뉴로모픽 환경에서 QoS를 고려한 최적의 SNN 모델 파라미터 생성 기법

(QoS-Aware Optimal SNN Model Parameter Generation Method in Neuromorphic Environment)

김서연*, 김봉재**, 정진만***

(Seoyeon Kim, Bongjae Kim, Jinman Jung)

요 약

뉴로모픽 아키텍처 기반 하드웨어를 이용한 IoT 엣지 서비스는 단말 장치에서 지능형 처리를 수행할 수 있기 때문에 자율형 IoT 응용 지원에 적합하다. 그러나 IoT 개발자가 뉴로모픽 하드웨어에서 사용되는 SNN을 이해하기에는 어려움이 있다. 본 논문에서는 뉴로모픽 하드웨어의 제약조건을 고려하며 사용자의 요구 성능을 만족하는 SNN 모델 생성 기법을 제안한다. 제안 기법은 프로파일링된 데이터에서 최적의 SNN 모델 파라미터를 찾도록 전처리된 데이터로 사전 학습한 모델을 활용한다. 전체 탐색 기법과 비교 결과, 두 기법 모두 사용자 요구사항을 모두 만족하였지만, 제안 기법이 수행 시간 측면에서 더 좋은 성능을 보였다. 또한, 신규 하드웨어의 제약조건을 명확히 알지 못하더라도 새로운 하드웨어의 프로파일링된 데이터를 활용할 수 있으므로 높은 확장성을 제공할 수 있다.

■ 중심어 : 뉴로모픽 하드웨어 ; Loihi ; 스파이킹 뉴럴 네트워크 ; 지능형 IoT

Abstract

IoT edge services utilizing neuromorphic hardware architectures are suitable for autonomous IoT applications as they perform intelligent processing on the device itself. However, spiking neural networks applied to neuromorphic hardware are difficult for IoT developers to comprehend due to their complex structures and various hyper-parameters. In this paper, we propose a method for generating spiking neural network (SNN) models that satisfy user performance requirements while considering the constraints of neuromorphic hardware. Our proposed method utilizes previously trained models from pre-processed data to find optimal SNN model parameters from profiling data. Comparing our method to a naive search method, both methods satisfy user requirements, but our proposed method shows better performance in terms of runtime. Additionally, even if the constraints of new hardware are not clearly known, the proposed method can provide high scalability by utilizing the profiled data of the hardware.

■ keywords : Neuromorphic hardware ; Loihi ; Spiking neural networks ; Intelligent IoT

Ⅰ. 서 론

최근 인공지능 기술의 발전으로 IoT와 AI 기술을 결합한 지능형 IoT 응용 분야에 대한 수요가 증가하였다. 이에 따라 IoT 디바이스에서 자율적으로 학습하고 추론이 가능한 AI 서비스를 제공하는 기술에 대한 기대가 높아지고 있다. 그러나 하드웨어와 연결된 클라우드 서버에서 제공하는 AI 서비스는 데이터 처리 지연 및 프라이버시 문제가 발생할 수 있다[1]. 따라서 하드웨어 내에서 지능형 처리를 수행하는 IoT 엣지 서비스에 대

* 정회원, 인하대학교 인간중심컴퓨팅연구소 박사후연구원
** 정회원, 충북대학교 컴퓨터공학과 부교수
*** 종신회원, 인하대학교 컴퓨터공학과 부교수
이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2021R1F1A1062884).
접수일자 : 2023년 04월 30일 게재확정일 : 2023년 05월 25일
교신저자 : 정진만 e-mail : jmjung@inha.ac.kr

한 연구가 필요하다. 하드웨어의 지능형 처리를 통해 에너지 소비를 줄이고 향상된 성능으로 다양한 AI 기술과 결합된 응용을 제공할 수 있어야 한다.

