The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 21, No. 3, pp.107-114, Jun. 30, 2021. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2021.21.3.107
JIIBC 2021-3-15

오픈신경망 포맷을 이용한 기계학습 모델 변환 및 추론

Model Transformation and Inference of Machine Learning using Open Neural Network Format

김선민*, 한병현*, 허준영**

Seon-Min Kim*, Byunghyun Han*, Junyeong Heo**

요 약 최근 다양한 분야에 인공지능 기술이 도입되고, 학계 관심이 늘어남에 따라 다양한 기계학습 모델들이 여러 프레임워크에서 운용되고 있다. 하지만 이러한 프레임워크들은 서로 다른 데이터 포맷을 가지고 있어, 상호운용성이 부족하며 이를 극복하기 위해 오픈 신경망 교환 포맷인 ONNX가 제안되었다. 본 논문에서는 여러 기계학습 모델을 ONNX로 변환하는 방법을 설명하고, 통합된 ONNX 포맷에서 기계학습 기법을 판별할 수 있는 알고리즘 및 추론 시스템을 제안한다. 또한, ONNX 변환 전·후 모델의 추론 성능을 비교하여 ONNX 변환 간 학습 결과의 손실이나 성능 저하가 없음을 보인다.

Abstract Recently artificial intelligence technology has been introduced in various fields and various machine learning models have been operated in various frameworks as academic interest has increased. However, these frameworks have different data formats, which lack interoperability, and to overcome this, the open neural network exchange format, ONNX, has been proposed. In this paper we describe how to transform multiple machine learning models to ONNX, and propose algorithms and inference systems that can determine machine learning techniques in an integrated ONNX format. Furthermore we compare the inference results of the models before and after the ONNX transformation, showing that there is no loss or performance degradation of the learning results between the ONNX transformation.

Key Words : Deep Learning, Inference System, Keras, Machine Learning, Model Transformation, Nengo, Neural Network, ONNX, ONNX Runtime, Protobuf, Scikit-learn, Spike Neural Network

Ⅰ. 서 론

1. 연구내용

최근 다양한 분야에 인공지능 기술이 도입되고, 학계의 관심이 늘어남에 따라 머신러닝(ML, Machine learning), 딥러닝(DL, Deep Learning) 등의 다양한 기계학습 모델 운용 관련 연구가 활발하게 진행되고 있다. 또한, 여러 기업에서는 그러한 변화에 맞추어 다양한

*학생회원, 한성대학교 컴퓨터공학부
*학생회원, 한성대학교 컴퓨터공학부
**정회원, 한성대학교 컴퓨터공학부(교신저자)
접수일자 2021년 3월 25일, 수정완료 2021년 4월 25일
게재확정일자 2021년 6월 4일

Received: 25 March, 2021 / Revised: 25 April, 2021 / Accepted: 4 June, 2021
*Corresponding Author: jyheo@hansung.ac.kr
Dept. Division of Computer Engineering, Hansung University, Korea

기계학습 모델을 개발하고 운용할 수 있는 다양한 프레임워크를 개발하여 공개하고 있다.[1]

현재 여러 프레임워크들은 각기 다른 데이터 포맷을 가지는데 대표적인 예시로 Google Tensorflow의 .pb, Apache MXNet의 .model, Facebook Caffe2의 .caffe model, Pytorch의 .pt 등이 존재한다. 하지만 이러한 프레임워크들의 고유한 데이터 포맷 형식으로 작성된 기계학습 모델은 해당 프레임워크에 종속되어, 타 프레임워크와의 상호운용성이 부족하다는 문제점이 존재한다.

이러한 문제를 해결하기 위해 프레임워크에 독립적인 모델 표현들이 제안되었고, 대표적으로 오픈 신경망 교환 포맷(ONNX, Open Neural, Network eXchange)이 있다.

본 논문에서는 다양한 기계학습 모델을 오픈 신경망 포맷(ONNX)으로 변환하는 방법을 설명하고, 변환된 ONNX 포맷 구조 분석을 통해 사용된 기계학습 기법을 판별할 수 있는 방법을 설명한다. 최종적으로는 이러한 판별을 통해 적절한 프레임워크를 사용할 수 있는 추론 시스템을 제안하여 ONNX 라는 공동의 모델 포맷 운용을 통해 인공지능 프레임워크 개발 공동체에 더 나은 개방형 시스템을 기대하게 한다.

