International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
***Article Received**: 02 May 2023 **Revised**: 27 June 2023 **Accepted**: 29 July 2023*

Adversarial Sample Generation using the Euclidean Jacobian-based Saliency Map Attack (EJSMA) and Classification for IEEE 802.11 using the Deep Deterministic Policy Gradient (DDPG)

D. Sudaroli Vijayakumar [1], Sannasi Ganapathy[2]
[1]Centre for Cyber Physical Systems & School of Computer Science and Engineering
Vellore Institute of Technology
Chennai,India
0000-0001-8270-6223
e-mail: oli.sudar@gmail.com
[2]Centre for Cyber Physical Systems & School of Computer Science and Engineering
Vellore Institute of Technology
Chennai,India
0000-0001-9177-5378
e-mail: sganapathy@vit.ac.in

Abstract— One of today's most promising developments is wireless networking, as it enables people across the globe to stay connected. As the wireless networks' transmission medium is open, there are potential issues in safeguarding the privacy of the information. Though several security protocols exist in the literature for the preservation of information, most cases fail with a simple spoof attack. So, intrusion detection systems are vital in wireless networks as they help in the identification of harmful traffic. One of the challenges that exist in wireless intrusion detection systems (WIDS) is finding a balance between accuracy and false alarm rate. The purpose of this study is to provide a practical classification scheme for newer forms of attack. The AWID dataset is used in the experiment, which proposes a feature selection strategy using a combination of Elastic Net and recursive feature elimination. The best feature subset is obtained with 22 features, and a deep deterministic policy gradient learning algorithm is then used to classify attacks based on those features. Samples are generated using the Euclidean Jacobian-based Saliency Map Attack (EJSMA) to evaluate classification outcomes using adversarial samples. The meta-analysis reveals improved results in terms of feature production (22 features), classification accuracy (98.75% for testing samples and 85.24% for adversarial samples), and false alarm rates (0.35%).

Keywords- Wireless Intrusion Detection System (WIDS), Traffic Classification, AWID Dataset, Deep Reinforcement Learning (DRL), Feature reduction, Elastic Net, Recursive Feature Elimination (RFE), Euclidean Jacobian-based Saliency Map Attack (EJSMA).

I. INTRODUCTION

The number of devices in any network has exploded as we enter the era of gadget-based connections. In our home network, for example, we have smart TVs, smartphones, and other devices connected to a single Wi-Fi network. Any size network, from a little home network to a huge enterprise network, can use the wireless network to get connected. As more devices were added, the network's security became less secure. The IEEE 802.11 benchmark security technologies include WEP, TKIP, and LEAP. These conventional security measures are simply circumvented with a straightforward sniff attack. Several additional technologies, such firewalls, access control systems, and authentication techniques, are available to supplement and reinforce the security of wireless networks in light of the compromised IEEE 802.11 security standards. Nevertheless, effective monitoring and anticipating external threats are not yet fully implemented. Therefore, a second line of monitoring and detection systems is needed to track how devices are used in a network and spot unusual activity. The most common term for this level of security, which seeks to identify attacks on programs, logins, and confidential information, is intrusion detection system [1].

Like all other intrusion detection systems, the wireless intrusion detection system is further divided into host-based and network-based IDS [2]. The host-based IDS keeps track of each unique device connecting to the network, or host, and the kinds of verifications performed at this level cover things like record access, passwords, framework calls, application logs, and so on.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
***Article Received**: 02 May 2023 **Revised**: 27 June 2023 **Accepted**: 29 July 2023*

In contrast, network-based IDS gathers and examines data throughout the network at various points in order to identify malicious attacks. Regardless of the IDS type, it is necessary to evaluate the monitored data in order to determine its nature and whether it is normal or abnormal. The two main methods for categorizing the traffic are signature-based IDS [3] and anomaly-based IDS.The traditional database technique serves as the foundation of signature-based IDS [3]. Attack patterns from the past are tracked and stored in a database. The current traffic is compared with the database of assaults whenever the network encounters any kind of traffic to determine if it is an attack or not. If it does, the situation is referred to as an attack. Although this methodology produces acceptable results, the accuracy of threat detection only depends on an updated database. Regular updates are necessary for updated databases, which cannot be done automatically. This method's primary weakness is its incapacity to identify zero-day attacks and newer attack types.

Anomaly-based detection can more effectively address the problem faced by signature-based detection. In contrast to signatures, this method makes use of a baseline to know how the system functions properly. The current network activity is compared to this baseline value. It sets off alarms if it notices any deviations. The user attempting to enter into the network at erroneous times, the inclusion of additional devices, or even floods, are examples of deviations. As stated in the description, this is quite good at spotting anomalies; nonetheless, the rise in false positives is a major concern [4]. When contrasting the two detection methods, each has a unique combination of drawbacks and benefits. But these tactics work well together and are regularly employed in combination. The majority of intrusion detection systems on the market today use a mechanism that uses both techniques, each of which has advantages and disadvantages of its own. However, occasionally both techniques need human intervention, either to update the system or deal with false alarms. This necessitates the intrusion detection system having a system that can act like a person. The only technology that can make judgments and act in human-like ways is artificial intelligence, hence a quick analysis of current WIDS approaches utilizing these technologies is looked into.

A. State of the art limitations:

Support vector machines (SVM), k-nearest neighbor (KNN), artificial neural networks (ANN), decision trees, and other traditional machine learning algorithms have all been investigated for anomaly detection as a starting point for artificial intelligence usage in intrusion detection systems. However, the main drawback with the traditional algorithms is their inability to handle larger dimensions and unbalanced data. An ensemble classifier, as suggested in [11], solves the issue of huge dimensions and classification accuracy, but this is only confirmed for one kind of assault. Deep learning techniques address the issue of high-dimensional data in a manner similar to embedded classifiers. Some of the deep learning techniques utilized in the literature include deep neural networks [12], recurrent neural networks [13], and self-taught learning networks [14], and they can learn the input data at different granularities. The current intrusion detection system demands a key criterion in light of the methodologies: it must identify attacks even if the pattern changes with few false positives.

B. Approach and Contributions:

The goal of this work is to find a practical solution to the high-dimensional data problem with decent classification accuracy for a more recent batch of traffic. This is significant since the existing wireless network is intricate and different types of attacks are launched regularly. The approach used by the intrusion detection system should be able to integrate the newer attack types into the older attack data so that the system is aware of the newer attack types, leading to an improvement in both classification accuracy and how the wireless intrusion detection system should act in unclear situations without human intervention. Reinforcement learning, a fascinating topic that ensures the best results through trial and error, can be utilized to deal with uncertainty. Because reinforcement learning doesn't reach a conclusion until many trials have been completed, it can satisfy our need to manage network traffic in a completely dynamic setting. [15]. Although various machine, deep, and reinforcement learning techniques increase accuracy, they are unable to deal with adversarial attacks or the false-positive rate. In order to manage adversarial attacks, a deep reinforcement learning-based intrusion detection system is proposed in this study. Some of the work's major contributions include the following:

1. *Best feature subset selection:* The best subset of features resulting in the best classification accuracy are produced by a hybrid feature selection technique that combines the elastic net and recursive feature elimination.