인간의 뇌 구조를 모사하여 연산을 처리하는 뉴로모픽 컴퓨팅은 AI 알고리즘과 빠른 데이터 처리를 지원할 수 있는 많은 뉴런과 시냅스로 구성된 하드웨어 아키텍처이다. 일반적으로 SNN(Spiking Neural Networks)을 활용하는 뉴로모픽 아키텍처는 일부 학습 알고리즘을 적용하였을 때 GPU 기반 DNN(Deep Neural Network)보다 정확도가 낮을 수 있다[2]. 그러나 하드웨어적인 지원을 통해 에너지 소비를 줄일 수 있어 IoT 엣지 서비스의 에너지 소비 측면에서 더 효율적이다[3]. 뉴로모픽 하드웨어를 통해 자율형 IoT를 지원하려면 생물학적 지식이 요구되는 SNN에 대한 충분한 지식이 필요하다. 그러나 IoT 개발자가 인간의 뇌 구조와 유사하게 동작하는 SNN을 효과적으로 활용하려면 이러한 시스템을 이해하고 통합적으로 적용하기 위한 추가 비용과 시간이 요구된다.

뉴로모픽 컴퓨팅을 지원하는 뉴로모픽 칩으로는 TrueNorth[4], Loihi[5], SpiNNaker[6], BrainScaleS[7], Neurogrid[8] 등이 있으나 대부분 학술 연구를 위해 개발된 프로토타입이거나 실험용 플랫폼으로써 완전히 상용화되지 않았기 때문에 사용에 제한적이다. 현재 보편적으로 사용할 수 있는 FPGA 보드로는 DE1-SoC[9], PYNQ[10] 등이 있으며 Nengo 프레임워크[11]를 통해 SNN의 경량 모델 구현이 가능하다. 뉴로모픽 칩과 FPGA 보드는 각각 최대 지원 가능한 뉴런의 수와 시냅스의 수, 그리고 지원하는 SNN 모델이 서로 다르다. 따라서 자율형 IoT 응용 프로그램의 개발은 각 뉴로모픽 아키텍처의 제약으로 인해 제한될 수 있다.

본 논문에서는 Loihi 에뮬레이터에 초점을 맞추어 사용자 요구사항 만족하는 SNN 모델 파라미터 생성 기법을 제안한다. 사용자는 성능 요구사항을 입력하여 사전에 프로파일링 된 하드웨어 데이터를 기반으로 사용자의 요구사항을 충족하도록 최적화된 SNN 실행 코드가 자동으로 생성된다. 뉴로모픽 기반 하드웨어의 동작 과정이나 SNN에 익숙하지 않더라도 필요한 모델을 자동으로 생성하도록 사용자를 지원할 수 있다.

성능 평가를 통해 전체 탐색 기법과 비교하였을 때, 사용자 요구사항의 오차는 제안 기법에서 크게 나타났지만 수행 시간 측면에서는 더 좋은 성능을 보였다. 복잡한 모델을 제공할수록 탐색 시간이 길어지기 때문에 뉴로모픽 하드웨어 특성에 따라 제안 기법이 필요할 수 있다. 현재는 Loihi 에뮬레이터를 활용하였기 때문에 정확도에 초점을 맞추었으나 추후 하드웨어가 보편화되면 수행 시간도 함께 고려할 수 있을 것으로 보인다. 또한, 신규 하드웨어의 명확한 제약조건을 알지 못하더라도 한 번 수집한 프로파일링 데이터를 활용할 수 있기 때문에 새로운 하드웨어의 높은 확장성을 제공할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 소개하고 3장에서는 사용자 요구사항을 고려한 최적의 SNN 모델 생성 기법을 설명한다. 4장에서는 전체 탐색 기법과 비교를 통해 성능을 평가하고, 5장에서 결론을 맺는다.

Ⅱ. 관련연구

1. 프로파일링 데이터 기반 최적화 연구

기존의 프로파일링 데이터 기반 연구들은 성능 향상에 중점을 두고 있다. [12]에서는 성능에 영향이 큰 블록을 찾아 최적화 하도록 가상기계 코드 프로파일링을 수행하였다. [13]에서는 메모리 접근 패턴을 분석하여 프로그램 최적화에 초점을 맞추었으며 [14]에서는 머신러닝 모델의 GPU 사용량을 분석하여 효율적인 작업 배치 및 우선순위를 결정하였다. 그 외에도 저전력[15], 메모리 사용 최소화[16], 하드웨어 최적화[17]를 목표로 하거나 유전 알고리즘이나 머신러닝 모델을 이용한 연구도 있다[18]. 제안 기법은 프로

파일링 기반의 머신러닝 모델을 사용하여 기존 연구들과 유사할 수 있으나 최고의 성능을 찾기보다는 사용자 요구사항의 QoS를 고려한 SNN 모델 생성에 초점을 맞추었다.

