

Deep Learning based Feature Extraction for Network Intrusion Detection

Samyuktha M, Cryptology and Computing Research, SETS



Strategy and Synergy for Security

February 10, 2022

Contents

- 1 Intrusion Detection Systems
- 2 Feature Extraction using Deep Learning
- 3 Auto-Encoders and Sparse Auto-Encoders
- 4 Experimentation: Dataset used
- 5 Experimentation: Steps in detail and Results

- 1 Intrusion Detection Systems
- 2 Feature Extraction using Deep Learning
- 3 Auto-Encoders and Sparse Auto-Encoders
- 4 Experimentation: Dataset used
- 5 Experimentation: Steps in detail and Results

Intrusion Detection Systems

- Used to **detect unauthorized access**, malicious activity into computer systems and networks
- Intrusion detection as a technology is **not new**, it has been used for generations to defend valuable resources
- IDSes can detect and deal with **insider attacks**, as well as, external attacks, and are often very useful in detecting violations of corporate security policy and other internal threats

Why IDS / IPS is needed

a system to uncover malicious/unwanted activity on your network by inspecting the network traffic

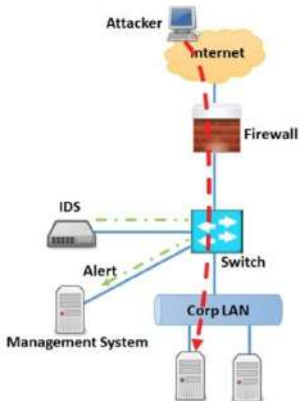
Intrusion Detection System

- Passive, it only looks and alerts the admin
- Compare to a security camera
- Works with a copy of network traffic

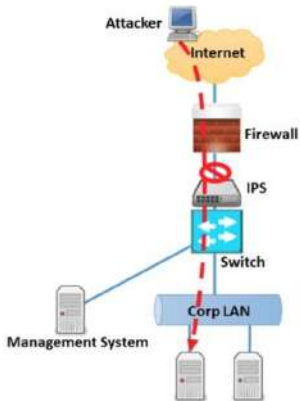
Intrusion Prevention System

- Active, tries to prevent malicious activity
- Compare to a security checkpoint
- Deployed next to router or firewall

Intrusion Detection System



Intrusion Prevention System



Goal of IDS

- Monitors network resources to detect intrusions and attacks that were **not stopped by preventative techniques** like firewalls, packet-filtering routers, proxy servers.
- To detect attacks as they happen
- To provide information about the attacks that have passed into the network

Types of IDS

- Network Based Intrusion Detection System : Network-based IDS log their activities and report or alarm on questionable events
- Host Based Intrusion Detection System : Detecting malicious activities on a single computer ; watches only specific host activities

Types of Detection Techniques in IDS

- Signature Based IDS
- Anomaly Based IDS
- Hybrid Detection

Signature Based IDS

- Network traffic is examined for pre-configured and **pre-determined attack** patterns known as signature.
- Signatures are **easy to develop** and understand if you know what network behavior you're trying to identify.
- IDS analyzes information it gathers and compares it to a database of known attacks, which are identified by their individual signatures
- It can be **very accurate**.
- **Highly effective** towards well known attack

Signature Based IDS - Pros Cons

Pros:

- Highly effective towards well known attacks.
- Low false positive rates

Cons:

- New vulnerabilities and exploits will not be detected until administrators develop new signatures
- Very large and it can be hard to keep up with the pace of fast moving network traffic.
- Can be Bypassed by changing the signature of attacks

Anomaly Based IDS

- Use network traffic baselines to **determine a “normal” state** for the network traffic and compare current traffic to that baseline.
- **Does not requires signatures** to detect intrusion.
- A **new attack** for which a signature doesn't exist can be detected
Cons: False Positive Rate is high

Contents

- 1 Intrusion Detection Systems
- 2 Feature Extraction using Deep Learning
- 3 Auto-Encoders and Sparse Auto-Encoders
- 4 Experimentation: Dataset used
- 5 Experimentation: Steps in detail and Results

Feature Extraction

- **Feature** is an attribute that has an **impact on a problem** or is useful for the problem
- Dataset consists of noisy data, irrelevant data, and some part of useful data. With noise and irrelevant data, the model may not predict and perform well.
- Moreover, the huge amount of data **slows down** the training process of the model
- **Causes overfitting** hence does not enhance generalization

