Adversarial Network Model Based on Feature Fusion Learner for Intrusion Detection in Sensor Networks

P. Sherubha[1], Mohammed Iqbal[2,*], Aileen Chris[3], Arivazhagi[3], Nandhagopal Subramani[4]

[1]Department of Information Technology, Karpagam College of Engineering, Coimbatore, India
[2]School of Computer Science Engineering, VIT-AP University, Amaravathi, Andhra Pradesh, India
[3]Department of Bachelor of Computer Application, The American College, Madurai, India
[3]Department of Computer Science and Engineering, University college of Engineering, Ariyalur, India
[4]Department of Computer Science and Engineering, KRP Institute of Engineering and Technology, India
Email: sherubha0106@gmail.com, iqbalmecse@gmail.com, aileen.cse@gmail.com, arivupra@gmail.com, nan.ecs1@gmail.com

Abstract

The adversarial machine learning approaches are modelled to provide a defence mechanism during the prediction of cloning and jamming attacks launched over the wireless communication process. The transmitter is supplied with a pre-trained classifier to analyze the status of the channel based on the sensing nature and determine the other transmission process. The learning method gathers all acknowledgements and fusion made between nodes and the channel's current state to build a learning model that can accurately identify the succeeding transmission constraint caused by network jamming. In this instance, compared to random jamming procedures, an inventive anti-clone detection strategy aims to minimize the number of clones and jamming found throughout the network model. The transmitter analyzes the power restrictions over the sensor networks using the learning-based fisher score (FS). Here, an adversarial network model (ANM-FS) is fused to diminish the computational time to collect the training dataset by examining the incoming samples. With this defence mechanism, the transmitter intends to predict the false prediction rate (FPR) and design a better model for providing a reliable classifier. Systematically, the transmitter identifies the floating of attacks over the network model and adopts the defending mechanism to mislead the injected clone, enhancing the throughput and reducing the prediction error.

Keywords: Wireless Sensor Networks; clone; jamming, adversarial network model; fisher score; classifier

1. Introduction

WSNs are a good platform in various industries, including intelligent residential and industrial production. Many investigators are worried about the safety of WSNs owing to the faster growth and widespread use of multiple sensor networking devices. For WSNs, there are two forms of vulnerability scanning: device intrusion detection (DID) and network intrusion detection (NID). Both systems have advantages and disadvantages. NID examines network traffic data, detects suspected intrusions concealed within the data, issues alerts and detects in response to the intrusions [1]. It can proactively detect and classify potential real-time threats disguised in network information. In contrast to NID, DID data sources mostly rely on the instrument platform's records for audit judgment. This approach to intrusion prevention can precisely find and specify the processes of invasions. However, it uses many of the device's capabilities and relies on its dependability. achine learning has recently been one of the most used techniques for handling classification difficulties, mainly digital technology [2]–[5].

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

Many deep learning-based attack testing procedures have exhibited superior detectability. For example, an interruption recognition system based on neural networks is proposed with DNNs of more robust detection capability than legendary classification algorithm; Ashfaq et al. [7] present a Long Short-Term Memory (LSTM) connectivity to evaluate and understand key cyberattacks against sensor equipment, which is more efficient than regular strategies against shape-shifting attacks; To reduce model instability, Wang et al. [8] proposed an intrusion detection infrastructure based on the k-nearest-neighbor method that makes use of hybrid approaches. Chang et al. [9] suggested a strategy for representing network flow as 2D pictures by combining the closest neighbour search with a segmentation approach to create dynamic traffic imagery for malware detection.

Furthermore, several researchers utilized autoencoders to perform data transformation intrusion detection to improve performance successfully. For example, on a learning algorithm, Zhao et al. [10] introduced a deep-feature extractor and decision approach to accomplish high-level abstracted detailed data to increase detection capability. Vincent et al. [11] suggested a stacking coevolutionary neural detection approach and produced an excellent autoencoder model. The stacked autoencoder extracts the original data and then fed into the LSTM for training to create a successful detection mechanism. Li et al. [12] reduced the factors that have driven large-scale datasets using the LSTM autoencoder (LAE) programming process. The approach is resistant to both underneath and over-fitting of the modelling. In the first step, these approaches train the fully convolutional separately to improve the existing data aspect correlation and reduce the variable dimensionality.

Although standalone fully convolutional development can reach optimal detection rate, the learning algorithm to discover suitable convolutionary neural network parameters for the classifying system would take a long time. Extensive training reduces the model's resilience, whereas inadequate training makes it harder for the algorithm to obtain outstanding results. At the same time, the machine learning technique's computation complexity and model size make it difficult to implement in the real world, particularly in WSN devices, which typically have similar functionality owing to power requirements and many other factors [13], [14] and [15]. To accomplish the aim of lightness, intrusion detection and prevention algorithms often employ principal component analysis (PCA) and auto-encoder to minimize the feature dimensionality [10], [12]. Based on the literature, compact deep neural networks have just a few uses in NID approaches. This study presents an adversarial network model (ANM) paired with fisher score (FS) to overcome the existing issues. The research proposal makes four substantial improvements:

• A technique for clearly representing network traffic features is proposed. The improved characteristic of the source information is acquired by employing the adversarial network model (ANM) to capture the network activity character, decreasing the information dimensional and increasing attack detection accuracy.

• Using the Fisher score, redundant features are removed. The feature selection process allows it to extract characteristics and collect crucial information to identify threats while drastically decreasing computing costs and filling positions.

• We develop our technique using the KDD99 dataset and test its efficacy in attack detection. Experiments show that our approach has a noticeable lightning impact and a necessary detector.

The arrangement of the study is as follows: Section 2 offers a comprehensive analysis of the benefits and drawbacks of several contemporary methods. The research flow is drafted in section 3, with the study of numerical outcomes in section 4. The work summary is outlined in section 5 with the idea for future advancements.

2. Related works

There is a range of categories for attacks on WSNs in the literature. We'll use the attacker's behaviour (neutral) as the main section and the desired Open Systems Interconnection (OSI) paradigm (layered definition of stack protocol) as sub-categories, with each item explained below: Passive attacks are carried out in such a way that they are undetectable by any means [16]. Because the enemies do not emit any electromagnetic radiation, this is the case. Because wireless communications are more straightforward to tap, unlicensed spectrum is more vulnerable to passive attacks like wiretapping, which may be readily carried out by listening to wireless transmission

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

across sensor nodes in a WSN without acquiring any of them [17]. Passive threats are mainly used to compromise privacy protection. Attackers are often disguised, i.e. concealed, and tap transmission cables to acquire data in passive attacks [18]. Surveillance, node malfunctions, nodes trying to interfere, base station outages, and network monitoring are passive attacks [19].