2. *Creation of Deep Reinforcement IDS*: Multiple deep reinforcement learning agents are generated in this scenario, each of which can adapt to changing network traffic.

3. *Adversarial Attacks*: Our model's remarkable ability to deal with hostile attacks is one of its most promising results. For adversarial attacks, the classification accuracy is likewise quite good.

The following is how the rest of the work is organized: Section 2 discusses state-of-the-art systems. Section 3 delves into the technical specifics of the proposed intrusion detection system.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
***Article Received**: 02 May 2023 **Revised**: 27 June 2023 **Accepted**: 29 July 2023*

Section 4 presents the outcomes of the experiment, as well as a detailed analysis based on the changing assault patterns. Finally, section 5 brings the paper to a close.

II. RELATED WORKS

The use of machine learning and deep learning in intrusion datasets is covered in numerous academic works. This section presents the most typical of these applications to the AWID datasets. Additionally, we discuss the most significant studies that utilize reinforcement learning (RL) to categorize and detect intrusions.

A. Machine Learning and Intrusion Detection:

In the NSL-KDD cup data, different machine learning-based algorithms are used for intrusion detection systems. In terms of the AWID dataset, the original study [17] checks practically all machine learning algorithms, including Adaboost, J48, Naive Bayes, OneR, Random Forest, and others, and the comparative findings reveal that the J48 delivered a better accuracy score of 96 percent.

[18] presents traditional machine learning methods that produce greater output in terms of exactness, achieving 90% accuracy. Whatever the case may be, this results in a slew of false positives. [19] explains another crossbred construct that uses SVM and arbitrary woodland to coordinate misuse and irregularity detection. Clustering by K-means with a discretization approach On the ISCX dataset, [20] achieved 97.5 percent exactness, with a little increase in the number of false positives.

B. Reinforcement learning and intrusion detection:

Recently, the viability of applying reinforcement learning to various datasets has been examined. They used deep Q learning to develop a self-learning model, which they then put to the test on the NSL KDD Cup dataset [21]. For instance, discount rates and learning rates are hyperparameter variables that can be automatically changed. This system's properties enable auto-learning, which produces classification results that are more optimal. A similar self-taught learning approach using a sparse autoencoder is suggested in [22]. The work reported at [23] provided excellent prediction performance because the classifier developed here includes a policy function that enables the model to understand what is happening around it and update the model accordingly. These self-learning skills produced superior results even with recent attacks.

For multiple agents using reinforcement learning, a hierarchical structure is created using a lookup table [23, 24]. Despite the extensive findings for adversarial reinforcement learning, obtaining adequate results in intrusion detection is still in its early phases. Numerous works by [25-28] concentrate on a simulated network environment where a classifier's faults are the only focus. [30–34] suggests a deep learning-based method that might potentially develop new attacks on its own while taking opposing threats into account. Markov reward-based models are utilized to achieve the goal of extracting correct classification with maximum novelty.

Despite the fact that the state of the art demonstrates a variety of works based on integrating reinforcement learning inside the intrusion system for attack detection and classification, one of the significant results in all the work is its inability to handle false alarms efficiently. Although this study suggests a similar approach using ensemble classifiers and deep reinforcement methodology, it demonstrates that using ensemble classifiers to combine the best feature set with a deep reinforcement learning algorithm allowed us to achieve good predictive accuracy with fewer false alarms.

III. METHODOLOGY

Figure 1 shows the overall process used to create the deep reinforcement learning-based system. This section describes the overall procedure used to identify and categorize the attack.

A. Data Preprocessing:

If the raw data isn't adequately prepared, the model that will be utilized to perform the attack categorization won't perform well. A critical step was the translation of strings and characters into numbers and Boolean values. All categorical data is encoded using a single hot encoding technique. After these first adjustments, the emphasis is now on acquiring the necessary features. The first step in feature selection is getting the normalized value. We are taking L2 into consideration because it is thought to be steady. Distance is the most widely used normalization technique, but because to the distance dependency of the classifier, this may lead to issues with the development of our model. As a result, we employ a method in which we scale the data so that the outcome is 1 when squared and combined together. Since this parameter affects the data representation rather than the parameters, normalization is crucial. The equation (1) is used to get the L2 normalization value:

$$x = \sum_{i=1}^{n} y^2 \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \quad (1)$$

where y is the component after normalization and x is the L2 norm value which is one.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
***Article Received**: 02 May 2023 **Revised**: 27 June 2023 **Accepted**: 29 July 2023*

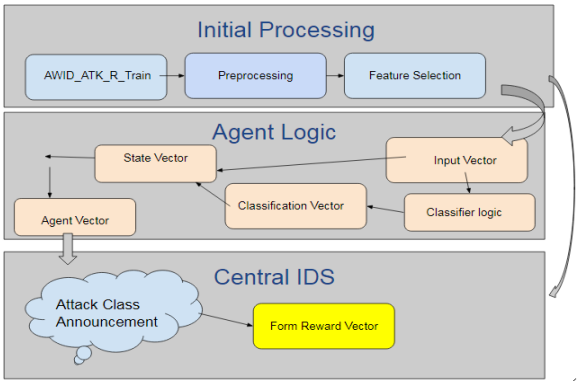

Figure 1: System Model of the Deep Reinforcement Learning Framework

B. Best Subset Selection Strategy:

The bigger set of input variables used in our model development reduces the predictive model's accuracy and uses up a lot of memory during training and development. Eliminating any factors that are unrelated to the target variable is therefore crucial. We have roughly 84 characteristics after the initial preprocessing stages, which is a significant number. A hybrid feature selection process is proposed integrating the embedded and wrapper methodology. This selection technique is used because the technique itself can evaluate the utility of the input variable. Recursive feature elimination removes unimportant features iteratively based on evaluation metric resulting with high variance and this is fine-tuned using the P value to identify the significance level of the considered features. Embedded approach in turn doesn't provide much variability and so the recursive feature elimination is embedded inside the Elastic Net. Figure 2 depicts the hybrid feature selections strategy and algorithm1 shows the step-by-step approach of the best subset selection strategy.

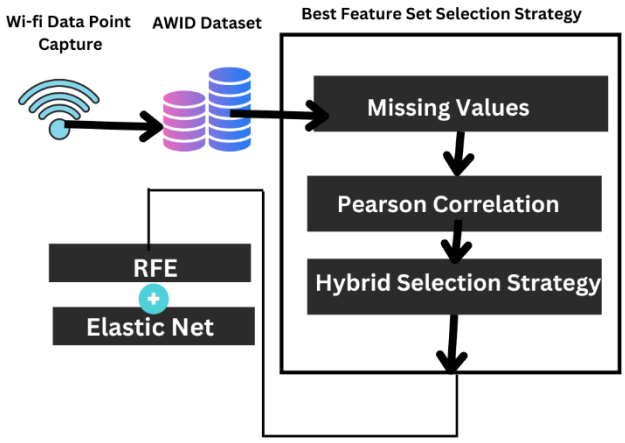

Figure 2: Best Feature Subset Selection Strategy

Algorithm 1: Feature Set Selection Strategy

Input: Set of all features N = {x_1, x_2, x_3, x_4 ..... x_{84}}
Output: Best features

do
step 1: for each feature in the feature set
step 2: Call Recursive Feature Elimination () to obtain subsamples
step 3: Recursive Feature Elimination RFE ()
step 4: set estimator value
do
step 5: for each feature in the feature set
step 6: find the feature with least absolute coefficient value
step 7: Eliminate and continue building with remaining feature set
step 8: Fit the RFE into Elastic Net
End for
End for
End

C. Deep Reinforcement Learning Model Description:

As was already mentioned, deep reinforcement learning serves as the foundation for the proposed intrusion detection system. The model is constructed following feature selection and preprocessing. Base service set networks (BSS) and extended service set networks (ESS) are the two types of network topologies that exist in our design because it is IEEE 802.11 compliant. A centralized association in which multiple clients attempt to join through a single access point is known as the infrastructure mode, also known as the basic service set mode. As soon as the relationship is established, the data is transmitted across a single channel. The wireless client alerts the access point (AP) of a device's desire to connect, and the AP either accepts or refuses the offer to join the network.

