https://doi.org/10.7840/kics.2024.49.6.847

SNN을 이용한 저전력 침입 탐지 시스템

허 정 윤◆, 이 현 종*, 임 재 한°

Low-Power Intrusion Detection System Using SNNs

Jeong-Yun Heo◆, Hyun-Jong Lee*, Jae-Han Lim°

요 약

사물 인터넷(internet of things, IoT) 기기의 확산으로 인해 일상생활의 편의성은 높아지고 있다. 그러나 IoT 기기는 낮은 전력과 성능으로 인해 보안에 할당할 수 있는 자원이 부족하여 네트워크 공격에 취약하다. 네트워크 공격에 대응하기 위해서 기계학습이나 인공 신경망을 이용한 연구가 높은 성능을 보여주었으나, 높은 연산력 및 전력 요구로 인하여 저전력 IoT 기기에서 사용하기에 부적합하다. 본 논문에서는 에너지 효율성이 높은 스파이킹 신경망(spiking neural network, SNN)를 사용하여, 악성 패킷을 탐지하는 이진 분류와 각각의 클래스를 분류하는 다중 분류를 위한 침입 탐지 시스템을 설계하였다. 해당 시스템을 통해, 이진 분류에서는 99.57%의 정확도로 공격을 탐지하였고 심층 신경망(deep neural network, DNN)에 비하여 전력 소모량을 40% 감소시켰다. 다중 분류에서는 주성분 분석(principal component analysis, PCA)을 사용하여 67.13%의 정확도로 클래스를 분류하였다. PCA를 사용하지 않은 DNN에 비하여 전력 소모량을 15.08% 감소시키고 정확도는 9.29%p 증가시켰다. 이러한 결과를 통해 본 논문에서는 저전력 IoT 기기에서 SNN을 사용하면 보다 효과적이고 효율적으로 네트워크 공격을 탐지하고 대응할 수 있음을 보였다.

Key Words : IoT, SNNs, Intrusion detection system, Malicious packets, Energy efficiency

ABSTRACT

As the number of Internet-of-Things (IoT) devices increases, our information can be easily leaked out by these devices. However, IoT devices have shown security problems due to lack of resources to use conventional security solutions. Previous studies using machine learning based on artificial neural networks propose useful security systems. However, they use lots of energy to ensure high performances and this point makes them unsuitable for IoT devices which use less energy. To address this issue, we propose a intrusion detection method using Spiking Neural Networks (SNNs). Using the proposed method, we detect malicious packets more accurately than the detection method using Deep Neural Networks (DNNs) with 15.08-40% reduction in power consumption. Furthermore, we propose the optimal network structure for the proposed method and analyze performance changes of the proposed method according to existence of a preprocessing scheme. The experimental results demonstrate SNNs-based detection method succeeds to detect malicious packets with high energy efficiency and we shows that our method can be a feasible security solution for defending IoT devices from network attacks.

※ 본 연구는 광운대학교 우수연구자 지원 사업 2023와 한국연구재단 신소자원천기술개발사업 (grant no. NRF-2021M3F3A2A01037962), 한국연구재단 기본연구사업 (grant no. RS-2023-00241628), 한국연구재단 기본연구사업 (grant no. 2021R1F1A1054915) 의 지원을 받아 수행된 연구입니다.

◆ First Author : Kwangwoon University Department of Electronic Convergence Engineering, jylemon1128@kw.ac.kr, 학생회원
° Corresponding Author : Kwangwoon University University Department of Sofatware, ljhar@kw.ac.kr, 정회원
* Kwangwoon University Department of Software, hetzer44@kw.ac.kr, 학생회원
논문번호 : 202311-127-C-RN, Received November 2, 2023; Revised February 14, 2024; Accepted February 26, 2024

Ⅰ. 서 론

사물 인터넷(internet of things, IoT) 기기의 급속한 상용화로 초연결 사회가 도래하였다. 스마트워치, IP CCTV, 등과 같은 IoT 기기는 우리 삶에 널리 퍼져 일상을 많이 변화시키고 있다. 많은 IoT 기기는 사용자의 편의를 위해 소형화가 이루어지면서, 저전력과 저성능의 구조로 설계되었다. 이러한 한계로 침입 탐지 시스템과 같은 네트워크 보안에 할당할 수 있는 자원이 제한적이다. 그러나 IoT 기기를 활용 및 기반으로 하는 연구에 비하여 침입 탐지와 같이 IoT 기기의 보안에 대하여 다룬 연구는 무척이나 부족하다[1,2]. 이러한 부족은 IoT 기기에 심각한 보안 문제를 야기할 수 있다.

많은 IoT 기기는 무선상으로 연결되어 있어 도청, 재밍(jamming), 스푸핑(spoofing)과 같은 공격에 취약하다. 이러한 보안 문제를 해결하기 위해 물리 계층 보안(physical layer security, PLS) 방법에 관한 연구가 진행되었다[29]. PLS는 적은 자원으로 높은 보안 성능을 달성할 수 있으므로, 자원이 한정된 IoT 기기에서 사용하기 적합하다[30]. 그러나 PLS는 OSI 상위 계층에서 이루어지는 서비스 거부 공격(denial of service, DoS), 분산 서비스 거부 공격(distributed denial of service, DDoS)과 같은 공격을 막는 것에 한계를 가진다.

이처럼 상위 계층을 타겟으로 하는 공격을 막기 위해 침입 탐지 시스템(intrusion detection system, IDS), 방화벽(firewall)과 같은 IoT 기기에 대한 보안 솔루션을 개발하려는 시도가 많이 이루어졌다[3]. 기존의 정적 사전 규칙 기반 및 알고리즘 방식의 IDS와 방화벽은 알려진 공격에 대해서는 높은 보안성을 보였다. 그러나 규칙과 패턴 분석에 기반을 두었기 때문에 이전에 보고된 적 없는 새로운 공격(zero-day attack)을 탐지하기 어렵다. 그래서 딥러닝(deep learning, DL)과 신경망(neural network, NN)을 활용한 탐지 방법이 대두되었다.

최근 NN 기반의 IDS 연구에 따르면, NN-IDS는 높은 침입 탐지 성능을 보였다. 이러한 방식은 DoS, DDoS 및 Mirai Botnet과 같은 다양한 유형의 네트워크 공격을 정확하게 비교 탐지하여 IoT 기기의 보안 문제를 효과적으로 해결할 수 있다[4]. 그러나 작동을 위해 많은 에너지를 요구하므로 가용 자원이 제한된 IoT 기기에서 사용하기 어렵다.

따라서 본 논문에서는 다른 신경망보다 에너지 소모량이 적은 스파이킹 신경망(spiking neural network, SNN) 기반의 침입 탐지 방법을 제안한다. SNN은 스파이크를 통해 계층 간에 정보를 전달하는 신경망이다. 여러 연구에 따르면 SNN은 심층 신경망(deep neural network, DNN), 순환 신경망(recurrent neural network, RNN), 합성곱 신경망(convolution neural network, CNN)과 같은 기존의 신경망보다 훨씬 적은 에너지를 소비할 뿐만이 아니라 다른 신경망만큼 높은 성능을 보인다[5-7]. 본 논문에서 설계한 SNN-IDS는 DNN 기반 방법보다 적은 에너지를 사용하면서 더 높은 정확도로 악성패킷을 탐지할 수 있다. 본 논문에서의 의의는 다음과 같다.

- 다양한 네트워크 구조와 전처리 방식을 실험하여 침입 탐지를 위한 가장 효율적인 네트워크 구조를 제안한다.
- 제안한 방법이 침입 탐지에서 DNN보다 더 정확하고 효율적으로 악성 패킷을 탐지함을 보인다.
- PCA가 공격을 탐지하고 분류함에 있어서 SNN의 학습에 어떠한 영향을 미치는지 분석한다.

본 논문의 나머지 부분은 다음과 같이 구성된다. 2장에서는 침입 탐지 방법과 SNN의 최신 기술 동향을 소개한다. 3장에서는 SNN, 스파이크 뉴런 모델, SNN 학습 규칙, 등에 대하여 설명한다. 4장에서는 제안하는 모델의 전처리 과정 및 네트워크 구조에 대하여 설명한다. 5장에서는 최적화된 모델을 다양한 실험을 통해 도출하고 그 결과를 분석하며, 6장에서 결론을 맺는다.

Ⅱ. 기존 연구

2.1 일반적인 탐지 기법

알고리즘 기반의 IDS가 다수 제안되었다. Simple service discovery protocol (SSDP) DDoS 공격을 완화시키기 위한 연구에서는 접근 제어 목록(access control list), 반사형 접근 제어 목록(reflexive access control list) 그리고 방화벽으로 구성된 IDS를 제안하였다[8]. 기존의 방화벽을 개선한 연구에서는 적응형 방화벽 시스템(adaptive firewall system)을 제안하였다[9]. 제안된 방법은 domain name system (DNS)과의 연계를 통해 방화벽의 정책을 적응적으로 결정한다. 앞서 제안된 방법들은 기존의 IDS보다 효율적으로 작동하였다.

