The Journal of the Convergence on Culture Technology (JCCT)
Vol. 8, No. 5, pp.613-618, September 30, 2022. pISSN 2384-0358, eISSN 2384-0366

http://dx.doi.org/10.17703/JCCT.2022.8.5.613
JCCT 2022-9-77

Spiking Neural Networks(SNN)를 위한 컴파일러 구조와 매핑 알고리즘 성능 분석

A Structure of Spiking Neural Networks(SNN) Compiler and a performance analysis of mapping algorithm

김용주*, 김태호**

Yongjoo Kim*, Taeho Kim**

요 약 SNN(Spiking Neural Networks) 기반의 인공지능 연구는 현재 유행하는 DNN(Deep Neural Networks) 기반의 인공지능의 한계를 극복할 수 있는 차세대 인공지능으로서 주목받고 있다. 본 논문에서는 SNN 형태의 입력을 뉴로모픽 컴퓨팅 시스템에서 구동시킬 수 있는 시스템 SW인 SNN 컴파일러의 구조에 대하여 설명한다. 또한 컴파일러 구현을 위하여 사용된 알고리즘을 소개하고 매핑 알고리즘의 동작 형태에 따라 뉴로모픽 컴퓨팅 시스템에서 수행시간이 어떻게 달라지는지에 대한 실험결과를 제시한다. 본문에서 제안한 매핑 알고리즘은 랜덤 매핑에 비해 최대 3.96배의 수행속도 향상이 있었다. 해당 연구 결과를 통해 SNN들을 다양한 뉴로모픽 하드웨어에서 적용할 수 있을 것이다.

주요어 : 인공지능, 두뇌모사, 신경망, 컴파일러, 매핑 알고리즘

Abstract Research on artificial intelligence based on SNN (Spiking Neural Networks) is drawing attention as a next-generation artificial intelligence that can overcome the limitations of artificial intelligence based on DNN (Deep Neural Networks) that is currently popular. In this paper, we describe the structure of the SNN compiler, a system SW that generate code from SNN description for neuromorphic computing systems. We also introduce the algorithms used for compiler implementation and present experimental results on how the execution time varies in neuromorphic computing systems depending on the the mapping algorithm. The mapping algorithm proposed in the text showed a performance improvement of up to 3.96 times over a random mapping. The results of this study will allow SNNs to be applied in various neuromorphic hardware.

Key words : AI, Neuromorphic, Neural Networks, Compiler, Mapping Algorithm

Ⅰ. 서 론

최근 들어 인공지능은 다양한 분야에서 발전을 지속하고 있다. 2000년대 초반에 시작된 DNN(Deep Neural Networks) 기술[1, 2]을 시작으로 급속도의 발전을 이루기 시작하였고 AlphaGo를 통해 바둑 분야에서 인간 대표를 이김으로써 대중에게도 인공지능의 발전을 알리게 되었다. 최근 들어서는 자율주행, 의료 인공지능,

*정회원, 한국전자통신연구원 인공지능연구소 (제1저자, 교신저자)
**정회원, 한국전자통신연구원 인공지능연구소 (참여저자)
접수일: 2022년 7월 28일, 수정완료일: 2022년 8월 18일
게재확정일: 2022년 9월 3일

Received: July 28, 2022 / Revised: August 18, 2022
Accepted: September 3, 2022
*Corresponding Author: y.kim@etri.re.kr
Artificial Intelligence Research Laboratory, Electronics and Telecommunications Research Institute (ETRI)

자율운용 로봇, 스마트팜, 무인국방, 스마트 홈 등 다양한 산업 분야에서 인공지능이 적용되며 발전이 지속되고 있다.

이렇게 인공지능이 폭발적인 발전을 이룰 수 있었던 이유로 크게 3가지가 꼽힌다. 첫 번째로 대규모의 병렬 컴퓨팅 기술의 발전이다. GPU등의 병렬 컴퓨팅 기술 및 클라우드 컴퓨팅 기술이 발전하면서 많은 인공지능 개발자들이 쉽게 대규모의 계산 자원을 확보할 수 있었고 이로 인해 인공지능 계산에 많은 계산 자원을 적용할 수 있었다. 두 번째로는 방대한 양의 데이터가 확보된 것이다. 통신기술이 발달하면서 서로 다른 장소에서 발생하는 데이터들을 쉽게 모을 수 있게 되었다. 또한 모바일 및 IoT 기기가 발달하면서 다양한 환경에서 많은 데이터를 수집할 수 있게 되었고 이로 인해 인공지능 학습에 사용할 데이터의 확보가 용이해졌다. 마지막으로는 딥러닝 방법론이 등장한 것이다. 딥러닝 기법을 통해 기존의 인공지능 이론의 한계를 극복함으로써 새로운 수준의 인공지능 개발이 가능해졌다.