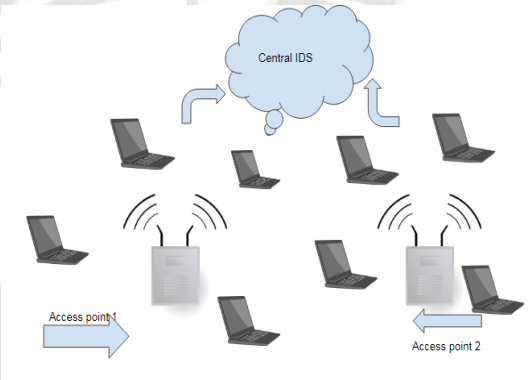

Figure 3: General Structure of the BSS Mode

Figure 3 depicts the whole configuration of the BSS network, which includes a central IDS. The association and security flaws are both simple, just as the association. In order to detect attacks, the central IDS employed in this case should behave somewhat intelligently. This calls for the use of reinforcement learning, which can pick up on its environment and modify its behavior in response to environmental changes. A Markov decision model, which consists of agents, environments, states, actions, and rewards, is in charge of choosing the best course of action to maximize the rewards. Let's examine how each of the elements is used in our imaginary IDS.Each BSS network has a

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
***Article Received**: 02 May 2023 **Revised**: 27 June 2023 **Accepted**: 29 July 2023*

single agent, which is set up at the access point. Feature selection and the learning algorithm are put into practice during the agents' preparation. This study used a value-based learning algorithm. With a few modifications, our agents employ the conventional Q learning technique. The agent tries to learn the policy whenever a change from one state to another is place by following the conventional Q learning process. At this moment, the access point agents are making an effort to record the result of this shift in the experience replay buffer. The pseudocode for the agent is given below:

for the State s, follow greedy(Q) to choose an action a

Agent takes action a, observe reward r and go to next state s'

Store(s,a,r,s',done)

The general DDPG policy has been modified, as indicated in [50,51], to deal with invasive behavior in a wireless network.

State:

By state, we mean all of the BSS network's participating nodes. A vector is commonly used to represent these nodes. The input and output attributes are included in these vectors. Thresholding is used to create the output class vector.

Action:

This action phase is critical since it entails making a decision. The agent's logic aids in the extraction of the action vector. The processes to determine whether the traffic is normal or under assault are usually included in the action vector. Following the methods below, the action vector *a* is produced from the state vector:

- The state vector is the input to the agent logic.
- The output is the final conclusion we make based on the Q values we acquired. The final output is compared to a threshold value based on the Q value acquired from each agent algorithm. The decision vector is made up of these compared values.
- The action vector is made up of the decision and categorization vectors.
- The decision and classification vectors are combined in the resultant action vector.

Reward:

The input created by the environment is referred to as "reward" based on the transitions made. A reward is given to the agent if the actual result and the categorization result are identical. Our intrusion detection system is powered by an agent algorithm that directs the agent's learning process, resulting in updated Q values. s, r, w, and are the parameters for state, reward, weights, and learning. Algorithm 2 explains the procedure for the agent learning. The central IDS is used to combine the data and determine whether or not an attack has occurred. The aggregate is carried out using a basic bagging technique.

So, the model's overflow begins with the agent receiving network traffic information, which is effectively provided by our dataset. Preprocessing and feature selection are applied to this data as a first step. The DDPG, which is our method that generates individual agent results, is fed the input feature vector acquired after feature selection. The bagging method is given the combined feature vector and agent vector, which determines if the traffic is normal or vulnerable. The central IDS transmits the final result to the individual agents by conversing with the environment. The core IDS plays a role in detecting the network's condition in the presence of malevolent behavior.

Algorithm 2: Agent Learning (DDPG)

Initialize memory and Replay Buffer
Initialize the target network Q
do
step 1: call feature selection ()
step 2: for each feature in the selected feature set
step 3: Classification_Vector←feature_vector≥ threshold
step 4: Confidence_Vector ← Input feature_vector to the classifier
step 5: State_Vector←feature_vector ∧Classification_Vector
DDPG(State_Vector)
Input: State _Vector
Output: Q Value decision_Vector
step 6: for episode =1, M
do
step 7: initialize a random process N for action exploration
step 8: receive initial observation state S_1
step 9: for t=1, T do
step 10: select action $a_t = \mu(s_t) + n_t$
step 11: Execute action at
step 12: Obtain reward r_t and next state S_{t+1}
step 13: R← (S_t, a_t, r_t, S_{t+1})
step 14: Sample a random minibatch of S transitions (S_t, a_t, r_t, S_{t+1}) from R
step 15: Action_Vector ← decision_Vector∧ Classification_Vector
step 16: Set $y_i = r_i + y_{Q'}(s_{i+1}, \mu'(s_{i+1}|\theta')|\theta'Q'$
step 17: Critic updation by minimizing the loss
step 18: L= $\frac{1}{N} P_i\left(y_i - Q(s_i, a_i|\theta Q)\right)^2$
step 19: Update the policy gradients
step 20: Update the target network
End for
End
End for
step 21: Initialize reward vector
step 22: Update the weights with new value then use the random samples
End

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
Article Received: 02 May 2023 Revised: 27 June 2023 Accepted: 29 July 2023

A proper architecture must be used to verify the complete model. Because wireless networks are so dynamic, a basic design is essential. The baseline here should work on a variety of contextual networks and with a variety of association techniques. With this in mind, we devised a simple baseline architecture to test our model, which consisted of a single base station and two related devices, as illustrated in Figure 4.