본 논문의 구성은 다음과 같다. 2장에서는 ONNX의 구조, 연산자 및 추론 엔진에 대해 설명하고, 3장에서는 머신러닝과 딥러닝의 ONNX 포맷으로의 변환을 설명한다. 4장에서는 SNN 스파이킹 신경망 모델과 ONNX 포맷간의 양방향 변환을 설명하고, 5장에서는 변환된 ONNX 모델에서의 기계학습 기법을 판별하는 알고리즘을 설명한다. 6장에서는 ONNX 변환 전·후 모델의 추론 결과를 비교하며, 마지막으로 7장에서 결론을 맺는다.

II. ONNX 포맷

1. Protobuf

Protobuf(Protocol Buffers)는 Google사에서 개발하고 오픈소스로 공개한 직렬화(Serialization) 된 데이터 구조로, 직렬화란 데이터를 파일로 저장하거나 네트워크로 전송하기 위해 바이너리 스트림 형태로 저장하는 행위를 일컫는다. Protobuf의 주요 장점으로는 한가지 언어에 종속되지 않고 C++, Java, Python, JavaScript, Ruby, Go 등 다양한 언어로 작성 가능하다는 점, 직렬화 속도가 빠르고 결과 파일의 크기 또한 작다는 점등의 장점이 있다. 프로토콜 버퍼는 다양한 언어를 지원하기 때문에, 그림 1. 처럼 특정 언어에 종속성이 없는 Proto file 이라는 형태(.proto)로 정의되며, 이렇게 정의된 데이터 타입은 protoc 컴파일러를 통해 각 언어에 맞는 형태의 데이터 클래스 파일이 생성된다.

2. ONNX

오픈신경망교환포맷(ONNX, Open Neural Network eXchange)은 Microsoft와 Facebook에서 공동 개발한 인공지능 프레임워크로, 내부는 Protobuf로 정의되어 있으며 다양한 종류의 프레임워크 데이터 포맷과의 변환라이브러리를 지원한다.[2] 현재 여러 인공지능 프레임워크들은 그래프를 표현하는 데 있어, 각기 다른 그래프 표현방식을 사용하는데. ONNX는 중간표현(IR Intermediate Representation)을 통해 공통의 그래프 형식으로 나타내어 진다. 결론적으로 ONNX는 Protobuf 포맷으로 구성되어 있어, 1.에서 언급한 장점을 가지고, 중간표현 그래프 표현 방식을 통해 다른 변형 포맷으로 만들기도 용이하기 때문에 공통된 데이터 포맷 형식으로 적합하다.

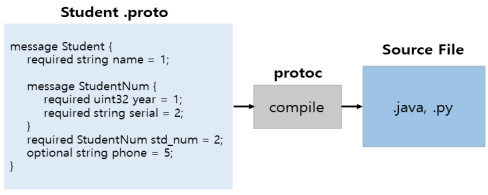
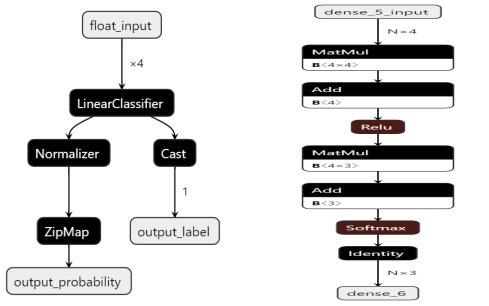


그림 1. protobuf 언어별 적용 과정
Fig. 1. protobuf language-specific application process

3. ONNX 연산자

ONNX는 크게 두 가지 종류의 연산자를 가지는데, 하나는 ONNX 연산자이고, 다른 하나는 ONNX-ML 연산자이다. ONNX-ML은 머신러닝(ML) 연산에 필요한 타입이나 연산 방법에 대해 보다 최적화된 정보를 가진 자체 연산자가 정의되어 있고, ONNX 연산자는 신경망 레이어 및 벡터 및 행렬 계산 관련 연산자들이 정의되어 있다. 즉 일반 머신러닝 알고리즘이 ONNX로 변환 및 표현 시 ONNX-ML 연산자를 이용하여 사용되고, 신경망 기반 알고리즘은 ONNX 연산자를 통해 표현된다. 다음 그림 2. 는 머신러닝 분류 알고리즘인 로지스틱 회귀 (Logistic Regression)와 딥러닝 신경망 알고리즘인 MLP(Multi-Layer-Perceptron)을 ONNX로 표현한 것이다.