하지만 이런 급속한 발전에도 불구하고 현재의 인공지능의 지능 수준은 실제 인간의 지능과 비교하였을 때 아직도 한참 걸음마 수준에 불과하다. AlphaGo가 이세돌과 바둑 대국 때 사용한 계산 자원은 그림.1에서 보이듯이 Google에서 자체 개발한 48개의 TPU가 들어간 서버급 컴퓨터를 사용하였다 [3]. (이후 발표한 Alpha Go Zero는 4개의 TPU 사용함) 바둑만 수행하도록 특화된 시스템임에도 불구하고 많은 공간을 차지하고 많은 양의 전력을 소모한다. 반면에 인간의 뇌는 1.3~1.4 kg의 무게를 가지며 단지 20W만의 전력을 소모하면서도 현재의 인공지능과는 다르게 범용적으로 다양한 일을 수행할 수 있다.

이런 인공지능과 실제 인간의 두뇌의 기술 간극을 좁히기 위해 인간의 뇌의 동작 원리를 흉내 낸 차세대 인공지능이 개발되고 있는데 이 연구를 두뇌모사 컴퓨팅(Neuromorphic computing)이라고 하며 SNN(Spiking Neural Networks)는 두뇌모사 컴퓨팅의 주요한 연구 분야 중 하나이다. SNN 방식은 두뇌가 뉴런(Neuron)과 시냅스(Synapse)로 이루어진 것과 뉴런간의 신호 전달이 스파이크(Spike) 형태의 신호 전달되는 것을 흉내내어 인공신경망을 구성하는 것을 말한다.

SNN 방식의 차세대 인공지능 연구는 대기업에서는 Intel의 Loihi[4], IBM의 TrueNorth[5], Qualcomm의 Zeroth[6] 등의 뉴로모픽 칩 형태로 개발되고 있다. 또한 유럽, 미국, 중국에서도 국가 과제 및 대학의 연구 결과물로 SpiNNaker[7], BrainScaleS[8], DynapSE[9], Neurogrid[10] 등 활발하게 연구가 진행되고 있다.

그림 1. AlphaGo 가 이세돌과 바둑 대국 때 사용한 서버
Figure 1. The AlphaGo server that used for a match against Sedol Lee

하지만 이런 SNN 기반의 인공지능 연구는 뉴로모픽 HW를 개발하는 연구 중심으로 이루어지고 있으며 이를 위한 소프트웨어적인 연구[11]는 아직 미비한 현실이다. 본 논문에서는 SNN 신경망을 뉴로모픽 HW에서 사용할 수 있는 형태로 변환하는 SNN 컴파일러 매핑 알고리즘에 대하여 소개한다.

II. 컴파일러 전체 구조

그림.2는 본 논문에서 제안하는 SNN 컴파일러의 전체 구조를 나타낸다. SNN 컴파일러는 입력으로 SNN 신경망 정보와 뉴로모픽 하드웨어 구조 정보를 받아들인다. 또한 현존하는 SNN 시뮬레이터(ex. CARLSim [12], Neuron[13], BRIAN[14] 등)를 사용하여 SNN에서 어떤 형태로 스파이크를 주고받는지의 트레이스를 추출하여 컴파일러 내부적으로 최적화를 수행할 때 사용하게 된다.