It is vital to note that network obliteration is deemed aggressive attacks in specific works. Therefore, we classify them as passive attacks since, compared to other replay attacks, they pose little risk to the network [20]. The rationale is that they do not create a design flaw since the internet may continue operating alone without failing nodes [21]. Malicious operations are carried out not only for data protection but also against sensitive data inactive attacks. Active attacks are used to gain illegal access to and use the equipment and disrupt an opponent's communications. An active adversary emits a radio signal or performs an activity that the WSN components can detect [22]. For example, DoS attacks at the physical and network layers can cause nodes to discard datagrams. A DoS attack primarily targets the accessibility of a system [23]. DoS is commonly defined as any circumstance that utilizes productivity and minimizes a network's capacity, preventing the connection from completing its process on time. An adversary tries to prohibit authorized users from accessing certain services in a DoS attack [24]. The traditional method of accomplishing this is to flood the packet to any centralized resource (e.g., an access point) within a connection, causing the capabilities to become unavailable to the program's other components, resulting in the channel failing to function as envisioned. It might result from a failure to supply assured available to end consumers. Inactive attacks, the attacker interferes with the network's functions. This impact might be the explosion's goal, and it's easy to spot [25]. As a result of these attacks, connectivity services may be impaired or discontinued. The attacker may attempt to remain unnoticed to obtain unauthorized administrator privileges or to jeopardize the platform's content's authenticity and integrity. Table 1 depicts the comparison of various attack types.

Table 1: Comparison of attack types

Attacks	Samples	Description
Routing	Black Hole, Worm Hole, Grey Hole, Sybil, Byzantine, Man in the Middle,	These exploits on the network level spoof, change or replicate periodic routing.
Sniffer	Phishing, Online Scams, Spam Detection, Account Hijack, Defacement, Illegal transaction	These exploits are carried out using network sniffing tools to capture or monitor data traffic.
Passive	eavesdropping, Traffic testing	These are encryption system attacks wherein the perpetrator does not interact with the system components and instead tries to breach the security by watching the data flow between components.
Insider	Flooding attack, User to starting place, Port Scan	These are destructive attacks on the communication system by a member permitted to access the networking. He could also know anything about the system's infrastructure.
Malicious	Botnet, Malware	It is defined as exploiting a user's system by implementing malicious programs, such as infected computers or manipulating media to access the compromised machine.

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

DoS/DDoS	Buferoverfow, UDP Flood, SYN Flood, Ping of Death, ICMP, Smurf,	These cyberattacks overload the internet by flooding it with packets, restricting legitimate customers from using the network's capacity.
Cyber	Cyberbullying, Cyber nuisance	These are targeted attacks against specific organizations and persons to acquire access to intellectual, confidential, or organizational advantages to cause harm or profit.
Vulnerabilities	Misconfiguration: the system utilizes	These flaws in the coding or development work make it possible for the platform to be hacked. These are the flaws that threats can exploit to perform their software or get access to firmware.
Others	Password harass, Brute force, Dictionary harass	These attacks are carried out to obtain the users' usernames and other personal and various components.

3. Methodology

In this case, a thorough analysis offers protection and aims to eliminate pointless feature patterns significantly affecting the network model. An adversarial network model based on attention mechanisms is provided to forecast the trajectory of attacks across the networks, and a personalized privacy preservation model is supplied to ensure network privacy. Python is used as the evaluation environment. The samples are trained and tested using the NSL-KDD dataset to ensure their validation. Fig 1 depicts the privacy preservation model framework.

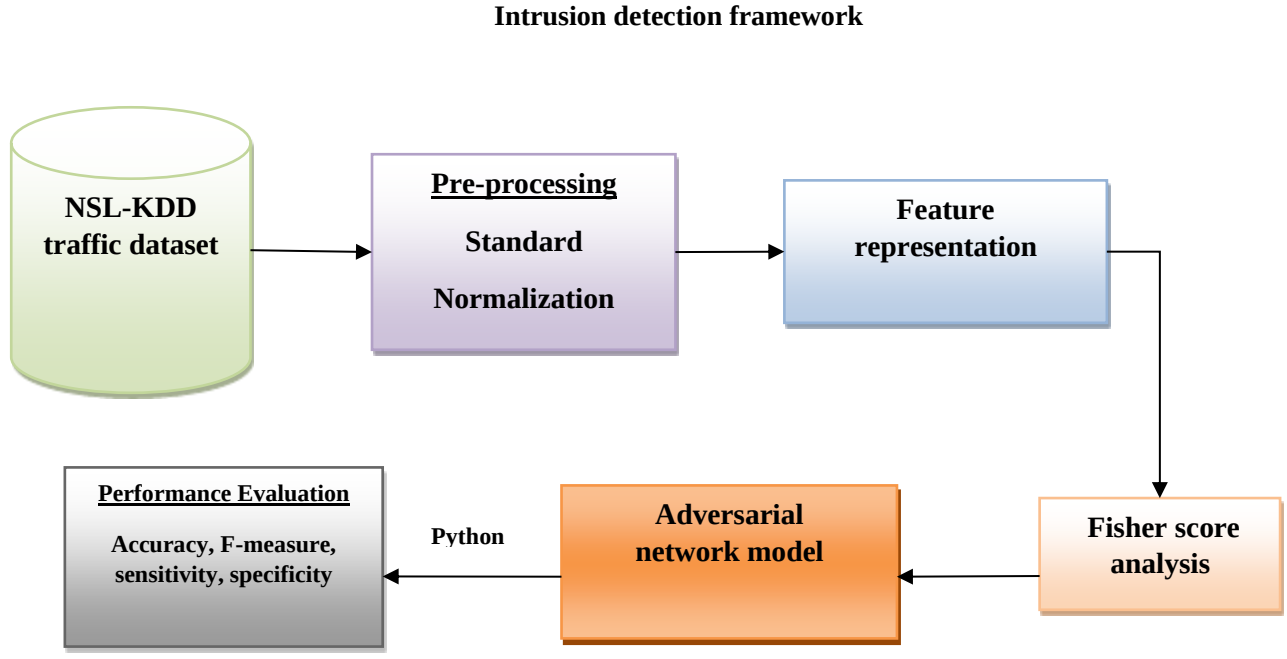

Figure 1: Block of ANM_FS model

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

a. NSL-KDD

In 2009, Travallaee et al. [3] suggested using the NSL-KDD dataset due to its inherent drawbacks compared to the KDD-CUP'99 dataset. The categories of the attacks are shown in Table 2.

