# Link-Based Xcorr Normalization and Attention Mechanism for Predicting the Threats over the Network Model

**V. Jemmy Joyce[1], K. Rebecca Jebaseeli Edna[2], P. Sherubha[3,*], Arivazhagi[4]**

[1]Department of Mathematics, Karunya Institute of Technology and Sciences, Coimbatore, India
[2]Department of Mathematics, Karunya Institute of Technology and Sciences, Coimbatore, India
[3]Department of Information Technology, Karpagam College of Engineering, Coimbatore, India
[4]Department of Computer Science and Engineering, University college of Engineering, Ariyalu, India

Emails: jemmy@karunya.edu; edna@karunya.edu; sherubha.p@kce.ac.in; arivupra@gmail.com

**Abstract**

Sensor Networks (SNs) play an essential role in upcoming technologies like the Internet of Things (IoT), where technical services are highly prone to crucial vulnerability due to attacks. This research motivates to provide a mechanism to identify the link reliability of connected sensor nodes. The privacy-preserving keys are distributed among the corresponding network nodes. When the nodes suffer from an attack, it damages the linking nodes' community. It has the nature of healing itself when the attacks are identified over the network. The self-healing nature is not so complex, and it is termed a lightweight process. A novel link-based intrusion prediction mechanism uses attention-based Deep Neural Networks ($a$-DNN) for lightweight linkage identification and labelling. This model helps predict basic network patterns using topological analysis with better generalization. The simulation is done with Python where the proposed $k$-DNN model outperforms the five different conventional approaches with the adoption of a benchmark dataset (network traffic) for extensive analysis. The AUC is improved in an average manner with the adoption of $k$-DNN. This model enhances the linkage connectivity to make different connectivity processes more efficient and reach the target non-convincing. It is sensed that the proposed $a$-DNN outperforms the existing approaches by improving the network resilience by maintaining higher energy efficiency.

**Keywords:** Sensor Network; link-based prediction; topological analysis; linkage identification; labelling; generalization; Kernel-based Deep Neural Networks

## 1. Introduction

Wireless Sensor Network (WSN) is a wireless system of many reduced micro-edge devices installed in a surveillance region. The monitoring nodes are linked to unstable and untrustworthy networks because they have restricted memory resources, processing capability, and energy. The WSN with secrecy demands has been more popular in industrial and commercial settings in recent years [1,2]. The primary duties of an intrusion detection system (IDS) are to catch attackers attempting to destroy or monitor the protection of a wireless sensor network (WSN) and discover vulnerabilities to ensure correct network operation. Due to the constraints of the WSN nodes, traditional intrusion detection solutions cannot satisfy security needs. As a result, one of the most challenging concerns is ensuring the security of the WSN. Intrusion prevention system has steadily been employed as the first line of defence in WSN as a practical approach to ensure information security. The author in [3] was only concerned with optimizing node proportions for intrusion detection following survey distribution. The goal of [4] was to employ offline detection for intrusion prevention and threat prevention to protect against attacks. Some abnormality detection methods attempted to discover abnormalities by observing normal activities [5].



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

Using classification techniques, penetration testing can assist the administration in successfully distinguishing "normal" from "abnormal" endpoints. The reference [6] looked for irregular network throughput and transmission power indication irregularities to locate the jammer. To protect against transportation assaults, reference [7] aimed to identify advanced infrastructure inconsistencies. Because the actual immune cells and the IDS are identical, intrusion prevention solutions based on the Artificial Immune System (AIS) is investigated in computerized networks and communication fields [8]. Artificial Immune Networks (LANs) and Negative Selection (NS) are the four divisions of AIS. An AIS based on Holland's Classifiers [9] was presented for detecting attacks. To accelerate convergence speed and eliminate the proteins slipping into the local minimum for optimizing, AIS with socialization was used [10]. In [11], the Hazard Theory is utilized to decrease false alerts.

In risk theory, a four-layer paradigm for networked IDS was built using dendritic cells [12]. Forrest et al. introduced the Negative Selection Algorithm (NSA), and it effectively reproduced the immunological resistance processes of the host immune for recognizing [13]. It may be utilized to solve anomaly [13] and error analysis problems. Because WSNs are highly homologous bodies in terms of system architecture and system attributes, such as self-learning, flexibility, memory pattern, personality, resilience, and so on, AIS has a lot of promise in WSNs [14]. Nevertheless, because nodes with limited funds, such as internal memory, battery capacity, and processing capabilities, the NSA has certain drawbacks when utilized in WSN intrusion prevention. To address the issues above, these authors propose WSN-NSA, WSN-based intrusion detection and prevention based on intrusion prevention model with V-detector technique and certain optimization strategies that reduce memory requirement and time to cost, and data processing utilization [15]. The primary strength is based on the immune paradigm, which automates a detection system in WSN that uses a better detection method. However, the model fails to provide privacy preservation support which needs to be addressed. This work concentrates on modelling efficient privacy preservation along with the intrusion detection mechanism. The following are the significant research contributions:

1) To consider available online resources for intrusion detection. Here, the NSL-KDD dataset is taken for predicting the flow of attacks over the network model;

2) To design an attention mechanism and intrusion detection approach using the modern learning approaches for predicting the traces of the network attacks;

3) To design a customized privacy preservation concept to protect the sensitive information that flows among the connected networks;

4) To evaluate various performance metrics like accuracy, precision, recall, F-measure and detection rate to highlight the significance of the model.

The work is provided as: Section 2 provides a comprehensive analysis on existing approaches; section 3 provides an elaborate discussion on the proposed methodology, including threat detection and privacy preservation. The numerical outcomes of the anticipated model are explained in section 4, with the research summary in section 5.

## 2. Related works

WSNs are gaining popularity, and their applications are being researched in various sectors. Wireless sensor security is now becoming increasingly vital. Therefore, intrusion detection is very crucial. Within the WSN security system domain, there are different existing works. We'll talk about the most recent significant results in this area. WSN detection in prior investigations has primarily relied on single-point independent observation, as described and implemented by [16] on a single detection node. When the number of defects caused by disobedience with rules exceeds the number of system failures produced by unintentional networks causes, an intrusion has occurred, according to intrusion detection methodology. Sharma et al. [17] built a Markov theoretical prediction system on a solitary sensor node using the time-domain concept. Offensive behaviour is identified if the exact value of the differential between expected and actual data traffic is greater than the specified threshold value. Single-point independent identification is not possible in WSN due to the resource constraints of the stations. Peer-to-peer cooperation monitoring is commonly employed in planar networking. Leveraging immunity theories, Wang et al. [18] created a multi detection mechanism. The surveillance agent is installed on each node, and the determination agency compares the data characteristics obtained. When a relay network is attacked, the Killer agent in the area is triggered, and the Killer product replies by isolating the aberrant node. Loven et al. [19] introduced a hierarchical intrusion detection approach that layers the WSN



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

network sequentially, which is mainly suited for wireless links. The first layer is the sensor nodes. The second level is the aggregate nodes. The third tier is the higher base stations, which can sense anomalies in incoming data, evaluate data, and determine if an intrusion has happened.

