Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.30, NO.4, Aug. 2020 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2020.30.4.617

비지도학습 오토 엔코더를 활용한 네트워크 이상 검출 기술

강 구 홍†‡
서원대학교(교수)

Network Anomaly Detection Technologies Using Unsupervised Learning AutoEncoders

Koohong Kang†‡
Seowon University(Professor)

요 약

인터넷 컴퓨팅 환경의 변화, 새로운 서비스 출현, 그리고 지능화되어 가는 해커들의 다양한 공격으로 인한 규칙 기반 침입탐지시스템의 한계점을 극복하기 위해 기계학습 및 딥러닝 기술을 활용한 네트워크 이상 검출(NAD: Network Anomaly Detection)에 대한 관심이 집중되고 있다. NAD를 위한 대부분의 기존 기계학습 및 딥러닝 기술은 '정상'과 '공격'으로 레이블링된 훈련용 데이터 셋을 학습하는 지도학습 방법을 사용한다. 본 논문에서는 공격의 징후가 없는 일상의 네트워크에서 수집할 수 있는 레이블링이 필요 없는 데이터 셋을 이용하는 비지도학습 오토 엔코더(AE: AutoEncoder)를 활용한 NAD 적용 가능성을 제시한다. AE 성능을 검증하기 위해 NSL-KDD 훈련 및 시험 데이터 셋을 사용해 정확도, 정밀도, 재현율, f1-점수, 그리고 ROC AUC (Receiver Operating Characteristic Area Under Curve) 값을 보인다. 특히 이들 성능지표를 대상으로 AE의 층수, 규제 강도, 그리고 디노이징 효과 등을 분석하여 레퍼런스 모델을 제시하였다. AE의 훈련 데이터 셋에 대한 재생오류 82-th 백분위수를 기준 값으로 KDDTest+와 KDDTest-21 시험 데이터 셋에 대해 90.4%와 89% f1-점수를 각각 보였다.

ABSTRACT

In order to overcome the limitations of the rule-based intrusion detection system due to changes in Internet computing environments, the emergence of new services, and creativity of attackers, network anomaly detection (NAD) using machine learning and deep learning technologies has received much attention. Most of these existing machine learning and deep learning technologies for NAD use supervised learning methods to learn a set of training data set labeled 'normal' and 'attack'. This paper presents the feasibility of the unsupervised learning AutoEncoder(AE) to NAD from data sets collecting of secured network traffic without labeled responses. To verify the performance of the proposed AE mode, we present the experimental results in terms of accuracy, precision, recall, f1-score, and ROC AUC value on the NSL-KDD training and test data sets. In particular, we model a reference AE through the deep analysis of diverse AEs varying hyper-parameters such as the number of layers as well as considering the regularization and denoising effects. The reference model shows the f1-scores 90.4% and 89% of binary classification on the KDDTest+ and KDDTest-21 test data sets based on the threshold of the 82-th percentile of the AE reconstruction error of the training data set.

Keywords: Network Anomaly Detection, NSL-KDD Data Set, AutoEncoder, Unsupervised Learning

Received(04. 23. 2020), Modified(1st: 07. 10. 2020, 2nd: 08 .05. 2020), Accepted(08. 05. 2020)

† 주저자, khkang@seowon.ac.kr
‡ 교신저자, khkang@seowon.ac.kr(Corresponding author)

I. 서 론

기계학습(ML: machine learning)과 딥러닝(DL: deep learning) 기술은 지난 수 십 년간 음성인식, 패턴인식, 그리고 컴퓨터 비전(computer vision) 등 다양한 분야에서 매우 성공적인 발전을 이루어 왔다[1]. 뿐만 아니라, 인터넷 컴퓨팅 환경의 변화와 새로운 서비스 출현, 그리고 지능화되어 가는 해커들의 다양한 공격으로 인한 규칙-기반(signature-based) 침입탐지시스템(IDS: intrusion detection system)의 한계점을 극복하기 위해 ML/DL 기술을 활용한 네트워크 이상 검출(NAD: Network Anomaly Detection)에 대한 관심이 집중되고 있다[2].

2009년 Tavallaee et al.[3]이 IDS 성능을 검증하기 위한 NSL-KDD 데이터 셋[4]을 제공하고, 이를 이용한 SVM(Support Vector Machine), MLP(Multi-layer Perceptron), Naive Bayes 등 몇몇 ML 알고리즘을 사용한 NAD 결과를 보였다. 이후, 이들 데이터 셋을 활용하여 ML 알고리즘 뿐 만아니라 CNN(Convolutional Neural Network)과 RNN(Recurrent Neural Network) 등 DL 기술을 활용한 많은 NAD 연구결과들이 발표되었다[5,6,19]. 이들 대부분의 기존 연구들은 네트워크 트래픽 레코드들이 '정상(normal)' 혹은 '공격(attack)'으로 레이블링(labeling)된 훈련 데이터 셋을 이용하는 지도학습(supervised learning) 모델을 사용하였다. 따라서 이들 연구 결과를 실제 네트워크에 적용하기 위해서는 해당 네트워크에서 수집된 완벽하게(ground-truth) 레이블링된 적절한 개수의 균형잡힌 '정상'과 '공격' 샘플수가 확보된 훈련용 데이터 셋이 반드시 필요하다. 궁극적으로 자신의 네트워크 환경에서 이들 훈련용 데이터 셋 확보가 어렵거나 불가능하다면, 이들 기술을 사용하는 것은 사실상 불가능하다.

오늘날 방화벽 및 침입탐지시스템 등 다양한 보안 장비로 보호된 실제 네트워크 환경에서 다양한 공격 시나리오가 포함된 네트워크 트래픽을 확보하는 것은 쉽지 않다. 뿐만 아니라, 수집된 네트워크 트래픽으로부터 '정상'과 '공격'을 정확하게 레이블링하는 작업은 매우 어렵고 복잡한 과정을 거쳐야 한다[7]. 따라서 본 논문에서는 평상시 제공되는 정상 네트워크 트래픽만을 이용하여 학습하는 비지도학습(unsupervised learning) 모델인 오토 엔코더(AE: AutoEncoder)를 이용한 NAD 기술에 대해 집중한다.

AE는 비지도학습 즉 레이블되지 않은 훈련 데이터 셋을 사용하여 입력 데이터의 효율적인 표현(representation)을 할 수 있는 인공 신경망이다. 이러한 표현을 코딩(coding)이라고 하며, 이런 코딩은 일반적으로 입력보다 훨씬 낮은 차원을 가지므로 AE는 비선형 차원 축소(nonlinear dimensionality reduction) 기술로 최근 많이 사용되고 있다[8,9]. 특히, 주목할 것으로는 AE가 다양한 분야에서 이상 검출 기술로도 활용되고 있다는 것이다. 예를 들어, Borghesi et al.[10]은 수퍼컴퓨터 노드의 이상을 검출하였으며, Sakurada와 Yairi[11]는 우주선의 텔리메트리(telemetry) 데이터의 이상을 검출하고, 그리고 기본적인 AE 모델 뿐만 아니라 변형된 AE 모델을 활용한 NAD 분야에 대한 연구가 일부 발표되고 있다[12,13,14,15].

