Random Neural Network for Lightweight Attack Detection in the IoT*

Katarzyna Filus, Joanna Domanska and Erol Gelenbe

Institute of Theoretical & Applied Informatics
IITIS-PAN, Polish Academy of Sciences
ul. Baltycka 5, 44100 Gliwice PL
https://www.iitis.pl

Abstract. Cyber-attack detection has become a basic component of all information processing systems, and once an attack is detected it may be possible to block or mitigate the effect of the attack. This paper addresses the use of a learning recurrent Random Neural Network (RNN) to build a lightweight detector for certain types of Botnet attacks on IoT systems. Its low computational cost based on a small 12-neuron recurrent architecture makes it particularly attractive for edge devices. The RNN can be trained off-line using a fast simplified gradient descent algorithm, and we show that it can lead to high detection rates of the order of 96%, with false alarm rates of a few percent.

Keywords: IoT Attack detection, Random Neural Network, Machine Learning, Lightweight attack detection

1 Introduction

Cyberattacks are increasingly common, sophisticated and malignant, while we constantly increase our reliance on Internet connected devices in almost every area of life: smart homes, cities, health monitoring, industry, military, agriculture etc. Also, the rapid increase in the number of IoT devices, their functionality and connectivity, has introduced additional security threats [11] because such simple devices are more prone to vulnerabilities that can lead to data theft, power supply drainage [31], and compromizes that lead to their use in Botnets for Distributed Denial of Service (DDos) attacks [34].
Since they are often Internet accessible and interconnected for machine-to-machine (M2M) communications, IoT devices are a natural "doorway" for attackers, and the use of wireless communications also increases their vulnerability [36].
IoT devices are often embedded with microcontrollers powered by batteries; therefore energy saving is important and complex attack detection

* This research was supported by the EU H2020 IoTAC Research and Innovation Action, funded by the European Commission (EC) under Grant Agreement ID: 952684. The EC's support does not constitute an endorsement of this paper, which reflects the views only of the authors.

techniques that require a large amount of computation cannot be installed on such systems [55]. Thus much work has been done on creating simple yet accurate intrusion detection techniques for IoT platforms and evaluating them on representative datasets [**?**, 10].
Over the years, many different techniques have been used for attack detection. In [39] it is indicated that the major algorithms used in the last decade for intrusion detection are based on the Artificial Neural Networks (ANNs), including Deep Learning based approaches that have gained popularity due to thei ability to extract patterns better than shallow learning methods despite their need for additional computational resources [60]. While for some IoT devices security is crucial, for others energy-efficiency is critical; thus the trade-off between security-effectiveness and energy-efficiency needs to be considered [36].
Hence we propose a recurrent Random Neural Network (RNN) for light-weight attack detection which can be trained off-line, creating a small but effective network that produces satisfactory results with a minimum of computationally demanding operations, and potentially low energy consumption. The small size of the RNN leads to less storage space and energy consumption for storage, as well as to lower computation times for detection which also save energy. We initialize the network weights in such a way as to establish the "neutraility" of the network prior to the learning process for faster and more accurate learning. We also reduce the number of computationally demanding operations during learning by fixing the total value of the RNN excitatory and inhibitory weights, so that only excitatory weights need to be updated with an automatic effect on the inhibitory weights.
In the sequel, Section 2 discusses the area of Intrusion Detection and provides a literature overview. In Section 3 we discuss the RNN and summarize its initialization and the simplified learning algorithm. Section 4 describes our experimental results based on training the RNN with attack data and testing it using disjoint attack data. Finally we draw some conclusions and suggest future work.