It is challenging to construct a mathematical template formulation due to the ELM algorithm's random initialization. KELM overcomes this challenge and demonstrates high model parameter resilience. Pitropakis et al. [20] advocated using the quick leave-one-out bridge to continuously model a kernel's supervised learning machine. The suggested technique enhances the detection accuracy of the previous kernels and neural network models, even though the randomized sample allocation significantly influences classifier performance. Singh et al. [21] developed a learning algorithm for multi-layer perceptrons, which they evaluated on the KDD CUP 99 dataset and proven helpful compared to past findings. Ferring et al. [22] used equality confined extreme learning computers to identify network intrusions and suggested an adjustable treatment efficiency for input layers, resulting in a system with a good attacker detection accuracy and a rapid number of iterations.

Bangui et al. [23] proposed the adaptive chicken swarm optimization method as an effective clustering tool. According to their findings, the hierarchical intrusion detection model outperforms the traditional technique in terms of accuracy. WSN's lifespan and scalability are increased, and its time consumption is significantly decreased. A two-stage classification model termed adaptive SVM is also presented, which reports fraudulent network devices using a formal recognition mechanism. The model employs DBN morphological operations to model, and the ELM and final output are selected by a majority of votes, according to [24] – [25]. This approach raises the computation intricacy while improving accuracy and lowering the number of false positives. The sources above focus on solving WSN intrusion prevention issues using prior learning techniques or solving intrusion detection problems in conventional networks using machine learning approaches (See Table 1).

Table 1: Various existing methods with their benefits

| Methods | Benefits |
|---|---|
| Statistics: monitors network traffic and process the data utilizing complicated statistical methods [26] | •Requires huge amount of statistical knowledge<br>•Less accurate but easier for implementation<br>•Real-time |
| Pattern: detects the information's characteristics, shapes, and structures [27] | •Easier implementation<br>•Hash function is adopted for prediction |
| Rule-based: detects a military strike on the suspect data traffic using an attacking "signature." [28] | •Because regulations need pattern recognition, the computation complexity of rule-based devices might be quite expensive.<br>•Estimating what activities will take place but when is extremely difficult.<br>•Determining all conceivable attacks necessitates a considerable number of rules.<br>•Low frequency of false positives<br>•High rate of detection |
| State: evaluates a series of events to spot any potential threats [29] | • Self-training and probabilistic<br>•Low false rate. |
| Heuristic: recognizes any abnormal behaviour that is outside of the average [30] – [32] | •Knowledge and practice are required.<br>•Learning that is both experimental and developmental |

## 3. Methodology

This work did a big study to make things more secure and fix problems with how features work on computer networks. We made a smart computer program using a special kind of network called a deep neural network to spot attacks on the network. We also made another program to keep people's privacy safe on the network. Here, we tested everything using a programming language called Python and a set of data called NSL-KDD. Check out Fig 1 to see how our privacy program works.



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

**Intrusion detection framework**

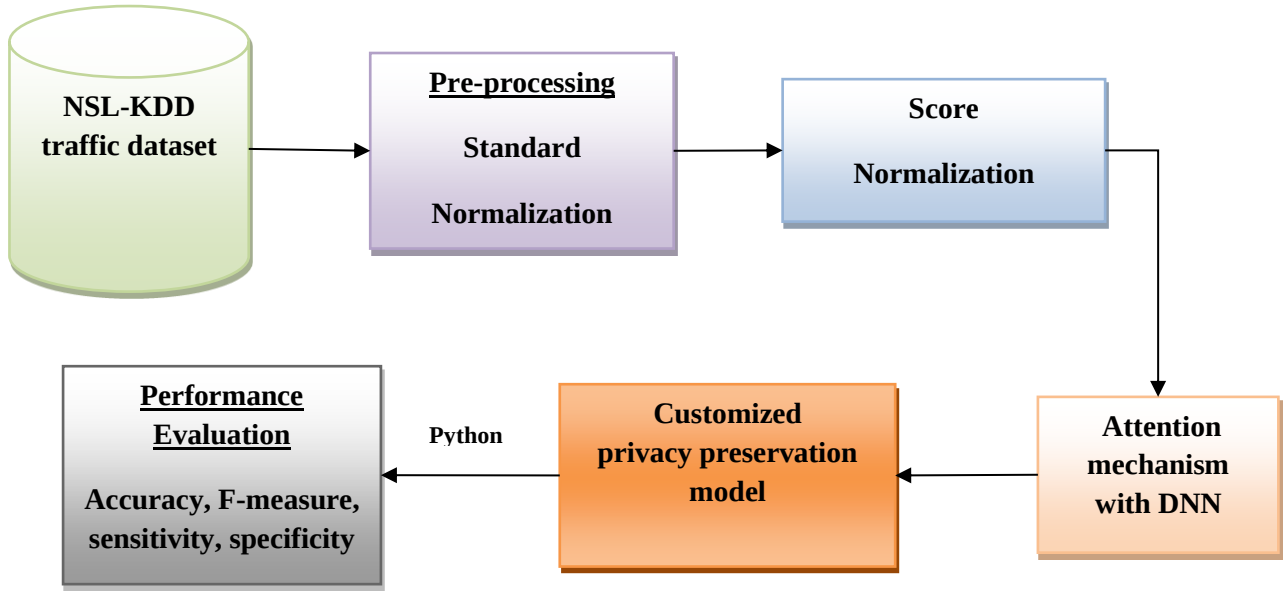


Figure 1: Block of $a-DNN$ model

**a. NSL-KDD**

Travallaee et al. [29] recommended the NSL-KDD dataset in 2009, which various inherent disadvantages identified over the KDD-CUP'99 dataset. The categories of the attacks are shown in Table 2.

