Journal of Korea Multimedia Society Vol. 23, No. 3, March 2020(pp. 476-482)
https://doi.org/10.9717/kmms.2020.23.3.476

머신러닝을 위한 블록형 모듈화 아키텍처 설계

오 유 수†

Design of Block-based Modularity Architecture for Machine Learning

Yoosoo Oh†

ABSTRACT

In this paper, we propose a block-based modularity architecture design method for distributed machine learning. The proposed architecture is a block-type module structure with various machine learning algorithms. It allows free expansion between block-type modules and allows multiple machine learning algorithms to be organically interlocked according to the situation. The architecture enables open data communication using the metadata query protocol. Also, the architecture makes it easy to implement an application service combining various edge computing devices by designing a communication method suitable for surrounding applications. To confirm the interlocking between the proposed block-type modules, we implemented a hardware-based modularity application system.

Key words: Machine Learning, Block Module, Modularity, Edge Computing, IoT

1. 서 론

최근 엣지 컴퓨팅 환경에서의 머신러닝 기술이 학계와 산업계에서 많은 관심을 받고 있다[1]. 또한, 다양한 엣지 컴퓨팅 기반의 머신러닝 응용이 개발되고 있다[2]. 엣지 컴퓨팅은 수집된 데이터를 이용하여 네트워크의 엣지에서 프로세싱을 수행하여 IoT 서비스를 제공한다[3]. 특히, 모바일 엣지 컴퓨팅은 클라우드 컴퓨팅 서비스를 모바일 사용자에게 제공한다[4,5]. 엣지 컴퓨팅 환경에서의 머신러닝 기술은 예측이 필요한 입력에 대하여 대응 가능한 모델을 만들기 위하여 많은 양의 데이터를 활용하는 것이다.

빅 데이터의 증가로 인해 대용량 데이터 세트와 수십억 개의 매개변수로 복잡한 모델을 학습하기 위한 머신러닝 시스템에 대한 새로운 요구가 발생하였다[1,22]. 특히, 데이터의 양이 증가하고 모델의 복잡도가 증가함에 따라 하나의 머신 러닝 장치로는 정확한 머신러닝 응용을 개발하기 어려운 문제가 있다. 이와 같은 문제를 해결하기 위하여 분산형 머신러닝 기술이 개발되었다[6,7,8]. 분산형 머신러닝 기술은 하나의 장치에서 학습이 느리거나 불가능한 모델을 다수의 서버를 활용하여 가능하게 한다.

엣지 컴퓨팅에서의 머신러닝은 클라우드 컴퓨팅 환경에서의 것과는 다르다. 클라우드 컴퓨팅에서의 클라우드는 저장 장치나 관리 센터로써 전체 시스템에 대한 중앙 제어를 하나, 엣지 컴퓨팅에서의 엣지 서버들은 근처의 IoT 장치들의 서비스 요구들에 적절하게 반응한다[9]. 특히, 엣지 컴퓨팅 환경에서의 분산형 머신러닝 기술은 IoT 장치들의 복잡한 모델의 학습을 가능하게 한다.

※ Corresponding Author : Yoosoo Oh, Address: (38453) 201, Daegudae-ro, Gyeongsan-si, Gyeongsangbuk-do, S.Korea, TEL : +82-53-850-6654, FAX : +82-53-850-6629, E-mail : yoosoo.oh@gmail.com
Receipt date : 2019-12-27, Revision date : 2020-02-13
Approval date : 2020-03-24
† Department of Artificial Intelligence, School of ICT Convergence, Daegu University
※ This research was supported by the Daegu University, 2019

다양한 종류의 IoT 장치들이 많은 양의 데이터와 학습된 모델을 이용하여 머신러닝을 수행하기 위한 방법으로는 엣지 컴퓨팅의 엣지 노드와 같이 머신러닝 블록을 생성하는 접근법들이 있다[10,11]. 이와 같은 접근법들은 머신러닝 프레임워크를 통하여 실시간 애드혹 블록을 생성하여 송수신이 가능한 학습 블록을 생성하거나[10], 모델링 단계에서 다양한 고수준의 기술을 사용하는 재사용이 가능한 소프트웨어 모듈을 생성한다[11].

