Research Article

pISSN : 2287-8920
https://doi.org/10.22716/sckt.2021.9.3.037

무선 네트워크 침입탐지를 위해 개선된 CNN 분석

박승균[1]
[1]신한대학교 IT 융합공학부 교수

Analysis of Improved CNN for Wireless Network Intrusion Detection

Seungkyun Park[1]
[1]Professor, School of IT Convergence Engineering, Shinhan University

[1]**Corresponding author:** skpark@shinhan.ac.kr

Received August 17, 2021; **Revised** September 15, 2021; **Accepted** September 16, 2021

ABSTRACT

최근 5G기술과 함께 공중 및 사설 WiFi 서비스 영역이 크게 확대되면서 사용자 트래픽의 종류와 크기도 폭발적으로 증가하고 있다. 이와 함께 무선 네트워크의 보안 취약성을 이용한 인가되지 않은 악의적인 사용자의 침입/공격 트래픽도 크게 증가하고 있다. 침입/공격 특성 또한 다양화되고 있어 기존 무선 네트워크 침입 탐지 시스템은 오탐률이 높고 탐지 효율성이 낮으며 침입 및 공격 트래픽에 대한 일반화 능력이 약하다. 본 논문에서는 과대적합 문제를 피하면서 일반화 능력을 개선하기 위한 방안으로 CNN의 커널 크기를 축소하고 콘볼루션 계층을 이중화하여 병렬 연산을 하는 구조를 제안한다. 테스트 데이터 세트로NSL-KDD CUP 데이터 세트를 사용하여, 실험 및 분석 결과 제안한 CNN은 침입/공격을 탐지하기 위한 샘플 테스트 수행에서 정확도와 참양성률(true positive rate)은 96.38%, 96.75%이며 이것은 기존 DBN과 RNN보다 2%이상 향상된 결과이다. 또한 위양성율(false positive rate)은 0.88%와 0.91% 보다 낮은 0.64%을 보여주었다.

Recently, along with 5G technology public and private WiFi service areas have been greatly expanded. Also, the types and sizes of user traffic are increasing explosively. At the same time, the frequency of intrusion/attack by unauthorized malicious users using security vulnerabilities of wireless networks is also increasing significantly. Intrusion/attack characteristics are also diversifying, so the existing wireless network intrusion detection system has a high false positive rate, low detection efficiency, and weak generalization ability for intrusion and attack traffic. In this paper, as a method to improve generalization ability while avoiding the overfitting problem, we propose a structure that reduces the size of the CNN kernel and duplicates the convolutional layer for parallel operation. The NSL-KDD CUP data set was used as the test data set. As a result of experiments and analysis, the proposed CNN show 96.38% and 96.75% accuracy and true positive rates in performing sample tests to detect intrusion/attack.This showed an improvement of more than 2% compared to the existing DBN and RNN. Also, the false positive rate was 0.64%, lower than 0.88% and 0.91%.

Keywords: Wireless network, Intrusion detection, Convolution neural network, Detection accuracy, True positive rate, False positive rate