Table 2: Attack types

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

**b. Standardization normalization**

It's a method to structure data in a database. The idea is to organize it to reduce repetition and keep related data in separate tables. This helps prevent problems when making changes to the database, like adding, removing, or updating information. Standardization is a way to make data more consistent by adjusting it based on the average and spread of values. This means making the average value 0 and the standard deviation 1. We can represent this mathematically, like in Equation (1).

$$Z = \frac{x - \mu}{\sigma} \quad (1)$$

In this equation, μ represents the average (mean) of the given data distribution, and σ represents the standard deviation (SD) of the distribution. The Z value is called the standard score, which tells us how many SDs above or below the mean a particular observation is. For example, if the Z value is 2, it means the observation is two SDs above the mean. This method of standardization works well when the input data follows a normal distribution and doesn't have any specific boundaries. It's commonly used in machine learning when we assume certain things about how the data is spread out.



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

**c. Xcorr normalization**

This is a method used to compare patterns in time series data over a period of time. It helps us understand how different time series relate to each other. Cross-correlation is a popular tool for finding patterns by measuring how similar two sequences are when one is shifted compared to the other. It's often used to normalize data and reduce the impact of individual data points. By adjusting certain parameters, like the peak matching function, we can maximize the cross-correlation to find the best match between sequences. We can express the optimal value mathematically, as shown in Equation (2).

$$J(m) = \int_{M1}^{M2} s(M-m, A)t(M)dM \tag{2}$$

Here, $M1$ specifies lower bound, M2 specifies as upper bound and $m$ is measured as the window value. The Correlations window allows for correlations between two or more input distributions.

**d. Model overview**

The NSL-KDD dataset comprises independent data with semantic vectors, i.e. $v_t^1, v_t^2, v_t^3 \; and \; v_t^4$. The proposed model includes attention-based LSTM (Long Short Term Memory) and a plain Deep Neural Network (DNN), used for vector representation $h_t^1, h_t^2, h_t^3, and \; h_t^4$. Finally, these vectors are concatenated as a feature representation to feed as an input to the softmax layer. The proposed model recommends the complete prediction of test data. First, the vector representation is fed to the attention-based LSTM model to establish the relationship among the dataset. The prediction dataset is independent and is provided with the security key to maintain privacy.

**e. Recurrent Neural Networks (RNN)**

RNN provides the superior performance to learn the data pattern explicitly with the sequential data. Moreover, with positive ordered data sequences, forward RNN from the input information and eliminate the latter using the present state. Furthermore, the input acquired after the present state is essential for inference. Thus, the work adopts bidirectional-RNN to avoid the shortcomings in forwarding RNN. It is composed of backward and forward RNN, which uses present and future information. The forward RNN is fed to the sequence from the starting to ending to evaluate the hidden forward states sequences. At the same time, the backward RNN understands the data sequence reversely and computes the sequential backward hidden state. The anticipated model uses the element-wise multiplying operation to merge the two hidden states, outperforming other existing approaches. However, the drive of handling the vanishing gradient difficult is incurable when the sequence length is considered to handle. Every forward LSTM unit is composed of memory cell state $S_i$, managed by the sigmoid gates: input $I_i$, forget $F_i$ and output $O_i$ gate. The intermediate gate is used to determine the information acquired from cell state $S_i$, while the input gate $I_i$ specifies the data that needs to be stored. At last, the output gate $O_i$ specifies what determines the information of cell state $S_i$, an output. The hidden state $\vec{h}_i$ of forwarding LSTM is computed as in Eq. (3) to Eq. (7):

$$F_i = \sigma\left(W_f[h_{i-1}; v_i) + b_f\right) \tag{3}$$
$$I_i = \sigma\left(W_i\,[h_{i-1}; v_i] + b_i\right) \tag{4}$$
$$O_i = \sigma(W_o\,[h_{i-1}; v_i] + b_o) \tag{5}$$
$$S_i = F_i \otimes S_{i-1} + I_i \otimes \tanh(W_s\,[h_{i-1}; v_i] + b_s) \tag{6}$$
$$\overrightarrow{h_i} = O_i \otimes \tanh(S_i) \tag{7}$$

Here, $[h_{i-1}; v_i] \in R^{q+m}$ specifies the concatenation of prior hidden state $h_{i-1}$ and embeds vector $v_i, q$ specifies the hidden state $h_i, W_f, W_i, W_o, W_s \in R^{q*(q+m)}$ specifies the weight matrix that needs to be learned, and $b_f, b_i, b_o, b_s \in R^q$ specifies the bias vector, $\sigma$ specifies the sigmoid function, and $\otimes$ determines the element-wise multiplying operation. Similarly, the hidden state $\overleftarrow{h}_i$ is attained using the LSTM cell. The hidden state $h_i$ of $bi - LSTM$ is expressed as in Eq. (8):



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

$$h_i = \vec{h}_i \otimes \overleftarrow{h}_i \quad (8)$$

**g. Attention mechanism**

To improve the ability of bi-LSTM, this work utilizes attention LSTM. The representation is provided in Fig 2. It uses RNN to transform input sequence $< x_1, x_2, \dots, x_t >$ into the hidden output vector $< h_1, h_2, \dots, h_t >$ where some data pattern information are disappear or ignored. Therefore, the attention mechanism is used to drive context vector $c_t$, which assists in capturing more attack information during risk prediction tasks and attains superior performance while executing the sequential data. Vector $c_t$ is evaluated using Eq. (9):

$$c_t = \sum_{i=1}^{t-1} \propto_{t_i} h_i \quad (9)$$

Where $h_i$ specifies $i^{th}$ hidden state, $\propto_{t_i}$ specifies the vector to capture the current hidden state weights $h_i$. $\propto_{t_i}$ is attained with Eq. (10) to Eq. (11):

$$\propto_{t_i} = W_{\propto} h_i + b_{\propto} \quad (10)$$

$$\propto_t = softmax\left([\propto_{t1}, \propto_{t2}, \dots, \propto_{t-(t-1)}]\right) \quad (11)$$