그러나 위와 같이 사전정의된 정적 알고리즘을 기반으로 하는 IDS는 더욱 복잡해지고 다양해지는 DoS나 DDoS와 같은 네트워크 공격을 탐지하기 어렵다[10]. 이러한 문제를 해결하기 위해 기계학습(machine learning, ML), 인공 신경망(artificial neural network, ANN) 그리고 엣지 컴퓨팅(edge computing)을 사용한 IDS가 활발히 연구되고 있다[11].

2.2 기계학습 기반 탐지 기법

다양한 기계학습 기법들을 기반으로 하는 ML-IDS가 연구되었다. 의사결정 나무(decision tree, DT)를 이용한 연구에서는 CIC-IDS 2017 데이터셋의 일반 패킷과 악성 패킷을 높은 정확도로 구분하였다[12]. 6LoWPAN 네트워크에서 웜홀(wormhole) 공격 탐지를 위한 연구에서는 K-평균 클러스터링 알고리즘(K-means clustering algorithm)을 사용한 방법이 가장 높은 성능을 보였다[33]. 서포트 벡터 머신(support vector machine, SVM)에 다른 기계학습 기법이나 이론을 동시에 사용한 하이브리드(hybrid) 방법들이 다수 제안되었다[31,32]. SVM에 증거 이론(evidence theory)을 결합한 방식과 SVM에 랜덤 포레스트(random forest)를 결합한 모델 모두 기존의 방법보다 향상된 성능과 낮은 오탐율을 보였다.

2.3 딥러닝 기반 탐지 기법

하드웨어의 발전과 학습에 필요한 대량의 데이터를 확보할 수 있게 되면서 딥러닝 기술이 부상하였다. 다층 퍼셉트론(multi-layer perceptron, MLP), 오토인코더(auto-encoder) 및 신경망과 같은 딥러닝 기술 기반의 다양한 DL-IDS가 연구되었다. K-최근접 이웃(k-nearest neighbors)이나 랜덤 포레스트 같은 기계학습 기반 IDS와 딥러닝 기반 IDS 간의 성능을 비교하는 연구가 진행되었다[13]. 그 결과 CIC-IDS 2019 데이터셋을 분석하였을 때, MLP를 사용한 모델이 가장 높은 정확도를 보였다. 방사 기저 함수(radial basis function) 기반의 오토인코더로 UNSW-NB15 데이터셋의 악성 패킷을 탐지한 연구에서는 제안한 오토인코더를 사용한 IDS가 SVM과 MLP를 사용한 IDS보다 높은 탐지 성능을 보였다[34]. 그리고 딥러닝 기술을 엣지 컴퓨팅을 접목한 IDS도 제안되었다[14]. 제안된 방법은 CIC-IDS 2017 데이터셋을 사용하였을 경우와 실제로 공격을 가하였을 경우 모두에서 높은 이상 탐지 성능을 보였다.

2.4 신경망 기반 탐지 기법

최근에는 신경망을 기반으로 하는 NN-IDS가 활발히 연구되고 있다. 전처리 과정에 주성분 분석(principal component analysis, PCA)을 사용한 DNN 기반 IDS가 제안되었으며, 적은 정확도 감소로 효과적인 연산량 및 전력 소모 감소를 이루었다[15]. DNN 외에도 CNN, RNN 그리고 장단기 메모리(long short-term memory, LSTM)와 같이 다른 신경망을 사용한 연구도 진행되었다[10][16]. CNN을 기반으로 하는 ResNet을 사용한 연구에서는 CIC-IDS 2019 데이터셋을 사용하여, 악성 패킷 탐지와 패킷 클래스 분류를 진행하였으며 모두 높은 탐지 및 분류 성능을 보였다[10]. LSTM을 사용한 연구에서는 라즈베리파이에서 message queuing telemetry transport (MQTT)를 기반으로 하는 IoT 환경에서 모델을 구축하였다[16]. 제안된 모델은 MQTTset 데이터셋과 자체적으로 생성된 데이터셋 모두에서 높은 정확도를 보였다.

여러 신경망을 하이브리드 구조로 설계한 연구도 다수 진행되었다. 시계열 합성곱 네트워크(temporal convolution network, TCN)과 게이트 순환 유닛(gated recurrent unit, GRU)를 하이브리드로 설계한 연구에서는 기존의 기계학습 기반 IDS와 TCN이나 GRU만을 사용한 IDS에 비하여 성능 향상을 이루었다[35]. CNN, LSTM과 TCN을 하이브리드로 설계한 연구에서는 YARS-IDS와 YARS-IDS-II를 제안하였으며, MLP 및 CNN 기반의 IDS에 비해 높은 정확도를 보였다[36].

2.5 SNN 응용 사례

SNN은 다양한 분야에서 활발히 연구되고 있다. Tor 브라우저 및 VPN을 통해서 암호화된 패킷을 SNN으로 구분하는 연구가 진행되었다[17]. 제안된 방법은 암호화되기 전 패킷과 암호화된 이후 패킷 모두 높은 정확도로 분류하였다. 또한 SNN이 저전력으로 작동하는 특징에 주목하여, SNN을 하드웨어로 구현하는 연구도 활발히 진행되고 있다[18,19]. 앞선 연구들에서는 SNN을 하드웨어로 설계하여 소프트웨어로 구동시키는 것보다 매우 높은 에너지 소모 감소를 이루었다.

Ⅲ. 배경지식

3.1 Spiking Neural Network

SNN은 스파이크 뉴런과 시냅스로 구성된 신경망이다. 스파이킹 뉴런은 실제 신경 세포를 모방한 뉴런으로 '스파이크'라는 이진 신호를 생성한다. 생성된 스파이크는 시냅스를 통해 다음 뉴런으로 전달되며, 다음 뉴런은 위의 과정을 반복한다. 이러한 과정을 통해 SNN은 기존 신경망보다 생물학적 특성을 더 많이 반영한다.

스파이킹 뉴런의 작동 방식은 그림 1과 같다. 시냅스를 통해 스파이크가 스파이킹 뉴런으로 들어오면 그에 비례하여 막전위를 상승시킨다. 막전위가 임계치에 도달하면 스파이크를 발화하여 시냅스를 통해 다음 뉴런으로 전달한다. 이와 동시에 스파이킹 뉴런은 막전위를 초기화한다. 스파이킹 뉴런은 뉴런에 입력이 들어올 때마다 연산을 수행하는 ANN과는 다르게, 특정한 상태에서만 작동한다. 이러한 event-driven 특징으로 인하

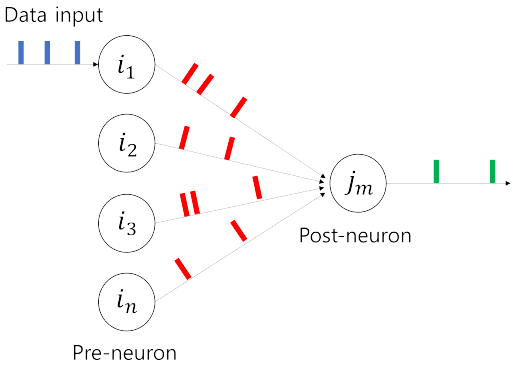


그림 1. 스파이킹 신경망의 작동 방식
Fig. 1. Mechanism of spiking neural networks

여 SNN은 낮은 전력으로 작동한다는 특징을 가진다.

시냅스는 스파이크를 전달하는 역할 뿐만이 아니라 SNN의 학습에도 중요한 역할을 한다. 시냅스는 시냅스 가중치를 통해 스파이크의 강도 및 빈도를 조절한다. 이때, 스파이크는 스파이킹 뉴런의 출력과 가중치의 곱으로 계산되며, 시냅스 가중치는 훈련을 통해 최적화된 값으로 수렴된다.

SNN은 다른 NN 모델과 달리 활성화 함수가 존재하지 않는다. 스파이킹 뉴런의 막전위 모델이 활성화 함수 역할을 하기 때문이다. Sigmoid, tanh, ReLU, Softmax 등과 같은 활성화 함수의 주요 목적은 선형 데이터를 비선형 데이터로 변환하는 것이다. 이를 통해 신경망은 입력과 출력 간의 복잡한 관계를 표현함으로써 다양한 패턴과 특징을 더 잘 학습할 수 있게 한다. 그러나 SNN은 스파이크와 막전위로 인하여 선형 데이터가 비선형 데이터로 변환되므로 활성화 함수가 필요하지 않다.