AE를 활용한 NAD 분야에 대한 기존 연구결과들을 살펴보면, 비지도학습 즉 레이블링이 필요 없는 모델임에도 불구하고 이상 검출을 위한 컨볼루션 층 사용[12], softmax 층 추가[13,14], 혹은 하이퍼 파라미터(hyper-parameter) 결정[15] 등을 위해 레이블링된 학습 데이터 셋을 요구한다. 즉 이들 기존 연구들은 AE를 활용하지만 레이블링된 훈련용 데이터 셋을 필요로 한다. 따라서 앞에서 언급한 바와 같이 비지도학습 모델의 장점을 최대한 확보하기 위해, 본 논문에서는 레이블링된 훈련 데이터 셋이 필요 없는 가장 기본적인 AE 구조와 하이퍼 파라미터를 활용하여 NAD 성능을 확인한다.

ML/DL 기술을 적용하고 원하는 성능을 검증하기 위해서는 적절한 훈련 및 시험 데이터 셋을 사용해야 한다. 본 논문에서는 IDS 및 NAD를 위한 많은 기존 연구들이 사용하는 NSL-KDD 데이터 셋[3,4,17]을 활용한다. NSL-KDD 데이터 셋은 훈련을 위한 KDDTrain+와 시험을 위한 KDDTest+와 KDDTest-21을 제공한다. 하지만, 일부 기존 연구들은 KDDTrain+의 일부 셋을 교차검증용으로 사용하여 높은 성능 결과를 보이거나[12], KDDTest+만 사용하거나[14,15], 혹은 NSL-KDD 데이터 셋의 이전 버전을 사용하여[13] 높은 성능 결과를 제시하는 오류를 범하고 있다. 본 논문에서는 KDDTest+와 KDDTest-21 시험용 데이터 셋에 대한 특정 레코드의 선택이나 삭제 없이

그대로 사용함으로써, 이들 데이터 셋을 사용하여 성능결과를 보여주는 다른 연구결과들과 객관적인 비교분석이 될 수 있도록 하였다.

서론에 이어, 제2장에서는 본 논문에서 사용하는 기본적인 AE와 디노이즈 오토엔코더(DAE: Denoising Autoencoder)에 대해 간략히 설명하고 제3장에서는 AE를 활용하여 이상 검출을 위한 기존 연구 결과와 문제점들을 간략히 알아본다. 제4장에서는 텐서플로우(TensorFlow), 사이킷런(Scikit-Learn), 그리고 케라스(keras)[16]를 이용해 AE 및 DAE를 모델링하고, 제 5장에서는 NSL-KDD 시험 데이터 셋을 대상으로 정확도(accuracy), 정밀도(precision), 재현율(recall) 그리고 F1-점수(F1 score)[17]를 알아봄으로써 AE 활용에 따른 NAD 적용 가능성을 보인다. 마지막으로 제6장에서 결론 및 향후 연구 방향에 대해 기술하였다.

II. 관련 연구

1988년 Bourlard 와 Kamp[8]는 auto-association 모드에서 다층 퍼셉트론(multilayer perceptron)이 정보처리와 같은 분야에서 데이터 압축과 차원 축소에 매우 효과적임을 보였다. 이것을 시작으로 AE는 최근 데이터 생성 모델(generative models)에 적용되기 까지 매우 다양한 분야에 활용되고 있다. 본 장에서는 AE를 활용한 이상 검출에 관한 기존 연구들에 대해서 간략히 소개한다.

Borghesi et al.[10]은 AE가 수퍼컴퓨터 노드들의 정상행위(노드 상태를 나타내는 다양한 속성. 예를 들어, 전력소모, 부하, 온도 등)를 학습하고 학습한 모델을 이용해 비정상적인 상태를 탐지하는 아이디어를 제안하였다. AE는 정상상태의 노드의 속성 값에 대해서는 RE를 최소화하도록 학습된다. 따라서 훈련 데이터 셋 내에는 존재하지 않지만 이들과 매우 유사한 새로운 입력에 대해서는 매우 높은 신뢰도로 이들 입력을 재생하게 될 것이다. 반대로, 훈련 기간 동안 보여주지 않은 새로운 속성 패턴을 가진 입력에 대해서는 정확하게 재생하지 못함으로써 결과적으로는 RE는 매우 큰 값을 가지게 된다. 이들은 88%에서 96% 수준의 이상 검출 정확도를 보였다. Sakurada와 Yairi[11]는 AE와 DAE를 사용하여 우주선 텔리메트리(telemetry) 데이터의 이상 검출을 시도하였다. 우주선 텔리메트리 데이터는 많은 종류의 센서 측정값을 가지고 있으며 이들 값들은 상호 연관성을 가진다. 따라서 부수적인 입력 값들은 제거할 수 있어 낮은 차원의 벡터로 표현될 수 있다. 이들은 AE가 선형 PCA(Principal Component Analysis) 보다 이상 검출 정확도가 높음을 보였다.

Chen et al.[12]은 컨볼루션 층을 사용한 AE를 이용해 NSL-KDD 데이터 셋에 대한 NAD 결과를 보였다. 그러나 이들은 훈련 데이터 셋의 일부를 교차검증(cross-validation) 데이터 셋으로 사용한 성능 결과를 보임으로써 훈련 시 딥러닝 모델에 보여주지 않은 시험 데이터 셋에 대한 결과는 확인할 수 없다. 뿐만 아니라, 이상 검출을 위한 AE의 RE에 대한 기준 값 설정을 어떻게 할지에 관한 구체적인 방법을 제시하지 않았다. Farahnakian과 Heikkonen은[13] 여러 개의 AE와 softmax를 사용한 출력 층 구조를 제안하고 레이블링된 학습 데이터 셋을 사용하였다. 이들은 NSL-KDD 데이터 셋의 이전 버전인 KDD-CUP'99 데이터 셋을 사용하여 94% 수준의 정확도를 보였다. 하지만, KDD-CUP'99 데이터 셋은 중복된 레코드 등으로 인해 지나치게 높은 성능을 보이는 것으로 조사되고 있다[3]. 뿐만 아니라, 이들이 제시한 모델은 AE의 비지도 학습과 softmax를 사용한 지도학습을 혼용해 사용함으로써 비지도학습을 지향하는 AE의 장점을 충분히 살리지 못하는 단점이 있다. Ieracitano et al.[14]은 MAD(Median Absolute Deviation) 예측기, 속성 검출기 그리고 AE+softmax로 구성된 구조를 사용하여 KDDTest+ 데이터 셋에 대한 87% 정확도를 보였다. 그러나 이들 역시 마지막 층을 지도학습 방법을 채택함으로써 AE의 비지도 학습에 대한 장점을 충분히 살리지 못한다. 뿐만 아니라, 이들은 복잡한 예측기 및 속성 검출기를 함께 사용해야 하는 단점이 있다. Aygun과 Yavuz[15]는 AE를 활용하여 KDDTest+ 데이터 셋에 대해 88% 정확도를 보였다. 하지만, 이들은 AE의 RE 기준 값 결정을 위해 레일블링된 훈련 데이터 셋으로부터 '정상'과 '공격' 트래픽 분포와 정확도 계산이 필요하다. 본 논문에서는 레이블링된 훈련 데이터 셋이 필요 없는 기본적인 AE를 활용하여 KDDTest+와 KDDTest-21에 대한 다양한 성능 지표 결과 값을 분석함으로써 실제 네트워크에 적용 가능 여부를 확인하고자 한다.

