Jestr

Journal of Engineering Science and Technology Review 15 (5) (2022) 158 - 169

JOURNAL OF
Engineering Science and Technology Review

Research Article

www.jestr.org

A Vulnerability based Attack Detection and Mitigation in Cloud SaaS Framework

Reddy Saisindhutheja[1,*], Gopal K Shyam[2] and Shanthi Makka[1]

[1]*Department of CSE, Vardhaman College of Engineering, Hyderabad, Telangana, India*
[2]*School of CSE, Cloud Computing lab, Presidency University, Bengaluru, Karnataka, India.*

Received 17 January 2021; Accepted 25 August 2022

Abstract

Cloud computing provides various cost-effective on-demand services to the user and so it is rising up as a real trend in the IT service model. More particularly, the security is the ultimate question in the mind of each cloud user. A huge interest is being paid by the research community to detect the attack exist in the network. This research work intends to introduce a novel framework that combines the attack detection and mitigation, which gives the SaaS provider to decide relevant fields to be extracted, when huge traffic is available. Deep learning-based attack detection is carried out and based on the vulnerability in the network, the model switches to mitigation process. Initially, the feature selection is carried out by Opposition Based Crow Search Algorithm (OCSA). The selected features are subjected to attack detection process via Deep Belief Network (DBN) model, where the presence of attacks is determined. Subsequently, the 'vulnerability assessment' is carried out by evaluating the risk level via correntropy variation features. This phase decides 'how vulnerable the network is' with the presence of attack. This decision is based on fixing a threshold on risk level (RL). Moreover, the decision tells whether to execute the 'mitigation process' or not. In the attack mitigation phase, bait-based mitigation process is carried out. The proposed vulnerability-based attack detection and mitigation system beat the traditional methods with a packet loss ratio of 16% and a throughput of 92%.

Keywords: Cloud Computing; Attack Detection and Mitigation; OCSA; DBN; Correntropy- Variation Features; Bait model.

1. Introduction

Over the decades, the digital devices like the smart phones and tablets are playing a ubiquitous role in human life [1], [2], [3]. These smart devices are typically free from wires and they permit the users to access their data (multimedia format) as well as applications wherever at any instance of time over the internet. A quick increase in universal internet practice entails an innovative means to accomplish the size, diversity and readiness of data [4], [5], [6]. Cloud computing is a booming technology that is significant in providing its users the reliable, on demand and scalable resources at any time with fewer infrastructures cost. Cloud computing is quickly gaining a lot of grounds in the current technological era [7], [8]. Moreover, this technology is the delivery of computing power that provides an easy way to access storage, servers, networking, software, and intelligence on the internet and manages the network-attached hardware of all application services through the web application [9], [10], [11], [12]. The companies like Amazon and Google have their own clouds and have taken their operations over it [13], [14], [15].

In spite of the increase in cloud usage, the security is still being a primary barrier that degrades the performance of the system and thereby reduces the reliability [16], [17]. The cloud computing security or cloud security is a sub-domain of information security, computer security, network security [18], [19]. As most of the government as well as privet sectors are relying upon the cloud as the only source of quicker data transfer; the users start worrying about the data security. "A recent study by insight has revealed that most businesses are shifting toward the cloud for efficiencies, but they continue to have concerns about security" [20], [21], [22], [23], [24]. The best way to secure the data against theft, leakage, and deletion is by identifying the attack nodes in the system.

There are three service layers of the cloud namely Infrastructure-as-a-service (IaaS), Platform-as-a-Service (PaaS), Software-as-a-Service (SaaS). SaaS is built upon PaaS, which in-turn built upon IaaS. So, as the properties may inherit, there is a probability of inheriting the risk too. So, we argue that SaaS is more prone to the attacks. Attacker gain unauthorized access to systems and infects them with malware or Trojan horse. The most common attacks include DDoS attack, data theft, unauthorised access etc [25], [26]. Most of the traditional security tools pick out threats based on the database of previous occurrences of malware.

Today, the industries are looking out for solutions that helps to broadcast the messages to companies that informs the existence of proactive threat in the website, portal or application. Each time the system gets updated at different levels of stack, at various layers of the application environment, including hardware, network or applications. There needs to be a tool to monitor the potential threats through constant surveillance. For achieving such a marvellous attack detection and mitigation model, the selection of optimal features is more significant via optimization algorithms [27], [28].

Apart from this, machine learning algorithms based on artificial intelligence is being the best detection model. Among these models, the deep learning seems to be the most promising approach in enhancing the attack detection accuracy. The major contributions of this research work are: (i) The decision about the attack mitigation framework is accomplished by evaluating the Risk Level (RL) via the

*E-mail address: thejasindhu@gmail.com
ISSN: 1791-2377
doi:10.25103/jestr.155.21

correntropy variation features. (ii) Vulnerability of the network is decided based on the RL, which decides the execution of bait-based mitigation process in network. The SaaS providers have the capability to design the detection module via this framework, which suits best to the features of attack vectors. This paper is structured as follows: Section 2 addresses the literature works done in the attack detection models. Section 3 describes about the proposed attack detection and mitigation in cloud: an overview. Section 4 depicts about the data encryption and feature selection. In addition, Section 5 manifests the attack detection and mitigation model. The results acquired with the proposed model are deliberated in Section 6. Finally, Section 7 offers conclusion.

2. Literature Review

A unique architecture for identifying the presence of a DDoS assault in the cloud has been disclosed by the authors at [29]. The "sparse AE for feature extraction and DNN for classification" stacking method was used to create the proposed architecture. Additionally, they can optimise the AE and DNN settings using a suitable tuning model. As a result, the overfitting issue was not present in the suggested model, which reduced the reconstruction error.

A CS_DDoS in public clouds has been introduced by the authors in [30]. In order to secure the recorded data and to determine the presence or absence of an attack using the classified results, the proposed CS_DDoS system categorised the incoming packets. Data packets are examined for signs of attacks during the detection phase, and malicious packets are prevented from using cloud services during the prevention phase. It was discovered that the CS_DDoS model performed better and required less time.

At [31], the authors offer a creative suggestion for the Cloud model's detection of intrusion. Here, a potent SCAE approach was used to automatically extract the robust low-dimensional properties. An SVM classifier was employed to categorise these retrieved features. This combination methodology was tested using the "KDD Cup 99 and NSL-KDD" dataset, and the findings demonstrated improved detection performance when compared to cutting-edge methods.

The authors of [32] have put forth a state-of-the-art security strategy to stop the DDoS attack brought on by IoT server malware. The authors' integration of the SDN paradigm with the cloud has decreased DDoS attacks on IoT servers. The LEDEM was additionally employed to locate and lessen the DDoS. The simulation results of the proposed approach demonstrated a high rate of accuracy in DDoS attack detection.

The fuzzy self-organizing maps-based DDoS mitigation technique was created in [33] by updating the neurons in the normal NN using fuzzy rules rather than the traditional Kohonen neural network model (FSOMDM). The authors have utilised this recommended mechanism's software-oriented traffic inspection property to both and identify DDoS assaults. Performance analysis of the proposed work had shown better classification accuracy.

A DDoS attack detection and mitigation models are built using an ICRPU and the feature selection method in [34]. The Hellinger distance function was utilised to analyse the traffic, and the recovered features were then used to classify the packets as DDoS and real request groups. The authors then routed the legitimate requests to the Normal Request Processing Unit and the DDoS requests to the ICRPU. Therefore, it was determined that the planned attempt had the best detection rate and the lowest FAR.

In [35], it is claimed that cloud-based services should analyse network flow in order to detect and stop FRC assaults. Since the authors compared the proposed method to a real-world benchmark, the proposed technique was more accurate and had lower overhead.

Convolution Recursively Enhanced Self Organizing Maps (CRESOM-SDNMS) are an SDN-based DDoS Attack Mitigation Scheme that is suggested in [36] as a way to fend off DDoS attacks in a cloud computing environment. To overcome the vector quantization problems, the authors have improved the topology preservation and initialization mechanisms in the SOM-based classification process. By minimising the FPR during DDoS mitigation, the presented technique was shown to have reduced FPR through simulated trials. Some of the characteristics and limitations of SaaS were addressed in Table 1.