3.2 뉴런 모델

Integrate-and-fire (IF) 뉴런은 이전 뉴런으로부터 입력 스파이크를 받을 때마다 막전위가 증가하고 막전위 값이 유지된다.

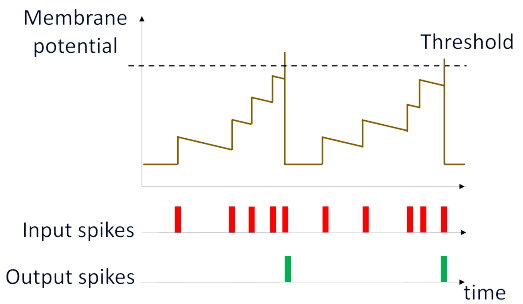


그림 2. LIF 뉴런의 작동 방식
Fig. 2. Mechanism of LIF neurons

반면에 leaky integrate-and-fire (LIF) 뉴런은 IF 뉴런의 막전위에 누수(leakage)가 추가된 모델로, 그림 2와 같이 막전위가 시간이 지남에 따라 점차 감소한다. 이러한 메커니즘은 잘못된 입력을 잊음으로써 잡음 저항성을 높인다. LIF 뉴런에 대한 수식은 다음과 같다[5].

$$\tau\frac{dU_m(t)}{dt} = -U_m(t) + RI_{in}(t) \quad (1)$$

(1)에서 U_m는 막전위, R은 시냅스 저항 상수, $I_{in}(t)$는 총 입력 전류, τ는 막 시간 상수이다. 시간에 따른 막전위의 변화율 $dU_m(t)/dt$는 이전 뉴런의 입력 $RI_{in}(t)$에 누수 $-U_m(t)$ 만큼 감산하고 τ로 나누어 구한다.

3.3 기울기 기반 학습

본 논문에서는 학습 규칙으로 확률적 경사 하강법(stochastic gradient descent, SGD)을 사용한다. 경사 하강법(gradient descent, GD)은 전체 데이터에 대해서 손실 함수의 기울기를 계산하기 때문에 상당한 연산량이 필요하다. 반면에 SGD는 mini-batch를 사용하여 전체 데이터가 아닌 정의한 mini-batch만큼 손실 함수의 기울기를 계산하기 때문에 GD에 비해 상대적으로 빠른 학습이 이루어진다.

그러나 SNN은 다른 NN 모델과 달리 델타 함수의 형태를 가지는 스파이크를 사용하기 때문에 기존 역전파 알고리즘을 사용하기 어렵다[20]. 이러한 문제를 해결하기 위해 대리 기울기(surrogate gradient)를 사용한다. 대리 기울기는 미분이 불가능한 함수의 근사함수를 구하고 이를 미분한 값이다. 본 논문에서는 스파이크의 근사 함수로 fast sigmoid를 사용하여 대리 기울기를 구하고 역전파를 계산한다.

3.4 Principal Component Analysis

PCA는 차원 축소 기법 가운데 하나로 대량의 특성을 포함하고 있는 고차원의 데이터 집합을 분석할 때 유용하다. PCA는 주성분이라고 하는 데이터의 구조를 잘 표현할 수 있는 벡터를 찾아 새로운 축으로 사용한다. 주성분을 기준으로 하여 고차원 데이터를 저차원 데이터로 변환시킴으로써 PCA는 정확도 향상, 전력 소비 및 학습 시간 감소라는 주요 이점을 가진다[15,21,22]. PCA는 원본 데이터의 특성을 최대한 유지하면서 차원을 줄임으로써 불필요한 정보로 인하여 신경망이 잘못 학습되는 경우를 줄여 성능을 증가시킨다. 또한 데이터의 차원이 감소하였기 때문에 학습에 필요한 뉴런의 수도 감소하여 시간 효율성이 향상된다.

3.5 데이터셋

네트워크 공격을 효율적으로 탐지하고 예방하기 위해서는 뛰어난 알고리즘과 기술은 물론 잘 설계된 데이터셋이 중요하다. 본 논문에서는 최신 DDoS 데이터셋인 CIC-IDS-2019를 사용한다. 데이터셋은 표 1과 같이 12개의 악성 패킷과 일반 패킷을 제공하며, 클래스별 데이터의 수는 그림 3과 같다[23]. 또한 데이터셋은 timestamp, protocol source IP/port, destination IP/port 등과 같이 패킷에 포함되어 있는 네트워크 트레픽 데이터를 CICFlowMeter-V3로 가공하여, 표 2와 같이 80개 이상의 특징을 가지는 메타 데이터(meta data)로 제공

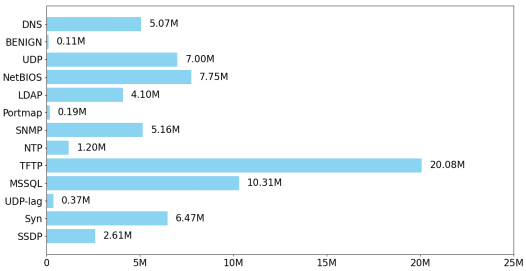


그림 3. 클래스별 데이터 분포
Fig. 3. Number of data per class

표 1. 데이터셋에 포함된 공격 패킷
Table 1. Attack packets included in the dataset

DDoS	Based	Attack Name
Reflection Attack	TCP	MSSQL
		SSDP
	TCP/UDP	DNS
		LDAP
		Net BIOS
		SNMP
		Port Map
	UDP	NTP
		TFTP
Exploitation Attack	TCP	SYN Flood
	UDP	UDP Flood
		UDP-Lag

표 2. 데이터셋의 특징 구성
Table 2. Dataset feature composition

Num	Feature Name	Feature Type
F0	Flow ID	continuous
F1-F4	Source (IP/Port), Destination (IP/Port)	categorical
F5-F6	Protocol, Timestamp	categorical
F7	Flow Duration	continuous
F8-F9	Total (Fwd/Bwd) Packets	continuous
F10-F11	Total Length of (Fwd/Bwd) Packets	continuous
F12-F19	(Fwd/Bwd) Packet Length (Max/Min/Mean/Std)	continuous
F20-F21	Flow (Bytes/s / Packets/s)	continuous
F22-F25	Flow IAT (Mean/Std/Max/Min)	continuous
F26-F35	(Fwd/Bwd) IAT (Total/Mean/Std/Max/Min)	continuous
F36-F37	(Fwd/Bwd) PSH Flags	continuous
F38-F39	(Fwd.Bwd) URG Flags	continuous
F40-F41	(Fwd/Bwd) Header Length	continuous
F42-F43	(Fwd/Bwd) Packets/s	continuous
F44-F45	(Min/Max) Packet Length	continuous
F46-F48	Packet Length (Mean/Std/Variance)	continuous
F49-F56	(FIN/SYN/RST/PSH/ACK/URG/CWE/ECE) Flag Count	continuous
F57-F58	Down/Up Ratio, Avg Packet Size	continuous
F59-F60	Avg (Fwd/Bwd) Segment Size	continuous
F61	Fwd Header Length.1	continuous
F62-F67	(Fwd/Bwd) Avg (Bytes/Bulk / Packets/Bulk / Bulk Rate)	continuous
F68-F71	Subflow (Fwd/Bwd) (Packets/Bytes)	continuous
F72-F73	Init WIn Bytes (Fwd/Bwd)	continuous
F74-F75	Act Data Paket Fwd, Min Seg Size Fwd	continuous
F76-F79	Active (Mean/Std/Max/Min)	continuous
F80-F83	Idle (Mean/Std/Max/Min)	continuous
F84	Simillar HTTP	categorical
F85	Inbound	continuous

한다. 본 논문에서는 표 2와 같이 특징을 연속형(continuous)과 범주형(categorical) 두 가지로 분류한다. 연속형은 flow bytes/s, flow packets/s와 같이 시간에 따라 값이 변하는 특징들이며, 범주형은 IP, port, protocol과 같이 시간에 따라 값이 변하지 않는 특징들이다.

Ⅳ. 실험 방법

4.1 전처리

CIC-IDS-2019 데이터셋에는 이상치(outlier)와 inf 값이 포함되어 있어 데이터를 선별해야 한다. 이상치를 제거하기 위하여 Z-score를 사용한다. Z-score는 평균과 표준편차를 이용하여 특정 값이 평균으로부터 얼마나 멀리 분포되어 있는지 측정하는 방법이다.

$$z = \frac{x - \mu}{\sigma} \tag{2}$$

Z-score는 (2)와 같이 측정하고자 하는 값 x를 평균 μ만큼 감산하고 표준편차 σ로 나누어 구한다. 본 논문에서는 z가 3이상인 값을 이상치로 판단한다.