그러나 기존의 연구들은 모바일 환경에서의 IoT 장치와 연동된 엣지 노드의 유기적인 활용 방안이 부족하다. 기존의 연구들은 엣지 네트워크에 새로운 엣지 노드를 추가하거나 삭제하는 등의 확장이 어렵다. 또한, 기존 연구들은 엣지 노드들 간의 자유로운 통신이 가능한 유기적인 연동이 미흡하다. 그리고 기존 연구들은 엣지 노드에 다양한 종류의 머신러닝 알고리즘들을 탑재하여 조합하는 다양성이 부족하다. 따라서 분산형 엣지 컴퓨팅 환경에서 다양한 머신러닝 알고리즘들을 IoT 장치들과 연동하여 머신러닝을 수행할 수 있는 아키텍처가 필요하다.

본 논문에서는 분산형 머신러닝을 위한 모듈화 기술 설계 방법을 제안한다. 제안된 설계 방법은 다양한 머신러닝 알고리즘을 탑재한 블록형 모듈 아키텍처를 포함한다. 또한, 본 논문에서는 블록형 모듈 간의 자유로운 추가 및 삭제가 가능하도록 확장 가능한 아키텍처를 설계한다. 설계된 아키텍처는 블록 간 제한 없는 조합이 가능하다. 마지막으로 다양한 머신러닝 알고리즘들을 상황에 맞게 유기적으로 연동되도록 블록 모듈을 설계한다.

제안된 설계 방법은 블록형 모듈로 인하여 다양한 엣지 컴퓨팅 장치들을 조합한 응용화가 수월하다. 또한 정규화된 메타 데이터에 기반을 둔 프로토콜을 이용하여 블록형 모듈 간의 빠른 통신이 가능하다. 그리고 제안된 방법은 블록형 모듈 간 단방향 혹은 양방향 통신을 지원한다. 설계된 아키텍처는 응용 서비스에 따른 유무선 통신 및 전송 방법을 제공한다.

2. 제안된 설계 방법

2.1 AI 블록형 모듈화 아키텍처

본 논문에서는 주변 IoT 장치들과 연동가능한 분산형 머신러닝을 위하여 블록형 모듈 구조를 가지는

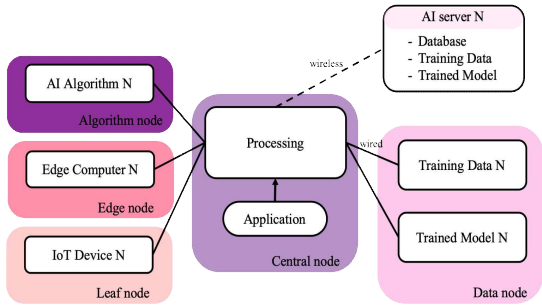


Fig. 1. The Proposed AI-Block Architecture.

머신러닝 모듈화 아키텍처를 설계한다. Fig. 1은 블록형 모듈 아키텍처인 AI-Block Architecture의 설계 다이어그램이다. 설계된 아키텍처는 응용 서비스와 연동되는 프로세싱 유닛, N개의 AI 알고리즘, AI 서버, 학습 데이터, 학습된 모델, 엣지 컴퓨터, 그리고 IoT 장치들로 구성된다. 프로세싱 유닛은 유선이나 무선으로 연결된 학습 데이터를 활용하여 모델을 학습시키고, 학습된 모델을 이용하여 해당 AI 알고리즘으로 연결된 엣지 컴퓨터나 IoT 장치들을 구동시킨다. 학습 데이터나 학습된 모델들은 유선으로 연결된 하드웨어 노드나 무선으로 연결된 AI 서버에 저장되며 학습 데이터를 이용하여 모델을 실시간 학습시킨다. 엣지 컴퓨터나 IoT 장치들은 프로세싱에 연동된 응용 서비스의 특징에 따라서 동작된다. 예를 들어, 자동차 종류를 인식하는 응용 서비스에서는 외부 카메라가 IoT 장치가 되며, 이미지인식 딥러닝 알고리즘이 연동되어 사용될 수 있다. 엣지 컴퓨터로 사용될 경우에는 딥러닝 알고리즘과 카메라, 프로세싱, 네트워킹 기능을 모두 엣지 컴퓨터에 포함하여 자동차 종류 인식 응용 서비스에 연결될 수 있다.