SNN 컴파일러는 내부적으로 크게 4개의 파트로 나눌 수 있다. (1)파싱모듈에서는 텍스트 형태로 입력으로 들어온 신경망과 뉴로모픽 하드웨어 정보를 파싱하여 입력으로 받아들이고, 그 내용 중 일부는 내부적으로

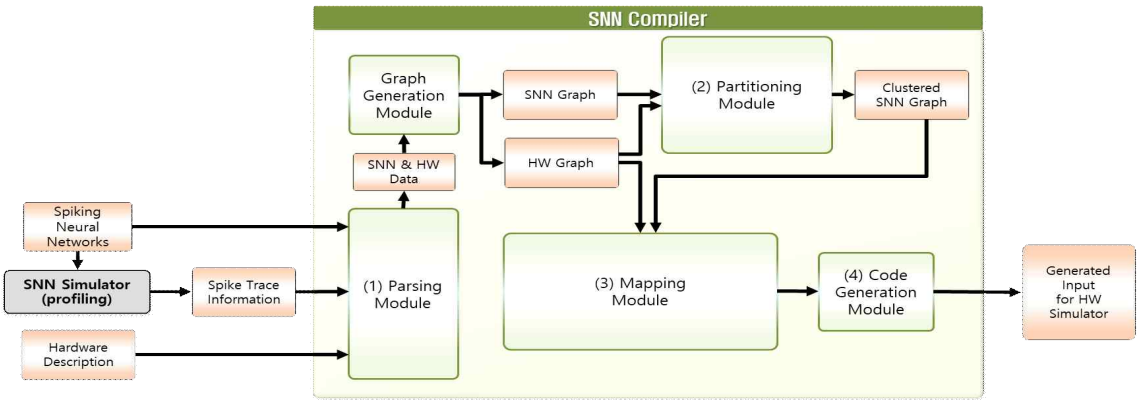

그림 2. SNN 컴파일러 전체 구조
Figure 2. The overall structure of SNN compiler

처리할 수 있는 신경망 그래프 및 하드웨어 그래프 형태로 변환하여 저장한다. (2)파티셔닝 모듈에서는 목표 뉴로모픽 컴퓨팅 환경의 구조 정보를 반영하여 신경망 그래프를 하나의 뉴로모픽 코어에서 구동 시킬 수 있는 크기의 그래프(클러스터)로 분할한다. (3)매핑 모듈에서는 신경망 그래프를 뉴로모픽 컴퓨팅 환경에서 최소한의 통신을 통해 최적 수행이 가능하도록 세부적으로 분할된 클러스터들을 각 뉴로모픽 코어에 할당하는 작업(매핑)을 진행한다. (4)코드 생성 모듈에서는 매핑된 결과를 타겟 뉴로모픽 시스템에서 동작할 수 있도록 코드를 생성하는 작업을 진행한다.

III. 파티셔너

입력으로 들어올 SNN 신경망은 응용의 분야와 종류에 따라서 그 크기가 매우 다양하다. 작게는 몇백개의 뉴런으로 구성된 신경망부터 크게는 몇백만, 몇천만개의 뉴런으로 이루어진 대규모 신경망도 존재한다. 뉴로모픽 시스템을 구성하는 뉴로모픽 코어 각각은 한번에 처리할 수 있는 뉴런의 개수가 제한되어 있어서 크기가 큰 신경망의 경우에는 신경망을 분할하여 각각의 뉴로모픽 코어에 분산하여 수행하는 작업을 하게 된다.

파티셔닝 알고리즘은 상황에 따라 여러 가지 알고리즘을 사용할 수 있으나 본 SNN 컴파일러에서는 Kernighan-Lin 알고리즘(KL 알고리즘)[15]을 사용하였다. KL 알고리즘은 그래프를 파티셔닝 하는 알려진 유명한 알고리즘으로 CAD 분야에서 자주 사용된다. 이 알고리즘은 휴리스틱 알고리즘으로 주어진 문제에 대하여 최적 결과를 찾아주지는 못하지만 NP complete 문제에 대하여 그리디(greedy)하게 지역 최적화 결과를 빠른 시간내에 찾을 수 있어 많이 사용되는 알고리즘이다.

본 SNN 컴파일러에서는 KL 알고리즘에 제약조건으로 파티셔닝 되어 나누어질 클러스터들에 최대 뉴런의 개수와 최대 시냅스의 개수를 제한을 두어, 파티셔닝 후에 생성된 클러스터가 하나의 뉴로모픽 코어에서 수행될 수 있도록 하였다. 해당 조건하에서 대형 신경망을 파티셔닝하여 여러 개의 작은 신경망 클러스터로 쪼개는 작업을 진행하였다.