그림 3에서 볼 수 있듯이 데이터셋의 데이터 수가 매우 방대할 뿐만이 아니라 클래스별 데이터의 수가 매우 큰 차이가 있다. 따라서, 실험의 용이성과 정확성을 위해 서브 데이터셋을 구성한다. 각 악성 패킷 및 일반 패킷에서 z가 3 이하인 값으로 50k의 데이터(총 650k)를 추출한다. 표 2와 같이 데이터셋에서 제공되는 특징들은 패킷의 어떤 부분을 가공하였는지에 따라 최댓값과 최솟값이 상이하다. 이는 학습에 악영향을 미치기 때문에 min-max scaler(3)를 사용하여 값들을 [0,1]로 변환한다.

$$\min - \max = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{3}$$

4.2 신경망 구조

전체적인 네트워크의 구조는 입력 계층, 출력 계층, 그리고 은닉 계층으로 구성된다. 입력 계층의 노드 수는 라벨을 제외한 특징의 수이며, 출력 계층의 노드 수는 구별하고자 하는 클래스의 개수이다. 입력 계층, 출력 계층, 은닉 계층을 구성하는 모든 노드는 LIF 뉴런을 사용한다. 본 논문에서는 은닉층의 깊이와 노드를 변경하여 실험을 통해 가장 높은 정확도를 보이는 네트워크 구조를 얻는다.

최적화 함수는 adaptive moment estimation (Adam)을 사용한다. Adam은 메모리 요구 사항이 낮으므로 저사양 기기에서 사용하기에 적합하다. 또한, 크기가 방대하고 잡음이 있는 데이터에 대하여 높은 성능을 보이기 때문에[24] DDoS 문제를 해결하는 데 적합하다. Adam은 경사 하강법을 기반으로, momentum과 RMSProb을 합친 최적화 알고리즘이다. Momentum으로 관성을 부여하여 지역 최솟값에 수렴되는 것을 방지하고, RMSProp으로 이전 기울기의 제곱 값을 지수 이동 평균으로 사용해서 기울기를 보정한다.

손실 함수는 mean squared error (MSE)를 사용한다. MSE는 예측값과 실제 값 사이의 차이를 측정하기 위한 함수 중 하나이다. (4)는 MSE에 대한 수식으로, N은 데이터의 총 개수, y_i는 예측값, x_i는 실제 값이다. MSE는 각 데이터마다 예측값과 실제 값의 차를 제곱하며, 오차가 클 경우, 더 크게 증폭시켜 모델이 빠르게 최적화 되도록 한다.

$$Loss = \frac{1}{2}\sum_{i=1}^{N}(y_i - x_i)^2 \tag{4}$$

4.3 벤치마크 모델 설정

벤치마크 모델은 5계층으로 구성된 DNN을 사용한다[15]. 입력 계층, 3개의 은닉 계층, 출력 계층으로 이루어져 있으며, 각 계층은 완전 연결(fully connected, FC) 구조를 갖는다. 은닉 계층은 3계층 모두 64개의 노드로 구성된다. 벤치마크 모델의 파라미터는 SNN과 같이 에폭(epoch)은 1, 배치 크기(batch size)는 128로 하며, 함수는 다음 같이 구성한다. 각 은닉층의 활성화 함수는 ReLU로 한다. 출력층의 활성화 함수는 이진 분류에서는 Sigmoid 함수를, 다중 분류에서는 Softmax 함수를 사용한다. 최적화 함수는 Adam(lr=0.001)을 사용하며 손실 함수는 cross entropy loss를 사용한다.

Ⅴ. 실험 결과

5.1 실험 환경

데이터 전처리 및 모델 학습은 Ubuntu 20.04에서 NVIDIA GeForce GTX 1060을 사용한 환경에서 이루어졌으며, 파이썬 라이브러리 중 하나인 snnTorch를 사용하여 SNN을 구축한다.

SnnTorch는 SNN으로 기울기 기반 학습을 수행하기 위한 파이썬 패키지다. snnTorch는 널리 사용되는 딥러닝 프레임워크인 PyTorch를 기반으로 하기 때문에, PyTorch에서 제공하는 기능들을 쉽게 사용할 수 있어 SNN을 설계하기 용이하다.

5.2 실험 설계

본 논문에서는 이진 분류와 다중 분류로 나누어서 실험한다. 이진 분류에서는 악성 패킷을 탐지하는 반면, 다중 분류에서는 패킷 클래스를 각각 구분한다. 실험은 다음과 같은 순서로 진행된다. 첫째, 이진 분류와 다중 분류에서 가장 높은 정확도를 보이는 신경망 구조를 구한다. 둘째, 앞서 얻은 신경망 구조와 함께 전처리 구조에 PCA를 추가 적용하였을 때의 성능 변화를 확인한다. 마지막으로 PCA가 성능에 어떠한 영향을 미치는지에 대해 분석한다. 이때, 이진 분류 및 다중 분류의 파라미터는 다음과 같이 설정한다. 에폭은 1, 배치 크기는 128, 최적화 함수는 Adam, 손실 함수는 MSE를 사용한다. 학습과 테스트에 사용하는 데이터의 비율은 8:2로 학습에 520k, 테스트에 130k를 사용한다.

5.2.1 이진/다중 분류 실험

이진 분류와 다중 분류에서 할당한 라벨은 표 3과 같다. 이진 분류에서는 일반 패킷은 True(0)로, 악성 패킷들에 대해서는 False(1)로 할당한다. 다중 분류에서는 일반 패킷은 0, 악성 패킷은 1-12로 할당한다. 이진 분류에서는 은닉 계층을 사용하지 않거나 하나만 사용하였을 때 가장 높은 정확도를 보이는 노드 수를 구한다. 다중 분류에서는 은닉 계층의 깊이 및 노드 각각에 대하여 실험을 진행한다. 우선 가장 높은 정확도를 보이는 계층 깊이를 구한 후 이 결과를 바탕으로 계층을 고정하여 가장 높은 정확도를 보이는 노드 수를 구한다. 이진 분류와 다중 분류 모두 표 2의 연속적인 특징들을 모두 사용하며, 라벨을 포함하여 총 80개의 특징이 사용된다.

표 3. 클래스별 라벨 할당
Table 3. Label assignment per class

Type	Label	
	Binary	Multi
Benign	0	0
SYN Flood	1	1
DNS		2
NTP		3
TFTP		4
UDP-Lag		5
SNMP		6
SSDP		7
Port Map		8
LDAP		9
Net BIOS		10
MSSQL		11
UDP Flood		12

표 4. 혼동 행렬
Table 4. Confusion matrix

		Actual	
		Positive	Negative
Prediction	Positive	TP	FN
	Negative	FP	TN

5.2.2 PCA 이진/다중 분류 실험

위 실험에서는 PCA를 사용하여 각 클래스를 가장 잘 구분할 수 있는 특징 12개(flow duration, source port, total of backward packets 등)를 추출하여 데이터셋을 구성한다[15]. 이때 source port는 범주형 특징이므로 one-hot encoding (OHE)을 사용해 변한다. OHE는 구별하고자 하는 n개의 데이터에 대해서 n의 문자열을 생성한다. 각 데이터마다 index를 정하여 그 위치에 1을 할당하고 나머지 위치에는 0을 할당한다.

5.3 성능 평가

성능 평가 지표로 정확도, 정밀도, 재현율, F1-score 4가지를 사용한다. 전통적으로 기계학습과 딥러닝 분야에서는 분류 모델의 성능을 혼동 행렬 (confusion matrix) 상의 4개의 값을 이용하여 계산한다[16].

표 4에 대한 설명은 아래와 같다.

- TP: 정상 데이터를 정상으로 판단한 경우
- TN: 비정상 데이터를 비정상으로 판단한 경우
- FP: 비정상 데이터를 정상으로 판단한 경우
- FN: 정상 데이터를 비정상으로 판단한 경우

정확도(Accuracy)(5)는 학습 모델에서 나온 분류 결과와 실제 데이터의 종류가 얼마나 일치하였는지 나타내는 지표이다. 본 논문에서는 아래의 수식 결과에 100을 곱하여 백분율로 나타낸다.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

정밀도(Precision)(6)는 정상이라고 예측한 것들 중에서 실제로 정상이었던 것의 비율을 나타낸다. 정상이라고 예측한 결과인 TP와 FP의 합을 실제로 정상인 값인 TP에 나누어 구한다. 값이 1에 근접할수록 오경보율이 낮아져 시스템 효율성이 높아진다[25].