1. 서 론

최근 5G 기술의 등장과 함께 무선 네트워크의 트래픽도 폭발적으로 증가하고 있다. 이와 함께 무선 네트워크의 보안 취약성을 노린 새로운 유형의 무선 네트워크 공격도 증가하고 있다. 침입 탐지 시스템(IDS:Intrusion Detection System)은 현대 보안 시스템에서 매우 중요한 역할을 하고 있다. 침입/공격 트래픽은 반드시 이 시스템을 경유하여 네트워크 내부로 유입되도록 하여 사전에 침입/공격을 차단하기 위해 네트워크 최전방에 배치된다. 특히 새로운 유형의 공격 트래픽은 높은 적응성과 안정성 그리고 효율성을 갖는 침입 탐지 시스템이 필요하다. 현재 대부분의 무선 네트워크 인증 메커니즘과 방화벽 시스템은 사용자들의 기본적인 보안 요구 사항들을 만족시키고 있으나 사용자들을 안전하게 보호하는 기능은 상대적으로 부족하다. 만약 악의적인 전문 해커의 공격이 발생 한다면 기존 침입 탐지 시스템은 만족스러운 효과를 보여 주지 못할 것이다. 현재 일반적으로 사용되고 있는 비정상 탐지(abnormal detection)와 오용 탐지(misuse detection) 기술은 낮은 검출 정확도와 특성 추출 효율 그리고 높은 위양성률(false positive rate)을 가지고 있다[1]. 최근 네트워크 침입/공격 같은 트래픽을 분석하고 인식하는 대부분의 연구들은 컴퓨터의 성능이 크게 향상됨에 따라 주로 딥러닝 기술을 기반으로 한 방안들에 집중되고 있다[1-11]. 현재 가장 많이 사용되는 딥러닝 방법으로 deep belief network(DBN), convolutional neural network(CNN), autoencoder and recurrent neural network(RNN) 등이 있다. 이러한 연구들 중 CNN 구조에 autoencoder를 다중화 하는 방안[3] 은 기본적으로 공간적 특성과 시간의 흐름에 따른 시간적 특성들 간의 관계성을 잘 찾아냈으며, 다차원 CNN 구조 방안[4]은 무선 네트워크 트래픽의 특성을 잘 분류해 내고 있다. deep convolutional neural network(deep CNN) 기반 학습 모델을 사용하는 방안[4]은 네트워크 완전한 모양의 트래픽의 입출력 신호로부터 안정적인 네트워크 트래픽의 특성을 추출해 냄을 보여 주고 있다. 또한 암호화된 네트워크 트래픽으로부터 침입/공격 징후를 추출해 낼 수 있는 이미지 기반의 방안[8]도 제안되었다. 침입탐지 시스템에 적용되고 있는 인공지능 기술은 주로 deep neural network(DNN) 기반의 침입 탐지처럼 신경망(neural network), 유전 알고리즘(generic algoritm), 면역 알고리즘(immune algorithm)을 포함하고 있으며 퍼지 시스템(fuzzy system) 기반의 침입 탐지 방법도 있다. 이러한 방법들은 데이터 세트를 처리하는 과정에서 키 정보 손실이 쉽게 일어나며, 샘플 데이터의 왜곡과 비효율적인 데이터 특성 추출 결과를 낳아 테스트 결과의 변동성을 크게 만든다. 이러한 문제를 해결하기 위해 최적화된 데이터 처리과정을 기반으로 한 침입 탐지, deep belief network (DBN) 기반의 침입 탐지, recurrent neural network(RNN) 기반의 침입 탐지 등 다양한 딥러닝 기술이 적용된 방안들이 연구 되었다. 이러한 다양한 연구들은 침입 탐지 정확도를 어느 정도 개선하였으나, 실험 파라미터를 조정하기 어렵고 보다 큰 계산량과 낮은 추출률의 문제에 직면하게 된다. 이러한 결점들을 극복하기 위해 convolution neural network(CNN)이 적용된 무선 침입 탐지 방안들은 확실히 분류 정확성을 개선하였음을 보여준다. 그러나 모델 학습 과정에서 낮은 수렴 속도와 일반화 능력으로 참양성률(true positive rate(TPR))이 상대적으로 낮고 위양성률(false positive rate(FPR))은 상대적으로 높게 나온다. CNN 적용된 침입 탐지 방안들은 샘플 인식 능력과 성능에서 개선되었으나 낮은 수준의 적합성과 일반화 능력이 결점으로 남아 있으며 검출 정확성과 효율성은 개선이 필요하다.

본 논문에서는 과대적합 문제를 피하면서 일반화 능력을 개선하기 위한 방안으로 CNN의 커널 크기를 축소하고 콘볼루션 계층을 이중화하여 병렬 연산을 하는 구조를 제안한다. 모델은 선 처리된 샘플 데이터를 이용하여 훈련하였다. 좋은 수렴 결과가 나올 때까지 훈련된 분류기는 실험을 통해 의미 있는 침입/공격 탐지 능력을 보여 주었다. 2장에서는 제안한 CNN 구조와 순방향전파 및 역전파 훈련 과정에 대한 설명과 이 과정에서 필요한 파라미터와 연산에 대해 알아본다. 3장에서는 훈련 데이터 및 테스트 데이터에 대한 특성 분석과 사전처리 과정을 다룬다. 4장에서는 제안한 방식과 기존 방식들에 대한 실험 및 결과를 마지막 5장은 분석 및 결론으로 구성하였다.