Table 1. Characteristics and limitations of SaaS

Author and Reference	Methodology used	Characteristics	Limitations
Bhardwaj [29]	Hyperband Tuned Deep Neural Network	• Effectively mitigate illegitimate flows within the domain. • Lower detection time • Higher detection accuracy • Less computational complexity	• Lower precision and F1-score • Noisy, voluminous and high dimensionality data were not handled well
Sahi [30]	CS_DDoS system	• Improved Kappa coefficient and detection accuracy • Applicable for multiple attackers • Reduce bandwidth consumption.	• Tedious • Need to improve the security of records.
Wang [31]	SCAE and SVM	• Detection performance is high • Reduce controller's bottleneck	• Total detection performance is lower • Low Precision and f-measure.
Ravi and Shalinie [32]	LEDEM	• Exhibited higher accuracy rate in DDoS attack detection	• Show lower performance with increase in data rate
Pillutla and Arjunan [33]	FSOMDM	• Higher classifier accuracy (94%) • Flexible in controlling malicious data traffic flow.	• Higher computational complexity

			• Need to decrease the false positive percentage.
Bharot et al.[34]	ICRPU	• Provides best detection rate, accuracy	• Higher FAR • Attack detection rate is lower
Bhushan and Gupta [35]	T-distribution based flow-confidence technique	• Minimal computational overhead	• Lower attack flow detection • Lower confidence level.
Harikrishnaand Amuthan [36]	CRESOM-SDNMS	Reliable detection model • Precision, recall, and F-measure is higher	• Testing and training accuracy is lower • Lower Classification Accuracy

3. Proposed attack detection and mitigation in cloud

This research work intends to develop a attack detection and mitigation framework based on the network vulnerability which is given in Figure 1. Steps followed in this framework are as follows:

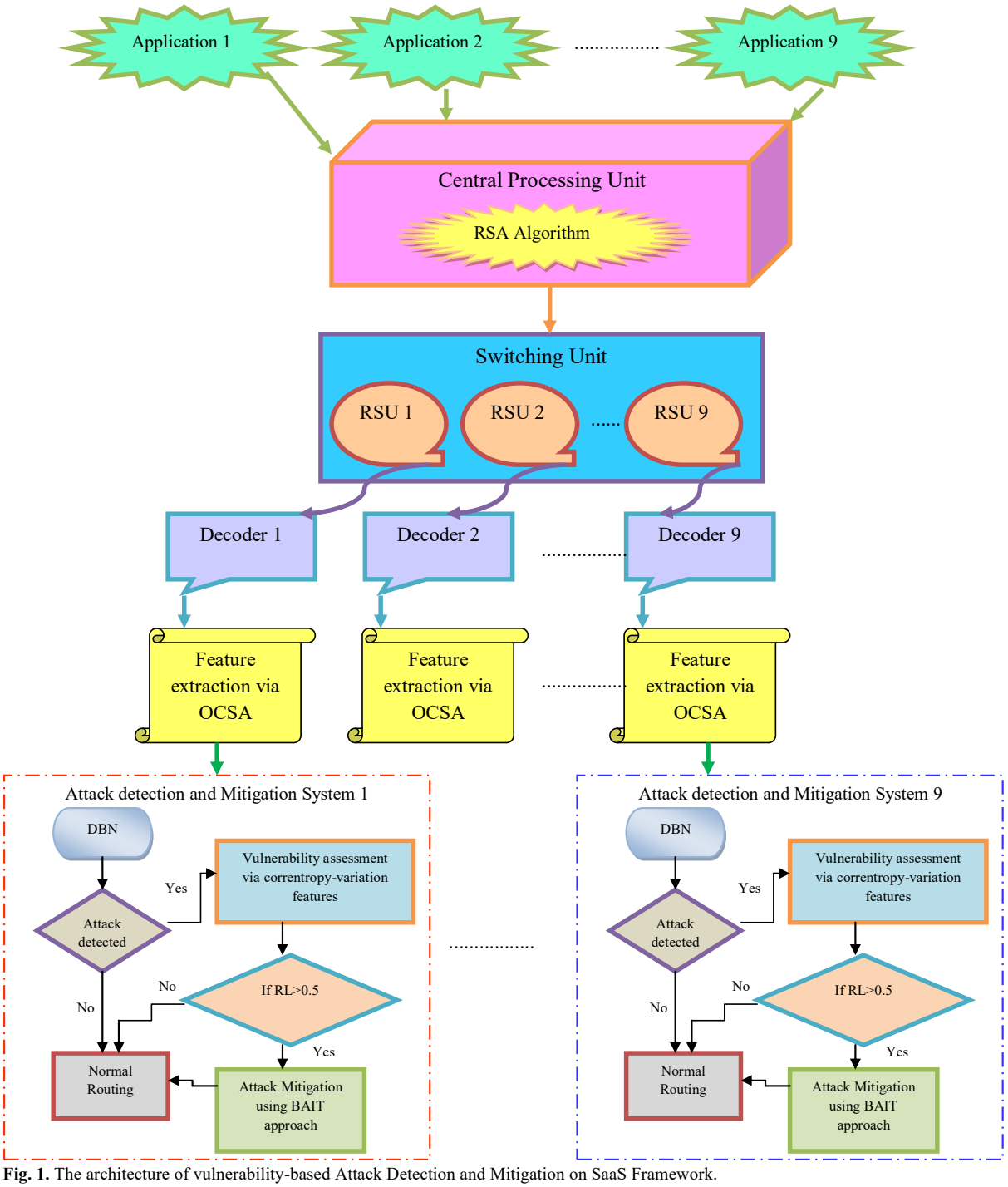


Fig. 1. The architecture of vulnerability-based Attack Detection and Mitigation on SaaS Framework.

Step 1: The collected original data In^{Data} is given to RSA for encryption and the resultant data based on encryption En^{Data} is decrypted in the switching system by mapping En^{Data} onto 9 RSU's. The count of RSU is to be equivalent to the count of applications and therefore, 9 RSU are selected here. The decrypted outcome is De^{Data}.

Step 2: Since, De^{Data} has huge count of features, the training process might be likely to be in accurate and tedious. Therefore, the optimal features *F* are ought to be chosen from De^{Data}. In this research work, the feature selection is accomplished using the with OCSA.

Step 3 (Attack Detection): Once the features *F* are selected, they are subjected for the attack detection process via DBN model, where the presence of attacks is determined.

Step 4 (Proposed Vulnerability Assessment Framework): In this step, the vulnerability assessment is carried out by evaluating the security risk. Here, the RL of attack is estimated using the correntropy-variation features. This phase decides how vulnerable the network is with the presence of attack by fixing a threshold on risk level.

Step 5 (Attack Mitigation): When the risk level is greater than 0.5, the control is transferred to the attack mitigation phase else the normal routing takes place. In the attack mitigation phase, lightweight bait-based mitigation process takes place.

4. Data encryption and feature selection

4.1 RSA Algorithm

The RSA is the most significant public-key cryptosystem for encrypting as well as decrypting the input data In^{Data}. This algorithm involves both the private key $Key_{private}$ and public key Key_{public}. The public keys are utilized for encrypting the plaintext messages to cipher text, such that it can be published to anyone [37]. At the same time, the encrypted data with the specific public key be recovered by decrypting the message suing the corresponding private key. This algorithm includes the following steps: (i) generation of the key, (ii) encryption and (iii) decryption. Among all these steps, the key generation process is the most crucial one as it makes the data secured and reliable.

The key pairs $Key_{private}, Key_{public}$ are ought to be computed for the public key algorithms. The key generation mechanism being the most important and the central part of the algorithm is computed using mathematics and not by the randomly generated numbers. The key generation of RSA consists of 5 steps:

- Step 1: Two large distinct prime numbers *I* and *J* are chosen.
- Step 2: Evaluate *G=I.I*, where *G* is the modulus of $Key_{private}, Key_{public}$.
- Step 3: Compute the $\emptyset(G)$, where *G* is the Euler's totient function.
- Step 4: Select an integer and it should be within the limit $1 < < E < < \emptyset(G)$.
    1. Guarantee that gcd $(e, \emptyset(G))=1$.
    2. Guarantee that $\emptyset(G)$ and *E* are co-prime.
- Step 5: Compute the value of an integer *D* as $D \equiv E^{-1} mod \emptyset(G)$.

At the end, the two asymmetric keys for decryption and encryption are generated. The *G* and *E* are in the public key, and *D* is in the private key. Key_{public} = *(G, E)* and $Key_{private}$= (D).

Encryption and decryption: Once the necessary variables that are utilized for the key generation are computed, then the message can be encrypted and decrypted. The message *Mes* is in the form of plaintext and it is to be send from the sender to the receiver in the form *m (i.e., 0≤ m≤ G)*. Since the generation of Key_{public} is done using *G* and *E*. Mathematically, the encryption and decryption process is defined as per Eq. (1) and Eq. (2), respectively. The decryption process takes place at the receiver end. Here, the cipher-text is decrypted with the private key.

$$Encryption: c \equiv Mes^{E}\ mod\ (G) \tag{1}$$

$$Decryption: Mes \equiv c^{D}\ mod\ (G) \tag{2}$$

4.2 Feature selection

The collection of the relevant features of Dec^{Data}enhances the system potential in attack detection. In this research work, the suitable features are opted by expelling both the irrelevant and duplicate features. To achieve this goal the OCSA [3] model is utilized and this OCSA is the conceptual blend of CSA and OBL respectively.