표 5. SNN과 DNN 모델의 이진 분류 성능 비교
Table 5. Performance comparison of SNNs and DNNs in binary classification

Model	Accuracy	Precision	Recall	F1-Score	Energy
SNN	**99.57%**	**0.9500**	**0.9969**	**0.9729**	**2268[J]**
DNN	99.52%	0.9460	0.9940	0.9694	3780[J]

$$Precision = \frac{TP}{TP+FP} \tag{6}$$

재현율(Recall)(7)은 정상 데이터 중에서 정상 데이터라고 제대로 예측한 것의 비율을 나타낸다. 실제로 정상인 값을 가지는 TP와 FN을 더하고, 이 값에 정상이라고 제대로 예측한 TP에 나누어 구한다. 값이 1에 근접할수록 비정상 데이터를 놓치는 비율이 줄어든다.

$$Recall = \frac{TP}{TP+FN} \tag{7}$$

F1-score(8)는 Precision과 Recall의 조화평균으로 구성된다. 정밀도와 재현율을 곱한 값에 정밀도와 재현율을 더한 값을 나누고 2를 곱하여 구한다. 데이터가 불균형적인 분포를 보일 경우, 높은 성능을 가지는 평가 지표이며 1에 가까울수록 높은 성능을 가진다.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{8}$$

마지막으로 전력 소모량(9)은 매초 훈련할 때의 전력량 W_t에 컴퓨터에서 기본적으로 소모하는 전력량 W_c을 감산한다. 그리고 총 학습 시간(l.t.)만큼 앞의 결과를 합산하여 구한다. 전력 사용량 측정은 NVIDIA에서 제공하는 GPU 모니터링 도구인 'nvidia-smi'를 사용한다[26].

$$E_{cost}[J] = \sum_{0}^{l.t.}(W_t - W_c) \tag{9}$$

5.4 성능 측정

5.4.1 이진 분류 실험

이진 분류 실험에서 은닉층은 은닉층이 존재하지 않는 경우와 하나만 있는 경우로 제한한다. 은닉층이 하나일 경우 노드의 수는 2, 4, 8, 16, 32 와 같이 2의 배수로 증가하여 실험을 진행한다.

그림 4는 각 노드별로 10회씩 진행한 결과를 Box and Whisker Plot으로 나타낸 것이다. Box and

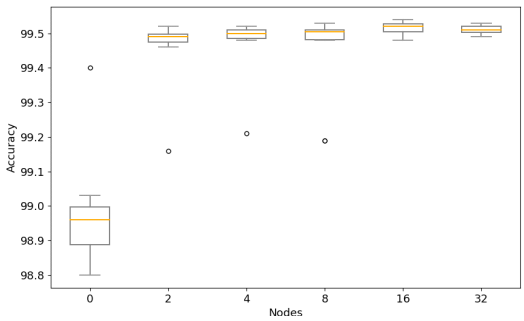


그림 4. 이진 분류에서의 정확도
Fig. 4. Accuracy for binary classification

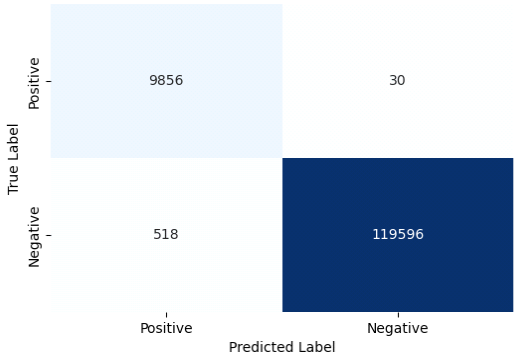


그림 5. 이진 분류에서의 혼동 행렬
Fig. 5. Confusion matrix in binary classification

Whisker Plot은 상자의 아랫단이 데이터의 25% (Q1), 윗단이 데이터의 75% (Q3), 상자 내부의 주황색 선이 데이터의 중앙값을 의미한다. 수염(whisker)은 상자 길이의 2.0배 거리의 데이터의 최대 최소이며, 수염 밖의 점은 이상치다. 그림 4를 보면 은닉층을 사용하지 않았을 경우 가장 낮은 정확도를 보이고 16개의 노드를 사용하였을 때 99.57%로 가장 높은 정확도를 보인다.

그림 5는 SNN에서 노드를 16개 사용하였을 때의 혼동 행렬이고, 표 5는 SNN과 DNN의 실험 결과를 앞에서 정의한 성능 지표로 나타낸 것이다. 표 5를 보면 SNN의 정확도는 99.57%로 DNN 모델보다 정확도가 0.05%p 높다. 또한 정밀도, 재현율, F1-score 모두 DNN보다 소폭 개선된 성능을 보인다. 특히 에너지 소모량이 DNN에 비하여 40.00% 감소하여 매우 우수한 성능을 보인다.

5.4.2 다중 분류 실험

다중 분류 실험에서는 우선 가장 높은 성능을 보이는 은닉 계층의 깊이를 측정한다. 이후 측정 결과를 바탕으로 노드의 개수에 대한 실험을 진행한다. 깊이에 대한 실험에서는 은닉 계층을 사용하지 않는 모델을 시작으로, 32개의 노드를 가지는 은닉 계층을 점차 추가한다.

그림 6은 계층 깊이에 대하여 각각 10회씩 실험한 결과이며, 은닉층이 2개가 있을 때 최고 정확도가 60.45%로 가장 높다. 위 실험을 근거로 은닉층이 2개일 때 최적의 노드 수에 대한 실험을 진행한다. 2개의 은닉층을 구성하는 노드의 수는 각각 동일하며, 노드를 32개부터 512까지 2의 배수로 증가하여 실험한다.

그림 7은 노드에 대하여 각각 10회씩 실험한 결과이다. 노드가 256개일 때 61.68%로 가장 높은 정확도를 보이며, 그 이후부터는 노드가 증가하여도 정확도가 하락한다.

표 6을 보면 SNN은 DNN에 비하여 정확도는 61.68%로 3.84%p 높고, 전력 소모는 12.84% 감소한다. 표 5와 표 6을 보면 SNN이 DNN에 비하여 높은 성능을 보인다. 이는 일반적으로 DNN이 SNN보다 뛰어난 성능을 보이는 것과는 다른 양상을 띤다[27].

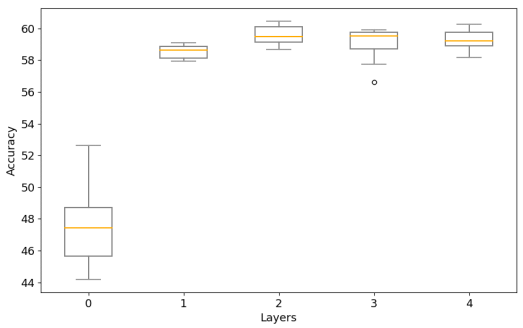


그림 6. 다중 분류에서 은닉 계층 깊이별 정확도
Fig. 6. Accuracy for multi-class classification (layers)

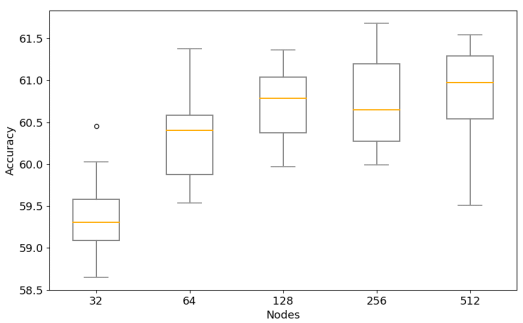


그림 7. 다중 분류에서 은닉 계층 노드 수별 정확도
Fig. 7. Accuracy for multi-class classification (nodes)

표 6. 다중 분류에서 SNN과 DNN의 성능 비교
Table 6. Performance comparison of SNNs and DNNs in multi-class classification

Model	Accuracy	Energy
SNN	**61.68%**	**3276[J]**
DNN	57.84%	3759[J]

이에 대한 첫 번째 이유로는 SNN이 시간적 데이터에 대하여 상대적으로 강점을 가지기 때문이다. SNN은 이진 스파이크로 연산 및 정보 전달을 이루는 생물학적 특징으로 인하여, 처리하는 데이터의 시간적 특징에 높은 민감성을 가진다[28]. DDoS 데이터는 표 2와 같이 많은 시간적 특징으로 인하여, SNN이 DNN보다 높은 성능을 가능하게 한다.

두 번째 이유로는 실제 탐지 방법과의 유사성이다. 실제 모니터링 방식은 트레픽을 실시간으로 감지하고 있다가 트레픽이 일정 수치를 넘어가면 DDoS 공격이 이루어지고 있다고 판단한다. 이는 SNN이 막전위와 임계값으로 악성 패킷을 탐지하는 방식과 매우 유사하므로 DNN보다 높은 성능을 가능하게 한다.