Here, $W_{\propto} \in R^q$ and $b_{\propto} \in R$ specify the learned parameter and determine the weight and bias. Based on Eq. (11), this work adopts the softmax function to execute attenuated weight vector $<\propto_{t1}, \propto_{t2,} \dots, \propto_{t(t-1)}>$ where size is $t-1$, and the element specifies the hidden state significance. Vector element $\alpha_t$ is derived from the softmax $c_t$. With the present hidden state $h_t$ and context vector $c_t$ is expressed above, and captures the attention hidden state by integrating the vector $\tilde{h}_t$ based on Eq. (12):

$$\tilde{h}_t = \tanh(W_c\,[c_t; h_t]) \quad (12)$$

Here, $W_c \in R^{r*2q}$ specifies the learned weighted matrix and $r$ specifies $\tilde{h}_t$ dimensionality. In this regard, $\tilde{h}_t$ selects the final representation of $'t'$ times visited observation window. The attention-based LSTM is used to capture the absolute representation of vectors $\tilde{h}_t^1 \in R^{r1}$, $\tilde{h}_t^2 \in R^{r2}$ $and$ $\tilde{h}_t^3 \in R^{r3}$ for various attack-based data; however, to attain the lower-dimensional vector $\tilde{h}_t^4 \in R^{r4}$ for the DNN module. The input from this layer is fed to the softmax layer based on the concatenation of $\tilde{h}_t^1, \tilde{h}_t^2, \tilde{h}_t^3, \tilde{h}_t^4$ to evaluate the probability distribution $\hat{y}$ for condition prediction during the attack prediction and it is expressed as in Eq. (13):

$$\hat{y} = softmax\,(W_x\left[\tilde{h}_t^1, \tilde{h}_t^2, \tilde{h}_t^3, \tilde{h}_t^4\right] + b_x) \quad (13)$$

Where $W_x \in R^{u*(r_1+r_2+r_3+r_4)}$ and $b_x$ specifies the parameters that are learned establishing the bias and weight, $u$ specifies the number of stages and $u$ is set as 2 in this work.



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

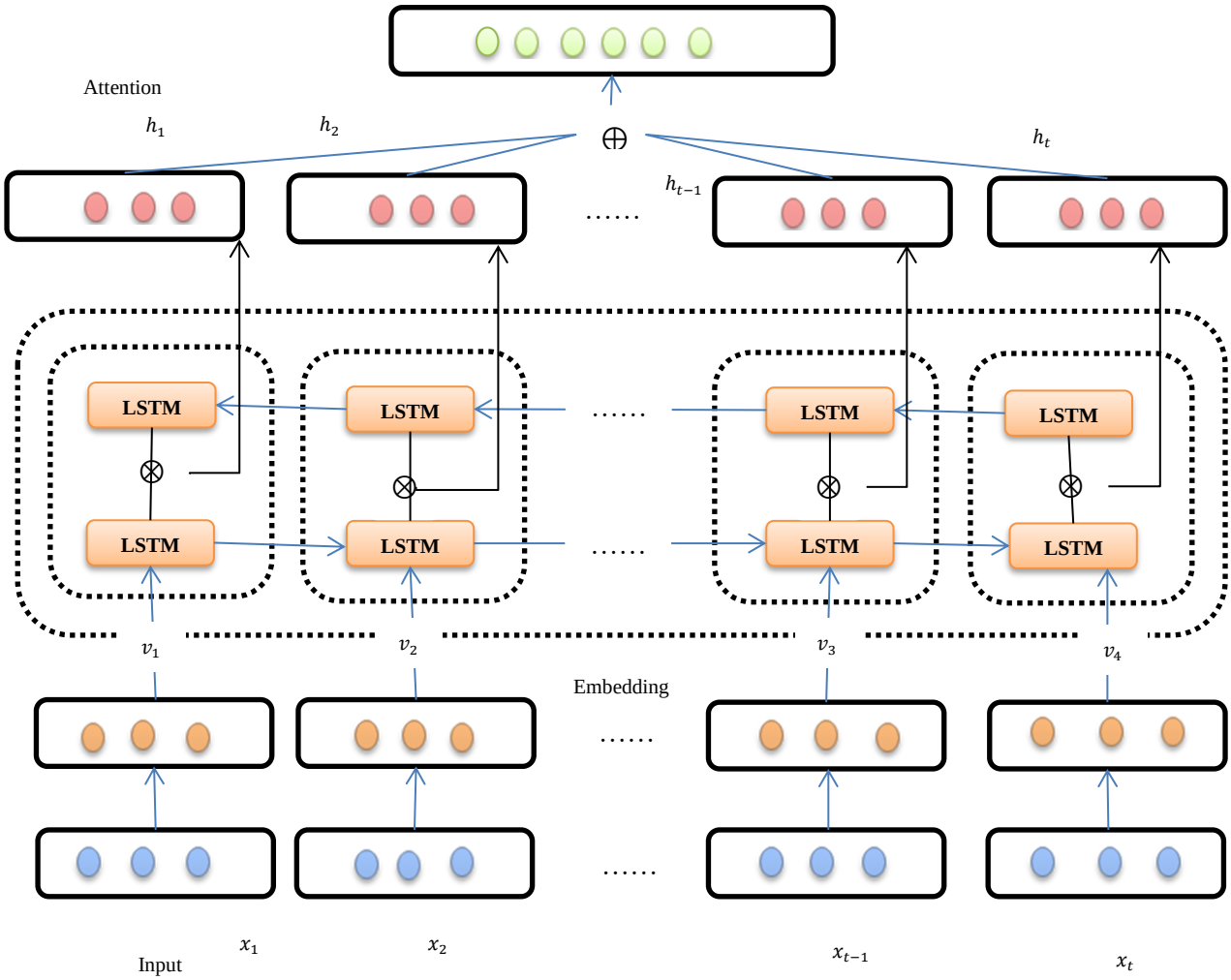


Figure 2: Attention-based mechanism

**h. Loss function**

To attain appropriate model parameters, the model employed cross-entropy among the information $y_i$ and predicted to the layers $\hat{y}_i$ is evaluated to measure the loss as in Eq. (14):

$$Loss = -\frac{1}{N}\sum_{i=1}^{N} y_i^T \log \hat{y}_i + (1-y_i)^T \log(1-\hat{y}_i) \quad (14)$$

Here, $y_i$ specifies the high-risk attack indicators for predicting the network attacks for various attack types. It identifies high-risk cases when $y_i$ is equal to 1, whereas in normal cases, $y_i$ is equal to 0. Here, $\hat{y}_i$ specifies the risk score of $i^{th}$ data evaluated by the model. Here, optimization is computed using the mini-batch stochastic gradient descent and back-propagation model, computed with Python.