2 Related Work on CyberAttack Detection

Over the years, many different cyberattack datasets have been created such as the DARPA datasets [15, 16], the KDDCup'99 datasets [38], and their successors. However, since the time when these datasets were created more than twenty years ago, attacks have changed and detectors that were trained with that data have become less reliable [39]. Thus in recent years, datasets with IoT traffic have been created such as N-Baiot [49], IoT host-based datasets for IDS research [8], the IoT Network Intrusion Dataset [46], the BoT-IoT dataset [47] and MedBIoT [34].
The Bot-IoT dataset was created in the Cyber Range Laboratory of the University of New South Wales' Canberra Cyber Center (Australia). Their testbed consists of network platforms and simulated IoT services. The network platforms include normal and attacking virtual machines with additional network devices such as a packet filtering firewall and Network Interface Cards. The Ostinato software [51] is used to generate

realistic normal network traffic, and four Kali Linux machines are used to simulate a set of standard Botnet attacks. The IoT sensors are simulated and use MQTT, the machine-to machine connectivity protocol, to transfer messages to a Cloud Service provider (AWS). The MQTT protocol runs over TCP/IP. The network protocol analyser (tshark) is used to capture the normal and attack raw data. Packet capturing is performed with the pcap library. The pcap files that are collected, are 69.3 GB in size and consist of 72 million records.
While in some datasets [8, 46, 49] only a very limited range of IoT multimedia devices are used, the Bot-IoT dataset [47] uses a variety of devices such as smart lights, a smart thermostat, a weather monitoring station, smart garage doors and a refrigerator, and it has been used in many recent papers [9, 18–20, 35].
In the sequel we will use the Bot-IoT dataset which consists of real and simulated IoT traffic, and contains three main types of attacks: both DoS and DDoS attacks, information gathering, and information theft.

2.1 Attack Detection

Many different approaches have been used for attack detection. Signature-based detection [12, 48] utilizes known patterns of the attacks to detect abnormal behaviours in the network traffic. This type of detection does not have a strong generalization power and can be easily bypassed by novel types of attacks [34].
Anomaly-based detection systems operate by identifying the patterns that define normal and abnormal traffic, and can be divided into three main groups: knowledge-based, statistical and Machine Learning (ML) based. Knowledge-based systems use a set of rules and finite state machines. Statistically based techniques [4] are typically based on time series. The ML approaches covers include clustering techniques [59], Genetic Algorithms [56], and ANNs [37] and Deep Learning.
As indicated in [39], the most popular approach in the last decade has been to use ML based algorithms, especially ANNs that are successful because of their classification power and their ability to generalize to datasets for which they have not been specifically trained. A recent trend is also the increased usage of Deep Learning techniques for IDS [18,20,61].
Attack detection aims to identify within a given traffic stream, those sub-streams that are viewed as being "normal", and those sub-streams that are likely to contain various forms of attacks. Typically, a binary classification into "normal" and "attack" traffic is preferred because of its simplicity [45]. Indeed, detectors that attempt to seek more detailed classifications for instance into different types of attacks, tend to be more error prone, leading to a decrease in classification accuracy [60].

3 The Random Neural Network

The Random Neural Network (RNN) was introduced in [21]. It is a biologically inspired spiking neural network model, which has a recurrent structure that incorporates feedback loops. It has been proved to be

a universal approximator for continuous and bounded functions [27, 28]. Different gradient descent based learning algorithms have been suggested for the RNN [**?**, 7].
RNNs have found application in many different fields, including modelling, optimization, image processing, communication systems, pattern recognition and classification [57]. For instance, they have been used for combinatorial optimization [23,26]. In [5,6] it has been shown that RNNs can be utilized to generate textures.
Other applications include the detection of explosive mines [**?**], medical image segmentation [24], the detection and classification of vehicles [41], video compression [13], the evaluation of subjective metrics of user satisfaction regarding the quality of service in networks [54], and as a detector of DD0S Attacks [50]. The image texture recognition capability of the RNN exchibited in [30], was used to accurately insert 3-D images in moving virtual reality scenes [25, 40] and applied to the augmented reality simulation of transportation systems [42].
The RNN has also been used to evaluate the voice and video quality [32, 53] of multimedia data streams. In smart buildings they have been used for the dynamic management of energy [2,3] and of heating, ventilation, airconditioning and cooling systems [43, 44].
In the field of communications, RNNs have been used to control the modulation of downlink traffic in LTE systems [1], to construct adaptive network routing algorithms [29] in smart networks [17], to design intrusion detection in networks [52], and to optimally schedule video sequences for content delivery [33].