AI-Block Architecture는 Central node, Algorithms node, Data node, Edge node, Leaf node를 구

Table 1. Block Node Description

Node	Description
Central node	processing, application
Algorithm node	machine learning algorithm
Data node	AI server or training data, training model
Edge node	edge computer or another central node
Leaf node	IoT device, sensing device, display

성한다. Table 1은 AI-Block Architecture의 각 노드에 대한 설명이다. Central node는 블록화 모듈의 엣지 처리 노드로써 프로세싱과 응용 서비스를 포함한다. Algorithm node는 다양한 머신러닝 알고리즘들을 모듈화하여 블록형태로 구성한 것이다. 특히, Algorithm node는 naive bayes, decision tree, SVM, kNN, Random forest, K-means, Gaussian mixture 등의 알고리즘들을 사용한다. Data node는 학습 데이터와 학습된 모델을 저장하는 역할을 하며, 유선으로 연결된 하드웨어 노느나 무선으로 연결된 AI server에 분산 저장된다. 다수의 AI server는 학습 관련 데이터뿐만 아니라 응용서비스의 데이터베이스로써의 역할도 함께 수행한다. Edge node는 엣지 컴퓨팅을 위한 엣지 장치로써 동작하거나 또 다른 프로세싱과 응용 서비스를 포함하는 Central node로도 동작한다. 분산 머신러닝 환경에서 다수의 Central node들이 Edge node로써 역할을 하며 상호 네트워크 통신을 한다. 마지막으로 Leaf node는 가장 최하단에 배치되는 IoT 장치이거나, 센싱 장치, 디스플레이 등의 입출력 형태로 연결된다. Fig. 2는 AI-Block의 유선 하드웨어 연결의 개념을 보여주는 다이어그램 예시이다.

제안된 블록형 모듈 아키텍처는 블록형 모듈인 노드들 간의 자유로운 추가 및 삭제가 가능하다. 노드들 간의 유무선 연결 구조가 느슨한 결합 구조로 되어있어 제한 없는 조합이 가능하다. 또한, N개의 AI Algorithm 노드들을 이용하여 다양한 머신러닝 알고리즘들을 상황에 맞게 유기적으로 연동할 수 있다. N개의 Edge node들은 다양한 엣지 컴퓨팅 장치들을 조합한 응용 서비스 제공이 쉽게 한다.

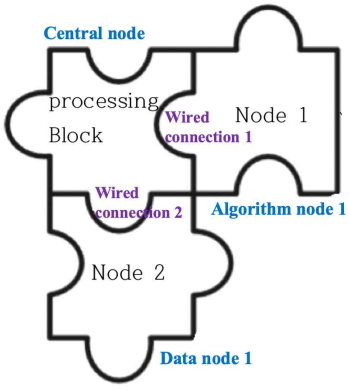


Fig. 2. AI-Block Concept Diagram.

2.2 아키텍처 확장성

제안된 아키텍처는 메타데이터 쿼리 프로토콜을 정의하여 사용한다. 메타데이터 쿼리 프로토콜은 정규화된 메타데이터를 통해서 네트워크 통신 프로토콜을 이용하여 블록형 모듈 간의 빠른 통신을 가능하게 한다. 정규화된 메타데이터는 시맨틱 정보를 나타내는 5W1H(Who, What, Where, When, Why, How)의 컨텍스트 형식[12]을 따른다. 5W1H의 정규화된 메타데이터는 MQTT, HTTP, 시리얼 통신의 네트워크 프로토콜 통신으로 쿼리 형태로 블록 모듈 간 전송된다. 이와 같은 메타데이터 쿼리 프로토콜은 블록형 모듈 아키텍처에서 블록형 모듈 간의 자유로운 조합이 가능하여 제한 없는 확장성을 지원한다.