2. 자율형 IoT 지원 연구

Node-RED[19]는 경량 기계학습을 지원하는 플로우 기반의 프로그래밍 도구이다. 파이썬 및 Node.js 엔진과 결합된 지능형 IoT 응용을 지원하여 다양한 디바이스를 활용한 데이터 처리가 가능하다. 드래그 앤 드롭 방식의 웹 기반 편집기를 이용하여 입출력 노드를 구성하고 필수 파라미터를 설정하여 손쉬운 프로그래밍이 가능하다. 필요에 따라 노드를 새로 설치하거나 JSON 포맷으로 구성된 플로우를 공유할 수 있다. 제안 기법을 통해 생성된 SNN 수행 코드는 Node-RED에 적용하여 자율형 IoT를 지원할 수 있다.

선행 연구[20]를 통해 AI 컴포넌트 추상화 모델을 제안하였으며 다양한 하드웨어 환경에 적용되는 AI 모델 자동생성도구를 구현하였다. 추상화 계층으로 인해 직접 변환하는 것보다 지연시간이 발생할 수는 있으나 그 차이가 크지 않으므로 성능에 큰 영향을 끼치지 않으면서 IoT 응용이 가능한 AI 모델을 생성할 수 있다. 또한, [20]를 확장하여 FPGA에서 SNN을 위한 모델 최적화 기법[21]과 오픈 플랫폼과 호환되는 지능형 IoT 컴포넌트 자동생성도구[22]를 구현하였다. 이 논문에서 제안한 가상 컴포넌트 계층을 통해 IoT 하드웨어와 뉴로모픽 하드웨어에 적용할 수 있는 컴포넌트를 자동으로 생성할 수 있다. 본 논문에서는 선행 연구들에서 구현된 프레임워크를 통해 사용자 요구사항을 입력받고 뉴로모픽 하드웨어에 적합한 최적의 스파이킹 뉴럴 네트워크 모델을 생성할 수 있도록 지원한다.

Ⅲ. 최적의 SNN모델 파라미터 생성기법

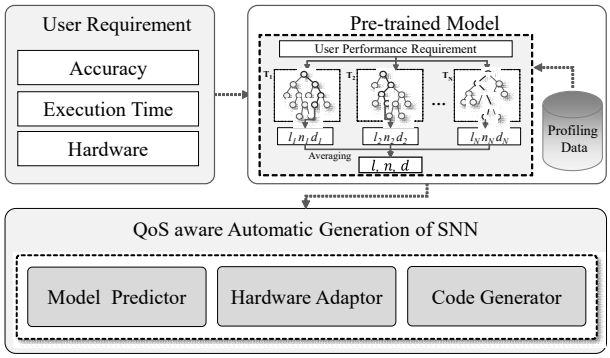

그림 1. 시스템 모델

FPGA 및 뉴로모픽 칩과 같은 뉴로모픽 하드웨어에서 동작하는 SNN은 GPU 기반의 인공신경망(ANN)에 비해 효율적인 에너지 소모량을 제공하므로 실시간 처리 작업에 적합하다. 그러나 IoT 개발자를 위한 지능형 IoT 응용을 지원하기 위해서는 하드웨어별 특성과 제약조건에 대한 이해가 필요하며, 원하는 SNN 모델이 생성되기까지 개발 시간 및 비용이 증가할 수 있다. 따라서 제안 기법은 SNN 모델을 생성할 때 프로파일링 정보를 기반으로 하드웨어 특성 및 제약조건을 자동으로 고려하도록 설계되었다.

SNN 모델을 자동으로 생성하기 위한 시스템 모델은 그림 1과 같다. 모델 예측기, 하드웨어 어댑터, 코드 생성기 세 가지로 구성되며 모델 예측기는 프로파일링 결과에서 사전 훈련된 모델을 사용하여 SNN 모델 파라미터를 예측한다. 하드웨어 어댑터는 하드웨어별 제약조건을 확인하고 최적화된 성능을 위해 매개 변수를 조정하며, 코드 생성기는 대상 하드웨어의 IoT 플랫폼에서 작동할 입력/출력 인터페이스를 생성한다.