3.1 The Random Neural Network

In this section, we present the notation for the RNN [21]. The RNN with N neurons is represented by a vector of non-negative integers $K(t)$ and by a probability distribution $P[K(t) = k]$:

$$K(t) = (K_1(t),\ ...\ K_N(t)),\ K_i(t) \geq 0, \tag{1}$$

$$p(k,t) = Prob[K(t) = k],\ where \tag{2}$$

$$k = (k_1,\ ...\ , k_N),\ k_i \geq 0,$$

and k is a specific value taken by $K(t)$, where $K_i(t)$ represents the excitation level of neuron i in the network, and it is non-negative (as already indicated) and unbounded. Each neuron in a RNN receives *external excitatory and inhibitory spikes* according to independent Poisson processes of rate $\Lambda_i \geq 0$ and λ_i, respectvely.
If $K_i(t) > 0$, neuron i can "fire" or spend a spike after an exponentially distributed interval of parameter $r_i \geq 0$, either to some other neuron j with probability p^+_{ij} as an excitatory spike, or with probability p^-_{ij} as an inhibitory spike. We denote by $w^+_{i,j} = r_i p^+_{ij}$ and $w^-_{i,j} = r_i p^-_{ij}$ the excitatory and inhibitory outgoing weights of neuron i. The spike leaving neuron i when it is excited, may also leave the network as a whole with probability $d_i \geq 0$ so that $d_i + \sum_{j=1}^{N} [p^+_{ij} + p^-_{ij}] = 1$ for all neurons i.
When neuron i receives an excitatory spike at time t, it state increases by $+1$, i.e. $K_i(t^+) = K_i(t) + 1$. If it receives an inhibitory spike then it

decreases by one, but only if it was previously positive, i.e. : $K_i(t^+) = max\ [0,\ K_i(t) - 1]$. The key theoretical result that was proved in [22] regarding the RNN is as follows:

Theorem Define $p(k) = \lim_{t\to\infty}\ p(k,t)$. If:

$$q_i = \frac{\Lambda_i + \sum_{j=1}^{N} q_j w^+ j, i}{r_i + \lambda_i + \sum_{j=1}^{N} q_j w^-_{j,i}} < 1, \tag{3}$$

$$It\ follows\ that\ for\ k = (k_1,\ ...\ k_N)$$

$$p(k) = \prod_{i=1}^{N} q_i^{k_i}(1 - q_i),\ and$$

$$q_i = \lim_{t\to\infty} Prob[K_i(t) > 0].$$

The conditions under which for all neurons we have $q_i < 1$ are discussed in [22].

3.2 Initialization of the Network Weights

It is quite common to initialize a neural network with randomly generated weights, or to select them using some other method that may optimize some criterion [58]. Here we select network weights so that prior to learning, the excitation probability of each neuron is given by the "neutral" value $q_i = 0.5$ for $i = 1,\ ...\ , N$ when the input to each neuron is also set to a neutral value. In particular the neutral input value is selected as $\lambda_i = 0$, $\Lambda_i = \Lambda^o > 0$ and $w^+_{ij} = w^-_{ij} = w$, for $i, j = 1,\ ...\ , N$, so that:

$$q_i = \frac{\Lambda^o + wQ}{2Nw + wQ}, \tag{4}$$

$$where\ Q = \sum_{i=1}^{N} q_i = Nq, \tag{5}$$

$$so\ that\ if\ q_i = q = 0.5,\ we\ have\ w = \frac{4\Lambda^o}{3N}.$$

3.3 Learning

The biggest asset of ANNs is their ability to adapt by learning from a given set of examples. An ANN llearning algorithm typically sets the network weights in such a way as to map the values of the output neurons in a manner that matches the requirements of a classification or decision scheme, as a function of the input values received by the ANN. In a recurrent network, which is obviously not going to be "feedforward" with data going from a set of inputs to a set of outputs, even though still we distinguish the input and output values of the network, some or all of the neurons may act as both input and output neurons.