CSA is a new machine learning model that depends on behavioural actions and mastery of crows [38], [39], [40], [41]. The opportunity of this algorithm is driven from the routine of crow storing plenty of its eatables in undisclosed places and getting it again during a specified time. To guard its foodstuffs, it conceals and misleads other crows away from its original place. Such kind of unique behaviour of crows by possessing a good memory, societal behaviour, and the intake aspects; directed to the development of this algorithm. CSA is subjected to the following ideologies. (i) Crows live in a group. (ii) They recall the site of their hiding places. (iii) They chase others for stealing. (iv) They conceal their assets from being taken by a possibility.

OBL smashes out the calculation to get best results in some of the optimization problems [42], [43], [44], [45]. It is mainly encouraged by the opposite bond between the real-world entities. This algorithm ensures that, searching the random and its opposite bounces a huge probability to find most wanted regions of getting an optimal result. If the existing studies are very far from optimal result, then figuring out the opposite studies may fall close to optimal results.

The steps followed in the OCSA model is depicted below:

Step 1: The population size *Pop* of the search agent (crows), Flight length $Flight^{length}$, extreme quantity iteration *(K)* and awareness probability $awar^{prob}$are initialized. In addition, the crows (i.e., attributes) are initialized as per Eq. (3).

$$Crow_i = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{11} \\ A_{21} & A_{22} & \dots & A_{2D} \\ A_{n1} & A_{n2} & \dots & A_{nD} \end{bmatrix} \tag{3}$$

Step 2: The position corresponding to the flow of crows are randomly initialized as shown in Table 2. Here, the number of attributes is 115 so, *Pop*=115. The crows are subjected to either 1 or 0 in a random manner. If the crow's i^{th} position is 0, then its corresponding column is not selected for classification, else it is selected.

Step 3 (Opposite solution generation): Using Eq. (4), compute the opposite localization of the crow flocks.

$$Attribute_i = a_i + b_i - Attribute_i \quad (4)$$

For each of the solution $Crow_i$, there is a unique opposite solution $OCrow_i$. The computation of the opposite localization of the crow flocks $OCrow_i(A'_1)$ is defined as per Eq. (5). Here, $i \in 1, 2, \ldots, n$.

$$OCrow_i = \begin{bmatrix} A'_{11} & A'_{12} & \cdots & A'_{1D} \\ A'_{21} & A'_{22} & \cdots & A'_{2D} \\ A'_{n1} & A'_{n2} & \cdots & A'_{nD} \end{bmatrix} \quad (5)$$

Step 4: The memories of the crows are initialized and are recorded. Then, the iteration count is set as *K=1*.

Step 5: When the termination criteria is not met, then do for *i=1, 2, …, n*, select a crow arbitrarily to follow the n^{th} crow.

Step 6: Then, the random number R_n is created and if $R_n \geq aware^{prob(t)}$, then update the position of the solution using Eq. (6). This is the update expression for the positional update of the cheat crow *m* that follows the owner crow *n*.

$$S_m^{t+1} = S_m^t + R_m * Flight_m^{length(t)} * (B_n^t - S_m^t) \quad (6)$$

Here, B_n^t denotes the food source of owner crow *n* and is a random number R_m generated within the range 0 and 1. At the iteration, the t^{th} iteration, the m^{th} flight length is denoted as $Flight_m^{length(t)}$. In case, if $R_n < aware^{prob(t)}$, then update the random position.

Step 7: The new position of the crows is assessed and the memories of the crows are updated. The best position is chosen and the results are visualized.

From the feature selection out of 115 features, OCSA selects 36 features. The selected features are subjected for attack detection via DBN.

5. Attack detection and mitigation

5.1 DBN based attack detection

In 1986, Smolensky designed DBN with multiple layers, in which each layer is comprised of visible as well as hidden neurons. The hidden and visible neurons are fully interconnected [7]. In Boltzmann networks, the stochastic neuron's outcome is probabilistic and it is represented as *out*. Then, *out* in Eq. (7) is expressed by the probability *prob(τ)* in Eq. (8). In Eq. (8), τ denotes the parameter corresponding to the pseudo-temperature and it is utilized for controlling the noise level.

$$out = \begin{cases} 0 & with\ probability\ of\ 1 - prob(\tau) \\ 1 & with\ probability\ of\ prob(\tau) \end{cases} \quad (7)$$

$$prob(\tau) = \frac{1}{1+e^{\frac{-\tau}{T}}} \quad (8)$$

In the configured energy state *State*, the energy function E^{Fun} of the Boltzmann machine is defined as per Eq. (9). Here, $wt_{A,B}$ is the weight function between the neurons *A* and *B*, and ψ_A is their biases.

$$E^{Fun}(state) = -\sum_{A<B}(State_A)(State_B)\left(wt_{A,B}\right) - \sum_A \psi_A State_A \quad (9)$$

Then, sequentially compute the energy differential function ΔE^{Fun}. Then, for visible and hidden neurons *VIS, HID,* the new energy function acquired in the Restricted Boltzmann Machine (RBM) is given as per Eq. (10), Eq. (11) and Eq. (12), respectively.

$$E^{Fun}\left(\overrightarrow{VIS}, \overrightarrow{HID}\right) = \sum_{A,B} wt_{A,B}(VIS_A)(HID_B) - \sum_A VIS_A P_A - \sum_B HID_B Q_B \quad (10)$$

$$E^{Fun}\left(VIS_A, \overrightarrow{HID}\right) = \sum_B wt_{A,B} HID_B + P_A \quad (11)$$

$$E^{Fun}\left(\overrightarrow{VIS}, HID_B\right) = \sum_A wt_{A,B}\, VIS_A + Q_B \quad (12)$$

The RBM is trained in an unsupervised manner. Let the training set be *Train*, in which the matrix of rows represents $\overrightarrow{VIS}$. The maximal probability is attained with the weight assignment $Weight_m$ and it is shown in Eq. (13).

$$Weight_m = max_{weight} \prod_{\overrightarrow{VIS} \in Train} Prob\left(\overrightarrow{VIS}\right) \quad (13)$$

Using the energy function defined in Eq. (9), a probability is assigned to each *(VIS, HID)* pair. Here, the partition function is represented as *Z* and it is mathematically shown in Eq. (14).

$$Prob\left(\overrightarrow{VIS}, \overrightarrow{HID}\right) = \frac{1}{Z} e^{-E^{Fun}\left(\overrightarrow{VIS}, \overrightarrow{HID}\right)} \quad (14)$$

$$Z = \sum_{\overrightarrow{VIS}, \overrightarrow{HID}} e^{-E^{Fun}\left(\overrightarrow{VIS}, \overrightarrow{HID}\right)} \quad (15)$$

The network error is computed on the basis of Root Mean Square Error (RMSE) which is given in Eq. (16), which is the difference between predicted outcomes Pre and the actual outcomes Act. Mathematically, the objective function or the fitness function is shown in Eq. (17).

$$RMSE = Pre - Act \quad (16)$$

$$Obj = Min\ (RMSE) \quad (17)$$

The Contrastive Divergence (CD) is developed for initializing the visible states in a non-random manner. The steps followed in CD are discussed below:

1. Onto visible neurons, clamp the samples.
2. Compute $Prob_{HID}$ by multiplying *Weight* and *VIS* as $Prob_{HID} = \sigma(VIS.Weight)$
3. For $\overrightarrow{VIS}$ and $\overrightarrow{Weight}$ compute the outer product and it is said to be the positive gradient $\emptyset^+ = VIS.Prob_{HID}^T$.
4. The reconstruction of the visible state VIS' from *HID* is sampled.
5. Compute Negative gradient $\emptyset^-$ by multiplying VIS' and HID' as $\emptyset^- = VIS'.HID'$.
6. The weight update $\Delta Weight = \delta(\emptyset^+ - \emptyset^-)$.
7. The new weight updating is as per Eq. (18).

$$Wt_{VIS,HID} = \Delta Wt_{VIS,HID} + Wt_{VIS,HID} \quad (18)$$

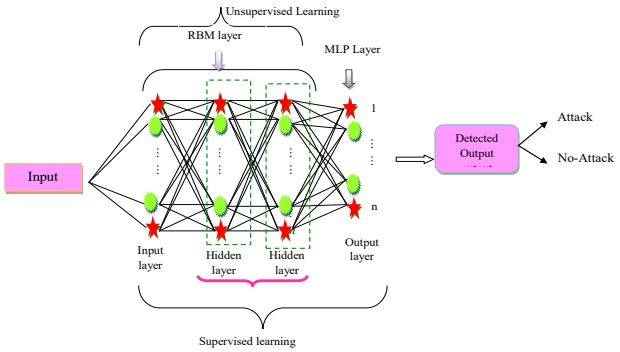


Fig. 2. Architecture of DBN based detection model

The presence or absence of the defects in the network is decided by DBN. When the presence of an attacker is identified in the network, the vulnerability assessment is undergone to predict the risk or severity level of the network. Figure 2 shows the architecture of the attack detection model in DBN.

