LID-DS Report 11/22

LID-DS(2021) Unsupervised (1)

LID-DS Dataset 설명

LID-DS는 호스트 기반 침입 탐지(Host Based Intrusion Detection System, HIDS) 데이터 셋으로 Ubuntu 18.04버전에서 기록이 된 데이터이다. Attack Type은 15가지가 있으며,각 공격 유형별로 시나리오 폴더가 존재하고, 폴더안의 .sc파일을 통해 로그를 확인 가능하다. 이번 연구에서는 데이터의 크기로 인하여 'Bruteforce_CWE-309', 'CVE-2014-0160' 2가지 공격에 대해서만 이상탐지를 진행하였다.

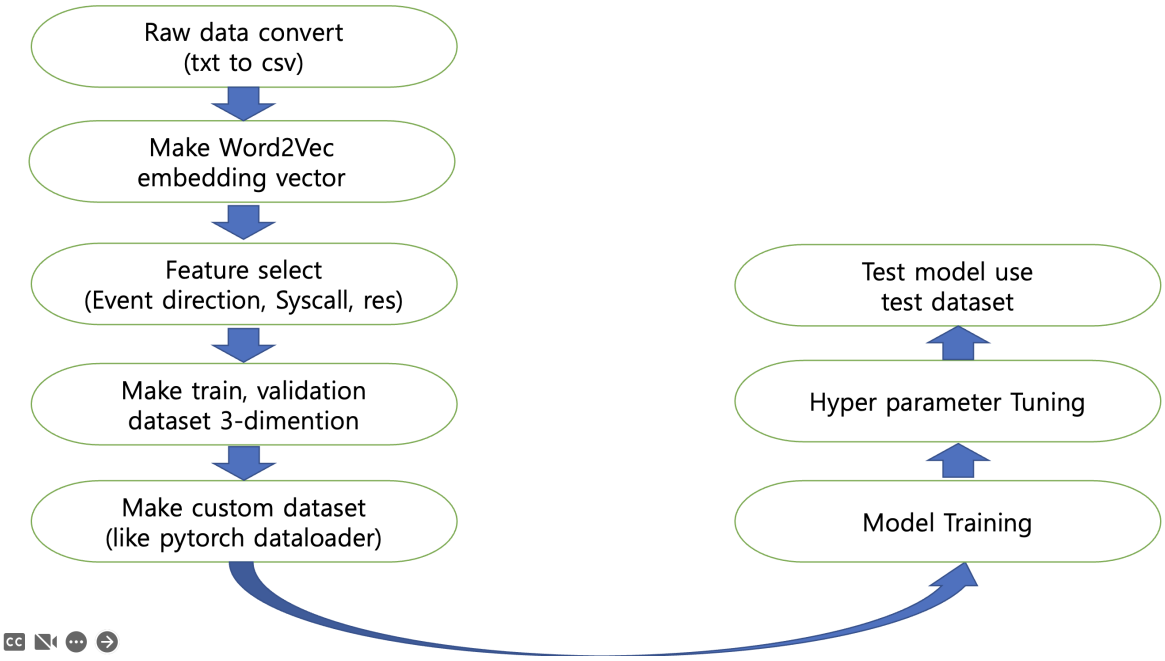
Attack Type	Train Scenario	Validation Scenario	Test Scenario (Contain Attack)	Test Scenario (Only Normal)	Data Size (모든 파일 압축되어 있음)
Bruteforce_CWE-309	210	60	139	768	1.3GB
CVE-2012-2122	210	60	120	762	0.69GB
CVE-2014-0160	210	60	120	758	0.39GB
CVE-2017-7529	210	52	119	729	5.7GB
CVE-2017-12635_6	208	58	119	753	12.6GB
CVE-2018-3760	210	60	122	737	4GB
CVE-2019-5418	210	60	120	760	7.3GB
CVE-2020-9484	210	60	120	759	5.2GB
CVE-2020-13942	211	60	118	756	15.3GB
CVE-2020-23839	210	57	120	760	1.4GB
CWE-89-SQL-injection	210	60	120	760	3.7GB
EPS_CWE-434	210	60	121	759	10GB
Juice-Shop	210	60	121	760	7.3GB
PHP_CWE-434	210	59	118	760	3.8GB
ZipSlip	210	59	118	760	8.7GB

원본 데이터의 특성은 총 8개로 구성되어 있고 정보는 아래의 표와 같다.