Table 2: Attack types

Type	Categories
U2R	Xterm, Ps, Sqlattack, Perl, rootkit, buffer_overflow, load module
R2L	send_fmail, Named, snmp_getattack, httptunnel, xsnoop, snmp_guess, spy, xlock, waremaster, imap, ware-relient, ftp_write, multi-hop, phf, guess_password
Probe	mscan, saint, portsweeop, satan, ipsweep
DoS	Worm, process_table, udpsworm, tear_drop, smurf, land_neptune, apache_2, back

Table 3: Dataset features

Features	S. No	Names	Data_type	Features	S. No	Names	Data_type
Basic	1	Duration	C	Content	21	is_host_login	S
	2	Protocol_tye	S		22	Is_guest_login	S
	3	Service	S	Traffic	23	Count	C
	4	Flag	S		24	server_count	C
	5	Source_bytes	C		25	serror_rate	C
	6	Destination_bytes	C		26	server_serror_rate	C
	7	Land	C		27	rerror_rate	C
	8	Wrong_fragment	C		28	server_rerror_rate	C
	9	Urgent	C		29	same_reeror_rate	C
	10	Hot	C		30	different_server_rate	C
Content	11	Number_failed_logins	C		31	server_different_host_rate	C
	12	Logged_in	S		32	destination_host_count	C
	13	Number_compromised	C		33	destination_host_server_count	C
	14	Root_shell	C		34	destination_host_same_ser	C

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

						ver_rate	
	15	Su_attempted	C		35	destination_host_different_server_rate	C
	16	Number_root	C		36	destination_host_same_source_port_rate	C
	17	Number_file_creations	C		37	destination_host_server_different_host_rate	C
	18	Number_shells	C		38	destination_host_serror_rate	C
	19	Number_access_files	C		39	destination_host_server_serror_rate	C
	20	Number_outbounded_commands	C		40	destination_host_rerror_rate	C
					41	destination_host_server_rerror_rate	

C- continuous, S- symbolic

Various training/testing set records are contained in the dataset mentioned above. There are 1 27,973 records overall for training and 22,544 for testing. There are 41 aspects in the traffic record: 35 continuous and six symbolic. Table 3 explains the categorized dataset, encompassing denial-of-service attacks, probing, R2L, and U2R. The feature categories—primary, content, and traffic—are displayed in Table 2. The differences between the training and testing datasets demonstrate a realistic foundation for intrusion detection.

b. Pre-processing

Another name for this pre-processing stage is scrubbing or data cleansing. It speaks about predicting and correcting errors across the dataset that illustrates the detrimental effects of the prediction model. It encompasses all the jobs and endeavours involved in anticipating and fixing data problems. To handle inconsistencies, the work consists of smoothing, eliminating noise and outliers, and filling in missing numbers.

c. Filter model for data privacy attainment

The redundant filter diminishes the bottleneck while handling massive incoming data over the network. The data speed is reduced while the successive request by the incoming data is triggered from certain constraints. Implementing the redundant filter module offers an opportunity for parallel data execution during system failure by provisioning the load to be scattered among diverse resources. A redundancy filter provides a pipeline-based structure to run on various machines, enabling them to scale independently and be scalable with the available sensor environment. The redundant filter processing is computationally intensive and functions on the host, while the other filters are less demanding and have less performance. The prevailing filter module does not possess

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

any specific location over the data centre by facilitating the queue's task to run in the provided environment nearer to the resources it needs. When the input to the filter is provided in the stream (queue), it is probable to carry out the filter's processing in a parallel manner without any redundancy over task allocation. The filter module initiates the process in a pipeline devoid of redundancy. The task is provided to the scheduler for suitable scheduling with the delayed response time. Filtering is done before the model filters the redundant incoming data before completion. The significant advantage of this module is its resiliency.

c. Fisher score analysis

Fisher score predicts the feature subset where the chosen features span the dataset space. The distance among the data points in diverse classes is larger, and the distance among the data points in the class is smaller. Specifically, with the chosen m features, the input data matrix $X \in \mathbb{R}^{d*n}$ diminishes to $Z \in \mathbb{R}^{m*n}$. It is evaluated as in Eq. (1):

$$F(Z) = \mathrm{tr}\{(\bar{S}_b)\left(\bar{\bar{S}}_t + \gamma I\right)^{-1}\} \tag{1}$$

Where γ specifies the positive regularization parameter, $\bar{S}_b$specifies scatter matrix, $\overline{\overline{\text{and } S}}_t$ specifies the total scatter matrix, and it is depicted as in Eq. (2):

$$\bar{S}_b = \sum_{k=1}^{c} n_k\ (\bar{\mu}_k - \bar{\mu})(\ \bar{\mu}_k - \bar{\mu})^T \tag{2}$$

$$\bar{S}_t = \sum_{i=1}^{n} (z_i - \bar{\mu})\,(z_i - \bar{\mu})^T$$

Here, $\bar{\mu}_k$ and n_k specify the mean vector and class size over the diminished data space, i.e. $Z, \bar{\mu}_k = \sum_{k=1}^{c} n_k \bar{\mu}_k$ determines the complete mean vector over the diminished data. When $\tilde{S}_t$ defines singular and includes the perturbation time to provide the semi-definite values. There are $\binom{\ }{\ }$DM candidates Z over X. Feature selection is seen as a combinatorial optimization issue, which is highly challenging. Some widely adopted heuristic strategies are applied to measure the score for every feature independently to handle the complexity. Subsequently, it considers $x^j \in \mathbb{R}^{1*n}$. In some cases, there are $\binom{d}{1} = d$ candidates. Let μ_k^j and σ_k^j specify the mean and SD of the k^{th} class related to the j^{th} feature. Let μ^j and σ^j select the mean and SD of the data associated with the j^{th} feature. According to Eq. (3), the fisher score of the jth feature is assessed.

$$F(x^j) = \frac{\sum_{k=1}^{c} n_k \left(\mu_k^j - \mu^j\right)^2}{(\sigma^j)^2} \tag{3}$$

Here, $(\sigma^j)^2 = \sum_{k=1}^{c} n_k \left(\sigma_k^j\right)^2$. It selects the highest-scoring feature with the highest score after calculating the fisher score for each feature. The sub-optimal algorithm determines the features as the feature score is independently assessed. Some algorithms fail to select features with relatively lower individual scores; however, a high score is combined. Additionally, it cannot deal with redundant features. However, some generalized fisher scores are used to handle these problems. The problem is more challenging to maximize owing to the combinatorial. The optimal objective function value lower bounds the optimal value, and it is expressed as in Eq. (4):

$$F(W, p) = \mathrm{tr}\{(W^T \mathrm{diag}(p) s_b \mathrm{diag}(p) W)(W^T \mathrm{diag}(p)(S_t + \gamma I)\mathrm{diag}(p)(W)^{-1}\} \tag{4}$$

$$\text{s.t. } p \in \{0,1\}^d, p^T 1 = m$$

The key is to prove the model's feasibility; the objective function of others lowers the objective function.