제안 기법은 현재 접근이 쉬운 Loihi 에뮬레이터에 중점을 두었으나 다양한 하드웨어로 확장될 수 있다. 그림 2와 같이 Loihi 플랫폼에 대한 분석을 기반으로 하드웨어 제약조건이 지원하는

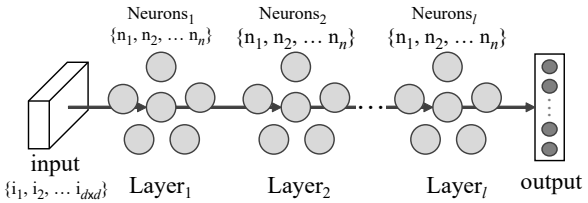

그림 2. Loihi 에뮬레이터에서의 SNN 모델

간단한 SNN 모델을 고려하였다. 현재 Loihi 하드웨어의 최대 Axons은 4,096이며 시냅스 비트의 수는 128KB이다[5]. 하지만 에뮬레이터의 경우 시냅스 비트가 1,024KB로 설정되어 있다[23]. 이처럼 하드웨어에 따라 최대/최소 성능 범위, 최대 뉴런 수, 최대 시냅스 비트 수 등 연산에 필요한 파라미터는 각각 다를 수 있다. 하드웨어 어댑터는 모델 예측기가 예측한 파라미터가 하드웨어 제약조건을 준수하면서 사용자가 지정한 성능 요구사항을 충족하는지 확인한다. 사용자가 원하는 성능을 만족하는 최적의 모델을 생성하기 위해 하드웨어 별로 레이어 수 *l*, 뉴런 수 *n*, 입력 데이터 크기 *d*를 달리하여 사용자 요구사항에 대한 프로파일링 정보를 얻는다.

모델 예측기는 사용자 요구사항을 모두 만족하면서 최상의 성능을 낼 수 있는 최적의 파라미터 *l, n, d*를 찾는 것을 목표로 한다. 이를 위해 사전에 프로파일링 데이터에서 사용자 요구사항을 만족하는 모든 후보를 찾고 SNN을 구성하는 파라미터의 크기가 가장 작은 데이터를 추출하여 학습한 모델을 이용하였다. 또한 제안 기법은 IoT 하드웨어에서 실행되는 다른 구성 요소와 호환되는 입/출력 인터페이스를 선택하여 상호작용할 수 있도록 SNN 모델 생성을 지원한다. 이는 SNN이 일반적인 인공신경망과 달리 스파이크 트레인 형태가 입력 데이터로 사용되기 때문이며, 입력 데이터를 스파이크로 변환하는 인코더와 스파이크를 데이터로 변환하는 디코더가 적용되어야 한다. 따라서 IoT 응용과 데이터 처리 및 전송을 관리하는 AI 구성 요소 간의 인코더 및 디코더 구조를 적용한 코드 자동 생성과 성능에 큰 영향을 미치지 않는 파라미터들의 자동 구성을 지원한다.

제안 기법을 통해 뉴로모픽 하드웨어의 특성 및 제약조건에 대한 자세한 정보가 없이도 사용자가 요구하는 성능을 만족하면서 최적화된 SNN 모델을 쉽게 생성할 수 있도록 한다. 특히, 오픈 IoT 플랫폼의 인터페이스를 적용하여 SNN 모델을 생성함으로써 자율형 IoT 응용 개발을 쉽게 수행할 수 있다.

표 1. 실험 환경

Parameter	Naïve Model	Proposed Model
Data	Original data	Pre-processing data
Hardware	Loihi Emulator	
User requirement accuracy(%)	[75-97]	

Ⅳ. 성능 평가

1. 실험 환경

제안 기법은 Loihi 하드웨어에 초점을 맞추어 사용자의 성능 요구사항을 만족하는 SNN 모델의 파라미터를 생성한다. 전체 탐색 기법과는 다르게 사용자의 요구사항을 만족하면서 모델의 사이즈를 최대한 줄이기 때문에 최적의 SNN 모델을 만들어 낼 수 있다. 또한, 프로파일링된 전체 데이터를 탐색하지 않기 때문에 수행 시간 측면에서 더 좋은 결과를 보일 수 있다.

Loihi 에뮬레이터에 대해 전체 탐색 기법과 제안 기법의 평균 수행 시간과 사용자의 요구사항에 따른 정확도 성능을 비교하였다. Loihi는 에뮬레이터 형태로 제공되므로 수행 환경에 따라 수행 시간이 달라질 수 있어, 사용자 정확도 요구사항만 고려하였다. 실험 파라미터는 최소 75% 이상 최대 97% 이하로 구성하였으며 사용된 데이터는 MNIST 데이터 셋이다. 손글씨 숫자를 인식하는 시각 분류 응용을 수행하는 SNN 모델 파라미터를 찾는다.