2. 관련연구

기존의 분류 검출 알고리즘은 훈련 과정에서 많은 수의 주요 파라미터를 수동으로 설정해야 하고 주요 특성 정보 손실 이 발생하며 파라미터의 튜닝이 쉽지 않다. 이러한 문제를 해결하기 위해 종단 간 반지도 방식의 훈련 분류기와 샘플 특성을 학습하고 검출하기 위해 다층 기능을 수행하는 구조의 CNN(Convolutional Neural Network)을 제안하였다. 제안한 구조의 CNN은 5개의 컨벌루션 계층(convolution layer), 2개의 풀링 계층(pooling layer), 4개의 완전 연결 계층(fully connected layer) 그리고 1개의 소프트맥스 계층(SoftMax layer)으로 구성되며 컨볼루션 계층, 풀링계층, 완전 연결 계층은 이중화 된 구조이다. 컨볼루션 계층은 활성화 함수로 렐루(Relu)를 사용하며 신경망의 연결 희소성을 높여 데이터 처리를 가속화시킨다. 침입 탐지 모델의 학습 훈련 중 과대적합을 피하기 위해 드롭아웃(dropout)을 두 개의 완전연결 계층에서 사용하였다. 출력은 concat()에 의해 병합되고 마지막 소프트맥스 계층에서 훈련 결과를 분류하게 된다. CNN 모델은 두 번째와 네 번째 컨볼루션 계층에서 연산을 수행하고 그 결과를 저장하는 것에서부터 시작한다. 다음은 컨볼루션, 풀링 및 전체 연결 작업을 별도로 수행한다. 세 번째와 다섯 번째 컨볼루션 연산에서는 출력 결과에 대해 동일한 작업을 수행한 후 텐서플로(tensorflow)의 concat() 함수를 이용하여 완전연결 계층의 출력 데이터에 대한 병합 작업을 수행한다. 마지막에 소프트맥스 분류 결과에 따라 손실 값을 계산하고 그 값을 사용하여 역전파를 수행하며 가중치는 수렴이 최적화될 때까지 반복 훈련을 수행한다.

첫 번째, 순방향 전파에서 입력 샘플은 사전 처리된 훈련 데이터 세트에서 고정 크기 블록을 무작위로 선택하도록 한다. 데이터 파라미터는 block_size, height, width, channel으로 설정하며 각 훈련에 대해 데이터 세트에서 크기 M인 블록이 선택된다. 입력 데이터의 높이와 너비는 1과 122으로 설정하며 단일채널을 사용한다. 입력 데이터는 처음 두 개의 컨볼루션 계층을 통과하고 다중 커널을 사용하여 초기 입력의 모든 특성 맵에 대해 컨볼루션 연산을 수행한다. 각각의 컨볼루션 커널은 신경망의 가중치 학습을 위한 특정 특성 맵과 같은 것이다. 두 번째 컨볼루션 출력은 세 번째와 네 번째 컨볼루션 입력으로 사용되며 활성화 함수로 렐루(ReLU)를 사용한다.

순방향 및 역전파 학습에 사용된 식들[4]은 다음과 같다. 식 (1)은 콘볼루션 연산식이며 y_j^l는 j 컨볼루션 커널로 l 컨볼루션 계층을 처리한 출력을 나타낸다. 그리고 σ는 활성화 함수, w는 가중치, b는 절편을 나타낸다. 식 (2)는 활성화 함수 렐루를 나타낸다.

$$y_j^l = \sigma\left(\sum_{i \in M_j} y_i^{l-1} w_{ij}^l + b_i^l\right) \tag{1}$$

$$ReLU(y) = \begin{cases} y\ (y > 0) \\ 0\ (y \leq 0) \end{cases} \tag{2}$$

네 번째와 다섯 번째 컨볼루션 계층의 출력인 특성 맵은 첫 번째와 세 번째 풀링 계층에서 최대 풀링(max pooling)을 적용한다. 식 (3)은 풀링 계층에서의 계산식이며 $down(z)$는 행렬 원소 z에 대한 다운 샘플링을 나타낸다. 풀링 계층의 데이터 출력 크기는 M/2이며 이것은 완전 연결 계층 2와 4를 통과하여 텐서플로(tensorflow)의 concat() 함수를 이용하여 병합 작업을 수행하여 크기 M의 데이터 세트를 만든다. 마지막으로 병합된 데이터는 소프트맥스 계층을 이용하여 분류된다.