5.4.3 PCA 이진/다중 분류 실험

위 실험에서는 전처리 과정에 PCA를 추가하고, 이진 분류 실험과 다중 분류 실험을 통해서 구한 가장 높은 정확도를 가지는 네트워크 구조를 사용한다. 표 7은 실험을 각각 10회씩 진행하여 가장 높은 정확도를 보인 것이다. PCA를 사용하였을 때 SNN과 DNN 모두 전력 사용량이 소폭 감소한다. SNN에서 이진 분류를 진행할 때는 정확도가 2.27%p 감소하지만, 다중 분류에서는 정확도가 5.45%p 증가한다. 특히 PCA를 사용하지 않는 DNN과 PCA를 사용한 SNN을 비교하였을 때 전력 소모량은 15.08% 감소하고 정확도는 9.29%p 증가함을 보인다. PCA를 사용할 경우 일반 패킷의 특

표 7. PCA 유무에 따른 이진/다중 분류 성능 비교
Table 7. Comparison performance of SNNs and DNNs in PCA binary/multi-class classification

Model		Accuracy	Energy
PCA Binary	SNN	97.30%	2247[J]
	DNN	98.15%	3402[J]
Binary	**SNN**	**99.57%**	**2268[J]**
	DNN	99.52%	3780[J]
PCA Multy	**SNN**	**67.13%**	**3192[J]**
	DNN	64.31%	3423[J]
Multy	SNN	61.68%	3276[J]
	DNN	57.84%	3759[J]

징이 차원 축소의 영향으로 소실되어 이진 분류에서는 DNN과 SNN 모두 전체 정확도가 감소한다. 반면에 다중 분류에서는 이진 분류와 마찬가지로 일반 패킷의 분류 성능은 하락하였지만, 다른 클래스의 특징이 더욱 잘 분류되어서 전체 정확도가 증가한다. 이에 대한 세부적인 분석은 다음 항에서 다룬다.

5.4.4 Non-PCA 및 PCA 다중 분류 분석

PCA가 데이터의 차원을 바꿔주기 때문에, PCA 사용 여부에 따라 다중 분류 성능 차이가 발생한다. 그림 8은 PCA를 사용하지 않은 다중 분류와 PCA를 사용한 다중 분류에 대한 히트맵(heatmap)이다. 세로축은 실제 값, 가로축은 예측한 값이며, 축의 지정된 값은 표 3과 같다. 이때, 그림 8-a는 PCA를 사용하지 않았을 때의 다중 분류 결과이며, 그림 8-b는 PCA를 사용하였을 때의 다중 분류 결과이다.

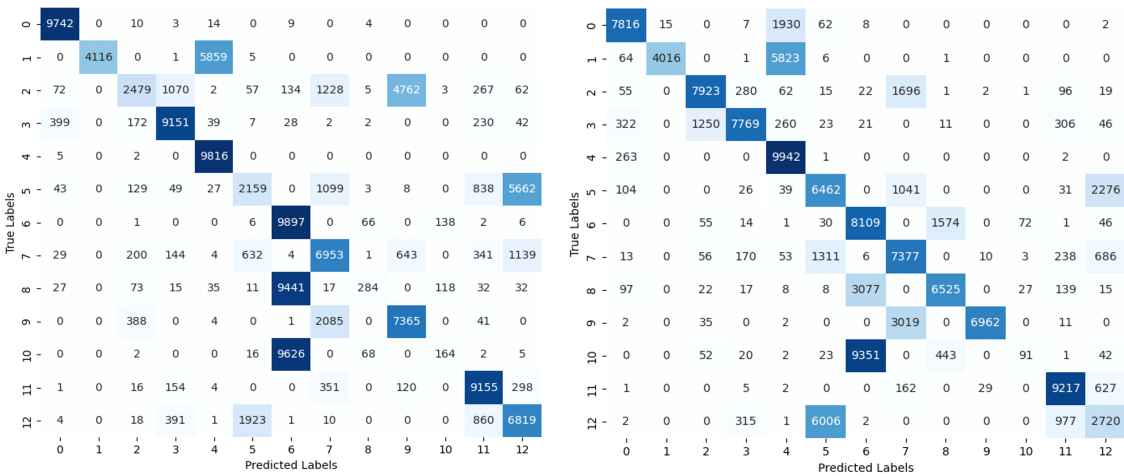
그림 8-a를 보면 PCA를 사용하지 않았을 때 SNMP(6번), Port Map(8번), Net BIOS(10번)의 분류가 이루어지지 않는다. 또한 DNS(2번)를 LDAP(9번)로, UDP-Lag(5번)를 UDP Flood(12번)로 오판단이 발생한다. 그림 8-b를 보면 PCA를 사용한 결과 Port Map에 대하여 상당한 분류 성능의 향상을 보인다. 또한 DNS(2번)를 LDAP(9번)로 오판단이 발생하는 비율과 UDP-Lag(5번)를 UDP Flood(12번)로 오판단이 발생하는 비율 모두 큰 폭으로 감소한다. 그러나 PCA를 사용하였을 때, 탐지 성능이 큰 향상을 보이지 않거나 소폭 하락하는 클래스도 존재한다.

그림 8-b를 보면 SYN FLood(1번)은 TFTP(4번)로, Net BIOS(10번)는 SNMP(6번)로, UDP Flood(12번)는 UDP-Lag(5번)으로 50% 이상의 오판단을 보였다. 그림 9는 PCA를 사용한 다중 분류에서 분류 정확도가 낮은 클래스들의 평균 특징값을 히트맵으로 나타낸 것이다. 그림 9를 보면 분류에 혼동이 발생하는 클래스는 특징값들이 유사한 것을 볼 수 있다. Benign(0번)은 PCA 사용 이전에는 매우 정확하게 구분하였으나 PCA 사용 후 약간의 성능 하락을 보인다. 그 이유는 PCA를 사용함으로써 많은 시간적 특징의 축소가 이루어졌고, 이는 구분 성능의 저하로 이어졌기 때문이다.

5.4.5 ML-IDS와의 비교 실험

위 실험에서는 앞선 실험을 통해 구한 최적의 SNN 모델을 기존의 ML-IDS에 주로 사용되는 의사결정 나무(DT)와 소프트 벡터 머신(SVM) 기반 IDS와 비교한다[37-39]. 각 모델에 대한 세부 설정은 다음과 같다. DT는 PCA를 사용하지 않는다. SVM은 결정 함수로 One-vs-Rest을 사용하며, PCA는 사용하지 않는다. DNN은 앞서 벤치마크 모델 설정 섹션에서 정의한 네트워크 구조를 사용한다. 이진 분류에서는 PCA를 적용하지 않고, 다중 분류에서는 PCA를 적용한다. SNN은 다음과 같이 설정한다.

- 이진 분류: 전처리 과정에 PCA를 사용하며, 16개 노드의 은닉 계층 한 개로 구성된 모델을 사용한다.
- 다중 분류: 전처리 과정에 PCA를 사용하며, 256개 노드의 은닉 계층 두 개로 구성된 모델을 사용한다.



a) Non-PCA b) PCA

그림 8. 다중 분류에서 PCA를 사용하였을 경우와 사용하지 않았을 경우의 혼동행렬 결과
Fig. 8. Multi-class classification confusion matrix results of Non-PCA and PCA.

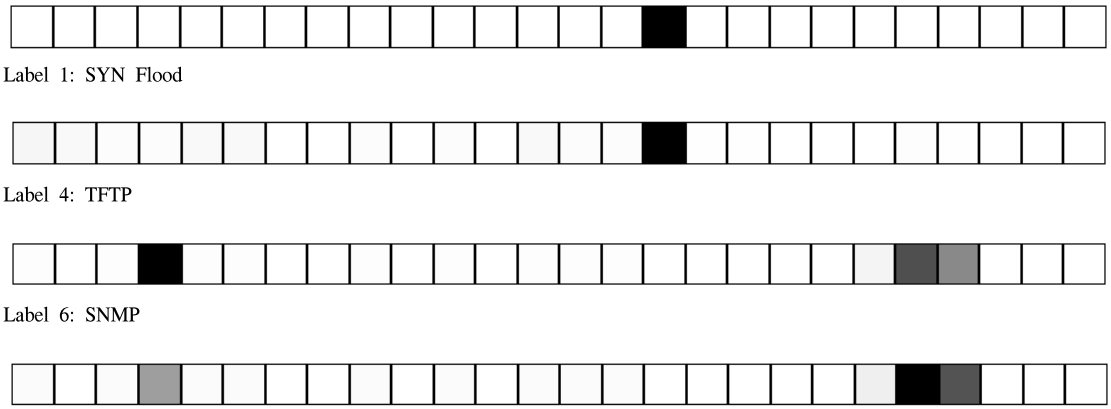


그림 9. PCA 다중 분류에서 낮은 탐지 성능을 보이는 클래스들의 평균 특징 값
Fig. 9. Mean feature value of classes with low detection accuracy in PCA multi-classification

이진 분류는 SNN과 같이 DT, SVM 모두 학습 데이터의 수는 520k로 하며, 테스트 데이터의 수는 130k로 한다. 다중 분류에서는 이진 분류와 같이 테스트 데이터의 수는 130k로 고정한다. 그러나 학습 데이터의 수는 0.5k씩 증가시킨다. 만일 DT와 SVM의 학습 시간이 SNN의 학습 시간의 2.5배를 초과하면 중단한다.