(a) Logistic regression (b) Multi-Layer-Perceptron

그림 2. ONNX 모델 표현

Fig. 2. ONNX Model Representation

4. ONNX Runtime

ONNX Runtime은 ONNX 형식의 기계학습 모델을 위한 고성능 추론용 엔진으로 Linux, Window, MacOS 등을 지원한다. 또한, ONNX Runtime은 SNN 모델을 제외한 다양한 ML/DL 모델의 추론 작업이 가능하며, 추론시 CPU 외에 GPU도 사용할 수 있다. ONNX Runtime은 많은 연산자로 된 그래프의 최적화를 수행해 메모리 복사 횟수를 줄이는 방식으로 성능을 개선하며, Microsoft의 2020년 1월 실험결과에 따르면 ONNX Runtime을 이용해서 BERT 알고리즘의 추론시간을 17배 이상으로 가속화가 가능하다.[3][4] 이와 같이 ONNX 및 ONNX Runtime을 활용해 기존에 익숙한 오픈소스 개발 툴과 함께 상호운용성과 빠르고 안정된 성능을 보장되는 AI 모델 운용이 가능하다.

Ⅲ. ML, DL 모델의 ONNX 변환

1. Mahine Learning Model to ONNX

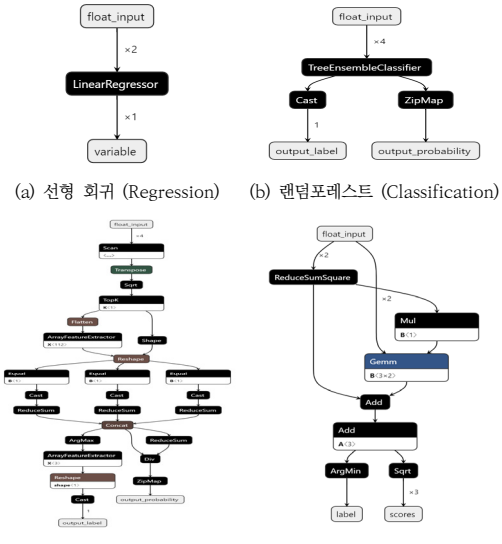
주요 머신러닝 라이브러리 중 하나인 Scikit-learn은 기본적인 데이터 전처리 기능 와에, 분류(classification), 회귀(regression), 의사결정 트리(decision tree), 군집화(clustering), SVM(Support Vector Machine)등 과 같은 대다수의 전통적인 머신러닝 기법들의 API를 가지고 있으며 예제 데이터를 이용하여 손쉽게 모델 구현이 가능하다. scikit-learn으로 구현 후 학습된 머신러닝 모델은 sklearn-onnx 라이브러리의 설치 후 해당 라이브러리의 API를 통해 ONNX 모델로 변환할 수 있다. 다음 그림 3. 은 sklearn-onnx 패키지의 API를 이용해 변환된 머신러닝 기반 ONNX 모델들이며, 기계학습 시각화 툴인 Netron을 이용하여 시각화하였다.

왼쪽상단부터 차례대로 선형회귀 (Linear Regression), 랜덤 포레스트 (Random Forest)[5], K-최근접 이웃 알고리즘 (K-Nearest Neighbor, KNN), K-Means 군집 모델이다. 머신러닝 모델에서 ONNX로 변환된 모델들은 ONNX-ML 연산자 노드들로 그래프가 표현된 것을 확인할 수 있다.

2. Deep Learning Model to ONNX

신경망 딥러닝 모델은 대표적인 딥러닝 라이브러리인 Tensorflow의 Keras를 이용하였다. 해당 변환도 기존 오픈소스로 제공되는 keras2onnx 라이브러리를 설치 및 해당 라이브러리의 API를 통해 해당 딥러닝 모델을 ONNX 모델로 변환하였다. 아래 그림 4. 는 딥러닝 신경망 기반 모델을 Netron으로 시각화한 것이다.