The ANNs' ability to learn is closely related to their property of being universal approximators for bounded and continuous functions [14], which was also established for the RNN in [27,28]. Of the many different

types of learning algorithms, the ones based on Gradient Descent – that also include most Deep Learning Algorithms – are commonly used. In particular, the RNN's Gradient Descent learning algorithm was introduced in [22] and extended in [30], both for feedforward and recurrent (feedback) networks. In [7] a further extension was presented for feedforward RNNs.

Denote by $\Lambda = (\Lambda_1, \ ... \ , \Lambda_N)$ and $\lambda = (\lambda_1, \ ... \ , \lambda_N)$. Denote by $\iota = (\Lambda, \lambda)$ and let the vector $Y = (y_1, \ ... \ , y_N)$ be such that $y_i \in [0, 1[$. Suppose we are given a data set D which is composed of pairs (ι, Y). Then a simple objective of a learning algorithm can be stated as follows. Let W be the set of all weights of the network $W = \{w^+_{i,j}, w^-_{i,j} : \ 1 \leq i, j \leq N\}$. Then the learning algorithm approximates the following optimization problem:

$$\arg\min_W C, \ where \ C = \frac{1}{2} \sum_{(\iota,Y)\in D} a_i [q_i(\iota, W) - y_i]^2. \tag{6}$$

and $a_i \geq 0$ is a constant which determines the relative importance of neuron i. In the experiments presented in the sequel, the network has just one output neuron, so that we will set $a_i = 1$ only for that neuron, while $a_j = 0$ for all other neurons.

Note that we have written $q_i(\iota, W)$ to stress the fact that q_i depends only on the inputs and on the weights of the network. The learning carries out this optimization iteratively, by iterating through all the weights for a given $()\iota, Y)$, and repeating this process for all the $(\iota, Y) \in D$.

After the initialization we maintain the equality $W_{i,j} = w^+_{i,j} + w^-_{i,j} = 2w$ so that we only need to compute each $w^+_{i,j}$ using the gradient iterations for k and each pair of neurons (u, v)::

$$w^{+,(k+1)}_{u,v} = w^{+,(k)}_{u,v} - \eta \frac{\partial C}{\partial w^+_{u,v}}|_{W=W^k,(\iota,Y)}. \tag{7}$$

where $\eta > 0$ is known as the learning rate. The details of the computational algorithm can be found in [22].

4 Experimental Results and Conclusions

We use the "10-best features version" of the Bot-IoT dataset [47] and limit the number of samples to 1177. To generate these features, the authors of [47] used data gathered in pcap files, and applied Correlation Coefficient and Entropy techniques to chose the best training features. Our testing dataset includes 589 samples with 350 attack instances and 239 non-attack instances.

The learning algorithm is implemented using Python, and the RNN used in the experiments consists of $N = 12$ neurons, of which 10 them receive external signals, and we have one output neuron. We also tried using a minimal network with eleven neurons, as well as a larger one, but achieved the best results using twelve neurons. When the number of neurons was smaller, the results were worse, and for bigger topologies the results did not improve. The weight updates are carried out after

determining the output for every individual input data idem sample, in contrast to batch learning.

In Table 1 we show the precision that is obtained with Random Intitalization, as compared to the initialization with "neutral weights", and thirdly the advantage of also using the additional feature of just learning the excitatory weighs and maintaining the inhibitory weights so that the sum of the two remain constant (Weight Restriction or WR). We clearly see that the third approach is the best, both for training accuracy and for recall (testing) accuracy.

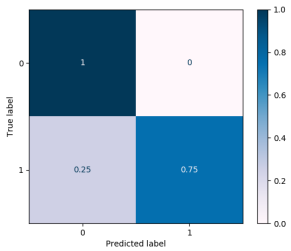
The computation times presented in Table 2 are the average time values for training the network. Introducing the "neutral" initialization significantly speeds up the learning time from the $488.48sec$ to $166.16ses$ on average. When only the excitatory weights need to be learned, the learning time drops to $100.78sec$. Thus the most accurate learning approach is also the fastest.