실험 결과 이진 분류에서는 그림 10과 같이 각 모델들은 큰 성능 차이 없이 높은 이상 탐지 능력을 보인다. 그러나 다중 분류에서는 그림 11과 같이 상당한 차이를 보인다. SVM은 58k의 데이터로 학습하였을 때, SNN의 학습 시간을 2.5배를 초과하여 중단된다. 최종 정확도는 57.49%로 다른 모델들에 비하여 가장 낮은 성능을 보인다. DT는 5.5k의 데이터로 학습하였을 때, 정확도가 64.38%로 64.31%인 DNN의 성능을 뛰어넘는다. 350k의 모든 데이터를 학습에 사용하였을 경우, 정확도는 66.00%로 SNN에 비하여 낮은 성능을 보인다.

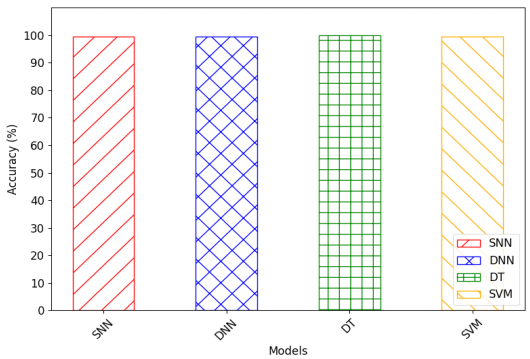


그림 10. 이진 분류에서 ML-IDS와의 비교
Fig. 10. Comparison performance of binary classification with ML-IDS

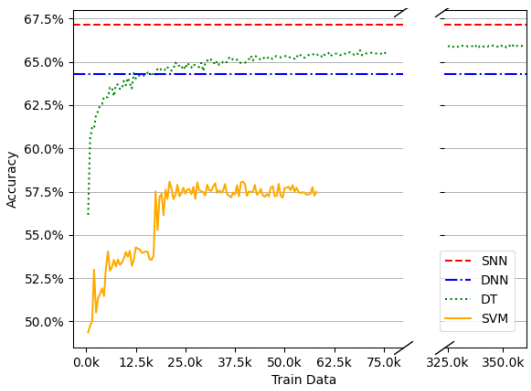


그림 11. 다중 분류에서 ML-IDS와의 비교
Fig. 11. Comparison performance of multi-class classification with ML-IDS

Ⅵ. 결 론

IoT 기기가 가지는 저전력 및 저성능 문제로 인하여 기존의 보안 솔루션을 IoT 기기에 적용하기 어렵다. 그래서 본 논문에서 저전력 특징을 가지는 SNN을 사용한 침입 탐지 시스템을 설계하였다. 설계한 침입 탐지 시스템은 이진 분류와 다중 분류 각각 다르게 전처리 과정 및 네트워크 구조를 구성하여 최적의 모델을 설계하였다.

이진 분류에서는 기존의 DNN에 비하여 정확도를 소폭 개선하였고 전력 소모를 40% 감소시켰다. 다중 분류에서는 DNN에 비하여 3.84%p 개선된 정확도 및 12.84% 전력 소모 감소를 이루었다. SNN에 PCA를 사용하였을 경우, 전력 소모는 이진 분류와 다중 분류 모두 소폭 감소하였다. 정확도는 이진 분류에서는 소폭 하락하였으나 다중 분류에서는 67.13%로 PCA를 사용

하지 않았을 경우보다 5.45%p 상승하였다. 특히 다중 분류에서 PCA를 사용하지 않은 DNN 구조와 비교하였을 때, 정확도는 9.29%p 상승하였고 전력 소모량은 15.08% 감소하였다. 기존의 기계학습 기반 IDS와 비교하였을 경우에도 이진 분류와 다중 분류 모두에서 우수한 성능을 보였다.

본 논문에서는 SNN이 DNN 및 다른 기계학습 기반 IDS보다 우수한 성능을 입증하여, 침입 탐지에서 충분히 SNN이 사용될 수 있음을 보였다. 또한 SNN이 DDoS를 탐지하고 각각의 클래스를 분류함에 있어서 필요한 개선 사항을 확인하였다. 향후 연구에서는 포괄적으로 네트워크 공격을 탐지 및 대응하는 시스템을 개발하고자 한다.

References

[1] J. Wu, Y. Wang, H. Dai, C. Xu, and K. B. Kent, "Adaptive bi-recommendation and self-improving network for heterogeneous domain adaptation-assisted IoT intrusion detection," *IEEE Internet of Things J.*, vol. 10, no. 15, pp. 13205-13220, 2023. (https://doi.org/10.1109/JIOT.2023.3262458)

[2] M. M. Alani, "IoTProtect: A machine-learning based IoT intrusion detection system," in *2022 6th Int. CSP*, pp. 61-65, 2022. (https://doi.org/10.1109/CSP55486.2022.00020)

[3] O. A. Mahdi, A. Alazab, S. Bevinakoppa, N. Ali, and A. Khraisat, "Enhancing IoT intrusion detection system performance with the diversity measure as a novel drift detection method," in *2023 9th Int. Conf. ITT,* pp. 50-54, 2023. (https://doi.org/10.1109/ITT59889.2023.10184268)

[4] S. Zheng, "Network intrusion detection model based on convolutional neural network," in *2021 IEEE 5th Advanced Inf. Technol., Electr. and Automat. Control Conf.(IAEAC)*, vol. 5, pp. 634-637, 2021. (https://doi.org/10.1109/IAEAC50856.2021.9390930)

[5] H.-J. Lee, D.-H. Kim, and J.-H. Lim, "Wi-Fi frame detection via spiking neural networks with memristive synapses," *Comput. Commun.*, vol. 208, pp. 256-270, 2023. (https://doi.org/10.1016/j.comcom.2023.06.006)

[6] H.-J. Lee, D.-H. Kim, D.-G. Kim, Y.-S. Lim, and J.-H. Lim, "Performance analysis of spiking neural networks with memristive synapse in detecting and classifying RF wave signals," in *2022 13th Int. Conf. ICTC*, pp. 303-308, 2022. (https://doi.org/10.1109/ICTC55196.2022.9952889)

[7] Y. Kim, H. Park, A. Moitra, A. Bhattacharjee, Y. Venkatesha, and P. Panda, "Rate coding or direct coding: Which one is better for accurate, robust, and energy-efficient spiking neural networks?," in *ICASSP 2022-2022 IEEE ICASSP*, pp. 71-75, 2022. (https://doi.org/10.48550/arXiv.2202.03133)

[8] B. A. Sassani, A. Palle, S. Dhakal, S. Bobuwala and A. David, "Analysis of SSDP DRDoS attack's performance effects and mitigation techniques," *2022 INCOFT*, pp. 1-5, Belgaum, India, 2022. (https://doi.org/10.1109/INCOFT55651.2022.10094381)

[9] Y. Jin and N. Yamai, "Proposal of an adaptive firewall system in collaboration with extended DNS," *2011 IEEE/IPSJ Int. Symp. Appl. and the Internet*, pp. 222-225, Munich, Germany, 2011. (https://doi.org/10.1109/SAINT.2011.40)

[10] F. Hussain, S. G. Abbas, M. Husnain, U. U. Fayyaz, F. Shahzad, and G. A. Shah, "IoT DoS and DDoS attack detection using ResNet," in *2020 IEEE 23rd INMIC*, pp. 1-6, 2020. (https://doi.org/10.1109/INMIC50486.2020.9318216)

[11] M. H. Rohit, S. M. Fahim, and A. H. A. Khan, "Mitigating and detecting DDoS attack on IoT environment," in *2019 IEEE Int. Conf. RAAICON*, pp. 5-8, 2019. (https://doi.org/10.1109/RAAICON48939.2019.5)

[12] S.-K. Youm and M.-W. Jin, "DDoS traffic analysis using decision tree according by

feature of traffic flow," *J. KIICE*, vol. 25, no. 1, pp. 69-74, 2021. (https://doi.org/10.6109/jkiice.2021.25.1.69)

[13] S. Mekala and K. B. Dasari, "NetBios DDoS attacks detection with machine learning classification algorithms," in *2023 InCACCT*, pp. 176-179, 2023. (https://doi.org/10.1109/InCACCT57535.2023.10141815)