5.2 Vulnerability Assessment Model

The correntropy-variation technique [46] is utilized for evaluating the similarities between the attacked samples denoted as *Attack* and the normal samples denoted as *Normal* detected from DBN. The correntropy V_σ of Attack and Normal features is computed as per Eq. (19). Here, σ is the kernel size and *E[.]* are the features of the expected values, and $K_\sigma(.)$ is the Gaussian kernel function. Mathematically, $K_\sigma(.)$ is expressed as per Eq. (20). In addition, the correntropy is computed as per Eq. (21).

$$V_\sigma(Attack, Normal) = E[K_\sigma(Attack - Normal)] \quad (19)$$

$$K_\sigma(.) = \frac{1}{\sqrt{2\pi\sigma}} exp^{-\frac{(.)}{2\sigma^2}} \quad (20)$$

$$\hat{V}_{M,\sigma}(A,B) = \frac{1}{M}\sum_{i,j=1}^{M}\left[K_\sigma\left(Attack_i - Normal_j\right)\right] \quad (21)$$

The absolute variation $\mu\,|Corpy^{normal} - Corpy^{test}|$ is normalized to acquire the risk level RL as in Eq. (22).

$$RL = \frac{\left|Corpy^{\frac{normal}{test}} - minCorpy^{normal}\right|}{maxCorpy^{normal} - minCorpy^{normal}} \quad (22)$$

Over the normal samples, the acquired minimum and maximum correntropy values are denoted as min $Corpy^{normal}$ and max $Corpy^{normal}$, respectively. If *RL>0.5*, then the anomalous samples are found to have higher risk level and so the control is transferred to attack mitigation approach. If *RL< 0.5*, then perform normal routing for data transfer. Figure 3 manifests the attack mitigation phase.

5.3 Bait model

The bait model is utilized in this research work for mitigating the attack nodes in system. The route discovery, attack node detection and mitigation, and route maintenance are the three major phases in the bait model.

Discovery phase: The route request (RREQ) packet is sent into the network by the source node *(S)*. If the neighbour of the source node has the target's routing information within its cache routing tablet, it then replies *S* with a RREP. During the forecasting of the route request (RREQ) packet, the next hop nodes record the route address information into the RREQ packet. Further, while the RREQ packet reaches the destination node *(Dest)*, then *Dest* identifies the address of the each of the neighbours amongst the route. The illustration for the route discovery phase is shown in Figure 4a. In this model, there are 8 counts of nodes, and here the source *(S)* node is the 1st node and the destination node *Dest* is the 7th node. When a data is available in *S* to be transmitted to *Dest*, the available paths are discovered between *S* and *Dest* with RREP and RREQ.

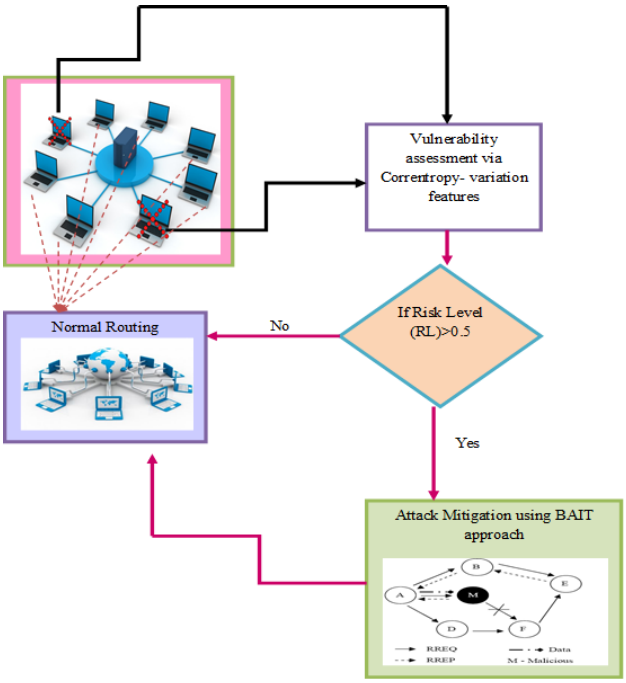
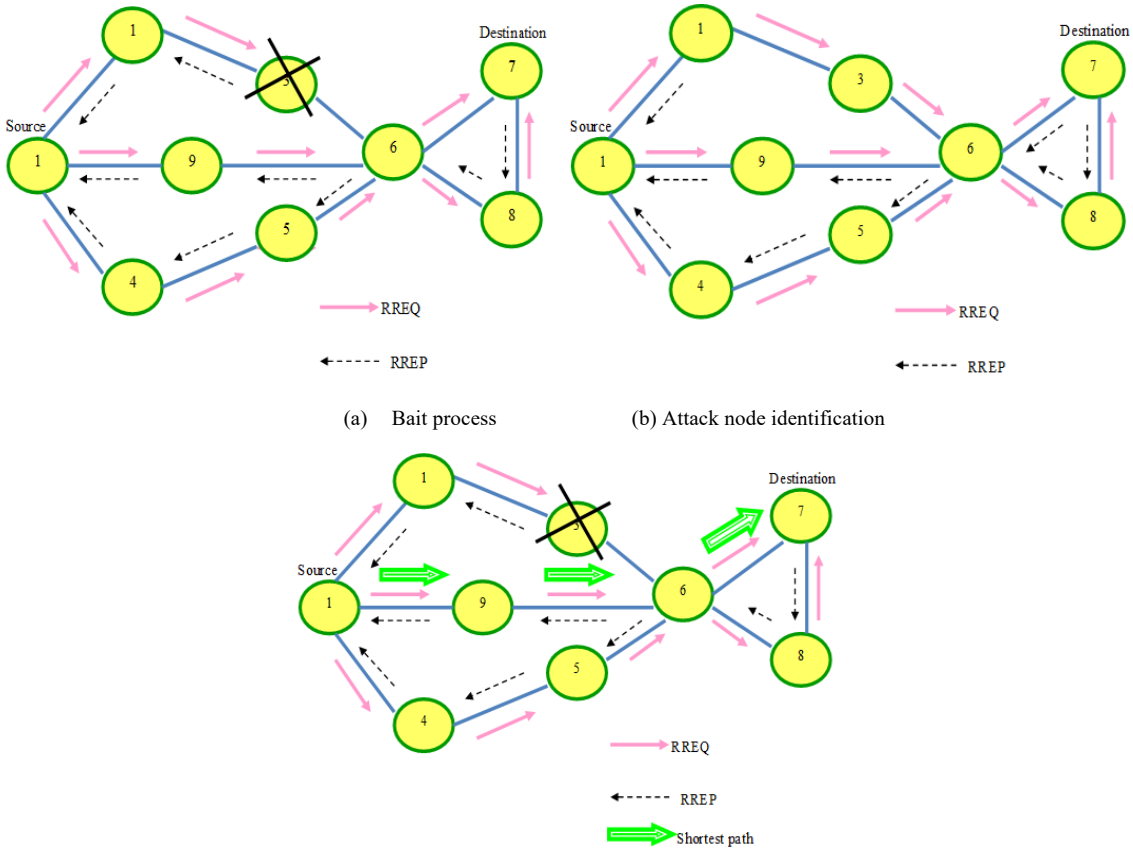


Fig. 3. Attack Mitigation Phase

Detection and mitigation of the attack node: In general, the node which is subjected to malicious, replies with false RREP that it is the most optimal node for transmitting and the data and claims itself as the shortest path to the destination. In this manner, it attracts all the data packets and discards them from reaching the target. Moreover, these attacker nodes instantly send the RREP packet despite having void routing table. This attacker node also sets the RREP packet with a maximum sequence number of the destination and unity hop count. However, during the reception of a new packet, there is an increment in the sequence numbers. Therefore, during the reception of a suspicious reply, it is essential to check the RREP possibility for the presence and absence of the malicious node.