Feature	Information
Event_time	Event가 발생한 시간
cpu	thread id를 나타냄
user_uid	해당 프로세스를 실행한 사용자의 정보
process	실행된 프로세스의 정보
process_id	프로세스 ID
event_type	호출된 System Call

event_direction	프로세스 호출, 반환을 나타냄(호출 : >, 반환 : < 으로 표시)
event_argument	System Call의 인자 (ex. res, fd..)

학습 구성도



주어진 데이터는 .txt파일로 되어있다. 데이터를 좀 더 편리하기 다루기위해 Pandas를 이용하여 .txt파일을 csv 파일로 변환하는 과정을 거쳤다.

System Call이 LID-DS에서 공격을 탐지하는데에 있어 중요한 특성이기 때문에, Gensim의 Word2Vec Library를 사용하여 Embedding vector를 생성한다. 각 Scenario의 System call sequence를 하나의 문장으로 보고 그것들을 모아서 Word2Vec의 학습 데이터로 사용한다. Word2Vec에 사용한 데이터는 2가지 공격 유형의 시나리오 파일들을 모두 사용하여 총 420개의 시나리오를 사용하였다.('Bruteforce_CWE-309', 'CVE-2014-0160')

Event argument 특성을 사용하기위하여 특성값을 파싱하고, 값을 불러오기 위한 "extract_params"를 만들어 System call의 반환 값인 "res"특성을 사용하였다.

Convoultion 1D의 입력데이터는 3차원이므로 데이터를 3차원으로 변환하여 주는 함수인 "making_model_input" 함수를 만들어주었다. Sequence의 길이는 30으로 고정하고 전처리를 진행하였다.

결론적으로 총 사용한 특성은 System call, Event direction, return value 3개의 특성이다. "res"특성의 스케일을 조정해주기 위하여 MinMaxScaler를 사용하였으며, event_direction은 2가지의 값을 가져 OneHotEncoding을 진행 하고, 하나의 행을 삭제하였다.(LabelEncoding과 같은 의미) 결론적인 입력데이터 차원은 System call embedding vector size에 2를 더한 값이다.

학습데이터는 각 공격당 30개의 시나리오를 사용하여 총 60개의 시나리오를 사용하였고, 검증데이터는 각 공격당 10개의 시나리오를 사용하여 총 20개의 시나리오를 사용하였다.

사용할 모델 구조는 Denosing Autoencoder이고, 오토인코더는 데이터가 들어오면 데이터를 축소시키는 인코더(Encoder)를 거쳐 잠재 벡터(Latent vector)를 만들어 데이터를 주요한 특성만 가지도록 압축하고, 압축한 벡터를 디코더(Decoer) 층을 거쳐 다시 복원하는 것이 목적인 모델 구조이다. 오토인코더는 데이터의 차원을 활성화 함수(Activation functino)에 비선형 활성화 함수를 사용하여 비선형 차원 축소가 가능하여 차원 축소에

도 많이 이용되는 방법이다. 인코더 층에 노이즈를 넣은 후 복원화는 과정을 거치는 모델이 Denosing Autoencoder인데, 이 구조는 Autoencoder 모델에 비해 일반화 성능이 더 뛰어나다는 장점이 존재한다.오토인코더의 잠재벡터를 사용하여 이상 시나리오를 분류하는 방식이 아닌. 정상 시나리오만을 사용하여 오토인코더가 학습을 한다면 이상 시나리오를 복구할 때 복원오차가 증가한다는 성질을 이용하여 이상 시나리오를 탐지하였다.