전체 탐색 기법은 레이어 수 *l*, 뉴런 수 *n*, 입력 데이터 크기 *d*를 달리하여 수집된 전체 프로파일링 데이터에서 사용자 요구사항을 만족하는 파라미터를 탐색한다. 이때 사용자 요구사항보다 좋은 성능을 보이면서 가장 경량의 SNN 모델을 만들 수 있는 파라미터를 반환한다.

제안 기법은 하드웨어에서 제약조건을 고려하며 사용자의 요구사항을 모두 만족할 수 있도록

표 2. 전체 탐색 기법을 적용한 실험 결과의 일부

No	User requirement	Generated hyper parameters by the proposed model			Actual performance	Result	
	Accuracy(U_{acc}) (%)	Layer(l)	Neuron(n)	Data size(d)	Accuracy(A_{acc}) (%)	Difference ($A_{acc}-U_{acc}$)	Execution time(ms)
1	91.5	1	120	10×10	91.506	0.006	27.898
2	95	1	231	15×15	95.02	0.02	21.941
3	84.5	1	100	5×5	89.78	5.28	29.920
4	97	2	922	10×10	97.001	0.001	10.989
5	92.5	1	164	10×10	92.509	0.009	24.933

표 3. 제안 기법을 적용한 실험 결과의 일부

No	User requirement	Generated hyper parameters by the proposed model			Actual performance	Result	
	Accuracy(U_{acc}) (%)	Layer(l)	Neuron(n)	Data size(d)	Accuracy(A_{acc}) (%)	Difference ($A_{acc}-U_{acc}$)	Execution time(ms)
1	95	1	293	15×15	95.97	0.97	7.978
2	96	1	293	15×15	95.97	−0.03	7.953
3	97	2	922	10×10	97.001	0.001	7.987
4	91	1	142	10×10	92.007	1.007	7.977
5	82.5	1	100	5×5	89.78	7.28	6.982

전처리 데이터를 적용한 Random Forest Regression 모델을 이용하여 SNN를 구성하는 최적의 레이어 수 l, 뉴런 수 n, 데이터 사이즈 d 파라미터를 찾는다.

2. 실험 평가

전체 탐색 기법과 제안 기법을 통해 사용자 요구사항을 만족하는 최적의 SNN 모델 파라미터를 예측하여 결과를 확인하였다. Loihi 실험 파라미터는 사용자 요구 정확도 75%부터 97%까지 0.5 간격으로 총 45개를 10번씩 반복하여 450개에 대한 수행 시간과 평균 오차를 확인하였다. 표 2는 전체 탐색 기법을 통한 결과 일부이며 표 3은 제안 기법을 통한 결과 일부를 보여준다.

두 기법의 실험 모델은 사용자가 요구하는 정확도를 입력받고 해당 조건을 만족하는 파라미터 레이어 수 l, 뉴런 수 n, 입력 데이터 크기 d를 반환한다. 해당 파라미터를 이용하여 프로파일링 데이터로부터 실제 수행 결과의 정확도 성능을 확인하였다. 제안 기법보다 전체 탐색 기법이 비교적 더 정확할 순 있지만, 그 차이가 크지 않으며 SNN 모델 파라미터를 빠르게 찾을 수 있다.

