The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 21, No. 5, pp.71-77, Oct. 31, 2021. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2021.21.5.71
JIIBC 2021-5-10

AI 컴포넌트 추상화 모델 기반 자율형 IoT 통합개발환경 구현

Implementation of Autonomous IoT Integrated Development Environment based on AI Component Abstract Model

김서연*, 윤영선**, 은성배**, 차신**, 정진만***

Seoyeon Kim*, Young-Sun Yun**, Seong-Bae Eun**, Sin-Cha**, Jinman Jung***

요 약 최근 이질적인 하드웨어 특성을 고려한 IoT 응용 지원 프레임워크의 효율적인 프로그램 개발이 요구되고 있다. 또한, 인간의 뇌를 모사하여 스스로 학습 및 자율적 컴퓨팅이 가능한 뉴로모픽 아키텍처의 발전으로 하드웨어 지원의 범위가 넓어지고 있다. 하지만 기존 대부분의 IoT 통합개발환경에서는 AI(Artificial Intelligence) 기능을 지원하거나 뉴로모픽 아키텍처와 같은 다양한 하드웨어와 결합된 서비스 지원이 어렵다. 본 논문에서는 2세대 인공 신경망 및 3세대 스파이킹 신경망 모델을 모두 지원하는 AI 컴포넌트 추상화 모델을 설계하고 제안 모델 기반의 자율형 IoT 통합개발환경을 구현하였다. IoT 개발자는 AI 및 스파이킹 신경망에 대한 지식이 없어도 제안 기법을 통해 자동으로 AI 컴포넌트를 생성할 수 있으며 런타임에 따라 코드 변환이 유연하여 개발 생산성이 높다. 제안 기법의 실험을 진행하여 가상 컴포넌트 계층으로 인한 변환 지연시간이 발생할 수 있으나 차이가 크지 않음을 확인하였다.

Abstract Recently, there is a demand for efficient program development of an IoT application support frameworks considering heterogeneous hardware characteristics. In addition, the scope of hardware support is expanding with the development of neuromorphic architecture that mimics the human brain to learn on their own and enables autonomous computing. However, most existing IoT IDE(Integrated Development Environment), it is difficult to support AI(Artificial Intelligence) or to support services combined with various hardware such as neuromorphic architectures. In this paper, we design an AI component abstract model that supports the second-generation ANN(Artificial Neural Network) and the third-generation SNN(Spiking Neural Network), and implemented an autonomous IoT IDE based on the proposed model. IoT developers can automatically create AI components through the proposed technique without knowledge of AI and SNN. The proposed technique is flexible in code conversion according to runtime, so development productivity is high. Through experimentation of the proposed method, it was confirmed that the conversion delay time due to the VCL(Virtual Component Layer) may occur, but the difference is not significant.

Key Words : AI Component, Autonomous IoT IDE, IoT Application, Neuromorphic Hardware, Spiking Neural Networks

*준회원, 한남대학교 정보통신공학과
**정회원, 한남대학교 정보통신공학과
***정회원, 인하대학교 컴퓨터공학과(교신저자)
접수일자 2021년 7월 16일, 수정완료 2021년 9월 16일
게재확정일자 2021년 10월 8일

Received: 16 July, 2021 / Revised: 16 September, 2021 / Accepted: 8 October, 2021
*Corresponding Author: jmjung@inha.ac.kr
Dept. of Computer Engineering, Inha University, Korea

Ⅰ. 서 론

최근 IoT 디바이스의 사이즈가 경량화되고 성능은 향상된 효율적인 하드웨어 개발이 진행되고 있다. 이에 따른 IoT 디바이스의 요구사항이 증가하며 IoT 응용뿐만 아니라 AI 프로그램 최적화를 위한 다양한 하드웨어 지원이 연구되고 있다. IoT 프레임워크는 새로운 아키텍처와 칩의 개발에 맞추어 이질적인 하드웨어 지원과 호환성도 고려해야 한다.

이처럼 하드웨어 발전에 따라 응용 프로그래밍 범위가 확장되어 다양하고 복잡한 기능이 요구된다. 또한, 응용 서비스 및 역할의 다양성으로 소프트웨어 아키텍처 측면에서 SOA(Service Oriented Architecture) 기법이나 플로우 기반의 다중 패러다임 및 다중 프로그래밍 언어로 진화하고 있다[1].

