An Adversarial Attack Detection Paradigm With Swarm Optimization

Ayyaz-Ul-Haq Qureshi* Hadi Larijani*, Nhamoinesu Mtetwa*, Mehdi Yousefi* and Abbas Javed†
* School of Computing, Engineering and Built Environment, Glasgow Caledonian University
Glasgow, United Kingdom
† Department of Electrical and Computer Engineering, COMSATS University Islamabad,
Lahore Campus, Pakistan

***Abstract*—The rise of smart devices and applications has increased the dependence of human beings on machine learning (ML) based code-driven systems. While many of the pragmatic problems such as image classification, medical diagnosis, and statistical arbitrage have been addressed by extensive recent research in machine learning, it still lacks substantial work in the field of adversarial attacks on safety-critical networked systems. It is a matter of significant importance, as using the adversarial samples, attackers are now able to evade pre-trained systems and mount black-box attacks hence increasing the false positives. In this research, we are proposing a Random Neural Network-based Adversarial intrusion detection system (RNN-ADV). For adversarial attack generation, the Jacobian Saliency Map Attack (JSMA) algorithm has been used. Swarm optimization capabilities have been implemented by training the system with the Artificial Bee Colony (ABC) algorithm. Different scenarios have been designed and the proposed system is then evaluated with benchmark benign NSL-KDD dataset, adversarial data, and the performance is compared with deep neural networks (DNN) using several performance metrics. The results suggest that the proposed scheme outperforms DNN in terms of adversarial attack detection where it has successfully classified benign samples from crafted samples with better accuracy and high F1 scores.**

***Index Terms*—Intrusion Detection, Swarm Intelligence, Adversarial Machine Learning, NSL-KDD, JSMA**

I. Introduction

IN today's hyper-connected world where digital technologies are easing the way we interact with each other, the threats to network security are ubiquitous. Computer networks are constantly getting proliferated and the risks towards securing user's specific information are increasing [1]. Although many techniques were proposed in the past to meet this challenge but the attackers are using more novel and sophisticated ways to accomplish their goal to bypass network security parameters such as firewalls etc. With the advent of Artificial Intelligence (AI), the future technologies are shaping towards designing self-operating vehicles to the telemedicine, etc, which would call-for the on-demand high-speed Internet to perform efficiently. In order to ensure such standards, the years 2020, would mark the release of new generation cellular networks known as 5G [2]. But many of the risk assessment organizations have warned that security threats would be more prominent in this year as attackers can now exploit the Internet-of-Things(IoT) user devices, easier than ever [3].

To mitigate against such extraordinary situations, intrusion detection systems (IDS) are developed. Unlike traditional firewalls which only monitor packets, IDS enables packet sniffing both inbound and outbound on the network due to their unique ability to search for novel threats. They are categorized as signature-based or anomaly-based [4]. Signature-based IDS requires a constant update to the known signature data so that IDS could work efficiently. Anomaly-based IDS learns from the existing data and uses patterns to quantify intrusion in the network [26]. The latter approach is more realistic and used widely to design intrusion detection applications. Throughout the literature, many techniques have been proposed to design an efficient intrusion detection system. Most popular among them is the machine learning (ML). In order to train the ML model, training algorithms are utilized. Such heuristic algorithms are necessary to increase the computational capacity of the technique under observation [5]. The solution to this complex problem could be swarm intelligence (SI) or evolutionary algorithms (EA) which can be iterative, population-based or stochastic [6]. We are focusing on swarm intelligence (SI), where a collective goal is achieved using individual entities/agents. The examples of such algorithms are Artificial bee colony (ABC), Ant colony optimization (ACO) and Particle swarm optimization (PSO).