III. 오토엔코더(AutoEncoder)

3.1 기본적인 오토엔코더(Basic AutoEncoder)

[그림 1]에서 보듯이, 기본적인 AE는 입력 벡터 $\mathbf{x} \in \mathbb{R}^{d}$ 를 수신하고 인코더 모델 $\mathbf{z} = f_{\theta}(\mathbf{x}) = s(\mathbf{W}\mathbf{x} + \mathbf{b})$를 통해 입력 벡터를 히든 표현 (hidden representation 혹은 latent vector) $\mathbf{z} \in \mathbb{R}^{d'}$ 으로 표현한다(여기서, s는 활성함수(activation function)를 나타낸다). $\mathbf{W}$는 $d' \times d$ 가중치 행렬(weight matrix)이고 $\mathbf{b}$ 는 바이어스(bias) 벡터이다. 코딩된 latent 벡터 $\mathbf{z}$는 디코더 모델 $\mathbf{y} = g_{\theta'}(\mathbf{z}) = s(\mathbf{W}'\mathbf{x} + \mathbf{b}')$을 통해 "재생성(reconstruction)" 벡터 $\mathbf{y} \in \mathbb{R}^{d}$ 로 매핑된다. 결국 i-th $(i = 1, \cdots, n)$ 번째 학습용 데이터 샘플 $\mathbf{x}^{(i)}$ (d 차원) 는 엔코더에 의해 latent 벡터 $\mathbf{z}^{(i)}$(d' 차원)로 차원 축소되고 다시 디코더에 의해 $\mathbf{y}^{(i)}$(d 차원)로 재 생성된다. 엔코더와 디코더에 사용된 모델 파라미터 $\theta = \{\mathbf{W}, \mathbf{b}\}$와 $\theta' = \{\mathbf{W}', \mathbf{b}'\}$는 평균 재생성 오류(RE: Reconstruction Error)를 최소화하도록 식1과 같이 최적화한다[8,9].

$$\theta^*, \theta'^* = \underset{\theta, \theta'}{\arg\min} \frac{1}{n} \sum_{n=1}^{n} L(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) = \underset{\theta, \theta'}{\arg\min} \frac{1}{n} \sum_{n=1}^{n} L(\mathbf{x}^{(i)}, g_{\theta'}(f_{\theta}(\mathbf{x}^{(i)}))) \quad (1)$$

여기서, L은 손실함수(loss function)로 다음과 같은 제곱 오류(squared error)와 같다.

$$L(x^{(i)}, y^{(i)}) = \sum_{j=1}^{d} (x_j^{(i)} - y_j^{(i)})^2 \quad (2)$$

AE의 RE, 즉 손실함수에서 알 수 있듯이 모델의 출력 $\mathbf{y}$를 입력 $\mathbf{x}$를 학습할 수 있도록 유도함으로써 AE는 비지도 학습 문제를 지도 학습 문제로 바꾸어서 문제를 해결하고 있다. 따라서 AE를 자가 학습(self-learning or self-supervised learning) 혹은 반지도 학습(semi-supervised learning)이

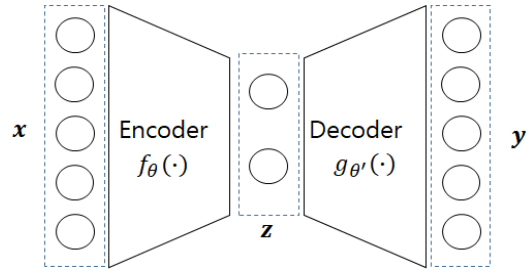

Fig. 1. Basic AutoEncoder

라고 부른다.

3.2 디노이즈 오토엔코더 (Denoising AutoEncoder)

Vincent et al.[9]은 일부 손상된 입력에 대한 내구성(robustness)을 보장하기 위해 AE의 변화를 주었다. 즉 일부 손상된 입력에 대해서도 원 입력을 재생할 수 있도록 AE가 학습할 수 있도록 하였다. 이들은 입력 $\mathbf{x}$에 손상(destruction) 파라미터 ν에 비례하도록 νd개를 임의로 선택해 입력 값을 '0'으로 강제함으로써 입력 패턴의 선택된 입력 값을 제거하게 된다. 이러한 노이즈를 추가하는 방법 이외에도 입력 $\mathbf{x}$에 정규분포를 갖는 잡음(noise)을 원래의 벡터 값에 합하는 등 다양한 방법을 고려해 볼 수 있지만 본 논문에서는 참고문헌[9]에서 사용한 방법을 사용한다.

Fig.2.에서 보듯이, 손상된 입력 $\tilde{\boldsymbol{x}}$는 히든 표현 $\mathbf{z} = f_{\theta}(\tilde{\mathbf{x}}) = s(\mathbf{W}\tilde{\mathbf{x}} + \mathbf{b})$로 코딩되고 $\mathbf{y} = g_{\theta'}(\mathbf{z}) = s(\mathbf{W}'\tilde{\mathbf{x}} + \mathbf{b}')$로 재 생성된다. RE를 최소화할 때 손상되지 않은 원래의 $\mathbf{x}$를 기준으로

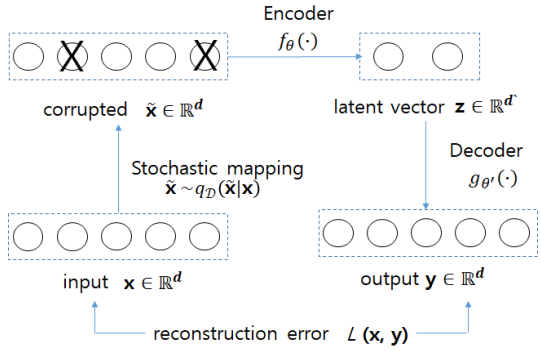

Fig. 2. Denoising AutoEncoder

AE가 학습하게 됨으로 출력 $\mathbf{y}$는 손상되지 않은 $\mathbf{x}$를 가능한 가깝게 표현하게 된다. 그러나 주요 차이점은 $\mathbf{y}$가 $\mathbf{x}$가 아닌 $\tilde{\boldsymbol{x}}$의 결정 함수이고 결과적으로 $\mathbf{x}$의 추계적(stochastic) 매핑 결과가 된다.

IV. 오토엔코더를 활용한 네트워크 이상 검출

4.1 실험 환경

AE를 설계하고 시험하기 위해, 본 연구에서는 Intel core i7-9700K 32GB RAM 64비트 윈도우즈10 개인컴퓨터를 사용하였다. 또한 NVIDA GeForce RTX 2080 8GB GPU를 사용하여 훈련 시간을 단축시켰다. 파이썬(Python) 개발 플랫폼은 아나콘다 주피터 노트북(Anaconda Jupyter Notebook) 버전 6.0.0)을 사용하고 연구에 필요한 다양한 AE 모델 개발을 위해 케라스(Keras) 버전 2.2.4-tf를 사용해 구현하였다. 한편 데이터 전처리 등 다양한 수치해석을 위해 싸이킷-런(scikit-learn) 버전 0.21.1을 사용하였다.