최근 들어 다양한 빅데이터 처리나 AI 기능이 추가된 연구가 진행되고 있어 미래에는 AI 지원 서비스가 필수적이다. 특히 IoT 디바이스나 센서들의 지능화에 따라 클라우드 컴퓨팅에 대한 연구가 증가하고 있으나 다량의 데이터로 인해 많은 트래픽량 요구와 실행시간 지연의 문제점이 발생할 수 있다[2][3]. 따라서 IoT 단말에서 인공지능의 추론 결과를 얻어오는 지능적 판단이 가능하도록 자율형 IoT 프로그래밍이 지원되어야 한다[4].

뉴로모픽 아키텍처는 인간의 뇌를 모사하여 스스로 학습 및 자율적 컴퓨팅이 가능한 새로운 방식의 하드웨어이다. 기존 컴퓨팅 방식에 비해 정확도는 떨어지지만 데이터 처리 속도가 빠른 편이며 에너지 소모량이 적어 효율적이다. 하지만 스파이크 신호 형태로 입력 및 출력이 표현되는 3세대 신경망인 SNN(Spiking Neural Networks)은 2세대 신경망인 ANN(Artificial Neural Networks)에 사용된 알고리즘을 수정하거나 새로운 알고리즘을 이용해야 하는 어려움이 있다.

본 논문에서는 2세대 인공 신경망 및 3세대 스파이킹 신경망 모델을 모두 지원하는 AI 컴포넌트 추상화 모델을 설계하고 제안 모델 기반의 IoT 응용이 결합된 서비스를 지원한다. IoT 개발자에게 AI 및 스파이킹 신경망에 대한 지식을 크게 요구하지 않고 자동으로 AI 컴포넌트 생성과 런타임에 대한 변환을 유연하게 제공한다. AI 컴포넌트 추상화 모델의 가상 컴포넌트 계층이 추가되어 변환 지연시간이 발생할 수 있으나 실험을 통해 그 차이가 크지 않음을 확인하였다.

본 논문의 구성은 다음과 같다. 2장에서는 IoT 및 AI 응용 통합개발환경에 관한 연구를 소개하고. 3장에서는 가상 컴포넌트 계층이 포함된 AI 컴포넌트 추상화 모델을 제안한다. 4장에서는 제안 모델의 성능평가를 위해 실험 환경을 소개하고 결과를 분석하며 5장에서 결론을 맺는다.

Ⅱ. 관련연구

Kura[5]는 M2M 서비스 게이트웨이를 위한 OSGi (Open Service Gateway initiative) 기반 응용 프로그램 프레임워크로 사물인터넷 게이트웨이가 요구하는 데이터, 클라우드, 입출력 서비스를 제공한다. IoT 게이트웨이로 데이터를 수집하고 직렬 포트, GPS, 워치 독, GPIO, I2C 등의 하드웨어 인터페이스에 액세스하기 위한 웹 기반 시각적 데이터 흐름 프로그래밍을 제공한다. Kura는 플로우 기반의 JVM (Java Virtual Machine) IDE 위에서 실행되며 소프트웨어 빌딩 블록 작성 프로세스를 단순화 할 수 있으나 IoT 기능만 제공한다.

Flogo[6]는 Go 언어 기반 프레임워크이며 액티비티들을 포함하는 플로우로 프로그래밍이 가능하다. 이벤트 드리븐 형식의 프레임워크로 클라우드와 IoT를 위한 응용 개발에 사용될 수 있으며 트리거와 액션은 들어오는 이벤트를 처리하는데 사용된다. 백그라운드 없이 제로코딩 환경에서 손쉬운 개발이 가능하나 IoT 기능만 사용할 수 있다.

Node-RED[7]는 와이어로 연결된 노드들로 이루어진 플로우 기반의 프로그래밍 도구이다. 파이썬 및 Node.js 엔진이 결합된 프레임워크로 Node.js가 제공하는 라이브러리를 사용할 수 있다. 다양한 디바이스와 프로토콜들이 입출력 노드로 구성될 수 있고 각 노드들의 필요한 인자들을 설정하여 새로운 서비스나 데이터 처리 흐름등을 손쉽게 구현할 수 있다. 웹 기반의 편집기를 제공하여 개발자들이 노드들이 구성되어있는 팔레트에서 드래그 앤 드롭 방식으로 플로우를 만들어 프로그래밍할 수 있고 이러한 플로우는 새롭게 설치된 노드들을 포함하여 JSON을 통해 쉽게 공유할 수 있으며 IoT 기능과 ANN(Artificial Neural Networks)을 사용할 수 있다.