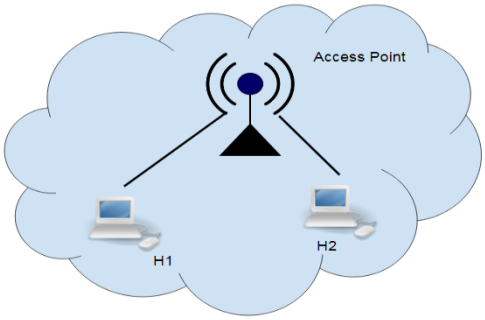

Figure 4: Deployment Architecture

There are two hosts in the simple baseline design, which means that traffic information is only exposed to one AP. So, in an ideal world, feature vector categorization happens at the access point, which then sends the results back to the central IDS. The central IDS will collect environmental data, which will be shared with the host, i.e., the agent, on a private basis. The base station computes vector rewards for H1 and H2 separately based on H1 confidence. Regardless of the number of base stations, this baseline design works on the same concept. The traffic information will be exposed to more than one AP if there are several base stations. Both the AP and the central IDS exchange the information in this instance, indicating that the feature vector is public. As a result, the feature set will be shared with both the AP host and the host reply reward vector will be determined based on the host response. This elucidates how computation occurs when the network topology is improved. As a result, this fundamental architecture is ideal for high-complexity topologies. This demonstrates that any complex topology can be simplified to this basic baseline architecture, and that each network can operate independently. Because each agent is self-contained, the outcomes of agents on different networks will differ. Our experimental investigation is completed entirely inside the confines of our basic baseline design. Algorithm 3 explains the attack classification logic that is applied on the baseline design.

Algorithm 3: Central Attack Classification Logic

Input: feature_vector, decision_vector
do
step 1: Group the agents based on context
step 2: For each topology
step 3: var1 = total number of set bits in each instance of agent_vector
step 4: var 2 = total number of unset bits in each instance of agent_vector
step 5: If var1 = var 2 then
step 6: Attack class = “Normal” Else
step 7: Attack class =” attack”
End
step 8: If (Attack class =” attack”) then
step 9: Feed the feature vector for attack classification
step 10: Attack type = classifier output Else
step 11: Attack type = Normal
End
step 12: Reward calculation for agent based on actual result
End for

D. Adversarial Sample Set:

Because it wasn't evaluated with a whole fresh set of assaults, most of the work done in intrusion detection systems employing datasets fails to function effectively when employed in real-world circumstances. As a result, it's critical to test the suggested system's stability when it encounters hostile samples. A piece of input data that has been slightly manipulated in order to induce a machine learning system to misclassify it is known as an adversarial example. In our system, we use EJSMA to generate adversarial samples. Euclidean Jacobian-based Saliency Map Attack (EJSMA) makes use of feature selection to reduce the number of features that need to be changed while still creating misclassification. Flat perturbations are applied to features in decreasing order iteratively based on their saliency value.

The distance is the metric used to generate adversarial samples, and they follow Lp norms. Every iteration, they seek to misclassify the target categorization, which is a greedy strategy. The gradient derivative $\nabla Z(x)t$ of the neural network causes the undesirable disturbance. The feature vector, which has been viewed as a function, is used to create the saliency map.The saliency map's mathematical formulation is as follows: The neural network is a multidimensional function $F: X \rightarrow Y$, with X representing the input feature vector and Y representing the output feature vector. The likelihood of the feature vector X mapping to the jth output class is denoted by Fj (X). The class with the highest probability is designated by the letter (X). To generate features that are similar to those in our test dataset, we use the Cleverhans module [42]. The assaults package provides the class Euclidean saliency map method. The random samples are generated via the saliency method in conjunction with the parameters that create the map matrix. Algorithm 4 explains the steps involved in producing the adversarial sample set.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
Article Received: 02 May 2023 Revised: 27 June 2023 Accepted: 29 July 2023

Algorithm 4: Adversarial Sample Generation

Input: Number of rows in test data in turn is the total number of samples

step 1: Empty array←[] // To store adversarial samples
step 2: Set the EJSMA parameters
step 3: Iterate on every sample till the range is met
step 4: Perform the derivative computation
step 5: If (derivative class value and if it is 0 target class =0)
Else
target class =1
end
step 6: call EJSMA generate.np (sample and EJSMA_parameter)
step 7: Sample Stack
End

The use of adversarial samples to test the proposed system is critical since we need to know how resistant it is. Although it demonstrates the model's robustness, it's also feasible that these samples will deceive our system. This needs a solution that ensures robustness even when feeding hostile samples. To develop such a model, a sparsity autoencoder is used to limit adversary sensitivity, which has an impact on the model's accuracy. An autoencoder is a type of neural network that can be used to learn unsupervised. Back Propagation is done in this case by assigning the target values to the input. The sparse neural arrangement tries to learn a close approximation to the identity so that the input and output have a direct map. This arrangement is trained on the adversarial dataset to predict the uncorrupted dataset.

The sparsity autoencoder is used in the following way in the system:

- The sparsity autoencoder receives the EJSMA adversarial sample as input.
- The sparsity autoencoder learns to create a compressed output that resembles the original sample exactly.

The L2 regularization parameter is used in the sparsity autoencoder, which has only one input and output layer. Both the input and output layers are made of 22 nodes There are 11 nodes in each of the three hidden layers. There are three nodes and eleven nodes, respectively. The first three are linked together. The layers serve as an encoder, allowing the input to be compressed. Similarly, the final three layers serve as a decoder, in the process of decompressing and producing the result.

IV. EXPERIMENTAL SETUP AND RESULT ANALYSIS

A. Dataset Description:

The exclusive IEEE 802.11 intrusion data set AWID is used to test our overall technique. We're focusing on a dataset that closely mimics a real wireless network because we're talking about guarding wireless networks. Because it is obtained using a well-defined wireless setup that includes terminals of many sorts of devices, the AWID dataset is the most well-known wireless intrusion dataset. The network is protected by the conventional WEP protocol, which allows many devices to connect. Because this is a universally accepted security standard for wireless networks, AWID is one of the benchmark datasets for doing various analysis. An AWID dataset is created by capturing real-time network traffic, therefore it contains information gathered from online surfing, flooding, and other sources. Because this data appears to be real traffic, the number of instances recorded using this Kali-based system is enormous, thus we'll focus on the reduced dataset that accounts for the imbalances between normal and attack observations. This contains 1,765,000 records, of which 1,62,358 are rivals of 17 different classes. The following is the course distribution plot of the AWID-ATK-R preliminary and test dataset:

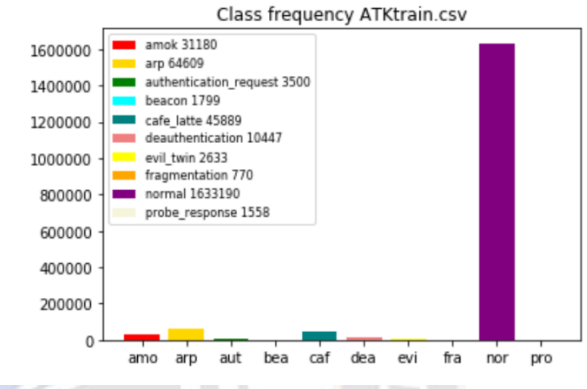

Figure 5: ATK_R_Train dataset Class Frequency Plot

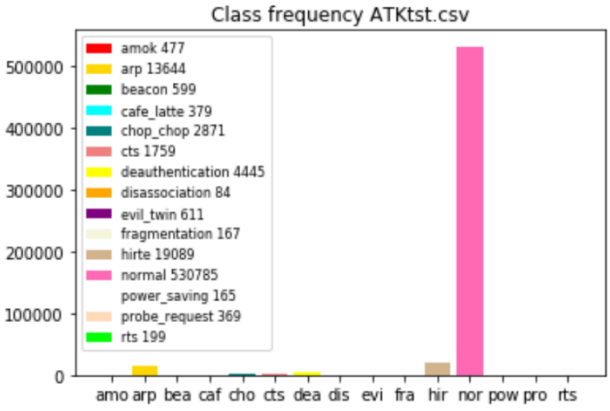

Figure 6: ATK_R_Test dataset Class Frequency Plot

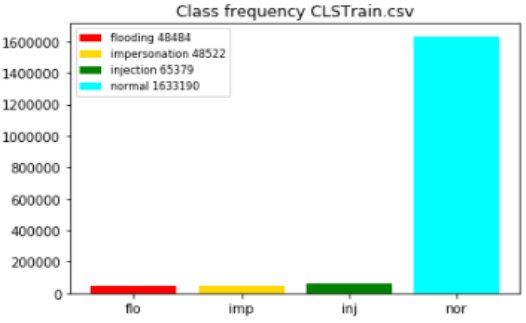

Figure 7: ATK_R_Training Classes of Attacks plot

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
***Article Received**: 02 May 2023 **Revised**: 27 June 2023 **Accepted**: 29 July 2023*

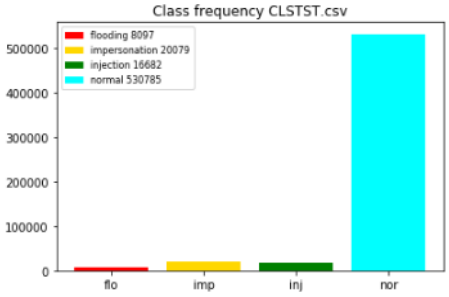

Figure 8: ATK_R_Test Classes of Attacks plot

In terms of size, the Aegean Wifi incursion dataset outperforms NSL-KDD, and the attack types are more current. The AWID ATK R, signifying the reduced form of the original AWID dataset, is represented by the class frequency plots in figures 5 to 8. This has a well-balanced training and testing dataset, as well as significant classification, which includes 17 attack classes. There are 154 features in this dataset, including categorical and continuous features. The number of features is significantly decreased to 20 classes after doing the standard preprocessing stages of missing, mean, and so on. Continuous features are scaled down to [0-10]. For categorical features, one hot encoding is used. Because the normal classes were higher than the anomalous classes, the dataset was unbalanced.

B. Results and Discussion:

This system was created entirely in Python and tested using the AWID wireless intrusion detection system benchmark data. The system is tested using the baseline design shown in Figure 7. The following are the usual machine learning measures that were utilised in the evaluation:

False positive rate (FPR) = FP/ FP + TN
True positive rate (TPR) = TP/TP + FN
Accuracy (ACC) = TP + TN/TP + TN + FP + FN

- TP (true positive): The number of samples that were anticipated to be positive turned out to be true.
- FP (false positives): The number of samples that were anticipated to be positive turned out to be negative.
- TN (true negatives): The number of samples that were anticipated to be negative turned out to be negative.
- FN (false negatives): The number of samples that were anticipated to be negative turned out to be positive.

A graphic having a TPR on the y-axis and an FPR on the x-axis is called a receiver operating characteristic (ROC). The likelihood that a classifier ranks a randomly chosen positive instance higher than a randomly chosen negative instance is equal to the area under the ROC curve (AUC). The ROC curve's area under the curve is projected to be larger.

C. Classifiers in Model:

On the AWID dataset, the performance of three distinct classifiers is: random forest (RF), Support Vector Machine (SVM), and AdaBoost (ADB). The scikit-learn library in Python is used to create all of the classifiers. The preprocessed feature chosen dataset is utilised to train each of the classifiers. Table 2 shows the results of the individual classifier performance.

Table 1: Performance Measures of the Classifiers

Random Forest				Support Vector Machine				AdaBoost			
P	Re	F1	Sup	P	Re	F1	Sup	P	Re	F1	Sup
0.86	0.87	0.86	85	0.00	0.00	0.00	85	0.00	0.00	0.00	85
1.00	1.00	1.00	134	0.00	0.00	0.00	134	0.00	0.00	0.00	134
0.99	1.00	1.00	71	0.00	0.00	0.00	71	0.00	0.00	0.00	71
1.00	0.99	0.98	565	0.00	0.00	0.00	565	0.00	0.00	0.00	565
1.00	1.00	0.99	386	0.00	0.00	0.00	386	0.00	0.00	0.00	386
0.96	1.00	1.00	931	0.00	0.00	0.00	931	0.00	0.00	0.00	931
0.98	1.00	1.00	25	0.00	0.00	0.00	25	0.00	0.00	0.00	25
0.90	0.99	0.99	129	0.00	0.00	0.00	129	0.00	0.00	0.00	129
1.00	0.90	0.90	30	0.01	0.83	0.01	30	0.01	0.83	0.01	30
1.00	1.00	1.00	3808	0.00	0.00	0.00	3808	0.00	0.00	0.00	3808
1.00	1.00	1.00	106089	0.95	0.99	0.97	106089	1.00	1.00	1.00	106089
1.00	1.00	1.00	36	0.00	0.00	0.00	36	1.00	1.00	1.00	36
1.00	1.00	1.00	77	0.00	0.00	0.00	77	0.99	1.00	0.99	77
1.00	1.00	1.00	42	0.00	0.00	0.00	42	1.00	1.00	1.00	42

From Table 1, we can see that the cross validated AdaBoost classifier gave higher accuracy compared to other classifiers. The comparative results of all the three classifiers are shown in figure 9.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
***Article Received**: 02 May 2023 **Revised**: 27 June 2023 **Accepted**: 29 July 2023*

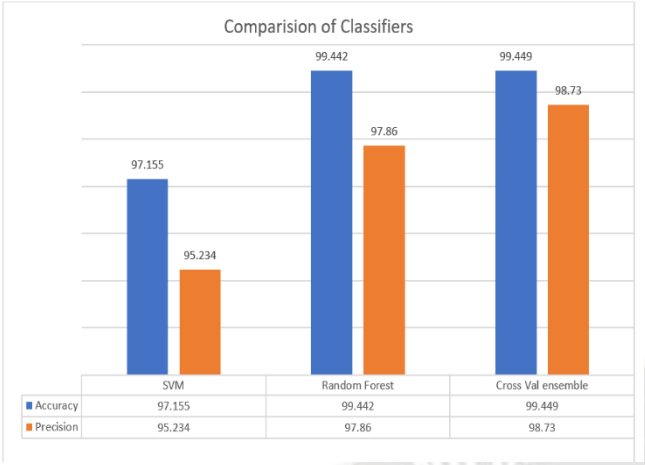

Figure 9: Comparison Analysis of the Classifiers

D. Evaluation of the proposed system with the classifiers:

As the Q values are modified, Table 1 shows how our system works with our three classifiers. The probability vector derived from deep networks is used for prediction. Any value above the Q threshold is considered an attack. There was no need to change the Q-values because all three classifiers considered here provided good accuracy. When compared to the SVM and cross-validated models, the random forest had the lowest FPR among the three classifiers. As a result, this combination was discovered to be the most effective in terms of achieving high accuracy while reducing false positives. For the AWID dataset, Table 2 displays the performance of our system when tested using combinations of three classifiers.