4.2 NSL-KDD 데이터 셋

NSL-KDD 데이터 셋은 NAD를 위한 de fecto 데이터 셋으로 지난 수 십 년 동안 전 세계적으로 사용되고 있다[2,5,6,18]. 본 논문에서는 모델 훈련을 위해 KDDTrain+에서 "정상"으로 레이블된 레코드를 추출해 80%는 훈련용으로 사용하고 나머지 20%는 모델을 검증(model fit을 위한 교차검증용)하기 위해 사용하였다. 한편 학습된 모델의 성능을 검증하기 위해, KDDTest+와 이들 데이터 셋 내 검출이 용이한 레코드를 제외한 KDDTest-21 시험 데이터 셋을 활용한다. Table 1.은 NSL-KDD 데이터 셋이 제공하는 훈련 및 시험 데이터 셋의 레코드(샘플) 수를 보여준다. 시험용 데이터 셋인 Test-21의 경우 '정상'과 '공격' 샘플 수에 있어 불균형을 이루고 있다. 따라서 Test-21의 모든 샘플을 공격이라고 극단적으로 판단하더라도 정확도(accuracy)는 81.8%를 보이게 된다. 따라서 Test-21 데이터 셋을 사용할 경우, 단순하게 정확도를 분석할 것이 아니라 ROC(Receiver Operating Characteristic) AUC(Area Under Curve) 등을 포함한 다양한 성능지표를 통해 제안한 모델을 분석하여야 한다.

Table 1. Number of records of NSL-KDD data set

	Total	Normal	Attack
KDDTrain+	125,973	67,343	58,630
KDDTest+	22,544	9,711	12,833
KDDTest-21	11,850	2,152	9,698

NSL-KDD 데이터 셋의 각 레코드들은 41개의 속성(feature) 값과 40개의 공격 타입으로 레이블되어 있다. 이들 속성 중 범주형(categorical) 속성 값은 원-핫 엔코딩(one-hot encoding)을 통해 정수 타입으로 변환하여 최종적으로 각 샘플들은 122개의 속성 값으로 변환한다. 또한 각 속성 값(x_i)들은 최소값($x_{\min}$)과 최대값($x_{\max}$)을 사용하여 다음과 같이 0~1 사이 값을 갖도록 정규화 과정을 진행한다.

$$x_i^{nor} = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \tag{3}$$

한편 본 논문에서는 KDDTest+와 KDDTest-21 데이터 셋 내 각 샘플을 정상(normal)과 공격(attack)으로 구분하는 2진 분류(binary classification)를 목표로 한다.

4.3 오토엔코더 모델링

Fig.1.에서 설명한 바와 같이 AE는 인코더와 디코드 부분으로 구분되며 다양한 네트워크 구조를 사용해 구현될 수 있다. 본 연구에서는 다음 Fig.3.과

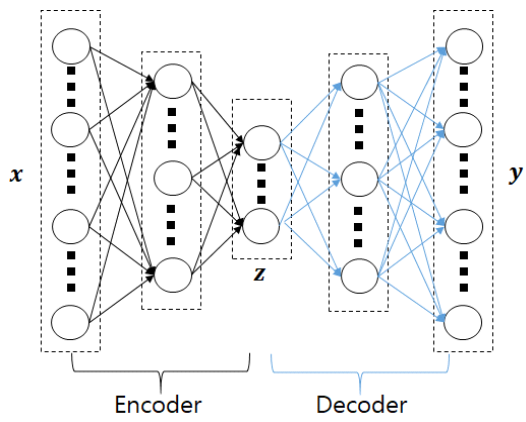

Fig. 3. Fully Connected AutoEncoder

같이 가장 기본적인 완전연결(fully connected) 구조로 모델링한다.

Fig.4.는 케라스를 이용해 입력 층(뉴런 수 122), 엔코더 히든 층(뉴런 수 64개), latent 층(뉴런 수 32개), 디코더 히든 층(뉴런 수 64개), 그리고 마지막 출력 층(뉴런 수 122개)으로 구성된 AE 모델을 보여준다.

본 논문에서 사용된 AE의 주요 하이퍼 파라미터 및 네트워크 모델 함수들은 다음과 같다.

- 입.출력 층을 제외한 층 수: 1(shallow), 3(deep), 5(stacked)
- 최적화(optimization): Adam
- 학습률(learning rate): 0.001
- 활성화 함수(activation function): relu
- 출력 층 활성화 함수: sigmoid
- 배치 크기(batch size): 1,024
- 에포크(epoch): 최대 1,000
- 조기 종료 한계(patience): 10
- 규제(regularization): L2
- 규제 강도(λ): 0, 0.01, 0.1, 0.2

대부분의 딥러닝 모델과 마찬가지로 AE는 우리가 사용하는 하이퍼 파라미터 선택에 따라 상당히 다른 성능 결과를 보이게 된다. 결국 적절한 하이퍼 파라미터 선택을 위해 많은 실험 데이터를 구축하고 상호 비교하는 작업이 필요하다. 본 논문에서는 이러한 상호 비교분석을 통해 앞에서 나열한 하이퍼 파라미터를 선택하였다. 특히, AE 모델의 층 수(입력과 출력 층은 제외)는 1, 3, 그리고 5로 구성하여 성능

```
autoencoder.summary()
Model: "model_13"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_6 (InputLayer)         [(None, 122)]             0
_________________________________________________________________
dense_24 (Dense)             (None, 64)                7872
_________________________________________________________________
dense_25 (Dense)             (None, 32)                2080
_________________________________________________________________
dense_26 (Dense)             (None, 64)                2112
_________________________________________________________________
dense_27 (Dense)             (None, 122)               7930
=================================================================
Total params: 19,994
Trainable params: 19,994
Non-trainable params: 0
_________________________________________________________________
```

Fig. 4. Keras AutoEncoder Modeling

비교 자료를 제시하였다(이하, 층수1은 shallow, 층수 3은 deep, 그리고 층수 5는 stacked로 칭함). 예를 들어, Fig.3.은 층 수가 3인 deep AE 모델이 된다. 또한 규제 강도 역시 다양한 값에 따른 성능 변화를 보여줄 것이다. 본 논문에서는 deep AE의 규제강도 λ가 0.1인 모델을 레퍼런스 모델로 선정하고 관련 하이퍼 파라미터의 변화에 따른 성능 변화를 비교 분석한다($\lambda = 0.1$ 선정에 대해서는 5.3절에서 설명). 딥 러닝 모델은 오차행렬(confusion matrix)를 통해 다양한 성능지표(metrics)를 제시할 수 있다[17]. 본 논문에서는 정확도, 정밀도, 재현율, 그리고 f1-점수를 주요 성능지표로 사용한다.