**i. Customized privacy preservation**

This privacy preservation model for attack detection over sensor networks is highly solicited to measure the customized functionality of multiple hops. The more significant number of hops requires a higher privacy preservation level. The user needs to evaluate sensitive data, incoming data to the network and data transferred to the other node. The sensitive data should guarantee privacy before sharing it outside. The present model



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

needs to ensure a uniform privacy protection level that is not feasible. To show the issue clearly, a sample with the customized privacy protection levels is provided below: Let $\in_i$ and $\in_{i+1}$ be two diverse privacy preservation levels fulfilling $\in_{i+1} > \in_i$ and $M_{\in_i} \to \in_{i+1}: D \to \Delta(y^2)$ is a randomized mechanism. The user $u_i$ deals with the noisy output $\{y_{i,j}, y_{i+1,j}\}$ to other users which rely on $\in_i$ and $\in_{i+1}$, respectively. The network needs to ensure the incoming data collude and are malicious to steal more appropriate sensitive information. The privacy protection mechanism relies on Eq. (15):

$$M_{DP}(\epsilon_i + \epsilon'_{i+1}) = M_{DP}(\epsilon_{i+1}) \tag{15}$$

Here, $\epsilon'_{i+1}$ specifies the privacy protection over the malicious activities. The composition mechanism is expressed as in Eq. (16):

$$M_{DP}(\epsilon'_{i+1}) = M_{DP}(\epsilon_{i+1} - \epsilon_i) \tag{16}$$

It is derived based on $\epsilon'_{i+1} < \epsilon_{i+1}$. The outcomes specify that the privacy protection mechanism is based on differential privacy's attack flow. This customized privacy preservation is extended from the traditional differential privacy model.

**4. Evaluation metrics**

In this study, we used the NSL-KDD dataset to test how well a-DNN (adaptive Deep Neural Network) can improve intrusion detection on networks. We ran our experiments on a computer with an Intel (R) Core i5 processor running at 2.71 GHz and 8GB of RAM. We used Python for the simulations, and the classifier model was based on the kernel. We measured the performance of a-DNN using various metrics. These metrics include True Positive (TP), which is when an actual anomaly is correctly identified as an anomaly; True Negative (TN), which is when normal samples are correctly identified as normal; False Positive (FP), which is when a normal sample is incorrectly classified as an anomaly; and False Negative (FN), which is when an actual anomaly is incorrectly classified as normal. These metrics help us understand how well the a-DNN performs in detecting network intrusions.

1) Accuracy- It is depicted as the proportion of total records in the NSL-KDD testing set. It is expressed as in Eq. (17):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{17}$$

2) Precision- It depicts the proportion of predicted intrusion to the predicted intrusion over the testing process. It is expressed as in Eq. (18):

$$Precision = \frac{TP}{TP + FP} \tag{18}$$

3) Recall: It is depicted as the proportion of predicted intrusion to the total intrusion samples over the testing set. It is expressed as in Eq. (19):

$$Recall = \frac{TP}{TP + FN} \tag{19}$$

4) F-measure: It is depicted as the ratio both recall and precision as shown in Eq. (20):

$$F - measure = \frac{2 * Precision * Recall}{Precision + Recall} \tag{20}$$

The experiment aimed to test how effective and efficient it is to use lower-level features in the classifier model for detecting different types of intrusions (normal or anomaly) on a dataset. It also looked at how well the model could handle different classes of intrusions, such as probe, U2R, R2L, and regular. The goal was to improve intrusion detection while reducing the computational complexity and storage needed.
The main focus was on finding the best way to represent the data and reduce the number of features used in the model. This is important because the kernel model used requires a lot of computational space and memory,



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

especially when dealing with support vectors. The performance of the a-DNN model was compared to other common approaches like SVM, Random Forest, Naive Bayesian, J48, and Decision Tree. Testing and training were done separately to see how well the model performed with fewer features. Through theoretical analysis, it was found that certain hidden parameters greatly influenced the accuracy and speed of training. Optimizing these hyper-parameters to create an efficient deep learning model for intrusion detection was challenging. However, by investigating sparse parameters in hidden units and tuning hyper-parameters using auto-encoders on the training dataset (NSL-KDD), better performance was achieved. The model showed improved performance in binary classification, demonstrating its effectiveness in intrusion detection

Table 3: Accuracy measurement

| **Iterations** | **5** | **10** | **15** | **20** | **30** |
|---|---|---|---|---|---|
| ***a − DNN*** | **85** | **87** | **92** | **95** | **97** |
| **SELF** | 81 | 86 | 90 | 92 | 95 |
| **Game theory** | 53 | 58 | 64 | 67 | 69 |
| **Single SVM** | 81 | 77 | 85 | 80 | 75 |
| **STL-IDS** | 64 | 70 | 70 | 75 | 79 |
| **Random forest** | 67 | 71 | 75 | 77 | 79 |
| **RNN** | 69 | 75 | 80 | 80 | 82 |

Table 4: Precision measurement

| **Iterations** | **5** | **10** | **15** | **20** | **30** |
|---|---|---|---|---|---|
| ***a − DNN*** | **78** | **89** | **94** | **96** | **99** |
| **SELF** | 75 | 86 | 92 | 94 | 99 |
| **Single SVM** | 70 | 77 | 85 | 89 | 93 |
| **STL-IDS** | 75 | 79 | 86 | 91 | 92 |

Table 5: Recall measurement

| **Iterations** | **5** | **10** | **15** | **20** | **30** |
|---|---|---|---|---|---|
| ***a − DNN*** | **69** | **80** | **84** | **90** | **90** |
| **SELF** | 65 | 76 | 81 | 86 | 88 |
| **Single SVM** | 49 | 54 | 57 | 59 | 60 |
| **STL-IDS** | 50 | 55 | 59 | 62 | 67 |

Table 6: F-measure measurement

| **Iterations** | **5** | **10** | **15** | **20** | **30** |
|---|---|---|---|---|---|
| ***a − DNN*** | **75** | **80** | **86** | **92** | **96** |
| **SELF** | 72 | 79 | 82 | 89 | 94 |
| **Single SVM** | 54 | 59 | 61 | 68 | 73 |
| **STL-IDS** | 58 | 62 | 66 | 71 | 80 |