In our previous work [4] we used the Artificial bee colony (ABC) algorithm to train our proposed model. ABC is a meta-heuristic training algorithm that is used to find an optimal solution during the learning stage. It works on the same principle as honey bees in hives. A population-based SI approach is utilized and different agents such as onlookers, scout, and employed bees work together to locate the food source, measure the nectar value and abandon it after finding the best value. Recent research has suggested that ML techniques and many of the proposed intrusion detection systems are vulnerable to adversaries [7] [8] [9] [10]. They increase the false positive rate of a classifier many folds by adding a careful perturbation to a benign sample. Adversaries use class ambiguity to reduce the trust of classifier and then trick it further to generate the wrong result which resembles the required output when actually it belongs to another class.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

To generate adversarial samples from benign samples and to check the performance of classifiers in the adversarial environment a few algorithms are proposed. In an extension of our previous work [4] we propose the Random Neural Networks based Adversarial Intrusion Detection System (RNN-ADV) which is trained with Artificial bee colony (ABC) algorithm. For crafting adversarial samples, the Jacobian Saliency Map Attack (JSMA) algorithm [8] is utilized. To evaluate the performance of RNN-ADV, different scenarios are devised where the NSL-KDD dataset [11] and crafted adversarial data are used to train/test the system. Performance is further compared to the deep neural network (DNN) and results are elaborated in further sections.
The primary contributions of this paper are:

- For adversarial attack detection, a random neural network based intrusion detection system (RNN-ADV) is presented using swarm optimization based Artificial bee colony (ABC) algorithm.
- The Jacobian Saliency Map Attacks (JSMA) algorithm is used to generate adversarial sample by computing forward derivative.
- Performance of RNN-ADV is compared with deep neural network in terms of accuracy, precision, recall, F1 score.

The rest of the paper sections are organized as follows: review of the literature reported in this research related to adversarial attacks, intrusion detection and random neural networks (RNN) is discussed in Section II. The Artificial bee colony (ABC) algorithm and the methodology for Adversarial attack crafting using Jacobian Saliency map Attacks (JSMA) algorithm is explained in Section III. The experimental results and analysis is presented in Section IV while the conclusion and future research directions are outlined in Section V.

II. Related Work

Intrusion detection systems (IDS) are used to detect malware entering the networks. Many approaches have been presented in the past to design and develop such systems. In [12], authors used the Recurrent Neural Network (RNN) to classify attacks from normal patterns. The system was trained with benchmark NSL-KDD and the authors concluded that the accuracy of proposed IDS is higher in binary-class as compared to the multi-class due to problems such as vanishing gradient. In [13], the authors concluded that Convolutional Neural Networks (CNN) based IDS performed better with respect to other ML platforms but performance needs further improvements in terms of U2R and R2L attacks. In [14], the random neural networks model is trained with the ABC algorithm and then performance was further compared with the GD algorithm. The authors conclude that ABC performed better for binary-class of NSL-KDD due to the novel ability of bees to measure the nectar value and optimize the network weights accordingly.

In [15] authors have used principal component analysis (PCA) to extract useful features from NSL-KDD and trained deep neural network (DNN) to estimate the performance with other benchmark techniques. Results suggest that proposed IDS performed better for binary-class of the dataset while classifier needs further improvement to enhance the performance for the multi-class category. In [16], Chaithanya et.al developed IDS using Moth-Flame Optimization Algorithm-based Random Forest (MFOA-RF) and compared the result based on different performance metrics while to enhance efficiency, in [17] authors developed IDS using State Preserving Extreme Learning Machine(SPELM) algorithm.

Even though neural networks are reported to have good performance in intrusion detection but Szedgy et.al in [18] concluded that the false rate of a classifier can be increased if we add careful perturbation to the existing data, thus making them vulnerable to adversarial attacks. Samples were generated by Broyden-FletcherGoldfarb-Shanno (LBFGS) optimization algorithm. After that, a lot of research has been carried out to check the performance of ML classifiers against adversaries but most of them were addressed in the field of image processing. Many of the algorithms were further proposed to craft the adversarial samples.

To improve the process, Goodfellow et.al in [19] developed the Fast Gradient Sign Method (FGSM) algorithm to craft the adversarial samples where the algorithm calculates the loss of gradient of a function to produce samples by producing perturbation which is the sum of controlled parameter and input gradient. Moosvi et.al, in [20] proposed the DeepFool algorithm to craft adversarial samples. The minimum perturbation is added. The adversary is located by establishing a clear boundary between different class labels and perturbations are added in fashion. Performance is compared with other ML techniques and enhanced via fine-tuning parameters. While in [8], Papernot et al, proposed the Jacobian Saliency Map Attack (JSMA) algorithm where adversaries are crafted by mapping inputs to the corresponding outputs. Perturbations are added to the minimum level and saliency maps are calculated. Authors conclude that JSMA can increase false positives of the system even if a few features are altered after adding perturbations to the benign samples.