4.4 규제(Regularization)가 반영된 손실함수

훈련 기간 중에 Eq.(1)에 나타난 목적 함수를 모델 파라미터 $\theta = \{\mathbf{W}, \mathbf{b}\}$와 $\theta' = \{\mathbf{W}', \mathbf{b}'\}$에 대해 최소화하게 된다. 이때 목적함수를 다음 Eq.(4)와 같이 규제(regularization) 항을 포함시킴으로써 특정 가중치(weight)가 지나치게 큰 값을 가지지 못하도록 규제함으로써 훈련된 모델의 가중치가 과대적합(overfitting) 되지 않도록 한다. 한편 규제 파라미터 λ는 규제 강도를 결정하게 된다. 본 논문에서는 각각의 벡터에 대해 항상 유일한(unique) 값을 가지는 ι_2-규제 (가중치의 제곱에 비례하는 비용을 추가)를 사용한다[20].

$$\theta^*, \theta'^* = \underset{\theta, \theta'}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} \left\{ L\left(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\right) + \frac{\lambda}{2} \| \mathrm{W} \|^2 \right\} \tag{4}$$

V. 실험 결과

5.1 레퍼런스 모델의 재생오류 분포

NSL-KDD KDDTrain+ 훈련 데이터 셋에서 정상인 샘플들을 추출하여 AE를 학습하게 된다. 학습 시 최대 에포크 수 1,000회를 기준으로 과대적합(over-fitting)을 방지하기 위한 조기 종료(케라스 EarlyStopping)를 위해 전체 훈련 셋 중 20%를 교차검증 데이터로 활용하고, 교차검증 손실이 10회

이상 개선되지 않으면 학습을 종료한다. 한편 학습된 AE 모델을 이용해 KDDTest+ 와 KDDTest-21 시험 데이터 셋을 활용하여 정상과 이상 샘플을 2진 분류한다. 본 논문에서는 AE의 RE(Eq 2)를 NAD를 검출하기 위한 측도로 사용한다. 즉 AE의 훈련 기간 중에 학습한 정상 트래픽과 유사한 샘플들은 RE가 매우 작을 것이고 학습된 트래픽과 유사하지 않은 샘플의 경우 정상적인 latent 벡터를 만들 수 없기 때문에 매우 큰 RE를 나타낼 것이다. 다음 Fig.5.는 deep AE 그리고 규제강도 0.1을 적용한 경우, Train+의 정상 샘플들과 공격 샘플들에 대한 RE 값의 분포를 보여준다. 그림에서 보듯이 이들 RE 분포는 확연한 차이를 볼 수 있다.

Fig.6.은 학습된 AE 모델에 시험 데이터 KDDTest+와 KDDTest-21의 정상 및 공격 샘플들에 대한 RE 값의 분포를 각각 보여준다. 이들 결과 그림과 AE의 학습용 데이터 셋에 대한 결과(Fig.5.)와 비교하면 '정상'과 '공격'에 대한 RE 분포의 차이가 다소 줄어든 것을 볼 수 있지만, 여전히 두 샘플 집단에 대한 RE 분포의 차이가 존재하는 것을 재확인할 수 있다.

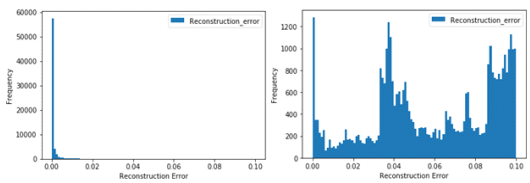

Fig. 5. Reconstruction error distribution of NSL-KDD Train+ data set (Left: Normal samples, Right: Attack samples)

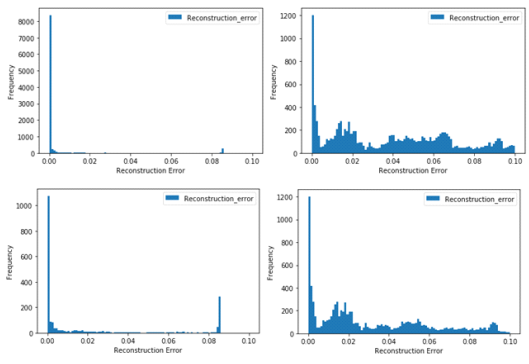

Fig. 6. Reconstruction error distribution of NSL-KDD Test+ (upper) and Test-21 (lower) data set (Left: Normal samples, Right: Attack samples)

Fig.6.에서 보듯이 상대적으로 검출이 어려운 샘플들을 모아둔 KDDTest-21의 경우, KDDTest+에 비해 '정상'과 '공격'에 대한 RE 분포의 차이가 더욱 줄어든 것을 볼 수 있다. Fig.7.은 KDDTest-21에 대한 RE 분포의 x축 범위를 줄여서 0~0.001 사이의 RE 분포를 보여준다. 그림에서 보듯이 RE가 작은 범위 내에서도 '정상' 과 '공격'에 대한 RE 분포의 차이를 일정 수준 확인할 수 있다.

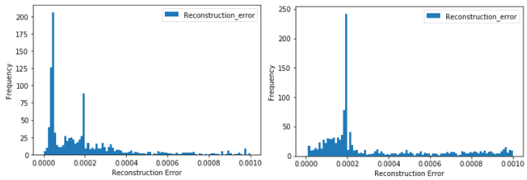

Fig. 7. Reconstruction error distribution of KDDTest-21 data set (Left: Normal samples, Right: Attack samples)

5.2 레퍼런스 모델에 대한 성능 분석

Fig.5. 와 Fig.6.에서 보듯이, AE의 RE는 정상 샘플과 이상(공격) 샘플에 대해 뚜렷한 차이를 보여준다. 본 논문에서는 해당 샘플의 RE 값이 특정 기준 값(threshold value)을 넘어서면 이상 샘플로 분류하고, 그렇지 않으면 정상 샘플로 분류한다. 따라서 이들 기준 값 설정은 제안한 딥러닝 모델의 성능을 결정하는 매우 중요한 파라미터가 된다. 본 논문에서는 AE 학습 시 사용된 정상 샘플 데이터 셋에 대한 RE 분포의 n-th 백분위수(percentile)를 기준 값으로 사용한다. 예를 들어, n=95는 훈련용 정상 샘플의 RE 분포의 95%를 포함하는 RE 값이 된다.

Fig.8.은 훈련 데이터 셋의 RE 백분위에 따른 KDDTest+에 대한 정밀도(precision), 재생율(recall), 정확도(accuracy), f1-점수(f1-score), 그리고 ROC(Receiver Operating Characteristic)의 AUC(Area Under Curve) 값을 보여준다. 백분위가 증가하면, 즉 RE 기준 값이 상승하면 정상샘플에 대한 이상 판정의 기회(오탐율 false positive rate)가 감소하는 장점이 있지만, 공격샘플에 대한 이상 판정(정탐율 true positive rate)의 기회는 줄어들게 된다. Fig.8.은

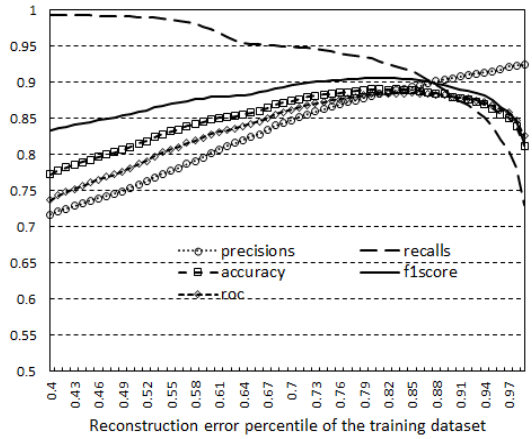

Fig. 8. Precisions, recalls, accuracies, f1-scores, and ROC AUCs on KDDTest+ with the RE percentiles of the training dataset