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

d. Adversarial network model

The generative network model helps handle more challenges, like an approximation of probabilistic computations, which triggers the maximal likelihood evaluation and related approaches. It also deals with the complexity of averaging the advantages of various methods for discriminative models. The model attempts to predict a hierarchical model that specifies the probability distribution among several data types (See Fig 2). It comprises a particular NN architecture with a large multilayer perceptron: generator G, which is related to the differentiable function; inputs are associated with the prior noise variables p_z (z) and evaluated parameters θ_g. Discriminator D is also associated with the differentiable function; in this case, the input is related to the data (fake or real), and the output is a binary scalar that labels the generated data (real or fake). The networks have received training: When input from data is obtained from G and trained to create samples with p_g distribution similar to data distribution $p_{(data)}$, D is prepared by probability maximization and label input. During the training process, D and G act as a mini-max among every other specified by Eq. (4):

$$\min_{G}\max_{D} V(D,G) = \{E_{y-p_{data}} \log D(y) + E_{z-p_z} \log(1-D(G(z)))\} \quad (4)$$

Here, $V(D,G)$ specifies the functional value, y specifies the generated data, $E_{y-p_{data}} \log D(y)$ specifies the D's log probability for evaluating genuine data, $E_{z-p_z}(1-D(G(z)))$ related to logging prediction probability of the generated data is not so simple. Z specifies the input vector of G. GAN training is performed with continuous gradient descent optimization. Initially, training data is partitioned into mini-batches; a mini-batch containing m samples is inputted to $G(z)$. It needs to return m simulated by G. The D is trained with mini-batches from real and fake data, pretending to find whether the data is fake or real. It is recognized that mini-batch is attained from original data where the discriminative network parameter is optimized with gradient-based optimization. Successive mini-batch is composed of m models. It is sampled where the parameters θ_g are optimized based on D feedback when the generated samples are competent or misled D. In general theory. $P_g=p_{data}$ is anticipated to be reached via the mini-max optimization. In addition, real-time achievement of this goal is difficult and leads to optimization problems such as training stability and vanishing gradients.

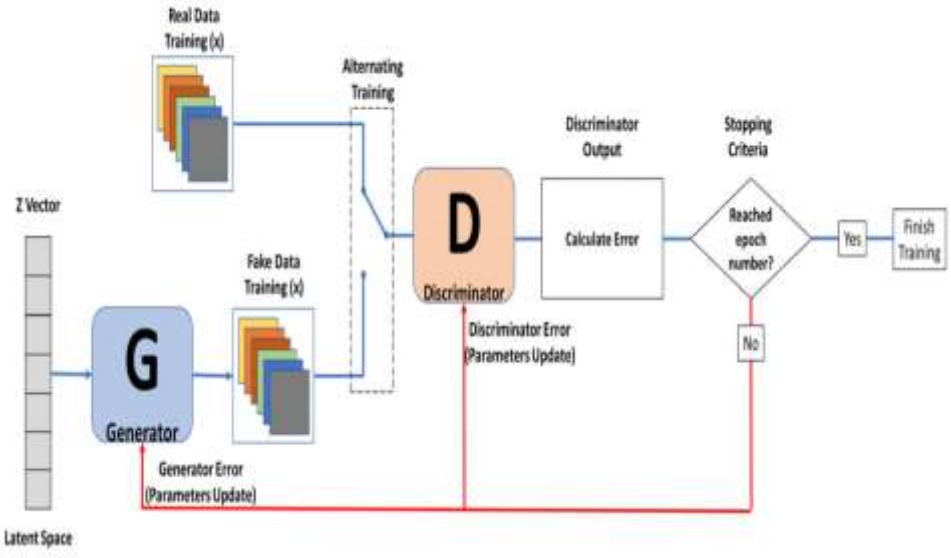

Figure 2 : Generic view of adversarial network model

4. Numerical results and discussion

The NSL-KDD dataset is used in this work to validate ANM-FS's functionality for improving network intrusion detection. An i5 CPU with 8GB RAM and 2.71 GHz of processing power is employed for the experiments. Python is used to run the simulation, and the training and testing

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

models' properties are used to evaluate the performance metric. ANM-FS performance is assessed using a variety of performance measures. Here's how the performance metrics are calculated: False Negative (FN), False Positive (FP), True Positive (TP), and True Negative (TN).

1) Eq. (5) shows that accuracy is the percentage of all NSL-KDD records over the testing set.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{5}$$

2) Eq. (6) expresses precision as the ratio of detected intrusion to total intrusion during the testing phase.

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

3) Recall: It is represented by Eq. (7) and is expressed as the ratio of predicted intrusion to all intrusion samples in the testing set.

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

4) F-measure: As seen in Eq. (8), it is represented as the recall to precision ratio.

$$\text{F} - \text{measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{8}$$

The experiment examines the effectiveness and efficiency of the lower-level characteristics provided to the classifier model with binary class ('0'-normal, '1'-anomaly) and multi-classes (probe U2R, R2L, and regular) concerning the provided dataset. Moreover, the model efficiency is calculated by testing and training. It aims to satisfy the storage needs, lessen computational complexity, and handle intrusion detection with the cryptosystem. Finding the best data representation and reducing the dimensionality of features at a lower level is the model's primary goal. Since the kernel model linearly uses memory and processing space proportionate to the number of support vectors, fewer support vectors are needed. Comparisons are made between the ANM-FS model performance and more general techniques such as SVM, Random Forest (RF), J48, Naive Bayesian (NB), and Decision Tree. With low-dimensionality features, independent testing and training assess the model's significance. Accuracy and training speed significantly impact the hidden parameters, according to the theoretical model analysis. Modelling an effective deep learning model for intrusion detection involves more hyperparameter tweaking. While sparse parameter investigations are more complex, they take less time to test and train. The search for the ideal parameters improves ANM-FS performance. Auto-encoders are used to fine-tune hyper-parameters on the training NSL-KDD dataset. Compared to the binary classification procedure, the ANM-FS performs better in selecting, training, and testing hyperparameter values.