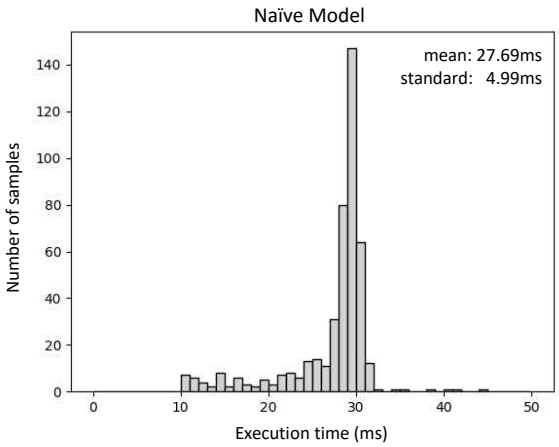

그림 3. 전체 탐색 기법의 수행 시간

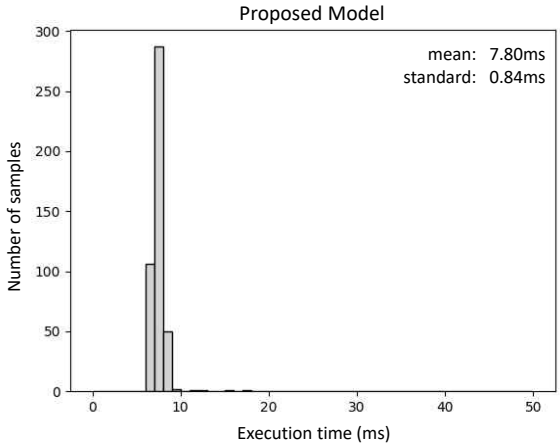

그림 4. 제안 기법의 수행 시간

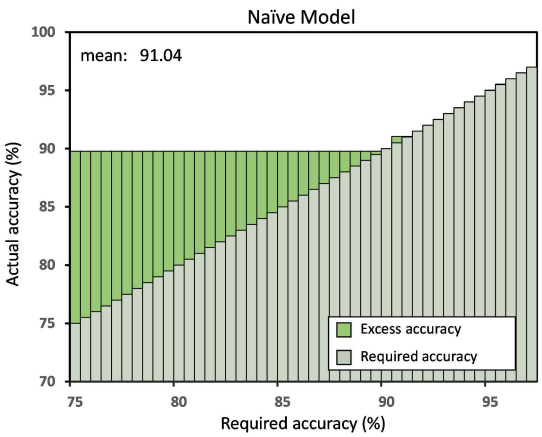

그림 5. 전체 탐색 기법의 요구사항에 따른 정확도 성능

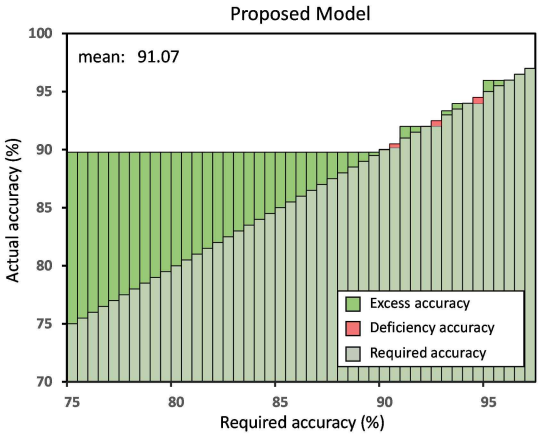

그림 6. 제안 기법의 요구사항에 따른 정확도 성능

그림 3과 4는 전체 탐색 기법과 제안 기법을 수행 시간 측면에서 분석하여 히스토그램으로 시각화한 결과를 보여준다. 450개의 데이터에 대해 전체 탐색 기법을 적용하였을 때 수행 시간은 평균 27.69ms로 측정되었으며 표준편차는 4.99를 보였다. 제안 기법을 적용하였을 때 수행 시간은 평균 7.80ms로 측정되었으며 표준편차는 0.84를 보였다. 전체 탐색 기법은 복잡한 모델일수록 탐색해야 하는 프로파일링 데이터가 많기 때문에 적합한 파라미터를 찾는데 소요되는 시간이 증가할 수 있다.

그림 5와 6은 전체 탐색 기법과 제안 기법을 사용자 요구사항에 따른 실제 정확도 측면에서 분석한 결과를 보여준다. 전체 탐색 기법은 모든 프로파일링 데이터에 대해 탐색하며 사용자 요구사항을 만족하면서 경량의 SNN 모델을 만들 수 있는 파라미터를 찾는다. 제안 기법은 하드웨어의 제약사항과 사용자의 요구사항을 고려하여 최적의 파라미터를 찾도록 전처리 된 데이터로 사전 학습된 모델을 이용하였다. 두 기법에서 찾은 파라미터를 이용하여 실제 성능을 확인하였으며 그 결과, 평균 정확도는 전체 탐색 기법에서 91.04%, 제안 기법에서 91.07%로 더 높게 나타났다. 다만, 제안 기법에서는 그림 6과 같이 사용자 요구사항을 만족하지 못하는 경우가 일부 있었으며 그 차이는 약 0.2% 정도로 미미하다.