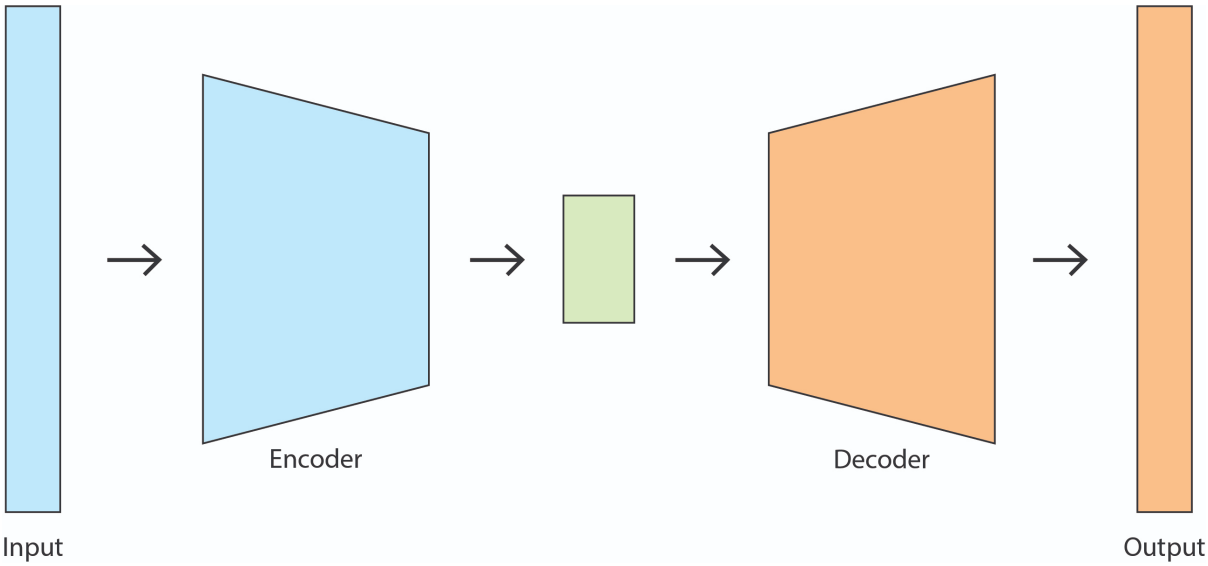


Figure 1. Autoencoder Structure

사용한 모델은 Convolution1D Denoising Autoencoder이고 Convoution1D

모델은 Convolution 1D로 구성되어 있기 때문에, 입력데이터는 3차원의 구조를 띄어야한다. 첫번째 요소는 배치크기, 두번째 요소는 시퀀스 길이, 세번째 요소는 특성의 개수를 의미한다. 모델에 알맞게 넣어주기위해 특성선택이 진행된 데이터를 3차원으로 변환하는 과정을 거쳤다.

데이터의 크기가 상당히 크기 때문에, Pytorch의 Dataloader와 비슷한 기능인 Keras의 Sequence 라이브러리를 사용하여 데이터를 로드하였다.

선정 모델은 Conv1D Denoising autoencoder이다. Convoution1D 레이어를 사용하여 모델이 연속된 시스템콜의 시계열 특성을 반영할 수 있게 하였다. 첫번째 모델은 잠재벡터를 나타내는 Dense층을 사용하지 않았고, 두번째 모델은 잠재벡터를 나타내는 Dense층을 사용하여 실험을 진행하였다. 노이즈를 생성하는 역할을 하는 Dropout의 비율은 0.3, 첫번째 모델의 노드 수는 [64-32-16-16-32-64], 두번째 모델의 노드 수는 [64-32-16-8-16-32-64]로 고정하고 실험을 진행하고, Word2Vec의 Embedding Vector크기를 조정하면서 실험을 진행하였다.