Threshold	Accuracy %	False Positive Rate(%)
0.3	92.50	4.1
0.4	93.80	3.8
0.5	94.80	3.9
0.6	96.81	1.2
0.7	98.75	0.35
0.8	95.48	0.30

Table 2: Proposed System Performance with varying Threshold

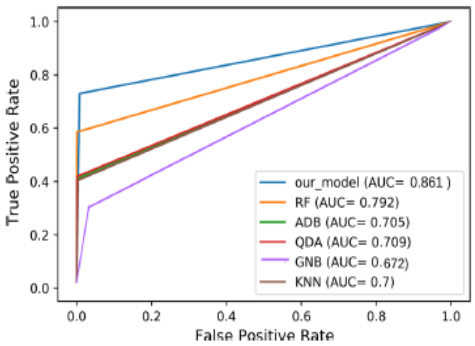

Figure 10: Comparative ROC of combined and individual classifiers

The results of the table and ROC analysis reveal that the suggested system outperforms the other systems when compared to random forest, adaboost, and KNN separately. While we performed the analysis for the individual classes of attacks, our combined classifier showed the best performance for the DoS attacks. The results are shown as a confusion matrix in Figure 11.

Type	Accuracy(%)	False positive rate(%)	AUC
Before Adversarial	98.75	0.35	0.9243
After Adversarial	85.24	2.6	0.85617
Sparsity	92.18	1.2	0.8010

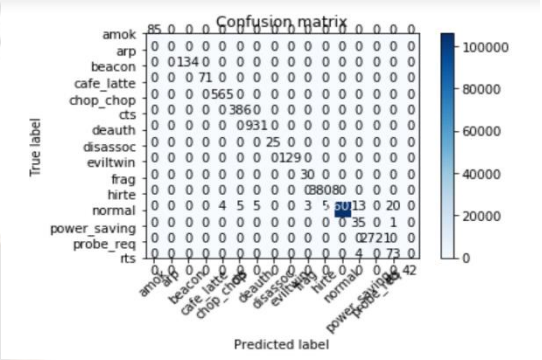

Figure 11: Overall performance of classifier for DoS Attack

The random forest performed admirably for dissociation and probe request assaults. Adaboost's maximum accuracy for impersonation and flooding assaults was cross-validated.

E. Adversarial Samples and Model Accuracy Evaluation:

When considering any machine learning-based intrusion detection system, one of the primary weaknesses that emerges is its inability to handle newer threats. By incorporating adversarial samples, we attempted to test our model. The system's performance is evaluated both before and after the

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
***Article Received**: 02 May 2023 **Revised**: 27 June 2023 **Accepted**: 29 July 2023*

hostile sample is introduced. The original testing and training were almost 1/3 percent upset. Figure 12 depicts the amount of perturbation after using EJSMA. The original feature values varied by 0.3 percent, as shown in the diagram.

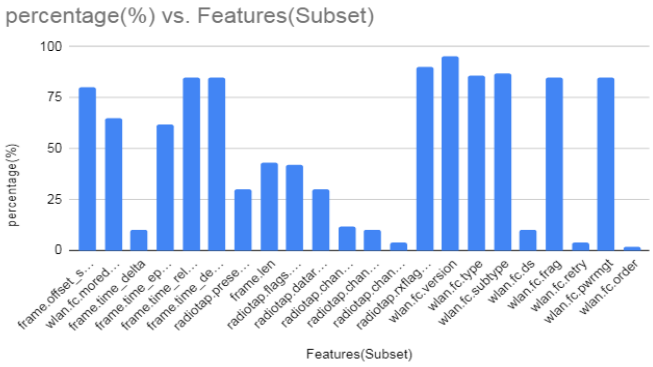

Figure:12 Perturbation percentage after applying EJSMA and Autoencoder

The feature vector was supplied to the agent after the perturbation, which observed the variation in terms of accuracy and false positive rate. The values before and after the adversarial samples are shown in Table 3.

Table 3: System performance under Adversarial Samples

Method	Dataset	Feature Selection	Classification	No.of features	Accuracy(%)	FAR(%)
Parker et.,al[43]	AWID-CLS-R	Autoencoder,MI	RBFC	7	98	2
Ran et.,al[44]	AWID-CLS-R	Not Available	Ladder Network	95	99.28	0.23
Alotaibi et., al[45]	AWID-CLS-R	IG	Random Tree	41	95.12	0.538
Thanthrige et.,al[46]	AWID-CLS-R	Not Available	Voting (ET, Fragging)	20	96.32	N/A
Vaca et.,al[47]	AWID-CLS-R	CFS	RF	18	99.096	0.248
Mikhail et.,al[48]	AWID-CLS-R	Not available	SBN	155	95.26	3.48
Proposed	AWID-CLS-R	Our Model	Deep Reinforcement Learning	22	98.75 85.24 92.18	0.35 2.6 1.2

Without any adversarial examples, the suggested model performed quite well, as seen in table 4. The model's accuracy drops to 81.80% after the adversarial samples are added, down from 98.75 percent following EJSMA. The introduction of the defensive model, which in our instance is the sparsity encoder, effectively handles the loss of accuracy. Figure 13 displays the ROC comparison, which shows that the addition of the defensive model leads to an increase in accuracy. All of the classifiers were tested again with and without defensive models, and we can see that the accuracy drops with adversarial data, whereas the sparsity encoder shows a progressive increase.

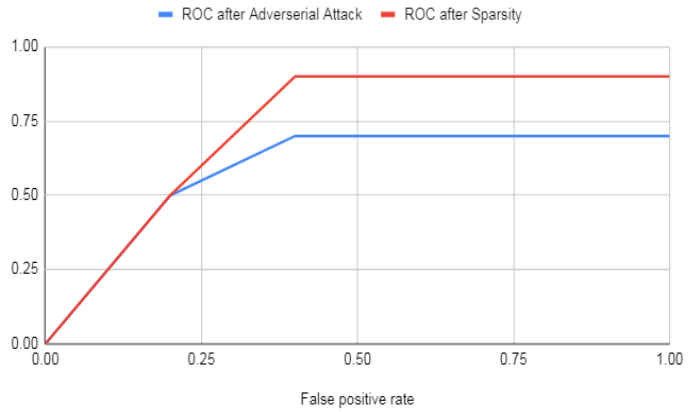

Figure 13: ROC of the proposed system after adversarial and with Sparsity

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
***Article Received**: 02 May 2023 **Revised**: 27 June 2023 **Accepted**: 29 July 2023*

F. Comparison with state-of-the-art methods:

Some state-of-the-art studies employing the AWID dataset are compared to this adversarial sample model to better interpret the reinforcement-based classifier for the wireless intrusion dataset. In terms of accuracy and false alarm rate, the comparison is conducted based on feature selection and classifier. The comparative results are shown in table 4.

The usage of adversarial samples in our system and the testing of the effectiveness of the suggested model following the introduction of adversarial samples are two observations made in regard to the state of the art. Our system will be more reliable as a result of the examination since it will be able to withstand new forms of attacks. The majority of current approaches fail to achieve robustness when dealing with adversarial samples.

