RNN 추론 프로세서를 위한 통합 검증 시스템

홍성재, 김지호, *김태환
한국항공대학교 항공전자정보공학부
e-mail : *taehwan.kim@kau.ac.kr*

An Integrated Verification System for an RNN Inference Processor

Seong-Jae Hong, Ji-Ho Kim, Tae-Hwan Kim
School of Electronics and Information Engineering
Korea Aerospace University

Abstract

This paper shows the design and implementation of a verification system for a recurrent-neural-network (RNN) inference processor. The proposed system has been designed by adding the hardware abstraction layer of the RNN inference processor, which is provided as the application programming interface of the deep learning framework, Pytorch. This simplifies the overall verification flow by reusing the software codes used for the model designs. Under the proposed system, the functionality of the processor has been verified successfully for several different RNN inference models.

I. 서론

Recurrent neural network (RNN)은 시간의 흐름에 따라 변화하는 시계열 데이터를 처리하는데 특화된 신경망이다. 대표적으로 실생활에서 자연어 처리나 음성 인식과 같은 입력의 순서가 중요한 분야에 광범위하게 활용되고 있다 [1-2]. 학습과 추론으로 구분되어 있는 신경망의 영역에서 추론은 서비스 단과 직결되며 빠르고 지연성 없는 처리를 요구한다. 그러나 RNN의 추론 과정은 매우 많은 연산을 필요로 하기 때문에 RNN을 위한 전용 추론 프로세서가 요구된다 [2].

타겟 시스템에 딥러닝 프로세서를 이식하기 위해서는 사용자가 프로세서의 특성과 시스템의 구성 요소를 이해하여 시스템과 프로세서를 연결해 주어야 한다. 하나의 프로세서로 여러 종류의 인공 신경망을 가속화할 수 있는 통합 인공 신경망 프로세서의 연구가 활발해지면서 하드웨어 변경 없이도 다양한 신경망을 가속할 수 있게 되었다. 그러나 통합 프로세서를 사용하는 사용자는 모델의 크기, 종류, 그리고 데이터셋에 맞춰 검증 과정을 재구성해야 한다. 모델과 데이터셋마다 소프트웨어와 하드웨어의 인터페이스를 맞추어 주는 과정이 필요하기 때문에 사용자는 프로세서에 대한 이해가 선행되어야 한다 [4]. 하드웨어 또는 FPGA 경험이 적은 개발자가 직접 검증 시스템을 구현하게 되면 빠른 시간 내에 정상적인 결과를 확인하기 어렵다. 또한 이 과정에서 하드웨어는 문제 원인의 파악과 해결이 .어렵기 때문에 전문적인 지식을 필요로 한다.

본 논문에서는 이러한 사용자의 오버로드를 줄이기 위해 하드웨어 프로세서의 내부 정보 없이 프로세서를 사용할 수 있도록 하드웨어와 소프트웨어 사이에

하드웨어 추상화 계층을 제공한다. 해당 하드웨어 추상화 계층은 딥러닝 프레임워크인 Pytorch의 응용 프로그래밍 인터페이스를 기반으로 Pythorch 클래스를 사용하는 방식과 동일하게 프로세서를 사용할 수 있는 환경을 제공한다. 학습된 가중치를 사용하는 추론 프로세서의 end-to-end 검증 통합 시스템을 설계하고 다양한 모델에 대해 검증한 결과를 시연한다.

본 논문의 II장에서는 제안하는 RNN 추론 프로세서를 기반으로 한 검증 시스템에 대해 구체적으로 기술하고, Ⅲ장에서는 검증 절차에 대해 설명한다. Ⅳ장에서는 제안하는 RNN 추론 프로세서 기반의 시스템을 사용하여 다양한 RNN 모델을 검증한 결과를 보인다. 마지막으로 Ⅴ장에서는 본 연구의 결론을 기술한다.