[14] J. Kim and M. Choi, "Design and implementation of a deep learning-based intrusion detection system in edge computing," *J. KICS*, vol. 47, no. 8, pp. 1114-1127, 2022. (https://doi.org/10.7840/kics.2022.47.8.1114)

[15] M. Al-Fawa'reh, M. Al-Fayoumi, S. Nashwan, and S. Fraihat, "Cyber threat intelligence using PCA-DNN model to detect abnormal network behavior," *Egyptian Inform. J.*, vol. 23, no. 2, pp. 173-185, 2022. (https://doi.org/10.1016/j.eij.2021.12.001)

[16] S. C. Deun Lee and S. S. Im, "Malicious traffic detection method using LSTM and sliding window in MQTT based IoT environment," *J. Korean Inst. Inf. Technol.*, vol. 21, no. 5, pp. 111-120, 2023. (http://dx.doi.org/10.14801/jkiit.2023.21.5.111)

[17] A. Rasteh, F. Delpech, C. Aguilar-Melchor, R. Zimmer, S. B. Shouraki, and T. Masquelier, "Encrypted internet traffic classification using a supervised spiking neural network," *Neurocomputing*, vol. 503, pp. 272-282, 2022. (https://doi.org/10.1016/j.neucom.2022.06.055)

[18] J. Yang and T. Song, "A prediction scheme in spiking neural network (SNN) hardware for ultra-low power consumption," in *2020 ISOCC*, pp. 310-311, 2020. (https://doi.org/10.1109/ISOCC50952.2020.9333106)

[19] G. W. Kam, B. Jeong, D.-H. Youn, M. Jin, and S. Y. Kim, "Spike-predictable neuron circuits with adaptive threshold for low-power SNN systems," in *2023 IEEE ISCAS*, pp. 1-5, 2023. (https://doi.org/10.1109/ISCAS46773.2023.10181408)

[20] H. Jang, O. Simeone, B. Gardner, and A. Gruning, "An introduction to probabilistic spiking neural networks: Probabilistic models, learning rules, and applications," *IEEE Signal Proc. Mag.*, vol. 36, pp. 64-77, 2019. (https://doi.org/10.1109/MSP.2019.2935234)

[21] D.-H. Jeon and S.-J. Lee, "Light-weight classification model for android malware through the dimensional reduction of API call sequence using PCA," *J. Korea Soc. Comput. and Inform.*, vol. 27, no. 11, pp. 123-130, 2022. (http://doi.org/10.9708/jksci.2022.27.11.123)

[22] S. Salaria, S. Arora, N. Goyal, P. Goyal, and S. Sharma, "Implementation and analysis of an improved PCA technique for DDoS detection," in *2020 IEEE 5th ICCCA*, pp. 280-285, 2020. (https://doi.org/10.1109/ICCCA49541.2020.9250912)

[23] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *2019 ICCST*, pp. 1-8, 2019. (https://doi.org/10.1109/CCST.2019.8888419)

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2017. (https://doi.org/10.48550/arXiv.1412.6980)

[25] M. Kwon and H. Kye, "PCA-based low-complexity anomaly detection," *J. KICS*, vol. 46, no. 6, pp. 941-955, 2021. (https://doi.org/10.7840/kics.2021.46.6.941)

[26] E. Cai, D.-C. Juan, D. Stamoulis, and D. Marculescu, "NeuralPower: Predict and deploy energy-efficient convolutional neural networks," in *Asian Conf. Mach. Learn.*, pp. 622-637, 2017. (https://doi.org/10.48550/arXiv.1710.05420)

[27] S. Liu and Y. Yi, "Knowledge distillation between DNN and SNN for intelligent sensing systems on loihi chip," in *2023 24th ISQED*, pp. 1-8, 2023. (https://doi.org/10.1109/ISQED57927.2023.101

29306)

[28] A. Marchisio, G. Nanfa, F. Khalid, M. A. Hanif, M. Martina, and M. Shafique, "Is spiking secure? A comparative study on the security vulnerabilities of spiking and deep neural networks," in *2020 IJCNN*, pp. 1-8, 2020. (https://doi.org/10.1109/IJCNN48605.2020.9207297)

[29] P. Rojas, S. Alahmadi, and M. Bayoumi, "Physical layer security for IoT communications - A survey," *2021 IEEE 7th WF-IoT*, pp. 95-100, New Orleans, LA, USA, 2021. (https://doi.org/10.1109/WF-IoT51360.2021.9595025)

[30] N. Wang, P. Wang, A. Alipour-Fanid, L. Jiao, and K. Zeng, "Physical-layer security of 5G wireless networks for IoT: Challenges and opportunities," in *IEEE Internet of Things J.*, vol. 6, no. 5, pp. 8169-8181, 2019. (https://doi.org/10.1109/JIOT.2019.2927379)

[31] S. Raj, K. N. Singh, N. K. Gupta, R. Nigam, B. Verma, and S. Karsoliya, "High accuracy of hybrid IDS system using evidence theory and SVM ML technique," *2021 ICAIS*, pp. 1261-1264, Coimbatore, India, 2021. (https://doi.org/10.1109/ICAIS50930.2021.9396054)

[32] S. R. Deshmukh and C. Mankar, "ML hybrid approach for intrusion detection in networks," *2022 IEEE IAS GlobConET*, pp. 924-929, Arad, Romania, 2022. (https://doi.org/10.1109/GlobConET53749.2022.9872392)

[33] P. Shukla, "ML-IDS: A machine learning approach to detect wormhole attacks in Internet of Things," *2017 IntelliSys*, pp. 234-240, London, UK, 2017. (https://doi.org/10.1109/IntelliSys.2017.8324298)

[34] Y. Kadhim and A. Mishra, "Radial basis function (RBF) based on multistage autoencoders for intrusion detection system (IDS)," *2019 1st Int. Inf. and Softw. Eng. Conf. (UBMYK)*, pp. 1-4, Ankara, Turkey, 2019. (https://doi.org/10.1109/UBMYK48245.2019.8965627)

[35] B. Cao, C. Li, J. Sun, and Y. Song, "IoT intrusion detection technology based on Deep learning," *2022 3rd Int. Conf. CVIDL & ICCEA*, pp. 284-289, Changchun, China, 2022. (https://doi.org/10.1109/CVIDLICCEA56201.2022.9825291)

[36] Y. Madwanna, B. Annappa, A. R. Rashmi, and H. R. Sneha, "YARS-IDS: A novel IDS for multi-class classification," *2023 IEEE 8th I2CT*, pp. 1-6, Lonavla, India, 2023. (https://doi.org/10.1109/I2CT57861.2023.10126301)

[37] M. V. Kotpalliwar and R. Wajgi, "Classification of attacks using support vector machine (SVM) on KDDCUP'99 IDS database," *2015 Fifth Int. Conf. Commun. Syst. and Netw. Technol.*, pp. 987-990, Gwalior, India, 2015. (https://doi.org/10.1109/CSNT.2015.185)

[38] M. Kumar, M. Hanumanthappa, and T. V. S. Kumar, "Intrusion detection system using decision tree algorithm," *2012 IEEE 14th Int. Conf. Commun. Technol.*, pp. 629-634, Chengdu, China, 2012. (https://doi.org/10.1109/ICCT.2012.6511281)

[39] R. Panigrahi, S. Borah, A. K. Bhoi, M. F. Ijaz, M. Pramanik, Y. Kumar, and R. H. Jhaveri, "A Consolidated decision tree-based intrusion detection system for binary and multiclass imbalanced datasets," *Mathematics*, vol. 9, no. 7, 751, 2021. (https://doi.org/10.3390/math9070751)

허 정 윤 (Jeong-Yun Heo)

2028년 3월 : 광운대학교 전자융합공학과 졸업예정

<관심분야> 전자공학, 정보보안, 기계학습, 뉴럴 네트워크

[ORCID:0009-0007-9629-980X]

이 현 종 (Hyun-Jong Lee)

2024년 2월 : 광운대학교 소프트웨어학부 공학사

2026년 8월 : 광운대학교 컴퓨터과학과 졸업예정

<관심분야> 무선 신호 처리, 기계학습, 뉴럴 네트워크

[ORCID:0000-0001-8493-1002]

임 재 한 (Jae-Han Lim)

2004년 2월 : 서울대학교 전기공학부 공학사

2006년 2월 : 서울대학교 전기컴퓨터공학부 공학석사

2014년 12월 : UCLA Electrical Engineering 공학박사 한국전자통신연구원 선임연구원

광운대학교 소프트웨어학부 부교수 (현)

<관심분야> 스파이킹 뉴럴 네트워크, 차량용 네트워크, 자율주행, 모바일 컴퓨팅, 기계학습, 백스캐터 통신