AI-Block 설계 방법은 Fig. 2와 같이 머신러닝을 위한 모듈화 구조로 하드웨어 블록(block)형태 혹은 브릭(brick)형태로 설계되어 비접촉식(무선)/접촉식(유선) 통신이 가능하다. 비접촉식 블록 연동의 경우에는 MQTT나 HTTP를 통한 메타데이터 쿼리 프로토콜로 데이터를 송수신한다. 접촉식 블록 연동의 경우에는 직접적인 블록 맞춤으로 시리얼 통신 데이터를 송수신한다.

2.3 모듈 통신 방법

제안된 아키텍처는 단방향 혹은 양방향 통신으로 다양한 블록 모듈들을 상황에 맞게 유기적으로 연동한다. Fig. 3은 AI-Block 아키텍처의 상황에 따른 통신 방향을 나타낸다. 단방향 통신은 입력(센싱, 머신러닝 알고리즘)과 출력(디스플레이) 노드에 사용된다. 양방향 통신은 프로세싱(퓨전, 추론) 노드와 데이터(학습, 모델링) 노드에 사용된다. Fig. 4와 Fig. 5는

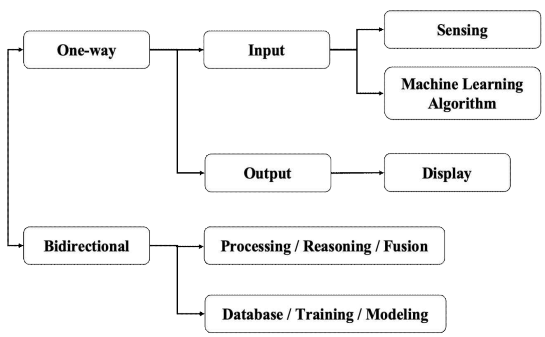


Fig. 3. AI-Block Communication Direction.

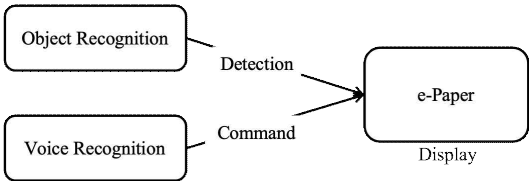


Fig. 4. Direct Communication for AI-Block Architecture.

제안된 아키텍처의 통신 방식에 대한 예시를 나타낸다. Fig. 4는 직접적인 통신방식으로 물체인식과 음성인식의 머신러닝 결과 데이터를 직접적으로 디스플레이인 전자종이(e-Paper) 시스템에 전송하는 것을 나타낸다. Fig. 5는 융합 통신방식으로 물체인식과 음성인식의 인식 데이터를 프로세싱 노드가 융합(fusion)하여 머신러닝을 수행한 후에 그 결과 데이터를 디스플레이인 전자종이 시스템에 전송하는 것을 나타낸다. 이와 같이, AI-Block 아키텍처는 주어진 응용 상황에 따라 블록형 모듈들을 연동함으로써 유기적인 통신이 가능하게 한다.

3. 구 현

3.1 블록형 모듈화 비교 분석

제안된 아키텍처의 블록형 모듈화 구조를 관련된 연구를 기반으로 다음 Table 2와 같이 비교 분석하였다. Table 2는 제안된 아키텍처의 특징들을 기반으로 분석하였다. 분석 결과, 기존 연구들 중 일부는 대용량의 데이터를 활용하여 구조적 확장성을 가능하게 하고, 개별적인 통신 방법으로 머신러닝 알고리즘들을 연동하였다. 그러나 기존 연구들은 머신러닝 알고리즘 및 학습 데이터들을 모듈화하여 블록 형태의 개별 분산 구조로 구성하지 않는다. 특히, 정규화된 메타데이터 프로토콜을 정의하지 않기 때문에 개별 분산된 머신러닝 블록 간의 정규화된 통신 방법이 미흡하다. AI-Block 아키텍처는 분산형 엣지 컴퓨팅 환경에서 머신러닝 알고리즘 및 관련된 학습 데이터들의 모듈화된 블록 구조로 설계되었으며, 모듈 간 5W1H의 형태로 정규화된 메타데이터 쿼리를 정의