In Figure 2 we present the evaluation (on the training and validation datasets) during the learning process. In these figures, the "number of iterations" is the number of samples that were used in the training. The accuracy curves with neutral initialization have the desired shape - the increase in the accuracy is visible almost from the beginning of the training. As can be seen in Figure 2a it took more that 100 iterations to initiate the increase in the accuracy. After this slow start, a rapid increase occurs and then it stops and no further increase in the accuracy can be observed. Introducing WR optimizes the learning process not only by limiting the number of calculations needed but also by decreasing the number of iterations needed to achieve results comparable with those of the RNN without it. Also, it can be observed that the RNN with WR is also much better at generalization.

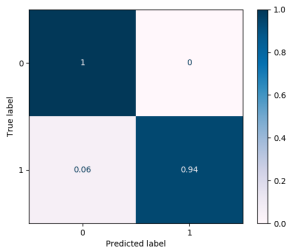
In Figure 1 a graphical representation of the results is shown in the form of confusion matrices. It is clearly seen that using 'neutral' initialization we can also improve the RNN's accuracy for attack detection.

Table 1: Comparison of Accuracy, Precision and Recall for the best models achieved with the two weight initializations

Random		**'Neutral'**		**'Neutral'+WR**	
Train	**Valid**	**Train**	**Valid**	**Train**	**Valid**
Accuracy					
86.29	83.71	96.90	96.09	96.80	96.09
Precision					
100.0	100.0	99.79	100.0	99.48	100.0
Recall					
77.83	75.0	94.80	94.00	95.33	94.00



(a) Random initialization



(b) Neutral initialization (both for Weight Restriction applied and without it)

Fig. 1: Confusion matrices of the networks with different types of weight handling approaches implemented after the stabilization of the learning process

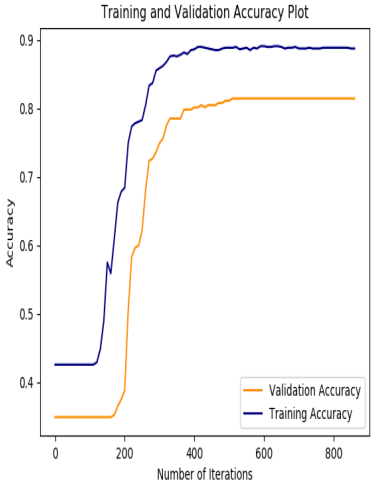
Table 2: Comparison of average execution times [s] for the models achieved with different types of weight handling implemented

Random	**'Neutral'**	**'Neutral'+WR**
488.48	166.16	100.78

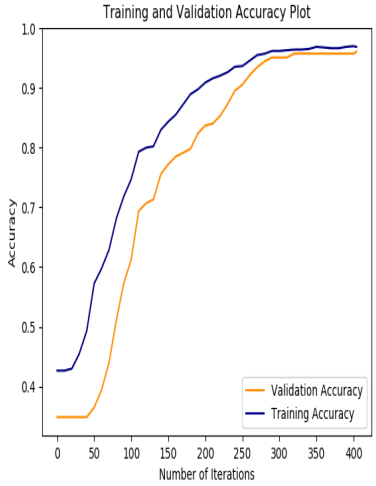
In future work we will examine the design of attack detectors that can identify multiple forms of attacks simultaneously. This is a a very challenging task that needs to be addressed with more sophisticated techniques include the use of multiple simultaneously operating neural networks.
We also plan to integrate these attack detection techniques in our existing test-bed which has been reported recently [**?**], so that we may evaluate the capacity of an integrated system not only to detect attacks, but also to react in a manner which mitigates or eliminates their effect.