Several machine learning and ensemble methodologies for improving prediction accuracy have been proposed in the literature. The KDD Cup dataset is used for the majority of the work, while the AWID dataset is used for some of the work in Table 4. The majority of the methods, however, do not employ deep reinforcement-based learning. Deep reinforcement learning is used to build a reinforcement-based model in [49]. However, our model differs in terms of how adversarial samples are created and evaluated, as well as how classifiers are used. Elasticnet with recursive feature eliminated ensemble classifiers combined with reinforcement learning achieved a good mix of accuracy and false positive rate.The deployment architecture is not only basic, but it also serves as a foundation that can be expanded to a more complicated architecture and used in various scenarios. The use of a sparsity autoencoder also ensures accuracy and false positive rate stability, even when hostile samples are included. This clearly shows that our model evaluation determined that this model is a feasible scenario in the majority of wireless networks. Any reinforcement-based learning model's success is determined by how it reacts when it comes into direct contact with the environment. Because the performance of the adversarial and defensive techniques was superior, it's likely that if evaluated in a real network, this model will perform better in real-world conditions. The experimental evaluation of our system comes to a close with that idea.

V. CONCLUSION

Major problems faced in today's intrusion detection system are the high number of false alarms and the inability to handle the newer types of attacks. This work thus aims to address both problems for wireless networks. With the assistance of AWID traces of data, a wireless intrusion detection system is built using reinforcement learning. The methodology proposed using the deep deterministic policy gradient provided a better classification result in terms of accuracy, and the number of false alarms generated was also considerably reduced compared to the state of the art. With the sparsity encoder, a methodology to generate and handle the adversarial attack is verified successfully, and the accuracy rate and false alarm rate are better than state-of-the-art. However, this accuracy can be improved further. Our future endeavour will be to develop a methodology that can produce better classification results using evolutionary algorithms.

REFERENCES

[1] Mohamed, A.B., Idris, N.B., Shanmugum, B.: A brief introduction to intrusion detection systems. In: International Conference on Intelligent Robotics, Automation, andManufacturing, pp. 263–271. Springer, Berlin (2012). https://doi.org/10.1007/978-3-642-35197-6_29.

[2] Ranjit Panigrahi, Samarjeet Borah, Moumita Pramanik, Akash Kumar Bhoi, Paolo Barsocchi, Soumya Ranjan Nayak, Waleed Alnumay. Intrusion detection in cyber–physical environment using hybrid Naïve Bayes—Decision table and multi-objective evolutionary feature selection, Computer Communications,Volume 188,2022, Pages 133-144, ISSN 0140-3664, https://doi.org/10.1016/j.comcom.2022.03.009.

[3] Buczak, A.L., Guven, E.: A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Commun. Surv. Tutor. 18(2), 1153–1176 (2016). https://doi.org/10.1109/COMST.2015.2494502

[4] Singh, K.P., Kesswani, N. An Anomaly-Based Intrusion Detection System for IoT Networks Using Trust Factor. SN COMPUT. SCI. 3, 168 (2022). https://doi.org/10.1007/s42979-022-01053-9

[5] Kuang, F.,Xu,W., Zhang, S.:Anovel hybrid KPCA and SVM with GAmodel for intrusion detection. Appl. Soft Comput. 18, 178–184 (2014)

[6] Reddy, R.R., Ramadevi, Y., Sunitha, K.V.N.: Effective discriminant function for intrusion detection using SVM. In: Proceedings of International Conference on Advance in Computing, Communication and Information (ICACCI), pp. 1148–1153 (2016)

[7] Li, W., Yi, P., Wu, Y., Pan, L., Li, J.: A new intrusion detection system based on KNN classification algorithm in wireless sensor network. J. Electron. Comput. Eng. 2014, 240217 (2014)

[8] Bivens, A., Palagiri, C., Smith, R., Szymanski, B., Embrechts, M.: Network-based intrusion detection using neural networks. Intell. Eng. Syst. Artif. Neural Netw. 12(1), 579–584 (2002)

[9] Quinlan, R.: Induction of decision trees.Mach. Learn. 1(1), 81–106(1986)

[10] Ross Quinlan, J.: C4.5: Programs for Machine Learning. Morgan Kaufmann, Burlington (1993)

[11] Vijayakumar, D.S., Ganapathy, S. Multistage Ensembled Classifier for Wireless Intrusion Detection System. Wireless Pers Commun 122, 645–668 (2022). https://doi.org/10.1007/s11277-021-08917-y

[12] Javaid,A., Niyaz, Q., Sun,W.,Alam, M.:Adeep learning approach for network intrusion detection system. In: Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (BICT), pp. 21–26 (2015)

[13] Yin, C., Zhu, Y., Fei, J., He, X.: A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access 5, 21954–21961 (2017). https://doi.org/10.1109/ACCESS.2017.2762418

[14] Mnih, V., et al.: Human-level control through deep reinforcement learning. Nature 518, 529–533 (2015). https://doi.org/10.1038/nature14236

[15] Mahmud, M., Kaiser, M.S., Hussain, A., Vassanelli, S.: Applications of deep learning and reinforcement learning to biological data. IEEE Trans. Neural Netw. Learn. Syst. 29(6), 2063–2079 (2018). https://doi.org/10.1109/TNNLS.2018.2790388

[16] Kober, J., Bagnell, J.A., Peters, J.: Reinforcement learning in robotics: a survey. Int. J. Robot. Res. 32(11), 1238–1274 (2013).https://doi.org/10.1177/0278364913495721

[17] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset," IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 184-208, 2016.

[18] Kamble, V. S. ., Khampariya, P. ., & Kalage, A. A. . (2023). A Survey on the Development of Real-Time Overcurrent Relay Coordination Using an Optimization Algorithm. International Journal of Intelligent Systems and Applications in Engineering, 11(3s), 104–114. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/2537

[19] Yin, C.; Zhu, Y.; Fei, J.; He, X. A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. IEEE Access 2017, 5, 21954–21961. doi: 10.1109/ACCESS.2017.2762418

[20] Kim, G.; Lee, S.; Kim, S. A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. Expert Syst. Appl. 2014, 41, 1690–1700. doi: 10.1016/j.eswa.2013.08.066

[21] Tahir, H.M.; Said, A.M.; Osman, N.H.; Zakaria, N.H.; Sabri, P.N.A.M.; Katuk, N. Oving K-means clustering using discretization technique in network intrusion detection system. In Proceedings of the 3rd International Conference on Computer and Information Sciences (ICCOINS), Kuala Lumpur, Malaysia, 15–17 August 2016; pp. 248–2 doi: 10.1109/ICCOINS.2016.7783222

[22] Alavizadeh, H.; Alavizadeh, H.; Jang-Jaccard, J. Deep Q-Learning Based Reinforcement Learning Approach for Network Intrusion Detection. Computers 2022, 11, 41. https://doi.org/10.3390/computers11030041

[23] Javaid,A., Niyaz, Q., Sun,W.,Alam, M.:Adeep learning approach for network intrusion detection system. In: Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (BICT), pp. 21–26 (2015)

[24] Guillermo Caminero, Manuel Lopez-Martin, Belen Carro, Adversarial environment reinforcement learning algorithm for intrusion detection, Computer Networks, Volume 159, 2019,Pages 96-109, ISSN 1389-1286, https://doi.org/10.1016/j.comnet.2019.05.013.

