RLC 압축 기법 적용을 통한 하드웨어 연산 효율 및 메모리 사용 절감 효과 평가

이종윤, 이하림*
금오공과대학교,
20246116@kumoh.ac.kr, *hrlee@kumoh.ac.kr

Evaluation of RLC Compression for Improving Computational Efficiency and Reducing Memory Usage

Lee Jong Youn, Lee Harim*
Kumoh National Institute of Technology

요 약

딥러닝 모델의 복잡도 증가로 인해 연산량과 메모리 사용량이 급증하면서, 자원이 제한된 모바일 및 엣지 환경에서 효율적인 신경망 연산 및 메모리 활용의 필요성이 대두되고 있다. 이에 본 연구는 완전 연결 층에 Run-Length Coding(RLC) 기반 압축 기법을 적용한 하드웨어 아키텍처를 제안하고, Verilog HDL로 구현한다. RTL 시뮬레이션 결과, 입력 데이터 내 0 값 비율이 높을수록 메모리 사용량과 연산량이 효과적으로 감소함이 확인된다. 제안된 구조는 희소성이 높은 신경망 데이터에 적합하며, 연산 자원이 제한된 환경에서도 높은 효율을 기대할 수 있다.

Ⅰ. 서 론

최근 딥러닝 기술의 급속한 발전과 함께 신경망 모델은 점점 더 깊고 복잡한 구조로 진화하고 있으며, 이에 따라 처리해야 할 파라미터 수 또한 기하급수적으로 증가하고 있다 [1-3]. 특히 최근 개발되는 딥러닝 네트워크들은 수백만 개의 가중치를 포함하고 있어, 합성곱 연산이나 행렬 연산 과정에서 수백만 회 이상의 곱셈 및 덧셈 연산(Multiply-ACcumulate, MAC)이 발생하게 된다. 이러한 대규모 연산은 해당 딥러닝 네트워크의 추론 과정에서 하드웨어 메모리 및 대역폭 요구량을 증가시킨다.

이러한 딥러닝 네트워크 추론의 대규모 연산 구조를 효율적으로 처리하기 위해 최근 추론향 AI 하드웨어 가속기 개발 연구가 대두되고 있다. 추론향 AI 하드웨어 가속기는 모바일 기기나 엣지 디바이스처럼 자원이 제한된 환경을 목표로 설계되는 경우가 많다. 따라서, 이러한 AI 하드웨어 가속기에서 대규모 연산을 처리하기 위해서는 설계 시 하드웨어 연산 효율 및 메모리 사용의 최적화를 고려해야 한다 [4].

이에 본 연구에서는 하드웨어에서 딥러닝 네트워크 구동 시 최적의 메모리 및 연산 효율 향상을 위해 완전 연결 네트워크에 Run-Length Coding(RLC) 압축 기법을 적용하는 하드웨어 구조를 설계한다. 해당 적용을 실제 하드웨어에서 구현하기 위해 하드웨어 기술 언어인 Verilog HDL을 사용하여 설계 및 구현한다. 구현된 하드웨어는 정수형 데이터를 처리하도록 설계되었으며 메모리 최적화와 연산 효율 향상을 고려하여 하드웨어 구조를 구성하였다. 또한, RLC 압축 기법의 적용 유무에 따른 하드웨어 성능을 비교 분석함으로써 제안 방식의 실효성을 정량적으로 평가한다.

Ⅱ. 하드웨어 설계

본 장에서는 RLC 압축 기법에 대한 설명, RLC 압축 기법을 적용한 완전 연결 층 네트워크를 하드웨어로 설계하는 과정에 대해 설명한다.

2.1. RLC 압축 기법

RLC는 동일한 값이 반복될 때 해당 값과 반복 횟수만으로 데이터를 표현하는 방식의 압축 기법으로 신경망처럼 0 값이 자주 등장하는 데이터 구조에 효과적이다 [4]. 완전 연결 네트워크는 각 뉴런이 이전 층의 모든 뉴런과 연결되어 있어 연산량과 가중치 수가 많고 ReLU와 같은 활성화 함수를 거치면서 출력값 중 많은 수가 0으로 변하는 특징을 갖는다. 본 연구는 이러한 희소성(sparsity)에 주목하여 RLC를 통해 메모리에 저장되는 데이터의 양을 줄이고 0 값을 배제한 연산을 통해 전체 연산량을 효과적으로 감소시키고자 한다.

2.2. RLC가 적용된 완전 연결 층 아키텍처

그림 1은 RLC가 적용된 완전 연결 층의 전체 하드웨어 아키텍처를 나타낸다. 해당 아키텍처는 입력 데이터를 전달하는 데이터 전송 모듈(Data Transfer Module), 입력값에 대해 압축 연산을 수행하는 RLC 인코더 모듈(RLC Encoder Module), 그리고 신경망 연산을 수행하는 층 모듈(Layer Module)로 구성된다.

해당 아키텍처의 데이터 전송 및 연산 흐름은 다음과 같다. 이전 층 모듈에서 생성된 출력 노드 값은 데이터 전송 모듈을 통해 데이터를 한 개 단위로 RLC 인코더 모듈로 송신하며 해당 인코더 모듈에서는 수신 데이터를 압축하여 메모리에 저장한다. 저장된 RLC 데이터는 다음 층 모듈에서

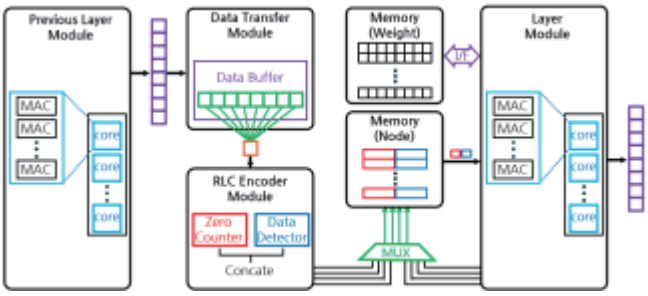

그림 1. 완전 연결 층의 전체 하드웨어 아키텍처.