사용자 요구사항 수준이 낮을수록 두 기법에서 모두 높은 성능을 보이는 파라미터를 찾았다. 하지만 사용자 요구사항 수준이 높아질수록 실제 정확도와의 차이가 크지 않은 것을 확인하였다. 이는 하드웨어의 제약사항에 따라 더 높은 성능의 복잡한 SNN 수행 모델을 제공하기에는 한계가 있기 때문으로 보여진다.

V. 결 론

본 논문에서는 뉴로모픽 아키텍처 기반 하드웨어의 제약조건을 고려하면서 사용자 요구사항을 만족하는 최적의 SNN 모델 파라미터 생성 기법을 제안하였다. 프로파일링된 데이터를 활용하여 하드웨어에서 최적의 결과를 보일 수 있도록 사전 학습된 모델을 사용한다. 제안 기법을 통해 IoT 개발자가 뉴로모픽 하드웨어의 내부 동작 원리를 이해하지 못하더라도 최적의 SNN 모델을 생성할 수 있도록 제공한다. 전체 탐색 기법과의 비교를 통해 제안 기법이 수행 시간 측면에서 더 좋은 성능을 보였으며 사용자 요구사항보다 향상된 결과를 보여준다. 또한, 제안 기법은 프로파일링 데이터를 활용하기 때문에 새로운 하드웨어가 등장하여도 유연하게 확장할 수 있다. 추후 뉴로모픽 하드웨어가 보편화되면 요구 수행 시간도 함께 적용하여 사용자의 정교한 요구사항을 만족할 수 있도록 연구할 예정이다.

REFERENCES

[1] 김선욱, 홍성은, 방준일, 김화종, “IoT 장치의 개인정보 데이터 보호 시스템 구현에 관한 연구," *스마트미디어저널*, 제10권, 제2호, 84-91쪽, 2021년 06월

[2] D. A. Nguyen, X. T. Tran, K. N. Dang, F. Iacopi, “A low-power, high-accuracy with fully on-chip ternary weight hardware architecture for Deep Spiking Neural Networks,” *Microprocessors and Microsystems*, vol. 90, Apr. 2022.

[3] T. N. N. Nguyen, B. Veeravalli, X. Fong, “Hardware Implementation for Spiking Neural Networks on Edge Devices," *Predictive Analytics in Cloud, Fog, and Edge Computing: Perspectives and Practices of Blockchain, IoT, and 5G*, pp.227-248, 2022.

[4] F. Akopyan, J. Sawada, A. Cassidy et al., “TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1537-1557, 2015.

[5] M. Davies, N. Srinivasa, T. H. Lin et al., “Loihi: A Neuromorphic Manycore Processor with On-Chip Learning," *IEEE Micro*, vol. 38, no. 1, pp. 82-99, Jan. 2018.

[6] A. D. Brown, J. Chad, Raihaan, B. K. M. Raihaan, K. Dugan, S. Furber, “SpiNNaker: Event-based simulation - quantitative behaviour," *IEEE Transactions on Multiscale Computing Systems,* vol. 4, pp. 450-462, Jul. 2018.

[7] J. Schemmel et al., “Live demonstration: A scaled-down version of the BrainScaleS wafer-scale neuromorphic system," *2012 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 702-702, Seoul, Korea (South), 2012.

[8] B. V. Benjamin et al., “Neurogrid: A Mixed-Analog-Digital Multichip System for Large-Scale Neural Simulations," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 699-716, May 2014.

[9] Terasic De1-SoC(2016).http://de1-soc.terasic.com/ (accessed Apr., 14, 2023).

[10] Xilinx PYNQ(2017). http://www.pynq.io/ (accessed Apr., 14, 2023).

[11] T. Bekolay et al., “Nengo: a Python tool for building large-scale functional brain models," *Frontiers in Neuroinformatics*, vol. 7, 2014.