[25] Roderick,M., MacGlashan, J.: Implementing the deep Q-network. Stefanie Tellex, Humans To Robots Laboratory, Brown University, Providence, RI 02912, CoRR (2017)

[26] Mark White, Kevin Hall, Ana Silva, Ana Rodriguez, Laura López. Predicting Educational Outcomes using Social Network Analysis and Machine Learning . Kuwait Journal of Machine Learning, 2(2). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/182

[27] RotaBulò, S., Biggio, B., Pillai, I., Pelillo,M., Roli, F.: Randomized prediction games for adversarial machine learning. IEEE Trans. Neural Netw. Learn. Syst. 28(11), 2466–2478 (2017). https://doi. org/10.1109/TNNLS.2016.2593488

[28] Biggio, B., Fumera, G., Roli, F.: Security evaluation of pattern classifiers under attack. IEEE Trans. Knowl. Data Eng. 26(4), 984–996 (2014). https://doi.org/10.1109/TKDE.2013.57

[29] Biggio, B., et al.: Security evaluation of support vector machines in adversarial environments. In: Ma,Y., Guo, G. (eds.) SupportVector Machines Applications, pp. 105–153. Springer, Cham (2014)

[30] Papernot, N., McDaniel, P.,Wux, X., Jhax, S., Swamiz, A.: Distillation as a defense to adversarial perturbations against deep neural networks. CoRR (2016). https://doi.org/10.1109/SP.2016.41.

[31] Pattanaik, A., Tang, Z., Liu, S., Bommannan, G., Chowdhary, G.: Robust Deep Reinforcement Learning with Adversarial Attacks. University of Illinois at Urbana-Champaign, CoRR (2017)

[32] Wang, Z.: Deep learning-based intrusion detection with adversaries. IEEE Access 6, 38367–38384 (2018)

[33] Blowers, M., Williams, J.: Machine learning applied to cyber operations. In: Network Science and Cybersecurity, pp. 55–175. Springer, New York (2014)

[34] Farnaaz, N., Jabbar, M.A.: Random forest modelling for network intrusion detection system. Procedia Comput. Sci. 89, 213–217 (2016)

[35] Pallathadka, D. H. . (2021). Mining Restaurant Data to Assess Contributions and Margins Data . International Journal of New Practices in Management and Engineering, 10(03), 06–11. https://doi.org/10.17762/ijnpme.v10i03.128

[36] Tajbakhsh, A., Rahmati,M., Mirzaei, A.: Intrusion detection using fuzzy association rules. Appl. Soft Comput. 9, 462–469 (2009)

[37] Polikar, R.: Ensemble based systems in decision making. IEEE Circuits Syst. Mag. 6(3), 21–45 (2006). https://doi.org/10.1109/MCAS.2006.1688199

[38] Gharibian, F., Ghorbani, A.: Comparative study of supervised machine learning techniques for intrusion detection. In: Fifth Annual Conference on Communication Networks and Services Research (CNSR'07), pp. 350–358 (2007)

[39] Zhang, J., Zulkernine, M., Haque, A.: Random-forests-based network intrusion detection systems. IEEE Trans. Syst. Man Cybern.C Appl. Rev. 38(5), 649–659 (2008). https://doi.org/10.1109/ TSMCC.2008.923876

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v11i8.7946
***Article Received**: 02 May 2023 **Revised**: 27 June 2023 **Accepted**: 29 July 2023*

[40] Mukkamalla, S., Sung, A.H., Abhraham, A.: Intrusion detection using an ensemble of intelligent paradigms. J. Netw.Comput.Appl. 28, 167–182 (2005)

[41] Shone, N., Ngoc, T.N., Phai, V.D., Shi, Q.: A deep learning approach to network intrusion detection. IEEE Trans. Emerg. Topics Comput. Intell. 2(1), 41–50 (2018). https://doi.org/10.1109/ TETCI.2017.277279210.1109/TETCI.2017.2772792

[42] A. Servin, Multi-Agent Reinforcement Learning for Intrusion Detection. Ph.D. thesis, University of York, 2009.

[43] K. Malialis, Distributed Reinforcement Learning for Network Intrusion Response. Ph.D. thesis, University of York, 2014

[44] Kumar, N., Swain, S.N., Siva Ram Murthy, C.: A novel distributed Q-learning based resource reservation framework for facilitating D2D content access requests in LTE-A networks. IEEE Trans. Netw. Serv. Manag. 15(2), 718–731 (2018). https://doi.org/10.1109/TNSM.2018.2807594

[45] Google Inc.: OpenAI and Pennsylvania State University, a repository for cleverhans library [Github Repository]. https://github.com/tensorflow/cleverhans (2016). Accessed 4 May 2022

[46] Parker, L.R., Yoo, P.D., Asyhari, T.A., Chermak, L., Jhi, Y., Taha, K., 2019. Demise: interpretable deep extraction and mutual information selection techniques for iot intrusion detection, in: Proceedings of the 14th International Conference on Availability, Reliability and Security, pp. 1–10. doi:10.1145/3339252.3340497.

[47] Ran, J., Ji, Y., Tang, B., 2019. A semi-supervised learning approach to ieee 802.11 network anomaly detection, in: 2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring), IEEE. pp. 1–5. doi:10. 1109/VTCSpring.2019.8746576

[48] Alotaibi, B., Elleithy, K., 2016. A majority voting technique for wireless intrusion detection systems, in: 2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT), IEEE. pp. 1–6. doi:10.1109/LISAT.2016.7494133

[49] Thanthrige, U.S.K.P.M., Samarabandu, J., Wang, X., 2016. Machine learning techniques for intrusion detection on public dataset, in: 2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), IEEE. pp. 1–4. doi:10.1109/CCECE.2016.7726677.

[50] Vaca, F.D., Niyaz, Q., 2018. An ensemble learning based wi-fi network intrusion detection system (wnids), in: 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA), IEEE. pp. 1–5. doi:10.1109/NCA.2018.8548315

[51] Mikhail, J.W., Fossaceca, J.M., Iammartino, R.: A semi-boosted nested model with sensitivity-based weighted binarization for multi-domain network intrusion detection.ACMTrans. Intell. Syst. Technol. 10, 1–27 (2017). https://doi.org/10.1145/3313778

[52] Blanco, R., Cilla, J.J., Briongos, S., Malagon, P., Moya, J.M.: Applying cost-sensitive classifiers with reinforcement learning to IDS. In: International Conference on Intelligent Data Engineering and Automated Learning, pp. 531–538. Springer, Berlin (2018). https://doi.org/10.1016/j.neucom.2019.02.056

[53] L. Nie et al., "Intrusion Detection in Green Internet of Things: A Deep Deterministic Policy Gradient-Based Algorithm," in IEEE Transactions on Green Communications and Networking, vol. 5, no. 2, pp. 778-788, June 2021, doi: 10.1109/TGCN.2021.3073714.

[54] Singh, P., Ranga, V. Attack and intrusion detection in cloud computing using an ensemble learning approach. Int. j. inf. tecnol. 13, 565–571 (2021). https://doi.org/10.1007/s41870-020-00583-w