왼쪽상단부터 차례대로 CNN 기반의 Lenet-1, VGGNet, ResNet50, DenseNet121이고[6], Keras 저장 포맷인 .h5 확장자로 저장된 모델을 .onnx 모델로 변환하였다. 일반적으로 CNN 기반의 모델에서의 Convolution 층, Pooling층 등은 변화가 없었고, Dense층 외의 모델 일부분에서 onnx 연산자로 노드가 일부 변환되어 그래프가 표현된 것을 알 수 있다.



(a) 선형 회귀 (Regression) (b) 랜덤포레스트 (Classification)

(c) KNN (Classification) (d) K-Means 군집 (Clustering)

그림 3. ONNX 머신러닝 모델

Fig. 3. ONNX Machine-learning Model

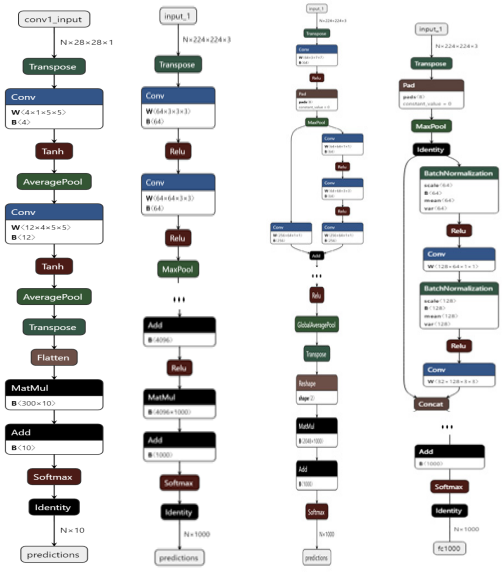


그림 4. ONNX 딥러닝 모델
Fig. 4. ONNX Deep-learning Model

Ⅳ. SNN 모델의 ONNX 양방향 변환

1. SNN

SNN(Spiking Neural Network)은 생물학적인 뇌의 뉴런이 출력을 결정하는 메커니즘을 모방한 차세대 스파이킹 신경망이다. 스파이킹 신경망(SNN)은 기존의 신경망과는 달리 학습 데이터에 Time Sequence를 부여하고, 시간에 따라 각 노드에서 다음 뉴런으로 값을 전달하기 위해 활성화 함수에 0 또는 1의 바이너리 값을 넘긴다. 그리고 이 값을 누적하여 해당 활성화 함수에 설정된 역치(Threshold)값을 넘어설 때, 다음 뉴런으로 신호가 전달되게 한다. 스파이킹 신경망의 학습은 시냅스 전 스파이크와 시냅스 후 스파이크의 시간 차이를 통한 시냅스 가중치를 활용해 학습이 진행되며, 다양한 지도, 비지도 학습법들이 개발되고 있다.[7][8] 스파이킹 신경망 관련 연구는 현재는 초기 단계이지만, 향후 적은 전기신호로도 동작이 가능한 뉴로모픽 칩의 성장과 더불어 기존 신경망들의 연산에 시간이 많이 소요된다는 문제점의 대안으로 높은 연구가치를 지닌다.[9]

2. Nengo

현재 SNN과 같은 차세대 신경망의 시뮬레이터를 지원하는 프레임워크로는 Nengo와 다양한 딥러닝 연산을 SNN으로 변환하여 사용할 수 있도록 지원하는 Nengo-dl등 이 존재한다. Nengo는 스파이킹 신경망과 비 스파이킹 신경망 모두 연산이 가능하며, Nengo-GUI를 이용한 웹 브라우저 모델 시각화 등의 높은 확장성과 유연한 기능을 가진 프레임워크이다.[10] Nengo-dl은 Tensorflow 라이브러리를 이용하여 Nengo 모델을 돌릴 수 있는 프레임워크로 Nengo 모델 매개변수를 최적화하는 파라미터 설정 및 학습 및 추론이 가능하다.[11]

3. Spiking Neural Network & ONNX 양방향 변환

본 논문에서는 SNN을 ONNX로 변환하기 위해, Nengo-dl로 SNN 모델을 학습하였고,[12] 해당 모델을 Protobuf 구조의 ONNX 포맷 형태로 변환 및 학습 된 SNN 가중치를 ONNX 모델에 삽입하였다.