II. RNN 추론 프로세서 검증 시스템

2.1 RNN 추론 프로세서

시스템에 사용된 RNN 추론 프로세서는 RNN의 추론 과정에서 일어나는 연산을 가속화하기 위해 여러 개의 데이터를 한 번에 처리하는 SIMD 구조의 프로세서이다 [5]. 하드웨어에서 부동 소수점 데이터 기반의 연산이 고정 소수점 연산에 비해 자원을 많이 사용하며, 더 복잡하다. 따라서 RNN 추론 프로세서는 고정 소수점 기반의 수 체계 방식으로 데이터를 입력 받고, 연산 결과를 고정 소수점 기반의 수 체계로 반환한다. 이후 프로세서의 결과를 검증 시스템에서 사용하기 위해서는 다시 부동 소수점 데이터로 변환하여 사용한다. 따라서 프로세서를 사용하기 전에 부동 소수점 기반의 수 체계 방식으로 저장된 데이터들을 고정 소수점 기반의 수 체계 방식으로 전환하는 과정이 필요하다.

2.2 인터페이스

RNN 프로세서를 사용하기 위해서는 훈련된 가중치를 주변 메모리에 저장하기 위해 데이터를 전송해야 한다. 이런 과정을 수행하기 위해서 다양한 소프트웨어/하드웨어 사이의 시스템 구성 요소가 필요하다. Python program으로부터 data를 전송하기 위해서는 AXI CDMA를 사용한다. Allocate 함수로 PS영역의 DRAM 중 사용할 크기를 정하고 해당 영역를 numpy 배열에 접근하듯 jupyter notebook상에서 DRAM에 data를 읽고 쓸 수 있다. CDMA는 MMIO(Memory Mapped I/O)방식으로 PS 영역에서 정의해준 DRAM의 address와 PL 영역의 BRAM의 address를 알면 memory에 접근하듯

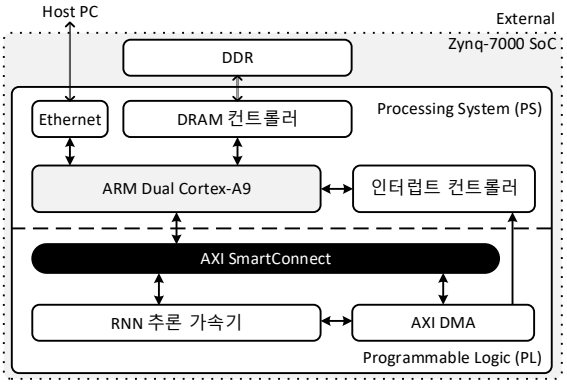

그림 1. 검증 시스템

데이터를 쉽게 전송할 수 있게 된다. register_map함수를 사용해서 Memory mapped source address와 memory-mapped destination address, buffer에 저장된 데이터의 bytes를 정의하면 데이터가 고속의 인터페이스를 통해 source address에서 destination address로 전송된다.

2.3 하드웨어 추상화 계층

RNN 프로세서를 사용하기 위해서는 프로세서의 인터페이스와 데이터의 저장 방식과 같은 프로세서에 대한 기본 지식이 필요하다. 프로세서를 동작시키기 위해 사용자가 검증 절차를 설계하는 것부터 프로세서의 정상 동작을 확인하는 과정까지 참여해야 한다.

하드웨어로 구현된 프로세서 내부에 대한 이해 없이도 프로세서를 사용하여 동작시킬 수 있도록 하드웨어의 인터페이스를 pytorch 기반의 torch 클래스와 동일하게 맞춰준 API화 한다. 단순히 모델만 수정하였을 때 하드웨어 프로세서를 사용할지, 소프트웨어로 동작시킬지 결정할 수 있게 된다. 또한 모델을 수정하였을 때 모델에 대한 파라미터 설정 이외에는 모두 동일한 시스템을 사용하여 다양한 모델에 대해 검증이 가능하여 모델마다 인터페이스를 변경할 필요가 없다.