Neuroph[8]는 뉴럴 네트워크 개발을 위한 자바 기반 경량 프레임워크로 Neuroph Studio라는 GUI 기반 뉴럴 네트워크 에디터를 제공한다. GUI 에디터를 통해 드래그 앤 드롭으로 레이어와 뉴런을 설정하여 일반적인 뉴럴 네트워크 아키텍처 실험을 지원한다. 핵심 클래스부터 기본 신경망 개념까지 포함하는 오픈소스 자바 라

이브러리로 구성되어 신경망을 생성, 훈련 및 저장하는데 사용되며, 신경망 구조를 잘 모르는 개발자들에게 복잡한 이론 및 구현을 하지 않고도 신경망 모델을 만들어 작동 가능하도록 지원한다. 하지만 일반적인 인공신경망 지원은 가능하나, 차세대 신경망인 스파이킹 신경망과 IoT 기능은 지원하지 않는다.

Nengo[9]는 소프트웨어 개발 패키지로 뉴로모픽 아키텍처 기반의 파이썬 언어를 사용하며 복잡한 구조의 스파이킹 및 비스파이킹 신경망 시뮬레이션이 가능하다. 확장성과 유연성 있게 설계되어 다양한 뉴런 타입과 학습 규칙을 추가할 수 있으며 뉴로모픽 하드웨어로부터 바로 입력을 연결할 수 있어 다양한 응용에 사용할 수 있다. 차세대 신경망인 SNN(Spiking Neural Networks)을 사용할 수 있지만 IoT 기능은 지원하지 않는다.

BindsNET[10]은 PyTorch Tensor 기능을 사용하여 스파이킹 신경망을 시뮬레이션하는 Python 패키지이다. 기계학습을 위한 생물학적 알고리즘 개발을 지원하는 스파이킹 신경망 시뮬레이션 라이브러리가 포함되어 있다. 개발자들은 BindsNET을 통해 스파이킹 신경망 시뮬레이션 프레임워크를 쉽고 효율적으로 사용할 수 있다. 하지만 IoT 응용 개발이 불가능하며 스파이킹 신경망의 복잡한 구조와 이론적 지식을 필요로 한다.

Ⅲ. AI 컴포넌트 추상화 모델

1. 제안 모델

제안 기법에서는 그림 1과 같이 그래픽 사용자 인터페이스에서 입력된 파라미터들을 가상 컴포넌트 계층에서 네이티브 코드로 변환하여 시뮬레이션 또는 FPGA 보드에 적용될 수 있는 AI 컴포넌트를 생성할 수 있다. IoT 개발자에게 인공지능의 복잡한 학습 기술을 선행하지 않아도 주요 파라미터의 입력만으로 2세대 신경망 ANN 및 3세대 신경망 SNN 기반 응용 컴포넌트를 개발할 수 있도록 한다. GUI를 통해 생성할 컴포넌트의 런타임 및 모델 파라미터를 입력받고 가상 컴포넌트 계층을 거쳐 파라미터 전달과 컴포넌트 수행코드를 생성할 수 있다. 생성된 코드는 컴포넌트 형태의 패키지로 저장되며 런타임 환경에 맞는 컴포넌트로 구분될 수 있고 뉴로모픽 아키텍처 하드웨어에 적용될 수 있다.

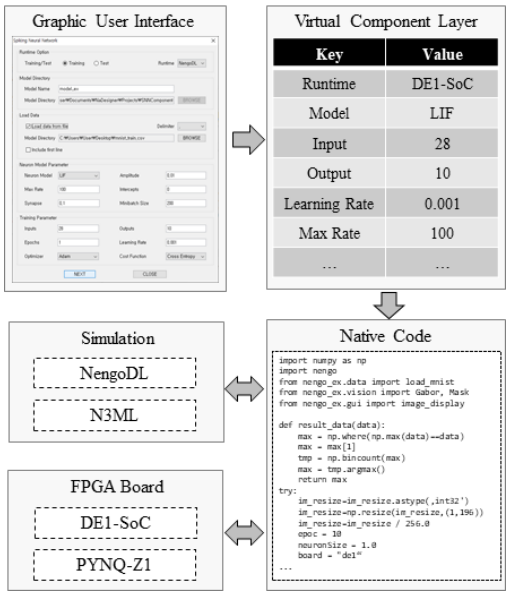

그림 1. 컴포넌트 생성 개요
Fig. 1. The structure of generating component.