$$z_j^l = \beta\left(w_j^l down\left(z_j^{l-1}\right) + b_j^l\right) \tag{3}$$

순방향 전파에서 전체 오차 파라미터 값 손실은 훈련 세트의 샘플을 분류하기 위한 소프트맥스 계층을 이용하여 계산된다. 계산된 손실을 기반으로 역전파를 수행하며 실제 출력 값과 이상적인 출력 값 사이 오차에 따라 각 계층의 가중치와 절편을 조정하기

위해 모델이 좋은 수렴 효과를 얻을 때까지 반복적으로 수행된다. 최적의 가중치 w와 절편 b 그리고 각 계층의 출력 $f(x)$는 모든 훈련에 적합할 수 있다. 입력x와 손실 함수 $C(w,b)$는 모델 적합도를 정량화하기 위해 최적의 파라미터의 조합을 찾기 위해 설정된다. 손실 함수를 최소화하는 방법으로 SGD(Stochastic Gradient Descent) 알고리즘이 적용되며 손실 함수는 다음과 같이 정의된다.

$$C(w,b) \equiv \frac{1}{2n}\sum_{x} y^{(x)} - a^2 \tag{4}$$

여기서, w와 b는 가중치와 옵셋 집합을 나타내고 n은 훈련 입력 데이터의 수 그리고 a는 입력이 x일 때 벡터 출력을 나타낸다. w와 b는 각각 식 (5), (6)과 같이 정의된다.

$$w \rightarrow w_k^{'} \equiv w_k - \eta \frac{\theta C}{\theta w_k} \tag{5}$$

$$b \rightarrow b_l^{'} \equiv b_l - \eta \frac{\theta C}{\theta b_l} \tag{6}$$

여기서, η는 학습률이고 편미분 $\theta C/\theta w_k$와 $\theta C/\theta b_l$는 임의 편향 가중치와 절편의 변화율을 나타낸다.

3. 데이터 특성 분석

본 논문에서는 KDD CUP 99[5]를 개선한 NSL-KDD CUP 데이터 세트[15]를 사용하였다. 모델 훈련 및 테스트를 위한 NSL-KDD는 KDD CUP 99 데이터 세트에서 많은 양의 중복 데이터를 삭제하고, 샘플 데이터의 비율 분포를 더 균형 잡히고 합리적이며 더 유용하도록 만들어졌다. 그래서 이것은 검증 실험 요구 사항을 보다 더 충족할 수 있다. 두 종류으 데이터 세트의 샘플 데이터 특성은 동일하다. 데이터 세트에 포함되어 있는 침입 레코드는 42 종류의 특성을 가지고 있으며 이것은 38 종류의 디지털 특성, 3 종류의 기호 특성 및 하나의 공격 유형 레이블로 세분화된다. NSL-KDD CUP 데이터 세트의 데이터 유형에는 주로 일반 데이터와 4대 공격 유형 데이터 Probe, DOS, U2R, R2L등이 있다. 4가지 공격 유형의 데이터는 구체적으로 39개의 하위 클래스로 세분화된다. NSL-KDD 데이터 세트는 3개의 하위 데이터 세트인 훈련 세트 KDDTrain, 두 개의 테스트 세트 KDDTest+, KDDTest-21를 포함한다. 샘플의 범주 분포, 특성 분류는 Table 1, Table 2 그리고 KDDTrain의 샘플 데이터는 Fig. 1에서 보여준다.

Table 1. Simple category distribution table

Category	KDDTrain	KDDTest+	KDDTest-21
Probe	45927	2421	4342
DOS	11656	7458	2402
R2L	995	2754	2754
U2L	52	200	200
Normal	67343	9711	2152
Total	125973	22544	11850

Table 2. Feature classification table

No.	Feature	Type	No.	Feature	Type
1	duration	continuous	22	is_guest_login	discrete
2	protocol_type	symbolic	23	count	continuous
3	service	symbolic	24	srv_count	continuous
4	flag	symbolic	25	serror_rate	continuous
5	src_bytes	continuous	26	srv_serror_rate	continuous
6	dst_bytes	continuous	27	rerror_rate	continuous
7	land	discrete	28	srv_rerror_rate	continuous
8	wrong_fragment	continuous	39	same_srv_rate	continuous
9	urgent	continuous	30	diff_srv_rate	continuous
10	hot	continuous	31	srv_diff_host_rate	continuous
11	num_failed_logins	continuous	32	dst_host_count	continuous
12	logged_in	discrete	33	dst_host_srv_count	continuous
13	num_compromised	continuous	34	dst_host_same_srv_rate	continuous
14	root_shell	discrete	35	dst_host_diff_srv_rate	continuous
15	su_attempted	discrete	36	dst_host_same_src_port_rate	continuous
16	num_root	continuous	37	dst_host_srv_diff_host_rate	continuous
17	num_file_creations	continuous	38	dst_host_serror_rate	continuous
18	num_shells	continuous	49	dst_host_srv_serror_rate	continuous
19	num_access_files	continuous	40	dst_host_rerror_rate	continuous
20	num_outbound_cmds	continuous	41	dst_host_srv_rerror_rate	continuous
21	is_host_login	discrete			

duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_logins	logged_in	num_compromised	root_shell	su_attempted	num_root	num_file_creations	num_shells	num_access_files	num_outbound_cmds	is_host_login	is_guest_login	count	srv_count	serror_rate	srv_serror_rate	rerror_rate	srv_rerror_rate	same_srv_rate	diff_srv_rate	srv_diff_host_rate	dst_host_count	dst_host_srv_count	dst_host_same_srv_rate	dst_host_diff_srv_rate	dst_host_same_src_port_rate	dst_host_srv_diff_host_rate	dst_host_serror_rate	dst_host_srv_serror_rate	dst_host_rerror_rate	dst_host_srv_rerror_rate	class
0	tcp	private	S0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	74	16	1	1	0	0	0.22	0.05	0	255	5	0.02	0.06	0	0	1	1	0	0	anomaly
0	tcp	ftp_data	SF	567	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	5	38	0.25	0.09	0.25	0.05	0	0	0	0	normal
0	tcp	hostnames	REJ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	225	2	0	0	1	1	0.01	0.07	0	255	2	0.01	0.08	0	0	0	0	1	1	anomaly
0	tcp	telnet	S0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	66	16	0.97	1	0.03	0	0.24	0.09	0	255	32	0.2	0.04	0	0	1	1	0	0	anomaly
5943	tcp	ftp_data	SF	5131424	0	0	0	0	4	0	1	0	0	0	0	0	0	0	0	0	0	8	7	0	0.14	0	0	1	0	0.29	7	43	1	0	1	0.14	0	0.02	0	0	anomaly
0	tcp	http	SF	330	420	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	7	7	0	0	0	0	1	0	0	255	255	1	0	0.01	0	0	0	0	0	normal
0	icmp	eco_i	SF	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	35	0	0	0	0	1	0	1	1	61	1	0	1	0.51	0	0	0	0	anomaly
0	tcp	smtp	SO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0.5	0.5	0	0	1	0	0	160	101	0.4	0.05	0.01	0.02	0.01	0.02	0	0	normal
0	tcp	http	SF	294	1224	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	4	4	0	0	0	0	1	0	0	54	255	1	0	0.02	0.04	0	0	0	0	normal
0	tcp	time	RSTR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	255	1	0	0.16	0.15	0	0	0	0.15	1	anomaly
0	tcp	ftp_data	SF	10388	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	255	81	0.32	0.02	0.21	0	0	0	0	0	normal
0	tcp	auth	SO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	139	53	1	1	0	0	0.09	0.06	0	255	13	0.05	0.07	0	0	1	1	0	0	anomaly
0	tcp	http	SF	344	573	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	28	66	0.04	0.03	0	0	1	0	0.86	255	255	1	0	0	0	0	0	0	normal
0	tcp	exec	RSTO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	144	16	0	0	1	1	0.11	0.06	0	255	9	0.04	0.06	0	0	0	0	1	1	anomaly
0	tcp	netbios_ssn	RSTO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	241	19	1	1	0	0	0.08	0.05	0	255	19	0.07	0.07	0	0	1	1	0	0	anomaly
0	tcp	http	SF	245	827	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	15	19	0	0	0	0	1	0	0.11	107	255	1	0	0.01	0.01	0	0	0	0	normal
0	icmp	eco_i	SF	1032	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	213	213	0	0	0	0	1	0	0	255	255	1	0	0	0	0	0	0	0	anomaly
0	icmp	eco_i	SF	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	45	0	0	0	0	1	0	1	1	11	1	0	1	0.55	0	0	0	0	anomaly
0	tcp	private	REJ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	366	1	0.17	0	0.83	1	0.03	1	0	255	1	0	0.47	0	0	0.82	0	0.13	1	anomaly
0	udp	private	SF	105	146	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3	0	0	0	0	0.67	0.67	0	255	239	0.94	0.01	0.01	0	0	0	0	0	normal
0	tcp	private	S0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	254	17	1	1	0	0	0.07	0.05	0	255	17	0.07	0.08	0	0	1	1	0	0	anomaly
0	tcp	smtp	SF	1088	327	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	1	0	1	202	154	0.76	0.02	0	0.01	0	0	0	0	normal
0	tcp	telnet	S0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	295	7	1	1	0	0	0.02	0.05	0	255	7	0.03	0.05	0	0	1	1	0	0	anomaly
0	tcp	private	S0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	171	3	1	1	0	0	0.02	0.07	0	255	14	0.05	0.07	0	0	1	1	0	0	anomaly
0	tcp	Z39_50	SO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	179	2	1	1	0	0	0.01	0.06	0	255	5	0.02	0.07	0	0	1	1	0	0	anomaly
0	tcp	smtp	SF	657	326	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	2	1	0	0	0	0	0.5	1	0	255	11	0.04	0.6	0	0	0	0.09	0	0	normal
0	tcp	smtp_443	SO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	101	7	1	1	0	0	0.07	0.06	0	255	3	0.01	0.07	0	0	1	1	0	0	anomaly
0	tcp	http	SF	284	765	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	9	9	0	0	0	0	1	0	0	255	255	1	0	0.01	0	0	0	0	0	normal
0	tcp	http	SF	261	15455	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	22	22	0	0	0	0	1	0	0	255	255	1	0	0	0	0	0	0	0	normal
0	tcp	http	SF	317	315	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	1	0	1	255	255	1	0	0	0	0	0	0	0	normal
0	udp	domain_u	SF	33	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	8	8	0	0	0	0	1	0	0	255	36	0.14	0.03	0.14	0	0	0	0	0	normal
0	tcp	private	REJ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1	0	0	1	1	0.5	1	0	255	1	0	0.6	0.58	0	0.01	0	0.57	1	anomaly
0	tcp	private	RSTR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	1	0	0	255	1	0	0.64	0.64	0	0	0	0.64	1	anomaly
0	udp	domain_u	SF	44	70	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	195	197	0	0	0	0	1	0	0.01	255	234	0.92	0.01	0.01	0	0	0	0	0	normal
0	tcp	http	SF	215	5864	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	4	4	0	0	0	0	1	0	0.66	255	255	1	0	0	0	0	0	0	0	normal