In this research, we are exploring the effect of adversarial attacks on random neural networks. Jacobian-based Saliency Map Attack (JSMA) algorithm is used to generate adversarial samples and the system is trained with the Artificial Bee Colony (ABC) algorithm. The effect of adversaries on the proposed system is then evaluated using different performance matrices.

Random Neural Network Model: A novel class of artificial neural networks was proposed by Gelenbe named Random Neural Network (RNN) [21]. RNNs have been used extensively for pattern recognition [22]. However, a little research has been reported to analyze the effectiveness

of RNN's for intrusion detection systems using NSL-KDD dataset.

As mentioned in our previously published work [4] [14], in an RNN model, neurons trigger the excitation and inhabitation states whenever any signal with positive or negative potential arrives [14] [25]. In Random neural Network Model, the neurons exchange information using positive excitation and negative inhibition signals. The information flows between neighbouring neurons in time t as an impulse. Based upon the behaviour of neuron g, the following probabilities can occur with its current state $J_g(t)$:

If $J_g(t) = 0$ The neuron n_g would remain inactive
If $J_g(t) > 0$ The neuron n_g transmits the information with firing rate o_g towards neighbouring neuron n_h

Mathematically,

$$b(i) + \sum_{h=1}^{N} [p^+(g,h) + p^-(g,h)] = 1, 1 \leq g \leq n, \quad (1)$$

Where, $p^+(g,h)$ and $p^-(g,h)$ represent probabilities due to excitation and inhibition signal whereas $b(g)$ symbolises the departure probability of the information. During training phase, Poison rate $\Lambda(g)$ is used to denominate positive signal while Poisson rate $\lambda(g)$ is for the negative signal. The output activation function $s(g)$ for neuron g is defined as:

$$s(g) = \frac{\lambda^+(g)}{o(g) + \lambda^-(g)}, \quad (2)$$

where

$$\lambda^+(g) = \sum_{h=1}^{n} p^+(h,g) \times s(h) \times o(h) + \Lambda(g), \quad (3)$$

and

$$\lambda^-(g) = \sum_{h=1}^{n} p^-(h,g) \times s(h) \times o(h) + \lambda(g), \quad (4)$$

Also,

$$o(g) = (1 - b(g))^- 1 \sum_{h=1}^{N} [w^+(g,h) + w^-(g,h)] \quad (5)$$

$o(g)$ is the firing rate between neurons whereas $w^+(g,h)$ and $w^-(g,h)$ represent weight updates between neurons g and h. Mathematical model is further explained in [21] [4] [23] and [1].

III. Methodology

In this paper, a random neural network-based adversarial intrusion detection system (RNN-ADV) is utilized. NSL-KDD data with full feature space is used and the adversarial attack crafting is accomplished using the Jacobian Saliency map Attack (JSMA) algorithm. A stochastic, population-based approach is used and the model is trained with an artificial bee colony (ABC) algorithm. Data is preprocessed where it is normalized and one-hot encoded. Since we are comparing