Supervised Learning

Feed Forward Neural Networks

Convolutional Neural Networks

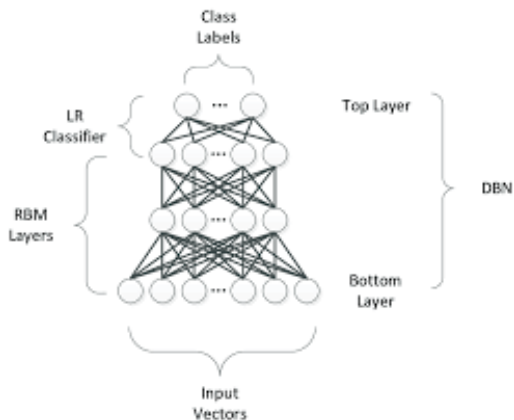
Recurrent Neural Networks

Unsupervised Learning

Auto-Encoders

Restricted Boltzmann Machines

Deep Belief Networks

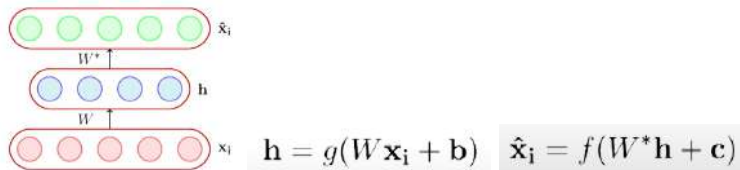


Contents

- 1 Intrusion Detection Systems
- 2 Feature Extraction using Deep Learning
- 3 Auto-Encoders and Sparse Auto-Encoders**
- 4 Experimentation: Dataset used
- 5 Experimentation: Steps in detail and Results

Auto-Encoders

- Very much like a feed forward neural network
- Has two phases: Encoding and Decoding Phase
- Encodes an input x_i into a hidden representation 'h'

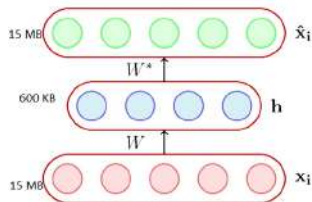


- Decodes the input from the hidden representation h
- The model is trained to reduce a certain loss function that will ensure that x_i is close to \hat{x}_i

Auto-Encoder Types

$$\dim(\mathbf{h}) < \dim(\mathbf{x}_i)$$

- If we are still able to reconstruct \mathbf{x}_i from \mathbf{h} what does it tell about \mathbf{h} ?
- \mathbf{h} is a loss-free encoding of \mathbf{x}_i ;
- Such an auto-encoder is called Under-Complete auto-encoder



$$\mathbf{h} = g(W\mathbf{x}_i + \mathbf{b})$$

$$\hat{\mathbf{x}}_i = f(W^*\mathbf{h} + \mathbf{c})$$

Loss Function of Auto-encoder

- The objective function of the auto-encoder is to reconstruct \hat{x}_i to be close to x_i as possible

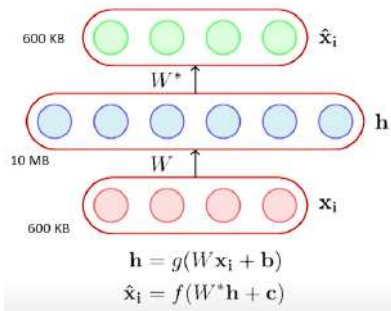
$$\min_{w, w^*, c, b} \frac{1}{m} \sum_{i=1}^m \sum_{j=1}^n (\hat{x}_{ij} - x_{ij})^2$$

- Train the auto-encoder like a normal feed forward neural network

Sparse Auto-Encoder

- A hidden neuron with sigmoid function will have values between 0 and 1
- Neuron is **activated** when its output is close to 1 and **not activated** when its output is close to 0
- A sparse auto-encoder tries to **ensure the neuron is inactive most of the times**
- In other words, **the average activation function** of the neuron is close to 0

Sparse Auto-Encoder



$$\hat{\rho}_l = \frac{1}{m} \sum_{i=1}^m h(\mathbf{x}_i)_l$$

Sparse Auto-Encoder

- If the **neuron 'l' is sparse** (i.e mostly inactive), then $\hat{\rho}_l$ **tends to 0**
- A sparse auto-encoder uses a **sparsity parameter** ρ (close to 0) and tries to enforce the constraint $\hat{\rho}_l = \rho$
- whenever it is **active**, it is going to find relevant information
- We have to ensure $\hat{\rho}_l = \rho$, one way of doing this is

$$\Omega(\theta) = \sum_{l=1}^k \rho \log \frac{\rho}{\hat{\rho}_l} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_l}$$

$$\mathcal{L}(\theta) = \mathcal{L}'(\theta) + \Omega(\theta)$$

- This is also known as the Kullback-Liebler divergence (KL)