In this bait model, when a RREP packet is received by an intermediate node, the value of the packet is parsed and the destination sequence number is verified to be maximal, and at the same time the hop count is verified to be the minimal value. If the value matches, then the received RREP packet is buffered and the data is transmitted to the destination. If the RREP is larger than original one, then the suspected node is identified to be malicious and it is discarded. This suspected node is then added into the malicious list and all other nodes in the network are alerted. The attacker node detection and mitigation in bait model is illustrated in Figure 4b. Let the suspected malicious node be node 3 and so it is discarded during the data transmission.



(a) Bait process (b) Attack node identification

(b) Route maintenance

Fig. 4. Bait model

Route maintenance: Once the malicious node is discarded and the network is said to be free from errors. Then, it is essential to choose the shortest path. This identification of the shortest path is accomplished by computing the throughput and packet delivery ratio. The data transmission via the shortest path is illustrated in Figure 4c. As per the example, the available path from to is 1-9-6-7, 1-9-6-8-7, 1-4-5-6-7 and 1-4-5-6-8-9. As per the computed throughput and packet delivery ratio, let the identified shortest path be1-9-6-7. The mathematical formula for computing the throughput Eq. (23).

Packet Loss Ratio: Fraction of the quantity of lost packets to the total quantity of sent packets.

Throughput: The quantity of data *(d)* that is transferred from one location to the other in a given amount of time is called throughput. Mathematically, throughput *Th* refers to the count of traffic is shown in Eq. (23), and here *size* denotes the size of the data packet and is the data rate. At the time of estimation, the penalty is maximum length and least penalty=0.

$$Th = \frac{Data^{rate} * size(1 - packet\ loss\ value\ (d))}{Penalty + time\ consumed(d)} \quad (23)$$

6. Experimental Results

6.1 Simulation process

This framework was executed in the MATLAB 2019a. The proposed model is tested with the data collected from UCI Machine Learning Repository. The analysis on feature selection is undergone for the proposed work with OCSA and the proposed work with Particle Swarm Optimization (PSO) [47], Genetic Algorithm (GA) [48], Whale Optimization Algorithm (WOA) [49], FireFly algorithm (FF) [50] and Rider Optimization Algorithm (ROA) [51], respectively. In addition, the classifier performance is evaluated for the proposed work with Deep Belief Network (DBN) [7], Neural Network (NN) [52], Support Vector Machine (SVM) [53] and Convolution Neural Network (CNN) [54], respectively. Evaluation was done in terms of positive and negative measures. Accuracy, specificity, precision and sensitivity are the positive measures and the negative measures like FNR, FOR, FDR and FPR need to be maintained as low as possible to prove that the proposed model is less prone to errors.

6.2 Comparison of performance analysis on feature selection with existing models

Performance analysis of proposed and existing models with respect to positive measures: The results acquired in terms of positive measures are represented in figure 5(a), 5(b), 5(c), 5(d). The accuracy being the key parameter is higher for the proposed work with OCSA for each learning percentage. On observing the 80th learning percentage, the proposed work with OCSA is 24.2%, 39.3% and 17.21%, 19.19% and 18% better than the proposed work with PSO, GA, WOA, FF and ROA models, respectively. From this single evaluation alone, it is vivid that the proposed work with OCSA makes the feature selection more optimal. In addition, the sensitivity of the proposed work with OCSA is higher than PSO and GA for each variation in the learning percentage. Apart from this, the specificity and precision of the proposed work with OCSA achieves the highest value (i.e., 100%), while the others with the proposed work records the specificity and precision values approximately with 92%. The precision of the proposed work with OCSA is 30%, 40%, 20%, 25% and 20% better than the better than the propose work with PSO, GA, WOA, FF and ROA models, respectively.

Performance analysis of proposed and existing models with respect to Negative measures: The results under negative measures like FNR, FOR, FDR and FPR by the proposed work with OCSA and the proposed work with PSO [47], GA [48], WOA [49], FF [50] and ROA [51], models are shown in figure 5(e), 5(f), 5(g), 5(h). The FDR of the proposed work with OCSA is the lowest value and it records the value approximately below 0.02. On observing the 70% of learning, the FDR of the proposed work with OCSA is 96.6%, 97.5%, 96%, 94.5% and 95% better than the proposed work with PSO, GA, WOA, FF and ROA models, respectively. On the other hand, the FNR of the proposed work is slightly higher than the proposed work with WOA, FF and ROA, respectively. Similar to this, the FOR of the proposed work with OCSA is lower than the proposed work with both the PSO and GA, respectively. Further, the FPR of the proposed work with OCSA is the lowest value and it is below the range of 0.01 for every variation. The FPR with OCSA is 97.1%, 98%, 96.6%, 96.8% and 96% better than the better than the proposed work with PSO, GA, WOA, FF and ROA models, respectively. Though the proposed work shows high FOR and FNR, the overall analysis proves the efficiency of the proposed work in detecting the attacks.