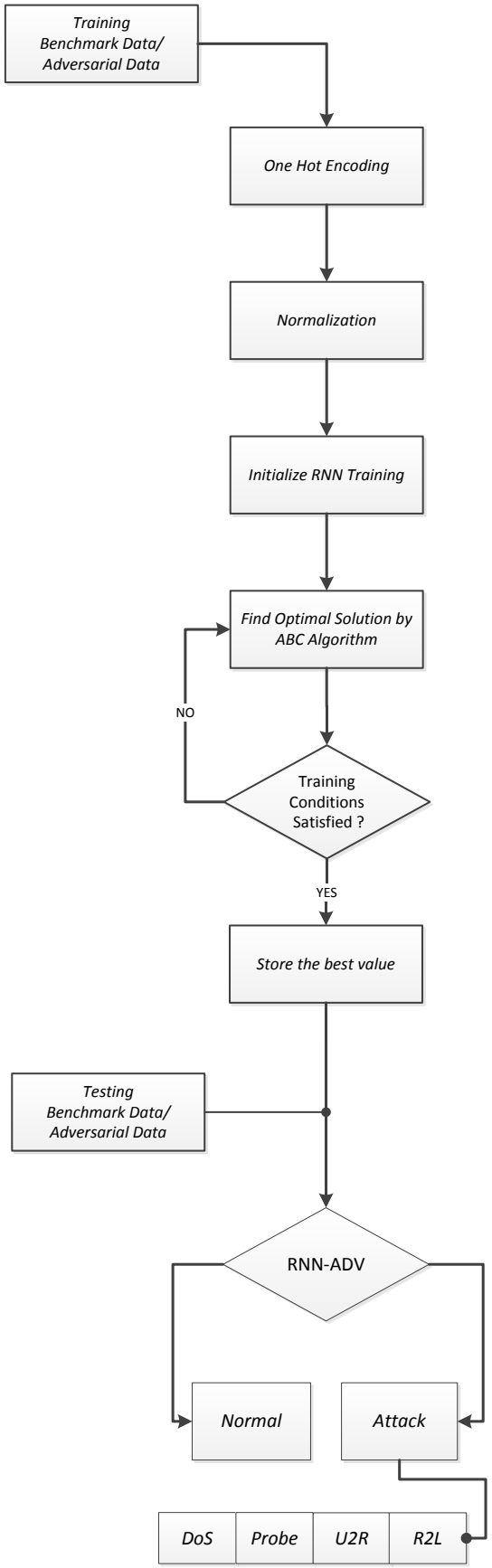

Fig. 1: Steps for Adversarial Attack Detection System (RNN-ADV) using Swarm Optimization

our work with deep neural networks, the neurons in the proposed RNN-ADV are structured among two hidden layers and 1 output layer. The size of the bee colony is not changed and the maximum iteration limit is applied to ensure fast optimization when bees are looking for an appropriate solution in identified food sources. Following steps as shown Figure: 1 are undertaken to design adversarial IDS.

The Dataset and Pre-Processing: For this study, the NSL-KDD dataset is used. Although a lot of datasets are not the actual representations of real-world data but NSL-KDD is still considered as a benchmark for training and testing of network intrusion detection systems (IDS). From the total of 42 features, first 41 feature is used as inputs which contain the traces from collected network traffic such as $Dstbytes, Protocol, ServerrRate$, etc. The 42^{nd} feature is the output label which contains information about the targeted class. The attack in the output label varies from Probe, Root-to-local (R2L), Denial-of-Service and User-to-root (U2R). The dataset is widely used due to its packet distribution among normal and abnormal packets in training and testing sets, where it contains 46.5% attack patterns.

The dataset also lists three features that are nominal in nature, and we know the fact that the ML algorithm can only process the information which is binary or numeric in nature. For this purpose, we have utilized "One-hot Encoding" where these features are converted to the designated numeric values. Many of the researchers [14] [23] utilized feature selection by using the gain ratio (GR), correlation-based feature extraction (CFS) and information gain (IG) which helps reduce the complexity of classifier. For this study we use the complete feature space of the dataset and performance is estimated.

To truncate the training time of proposed RNN-ADV, data is normalized and then used as corresponding input. Min-Max normalization is exploited and data is processed accordingly. Mathematically,

$$a_i = \frac{w_i - \min(w)}{\max(w) - \min(w)}, \tag{6}$$

where:

- $g = (w_1, ..., w_n)$ indicates output
- $a_(i)$ denotes input

Artificial bee Colony (ABC) Algorithm: ABC is a meta-heuristic training algorithm that is used to find an optimal solution during the learning stage. It works on the same principle as honey bees in hives. Population consists of onlookers, scout, and employed bees work together to locate the food source, measure the nectar value and abandon it when finding the best value.

The search process of a food source in a search space is initiated by scout bees with random values. Employed bees are responsible to visit the identified food sources. Based on the information onlooker bees take the further actions which include measuring the 'nectar' amount and fitness value (fit_p) is then estimated as [14]:

$$fit_p = \begin{cases} \frac{1}{1+f(p)} & f(p) \geqslant 0 \\ 1+|(f(p)| & f(p) < 0 \end{cases} \tag{7}$$

If the new solution is better than the previous one, the value is updated, else the bees would continue searching the data space of E dimensions and apply greedy selection process for more appropriate value iteratively until one of the following conditions is reached:

- A solution with better fitness value is obtained.
- Maximum iterations limit to look for search space is reached.
- Minimum value for mean squared error is achieved.