ONNX에서는 SNN에 대한 변환 툴 및 연산자가 아직 지원하지 않아, 그림 5. 의 우측 변환 방법 순서도를 통해 좌측의 SNN의 모델의 ONNX 포맷형태의 그래프를 생성했다. 또한, 모델 변환 과정에서의 가중치를 ONNX 포맷에서도 유지하기 위해, SNN과 ONNX와의 포맷 저장방식을 비교하여 양방향 변환 과정에서 가중치를 해당 포맷에 맞게 shape가 변형되도록 구현하였다.

동일한 모델의 SNN과 ONNX 간의 가중치 Shape은 표 1. 과 같았으며 크게 두 가지의 차이점을 알 수 있다. 첫 번째로 SNN Shape는 입력-출력 순서의 순방향으로 저장이 되어 있는 반면, ONNX의 경우에는 출력-입력의 순서로 역방향으로 저장이 되어 있음을 알 수 있다. 또한, 두 번째로 Convolution 레이어의 경우 Channel Ordering이 서로 역순의 차원으로 가중치가 저장되어

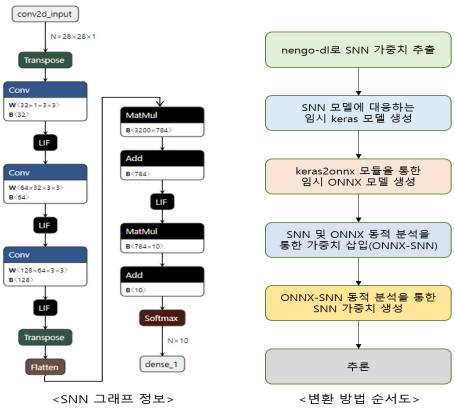


그림 5. SNN Graph 정보 및 변환 순서도
Fig. 5. SNN Graph Information and Conversion Flowchart

표 1. SNN과 ONNX 포맷 비교 결과
Table 1. SNN and ONNX format comparison results

Layer	SNN	ONNX
1	(3, 3, 1, 32)	(784, 10)
2	(32,)	(10,)
3	(3, 3, 32, 64)	(3200, 784)
4	(64,)	(784,)
5	(3, 3, 64, 128)	(128, 64, 3, 3)
6	(128,)	(128,)
7	(3200, 784)	(64, 32, 3, 3)
8	(784,)	(64,)
9	(784, 10)	(32, 1, 3, 3)
10	(10,)	(32,)

있음을 알 수 있다. 이러한 두 가지의 차이점을 고려하여 SNN 모델과 ONNX와의 가중치 변환 및 최종적인 양방향 변환을 구현하였다.

V. ONNX 모델 판별법

1. ONNX 모델 컴포넌트

ONNX 모델은 여러 종류의 컴포넌트(Component)들을 가지며 멤버 연산자(.)를 통해 하위 컴포넌트 및 속성(Properties)에 접근할 수 있다. ONNX 모델의 주요 컴포넌트로는 ir_version, producer_name, domain, graph, node, op_type이 있다. ir_version은 해당 모델의 버전 정보를, producer_name은 모델 생성 도구의 이름 정보를, domain은 해당 모델의 도메인 정보를 의미한다. 도메인 정보의 예시로는 onnx, ai.onnx가 있으며 이들은 신경망 전용 연산자 ONNX와 머신러닝 전용 연산자 ONNX-ML에 각각 매칭된다. graph는 모델을 구성하는 실제 정보로 내부는 해당 모델의 연산자로 구현된 node 들로 구성되어 있으며, op_type은 일반적으로 node 컴포넌트의 하위 속성으로 접근할 수 있는데 활성화 함수의 정보를 나타낸다.