2. 가상 컴포넌트 계층

AI 컴포넌트는 IoT와 AI가 결합된 형태로 ANN, SNN 및 IoT 컴포넌트를 지원한다. 가상 컴포넌트 계층 VCL(Virtual Component Layer)은 그림 2와 같이 AI 컴포넌트 제너레이터인 AI-CG(AI Component Generator)와 AI컴포넌트 모델인 AI-CM(AI Component Model)으로 나뉜다. AI-CG는 AI를 지원하는 추상화 모델에서 이질적인 AI 서비스를 일관된 오퍼레이션으로 제공한다. IoT 개발자로부터 GUI(Graphic User Interface)에 입력받은 파라미터들을 AI-CG를 통해 AI-CM(AI Component Model)에 저장하고 물리적인 뉴로모픽 디바이스 또는 ANN 및 SNN 모델이 수행가능한 AI 컴포넌트로 변환하여준다.

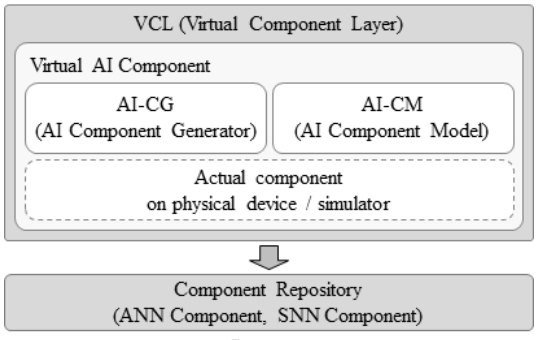

그림 2. 가상 컴포넌트 계층 구조
Fig. 2. The structure of VCL(Virtual Component Layer).

3. 구현 이슈

자율형 IoT 응용 컴포넌트의 가상 컴포넌트 모델은 그림 3과 같이 구성된다. VCModel 클래스에서 컴포넌트 특성과 상관없이 중복되는 I/O 파라미터를 저장하고 ANN 및 SNN에 공통으로 적용할 수 있는 파라미터는 AIVCModel에 저장한다. AI의 ANN, SNN 컴포넌트 각 특성에 따라 ANNVCModel과 SNNVCModel 클래스가 상속받아 사용된다. SNN의 시뮬레이션[11]을 위한 N3MLVCModel, 뉴로모픽 하드웨어 지원[12]을 위한 FPGAVCModel, 스파이킹 신경망의 ONNX 변환[13]을 위한 ONNXVCModel은 SNNVCModel을 상속받고 추가적으로 요구되는 파라미터를 저장할 수 있다. 네이티브 코드 생성을 위한 VCGenerator 클래스는 그림 4와 같이 구성된다. AI 컴포넌트 특성에 따라 공통으로 적용할 수 있는 메소드는 AIVCGenerator 클래스에서 동작한다. ANN 컴포넌트로 동작하는 VCToTensrflow 클래스는 AIVCGenerator 클래스를 상속받아 사용된다. SNN을 수행하기 위해 각 런타임에 맞는 VCToNengoDL, VCToFPGA, VCToN3ML, VCToONNX 클래스는 유형에 맞는 소스코드를 생성하고 AIVCGenerator 클래스를 상속받아 AI 컴포넌트에 해당 수행코드를 입력하여 생성한다.

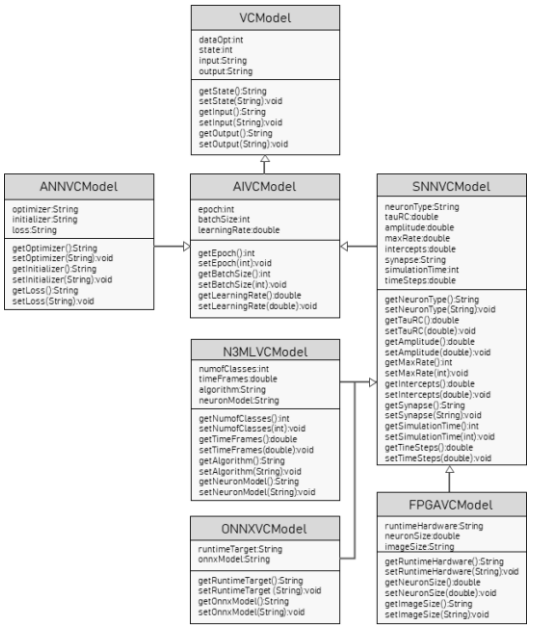

그림 3. 가상 컴포넌트 모델 클래스 다이어그램
Fig. 3. The class diagram of VCM(Virtual Component Model)