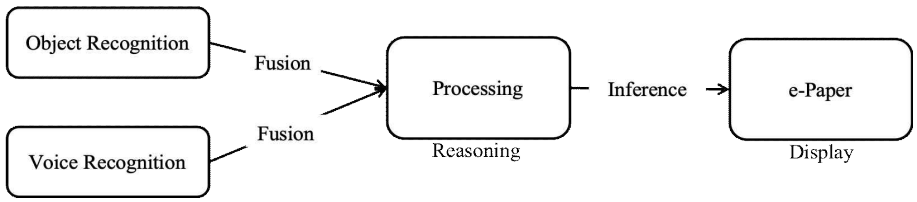


Fig. 5. Integration Communication for AI-Block Architecture.

Table 2. Block Modularity Comparison

Modularity	Scalability	Communication	Data Protocol	Related Research
Distributed learning machine	Large data	Bidirectional communication	Not defined	[8]
Distributed learning machine	Not defined	Bidirectional & Consensus communication	Not defined	[7]
Distributed learning machine	Large data	Data-Model parallelism, Continuous communication	Not defined	[6]
Distributed learning machine	Not defined		Synchronization parameter	[2]
Real-time ad hoc blocks	Ad hoc expansion	Bidirectional communication using channel media	Not defined	[10]
Distributed machine learning block module	Seamless ad hoc expansion	Wired/Wireless, One-way or Bidirectional communication	Unified metadata query protocol	AI-Block [The proposed work]

하여 HTTP나 MQTT 등의 통신 프로토콜로 상황에 맞는 단방향 혹은 양방향 통신을 지원한다.

3.2 AI-Block 아키텍처 구현

AI-Block 아키텍처의 효과를 입증하기 위하여 Fig. 6과 같이 시스템을 구현하였다. 머신러닝 개발과 관련된 기존의 소프트웨어 아키텍처로는 Tensor Flow [13,14], MXNet [15], Caffe2 [16] 등이 있고, 엣지 컴퓨팅을 위한 머신러닝 하드웨어 아키텍처로는 인텔 모비디우스[17], 엔비디아 젯슨 보드[18,19], 구글 코럴보드[20,21] 등이 있다. 기존의 머신러닝 소프트웨어 아키텍처들은 블록형 모듈화 구조와는 다르나, 기존의 머신러닝 하드웨어 아키텍처들은 엣지 노드의 블록 모듈로 활용이 가능하다. 그래서 본 논문에서는 기존의 머신러닝 하드웨어 아키텍처 중 하나인 구글 코럴보드를 활용하여 Fig. 6과 같은 응용 시스템을 개발하였다. Fig. 7은 개발된 응용 시스템에 사용된 하드웨어를 나타낸다. 개발된 시스템은 블록 모듈 간 연동을 나타내기 위하여 구현한 것으로

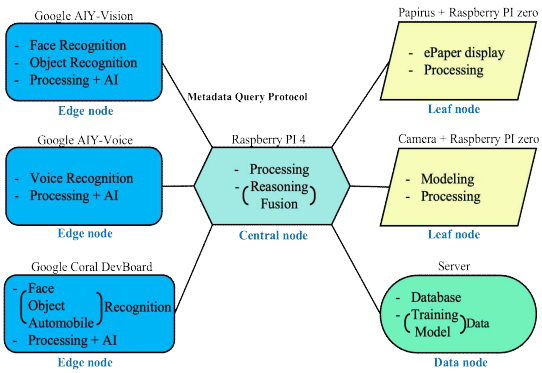


Fig. 6. Implementation for AI-Block Architecture.

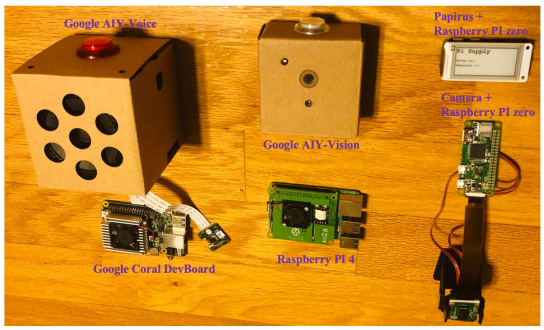


Fig. 7. Implemented Real Devices.

