수 존재한다. 이러한 레이어의 처리는 mini-batch 전체의 이미지를 내부 메모리에 일괄적으로 로드하는 것이 아닌, 이미지의 한 부분을 적절한 타일 단위로 나누어 파이프라인 방식으로 연산을 진행한다.

그림 1 은 이러한 합성곱 후처리 과정을 나타낸다. 합성곱 연산과 배치 정규화의 평균, 분산 계산 이후에 그 결과를 다시 불러와 배치 정규화, 활성화 함수 적용, 풀링 등의 후처리 과정을 수행한다. 여기서 데이터 접근으로 인한 지연 시간이 길어지는 문제가 발생하게 되어 처리 속도에 큰 영향을 미치게 된다. 따라서 본 연구는 읽기와 후처리를 동시에 연속적으로 처리할 수 있는 효율적인 입력 버퍼를 제안한다.

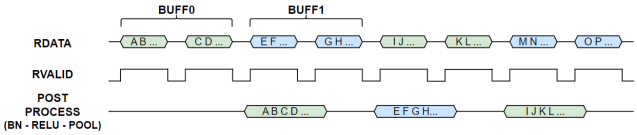

그림 2. 후처리를 위한 듀얼 입력 버퍼 다이어그램

B. 듀얼 입력 FIFO 를 통한 지연 최소화

이러한 외부 메모리 접근이 초래하는 시간 지연을 줄이기 위해, 그림 2 의 방법을 제시한다. 한 타일을 처리할 때, 모든 데이터를 기다리지 않고 두 column 단위로 데이터를 처리한다. 두 column 씩 처리하는 이유는 풀링이 요구되는 레이어의 경우, 근접한 네 개의 픽셀을 연속적으로 전달해주기 위함이다. 이 목표를 달성하기 위해, 본 연구는 듀얼 First In First Out (FIFO) 버퍼를 도입하여 두 column 씩 저장하고, 이를 활용하여 타일 단위의 계산을 외부 메모리 읽기와 병렬로 진행한다. 이 방식은 전체 타일 데이터를 저장하던 기존의 큰 버퍼 대신, 두 column 만을 저장하는 두 개의 작은 버퍼를 사용하여 내부 메모리 크기를 줄이고 처리 속도를 크게 향상시킨다. 그림 2 에서 나타내는 동작 방식은 다음과 같다:

(1) 처음 BUFF0 가 DRAM 에서 두 column 을 읽어 저장한다. 이 시점에 후처리 회로는 대기 상태에 있다. (2) 이후, BUFF 1 에 연속적으로 다음 데이터를 저장하면서 BUFF 0 에 저장된 데이터에 대한 후처리를 시작한다. (3) 한 버퍼의 처리가 진행되는 동안, 해당 버퍼 처리가 완료될 때까지 DRAM 의 데이터 입력을 중지하고, 완료되는 'done' 신호에 의해 다음 데이터 읽기를 시작한다. (4) 이 과정은 합성곱 출력 feature 의 모든 채널 정보에 대해 반복되며, 마지막 채널의 데이터 처리가 끝나면 다음 버퍼로 넘어간다.

C. 실험 및 결과

제안한 입력 버퍼를 포함한 디자인은 Vivado 환경에서 Verilog 언어를 사용하여 RTL 로 설계하였고, YOLO version2-Tiny 객체 탐지 CNN 모델을 목표로 하여 PASCAL Visual Object Classes(VOC) dataset 을 적용하였다.

그림 3 은 제안하는 듀얼 입력 버퍼의 Vivado 시뮬레이션 결과의 한 부분을 보여준다. DRAM 에서 두 column 을 읽어오고(w_en=1), 버퍼의 full 신호가 high 가 되면 후처리를 시작하며(r_en=1), 동시에 다음 버퍼를 채운다. 두 column 읽기가 완료되자마자 즉시 후처리를 연속적으로 처리함으로써, 한 타일의 모든 픽셀 데이터를 읽어온 뒤 진행하는 방식 대비 대략 29.9%의 속도 향상을 달성할 수 있다. 또한, 두 버퍼의 가로 크기는 고정적으로 2 로 설정되어 있으며, 기존 타일의 width 에 따라 감소 비율이 크게 증가한다. 예를 들어, 타일 width 가 100 일 경우 버퍼의 가로는 원래의 100 에서 2×2(듀얼)로 줄어들어, 원래 크기의 1/25 로 감소하고, width 가 200 일 경우 1/50 으로 감소한다. 따라서 전체 버퍼 크기는 기존의 width × height × channel 에서 2(width 고정) × 2(듀얼) × height × channel 로 변경되었다.

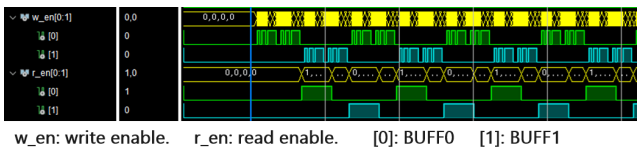

그림 3. 듀얼 입력 FIFO 의 Vivado 시뮬레이션 결과

Ⅲ. 결론

본 논문에서는 신경망 훈련의 속도를 향상시키기 위해 후처리 단계에서 외부 메모리 접근의 영향을 최소화하는 방안을 제시하였다. 이를 위해 두 개의 작은 FIFO 를 활용하여 DRAM 으로부터 데이터를 가져와 실시간으로 처리할 수 있는 시스템을 개발했다. 이러한 접근 방식은 CNN 의 forward propagation 단계에서 처리 효율을 크게 개선하고, 본 논문에서 제시한 입력 버퍼와 데이터 흐름 전략이 실질적인 성능 향상을 이끌어내는 것을 확인하였다.

현재 연구에서는 데이터의 읽기와 후처리를 동시에 처리하여 훈련 속도를 개선하였다. 그러나 결과 데이터의 기록 과정은 아직 별도로 진행되고 있다. 향후 연구에서는 이 결과 기록 과정도 읽기 및 후처리와 동시에 이루어지도록 시스템을 개선할 계획이다. 이 개선이 성공적으로 이루어질 경우, 본 논문에서 소개한 방법에 비해 추가적으로 약 44%의 속도 향상을 기대할 수 있으며, 기존 방법 대비 총 60%의 속도 향상이 가능할 것으로 예상된다.

ACKNOWLEDGMENT

This work was supported by Regional Leading Research Center (RLRC) of the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. 2022R1A5A8026986) and supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-01304, Development of Self-Learnable Mobile Recursive Neural Network Processor Technology). It was also supported by the MSIT (Ministry of Science and ICT), Korea, under the Grand Information Communication Technology Research Center support program (IITP-2024-2020-0-01462) supervised by the IITP (Institute or Information & communications Technology Planning & Evaluation).

참 고 문 헌

[1] Lee, Jinsu, and Hoi-Jun Yoo. "An overview of energy-efficient hardware accelerators for on-device deep-neural-network training." *IEEE Open Journal of the Solid-State Circuits Society* 1 (2021): 115-128.

[2] Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *International conference on machine learning*. pmlr, 2015.

[3] Santurkar, Shibani, et al. "How does batch normalization help optimization." *Advances in neural information processing systems* 31 (2018).

[4] Ting, Yu-Sheng, Yu-Fan Teng, and Tzi-Dar Chiueh. "Batch normalization processor design for convolution neural network training and inference." *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2021.