Contents

- 1 Intrusion Detection Systems
- 2 Feature Extraction using Deep Learning
- 3 Auto-Encoders and Sparse Auto-Encoders
- 4 Experimentation: Dataset used**
- 5 Experimentation: Steps in detail and Results

Dataset used

- Most commonly used datasets for intrusion detection are [KDD99 dataset](#) and [NSL-KDD](#)
- NSL-KDD dataset has been selected

- Cleaned up version of the [KDD99 dataset](#)
- Knowledge Discovery and Data mining Tools competition in 1999 to collect large number of traffic records
- The data set contained a total of 125973 training samples and 22543 test samples,

Features

- Contains 43 features per record
- With [41 of the features referring to the traffic input](#) itself and the [last two](#) are [labels](#) (whether it is a normal or attack) and [Score](#) (the severity of the traffic input itself).

- **Denial of service attacks (DOS)**: tries to shut down traffic flow to and from the target system. The IDS is flooded with an abnormal amount of traffic, which the system can't handle
- **Probe or surveillance**: tries to get information from a network
- **User to root attacks(U2R)**: starts off with a normal user account and tries to gain access to the system or network, as a super-user (root)
- **Remote to Local attacks(R2L)**: tries to gain local access to a remote machine

Sub - Classes

Classes:	DoS	Probe	U2R	R2L
Sub-Classes:	<ul style="list-style-type: none">• apache2• back• land• neptune• mailbomb• pod• processtable• smurf• teardrop• udpstorm• worm	<ul style="list-style-type: none">• ipsweep• mscan• nmap• portsweep• saint• satan	<ul style="list-style-type: none">• buffer_overflow• loadmodule• perl• ps• rootkit• sqlattack• xterm	<ul style="list-style-type: none">• ftp_write• guess_passwd• httptunnel• imap• multihop• named• phf• sendmail• Snmpgetattack• spy• snmpguess• warezclient• warezmaster• xlock• xsnoop
Total:	11	6	7	15

More than half of the records that exist in each data set are normal traffic and the distribution of U2R and R2L are extremely low.

Other Features

- **Intrinsic:** derived from the header of the packet without looking into the payload itself and holds basic information about the packet. This category contains [features 1–9](#).
- **Content:** hold information about the original packets, as they are sent in multiple pieces rather than one. This category contains [features 10–22](#).
- **Time-based:** hold the analysis of the traffic input over a two-second window and contains information like how many connections it attempted to make to the same host. These features are mostly counts and rates This category contains [features 23–31](#).
- **Host-based:** similar to Time-based features, except instead of analyzing over a 2-second window, it analyzes over a series of connections made (how many requests made to the same host over x-number of connections). This category contains [features 32–41](#).

Contents

- 1 Intrusion Detection Systems
- 2 Feature Extraction using Deep Learning
- 3 Auto-Encoders and Sparse Auto-Encoders
- 4 Experimentation: Dataset used
- 5 Experimentation: Steps in detail and Results**

- Feature value of `Num_outbound_cmds` is **all 0**, which has no effect on the classification process, so this feature is **removed**.
- Since the input of the SSAE network is a **numeric matrix**, we need to transform the symbolic features into numerical features
- In order **to facilitate comparison**, the original feature values are subjected to a **maximum-minimum normalization process** so that the feature values are in the same order of magnitude

Impact of number of Layers and Neurons in each Layer

Hidden Layer Structure	ACC(%)	DR(%)	FAR(%)	T_{train} (s)
[110,95,70,55,30,15]	99.32 \pm 0.096	99.27 \pm 0.089	0.096 \pm 0.052	23.04
[105,90,70,55,30]	99.17 \pm 0.103	99.02 \pm 0.114	0.109 \pm 0.054	11.03
[100,80,60,40,20]	99.01 \pm 0.098	98.92 \pm 0.121	0.117 \pm 0.032	10.32
[100,80,50,30]	98.63 \pm 0.101	98.32 \pm 0.120	0.131 \pm 0.021	5.03
[105,80,45,20]	98.54 \pm 0.109	98.15 \pm 0.125	0.136 \pm 0.024	5.26
[90,60,30]	93.39 \pm 0.259	93.12 \pm 0.364	0.158 \pm 0.019	3.19
[85,50,20]	92.78 \pm 0.234	93.01 \pm 0.318	0.164 \pm 0.038	2.98
[75,30]	87.97 \pm 0.198	87.32 \pm 0.256	0.305 \pm 0.029	2.18
[80,40]	88.74 \pm 0.218	88.04 \pm 0.187	0.289 \pm 0.259	2.25
[65]	84.92 \pm 0.356	83.78 \pm 0.321	0.351 \pm 0.298	1.45
[60]	86.12 \pm 0.328	85.93 \pm 0.315	0.348 \pm 0.384	1.37