After fulfilling the upper conditions, the bees abandon previously selected food source and scout bees initiate the search process all over by random values.
Mathematically:

$$h_{pq} = h^q_{min} + rand(0,1)(h^q_{max} - h^q_{min}) \tag{8}$$

where

- h_{pq} food source value, ranges from $p = 1....SS$,
- SS is population size.

Jacobian Saliency Map Attacks

As discussed before, Szedgy et.al [18] reported that neural networks are vulnerable to adversarial attacks. Many algorithms were proposed to craft adversarial samples from benign samples but most of them use gradients to produce desired output. In [8], Papernot et. al proposed a new algorithm known as Jacobian Saliency Map Attacks (JSMA) which is used to generate adversaries, where the input is mapped to desired output by establishing direct mapping. While preparing the dataset for RNN model, if the activation function $F : C \mapsto D$ where C original input and D represents desired output, then to generate adversary C^*, consider the following mathematical model:

$$\arg\max_{\sigma_c} \parallel \sigma_c \parallel \text{ s.t. } F(C + \sigma_c) = D^*, \tag{9}$$

where,

- σ_c is the perturbation vector
- $\parallel . \parallel$ is the relevant norm for RNN input comparison
- D^* is the required adversarial output data points/features
- $C + \sigma_c$ = C^* is the adversarial sample

For the benign samples, $F(C) = D$, the idea is to create adversary C^* in such a way that it satisfies the condition

$F(C^*) \neq D^*$ but resembles the original sample C. The *forward derivate* approach is used to add perturbation σ_c, which would then return the features which are altered as adversarial samples in a search space as mentioned in Algorithm 1. The non-negative integers m_1 and m_2 demonstrate the total change applied to the original features. Mathematically:

$$\nabla F(C) = [\frac{\delta F(C)}{\delta m_1}, \frac{\delta F(C)}{\delta m_2}] \tag{10}$$

Algorithm 1 Adversarial Attack Crafting with JSMA

Input: C, D^*, F, ζ, α
Initialization:
$C^* \leftarrow C$
for $F(C^*) \neq D^*$ and $\| \sigma_c \| \text{ ¡ } \zeta$ **do**
  Initiate Saliency Map
  compute $\nabla F(C^*)$
  $\| \sigma_c \| \leftarrow (C) - (C^*)$
end for
return D^*
end procedure

IV. Experimental Results and Analysis

In this section, we have explained the experimental results and presented an analysis of the effects of adversarial attacks on random neural networks. The system is trained with the artificial bee colony algorithm (ABC). NVIDIA GPU acceleration is used for the training. Several performance matrices such as Precision (P), Recall (R), F1 Score, False Alarm (FA) and Accuracy (ACC) are used to elaborate on the results which are denoted as φ, ρ, μ, and ν the True Positives (TP), True Negatives (TN), False Positives (FP) and False Negative (FN) respectively.

$$Accuracy\,(RNN-ADV) = ACC = \frac{\varphi+\rho}{\mu+\rho+\nu+\varphi} \tag{11}$$

$$Precision = P = \frac{\varphi}{\mu+\varphi} \tag{12}$$

$$Recall = R = \frac{\varphi}{\nu+\varphi} \tag{13}$$

$$FalseAlarm = FA = \frac{\mu}{\rho+\mu} \tag{14}$$

$$F1\,Score = 2x(\frac{PrecisonxRecall}{Precision+Recall}) \tag{15}$$

To estimate the performance of proposed RNN-ADV, different methods were adopted and scenarios are created. Results are then compared to deep neural networks. The network consists of two hidden layers and trained at the learning rate of 0.001. NSL-KDD data and Adversarial data are used to train/test the system. For adversarial attack crafting using the JSMA algorithm. Using L_0 distance metric, the perturbation added is 0.5 [8]. Cleverhans python library [24] is also utilized. As mentioned in [4], during the training phase, a total of 20 bees coordinate together to find the optimal value with a maximum iterations limit of 100. The number of food sources to be searched and employed bees are 10. Following scenarios were adopted.