IV. 매핑 알고리즘

본파티셔닝 알고리즘을 통하여 클러스터링 된 신경망은 매핑 모듈에서 뉴로모픽 컴퓨팅 시스템을 구성하는 노드들에 할당된다. 매핑 모듈은 Clustered SNN 그래프를 입력으로 받아들인다. 또한 매핑할 뉴로모픽 하드웨어 구조 그래프를 입력으로 받아들여서 해당 뉴로모픽 하드웨어 상에 클러스터를 매핑하게 된다. 입력으로 받아들이는 그래프의 구조는 표 1과 같다.

매핑 알고리즘은 입력된 SNN 그래프를 정해진 뉴로모픽 하드웨어 시스템에 최적 매핑하여, 정확성을 유지하면서도 수행속도 향상 및 에너지 소모 감소를 목표로 한다. 매핑 알고리즘은 SA(Simulated Annealing, 담금질 기법)을 기반으로 하여 구현하였다. SA는 전역 최적화 문제를 풀기 위하여 많이 사용하는 확률적 메타 알고리즘으로 탐색 공간이 방대할 경우에 많이 사용하며, 최적값을 찾기는 어렵지만, 최적 값에 근사한 결과를

정해진 시간 내에 찾아낼 수 있어 많이 사용하는 알고리즘이다.

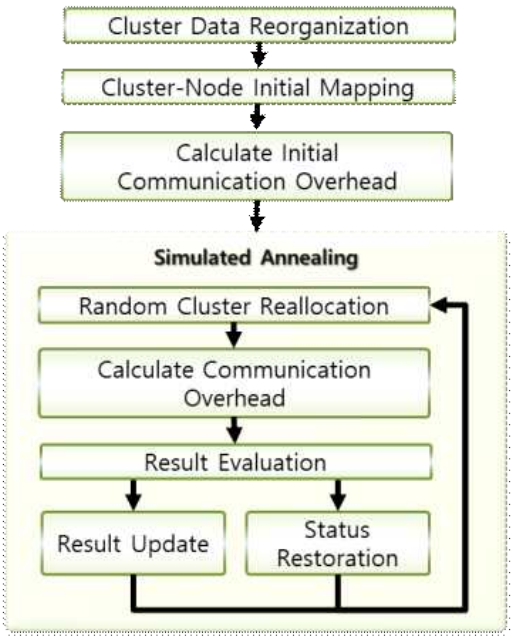

그림 3. SA 기반 매핑 알고리즘 전체 동작 흐름표
Figure 3. An flowchart of the SA based mapping algorithm

표 1. 입력 그래프 구조
Table 1. A input graph struction

입력 데이터	구조
Clustered SNN 그래프	Node : 파티셔닝 모듈에서 SNN의 뉴런들을 추후 NMU 에서 동작 가능한 사이즈로 분할하여 만든 작은 뉴런 그룹(클러스터) Edge : 각 node 내부의 뉴런에서 발생하는 스파이크 정보 중, 클러스터 외부로 나가는 스파이크 정보가 외부의 다른 node로 연결하는 edge로 표현되어 있음
하드웨어 그래프	Node : NoC 상에 존재하는 각각의 NMU를 하나의 node 형태로 구성함 Edge : NoC 상의 통신 연결상태를 edge로 표현함

그림.3은 매핑 알고리즘의 동작 구조를 나타낸다. 먼저 입력으로 받은 클러스터들에 대하여 특성 분석을 통하여 정리를 수행하고, 그 이후 클러스터들을 뉴로모픽 하드웨어 시스템이 랜덤 매핑하고 매핑 결과에서 예상되는 통신 오버헤드를 측정한다. 그 이후 본격적인 SA 단계를 수행하게 된다. SA 단계에서는 몇 가지 클러스터들을 랜덤으로 재구성하게 한다. 그리고 변동된 매핑 결과를 바탕으로 전체 통신 오버헤드를 측정한 뒤 새로운 매핑 결과에서 통신 오버헤드가 더 적을 것으로 예상되면 해당 매핑 결과를 업데이트를 한 뒤 다시 SA 단계를 반복 수행한다. 만약 통신 오버헤드가 더 나빠지는 것으로 예측되면 해당 단계의 매핑 구성을 버리고 그 전 단계의 매핑 상태를 복구한 뒤 SA를 다시 수행한다. SA 단계를 계속 반복을 하면 전체 통신 오버헤드가 줄어드는 방향으로 매핑 결과가 변화하게 된다.