Ⅲ. RNN 추론 프로세서 검증 절차

하드웨어 프로세서는 pytorch 기반으로 API화 했기 때문에 torch.nn 함수가 제공하는 python의 API 사용법과 동일하게 동작시킬 수 있다. RNN 모델을 정하고, 하드웨어 프로세서의 API에 input size, hidden size, batch size, layer num, step(length)를 결정한다.

검증 시스템의 순서는 그림2와 같다. 우선 추론에 필요한 파라미터를 메모리에 미리 적재해놓고 이후

	추론환경	정확도	소요시간
RNN	SW	97.20%	78.031s
	HW	97.13%	51.657s
GRU	SW	98.52%	222.217s
	HW	98.47%	52.774s
LSTM	SW	98.14%	310.828s
	HW	96.55%	55.397s

표 1. MNIST 데이터셋 검증 결과

RNN의 n번째 time-step의 입력 데이터 메모리에 적재한다. 이후 프로세서를 동작시키고, 그와 동시에 DMA를 사용하여 그 다음 time-step의 입력을 메모리에 적재한다. 따라서 프로세서의 동작과 그 다음 time-step의 입력의 적재가 동시에 이루어져 데이터 전송에 소요되는 시간을 줄일 수 있다. 이후 DMA를 사용한 데이터 전송이 끝나면, 프로세서의 결과를 다시 DMA를 사용하여 읽어온다. 이 때 DMA는 끝났다는 신호를 인터럽트 기반으로 PS에 넘겨주기 때문에 PS는 RNN 추론 프로세서의 동작이 끝났다는 인터럽트 신호를 받기 전까지 다른 작업을 수행할 수 있다.

Ⅳ. RNN 추론 프로세서 검증 결과

본 장에서는 PYNQ-Z2 ZYNQ XC7Z020 FPGA를 기반으로 한 FPGA 개발 보드를 사용한 검증 및 시연 시스템의 검증 결과에 대해 기술한다. PYNQ는 파이썬 생산성 향상 보드로 내장형 시스템에 사용되는 범용의, 프로그래밍 가능한 플랫폼이다. Jupyter Notebook을 기반으로 하는 Python productivity for Zynq 개발 환경의 출현으로 FPGA 경험이 적은 개발자도 FPGA 성능을 최대한 활용하여 컴퓨팅 집약적인 애플리케이션을 가속화할 수 있는 설계를 빠르게 구현할 수 있게 되었다. 따라서 python 프로그램을 구축할 수 있는 개발자를 위해 생산성이 높은 개발자 환경을 제공해준다.

표1은 MNIST 데이터셋에 대한 검증 결과에 대한 자료이다. RNN, GRU, LSTM 세 가지 모델에 대해서 RNN 추론 프로세서를 사용한 환경과 순수 소프트웨어상에서 연산한 정확도와 소요시간을 보여준다. RNN은 연산량이 GRU와 LSTM에 비해 크지 않아 소요 시간과 정확도의 차이가 크지 않지만, GRU와 LSTM 모델의 경우 연산량이 많기 때문에 소요시간이 소프트웨어 추론 환경보다 최대 5.6배 더 빨라진다. MNIST dataset 기반 데모의 경우, python의 matplotlib함수를 이용해서 MNIST data를 jupyter notebook에서 출력할 수 있다.

Ⅴ. 결론

본 논문은 RNN 추론 프로세서를 위한 검증 시스템의 설계 및 구현하였다. 제안하는 시스템은 딥러닝 프레임워크인 Pytorch의 응용 프로그래밍 인터페이스로 제공되는 RNN 추론 프로세서의 하드웨어 추상화 계층을 추가하여 설계되었다. RNN 프로세서를 사용할 때 내부 동작 원리를 이해할 필요가 없도록 기존의 Pytorch 딥러닝 api의 동작 방식과 동일한 인터페이스를 제공한다. 하드웨어와 소프트웨어 사이의 인터페이스를 단순화 하여 검증하는 데에 필요한 시스템적인 요소의 복잡한 확인 및 설계 과정을 건너뛸 수 있다. 이는 모델 설계에 사용된 소프트웨어 코드를 재사용하여 전체 검증 흐름을 단순화한다. 제안된 시스템에서 프로세서의 기능은 여러 다른 RNN 모델에 대해 성공적으로 검증되었다.