Baseline Scenario: For benchmarking, the proposed RNN-ADV is trained with the $Train^+$ and tested against $Test^+$ sample of the NSL-KDD data. This would help to establish the fundamental performance of the system.

Adversarial Scenario: After adversarial attack crafting using the Jacobian Saliency map Attack (JSMA) Algorithm, the proposed RNN-ADV is trained/tested with adversarial data to understand the performance when an attacker can compromise network integrity by introducing adversaries.

Attack Scenario: An attack scenario is devised, where the system is trained with predefined data from NSL-KDD but tested with crafted adversarial data. This resembles real-world attacks where pre-trained systems can be accessed by attackers with the help of crafted adversarial samples.

TABLE I: Baseline Scenario: Performance of RNN-ADV with 'Benchmark Data'

Metrics	Accuracy	Precision	Recall	False Alarm	F1- Score
Normal	89.42	99.89	93.52	43.22	96.60
Denial-of-Service (DoS)	97.51	99.47	97.81	3.24	98.63
Probe	92.69	98.24	79.33	3.98	87.77
User-to-Root (U2R)	46.12	92.93	51.21	0.96	66.03
Root-to-Local (R2L)	66.72	97.11	42.49	1.38	59.11

As discussed, a baseline scenario for benchmarking the performance of RNN-ADV is proposed where the system is trained with $Train^+$ data points. The system is trained with the ABC algorithm and the input layer is fed with complete feature space. The trained system is then tested with $Test^+$ data and results are reported in Table: I. Several performance metrics such as accuracy, precision, recall, false alarm, and

TABLE II: Adversarial Scenario: Performance of RNN-ADV with 'Adversarial Only Data'

Metrics	Accuracy	Precision	Recall	False Alarm	F1- Score
Normal	70.71	59.35	47.24	50.21	52.60
Denial-of-Service (DoS)	76.24	53.12	38.29	36.98	44.50
Probe	80.97	39.65	36.25	33.13	37.87
User-to-Root (U2R)	42.87	6.39	8.58	7.95	7.32
Root-to-Local (R2L)	59.22	33.11	32.98	31.11	33.04

TABLE III: Attack Scenario: Performance of RNN-ADV with 'Heterogeneous Data'

Metrics	Accuracy	Precision	Recall	False Alarm	F1- Score
Normal	66.85	48.21	37.35	47.98	42.09
Denial-of-Service (DoS)	72.58	41.58	31.12	33.57	35.59
Probe	75.02	30.14	27.77	29.42	28.90
User-to-Root (U2R)	38.59	3.57	5.69	5.14	4.38
Root-to-Local (R2L)	98.27	22.27	24.22	27.69	23.20

F1-score are used to demonstrate the results. For the multi-class category of NSL-KDD data, RNN-ADV successfully classified normal patterns from anomalous records by 89.42%. For attack classes, it classified denial-of-service attacks with an accuracy of 97.51% which is the highest among other attacks such as probe 95.69%, U2R 46.12% and R2L 66.72%. For better understanding F1-Score metric is used which is calculated by taking a harmonic mean of recall and precision values. It remained 96.60% for normal and the highest value is 98.63% for attack classes. Since there is no adversary added to the system, the false alarm rate for attack classes is recorded low.

To check the performance of RNN-ADV in an adversarial environment, a second scenario is devised where the system is trained and tested with adversarial only data. For this purpose, the Jacobian Saliency Map Attack (JSMA) algorithm is used, which unlike previous schemes that use output gradient to generate perturbation, maps inputs to the desired and calculates the difference. Perturbation is added and saliency maps are computed which would return the adversarial data/features. The main aim of this protocol is to craft adversarial samples with minimum added perturbation and more change from benign samples while keeping them related. The results reported in Table: II suggest that the accuracy of the normal class falls to 70.71% while attack classes also depict performance depreciation. Adding an adversary to the benign sample has significantly increased the false alarm rate for both normal and attack class, which validates our claim that RNN-ADV successfully classified adversarial packets. F1-Score is reported to 52.60% for normal and 44.50% for attack class in case of denial-of-service attacks.