Model Comparison

- Significant reduce in the training time and testing time of the classifiers.
- The results demonstrate that the SSAE can almost retain all the amount of information contained in the original data while learning the high-level representation of features

Method	ACC(%)	DR(%)	FAR(%)	Training Time(s)	Testing Time(s)
SVM	99.02 ± 0.135	98.97 ± 0.168	0.156 ± 0.054	73.96	59.20
KNN	98.93 ± 0.214	98.32 ± 0.237	0.171 ± 0.066	78.54	63.85
RF	99.13 ± 0.169	98.94 ± 0.187	0.153 ± 0.049	68.30	51.39
SSAE+SVM	99.35 ± 0.127	99.01 ± 0.134	0.130 ± 0.051	8.32	3.29
SSAE+KNN	98.87 ± 0.146	98.69 ± 0.153	0.152 ± 0.062	9.19	4.83
SSAE+RF	99.21 ± 0.138	98.53 ± 0.210	0.148 ± 0.044	8.25	3.49

Model Comparison

Method	Normal	DOS	Probe	R2L	U2R	
Feature Selection Method	FMIFS[13]	98.98	98.76	86.08	88.38	22.11
	TDTC[27]	94.43	88.20	87.32	42.00	70.15
	SVM-ELM[28]	98.13	99.54	87.22	31.39	21.93
	DNEDRON [29]	98.92	95.94	97.17	83.75	76.92
Shallow Learning Model	TVCPSO[6]	99.13	98.84	89.29	75.08	59.62
	NBC-A[8]	98.21	99.11	97.67	98.30	99.19
	CANN[9]	95.98	96.59	82.85	78.95	61.54
	TLHA[11]	99.26	97.37	98.61	79.39	14.02
Deep Learning Model	HAST-IDS[15]	N/A	99.10	83.35	74.19	64.25
	LSTM[16]	99.50	99.30	87.00	30.40	75.10
	DBN ⁺ +LR[17]	94.51	98.74	86.66	100.00	38.46
	RNN-IDS[30]	N/A	83.49	83.40	24.69	11.50
Proposed method	SSAE+SVM	99.43	99.35	99.03	83.43	67.94

Model Comparison

- In the multi-classification experiments, for the **detection rate** of two types of low frequency attack samples, **R2L and U2R**, the methods we proposed have not achieved satisfactory results, only 84.43% and 67.94%, respectively.
- The main reason for this result is that the **number of R2L and U2R** attack samples contain in the **training set** used in our experiments are **scarce**
- Classification features are insufficient, which result in the SVM classifier failing to learn the sample features effectively
- Low frequency attack samples may be ignored as noise points or outliers of the majority class.



B. Yan and G. Han.

Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system.

IEEE Access, 6:41238–41248.



K. Alrawashdeh and C. Purdy.

Toward an online anomaly intrusion detection system based on deep learning.



In *2016 15th IEEE international conference on machine learning and applications (ICMLA)*, page 195–200. IEEE.



M. Tavallaee, E. Bagheri, W. Lu, and A.A. Ghorbani.

'a detailed analysis of the kdd cup 99 data set,'.

In *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl. (CISDA)*, page 1–6, Ottawa, ON, Canada.

-  H.H. Pajouh, R. Javidan, R. Khayami, D. Ali, and K.-K.R. Choo. 'a twolayer dimension reduction and two-tier classification model for anomalybased intrusion detection in iot backbone networks,'. *IEEE Trans. Emerg. Topics Comput.*
-  M.A. Ambusaidi, X. He, P. Nanda, and Z. Tan. 'building an intrusion detection system using a filter-based feature selection algorithm,'. *IEEE Trans. Comput*, 65(10):2986–2998,.

Thank You!

samyuktha@setsindia.net