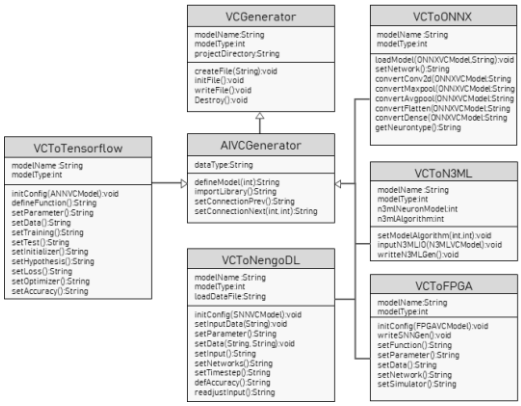

그림 4. 컴포넌트 제너레이터 클래스 다이어그램
Fig. 4. The class diagram of VCI(Virtual Component Generator)

Ⅳ. 실험 및 결과

1. 실험 환경

제안 기법에서 AI 컴포넌트 추상화 모델은 다양한 하드웨어에 유연하게 적용할 수 있다. 하지만 가상 컴포넌트 계층 VCL로 인해 약간의 변환시간이 지연될 수 있다. 제안하는 AI 컴포넌트 추상화 모델에서 VCL을 통해 런타임에 맞는 소스코드로 변환되는 지연시간을 실험하였다. 제안 기법으로 적용하였을 때 발생하는 지연시간은 VCL 없이 네이티브 코드를 직접변환하는 시간과 VCL을 거쳐 수행코드가 변환되는 시간의 비교를 통해 성능을 측정하였다.

NengoFPGA 보드와 NengoDL의 평균 변환시간을 측정하고 NengoDL은 레이어 수에 따른 변환시간을 비교하였다. 비교 파라미터는 컨볼루션 신경망을 SNN으로 구현하였을 때의 레이어 수로 선정하였다.

2. 실험 평가

다양한 하드웨어에 적용할 수 있는 유연성을 위한 VCL로 인해 발생할 수 있는 지연시간은 실험을 통해 매우 작은 것을 확인하였다. 제안 모델의 성능평가 실험을 위해 뉴로모픽 하드웨어인 FPGA보드인 Intel DE1-SoC[14]와 ABR NengoDL[15] 수행코드를 변환하여 비교하였다. 각각 뉴로모픽 하드웨어 및 SNN 수행코드 변환시간을 100번씩 총 10회 반복하여 측정하였으며 평균을 계산하여 비교 분석 하였다.

실험 결과 그림 5와 같이 AI 컴포넌트 추상화 모델에서 VCL를 이용하였을 때 직접변환 방법보다 추가지연시간이 발생함을 확인하였다. NengoFPGA보드는 VCL을 이용 하였을 때 발생하는 변환시간은 0.041ms, 직접변환 시간은 0.039ms로 0.002ms가 증가하였으며 NengoDL은 VCL 변환시간 0.058ms, 직접변환시간 0.050ms로 0.008ms가 증가하였다. 하지만 이는 매우 작은 오버헤드이며 VCL을 통한 변환시간이 직접변환 방법보다 더 작게 나타나는 경우도 있었다.

그림 6은 컨볼루션 신경망 수행 코드를 SNN인 NengoDL로 변환하여 레이어 수에 따른 변환시간을 비교한 결과이다. 레이어 수가 증가할수록 변환시간이 길게 나타났으며, 레이어의 수가 1개일 때 0.004ms, 2개일 때 0.012ms, 3개일 때 0.017ms, 4개일 때 0.018ms 정도 지연되었다.

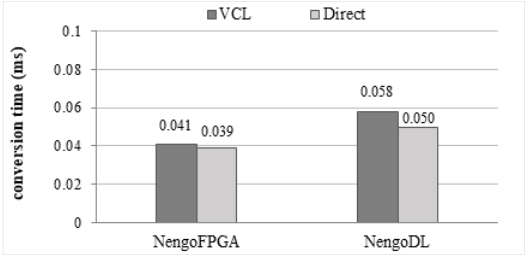

그림 5. NengoFPGA 및 NengoDL 변환시간 비교
Fig. 5. The comparison of NengoFPGA and NengoDL conversion times.