Since machine learning models are prone to adversarial attacks, we have developed an attack scenario where RNN-ADV is trained with benchmark $Train^{+}$ data it is tested with adversarial data. This scenario represents a real-time situation where an ML model is trained with required data as a black-box model but an attacker is manipulating the network integrity by adding perturbations and gain unauthorized access. Results are reported in Table: III, which shows that accuracy for the normal class is further decreased by 5% while attack classes also show performance depreciation. False alarm rate for normal class is 37.35% while it is 33.57%, 29.42%,5.14,% and 27.69% for DoS, probe, U2R and R2L attacks respectively. F1-score, which is used as an overall performance indicator is also decreased by 10% for both normal and attack classes. This change in performance is happening due to the class imbalance which is created by an adversary and targeted classes are misclassified based on added perturbation to the benign sample.

To estimate the performance of proposed RNN-ADV,

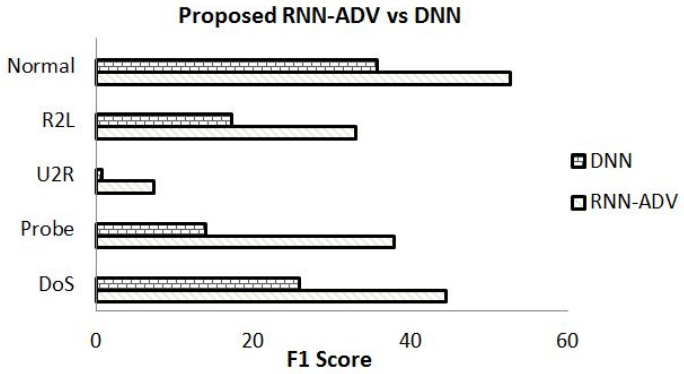

Fig. 2: Performance Comparison between Adversarial RNN-ADV and DNN

results are reported using different matrices in Table: I, II and III. F1-score which is the harmonic mean of precision and recall values [14] is used as a key performance indicator. We have compared the results with deep neural networks based adversarial IDS [7] in Figure: 2 where it shows that, when RNN-ADV is tested in an adversarial environment, the F1-score is 52.60% for normal class and 44.65% for denial-of-service attacks as compared to a deep neural network where these indicators are 35.69% and 25.89% respectively. This proves that proposed RNN-ADV performs better and classifies adversaries with improved precision and higher accuracy.

V. Conclusion

To detect adversarial attacks on computer networks, a random neural network-based adversarial intrusion detection system (RNN-ADV) has been proposed. The system is trained with an artificial bee colony (ABC) algorithm. Adversarial attack crafting is accomplished using the Jacobian Saliency map Attacks (JSMA) algorithm. After training/testing the system with benign samples and adversarial samples the result suggests that even though false alarms have increased but the proposed RNN-ADV successfully classified attacks from normal traffic under adversarial settings. More optimal solutions throughout the search space of the given data were identified by artificial bee colony algorithm. Since the JSMA only alters a few features to add perturbations which makes it more viable in real-time implementation, the RNN-ADV used those features to improve adversarial attack detection capabilities in terms of higher accuracy, precision, and better F1-Score. As, the detection of adversaries remains an open issue, in the future, we would extend this work to craft adversarial samples by Fast Gradient Sign Method (FGSM), DeepFool and CW attack algorithms. Better training can help classifiers to reduce misclassification for this purpose different training algorithms such as Levenberg-Marquardt algorithm, Particle Swarm Optimization (PSO) can be used.

References

[1] A. U. H. Qureshi, H. Larijani, J. Ahmad, and N. Mtetwa, "A heuristic intrusion detection system for internet-of-things (iot)," in *2019 Springer Science and Information (SAI) Computing Conference*. Springer, July 2019.

[2] R. P. Jover, "The current state of affairs in 5g security and the main remaining security challenges," *CoRR*, vol. abs/1904.08394, 2019.

[3] P. P. Sriram, H.-C. Wang, H. G. Jami, and K. Srinivasan, *5G Security: Concepts and Challenges*. Cham: Springer International Publishing, 2019, pp. 1–43.

[4] A. Qureshi, H. Larijani, A. Javed, N. Mtetwa, and J. Ahmad, "Intrusion detection using swarm intelligence," in *2019 UK/ China Emerging Technologies (UCET)*, Aug 2019, pp. 1–5.

[5] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, Nov 2007.