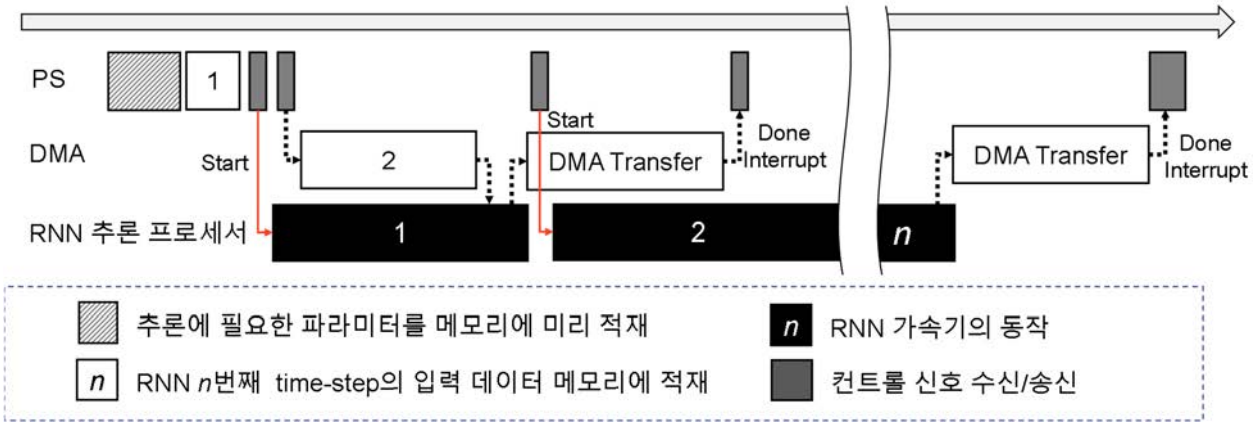

그림 2. 검증 시스템 추론 과정

Acknowledgments

본 연구는 경기도의 경기도 지역협력연구센터 사업 [GRRC항공2017-B02, 3차원 공간 데이터 처리 및 응용 기술 연구] 과 2017년도 정부 (과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행되었음 [2017-0-00528, 다중 모드 스마트 레이다용 지능형 반도체 개발 기초 연구실]. EDA 도구는 IDEC의 지원을 받았음.

참고문헌

[1] C. Gao, S. Braun, I. Kiselev, J. Anumula, T. Delbruck and S. Liu, "Real-Time Speech Recognition for IoT Purpose using a Delta Recurrent Neural Network Accelerator," *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1-5, May 2019.

[2] M. Lee, K. Hwang, J. Park, S. Choi, S. Shin and W. Sung, "FPGA-Based Low-Power Speech Recognition with Recurrent Neural Networks," *IEEE International Workshop on Signal Processing Systems (SiPS)*, pp. 230-235, Oct 2016.

[3] A. X. M. Chang and E. Culurciello, "Hardware accelerators for recurrent neural networks on FPGA," *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1-4, May 2017.

[4] Jiho Kim and Tae-Hwan Kim, "ROSETTA: A Resource and Energy-Efficient Inference Processor for Recurrent Neural Networks Based on Programmable Data Formats and Dynamic Activation Pruning," *Transactions on Emerging Topics in Computing (TETC)*, pp. 1-11, submitted.

[5] M. Vohra and S. Fasciani, "PYNQ- Torch: a framework to develop PyTorch accelerators on the PYNQ platform," *IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pp. 1-6, Dec. 2019.