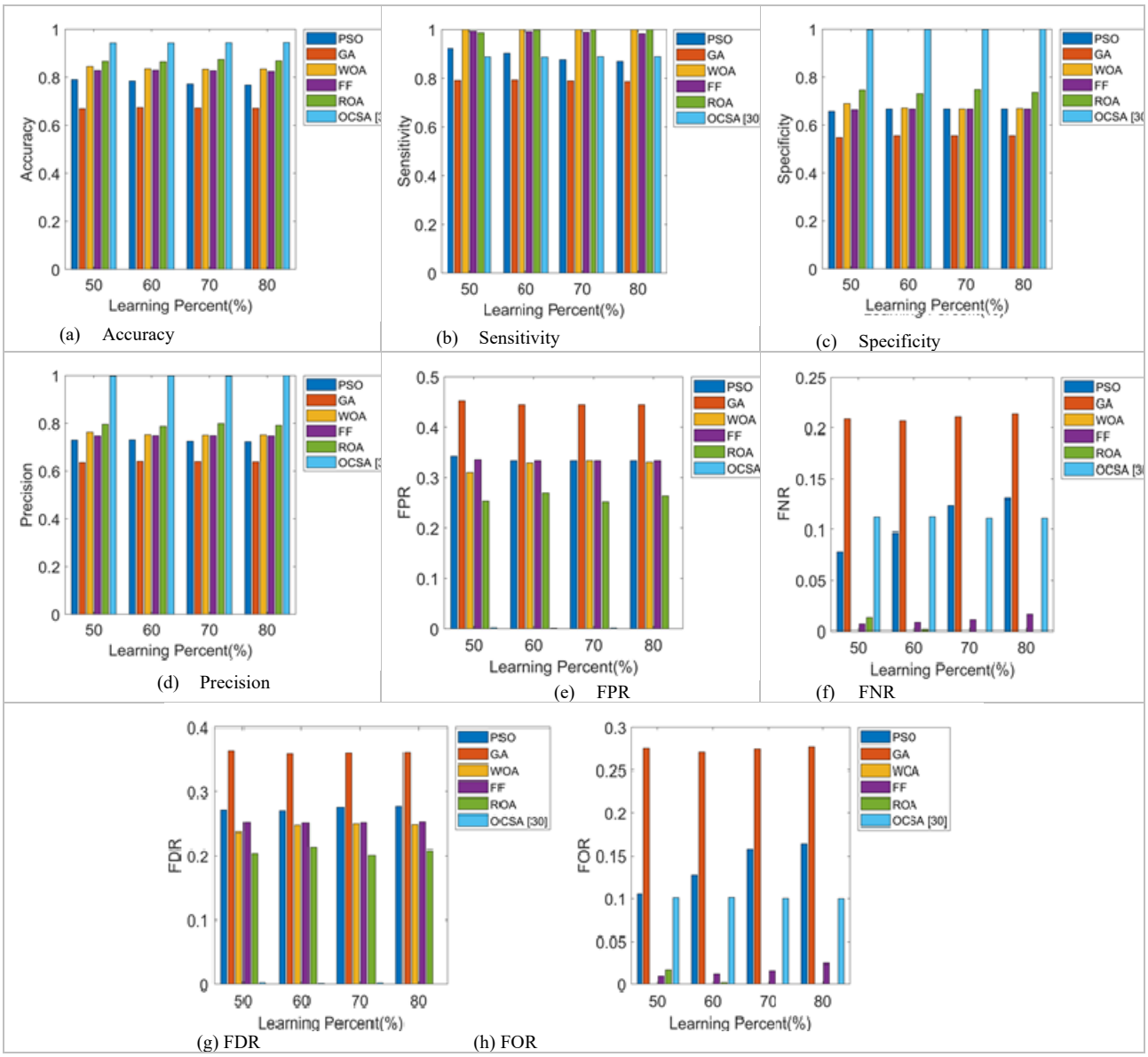


Fig. 5. Evaluation of the proposed and conventional algorithms

6.3 Performance analysis on Classifier Performance with existing classifiers: Attack Detection Phase

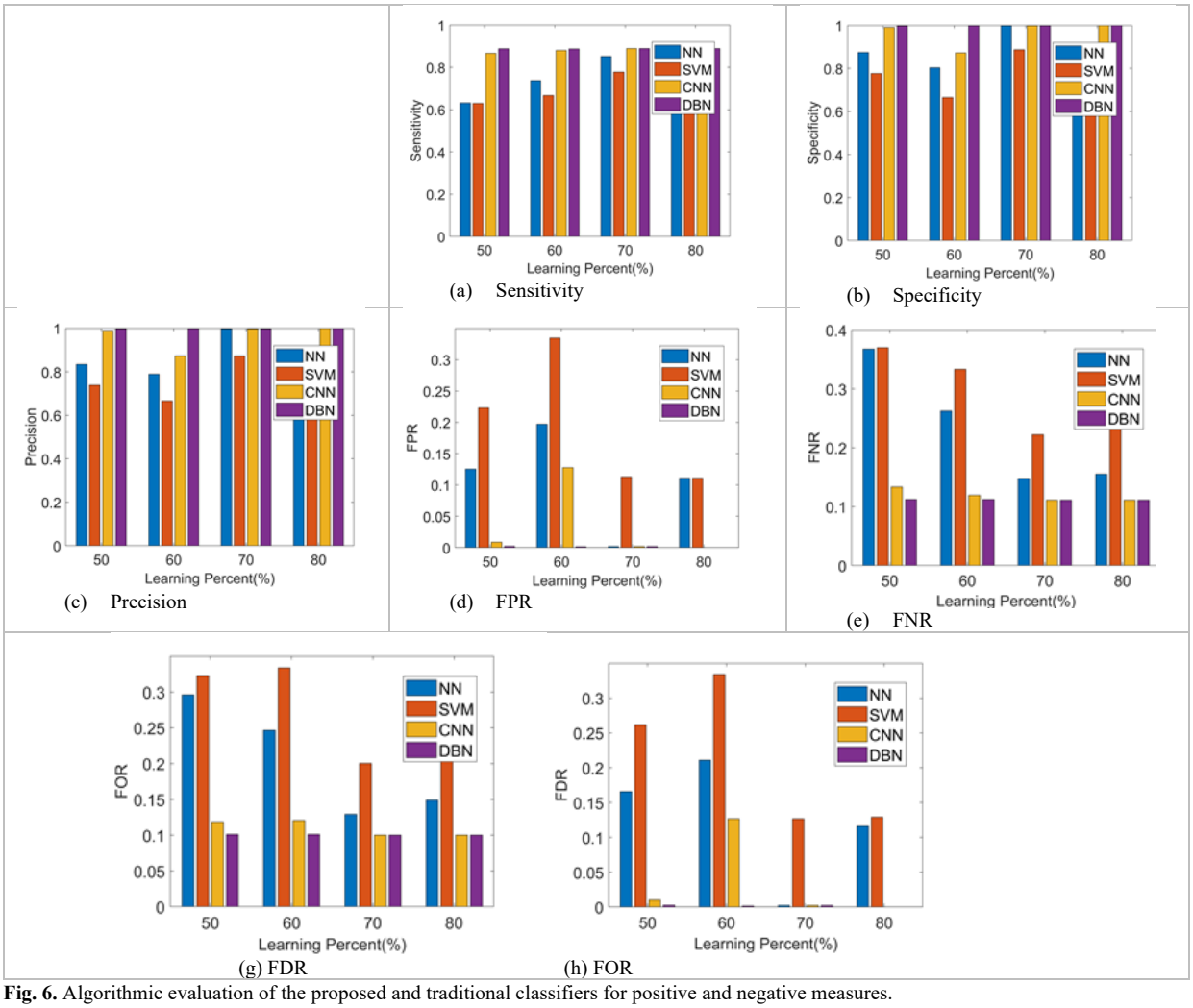
The positive and negative performance of this work with DBN for detecting the attacks is compared over the proposed work with NN [52], SVM [53] and CNN [54] classifiers. This evaluation is done in terms of both the positive and negative measures for different learning percentage like 50, 60, 70 and 80 respectively. The results acquired using the positive performance evaluation is shown in Figure 6(a), (b), (c), (d). The accuracy of the classifier shows how suitable is the selected classifier for detecting the attacks. The accuracy of the proposed algorithm with DBN records the highest value for all the learning percentage. The accuracy of the proposed work with DBN is 91.6%, 93.3% and 90.8% better than the proposed work with CNN, NN and SVM classifiers.

Further, the precision at learning percentage=60 of the proposed work with DBN is 20%, 35% and 12% better than the proposed work with NN, SVM and CNN classifiers, respectively. Similarly, FOR, FDR, FPR and FNR for the proposed work with DBN, SVM, CNN and NN are shown in figure 6(e), (f) (g), (h). On a quick glance, a clear-cut decision can be made that the proposed work with DBN based attack

detection records the lower values for all the error measures. On a quick glance, a clear-cut decision can be made that this work with DBN records the lower values for all the error measures The FNR of this work at 50th learning percentage is 75%, 71.4% and 30% better than the FNR of this work with NN, SVM and CNN classifiers, respectively. Further, the FDR and FNR of the proposed with DBN is the lowest value. Further, at learning percentage=60, the FPR of the proposed work with DBN is 90%, 95.4% and 66.6% on a quick glance, a clear-cut decision can be made that the proposed work with OCSA records the lower values for errors.

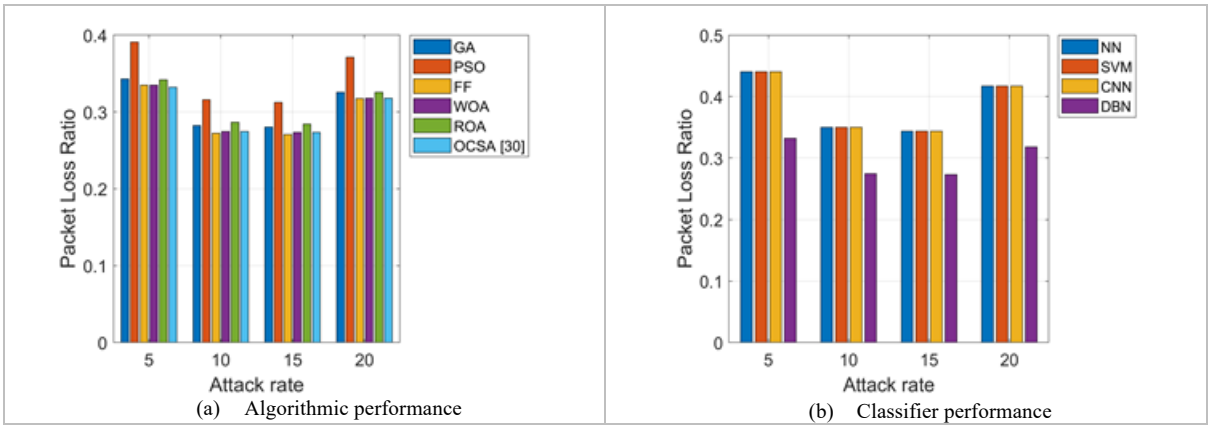
6.4 Performance analysis on Packet Loss Ratio of proposed and existing models forvarying attack rates

This section discusses about both the algorithmic and classifier-based evaluation of the proposed work in terms of the packet loss ratio. It is generally measured between source and destination. The packet loss ratio needs to be maintained with packet loss when one or more packet fails to reach its destination in a computer network. It is caused either by wireless network or network congestion. This evaluation is done by varying the rate of attacks from 5, 10, 15 and 20. The results acquired are shown in figure 7(a), 7(b). The lower is the packet loss ratio; the higher is the reliability of the system. The classifier performance of the proposed work with DBN is 33.3% and 28.5% and 29% better than the proposed work with DBN [7], NN [52], SVM [53] and CNN [54] classifiers, respectively.



(a) Sensitivity (b) Specificity (c) Precision (d) FPR (e) FNR (g) FDR (h) FOR

Fig. 6. Algorithmic evaluation of the proposed and traditional classifiers for positive and negative measures.