Fig. 1. Sample data of KDD Train+

데이터 전처리에는 수치 처리와 정규화 처리의 두 가지 프로세스가 있다. 첫 번째 수치처리 프로세스에서 입력 데이터 세트는 매트릭스이기 때문에 테스트 데이터 세트는 원-핫 인코딩(one-hot encoding)을 적용한다. 수치 처리 후 세 가지 유형의 상징적 특성은 84가지 이진 특성 벡터가 된다. 그리고 데이터 세트의 38가지 디지털 특성으로 데이터 세트에 있는 각 레코드의 42가지 특성은 결국 122가지 이진 특성 벡터가 된다. 다음은 3가지 유형의 인코딩을 나타낸다.

1) TCP, UDP 및 ICMP와 같은 protocol_type의 특성은 각각 이진 벡터 (1, 0, 0), (0, 1, 0), (0, 0, 1)로 인코딩
2) 70개의 심볼 속성이 70개의 이진 특성 벡터로 인코딩
3) 플래그 유형에 포함된 11가지 심볼 속성이 11가지 이진 특성으로 인코딩

정규화 프로세스에서 공식은 식 (7)과 같다. $X_{\max}$와 $X_{\min}$은 특성의 최대값과 최소값을 나타낸다. 데이터 세트에서 연속적인 데이터 특성 사이의 값 범위는 크게 다르다. 예를 들어, num_root 유형 특성의 값 범위 [0, 7468]이고 num_shells 유형 특성의 값 범위는 [0, 5]이다. 이와 같이 유형에 따라 최소값과 최대값의 범위는 매우 다르다. 산술 연산을 용이하게 하기 위해 정규화 처리 방법을 적용하여 각 특성 값의 범위는 간격 [0, 1]으로 균일하게 매핑된다.

$$X_{norm} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{7}$$

4. 실험 및 결과

테스트 모델의 검증 및 비교 테스트 및 계산 효율을 높이고 훈련 시간을 줄이기 위해 Tensorflow-GPU를 사용하였다. 실험을 위한 소프트웨어 및 하드웨어 구성 환경은 다음과 같다.