References

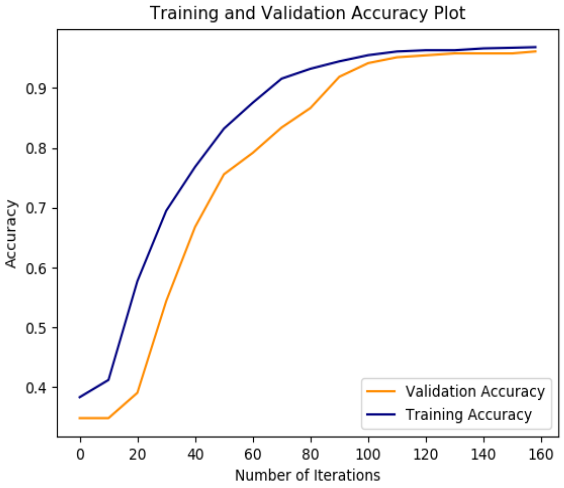
1. Adeel, A., Larijani, H., Ahmadinia, A.: Random neural network based cognitive engines for adaptive modulation and coding in LTE downlink systems. Comput. Electr. Eng. **57**, 336–350 (2017), `https://doi.org/10.1016/j.compeleceng.2016.11.005`
2. Ahmad, J., Larijani, H., Emmanuel, R., Mannion, M., Javed, A., Phillipson, M.: Energy demand prediction through novel random neural network predictor for large non-domestic buildings. In: 2017 Annual IEEE International Systems Conference, SysCon 2017, Montreal, QC, Canada, April 24-27, 2017. pp. 1–6. IEEE (2017), `https://doi.org/10.1109/SYSCON.2017.7934803`



(a) Random initialization



(b) Neutral initialization



(c) Neutral initialization. + WR

Fig. 2: Accuracy plots for the networks networks with different types of weight handling implemented after the stabilization of the learning process

3. Ahmad, J., Tahir, A., Larijani, H., Ahmed, F., Shah, S.A., Hall, A.J., Buchanan, W.J.: Energy demand forecasting of buildings using random neural networks. J. Intell. Fuzzy Syst. **38**(4), 4753–4765 (2020), https://doi.org/10.3233/JIFS-191458
4. Aldribi, A., Traoré, I., Moa, B., Nwamuo, O.: Hypervisor-based cloud intrusion detection through online multivariate statistical change tracking. Computers & Security **88**, 101646 (2020)
5. Atalay, V., Gelenbe, E.: Parallel Algorithm for Colour Texture Generation Using the Random Neural Network Model, p. 227–236. World Scientific Publishing Co., Inc., USA (1992)
6. Atalay, V., Gelenbe, E., Yalabik, N.: The random neural network model for texture generation. International Journal of Pattern Recognition and Artificial Intelligence **6**(1), 131–141 (1992)
7. Basterrech, S., Mohamed, S., Rubino, G., Soliman, M.A.: Levenberg-marquardt training algorithms for random neural networks. Comput. J. **54**(1), 125–135 (2011), https://doi.org/10.1093/comjnl/bxp101
8. Bezerra, V.H., da Costa, V.G.T., Martins, R.A., Junior, S.B., Miani, R.S., Zarpelao, B.B.: Providing IoT host-based datasets for intrusion detection research. In: Anais do XVIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais. pp. 15–28 (2018)
9. Brandon, A., Seekins, M., Joshua, B.V., Samuel, C., Haller, J.: Network data analysis to support risk management in an IoT environment. In: 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON). pp. 0063–0068 (2019)
10. Brun, O., Yin, Y., Gelenbe, E.: Deep learning with dense random neural network for detecting attacks against iot-connected home environments. Procedia Computer Science **134**, 458–463 (2018), https://doi.org/10.1016/j.procs.2018.07.183
11. Cisco 2018 Annual Cybersecurity Report. [online] (Cisco 2018), available: https://www.cisco.com/c/dam/en_us/about/annual-report/2018-annual-report-full.pdf [Accessed: 2020-06-26]
12. Cortés, F.M., Gómez, N.G.: A hybrid alarm management strategy in signature-based intrusion detection systems. In: 2019 IEEE Colombian Conference on Communications and Computing (COLCOM). pp. 1–6 (2019)
13. Cramer, C.E., Gelenbe, E.: Video quality and traffic qos in learning-based subsampled and receiver-interpolated video sequences. IEEE Journal on Selected Areas in Communications **18**(2), 150–167 (2000)
14. Cybenko, G.: Approximations by superpositions of sigmoidal functions. Mathematics of Control, Signals, and Systems **2**(4), 303–314 (1989). https://doi.org/10.1007/BF02551274
15. 1998 DARPA Intrusion Detection Evaluation Data Set. MIT Lincoln Laboratory. [online] (1998), available: https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset [Accessed: 2020-06-26]