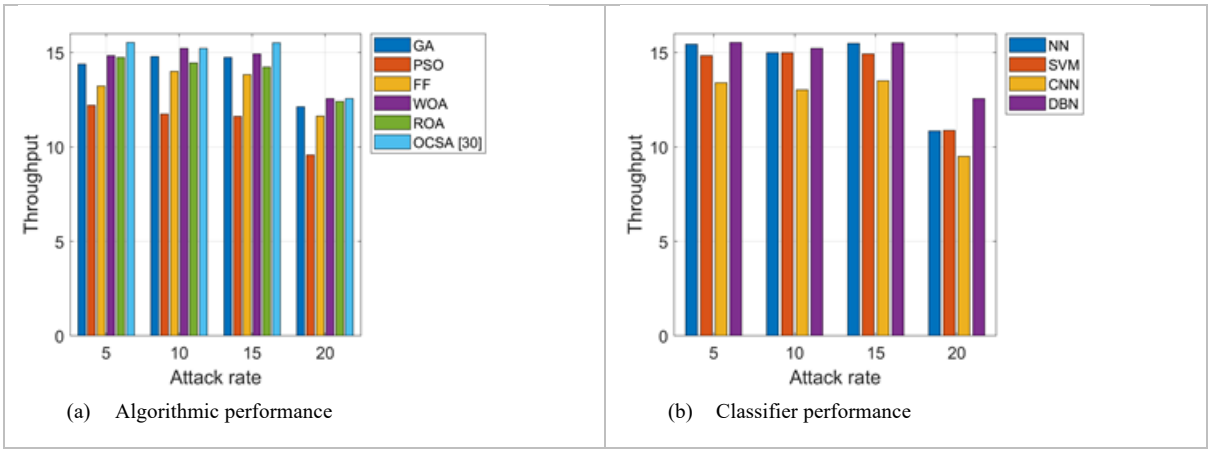


(a) Algorithmic performance (b) Classifier performance

Fig. 7. Evaluation on packet loss ratio of proposed and existing models on varying attack rate.

6.5 Performance analysis on Throughput of proposed and existing models for varying attack rates

In this section, the throughput analysis by both the optimization algorithms and classifiers are summarized. The results acquired by varying the attack rate are shown in figure 8(a), 8(b). The throughput with OCSA model is 79.7%, 60.25%, 79.7% and 41.4% better than the extant models like PSO [47], GA [48], WOA [49], FF [50], ROA [51] and AR-ROA. Hence results show that the proposed work achieves higher throughput in case of both the feature selection and classification.



(a) Algorithmic performance (b) Classifier performance

Fig. 8. Evaluation of throughput of proposed and existing models for varying attack rate.

7. Conclusion

In this paper, a framework for attack detection and mitigation was introduced. The attack detection was carried out using the deep learning model and based on the vulnerability in the network, the model switched to mitigation process. The features were extracted by OCSA initially and they were sent to attack detection process via DBN model, in which, the presence of the attacks was determined. Subsequently, vulnerability assessment was carried out by evaluating the risk level via correntropy variation features. This phase decides "how vulnerable the network is" with the presence of attack. This decision is based on fixing a threshold on risk level. Moreover, the decision tells whether to execute the mitigation process or not. In the attack mitigation phase, bait-based mitigation process was carried out. The proposed vulnerability-based attack detection and mitigation framework was evaluated in terms of throughput as well as packet loss ratio. The OCSA with DBN is better than the throughput of the work with NN, SVM and CNN, respectively. In the future work, we focus on working in a real time cloud environment, perform cross validation and cloud migration shall also be explored.

References

1. Reddy Saisindhutheja, Gopal K. Shyam, "A machine learning based attack detection and mitigation using a secure SaaS framework", In: Proceedings of the Journal of King Saud University - Computer and Information Sciences, Elsevier, https://doi.org/10.1016/j.jksuci.2020.10.005, 2020.
2. Ashok Kumar C, Vimala R, "Load Balancing in Cloud Environment Exploiting Hybridization of Chicken Swarm and Enhanced Raven Roosting Optimization Algorithm", In: Proceedings of the Multimedia Research, vol.3, no.1, pp.45-55, 2020.
3. Reddy Saisindhutheja, Gopal K. Shyam, "An Efficient Metaheuristic Algorithm Based Feature Selection and Recurrent Neural Network for DoS Attack Detection in Cloud Computing Environment", In: Proceedings of the Applied soft computing journal, Elsevier, vol.100, pp.1-11, 2020.
4. G. Somani, M. S. Gaur, D. Sanghi, M. Conti and M. Rajarajan, "Scale Inside-Out: Rapid Mitigation of Cloud DDoS Attacks", In: Proceedings of the IEEE Transactions on Dependable and Secure Computing, vol.15, no.6, pp.959-973, 2018.
5. O. Alkadi, N. Moustafa and B. Turnbull, "A Review of Intrusion Detection and Blockchain Applications in the Cloud: Approaches, Challenges and Solutions", In: Proceedings of the IEEE Access, vol.8, pp.104893- 104917, 2020.
6. S. Dong, K. Abbas and R. Jain, "A Survey on Distributed Denial of Service (DDoS) Attacks in SDN and Cloud Computing Environments", In: Proceedings of the IEEE Access, vol.7, pp.80813-80828, 2019.
7. Renjith Thomas and MJS. Rangachar, "Hybrid Optimization based DBN for Face Recognition using Low- Resolution Images", In: Proceedings of the Multimedia Research, vol.1, no.1, pp.33-43, 2018.
8. VhatkarKapilNetaji, Bhole G P, "Optimal Container Resource Allocation Using Hybrid SA-MFO Algorithm in Cloud Architecture", In: Proceedings of the Multimedia Research, vol.3, no.1, pp.11-20, 2020.
9. W. Wang, X. Du, D. Shan, R. Qin and N. Wang, "Cloud Intrusion Detection Method Based on Stacked Contractive Auto-Encoder and Support Vector Machine", In: Proceedings of the IEEE Transactions on Cloud Computing, doi: 10.1109/TCC.2020.3001017, 2020.
10. N. Ravi and S. M. Shalinie, "Learning-Driven Detection and Mitigation of DDoS Attack in IoT via SDN Cloud Architecture", In: Proceedings of the IEEE Internet of Things Journal, vol.7, no.4, pp.3559-3570, 2020.
11. Z. Tian, C. Luo, J. Qiu, X. Du and M. Guizani, "A Distributed Deep Learning System for Web Attack Detection on Edge Devices", In: Proceedings of the IEEE Transactions on Industrial Informatics, vol.16, no.3, pp.1963-1971, 2020.
12. H. Ma, H. Ding, Y. Yang, Z. Mi, J. Y. Yang and Z. Xiong, "Bayes-based ARP attack detection algorithm for cloud centers", In: Proceedings of the in Tsinghua Science and Technology, vol.21, no.1, pp.17-28, 2016.