- OS: 윈도우 10 (64bit)
- CPU: Intel Core i5-4570 3.2GHz
- RAM: 8G
- GPU: NVIDIA Geforce GT625
- Python 3.8.5

실험을 위한 파라미터들로 학습률(learning rate)은 0.1, 드롭 아웃(dropout)은 0.5 그리고 총 실험 훈련의 반복 횟수는 300으로 설정하였다. 데이터의 항목 개수는 1000이고 각 배치 크기에 대한 반복 횟수는 50이다. 제안한 수정된 CNN 구조에 대한 성능은 정확도(AC), 참 양성률(TPR) 및 위양성률(FPR) 3가지 지표[4]를 사용하여 기존 DBN과 RNN을 비교 평가하였다. 각 평가 지표는 다음과 같다.

$$AC = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

$$TPR = \frac{TP}{TP + FN} \tag{9}$$

$$FPR = \frac{FP}{FP + TN} \tag{10}$$

여기서, TP는 True Positive class를 나타내며 포지티브 클래스로 올바르게 분류된 포지티브 클래스에 속하는 샘플의 수를 의미한다. TN는 True Negative class를 나타내며, 네거티브 클래스로 올바르게 분류된 네거티브 클래스에 속하는 샘플의 수를 의미한다. FP는 False Positive class이며, 포지티브 클래스로 잘못 분류된 네거티브 클래스에 속하는 샘플의 수를 의미한다. FN는 False Negative class이며, 네가티브로 클래스로 잘못 분류된 포지티브 클래스에 속하는 샘플의 수를 의미한다. 위 평가 지표들에 대한 실험 결과는 Fig. 2에서 보여 주고 있다.