[12] 신양훈, 이창환, 오세만, “프로파일링 데이터를 이용한 가상기계 코드 최적화,” *정보처리학회논문지 컴퓨터 및 통신시스템,* 제14권, 제3호, 167-172쪽, 2007년 6월

[13] 이윤재, 염헌영, 한혁, “메모리 접근 프로파일링을 통한 프로그램의 Swap 사용 최적화,” *정보과학회논문지,* 제47권, 제5호, 466-472쪽, 2020년 5월

[14] 오지선, 김세진, 김윤희, “CPU-GPU 컨테이너 클러스터의 프로파일링을 활용한 계산응용 실행 계획 기법,” *정보과학회논문지*, 제46권, 제10호, 975-980쪽, 2019년 10월

[15] D. Chen, N. Vachharajani, R. Hundt et al., “Taming hardware event samples for FDO compilation," *Proceedings of the 8th annual IEEE/ACM international symposium on Code generation and optimization (CGO '10),* pp. 42 - 52, New York, NY, USA, Apr. 2010.

[16] T. D. Han, T. S. Abdelrahman, “Automatic tuning of local memory use on GPGPUs,” *Part of ADAPT Workshop proceedings*, arXiv:1412.6986, 2015.

[17] E. Ipek, S. A. McKee, K. Singh, R. Caruana, B. R. de Supinski and M. Schulz, “Efficient architectural design space exploration via predictive modeling," *ACM Transactions on Architecture and Code Optimization,* vol. 4, Issue 4, pp. 1-34, Jan. 2008.

[18] E. Schulte, J. Dorn, S. Harding, S. Forrest and W. Weimer, “Post-compiler software optimization for reducing energy," *Proceedings of the 19th international conference on Architectural support for programming languages and operating systems (ASPLOS '14)*, pp. 639 - 652, Feb. 2014.

[19] M. Blackstock and R. Lea, “Toward a Distributed Data Flow Platform for the Web of Things (Distributed Node-RED)," *Proceedings of the 5th International Workshop on Web of Things (WoT '14)*. Association for Computing Machinery, New York, NY, USA, pp. 34 - 39, Oct. 2014.

[20] 김서연, 윤영선, 은성배, 차신, 정진만, “AI 컴포넌트 추상화 모델 기반 자율형 IoT 통합개발환경 구현," *한국인터넷방송통신학회 논문지*, 제21권, 제5호, 71-77쪽, 2021년 10월

[21] 김서연, 윤영선, 홍지만, 김봉재, 이건명, 정진만, “FPGA상에서 스파이킹 뉴럴 네트워크 지원을 위한 모델 최적화,” *스마트미디어저널*, 제11권, 제2호, 70-76쪽, 2022년 03월

[22] 김서연, 장준혁, 정진만, 김봉재, 윤영선, “오픈 플랫폼 호환 지능형 IoT 컴포넌트 자동 생성 도구," *스마트미디어저널*, 제11권, 제11호, 32-39쪽, 2022년 12월

[23] Applied Brain Research (2018). http://www.nengo.ai/nengo-loihi/api.html#nengo_loihi.block.LoihiBlock (accessed Jan., 12, 2023).

저 자 소 개

김서연(정회원)

2016년 건양대학교 의료IT공학과 학사 졸업.
2018년 한남대학교 무인시스템공학과 석사 졸업.
2022년 한남대학교 정보통신공학과 박사 졸업.
2022년~현재 인하대학교 인간중심컴퓨팅연구소 박사후연구원.

<주관심분야 : 지능형 IoT, 임베디드소프트웨어>

김봉재(정회원)

2021.03~: 충북대학교 컴퓨터공학과 부교수.
2016.03~2021.02: 선문대학교 컴퓨터공학부 조교수.
2015.01~2016.02: 한국전자기술연구원 임베디드·SW연구센터 선임연구원.
2014.09~2014.12: 삼성전자 SW개발팀(메모리) 책임연구원.
2014.2: 서울대학교 전기·컴퓨터공학부 박사 졸업.
2008.2: 광운대학교 컴퓨터공학부 학사 졸업.

<주관심분야 : 시스템 소프트웨어, 임베디드 및 모바일 시스템>

정진만(종신회원)

2008년 서울대학교 컴퓨터공학과 학사 졸업.
2014년 서울대학교 전기컴퓨터공학과 박사 졸업.
2014년~2021년 한남대학교 정보통신공학과 부교수.
2021년~현재 인하대학교 컴퓨터공학과 부교수.

2014년~현재 ACM SAC Software Plat-forms 트랙 의장.
2014년~현재 한국정보과학회 고신뢰컴퓨팅연구회 운영위원.

<주관심분야 : 운영체제, 임베디드 시스템, 시스템소프트웨어>