비전 Edge node의 테스트 영상인식 결과, 음성 Edge node의 테스트 음성인식 결과, 코럴보드 Edge node의 테스트 영상인식 결과를 라즈베리파이 Central node에서 프로세싱하여 Papirus Leaf node에 단순 출력한 것이다. 그리고 카메라 영상 Leaf node의 영상을 입력으로 받아서 라즈베리파이 Central node에서 출력 확인하였다. 개발된 시스템은 단순 테스트 데이터들을 송수신하는 예제로써 제안된 아키텍처의 블록 모듈화 가능성을 확인하기 위함이다.

4. 결 론

본 논문에서는 분산형 머신러닝을 위한 블록형 모듈화 아키텍처 설계 방법을 제안하였다. 제안된 아키텍처는 다양한 머신러닝 알고리즘을 탑재한 블록형 모듈 구조로써 블록형 모듈 간의 자유로운 확장이 가능하고, 다양한 머신러닝 알고리즘들을 상황에 맞게 유기적으로 연동되게 한다. 제안된 아키텍처는 정의된 메타데이터 쿼리 프로토콜을 이용하여 자유로운 데이터 통신이 가능하게 한다. 또한, 주변 응용 상황에 적합한 통신 방법을 설계하여 다양한 엣지 컴퓨팅 장치들을 조합한 응용 서비스 구현을 쉽게 한다. 그리고 블록형 모듈 간의 연동을 확인하기 위하여 하드웨어 기반의 모듈화 응용 시스템을 구현하였다.

추후 연구에서는 제안된 블록형 모듈을 실제 하드웨어 블록이나 브릭 형태로 제작하고 다양한 엣지 컴퓨팅 응용 시스템에 적용할 것이다. 설계된 아키텍처의 성능과 자유도를 실험하고 정확한 머신러닝 응용 정도를 평가할 것이다. 본 논문에서의 블록형 모듈화 설계는 분산형 엣지 컴퓨팅 환경에서 다양한 머신러닝 알고리즘들을 IoT 장치들과 연동하여 머신러닝을 수행하는 기초 연구가 될 것이다.

REFERENCE

[1] R. Gu, S. Yang, and F. Wu, “Distributed Machine Learning on Mobile Devices: A Survey,” *Engineering*, Vol. 1, No. 1, pp. 1-28, 2019.

[2] Distributed Machine Learning through Heterogeneous Edge Systems, http://arxiv.org/abs/1911.06949 (accessed December 24, 2019).

[3] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu.

"Edge Computing: Vision and Challenges," *IEEE Internet of Things Journal*, Vol. 3, No. 5, pp. 637-646, 2016.

[4] C. Mouradian, D. Naboulsi, S. Yangui, R.H. Glitho, M.J. Morrow, P.A. Polakos, et al., "A Comprehensive Survey on Fog Computing: State-of-the-art and Research Challenges," *IEEE Communications Surveys and Tutorials*, Vol. 20, No. 1, pp. 416-464, 2018.

[5] Y. Mao, C. You, J. Zhang, K. Huang, and K.B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys and Tutorials*, Vol. 19, No. 4, pp. 2322-2358, 2017.

[6] E.P. Xing, Q. Ho, P. Xie, and D. Wei, "Strategies and Principles of Distributed Machine Learning on Big Data," *Engineering*, Vol. 2, No. 2, pp. 179-195.

[7] L. Georgopoulos and M. Hasler, "Distributed Machine Learning in Networks by Consensus," *Neurocomputing*, Vol. 124, pp. 2-12, 2014.

[8] M. Luo, L. Zhang, J. Liu, J. Guo, and Q. Zheng, "Distributed Extreme Learning Machine with Alternating Direction Method of Multiplier," *Neurocomputing*, Vol. 261, pp. 164-170, 2017.