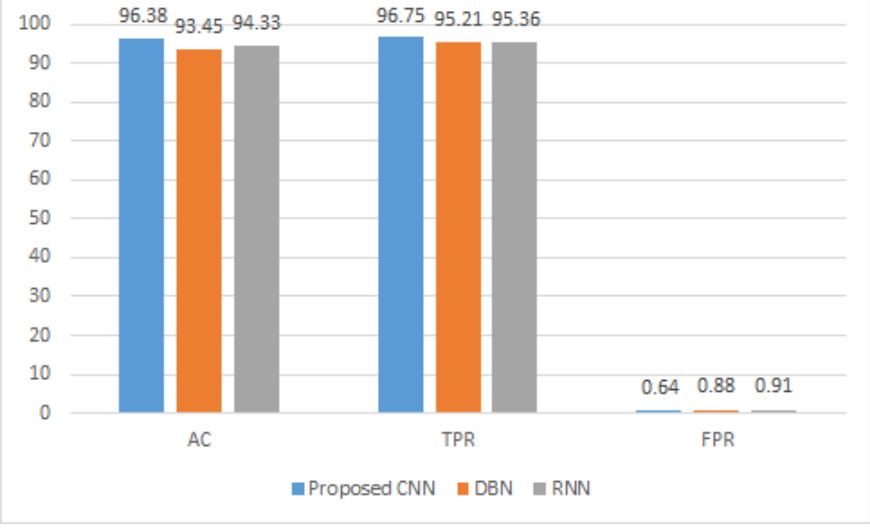

Fig. 2. Test result of proposed CNN, DBN and RNN

5 . 분석 및 결론

무선 네트워크 침입/공격 탐지[12-14]에서 기존 딥러닝 방안인 DBN과 RNN은 모델 학습 과정에서 과대적합 문제와 낮은 수준의 일반화 문제를 가지고 있어 낮은 탐지 효율성을 가지고 있다는 것이다. 본 논문은 과대적합 문제를 피하면서 일반화 능력을 개선하기 위한 방안으로 CNN의 커널 크기를 축소하고 콘볼루션 계층을 이중화하여 병렬 연산을 하는 구조를 제안하였다. 제안된 모델은 선 처리된 샘플 데이터를 이용하여 좋은 수렴 결과가 나올 때까지 분류기를 반복 훈련을 실험 하였다.분류 학습 및 테스트 실험은 NSL-KDD CUP 데이터 세트를 사용하여 사전 처리된 훈련 데이터 세트와 테스트 데이터 세트를 만들었다. 실험 및 분석 결과 제안한 CNN이 기존 DBN과 RNN에 비해 더 높은 탐지 정확도(AC)와 참양성률(TPR) 그리고 낮은 위양성율(FPR)을 보여 주었다. 제안된 방안이 샘플 데이터의 특성 추출을 위해 보다 나은 딥러닝 모델이 될 수 있음을 보여주었다. 실험 과정에서 확률적 경사 하강 알고리즘은 경사도 분산이 모델 학습 과정에서 쉽게 일어나는 문제와 손실 함수가 부분적으로 최적해에 쉽게 도달하는 문제를 나타냈다. 향후 파라미터에 대한 다양한 튜닝 테스트와 제안된 방안에 대해 다른 새로운 데이터 세트를 이용한 학습 훈련과 테스트 검증 실험을 할 것이다.

References

1. S. Zou, F. Zhong, B. Han, and H. Sun, “Network Intrusion Detection Method Based on Deep Learning”, Journal of Physics Conference Series, 1966(1):012051, 2021.
2. Y. He, and W. Li, “Image-based Encrypted Traffic Classification with Convolution Neural Networks[C]”, 2020 IEEE Fifth International Conference on Data Science in Cyberspace (DSC), IEEE, 2020.
3. L. Yang, J. Li, L. Yin, Z. Sun, Y. Zhao, and Z. Li, “Real-Time Intrusion Detection in Wireless Network: A Deep Learning-Based Intelligent Mechanism”, IEEE Access, Vol. 8, pp. 170128-170139, 2020.
4. H. Yang, and F. Wang, “Wireless Network Intrusion Detection Based on Improved Convolutional Neural Network”, IEEE Access, Vol. 7, pp(99): 1-1, 2019.
5. 김동훈, 김정재, 손인수, “KDD CUP 99를 이용한 기계학습 기반 네트워크 침입 탐지 기법 연구”, 한국통신학회 학술대회논문집, pp.861-862, 2019.
6. 정기문, “딥러닝 기반 네트워크 침입탐지를 위한 데이터 전처리 방안 연구”, 한국 컴퓨터정보학회 하계학술대회 논문집, 제26권 제2호, 2018.
7. 김동훈, 손인수, “네트워크 침입탐지 기술 연구 동향”, 전자공학회논문지, 제56권 제8호, pp.3-12, 2019.
8. S. S. Roy, A. Mallik, R. Gulati, M. S. Obaidat, and P. V. Krishna, “A Deep Learning Based Artificial Neural Network Approach for Intrusion Detection”, in Proceedings of the International Conference on Mathematics and Computing, Berlin, Germany, pp. 44-53, 2017.
9. B. A. Tama, and K.-H. Rhee, “A Detailed Analysis of Classifier Ensembles for Intrusion Detection in Wireless Network”, JIPS (Journal of Information Processing Systems), Vol. 13, No. 5, pp. 1203-1212, 2017.
10. I. Al-Shourbaji, and S. Al-Janabi, “Intrusion Detection and Prevention Systems in Wireless Networks”, Kurdistan Journal of Applied Research, Vol. 2, No. 3, 2017.
11. A. Y. Javaid, and W. Sun, “A Deep Learning Approach for Network Intrusion Detection System”, 9th EAI International Conference on Bio-inspired Information and Communications Technologies At: New York, 2015.
12. D. W. F. L. Vilela, E. W. T. Ferreira, A. A. Shinoda, N. V. de Souza Araújo, R. de Oliveira, and V. E. Nascimento, “A Dataset for Evaluating Intrusion Detection Systems in IEEE 802.11 Wireless Networks”, IEEE Colombian Communications Conference, 2014.
13. K. El-Khatib, “Impact of Feature Reduction on the Efficiency of Wireless Intrusion Detection Systems”, Parallel and Distributed Systems”, IEEE Transactions on, Vol. 21, No. 8, pp. 1143-1149, 2010.
14. 이형우, “무선 네트워크 침입탐지/차단 시스템(Wireless IPS) 기술”, 한국통신학회지, 제22권 제8호, pp.114-128, 2005.
15. https://web.archive.org/web/20150205070216/http://nsl.cs.unb.ca/NSL-KDD/