표 2. 컴포넌트 리스트
Table 2. Component list

ir_version	모델 버전 정보
producer_name	모델 생성 도구 이름 정보
domain	domain 정보 ex) onnx, ai.onnx
graph	모델을 구성하는 실제 정보들
node	onnx 그래프의 하나의 노드를 표현
op_type	연산자 타입 정보

```
ir_version: 6
producer_name: "keras2onnx"
producer_version: "1.6.1"
domain: "onnx"
model_version: 0
doc_string: ""
graph {
  node {
    input: "adjusted_input1"
    input: "conv1_2/kernel:0"
    input: "conv1_2/bias:0"
    output: "convolution_output1"
    name: "conv1"
    op_type: "Conv"
    attribute {
      name: "auto_pad"
      s: "NOTSET"
      type: STRING
    }
  node {
    input: "convolution_output1"
    output: "activation_3_1/Tanh:0
    name: "Tanh1"
    op_type: "Tanh"
    doc_string: ""
    domain: ""
  }
```

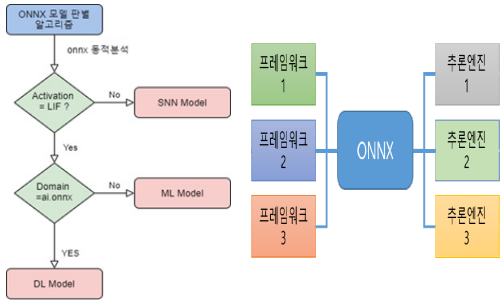
그림 6. ONNX모델 구조
Fig. 6. Structure of ONNX model

2. ONNX 판별 방법

ML, DL, SNN의 3가지 모델 종류에 대해 1차적으로 판별하는 기준은 활성화 함수 종류 정보를 나타내는 op_type을 이용하여 분류하였다. SNN의 경우 기존 ML, DL과 달리 일반적으로 사용되는 Relu, Sigmoid 등을 활성화 함수로 사용하지 않고, LIF와 같은 뉴런 모델을 활성화 함수로 사용한다. 따라서 op_type 속성을 통해서 SNN과 그 외 ML이나 DL 모델로 구별할 수 있었다.

두 번째 기준은 ML과 DL을 구분하는 것인데, 이는 domain 컴포넌트를 이용하여 구분할 수 있다. ONNX 모델이 ML 모델에서 변환된 모델이면 domain 정보는 ai.onnx이고, DL 모델이 ONNX로 변환된 것이라면 domain 정보는 onnx 이다. 따라서 domain 속성을 통해 ML과 DL 모델을 구별할 수 있다.

최종적으로 이 두 가지의 기준을 순차 적으로 적용해 ONNX 모델이 어떤 유형의 모델인지 판별할 수 있었으



(a) ONNX 판별 알고리즘 순서도 (b) ONNX 판별 알고리즘을 통한 추론 시스템

그림 7. ONNX 추론 시스템
Fig. 7. ONNX inference system

며, 모델에 맞는 추론 시스템을 제안할 수 있다. 아래의 그림 7a. 는 판별 알고리즘의 순서도를, 그림 7b. 는 추론 시스템을 나타낸 것이다.

Ⅵ. 실험 및 결과

1. 실험 방법

5장에서 제안한 판별 알고리즘을 이용해 다양한 ONNX 모델을 분류하고 추론 엔진 간의 성능 비교 실험을 진행하였다. ML의 경우 학습한 모델을 Scikit-learn과 ONNX-Runtime 추론 엔진을 사용하였고, DL의 경우 Keras 모델(.h5)과 ONNX-Runtime 추론 엔진을 사용하여 추론기 간의 연산 소요시간을 비교하였다. SNN 모델의 경우에는 ONNX 연산자를 지원하지 않아 ONNX-Runtime 추론 엔진을 사용할 수 없어, Nengo 추론 엔진을 이용한 ONNX 모델로의 변환 전·후의 성능을 평가하여 변환 방법에 대한 유효성을 검증하였다.

2. ML 모델 추론 실험결과

ML 모델과 ONNX 모델은 가중치가 그대로 포함되어 변경되었기 때문에, 모델 변환 전·후 정확도 등의 성능 평가지표는 같다. 성능 평가 실험에서는 boston 데이터셋, iris 데이터셋, 클러스터용 blobs 데이터셋을 이용하였고, 모두 해당 데이터셋을 1000회 추론하는데 걸린 시간을 측정 및 비교하였다. 회귀형 선형회귀 모델, 분류형 로지스틱 회귀, 랜덤 포레스트, 군집형 K-Means 모델에서는 ONNX Runtime에서의 소요시간이 더 적었고, 분류형 모델의 KNN의 모델의 경우는 ONNX Runtime이 추론 소요시간이 더 길었다.