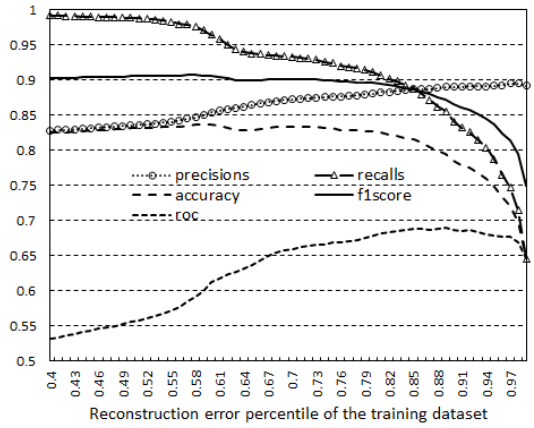

Fig. 9. Precisions, recalls, accuracies, f1-scores, and ROC AUCs on KDDTest-21 with the RE percentiles of the training dataset

RE 기준 값에 변화에 따른 이와 같은 상호절충(trade-off) 관계를 잘 보여준다. 즉 RE 기준 값이 상승하면 공격 샘플에 대한 미탐율(false negative rate)이 증가하고 정탐율은 감소함으로써 재현율은 점차 감소하게 된다. 또한 기준 값 상승에 따른, 정상 샘플에 대한 오탐율이 감소함으로써 정밀도는 점차 상승하게 된다. f1-점수는 정밀도와 재현율의 조화평균(harmonic mean)으로 RE 82-th 백분위에서 최고점을 보인 후 다시 감소하는 것을 확인할 수 있으며, 정확도와 ROC AUC 값의 변화의 추이도 f1-점수와 비슷한 형태를 보인다. 오탐에 따른 보안 관리자에게 많은 잘못된 경보를 제공할 여지는 있지만, 우리가 대부분의 공격 샘플을 검출하기를 원한다면 재현율이 높은 (상대적으로 정밀도는 낮음) 낮은 RE 백분위를 기준 값으로 설정해야 할 것이다. Table 2.는 레퍼런스 모델의 최대 f1-점수 0.90429를 보이는 백분위 82%를 기준으로 AE의 다양한 성능지표를 보여준다.

Fig.9.는 훈련 데이터 셋의 RE 백분위에 따른 KDDTest-21에 대한 다양한 성능 지표를 보여준다. RE 백분위가 낮은 기준 값에서 대부분의 성능 지표들이 좋아 보인다. 하지만 ROC AUC 값은 매우 낮은 0.6 이하 수준에 머물러 있음을 볼 수 있다(완벽한 분류기는 ROC의 AUC가 1이고, 완전한 랜덤 분류기는 0.5). 이러한 결과는 4.2절 NSL-KDD 데이터 셋에서 설명한 바와 같이 KDDTest-21 데이터 셋의 '정상'과 '공격' 샘플 수의 불균형에서 비롯된다. 결국 '정상'과 '공격'에 대해 비교적 균형 잡힌 샘플 수를 보이는 KDDTest+ 결과를 반영하여 백분위 82%를 기준으로 관련 성능 지표를 확인하는 것이 바람직하다(Table 2.).

정밀도와 재현율의 상호절충 점을 찾는 좋은 방법은 재현율에 대한 정밀도 곡선을 이용하는 것이다. Fig.10.에서 보듯이, KDDTest+와 KDDTest-21의 재현율 90% 근처에서 정밀도가 급격하게 줄어들기 시작한다. 이 하강 점 전후에서 정밀도/재현율을 상호 절충하는 것이 바람직하며 이러한 결과는 Table 2.의 결과로 보여 진다. 따라서 본 논문에서는 AE의 RE 백분위 82%를 기준 값으로 사용한다. 한편, KDDTest-21의 경우, 재현율을 감소시켜 희생시켜도 정밀도의 성능이 더 이상 개선되지 않고 일정 수준을 유지하고 있어 한계점이 존재한다(Fig.10. 참조).

Table 2. Performance of the reference AE model on KDDTest+ and KDDTest-21 data set

Dataset	Quantile	RE	Precision	Recall	Accuracy	f1-score	ROC AUC
KDDTest+	0.82	0.00071	0.88812	0.92106	0.88902	0.90429	0.88387
KDDTest-21	0.82	0.00071	0.88532	0.89555	0.81958	0.8904	0.68639

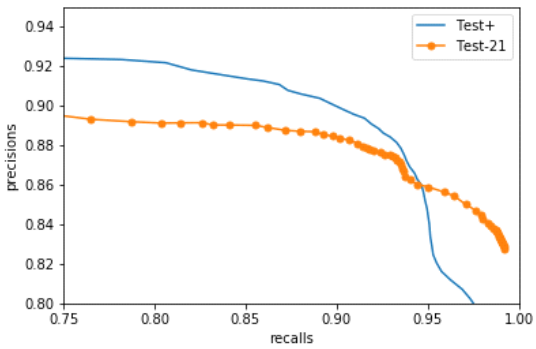

Fig. 10. Precision versus recall on KDDTest+ and KDDTest-21 datasets

5.3 하이퍼 파라미터 변화에 따른 성능 비교

Fig.11.은 AE의 훈련 데이터 셋의 RE 분포의 n-th 백분위 기준 값에 따른 shallow, deep, 그리고 stacked AEs의 이상 검출 성능에 대한 f1-점수를 보여준다. 그림에서 보듯이, shallow 네트워크와 비교해 층수 증가에 따른 deep과 stacked 네트워크 구조의 f1-점수가 상대적으로 상승하게 된다. 하지만 deep과 stacked AE의 최대 f1-점수를 비교(n=80%~90%)하면, 층수를 계속 추가한다고 해서 성능지표 향상이 효율적으로 이루어지는 것이 아님을 확인할 수 있다. 따라서 본 논문에서는 deep AE를 레퍼런스 모델로 선정하였다.

Fig.12.는 deep AE 모델에 규제 파라미터 λ의 변화에 따른 f1-점수를 보여준다. 규제를 전혀 반영

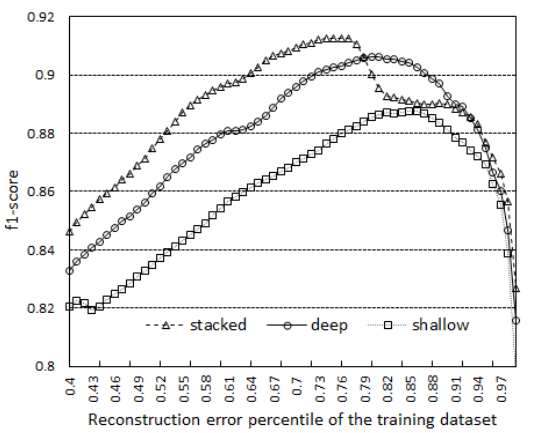

Fig. 11. f1-scores on KDDTest+ dataset for the shallow(square), deep(round), and stacket(triangle) AEs with the RE percentiles of the training dataset