16. 1999 DARPA Intrusion Detection Evaluation Data Set. MIT Lincoln Laboratory. [online] (1999), available: https://www.ll.mit.edu/r-d/datasets/1999-darpa-intrusion-detection-evaluation-dataset [Accessed: 2020-06-26]
17. Dobson, et al., S.: A survey of autonomic communications. ACM Transactions on Autonomous and Adaptive Systems (TAAS) **1**(2), 223–259 (2006)
18. Ferrag, M.A., Maglaras, L.: Deepcoin: A novel deep learning and blockchain-based energy exchange framework for smart grids. IEEE Transactions on Engineering Management (2019)
19. Galeano-Brajones, J., Cortés-Polo, D., Valenzuela-Valdés, J.F., Mora, A.M., Carmona-Murillo, J.: Detection and mitigation of dos attacks in sdn. an experimental approach. In: 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS). pp. 575–580. IEEE (2019)
20. Ge, M., Fu, X., Syed, N., Baig, Z., Teo, G., Robles-Kelly, A.: Deep learning-based intrusion detection for IoT networks. In: 2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC). pp. 256–25609 (2019)
21. Gelenbe, E.: Random neural networks with negative and positive signals and product form solution. Neural Computation **1**(4), 502–510 (1989)
22. Gelenbe, E.: Learning in the recurrent random neural network. Neural Computation **5**, 154–164 (1993)
23. Gelenbe, E., Batty, F.: Minimum graph vertex covering with the random neural network. In: Computer Science and Operations Research, pp. 139–147. Pergamon, Amsterdam (1992)
24. Gelenbe, E., Feng, Y., Krishnan, K.R.R.: Neural network methods for volumetric magnetic resonance imaging of the human brain. Proceedings of the IEEE **84**(10), 1488–1496 (1996)
25. Gelenbe, E., Hussain, K.F., Kaptan, V.: Simulating autonomous agents in augmented reality. Journal of Syst. and Softw. **74**(3), 255–268 (Feb 2005)
26. Gelenbe, E., Koubi, V., Pekergin, F.: Dynamical random neural network approach to the traveling salesman problem. In: Proceedings of IEEE Systems Man and Cybernetics Conference. vol. 2, pp. 630–635 (1993)
27. Gelenbe, E., Mao, Z.H., Li, Y.: Function approximation with spiked random networks. IEEE Transactions on Neural Networks **10**(1), 3–9 (1999)
28. Gelenbe, E., Mao, Z.H., Li, Y.: Function approximation by random neural networks with a bounded number of layers. Journal of Differential Equations and Dynamical Systems **12**(1-2), 143–170 (2004)
29. Gelenbe, E.: Steps toward self-aware networks. Communications of the ACM **52**(7), 66–75 (2009)
30. Gelenbe, E., Hussain, K.F.: Learning in the multiple class random neural network. IEEE Transactions on Neural Networks **13**(6), 1257–1267 (2002)