디코딩되어 신경망 연산에 활용되며 최종 출력 노드 값을 생성한다. 인코더 모듈의 데이터 저장 과정과 층 모듈의 데이터 읽기 및 디코딩 과정에서의 메모리 인터페이스는 MUX (Multiplexer)를 통해 각 모듈 간 제어된다.

2.3. 데이터 전송 모듈 (Data Transfer Module)

본 모듈은 층 모듈에서 생성된 출력 노드 값을 데이터 버퍼에 저장한 후, 인코더 모듈로 데이터를 한 개 단위로 전송하는 역할을 한다. 인코더 모듈이 수신 준비 상태가 되면, 데이터를 순차적으로 전송한다.

2.4. RLC 인코더 모듈 (RLC Encoder Module)

본 모듈은 데이터 전송 모듈로부터 데이터를 수신하여 RLC 압축을 수행한다. 내부적으로는 연속된 0 값을 계수하는 제로 카운터(Zero Counter)와, 0이 아닌 데이터를 감지하는 데이터 감지기(Data Detector)로 구성된다. 제로 카운터의 출력(연속된 0의 개수)과 데이터 감지기의 출력(0이 아닌 값)을 결합하여 압축된 출력 데이터를 생성한다.

2.5. 층 모듈 (Layer Module)

본 모듈은 메모리에 저장된 RLC 압축 데이터를 읽어 디코딩한 후, 이를 기반으로 완전 연결 층의 신경망 연산을 수행한다. 해당 모듈은 병렬 구조의 N개 코어(Core)로 구성되어 있으며, 각 코어는 하나의 출력 노드 연산을 담당한다. 이에 따라 총 N개의 출력 노드를 동시에 계산할 수 있어 연산 효율이 크게 향상된다.

Ⅲ. RTL 시뮬레이션 검증 및 RLC 기법의 영향 분석

그림 1에 제시된 전체 아키텍처를 기반으로, RLC 인코더 모듈을 적용한 구조와 적용하지 않은 구조를 각각 Verilog HDL로 구현하고, RTL 수준의 시뮬레이션을 통해 연산량 및 메모리 사용량을 비교 분석하였다. 본 실험에서 사용된 신경망은 총 3개의 완전 연결 층으로 구성되어 있으며, 각 층의 노드 수는 순서대로 96, 64, 10이며 입력 데이터의 노드수는 784이다. 입력 데이터와 가중치는 무작위 정수값으로 초기화하였으며 RTL 시뮬레이션 결과는 C 언어로 구현한 Golden Reference 모델의 출력과 비교하여 하드웨어 정확성을 검증하였다.

표 1은 입력 데이터 내 0 값의 비율에 따라 메모리 사용량 절감률과 연산량 감소율이 어떻게 변화하는지를 보여준다. 시뮬레이션 결과, 0 값의 비율이 높아질수록 RLC를 적용한 구조에서 메모리 사용량과 연산량이 모두 유의미하게 감소하는 경향을 보였다. 이는 RLC 압축 기법이 희소성이 높은 데이터에 대해 효과적으로 작동함을 입증한다.

Zero Value Ratio (%)	Memory Usage Reduction Rate (%)	Computation Reduction Rate (%)
80	71.19	62.28
60	54.13	43.40
40	37.71	25.25
20	20.23	5.84
10	13.24	-1.91

표 1. 입력 데이터의 0 값 비율에 따른 메모리 사용량 및 연산량.

다만, 0 값의 비율이 10% 이하로 낮은 경우 전체 연산량은 오히려 증가하는 현상이 나타나는데, 이는 RLC 인코더 모듈 자체에서 발생하는 추가 연산에서 기인한다.

결론적으로, RLC의 적용으로 메모리 사용률 및 연산량 감소 효과를 얻기 위해서는 입력 데이터의 0 값 비율이 20% 이상이 되어야 함을 알 수 있다.

Ⅳ. 결론

본 연구에서는 완전 연결 층에 RLC 압축 기법을 적용한 하드웨어 구조를 설계하고 RLC 적용 여부에 따른 성능 변화를 비교하였다. RTL 시뮬레이션 결과에서 볼 수 있듯이 입력 데이터의 희소성이 높을수록 RLC 적용에 의한 메모리 사용량과 연산량이 효과적으로 감소함을 확인하였다. 결론적으로, RLC를 적용하는 제안 구조가 자원 제약이 있는 환경에서도 효율적으로 활용될 수 있음을 보였다.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT: Ministry of Science and ICT) RS-2025-00557827.

참 고 문 헌

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Communications of the Association for Computing Machinery (ACM), 84-90, 2017.

[2] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Proceeding of 3rd International Conference on Learning Representations, (ICLR), May 7-9, 2015.

[3] M. tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," Proceedings of the 36th International Conference on Machine Learning (ICML), 9-15, 2015.

[4] Y. H. Chen, T. Krishna, J. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," IEEE journal of solid-state circuits, vol. 52, no. 1, pp. 127-138, 2016.