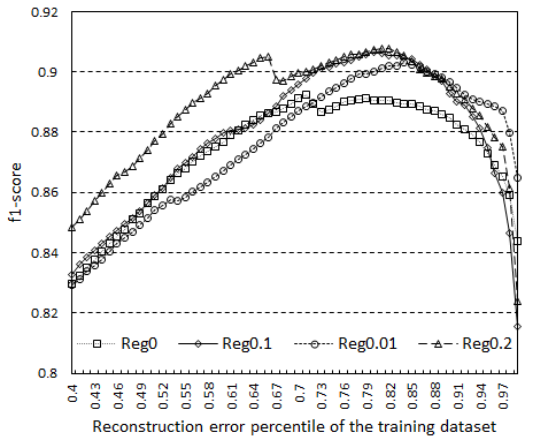

Fig. 12. f1-scores on KDDTest+ dataset for the deep AEs (λ=0(square), 0.01(round), 0.1(diamond), and 0.2(triangle)) with the RE percentiles of the training dataset

하지 않은 ($\lambda=0$) 모델과 비교해 규제가 반영된 모델이 훨씬 좋은 f1-점수(n=80%~90%)를 보임을 확인할 수 있다. 하지만 그림에서 보듯이, λ 값 0.1과 0.2의 결과를 분석하면 단순히 λ가 증가한다고 계속 성능이 향상되는 것은 아니다. 따라서 본 논문에서는 $\lambda=0.1$인 모델을 레퍼런스 모델로 선정하였다.

디노이즈 AE의 이상 검출 성능을 확인하기 위해 두 가지 비교 모델을 제시한다. 먼저, 규제가 적용되지 않은 모델과 규제가 적용된 모델을 분리하여 디노이즈 AE의 효과를 검증한다. 이러한 연구 방향은 규제와 디노이즈의 효과는 모두 모델의 과대적합을 방지하는 효과를 가져 오기 때문이다. Fig.13.은 규

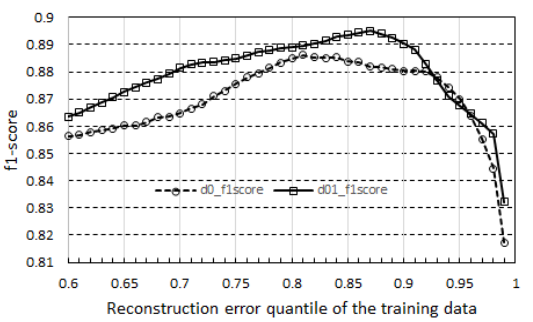

Fig. 13. f1-scores on KDDTest+ dataset for the deep AEs (λ=0, dropout=0(round points) and 0.1(square points)) with the RE percentiles of the training dataset

제가 적용되지 않은 AE 모델 입력 층에 드롭-아웃 층(dropout rate = 0.1)을 추가하여 디노이즈 효과를 추가한 모델의 f1-점수를 보여준다. 그림에서 보듯이 일부 RE 기준 값에서 1% 이상의 f1-점수가 상승하는 효과를 보인다. 하지만 Fig.14.에서 보듯이, 규제가 적용된 AE 모델은 디노이즈 영향이 오히려 성능을 저하하는 결과를 보였다. 이러한 디노이즈 적용에 따른 역효과는 규제를 통해 이미 과대적합 문제를 해결해 이상 검출 성능이 일정 수준 이상으로 높은 상황에서 디노이즈 고려는 오히려 성능을 떨어지게 한다.

Table 3.은 본 논문에서 제시된 비지도학습에 의한 AE 레퍼런스 모델과 Kwon et al.[5]이 제시한 지도학습에 의한 FCN(Fully Connected Network)과의 성능을 비교하였다. 표에서 보듯이 AE의 다양한 성능지표 값들은 FCN의 성능과 매우 유사한 수준에 있다. 이외에도 Table 3.에는 NSL-KDD 데이터 셋에 대한 대표적인 딥러닝 모델 CNN[19]과 RNN[6]에 의한 성능지표 값들을 나타내었다. 그러나 참고문헌 [6]과 [19]에서는 해당 딥러닝 모델에 대한 정확도만 제시하고 있어 본 논문에서 제시한 AE 레퍼런스 모델의 다양한 성능지표와 직접적인 비교는 불가능하지만, AE 레퍼런스 모델이 이들 딥러닝 모델에 비해 정확도가 우수한 것으로 판단된다(Table 3.에서 KDDTest+에 대한 RNN 성능지표 정밀도, 재현율, 그리고 f1-점수는 해당 논문에서 제시된 오차행렬로부터 직접 계산하여 구함).

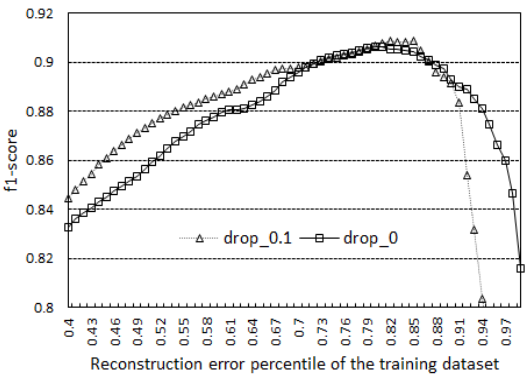

Fig. 14. f1-scores on KDDTest+ dataset for the deep AEs (λ=0.1, dropout=0(square points) and 0.1(triangle points)) with the RE percentiles of the training dataset

VI. 결론 및 토의

기계학습 및 딥러닝 기술을 개발하기 위해서는 관련 분야의 데이터 셋 확보가 먼저 선행되어야 한다. 특히, 네트워크 이상 검출과 같은 분야에 이들 개발 기술들을 직접 현장에 적용하기 위해서는 해당 네트워크에서 수집된 실제 데이터 셋이 추가적으로 확보되어야 한다. 오늘날 외부 인터넷 망과 내부 업무 망 분리에 따른 망 분리 사업과 다양한 보안장비로 보호되고 있는 실제 네트워크 환경에서 다양한 공격 시나리오가 반영된 완벽(ground-truth)하게 '정상'과 '공격'으로 레이블링된 훈련 데이터 셋을 확보하는 것은 현실적으로 매우 어렵다. 따라서 현재 진행되고 있는 대부분의 지도학습에 의한 관련 기술들을 실제 네트워크에 적용하기에는 데이터 셋 확보라는 기본적인 한계점이 존재한다.