Table 4: Accuracy measurement

Iterations	5	10	15	20	30
ANM-FS	88.15	90.89	95.89	87.57	99.81
$a - DNN$	85.90	87.98	92.80	95.75	97.88

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

SELF	81.88	86.97	90.78	92.57	95.78
Game theory	54	59	62	68	70
Single SVM	80	76	84	81	76
STL-IDS	65	69	71	76	80
Random forest	68	72	76	78	80
RNN	70	76	79	81	83

Table 5: Precision measurement

Iterations	5	10	15	20	30
ANM-FS	81.56	92.45	96.89	98.55	99.8
a – DNN	78.56	89.77	94.55	96.77	99
SELF	75.24	86.97	92.56	95	100
Single SVM	71	78	86	90	92
STL-IDS	76	80	87	92	93

Table 6: Recall measurement

Iterations	5	10	15	20	30
ANM-FS	74.56	84.56	88.89	84.51	94.56
a – DNN	69.88	80.55	84.58	90.15	90.58
SELF	65.98	76.11	81.52	86.47	89
Single SVM	50	55	58	60	61
STL-IDS	51	56	60	63	68

Table 7: F-measure measurement

Iterations	5	10	15	20	30
ANM-FS	79.87	84.78	90.56	96.54	99.58
a – DNN	75.88	80.56	86.88	92.55	96.89
SELF	72.58	79.56	82.98	89.92	94.17
Single SVM	55	60	62	69	74
STL-IDS	59	63	67	72	79

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

Table 8: Time measurement

Iterations	5	10	15	20	30
ANM-FS	256	259	264	546	758
a – DNN	278	350	500	1100	1300.5
SELF	305.22	400.98	529.47	1125.14	1325.42
Single SVM	709	1362	1867	2601	3503
STL-IDS	413	465	673	1611	1362

Table 9: Detection rate comparison

Iterations	5	10	15	20	30
ANM-FS	0.004	0.0045	0.0030	0.0030	0.0035
a – DNN	0.005	0.0055	0.0045	0.0045	0.0040
SILF	0.007	0.0062	0.0059	0.0048	0.0042