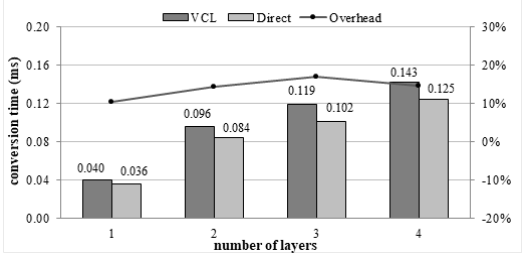

그림 6. NengoDL의 레이어 수에 따른 변환시간 비교
Fig. 6. The comparison of conversion time according to the number of layers in NengoDL.

V. 결 론

본 논문에서는 2세대 인공 신경망 ANN 및 3세대 스파이킹 신경망 SNN 모델을 모두 지원하는 AI 컴포넌트 추상화 모델을 설계하고 구현하였다. 제안 모델을 통해 IoT 개발자에게 AI 및 스파이킹 신경망에 대한 지식을 크게 요구하지 않아도 자동으로 AI 컴포넌트 생성과 런타임에 대한 변환을 유연하게 제공할 수 있다. 특히, IoT 응용과 AI가 결합된 응용 서비스를 쉽고 빠르게 지원할 수 있기 때문에 IoT 개발자 측면에서 편리하고 효율적이다.

제안 모델은 AI 컴포넌트 추상화 모델의 가상 컴포넌트 계층이 추가되어 변환 지연시간이 발생할 수 있다. 성능 평가를 위해 가상 컴포넌트 계층 VCL을 통해 런타임에 맞는 소스코드로 변환되는 시간과 VCL 없이 네이티브 코드를 직접변환하는 시간의 비교를 진행하였다. 실험 결과 NengoFPGA보드와 NengoDL의 VCL 및 직접변환 시간 차이가 약 0.002ms, 0.008ms 정도로 매우 크지 않음을 확인하였다. 또한, NengoDL의 레이어 수에 따른 변환시간이 레이어의 수가 1개일 때 0.004ms 정도 지연되었으며 레이어의 수가 4개일 때는 0.018ms 정도로 가장 큰 차이를 보였지만 성능에는 크게 영향을 끼치지 않는 것을 확인하였다.

References

[1] Seoyeon Kim, Young-Sun Yun, Jinman Jung, “A Survey of Programming Paradigm for IoT application developer”, Proceeding on 2021 Spring conference of Korean Institute of Next Generation Computing, pp.315-316, May 2021.

[2] Choon-Sik Park, "Study on Security Considerations in the Cloud Computing", Journal of the Korea Academia-Industrial cooperation Society(JKAIS), Vol. 12, No. 3 pp. 1408-1416, 2011.
DOI:https://doi.org/10.5762/KAIS.2011.12.3.1408

[3] Seung Je Park, Heeyoul Kim, "Improving Trusted Cloud Computing Platform with Hybrid Security Protocols", The Journal of Korean Institute of Information Technology(KIIT), Vol. 13, No. 5, pp. 65-72, May, 2015.
DOI: https://doi.org/10.14801/jkiit.2015.13.5.65

[4] Sanglok Yoo, Keonmyung Lee, Young-Sun Yun, Jiman Hong, “An Autonomous IoT Programming Paradigm Supporting Neuromorphic Models and Machine Learning Models”, Journal of The Korean Institute of Information Scientists and Engineers, Vol. 47, No. 3, pp. 310-318, Mar 2020.
DOI: https://doi.org/10.5626/JOK.2020.47.3.310

[5] Eclipse Kura Documentation, Available Online: http://eclipse.github.io/kura/ (accessed on 10 July 2021)

[6] Flogo, Available Online: https://www.flogo.io/ (accessed on 10 July 2021)

[7] Michael Blackstock, Rodger Lea, “Toward a distributed data flow platform for the web of things (distributed node-red)”, In Proceedings of the 5th International Workshop on Web of Things, pp. 34-39, Oct 2014.
DOI: https://doi.org/10.1145/2684432.2684439

[8] Neuroph & Neuroph Studio, Available Online: http://neuroph.sourceforge.net/ (accessed on 10 July 2021)