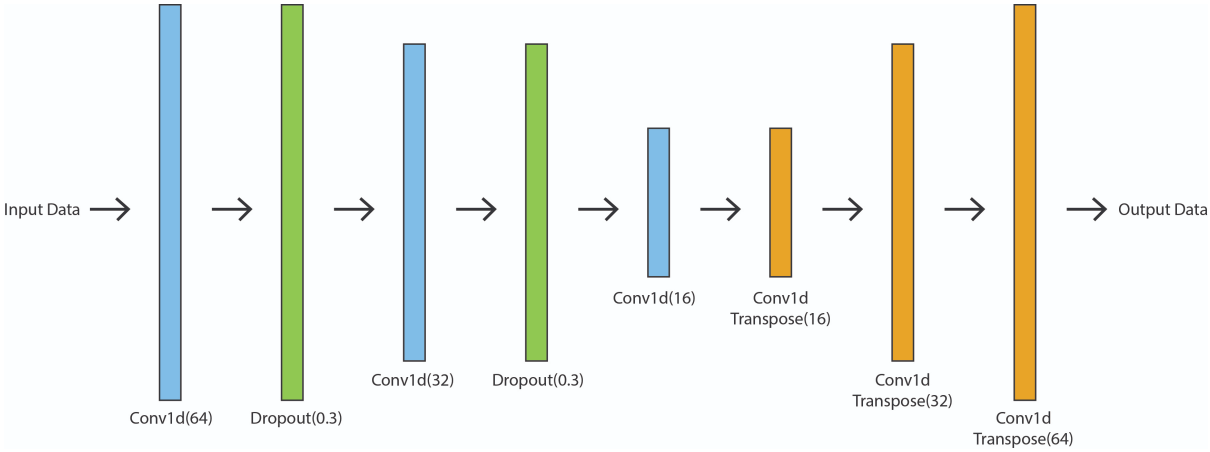


Figure 2. Conv1d Denosing Autoencoder No dense

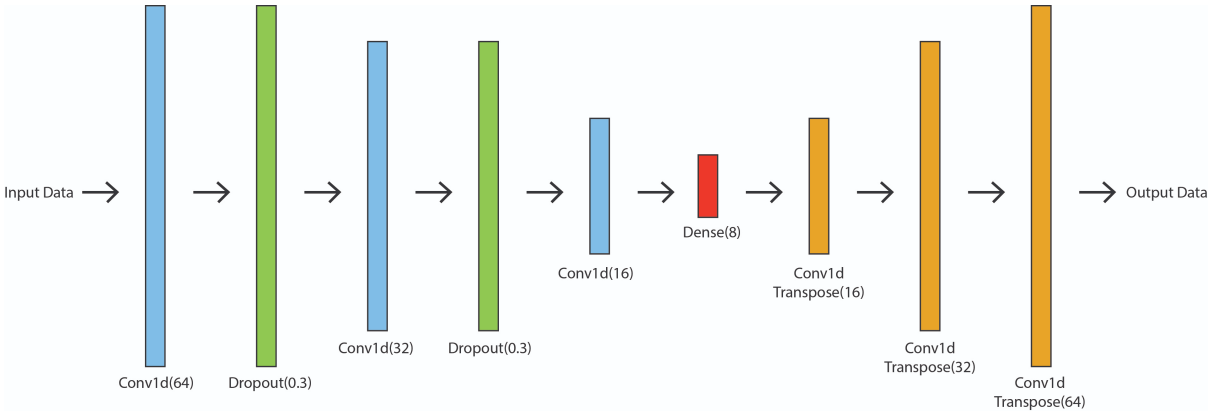


Figure 3. Conv1d Denosing Autoencoder with Dense

배치사이즈는 32, 30에폭동안 학습을 진행하였다. 옵티마이저는 Adam을 사용하였고 초기 학습률은 0.001로 정하였다. He normal 방법으로 가중치를 초기화하여 주었고 활성화함수는 sigmoid와 같은 함수에서 기울기 소실 문제와 relu의 Dying relu 문제를 해결한 leaky relu를 사용하였다. 손실함수는 MSE를 사용하였고, 검증 오차를 기준으로 조기종료를 사용, 학습률 스케줄러인 ReduceOnLRPlateau를 사용하여 과대적합을 피하였다. 학습 도중 가장 최적의 모델을 저장하기 위해 ModelCheckPoint를 사용하여 과적합이 발생하기 전 최적의 모델을 저장하였다. 정리한 모델 파라미터는 아래와 같다.

- Batch size = 32
- Epoch = 30
- Conv1d Layer kernel size = 3
- Optimaizer = adam
- Initial learning rate = 1e-3
- Kernel initializer = He initializer
- Activation function = leaky relu