13. T. V. Phan and M. Park, "Efficient Distributed Denial-of-Service Attack Defense in SDN-Based Cloud", In:Proceedings of the IEEE Access, vol.7, pp.18701-18714, 2019.
14. E. K. Lee, H. Viswanathan and D. Pompili, "Model-Based Thermal Anomaly Detection in Cloud Data centers Using Thermal Imaging", In: Proceedings of the IEEE Transactions on Cloud Computing, vol.6, no.2, pp.330-343, 2018.
15. N. Agrawal and S. Tapaswi, "Defense Mechanisms Against DDoS Attacks in a Cloud Computing Environment: State-of-the-Art and Research Challenges", In: Proceedings of the IEEE Communications Surveys & Tutorials, vol.21, no.4, pp.3769-3795, 2019.
16. C. Anglano, R. Gaeta and M. Grangetto, "Securing Coding-Based Cloud Storage Against Pollution Attacks", In: Proceedings of the IEEE Transactions on Parallel and Distributed Systems, vol.28, no.5, pp.1457- 1469, 2017.
17. P. Mishra, V. Varadharajan, E. S. Pilli and U. Tupakula, "VMGuard: A VMI-Based Security Architecture for Intrusion Detection in Cloud Environment", In: Proceedings of the IEEE Transactions on Cloud Computing, vol.8, no.3, pp.957-971, 2020.
18. Devagnanam J, Elango N M, "Optimal Resource Allocation of Cluster using Hybrid GreyWolf and Cuckoo Search Algorithm in Cloud Computing", In: Proceedings of the Journal of Networking and Communication Systems, vol.3, no.1, pp.31-40, 2020.
19. Brammya and T. Angelin Deepa, "Job Scheduling in Cloud Environment using Lion Algorithm", In: Proceedings of the Journal of Networking and Communication Systems, vol.2, no.1, pp.1-14, 2019.
20. O. AlKadi, N. Moustafa, B. Turnbull and K. R. Choo, "Mixture Localization-Based Outliers Models for securing Data Migration in Cloud Centers", In: Proceedings of the IEEE Access, vol.7, pp.114607-114618, 2019.
21. G. Li, S. X. Wu, S. Zhang and Q. Li, "Neural Networks-Aided Insider Attack Detection for the Average Consensus Algorithm", In: Proceedings of the IEEE Access, vol.8, pp.51871-51883, 2020.
22. O. Alkadi, N. Moustafa, B. Turnbull and K. R. Choo, "A Deep Blockchain Framework-enabled Collaborative Intrusion Detection for Protecting IoT and Cloud Networks", In: Proceedings of the IEEE Internet of Things Journal, doi: 10.1109/JIOT.2020.2996590, 2020.
23. W. Zhijun, LWenjing, L. Liang and Y. Meng, "Low-Rate DoS Attacks, Detection, Defense, and Challenges: A Survey", In: Proceedings of the IEEE Access, vol.8, pp.43920-43943, 2020.
24. Shahab Shamshirband, Mahdis Fathi Antonio Pescapè, "Computational intelligence intrusion detection techniques in mobile cloud computing environments: Review, taxonomy, and open research issues", In: Proceedings of the Journal of Information Security and Applications, https://doi.org/10.1016/j.jisa.2020.102582, 2020.
25. Karan B. Virupakshar, Manjunath Asundi, D. G. Narayan, "Distributed Denial of Service (DDoS) Attacks Detection System for OpenStack-based Private Cloud", In: Proceedings of the Procedia Computer Science, Elsevier, vol.167, pp.2297-2307, 2020.
26. José Tomás Martínez Garre, Manuel Gil Pérez, Antonio Ruiz-Martínez, "A novel Machine Learning-based approach for the detection of SSH botnet infection", In: Proceedings of the Future Generation Computer Systems, vol.115, pp-387-396, 2020.
27. Ninu Preetha NS, Brammya G, Ramya R, Praveena S, Binu D; Rajakumar B R, "Grey Wolf Optimisation based Feature Selection and Classification for Facial Emotion Recognition", In: Proceedings of the IET Biometrics, vol.7, no.5, pp.490-499, 2018.
28. B. R. Rajakumar, "Static and Adaptive Mutation Techniques for Genetic algorithm: A Systematic Comparative Analysis", In: Proceedings of the International Journal of Computational Science and Engineering, vol.8, no.2, pp.180-193, 2013.
29. A. Bhardwaj, V. Mangat and R. Vig, "Hyperband Tuned Deep Neural Network with Well Posed Stacked Sparse Auto Encoder for Detection of DDoS Attacks in Cloud", In: Proceedings of the IEEE Access, vol.8, pp.181916-181929, 2020.
30. A. Sahi, D. Lai, Y. Li and M. Diykh, "An Efficient DDoS TCP Flood Attack Detection and Prevention System in a Cloud Environment", In: Proceedings of the IEEE Access, vol.5, pp.6036-6048, 2017.
31. W. Wang, X. Du, D. Shan, R. Qin and N. Wang, "Cloud Intrusion Detection Method Based on Stacked Contractive Auto-Encoder and Support Vector Machine", In: Proceedings of the IEEE Transactions on Cloud Computing.
32. N. Ravi and S. M. Shalinie, "Learning-Driven Detection and Mitigation of DDoS Attack in IoT via SDN Cloud Architecture", In: Proceedings of the IEEE Internet of Things Journal, vol.7, no.4, pp.3559-3570, 2020.
33. Hari Krishna Pillutla, Amuthan Arjunan, "Fuzzy self organizing maps-based DDoS mitigation mechanism for software defined networking in cloud computing", In: Proceedings of the Journal of Ambient Intelligence and Humanized Computing, vol.10, pp.1547–1559, 2019.
34. Nitesh Bharot, Priyanka Verma, Sangeeta Sharma, Veenadhari Suraparaju, "Distributed Denial-of-Service Attack Detection and Mitigation Using Feature Selection and Intensive Care Request Processing Unit", In: Proceedings of the Arabian Journal for Science and Engineering, vol.43, pp.959–967, 2018.
35. Kriti Bhushan, Brij B. Gupta, "Network flow analysis for detection and mitigation of Fraudulent Resource Consumption (FRC) attacks in multimedia cloud computing", In: Proceedings of the Multimedia Tools and Applications, vol.78, pp.4267–4298, 2019.
36. Pillutla Harikrishna, A. Amuthan, "SDN-based DDoS Attack Mitigation Scheme using Convolution Recursively Enhanced Self Organizing Maps", vol.45, 2020.
37. H. Zhang et al., "Efficient and Secure Outsourcing Scheme for RSA Decryption in Internet of Things", In: Proceedings of the IEEE Internet of Things Journal, vol.7, no.8, pp.6868-6881, 2020.
38. Almoataz Y. Abdelaziz and Ahmed Fathy, "A novel approach based on crow search algorithm for optimal selection of conductor size in radial distribution networks", In: Proceedings of the Engineering Science and Technology, an International Journal, Elsevier, 2017, vol. 20, no. 2, pp. 391-402.
39. Hoda Zamani, Mohammad H. Nadimi-Shahraki, Amir H. Gandomi, "CCSA: Conscious Neighbourhood-based Crow Search Algorithm for Solving Global Optimization Problems", In: Proceedings of the Applied Soft Computing", Elsevier, 2019.
40. Rizk M. Rizk-Allah, Aboul Ella Hassanien, Siddhartha Bhattacharyya, "Chaotic Crow Search Algorithm for Fractional Optimization Problems", In: Proceedings of the Applied Soft Computing, Elsevier, vol. 71, pp. 1161-1175, 2018.
41. Gehad Ismail Sayed, Aboul Ella Hassanien, Ahmad Taher Azar, "Feature selection via a novel chaotic crow search algorithm", In: Proceedings of Neural Computing and Applications, Springer, vol. 31, pp. 171–188, 2019.
42. Hamid R. Tizhoosh, "Opposition-Based Learning: A New Scheme for Machine Intelligence", In: Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCAIAWTIC' 06), IEEE, Vienna, Austria, vol. 1, pp. 695-701, 2005.
43. Sedigheh Mahdavia, Shahryar Rahnamayana, Kalyanmoy Debb, "Opposition based learning: A literature review", In: Proceedings of the Swarm and Evolutionary Computation, Elsevier, vol. 39, pp. 1-23, 2018.
44. Ventresca, Mario, Shahryar Rahnamayan, Hamid R. Tizhoosh, "A note on Opposition versus randomness in soft computing techniques", In: Proceedings of the Applied Soft Computing, Elsevier, vol. 10, no. 3, pp. 956-957, 2010.
45. Rahnamayan, Shahryar, Hamid R. Tizhoosh, Magdy MA Salama, "Opposition versus randomness in soft computing techniques", In: Proceedings of the Applied Soft Computing, Elsevier, vol. 8, no. 2 pp. 906-918, 2008.
46. Nour Moustafa, Nour Moustafa and Kim-Kwang Raymond Choo, "An Ensemble Intrusion Detection Technique based on proposed Statistical Flow Features for Protecting Network Traffic of Internet of Things", In: Proceedings of the IEEE, vol.6, no.3, 2019.
47. M.R. Tanweer, S. Suresh, and N Sundararajan, "Self-regulating particle swarm optimization algorithm", In: Proceedings of the Information Sciences, vol.294, pp.182-202, 2015.
48. John McCall, "Genetic algorithms for modelling and optimisation", In: Proceedings of the Journal of Computational and Applied Mathematics, vol.184, no.1, pp.205-222, 2005.
49. Seyedali Mirjalili, Andrew Lewisa, "The Whale Optimization Algorithm", In: Proceedings of the Advances in Engineering Software, vol.95, pp.51-67, 2016.
50. IztokFister, IztokFisterJr, Xin-SheYang and JanezBrest, "A comprehensive review of firefly algorithms", In Proceedings of the Swarm and Evolutionary Computation, vol.13, pp.34-46, 2013.
51. Binu and Kariyappa, "Ride NN: A New Rider Optimization Algorithm-Based Neural Network for Fault Diagnosis in Analog Circuits", In: Proceedings of the IEEE Transactions On Instrumentation And Measurement, IEEE, pp.1-25, 2018.
52. Malige Gangappa, Kiran Mai C, Sammulal P, "Enhanced Crow Search Optimization Algorithm and Hybrid NN-CNN Classifiers for Classification of Land Cover Images", In: Proceedings of the Multimedia Research, vol.2, no.3, pp.12-22, 2019.

53. Renjith Thomas and Dr MJS. Rangachar, "Fractional Rider and Multi-Kernel-Based Spherical SVM for Low Resolution Face Recognition", In: Proceedings of the Multimedia Research, vol.2, no.2, pp.35-43, 2019.

54. Raviraj Vishwambhar Darekar, Ashwini Kumar Panjab Rao Dhande, "Emotion Recognition from Speech Signals Using DCNN with Hybrid GA-GWO Algorithm", In: Proceedings of the Multimedia Research, vol.2, no.4, pp.12-22, 2019.