V. 실 험

1. 실험환경

본 논문에서 제안한 매핑 알고리즘의 성능을 검증하기 위하여 시뮬레이션을 통하여 매핑 결과를 예상하였다. 본 SNN 컴파일러는 다양한 NMU HW 구조를 반영하여 코드 생성이 가능하다. 본 실험에서는 NMU 하드웨어는 27x27개의 node array를 가지는 NoC 기반의 NMU 컴퓨팅 시스템을 가정하였다. 노드간의 연결은 simple mesh 구조로 한 노드에 가장 근접한 4개의 다른 노드가 라우터를 통하여 연결된 환경을 가정하였다. 라우터 간의 라우팅 알고리즘은 XY 라우팅 알고리즘을 사용하는 것으로 설정하였다.

컴파일러 결과 검증을 위하여 입력으로 손글씨 인식을 위해 기존에 연구되어 공개된 신경망[12]을 사용하였다. 해당 신경망은 총 3178개의 뉴런과 2661294개의 시냅스로 이루어져 있다. 신경망에 자극을 주기 위한 이미지는 Yann LeCun 연구팀이 공개한 MNIST[16]의 데이터셋 일부를 사용하였다.

2. 실험결과

매핑 결과의 성능을 검증하기 위하여 신경망의 클러스터들을 NMU의 노드 상에 다양한 매핑 방식으로 매핑을 한 뒤 비교를 하였다. SA 기반 매핑 알고리즘을 적용할 때에는 SA 수행 횟수를 바꿔가면서 결과를 컴파일하였고 서로 다른 각 수행횟수의 결과들을 각각 비교하였다. 표 2 는 매핑 후 생성된 신경망 코드를 NoC 기반의 NMU 환경에서 수행하는 것을 시뮬레이션한 결과를 보여준다. 무작위로 파티션들을 배치한 결과는 Random Mapping 으로 표기하였고 SA를 수행한 매핑 결과는 SA Mapping 으로 표시하였다. SA Mapping 에 이어지는 괄호안의 횟수는 SA를 반복한 횟수를 나타낸다.

표 2. SA 반복 횟수에 따른 통신 오버헤드 시뮬레이션 결과
Table 2. A communication overhead simulation results based on the number of SA iterations

	Estimated Spike Comm. Time	Speedup	Compile Time
Random Mapping	22229770 cycles	-	
SA Mapping (1000번)	16429721 cycles	1.35x	212s
SA Mapping (10000번)	10707136 cycles	2.08x	1865s
SA Mapping (100000번)	5618103 cycles	3.96x	16225s

Random Mapping 한 결과와 비교를 하였을 때 SA Mapping을 적용한 경우 예상되는 스파이크 통신 시간이 줄어드는 것을 확인할 수 있었다. SA Mapping에서는 SA를 반복한 횟수를 다르게 하자 예상 스파이크 통신 시간이 줄어드는 것을 보여주었다. Random mapping와 비교하였을 때 SA 를 1000번 수행한 결과 전체 수행속도가 1.35배 상승하는 것을 확인할 수 있었다. 또한 SA를 10000번, 100000번 수행하였을 때 Speed-up은 각각 2.08배, 3.96배 상승 하였다. 이로 인해 SA mapping의 최적화가 올바르게 수행된 것을 확인할 수 있다. SA를 반복 수행하면 그 증가하는 횟수만큼 컴파일 시간도 증가하였는데 표. 2에서 SA mapping을 조건을 다르게 하여 수행하였을 때 걸리는 컴파일 타임을 확인할 수 있다. SA를 수행함으로써 얻을 수 있는 성능증가도 있으나 그에 따라 컴파일 시간도 증가하므로 적당한 목표 성능을 만족시키는 반복 횟수를 채택하는 것이 중요하다.