- Loss function = MSE

- EarlyStopping
  - patience = 5
  - monitor = val_loss
- Learning rate scheduler = ReduceOnLRPlateau(patience = 5)
- ModelCheckPoint
  - monitor = val_loss

원본 데이터는 2차원이기때문에 복원한 3차원 데이터를 2차원으로 복원하는 과정을 거쳤고,원본 데이터와 MSE로 재구현 오차를 계산한다. 그리고 시계열 데이터의 특성을 반영하기 위하여 이동평균으로 error smoothing 작업을 진행한 후 LID-DS 데이터에서 명시된 공격 시작 시점 이후에 복원오차가 일정 임계값 이상을 나타낸다면 해당 시나리오는 공격을 포함한 시나리오라고 판단하였다. 정상 데이터는 임계치 이상의 오차가 발생한다면 오탐이라고 판정하였다.

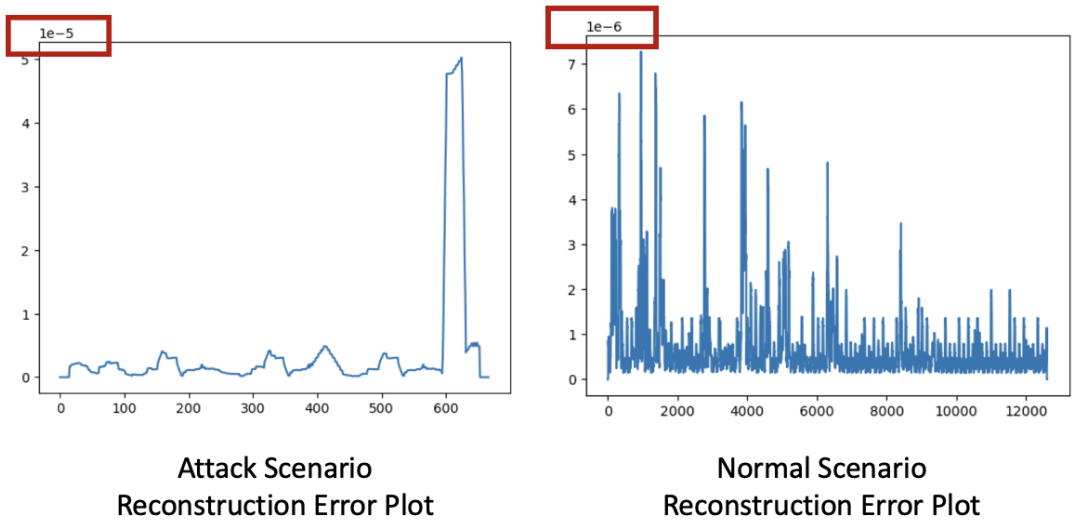


Figure 4. Reconstruction Error Plot

테스트 데이터는 공격이 포함된 시나리오 100개, 정상만을 포함하는 시나리오 300개를 사용하여 총 400개의 시나리오를 테스트 데이터로 사용하였다. 테스트 데이터의 비율은 1:3이므로, 균형을 이루고 있다고 보기 힘들기 때문에 Accuracy는 성능평가에 적합하지 않다고 판단하여 Recall과 Precision의 조화평균인 F1 score를 사용하여 성능을 평가하였다. 결과는 아래에 표에 정리하였다.