본 논문에서는 특별한 보안 이벤트가 발생되지 않은 일상의 네트워크 트래픽을 정상 트래픽으로 가정하고 이를 기준으로 네트워크 이상 탐지를 위해 비지

Table 3. Performance of AE and the other deep learning models in the binary classification on KDDTest+ and KDDTest-21 dataset

Dataset	Model	Precision(%)	Recall(%)	Accuracy(%)	f1-score(%)
KDDTest+	FCN[5]	87.7	94.6	89.4	91.0
	AE	88.8	92.1	88.9	90.4
	CNN[19]	-	-	79.4	-
	RNN[6]	96.9	72.9	83.2	83.2
KDDTest-21	FCN[5]	87.2	92.9	83.0	90.0
	AE	88.5	89.5	81.9	89.0
	CNN[19]	-	-	61.6	-
	RNN[6]	-	-	68.5	-

도학습 오토엔코더를 활용하는 딥러닝 모델을 제시하고 NSL-KDD 데이터 셋을 사용하여 제안 모델의 네트워크 이상 검출 성능을 검증하였다. 한편, 본 논문에서는 네트워크 이상 여부를 판단하기 위해 오토엔코더의 재생 오류 값의 백분위 점수(percentile score)를 기준 값으로 사용하였다. 이때 백분위수(percentile rank)는 이상을 탐지하는 성능에 결정적인 역할을 하게 된다. 만약 높은 백분위수를 사용하게 되면 오탐율(false positive)은 급격하게 줄어들지만 미탐율(false negative)은 증가하게 된다. 만약 낮은 백분위수를 사용하면 반대의 현상이 발생하게 된다. 즉 백분위수의 선택은 상호절충(trade-off) 관계를 가진다. 따라서 본 기술을 실제 네트워크에 적용하기 위해서는 해당 도메인의 인적/물적 보안 인프라를 적절히 고려하여야 한다. 만약 우수한 보안 인프라를 보유하고 있다면, 정탐율(true positive)을 높이기 위해 낮은 백분위수를 선택하고 오탐을 포함한 더 많은 보안 경고 메시지를 분석해야만 할 것이다. 본 논문에서는 주요 하이퍼파라미터 선정 과정을 통해 오토엔코더 레퍼런스 모델을 제안했으며, 백분위수 82th 백분위를 기준으로 KDDTest+와 KDDTest-21 시험 데이터 셋에 대해 각각 90.4% 그리고 89.0%의 f1-점수를 보였다. 따라서 본 논문에서 제시된 비지도학습에 의한 오토엔코더 모델은 네트워크 보안 분야에 실제적으로 적용 가능한 딥러닝 기술 사례가 될 수 있을 것이다.

본 논문에서는 오토엔코더의 자기 학습(self learning)에 사용되는 특별한 보안 이벤트가 발생하지 않은 일상의 네트워크 트래픽을 모두 정상으로 가정하고 있다. 하지만 현실적으로는 이상 트래픽이 포함될 수 있기 때문에 이에 대한 추가적인 분석이 필요하다. 즉 전체 트래픽 중에서 일부 이상 트래픽이 존재하는 환경을 구성하여 실험을 진행하고 있다. 또한 오토엔코더 재생오류 분포의 수학적 모델링을 통한 이상 여부를 판단하는 기준 값 설정 방법을 연구 중에 있다. 일부 기존 연구들이 재생오류를 최댓값 혹은 가우시안 분포로 모델링하여 사용하는 경향이 있으나 이에 대한 근거가 절대적으로 부족한 상황이다. 본 연구진은 이들 재생오류가 파워-로(power-law) 분포 즉 heavy-tail 분포로 모델링되는 것을 확인하였으며 재생오류 기준 값 설정을 위한 수학적 접근을 진행하고 있다.

References

[1] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F.E. Alsaadi, "A Survey of Deep Neural Network Architectures and Their Applications," Neurocomputing, vol. 234, pp. 11-26, Apr. 2017.

[2] M. Ahmed, A.N. Mahmood, and J. Ju, "A survey of network anomaly detection techniques," Journal of Network and Computer Applications, vol. 60, pp. 19-31, Jan. 2016.

[3] M. Tavallaee, E. Bagheri, W. Lu, and A.A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," Proceedings of the 2009 IEEE Symposium on Computational Intelligence, pp. 1-6, Jul. 2009.

[4] NSL-KDD dataset, Available on: https://www.unb.ca/cic/datasets/nsl.html, Mar. 2009.

[5] D. Kwon, H. Kim, J. Kim, S.C. Suh, I. Kim, and K.J. Kim, "A survey of deep learning-based network anomaly detection," Cluster Computing, vol.27, pp. 949-961, Jan. 2019.

[6] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," IEEE Access, pp. 21954-21961, Oct. 2017.

[7] J.J. Davis and A.J. Clark, "Data preprocessing for anomaly based network intrusion detection: A review," Computers & Security, vol. 30, no. 6-7, pp. 353-375, Sep. 2011.

[8] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptron and singular value decomposition," Biological cybernetics, vol. 59, no. 4-5, pp. 291-294, Sep. 1988.

[9] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, "Extracting and

Composing Robust Features with Denoising Autoencoders," Pro. of the 25th International Conference on Machine Learning, pp. 1096-1103, Jul. 2008.

[10] A. Borghesi, A. Bartolini, M. Lombardi, M. Milano, and L. Benini, "Anomaly detection using autoencoders in high performance computing systems", In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 9428-9433, Jul. 2019.

[11] M. Sakurada and T. Yairi, "Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction," Proc. of MLSDA'14, pp. 4-11, Dec. 2014.

[12] Z. Chen, C.K. Yeo, B.S Lee, and C.T. Lau, "Autoencoder-based Network Anomaly Detection," In 2018 Wireless Telecommunications Symposium, pp. 1-5, Apr. 2018.

[13] F. Farahnakian and J. Heikkonen, "A deep auto-encoder based approach for intrusion detection system," Proceedings of the 20th International Conference on Advanced Communication Technology, pp. 178-183, Feb. 2018.

[14] C. Ieracitano, A. Adeel, M. Gogate, K. Dashtipour, F.C. Morabito, H. Larijani, and A. Hussain, "Statistical analysis driven optimized deep learning system for intrusion detection," Proceedings of the International Conference on Brain Inspired Cognitive Systems, pp. 759-769, Jul. 2018.

[15] R.C. Aygun and A.G. Yavuz, "Network Anomaly Detection with Stochastically Improved Autoencoder Based Models," Proc. of 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing, pp. 193-198, Jun. 2017

[16] A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow: Concepts, tools, and techniques to build intelligent systems, 2nd Edition, O'Reilly Media, 2019.

[17] K. Kang, "Decision Tree Techniques with Feature Reduction for Network Anomaly Detection," Journal of the Korea Institute of Information Security and Cryptology, 29(4), pp. 795-805, Aug. 2019.

[18] A. Ozgur and H. Erdem, "A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015," PeerJ Preprints, vol. 4, Art. no. e1954, Apr. 2016.

[19] D. Kwon, K. Natarajan, S.C. Suh, H. Kim, and J. Kim, "An Empirical Study on Network Anomaly Detection Using Convolutional Neural Networks," Proceedings of the IEEE 38th International Conference on Distributed Computing Systems, pp. 1595-1598, Jul. 2018.

[20] C. Zhou and R.C. Paffenroth, "Anomaly Detection with Robust Deep Autoencoders," Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 665-674, Aug. 2017.

〈저 자 소 개〉

강 구 홍 (Koohong Kang) 정회원
1985년 8월: 경북대학교 전자공학과 졸업
1990년 2월: 충남대학교 전자공학과 석사
1998년 2월: 포항공과대학교 전자계산학과 박사
1985년 9월~1999년 3월: 한국전자통신연구원 선임연구원
2008년 1월~2009년 2월: Purdue University Visiting Scholar
2016년 1월~2017년 2월: 제주대학교 방문교수
2000년 9월~현재: 서원대학교 정보통신공학과 교수
〈관심분야〉 성능분석, 네트워크 보안, 머신 러닝