Ⅵ. 결 론

본 논문에서는 SNN 신경망을 입력으로 받아 뉴로모픽 컴퓨팅 시스템에서 사용할 수 있는 코드를 생성하는 SNN 컴파일러의 구조를 설명하고, 컴파일러 내부에서 사용한 알고리즘들을 설명하였다. 매핑 알고리즘에서는 자체적으로 설계한 SA 기반의 매핑 알고리즘을 제안하였고 그 효과에 대하여 실험에서 결과를 확인하였다. SA 단계를 증가시키면 뉴로모픽 컴퓨팅 시스템 내부 계산 자원간의 스파이크 통신 오버헤드를 줄일 수 있지만 그에 따른 컴파일 시간도 같이 증가하기 때문에 SA 수행 횟수를 적절한 횟수로 제한을 하는 것이 필요하다는 것을 확인할 수 있었다.

References

[1] G. E. Hinton, “Learning multiple layers of representation”. Trends in Cognitive Sciences. 11 (10): 428 - 434
[2] G. E. Hinton et al. “A Fast Learning Algorithm for Deep Belief Nets” Neural Computation. 18 (7): 1527 - 1554
[3] D. Silver et al.. “Mastering the game of Go with deep neural networks and tree search”. Nature. 529 (7587): 484 - 489. Bibcode:2016Natur.529..484S. doi:10.1038/nature16961. ISSN 0028-0836.
[3] A. Krizhevsky et al. “ImageNet Classification with Deep Convolutional Neural Networks” Communications of the ACM Vol. 60, No. 6
[4] M. Davies, et al., “Loihi: A Neuromorphic Manycore Processor with On-Chip Learning,” IEEE Micro, vol. 38, no. 1, pp. 82 - 99, 2018.
[5] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla et al. (2015). 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 34, 1537 - 1557
[6] B. F. Rubin “Qualcomm's Zeroth platform could make your smartphone much smarter”. CNET. Retrieved March 10, 2015.
[7] F. Steve, B. Petrut, SpiNNaker : A spiking neural network architecture. Boston-Delpht : Now Publishers Inc, 2020. p.350 Computational Neuroscience, Aug. 2015.
[8] C. Pehle et al. The BrainScaleS-2 Accelerated Neuromorphic System With Hybrid Plasticity. Front Neurosci. 2022 Feb 24;16:795876. doi: 10.3389/fnins.2022.795876. PMID: 35281488; PMCID: PMC8907969.

[9] A. Balaji, Y. Wu, A. Das, F. Catthoor, and S. Schaafsma, “Exploration of segmented bus as scalable global interconnect for neuromorphic computing,” in Great Lakes Symposium on VLSI. ACM, 2019.

[10]K. Boahen “Neurogrid: Emulating a Million Neurons in the Cortex,” 2006 International Conference of the IEEE Engineering in Medicine and Biology Society, 2006, pp. 6702-6702, doi: 10.1109/IEMBS.2006.260925.

[11]Y. Kim et al. “An analysis of learning performance changes in spiking neural networks(SNN)”, The Journal of the Convergence on Culture Technology (JCCT) Vol. 6, No. 3, pp.465-470, August 31, 2020. pISSN 2384-0358, eISSN 2384-0366

[12]T.S. Chou, H.J. Kashyap, J. Xing, S. Listopad, E. Rounds, M. Beyeler, N. Dutt, and J.L. Krichmar “CARLsim 4: An Open Source Library for Large Scale, Biologically Detailed Spiking Neural Network Simulation using Heterogeneous Clusters” International Joint Conference on Neural Networks 2018

[13]N.T. Carnevale and M.L. Hines The NEURON Book. Cambridge, UK: Cambridge University Press, 2006.

[14]D. F. M. Goodman and R. Brette, “The Brian simulator,” Frontiers in Computational Neuroscience, Sep. 2009.

[15]C. H. Papadimitriou “The complexity of the Lin-Kernighan heuristic for the travelling salesman problem”. SIAM Journal on Computing. 21 (3): 450 – 465. doi:10.1137/0221030.

[16]Y. LeCun, L. Bottou, Y. Bengio, and P., Haffner, “Gradient-based learning applied to document recognition.” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278 – 2324, Nov. 1998.

※ 이 논문은 2022년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임.
(No.2018-0-00769, 인공지능 시스템을 위한 뉴로모픽 컴퓨팅 SW 플랫폼 기술 개발)