실험결과

Model	Embedding Vector Size	Threshold	F1-score (Bruteforce)	F1-score (Heartbleed)	Avg. F1-score
Conv1d DAE(Use	10	1e-4	76.4%	79.7%	78.1%

Dense)					
Conv1d DAE(Use Dense)	20	1.1*1e-4	81.8%	79.4%	80.6%
Conv1d DAE(Use Dense)	30	4*1e-5	79.6%	78.7%	79.2%
Conv1d DAE(Use Dense)	40	1.05*1e-4	79.0%	78.7%	78.9%
Conv1d DAE(Use Dense)	50	4*1e-5	**81.8%**	**79.7%**	**80.8%**

Model	Embedding Vector Size	Threshold	F1-score (Bruteforce)	F1-score (Heartbleed)	Avg. F1-score
Conv1d DAE(No Dense)	10	9.2*1e-5	**81.8%**	79.1%	80.5%
Conv1d DAE(No Dense)	20	2.95*1e-4	80.3%	**81.1%**	**80.7%**
Conv1d DAE(No Dense)	30	3*1e-5	79.2%	24.2%	51.7%
Conv1d DAE(No Dense)	40	9*1e-5	81.1%	79.7%	80.4%
Conv1d DAE(No Dense)	50	2*1e-5	75.0%	17.6%	46.3%

참고문헌

[1] 박대경, et al. "LID-DS 데이터 세트를 사용한 기계학습 알고리즘 비교 연구." *정보처리학회논문지/소프트웨어 및 데이터 공학 제* 10.3 (2021): 3.

[2] 박대경, et al. "Few-Shot Learning 을 사용한 호스트 기반 침입 탐지 모델." *정보처리학회논문지. 소프트웨어 및 데이터 공학* 10.7 (2021): 271-278.

[3] Kim, Czangyeob, et al. "Intrusion detection based on sequential information preserving log embedding methods and anomaly detection algorithms." *IEEE Access* 9 (2021): 58088-58101.

[4] 김미르, 계효선, 권민혜. (2022). 오토인코더 모델의 은닉층 정보를 활용한 네트워크 이상탐지 시스템. 한국통신학회논문지, 47(9), 1310-1321.

[5] 강건하, 손정모, 심건우. (2021). 오토인코더를 사용한 이상탐지 모델의 비교분석 및 이상치 판별 기준 제안. 한국컴퓨터정보학회논문지 , 26(8), 23-30.

[6] 안주현, 윤태준, 김남기, 박준표, 왕지남. (2022). CNN, LSTM기반 오토인코더를 이용한 공정 사이클 이상 패턴 탐지. 대한산업공학회 춘계공동학술대회 논문집, (), 4014-4019.

[7] 서재현. (2017). 기계학습 방법에 기반 한 불균형 침입탐지 데이터 분류법의 성능평가에 관한 연구. 한국지능시스템학회 논문지, 27(5), 466-474.

[8] 서재현. (2018). 딥러닝 기반 불균형 침입탐지 데이터 분류에 관한 비교 연구. 한국지능시스템학회 논문지, 28(2), 152-159.

[9] 김도우, 구명완. (2017). Doc2Vec과 Word2Vec을 활용한 Convolutional Neural Network 기반 한국어 신문 기사 분류. 정보과학회논문지, 44(7), 742-747.

[10] 심재승, 이재준, 정이태, 안현철. (2020). 워드 임베딩을 활용한 한국어 가짜뉴스 탐지 모델에 관한 연구. 한국컴퓨터정보학회 학술발표논문집 , 28(2), 199-202.