Table 7: Time measurement

| **Iterations** | **5** | **10** | **15** | **20** | **30** |
|---|---|---|---|---|---|
| ***a − DNN*** | **277** | **350** | **500** | **1100** | **130** |
| **SELF** | 305 | 400 | 529 | 1125 | 1325 |
| **Single SVM** | 708 | 1360 | 1866 | 2601 | 3502 |
| **STL-IDS** | 414 | 466 | 674 | 1611 | 1360 |

Table 8: Detection rate comparison

| **Iterations** | **5** | **10** | **15** | **20** | **30** |
|---|---|---|---|---|---|
| ***a − DNN*** | **0.004** | **0.005** | **0.004** | **0.004** | **0.004** |
| **SILF** | 0.006 | 0.0061 | 0.0058 | 0.00487 | 0.004 |



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

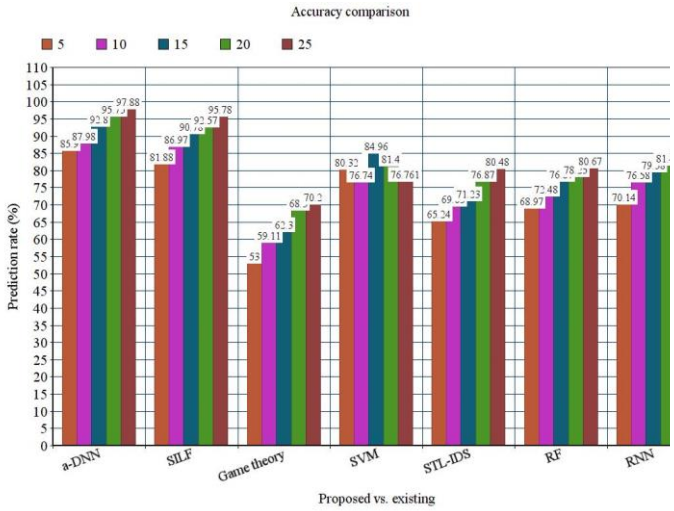


Figure 3 Accuracy evaluation

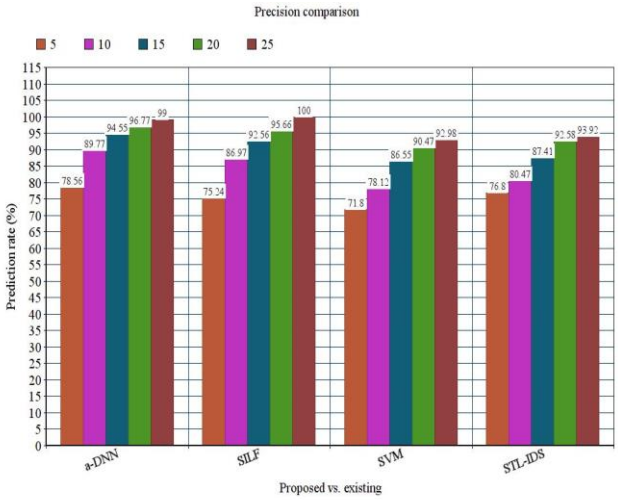


Figure 4 Precision evaluation

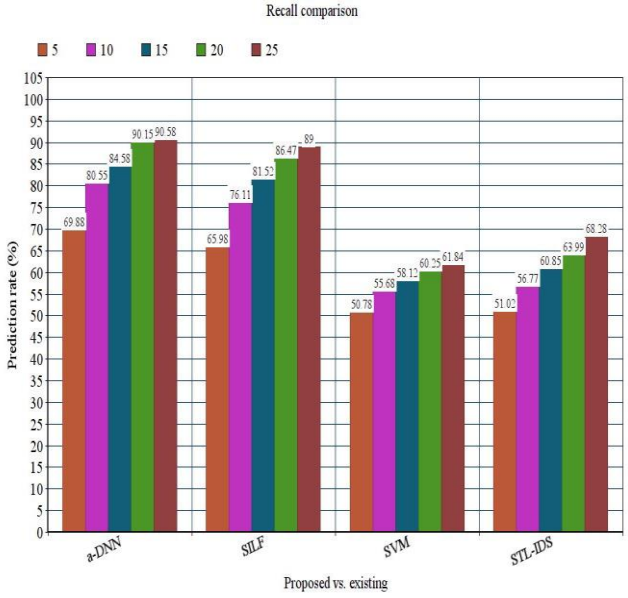


Figure 5: Recall evaluation

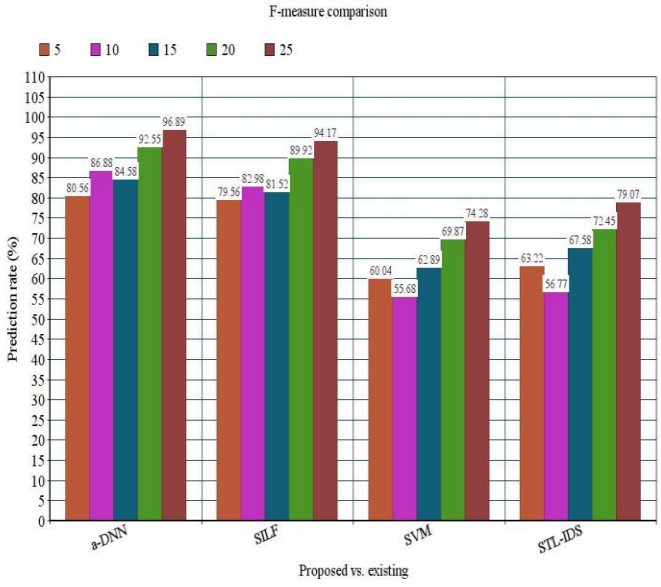


Figure 6: F-measure evaluation



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

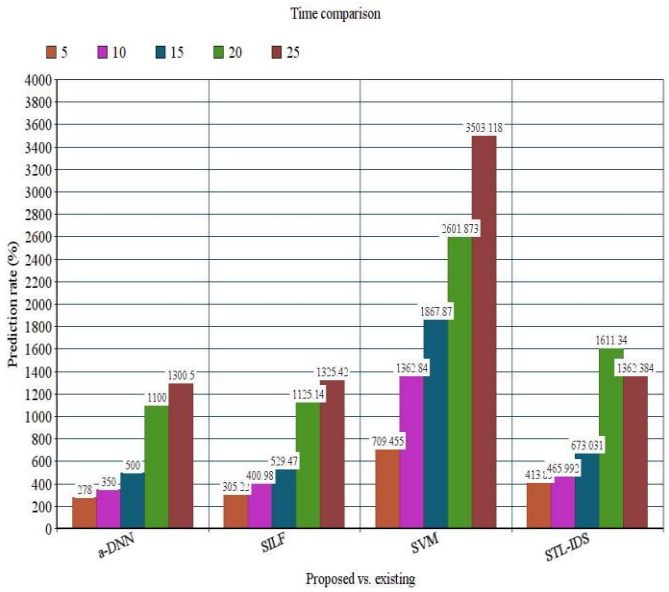


Figure 7: Time evaluation

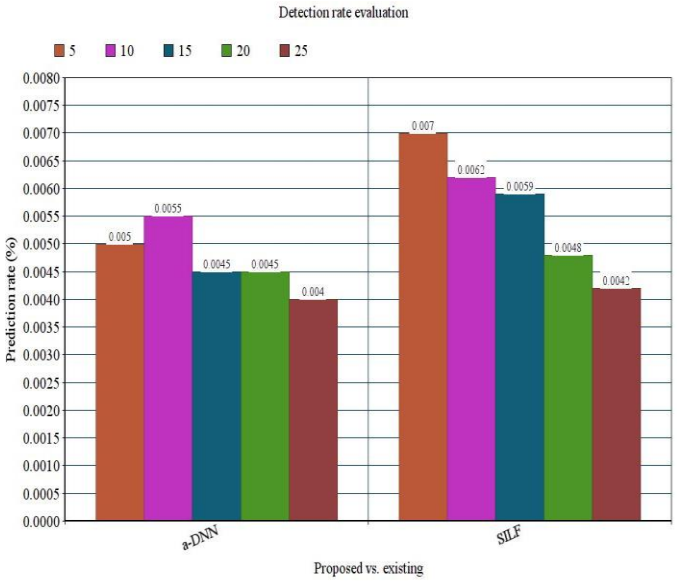


Figure 8: Detection rate evaluation

Table 1 compares the performance of the anticipated a-DNN with other methods like SILF, game theory, single SVM, STL-IDS, RF, and RNN (refer to Figures 3 to 8). The evaluation includes iterations of 5, 10, 15, 20, and 30. In the 30th iteration, the accuracy of the a-DNN model is 97.88%, surpassing other models by 2.1%, 27.68%, 21.119%, 17.4%, 17.21%, and 14.6%. Table 5 shows a comparison of precision between a-DNN and other approaches. The precision of a-DNN is 99%, slightly lower than SILF but higher than single SVM and STL-IDS by 6.02% and 5.08%, respectively. Table 6 compares recall rates, where a-DNN achieves 90.58%, outperforming other approaches by 1.58%, 28.74%, and 22.3%. In Table 7, the F-measure comparison indicates that a-DNN achieves 96.89%, surpassing other approaches by 2.72%, 