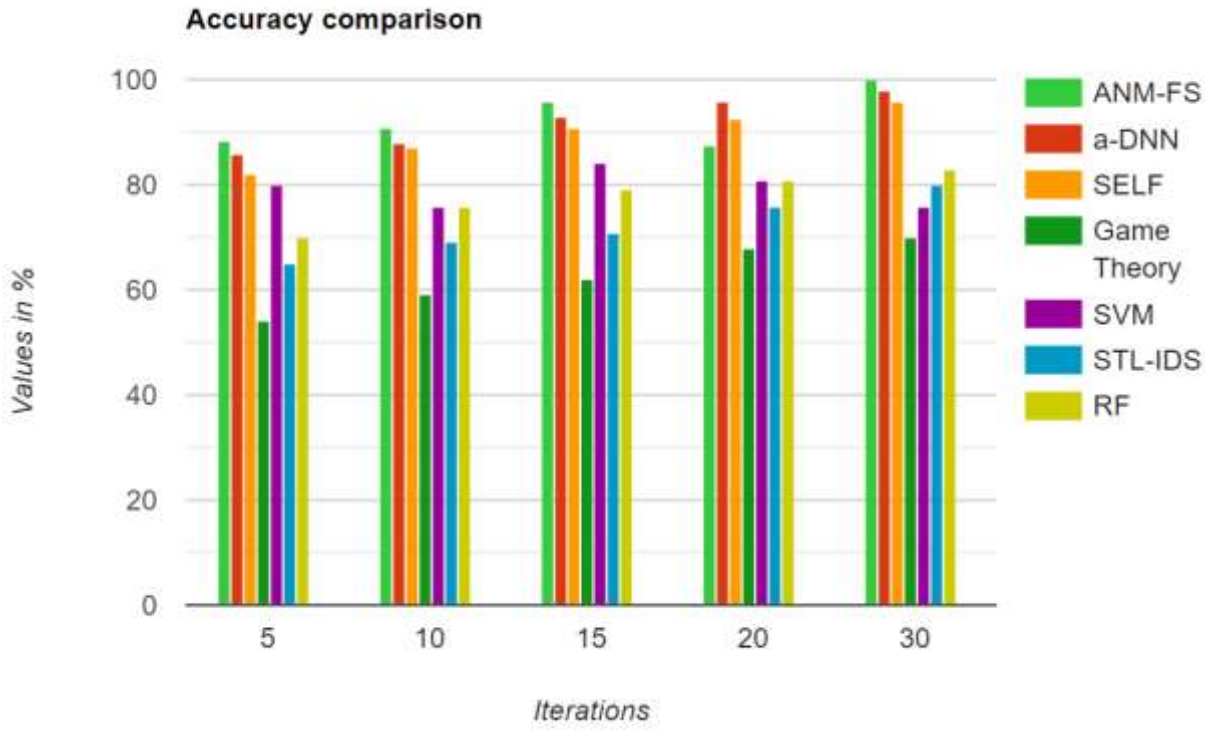

Figure 3: Accuracy comparison

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

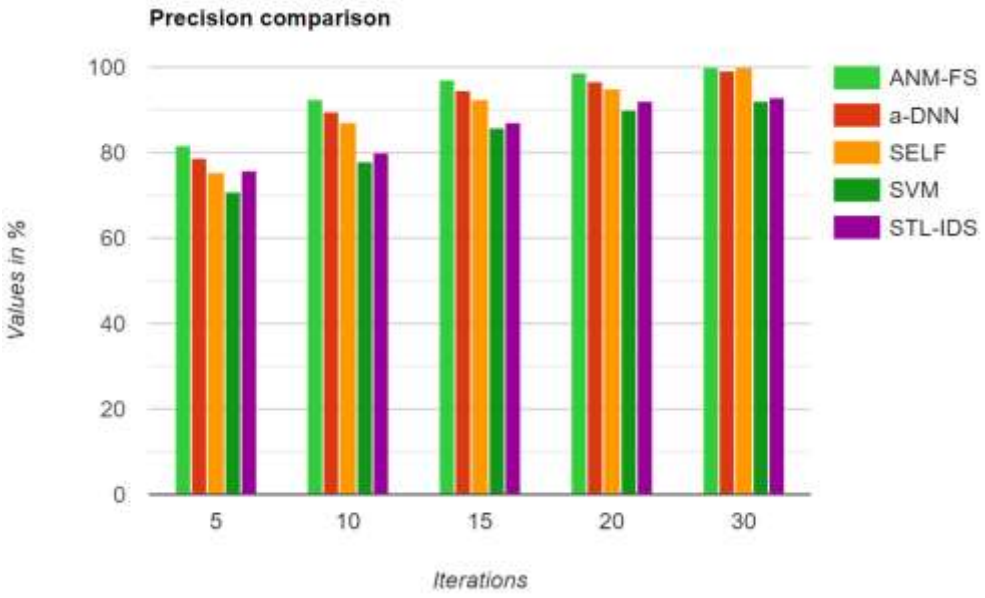

Figure 4: Precision comparison

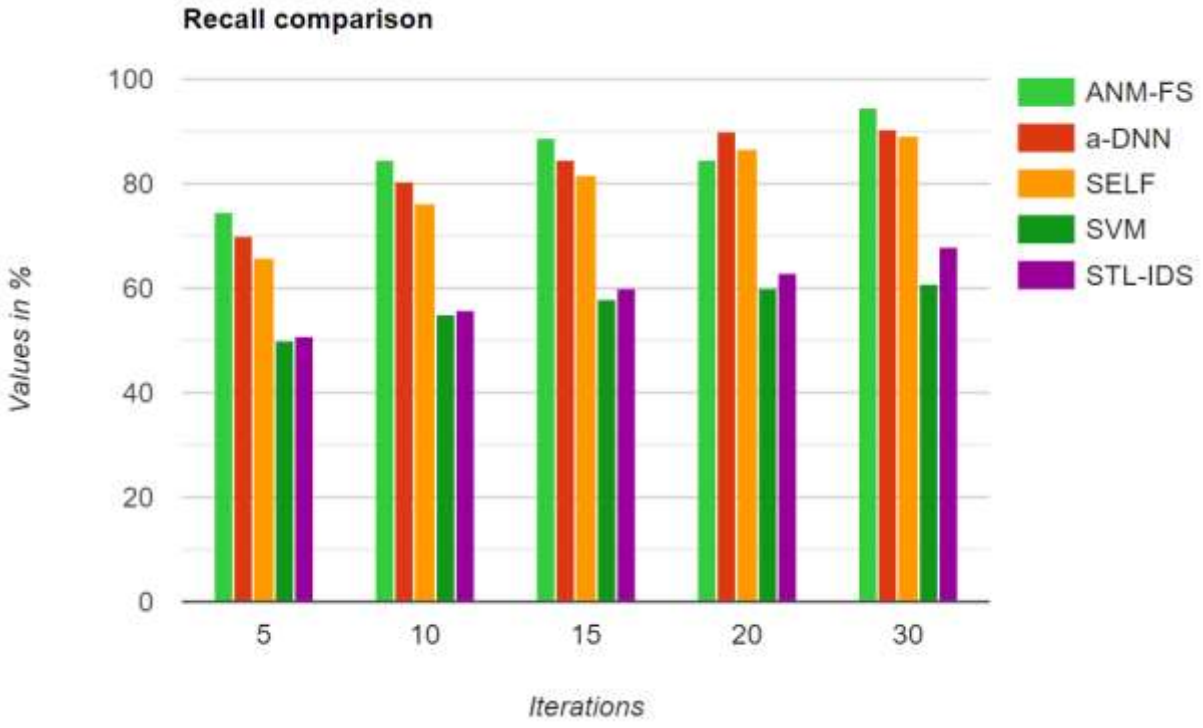

Figure 5: Recall comparison

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

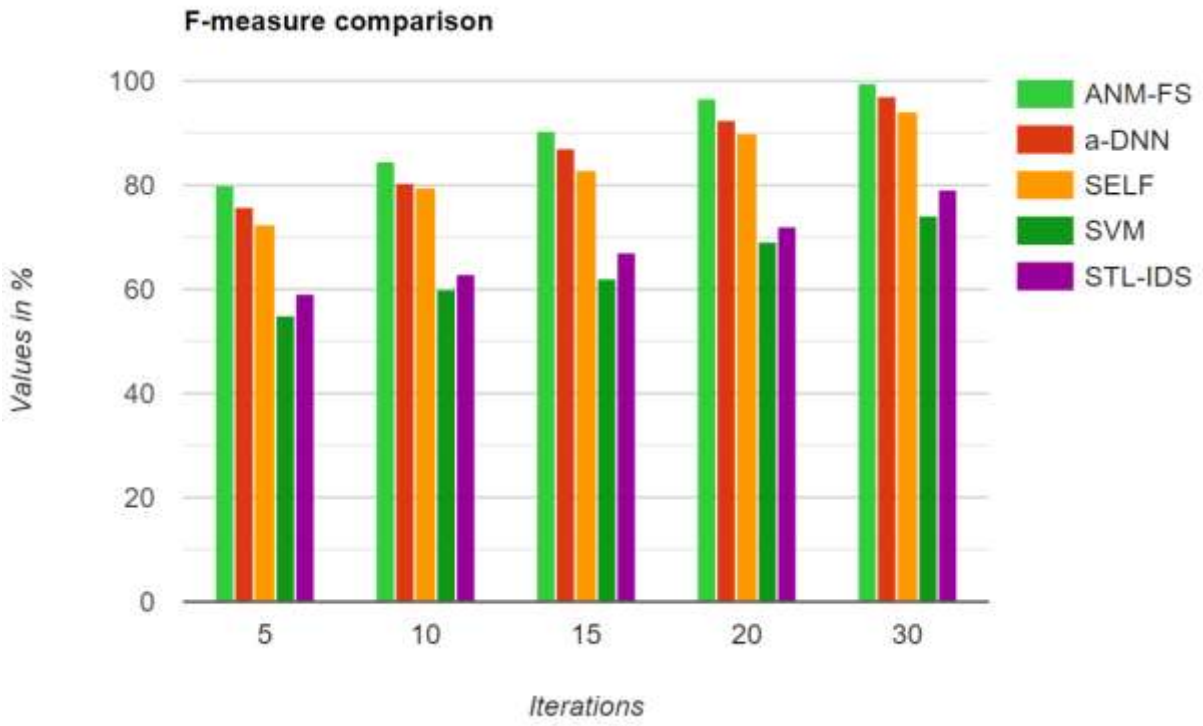

Figure 6: F1-score comparison

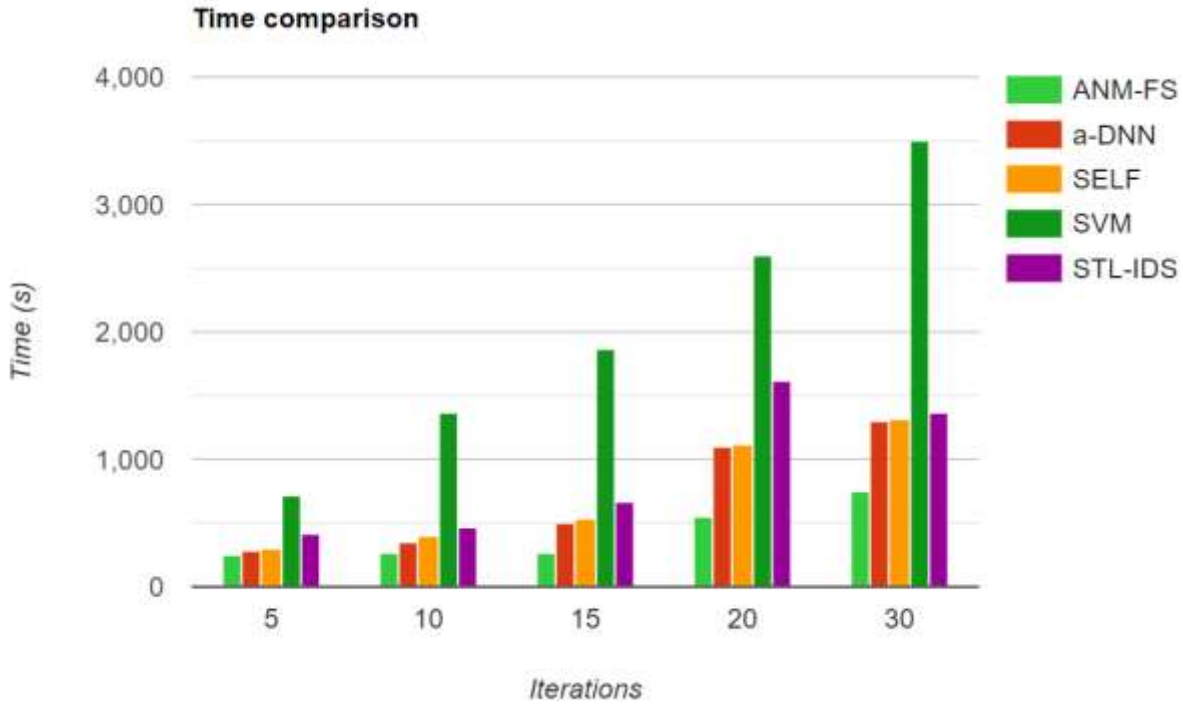

Figure 7: Time comparison

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

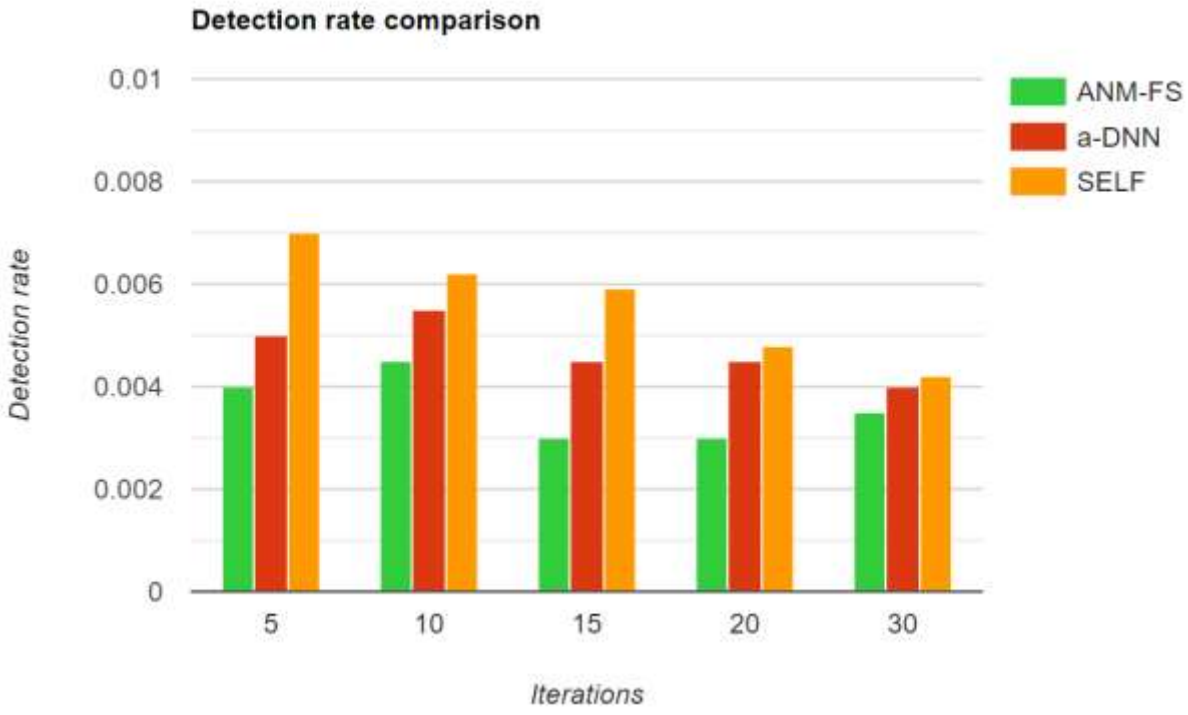

Figure 8: Detection rate comparisons

Table 4 depicts the evaluation of the anticipated ANM-FS with various other approaches like a-DNN, SILF, game theory, single SVM, STL-IDS, RF and RNN (See Fig 3 to Fig 8). The iterations considered for this analysis are 5, 10, 15, 20 and 30. The accuracy of the anticipated model for the 30th iteration is 99.81%, which is 1.93%, 4.03%, 29.81%, 23.81%, 19.81%, 19.81% and 16.81% superior to other models. Table 5 compares precision with the anticipated a-DNN with different approaches. With an accuracy of 99.8%, ANM-FS outperforms a-DNN, SVM, and STL-IDS by 0.8%, 7.8%, and 6.8%, respectively, and SILF by 0.20%. The recall comparison between the anticipated ANM-FS and alternative methods is shown in Table 6. The model attains 94.56% recall, 3.98%, 5.56%, 33.56%, and 26.56% higher than different approaches. Table 7 compares the anticipated model in terms of 99.58% F-measure, which is 2.69%, 5.41%, 25.58% and 20.58% higher than other approaches. The time consumed for execution is 758 seconds for the anticipated a-DNN model, which is less than a-DNN, SILF, SVM, and STL-IDS, as in Table 8. The detection rate is 0.0035, which is higher than the existing approaches. All these analyses prove that the model works well in terms of attack prediction with better privacy preservation policies.

5. Conclusion

The anticipated ANM-FS model possesses strong modelling competency for intrusion detection and provides superior accuracy for multi-class and binary classification. Contrary to conventional classification approaches like SILF, game theory, SVM, STL-IDS, RF and RNN, the performance attains superior detection and accuracy rate with lesser false positive rate precisely for multi-class classification tasks with benchmark datasets. The model effectively enhances intrusion detection accuracy and competency in predicting various intrusion types. However, in future research, this research will be extended by focusing on diminishing the training time consumed with the modern deep learning approaches, vanishing the gradients, avoiding exploding and learning the classification performance of the current network model and another algorithm in the intrusion detection field.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, ‘‘Deep learning,’’ Nature, vol. 521, no. 7553, pp. 436–444, May 2015.