표 3. ML vs ONNX 모델 추론 소요시간 비교
Table 3. ML vs ONNX model inference time comparison

시간 (초)	ML (scikit-learn)	ONNX
Linear Regerssion	0.07초	**0.04초**
로지스틱회귀	0.053초	**0.049초**
랜덤포레스트	7.47초	**0.30초**
KNN	**1.52초**	1.78초
K-Means	0.30초	**0.08초**

※굵은 글씨는 추론 소요시간이 더 적게걸림을 의미함

3. DL 모델 추론 결과

DL 모델도 가중치가 그대로 포함되어 변경되었기 때문에, 모델 변환 전·후 정확도 등의 성능 평가지표는 동일하다. 일반 DNN 모델과 CNN 기반의 모델 4종류로 총 5개의 모델을 실험하였다. DNN과 Lenet-1의 모델은 28x28 사이즈의 mnist data 이미지를 10회 추론하는데 걸리는 시간을 측정하였으며, VGGNet, ResNet DenseNet은 이미지넷 가중치를 사용했으며 224x224 사이즈의 이미지를 10회 추론하는데 걸리는 시간을 측정하였다. DL의 경우 DNN과 CNN 기반의 모델 4종류 모두에서 ONNX-Runtime의 추론 소요시간이 더 적게 소요 되었다.

표 4. DL vs ONNX 모델 추론 소요시간 비교
Table 4. DL vs ONNX model inference time comparison

시간 (초)	DNN	Lenet-1	VGG Net	ResNet	DenseNet
DL (keras)	0.245초	0.05초	3.93초	1.45초	4.48초
ONNX	**0.002초**	**0.01초**	**3.75초**	**0.54초**	**0.95초**

※굵은 글씨는 추론 소요시간이 더 적게걸림을 의미함

4. SNN 모델 Nengo 추론 결과

4장에서 언급한 변환 방법론에 대한 양방향 변환이 성공적으로 이루어졌는지 확인하기 위해, 변환 전 Nengo-dl에서 학습 후 평가한 추론 정확도와, ONNX로의 변환 후 그래프 동적 분석을 통해 얻은 가중치를 SNN 모델에 삽입하여 다시 Nengo 모델로 변환하여 얻어낸 모델의 추론 정확도를 비교하였다. 결과는 다음 표 5.에서 보이는 바와 같이 두 모델 간의 추론 정확도가 동일함을 확인하였고, 변환 방법의 유효함을 확인하였다.

표 5. SNN과 ONNX-SNN의 추론 정확도
Table 5. Inference Accuracy for SNN and ONNX-SNN

	변환 전	변환 후
추론 정확도	98.7%	98.7%

Ⅶ. 결 론

본 연구에서는 다양한 기계학습 모델의 ONNX 변환과 ONNX 모델 판별에 따른 추론 시스템 선택이 가능한 시스템을 제안하고, 모델 변환 전·후 추론 소요시간 성능과 변환 간의 유효함을 실험하였다. ML, DL 모델을 기존의 라이브러리를 통해 ONNX로 변환하였고, 아직

ONNX 포맷으로의 표현이 지원되지 않던 SNN 모델은 ONNX와의 양방향 변환을 직접 구현하고, 변환 전·후의 추론 정확도가 같음을 확인하여 변환법의 유효함을 확인하였다. 또한 ONNX 모델의 컴포넌트 정보 동적 분석을 통해 ONNX 모델의 종류(ML, DL, SNN 등)를 구분할 수 있는 판별식을 제안하고, ONNX 모델 변환 전·후 모델의 추론 결과를 비교함으로써 최종적으로 모델 종류에 알맞은 추론 시스템을 제안할 수 있었다.

향후 여러 프레임워크에서 ML, DL, SNN 등의 유형과 다양한 모델들의 ONNX 변환 및 추론 실험이 진행되고 그러한 실험결과들이 쌓이게 된다면, 다양한 인공지능 프레임워크들은 ONNX를 통해 프레임워크간 상호운용성을 높일 수 있을 것이다. 또한 그로인해 많은 인공지능 개발자들이 인공지능 프레임워크 선택에 있어 보다 높은 자유도를 가질 수 있을 것이며, 이는 인공지능 개발 공동체에 더 나은 생태계 시스템이 구축되는데 기여할 것으로 기대한다.