22.61%, and 17.82%. Table 8 displays the execution time, with the a-DNN model consuming 1300.5 seconds, which is less than SILF, SVM, and STL-IDS. The detection rate of the a-DNN model is 0.0040, higher than existing approaches. These analyses collectively demonstrate that the model performs well in predicting attacks while maintaining strong privacy preservation policies.

## 5. Conclusion

This research concentrates on modelling a feasible approach for intrusion detection using a DL approach known as attention-based DNN and compares the anticipated model performance with various other methods. A customized privacy preservation concept is provided to establish security and privacy among the network nodes. The functionality of the anticipated $a - DNN$ is compared with various other approaches where the model gives better prediction accuracy of 97.88%, which is substantially higher than other models. Compared to the conventional learning approaches, the anticipated model predicts attack traces over the network. However, the optimality of the prediction outcome is not measured. Thus, it leads to adopting the meta-heuristic optimization approach, which will be addressed later.

## References

[1] Alsarhan, A., Al-Dubai, A. Y., Min, G., Zomaya, A. Y., & Bsoul, M. (2018). A new spectrum management scheme for road safety in smart cities. IEEE Transactions on Intelligent Transportation Systems, 19(11), 3496–3506.

[2] Bitam, S., Mellouk, A., & Zeadally, S. (2015). Vanet-cloud: A generic cloud computing model for vehicular ad hoc networks. IEEE Wireless Communications, 22(1), 96–102.

[3] Huang, Z., Ruj, S., Cavenaghi, M. A., Stojmenovic, M., & Nayak, A. (2014). A social network approach to trust management in vanets. Peer-to-Peer Networking and Applications, 7(3), 229–242

[4] Kang, M. J., & Kang, J. W. (2016). Intrusion detection system using deep neural network for in-vehicle network security. PloS ONE, 11(6), e0155781.