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

[2] Reddy, Y. Ramadevi, and K. V. N. Sunitha, ‘‘Effective discriminant function for intrusion detection using SVM,’’ in Proc. Int. Conf. Adv. Comput., Commun. Inform. (ICAC), Sep. 2016, pp. 1148–1153.
[3] Ingre and A. Yadav, ‘‘Performance analysis of NSL-KDD dataset using ANN,’’ in Proc. Int. Conf. Signal Process. Commun. Eng. Syst., Jan. 2015, pp. 92–96.
[4] Farnaaz and M. A. Jabbar, ‘‘Random forest modelling for network intrusion detection system,’’ Procedia Comput. Sci., vol. 89, pp. 213–217, Jan. 2016.
[5] Khan and N. Jain, ‘‘A survey on intrusion detection systems and classification techniques,’’ Int. J. Sci. Res. Sci., Eng. Technol., vol. 2, no. 5, pp. 202–208, 2016
[6] Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software-defined networking," Proc. Int. Conf. Wireless Netw. Mobile Commun. (WINCOM), Oct. 2016, pp. 258–263.
[7] Ashfaq, X.-Z. Wang, J. Z. Huang, H. Abbas, and Y.-L. He, "Fuzziness based semi-supervised learning approach for an intrusion detection system," Inf. Sci., vol. 378, pp. 484–497, Feb. 2017.
[8] Ashfaq, X.-Z. Wang, J. Z. Huang, H. Abbas, and Y.-L. He, "Fuzziness based semi-supervised learning approach for an intrusion detection system," Inf. Sci., vol. 378, pp. 484–497, Feb. 20
[9] Chang, W. Li, and Z. Yang, "Network intrusion detection based on random forest and support vector machine," Proc. IEEE Int. Conf. Comput. Sci. Eng./IEEE Int. Conf. Embedded Ubiquitous Comput., Jul. 2017, pp. 635–638
[10] Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, “Deep learning and its applications to machine health monitoring: A survey,” Submitted to IEEE Trans. Neural Netw. Learn. Syst., 2016. [Online]. Available: http://arxiv.org/abs/1612.07640
[11] Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” J. Mach. Learn. Res., vol. 11, pp. 3371–3408, 2010.
[12] you, Y. Li, Y. Wang, J. Zhang, and Y. Yang, "A deep learning-based RNNs model for automatic security audit of short messages," in Proc. 16th Int. Symp. Commun. Inf. Technol., Qingdao, China, Sep. 2016, pp. 225–229.
[13] Alrawashdeh and C. Purdy, “Toward an online anomaly intrusion detection system based on deep learning,” in Proc. 15th IEEE Int. Conf. Mach. Learn. Appl., Anaheim, CA, USA, Dec. 2016, pp. 195–200
[14] Potluri and C. Diedrich, "Accelerated deep neural networks for an enhanced intrusion detection system," Proc. IEEE 21st Int. Conf. Emerg. Technol. Factory Autom., Berlin, Germany, Sep. 2016, pp. 1–8.
[15] Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software-defined networking," Proc. Int. Conf. Wireless Netw. Mobile Commun., Oct. 2016, pp. 258–263
[16] Hodo, X. J. A. Bellekens, A. Hamilton, C. Tachtatzis, and R. C. Atkinson, Shallow and deep networks intrusion detection system: A taxonomy and survey, Submitted to ACM Survey, 2017, [Online]. Available: http://arxiv.org/abs/1701.02145
[17] Kim, G., Yi, H., Lee, J., Paek, Y., Yoon, S.: Lstm-based system-call language modelling and robust ensemble method for designing host-based intrusion detection systems. arXiv preprint arXiv:1611.01726 (2016)
[18] Buczak, A.L., Guven, E.: A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications Surveys & Tutorials 18(2) (2016) 1153–1176
[19] Javaid, A., Niyaz, Q., Sun, W., Alam, M.: A deep learning approach for network intrusion detection system. In: Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), New York, NY, USA. Volume 35. (2015) 2126

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024

[20] Safa Otoum, Burak Kantarci, and Hussein T. Mouftah, "Adaptively supervised and intrusion-aware data aggregation for wireless sensor clusters in critical infrastructures," in 2018 IEEE International Conference on Communications (ICC), May 2018, pp. 1–6
[21] Arnaldo Gouveia and Miguel Correia, A Systematic Approach for the Application of Restricted Boltzmann Machines in Network Intrusion Detection, vol. 10305, 05 2017.
[22] Beigh and M. A. Peer, "Performance evaluation of different intrusion detection system: An empirical approach," in Intl Conf. on Computer Communication and Informatics, Jan 2014, pp. 1–7.
[23] Zhou W., Wen J., Koh Y., Xiong Q., Gao M., Dobbie G., Alam S. (2015), "Shilling Attacks Detection in Recommender Systems Based on Target Item Analysis" PLoS One, July, 10(7), p.e0130968
[24] Kumari T., Bedi P. (2017), "A Comprehensive Study of Shilling Attacks in Recommender Systems", IJCSI International Journal of Computer Science Issues, 14(4), 44-50
[25] Cao, J., Wu, Z., Mao, B. & Zhang, Y (2013). "Shilling attack detection utilizing semi-supervised learning method for attack detection utilizing semi-supervised learning method for collaborative recommender system". World Wide Web Jornal, 16(5-6): 729-748.
[26] Yu H, Gao R, Wang K, Zhang F (2017), "A novel robust recommendation method based on kernel matrix factorization". J Intell Fuzzy Syst 32(3):2101–2109
[27] Yang Z., Cai Z. (2017), "Detecting abnormal profiles in collaborative filtering recommender systems". Journal of Intelligent Information Systems, 48(3), 499-518
[28] Zhou W., Wen J., Qu Q., Zeng J., Cheng T. (2018), "Shilling attack detection for recommender systems based on credibility of group users and rating time series", PLoS One, May, 13(5), p.e0196533
[29] Turk A., Bilge A., (2019). "Robustness analysis of multi-criteria collaborative filtering algorithms against shilling attacks", Expert Systems with Applications, 115, p.386-402
[30] Moradi P., Ahmadian S., (2015), "A reliability-based recommendation method to improve trust-aware recommender systems", Expert Systems with Applications, 42, 7386-7389.
[31] Paradarami, N.D. Bastian, J.L. Wightman, "A Hybrid recommender system using artificial neural networks", Expert Systems with Applications, Vol. 83, (2017), 300-313.
[32] Agar ap, "A neural network architecture combines gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data," Proc. 10th Int. Conf. Mach. Learn. Comput., Feb. 2018, pp. 26–30.
[33] Around, M.-A. El Hussaini, A. El Hore, and J. Ben-Othman, "Real-time detection of MAC layer misbehaviour in mobile ad hoc networks,'' Appl. Comput. Information., vol. 13, no. 1, pp. 1–9, 2017.

Doi: https://doi.org/10.54216/FPA.150114
Received: August 18, 2023 Revised: December 22, 2023 Accepted: March 09, 2024