[9] J. Ren, D. Zhang, S. He, Y. Zhang, and T. Li, "A Survey on End-edge-cloud Orchestrated Network Computing Paradigms: Transparent Computing, Mobile Edge Computing, Fog Computing, and Cloudlet," *Association for Computing Machinery Computing Surveys*, Vol. 52, No. 6, pp. 125(1)-125(36), 2019.

[10] Deep Modulation (Deepmod): A Self-taught PHY Layer for Resilient Digital Communications, http://arxiv.org/abs/1908.11218 (accessed December 10, 2019).

[11] Machine Learning Blocks Ingles, https://dspace.mit.edu/handle/1721.1/100301 (accessed December 24, 2019).

[12] Y. Oh, J. Han, and W. Woo, "A Context Management Architecture for Large-scale Smart Environments," *IEEE Communications Magazine*, Vol. 48, No. 3, pp. 118-126, 2010.

[13] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, et al., "Tensorflow: A System for Largescale Machine Learning," *Proceeding of Operating Systems Design and Implementation*, Vol. 16, pp. 265-283, 2016.

[14] Use Tensorflow to Train a CNN on Cifar-10, https://github.com/tensorflow/models/tree/master/tutorials/image/cifar10 (accessed December 24, 2019).

[15] Mxnet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems, https://arxiv.org/abs/1512.01274 (accessed December 10, 2019).

[16] K. Hazelwood, S. Bird, D. Brooks, S. Chintala, U. Diril, D. Dzhulgakov, et al., "Applied Machine Learning at Facebook: A Datacenter Infrastructure Perspective," *Proceeding of High-performance Computer Architecture*, pp. 620-629, 2018.

[17] Intel Discontinues First-generation Movidius Neural Compute Stick, http://libproxy. daegu.ac.kr/_Lib_Proxy_Url/http://search.ebscohost.com/login.aspx?direct=true&db=edsggo&AN=edsgcl.584332327&lang=ko&site=eds-live (accessed December 15, 2019).

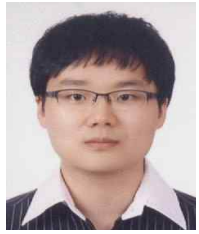
[18] S. Mittal, "A Survey on Optimized Implementation of Deep Learning Models on the NVIDIA Jetson Platform," *Journal of Systems Architecture*, Vol. 97, pp. 428-442, 2019.

[19] Y. Nam, H. Shin, S. Choi, B. Yoo, and H. Eom, "Performance Evaluation on the Nvidia Jetson TX2 for High Performance IoT Data Stream Processing," *Proceeding of Korea Software Congress*, pp. 1420-1422, 2018.

[20] Google Makes Coral AI Platform Available to Developers, http://libproxy.daegu.ac.kr/ _Lib_Proxy_Url/http://search.ebscohost.com/login.aspx?direct=true&db=edsggo&AN=edsgcl.578163393 &lang=ko&site=eds-live (accessed December 10, 2019).

[21] Google's Coral for Local AI Includes Raspberry Pi, http://libproxy.daegu.ac.kr/_Lib_Proxy_Url/http://search.ebscohost.com/login.aspx?direct=true&db=bth&AN=137193364&lang=ko&site=eds-live (accessed December 10, 2019).

[22] J. Ahn, U.G. Kang, Y.H. Lee, and B.M. Lee, "Improvement of Activity Recognition based on Learning Model of AI and Wearable Motion Sensors," *Journal of Korea Multimedia Society*, Vol. 21, No. 8, pp. 982-990, 2018.



오 유 수

1997년~2002년 경북대학교 전파공학, 공학사
2002년~2003년 GIST 정보통신공학, 공학석사
2003년~2010년 GIST 정보기전공학, 공학박사
2010년~2012년 GIST CTI, 총괄팀장
2017년~2019년 대구대학교 혼합현실융합연구센터, 센터장
2020년~현재 대구대학교 AZIT메이커스페이스 센터장
2012년~현재 대구대학교 ICT융합학부, 교수
관심분야 : 머신러닝, 인공지능 미들웨어, 인터랙티브 시스템, HCI 등