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

[5] Sedjelmaci, H., Senouci, S. M., & Abu-Rghef, M. A. (2014). An efficient and lightweight intrusion detection mechanism for service-oriented vehicular networks. IEEE Internet of Things Journal, 1(6), 570–577.
[6] Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). A deep learning approach to network intrusion detection. IEEE Transactions on Emerging Topics in Computational Intelligence, 2(1), 41–50.
[7] Sathya Preiya V, Kumar VDA. Deep Learning-Based Classification and Feature Extraction for Predicting Pathogenesis of Foot Ulcers in Patients with Diabetes. Diagnostics. 2023; 13(12):1983. https://doi.org/10.3390/diagnostics13121983.
[8] Balakrishnan C, Ambeth Kumar VD. IoT-Enabled Classification of Echocardiogram Images for Cardiovascular Disease Risk Prediction with Pre-Trained Recurrent Convolutional Neural Networks. Diagnostics (Basel). 2023 Feb 18;13(4):775. doi: 10.3390/diagnostics13040775. PMID: 36832263; PMCID: PMC9955174.
[9] Sun, G., Sun, S., Sun, J., Yu, H., Du, X., & Guizani, M. (2019). Security and privacy preservation in fog-based crowdsensing on the Internet of vehicles. Journal of Network and Computer Applications, 134, 89–99.
[10] Zaidi, K., Milojevic, M. B., Rakocevic, V., Nallanathan, A., & Rajarajan, M. (2015). Host-based intrusion detection for vanets: A statistical approach to rogue node detection. IEEE Transactions on Vehicular Technology, 65(8), 6703–6714.
[11] Zhang, C., Chen, K., Zeng, X., & Xue, X. (2018). Misbehaviour detection based on support vector machine and dempster-Shafer theory of evidence in vanets. IEEE Access, 6, 59860–59870.
[12] Hemamalini, Selvamani, and Visvam Devadoss Ambeth Kumar. 2022. "Outlier Based Skimpy Regularization Fuzzy Clustering Algorithm for Diabetic Retinopathy Image Segmentation" Symmetry 14, no. 12: 2512. https://doi.org/10.3390/sym14122512.
[13] Kumar, V.D.A., Sharmila, S., Kumar, A. et al. A novel solution for finding postpartum haemorrhage using fuzzy neural techniques. Neural Comput & Applic 35, 23683–23696 (2023). https://doi.org/10.1007/s00521-020-05683-z
[14] V. D. A. Kumar, M. Raghuraman, A. Kumar, M. Rashid, S. Hakak and M. P. K. Reddy, "Green-Tech CAV: Next Generation Computing for Traffic Sign and Obstacle Detection in Connected and Autonomous Vehicles," in IEEE Transactions on Green Communications and Networking, vol. 6, no. 3, pp. 1307-1315, Sept. 2022, doi: 10.1109/TGCN.2022.3162698.
[15] Soleymani, S. A., Abdullah, A. H., Zareei, M., Anisi, M. H., Vargas-Rosales, C., Khan, M. K., et al. (2017). A secure trust model based on fuzzy logic in vehicular ad hoc networks with fog computing. IEEE Access, 5, 15619–15629.
[16] Zhou H et al. (2020) Evolutionary V2X technologies toward the Internet of vehicles: Challenges and opportunities. Proceedings of the IEEE 108(2):308–323
[17] Awais Javed Muhammad, Zeadally Sherali, Hamida Elyes Ben (2019) Data analytics for cooperative intelligent transport systems. Vehicular Commun 15:63–72
[18] Afzal Z, Kumar M (2020) Security of vehicular Ad-Hoc networks (VANET): a survey. In: Journal of Physics: Conference Series. Vol. 1427. 1. IOP Publishing
[19] Tang F et al. (2019) Future intelligent and secure vehicular network toward 6G: Machine-learning approaches. Proceedings of the IEEE 108(2):292–307
[20] Ahmad T, Anwar MA, Haque M (2020) Machine learning techniques for intrusion detection. In: Handbook of research on intrusion detection systems. IGI Global, pp 47–65
[21] Agarwal Y, Jain K, Karabasoglu O (2018) Smart vehicle monitoring and assistance using cloud computing in vehicular Ad Hoc networks. Int J Transport Sci Technol 7(1):60–73
[22] Abhishek Kumar, Kamred Udham Singh, Visvam Devadoss Ambeth Kumar, Tapan Kant, Abdul Khader Jilani Saudagar, Abdullah Al Tameem, Mohammed Al Khathami, Muhammad Badruddin Khan, Mozaherul Hoque Abul Hasanat, Khalid Mahmood Malik, " Robust Watermarking Scheme for NIfTI Medical Images", Vol.71, No.2, 2022, pp.3107-3125, doi:10.32604/cmc.2022.022817
[23]V.D.Ambeth Kumar and M.Ramakrishan (2013), “Temple and Maternity Ward Security using FPRS” in the month of May for the Journal of Electrical Engineering & Technology (JEET) ,Vol. 8, No. 3, PP: 633-637.
[24] Sharma Sachin, Mohan Seshadri (2020) Cloud-Based Secured VANET with Advanced Resource Management and IoV Applications. In: Connected Vehicles in the Internet of Things. Springer, 2020, pp. 309–325



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024

[25] Wang W, Wu L, Qu W, Liu Z, Wang H (2021) Privacy-preserving cloud-fog–based traceable road condition monitoring in VANET. Int J Netw Manag 31(2):e2096
[26] Lovén L et al. (2019) EdgeAI: a vision for distributed, degenerative artificial intelligence in future 6G networks. In: The 1st 6G Wireless Summit (2019), pp 1–2
[27] Pitropakis Nikolaos et al. (2019) A taxonomy and survey of attacks against machine learning. Computer Sci Rev 34:100199
[28] Singh T, Kumar N (2020) WITHDRAWN: Machine learning models for intrusion detection in IoT environment: a comprehensive review. In: Computer Communications, Elsevier. https://doi.org/10. 1016/j.comcom.2020.02.001
[29] Ferran Mohamed Amine et al. (2020) Deep learning for cyber security intrusion detection: approaches, datasets, and comparative study. J Information Secure Appl 50:102419
[30] Bangui H, Ge M, Buhnova B (2018) Exploring big data clustering algorithms for Internet of things applications. In: IoTBDS, pp 269–276
[31] Osanaiye O, Alfa A, Hancke G (2018) A statistical approach to detect jamming attacks in wireless sensor networks. Sensors 18(6):1691
[32] Chonka Ashley et al. (2011) Cloud security defence protects cloud computing against HTTP-DoS and XML-DoS attacks. J Netw Computer Appl 34(4):1097–1107
[33] Belenko V, Krundyshev V, Kalinin M (2018) Synthetic datasets generation for intrusion detection in VANET. In: Proceedings of the 11th international conference on security of information and networks. pp 1–6
[34] Bangui Hind et al. (2017) Multi-criteria decision analysis methods in the mobile cloud offloading paradigm. J Sensor Actuator Netw 6(4):25
[35] Grover J, Laxmi V, Gaur MS (2011) Misbehavior detection based on ensemble learning in vanet. In: International Conference on Advanced Computing, Networking and Security. Springer, pp 602–611
[36] Mehdi MM, Raza I, Hussain SA (2017) A game theory-based trust model for vehicular Ad hoc networks (VANETs). Computer Netw 121:152–172
[37] Liang J et al. (2019) A filter model for intrusion detection system in vehicle Ad Hoc networks: a hidden Markov methodology. Knowl-Based Syst 163:611–623.
[38] P. Sherubha, P Amudhavalli, SP Sasirekha, “Clone attack detection using random forest and multi-objective cuckoo search classification”, International Conference on Communication and Signal Processing (ICCSP), pp. 0450-0454, 2019.
[39] S. Dinesh, K. Maheswari, B. Arthi, P. Sherubha, A. Vijay, S. Sridhar, T. Rajendran, and Yosef Asrat Waji, “Investigations on Brain Tumor Classification Using Hybrid Machine Learning Algorithms”, Hindawi Journal of Healthcare Engineering, Volume 2022.



Doi : https://doi.org/10.54216/JCIM.130208
Received: January 09, 2024 Revised: Mrach 02, 2024 Accepted: May 01, 2024