[6] S. Mirjalili, *Genetic Algorithm*. Cham: Springer International Publishing, 2019, pp. 43–55.

[7] Z. Wang, "Deep learning-based intrusion detection with adversaries," *IEEE Access*, vol. 6, pp. 38 367–38 384, 2018.

[8] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European Symposium on Security and Privacy (EuroS P)*, March 2016, pp. 372–387.

[9] N. Martins, J. M. Cruz, T. Cruz, and P. H. Abreu, "Analyzing the footprint of classifiers in adversarial denial of service contexts," in *Progress in Artificial Intelligence*, P. Moura Oliveira, P. Novais, and L. P. Reis, Eds. Springer International Publishing, 2019, pp. 256–267.

[10] W. Brendel, J. Rauber, A. Kurakin, N. Papernot, B. Veliqi, S. P. Mohanty, F. Laurent, M. Salathé, M. Bethge, Y. Yu, H. Zhang, S. Xu, H. Zhang, P. Xie, E. P. Xing, T. Brunner, F. Diehl, J. Rony, L. G. Hafemann, S. Cheng, Y. Dong, X. Ning, W. Li, and Y. Wang, "Adversarial vision challenge," in *The NeurIPS '18 Competition*, S. Escalera and R. Herbrich, Eds. Cham: Springer International Publishing, 2020, pp. 129–153.

[11] "NSL-KDD — Datasets — Research — Canadian Institute for Cybersecurity — http://www.unb.ca/cic/datasets/nsl.html, Last Accessed 2018-05-03."

[12] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks," *IEEE Access*, vol. 5, pp. 21 954–21 961, 2017.

[13] S. Z. Lin, Y. Shi, and Z. Xue, "Character-Level Intrusion Detection Based On Convolutional Neural Networks," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, jul 2018, pp. 1–8.

[14] A.-U.-H. Qureshi, H. Larijani, N. Mtetwa, A. Javed, and J. Ahmad, "Rnn-abc: A new swarm optimization based technique for anomaly detection," *Computers.*, vol. 8, no. 3, 2019-9-14.

[15] S. Rawat, A. Srinivasan, and V. R, "Intrusion detection systems using classical machine learning techniques versus integrated unsupervised feature learning and deep neural network," 2019.

[16] P. S. Chaithanya, M. R. Gauthama Raman, S. Nivethitha, K. S. Seshan, and V. S. Sriram, "An efficient intrusion detection approach using enhanced random forest and moth-flame optimization technique," in *Computational Intelligence in Pattern Recognition*, A. K. Das, J. Nayak, B. Naik, S. K. Pati, and D. Pelusi, Eds. Singapore: Springer Singapore, 2020, pp. 877–884.

[17] K. singh and K. J. Mathai, "Performance comparison of intrusion detection system between deep belief network (dbn)algorithm and state preserving extreme learning machine (spelm) algorithm," in *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, Feb 2019, pp. 1–7.

[18] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," 2013.

[19] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014.

[20] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[21] E. Gelenbe, "Random Neural Networks with Negative and Positive Signals and Product Form Solution."

[22] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," sep 2014.

[23] A. Qureshi, H. Larijani, J. Ahmad, and N. Mtetwa, "A novel random neural network based approach for intrusion detection systems," in *2018 10th Computer Science and Electronic Engineering (CEEC)*, Sep. 2018, pp. 50–55.

[24] N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, A. Matyasko, V. Behzadan, K. Hambardzumyan, Z. Zhang, Y.-L. Juang, Z. Li, R. Sheatsley, A. Garg, J. Uesato, W. Gierke, Y. Dong, D. Berthelot, P. Hendricks, J. Rauber, and R. Long, "Technical report on the cleverhans v2.1.0 adversarial examples library," *arXiv preprint arXiv:1610.00768*, 2018.

[25] O. Brun and Y. Yin, "Random neural networks and deep learning for attack detection at the edge," in *2019 IEEE International Conference on Fog Computing (ICFC)*, 2019, pp. 11–14.

[26] S. Naseer, Y. Saleem, S. Khalid, M. K. Bashir, J. Han, M. M. Iqbal, and K. Han, "Enhanced Network Anomaly Detection Based on Deep Neural Networks," *IEEE Access*, vol. 6, pp. 48 231–48 246, 2018.