[9] Terrence C. Stewart, Bryan Tripp, Chris Eliasmith. “Python scripting in the Nengo simulator”, Frontiers in Neuroinformatics, Vol. 3, pp. 1-9, Mar 2009.
DOI: https://doi.org/10.3389/neuro.11.007.2009

[10] Hananel Hazan, Daniel J. Saunders, Hassaan Khan, Devdhar Patel, Darpan T. Sanghavi, Hava T. Siegelmann and Robert Kozma. “Bindsnet: A machine learning-oriented spiking neural networks library in python”, Frontiers in neuroinformatics, Vol. 12, pp. 89, Dec 2018.
DOI: https://doi.org/10.3389/fninf.2018.00089

[11] Chan Sik Han, Keon Myung Lee. “Spiking Neural Network Transformer for Deploying into a Deep Learning Framework”. In Proceedings of the International Conference on Research in Adaptive and Convergent Systems (RACS '20). Association for Computing Machinery, New York, NY, USA, pp. 82–83, 2020.
DOI:https://doi.org/10.1145/3400286.3418272

[12] Kicheol Park, Bongjae Kim. "Dynamic neuromorphic architecture selection scheme for intelligent Internet of Things services." Concurrency and Computation: Practice and Experience (2021): e6357.
DOI: https://doi.org/10.1002/cpe.6357

[13] Sangmin Park, Junyoung Heo, "Conversion Tools of Spiking Deep Neural Network based on ONNX", The Journal of The Institute of Internet, Broadcasting and Communication (IIBC), Vol. 20, No. 2, pp.165-170, Apr 2020.
DOI: https://doi.org/10.7236/JIIBC.2020.20.2.165

[14] NengoFPGA, Available Online: https://www.nengo.ai/nengo-fpga/ (accessed on 14 July 2021)

[15] NengoDL, Available Online: https://www.nengo.ai/nengo-dl/ (accessed on 14 July 2021)

저 자 소 개

김 서 연(준회원)

- 2016년 : 건양대학교 의료IT공학과 졸업 (학사)
- 2018년 : 한남대학교 무인시스템공학과 졸업(석사)
- 2018년 ~ 현재 : 한남대학교 정보통신공학과 박사과정
- 주관심 분야 : 임베디드 소프트웨어, 지능형 IoT

윤 영 선(정회원)

- 2001년 : KAIST 전산학전공(박사)
- 2001년 ~ 현재 : 한남대학교 정보통신공학과 교수
- 2006년 : 한국전자통신연구원 초빙연구원
- 2012년 : University of Washington 방문학자
- 2004년 ~ 현재 : Interspeech Scientific Reviewer
- 2017년 ~ 현재 : ICASSP Scientific Reviewe
- 주관심 분야 : 인공지능, 음성인식, 음성처리, 웹 접근성, 내장형 시스템 등

은 성 배(정회원)

- 1985년 : 서울대학교 컴퓨터공학과 졸업(학사)
- 1987년 : KAIST 전산학과 졸업(석사)
- 1987년 ~ 1990년 : 한국전자통신연구원 TDX개발단 연구원
- 1995년 : KAIST 전산학과 졸업(박사)
- 1995년 ~ 현재 : 한남대학교 정보통신공학과 교수
- 주관심 분야 : 실시간 시스템, 임베디드 시스템, USN, IoT 등

차 신(정회원)

- 1995년 : KAIST 전산학과 졸업 (박사)
- 1986년 ~ 2000년 : LG전자기술원 책임연구원
- 2000년 ~ 2013년 : ㈜IA멀티미디어 통신사업부 사업본부장
- 2013년 ~ 2015년 : ㈜슈어소프트테크 고신뢰검증센터 센터장
- 2016년 ~ 현재 : 한남대학교 정보통신공학과 교수
- 주관심 분야 : 소프트웨어 신뢰성, 안전성공학, IoT 보안 등

정 진 만(정회원)

- 2008년 : 서울대학교 컴퓨터공학과 졸업(학사)
- 2014년 : 서울대학교 전기컴퓨터공학과 졸업(박사)
- 2014년 ~ 2021년 : 한남대학교 정보통신공학과 부교수
- 2021년 ~ 현재 : 인하대학교 컴퓨터공학과 부교수
- 주관심 분야 : 운영체제, 임베디드 시스템, IoT, 시스템 보안

※ 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구(No.2019-0-00708, 뉴로모픽 아키텍처 기반 자율형 IoT 응용통합개발환경)이며 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2021R1F1A1062884).