References

[1] D. Gang.“A Study on the Neural Network Standard Format for Deep Learning” TTA.Journal Vol.179 Oct. 2018

[2] “onnx”, https://github.com/onnx/onnx

[3] “onnxruntime”, https://github.com/microsoft/onnxruntime

[4] “Microsoft open sources breakthrough optimi- zations for transformer inference on GPU and CPU“, cloudblogs.microsoft.com, Jan 21. 2020, https://cloudblogs.microsoft.com/opensource/2020/01/21/microsoft-onnx-open-source-optimizations-transformer-inference-gpu-cpu/

[5] Hoon Jung, Moonsung Park. “A Study of Big data-based Machine Learning Techniques for Wheel and Bearing Fault Diagnosis”, Korea Academy Industrial Cooperation Society, Vol. 19, No. 1 pp. 75-84, 2018 DOI: https://doi.org/10.5762/KAIS.2018.19.1.75

[6] Yeong-Hyeon Byeon, Keun-Chang Kwak. “A Transfer Learning and Performance Comparison of Deep Learning Models for Pedestrian Classification under Automobile Driving Environment”, The Journal of Korean Institute of Information Technology 16(10), 2018.10, 83-92 (10 pages), Oct. 2018
DOI: http://dx.doi.org/10.14801/jkiit.2018.16.10.83

[7] Amirhossein Tavanaei, Masoud Ghodrati, Saeed Reza Kherad pisheh, Timothee Masquelier,Anthony Maida, “Deep Learning in Spiking Neural Networks” arXiv:1804.08150(v4), Apr. 2018
DOI: https://doi.org/10.1016/j.neunet.2018.12.002

[8] P. U. Diehl, M. Cook, “Unsupervised learning of digit recognition using spike-timing-dependent plasticity,” Frontiers in Computational Neuroscience, Aug. 2015. DOI:https://doi.org/10.3389/fncom.2015.00099

[9] S.M. Park, B.G. Choi, J.J. Lee, S. Yeo, G. Park “The Trend of Neuromorphic Technology Based on Spiking Neural Networks”, TTA.Journal, Vol.188 Apr. 2020

[10] T. Bekolay, J. Bergstra, E. Hunsberger, T. DeWolf, T. C. Stewart, D. Rasmussen, X. Choo, A. R. Voelker, and C. Eliasmith. “Nengo: a Python tool for building large-scale functional brain models. Frontiers in Neuroinformatics”, vol. 7 no.48 pp. 1-13, Jan, 2014. DOI: https://doi.org/10.3389/fninf.2013.00048

[11] Daniel Rasmussen,“NengoDL: Combining deep learning and neuromorphic modelling methods”arXiv. 1805.11144(v3), May. 2018
DOI: https://doi.org/10.1007/s12021-019-09424-z

[12] S. Park and J. Heo, “Conversion Tools of Spiking Deep Neural Network based on ONNX,” The journal of the institute of internet, broadcasting and communication, vol. 20, no. 2, pp. 165-170, Apr. 2020. DOI:https://doi.org/10.7236/JIIBC.2020.20.2.165

저 자 소 개

김 선 민(학생회원)

- 2021년 : 한성대학교 컴퓨터공학부 졸업
- 2021년 ~ 현재 : 뉴럴웍스 AI Engineer
- 관심분야 : 기계학습, 딥러닝, CNN, ONNX

한 병 현(학생회원)

- 2018년 ~ 현재 : 한성대학교 컴퓨터공학부(학사)
- 관심분야 : 기계학습, 자연어 처리, 자기지도 학습

허 준 영(정회원)

- 1998년 : 서울대학교 컴퓨터공학과졸업
- 2009년 : 서울대학교 컴퓨터 공학과 졸업(박사)
- 2009년 ~ 현재 : 한성대학교 컴퓨터 공학부 교수
- 관심분야 : 운영체제, 무선 센서 네트워크, 임베디드 시스템, 기계 학습

※ 본 연구는 한성대학교 교내학술연구비 지원과제 임