31. Gelenbe, E., Kadioglu, Y.M.: Energy life-time of wireless nodes with network attacks and mitigation. In: 2018 IEEE International Conference on Communications Workshops, ICC Workshops 2018, Kansas City, MO, USA, May 20-24, 2018. pp. 1–6. IEEE (2018), https://doi.org/10.1109/ICCW.2018.8403561
32. Ghalut, T., Larijani, H.: Non-intrusive method for video quality prediction over LTE using random neural networks (RNN). In: 9th International Symposium on Communication Systems, Networks & Digital Signal Processing, CSNDSP 2014, Manchester, UK, July 23-25, 2014. pp. 519–524. IEEE (2014), https://doi.org/10.1109/CSNDSP.2014.6923884
33. Ghalut, T., Larijani, H.: Content-aware and QOE optimization of video stream scheduling over LTE networks using genetic algorithms and random neural networks. J. Ubiquitous Syst. Pervasive Networks **9**(2), 21–33 (2018), https://doi.org/10.5383/JUSPN.09.02.003
34. Guerra-Manzanares, A., Medina-Galindo, J., Bahsi, H., Nõmm, S.: MedBIoT: Generation of an IoT botnet dataset in a medium-sized IoT network. In: 6th International Conference on Information Systems Security and Privacy. pp. 207–218 (2020)
35. Guizani, N., Ghafoor, A.: A network function virtualization system for detecting malware in large IoT based networks. IEEE Journal on Selected Areas in Communications **38**(6), 1218–1228 (2020)
36. Hamdi, M., Abie, H.: Game-based adaptive security in the internet of things for ehealth. In: 2014 IEEE International Conference on Communications (ICC). pp. 920–925 (2014)
37. Hanif, S., Ilyas, T., Zeeshan, M.: Intrusion detection in iot using artificial neural networks on unsw-15 dataset. In: 2019 IEEE 16th International Conference on Smart Cities: Improving Quality of Life Using ICT & IoT and AI (HONET-ICT). pp. 152–156 (2019)
38. Hettich, S., Bay, S.: The UCI KDD Archive. Irvine, CA: University of California, Department of Information and Computer Science. [online] (1999), available: http://kdd.ics.uci.edu [Accessed: 2020-06-26]
39. Hindy, H., Brosset, D., Bayne, E., Seeam, A., Tachtatzis, C., Atkinson, R., Bellekens, X.: A taxonomy and survey of intrusion detection system design techniques, network threats and datasets. arXiv preprint arXiv:1806.03517 (2018)
40. Hussain, K.F., Kaptan, V.: Modeling and simulation with augmented reality. Int. J. Oper. Res. **38**(2), 89–103 (April 2004)
41. Hussain, K.F., Moussa, G.S.: On road vehicle classification based on random neural network and bag of visual words. Probability in the Engineering and Informational Sciences **30**, 403–412 (2016). https://doi.org/doi:10.1017/S0269964816000073
42. Hussain, K.F., Radwan, E., Moussa, G.S.: Augmented reality experiment: Drivers' behavior at an unsignalized intersection. IEEE Transactions on Intelligent Transportation Systems **14**(2), 608–617 (2013)
43. Javed, A., Larijani, H., Ahmadinia, A., Emmanuel, R., Mannion, M., Gibson, D.: Design and implementation of a cloud enabled random neural network-based decentralized smart controller with intel-

ligent sensor nodes for HVAC. IEEE Internet Things J. **4**(2), 393–403 (2017), https://doi.org/10.1109/JIOT.2016.2627403
44. Javed, A., Larijani, H., Ahmadinia, A., Gibson, D.: Smart random neural network controller for HVAC using cloud computing technology. IEEE Trans. Ind. Informatics **13**(1), 351–360 (2017), https://doi.org/10.1109/TII.2016.2597746
45. Jeatrakul, P., Wong, K.W.: Comparing the performance of different neural networks for binary classification problems. In: 2009 Eighth International Symposium on Natural Language Processing. pp. 111–115 (2009)
46. Kang, H., Ahn, D.H., Lee, G.M., Yoo, J.D., Park, K.H., Kim, H.K.: IoT network intrusion dataset (2019). https://doi.org/10.21227/q70p-q449, http://dx.doi.org/10.21227/q70p-q449
47. Koroniotis, N., Moustafa, N., Sitnikova, E., Turnbull, B.: Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset. Future Generation Computer Systems **100**, 779–796 (2019)
48. Li, W., Tug, S., Meng, W., Wang, Y.: Designing collaborative blockchained signature-based intrusion detection in iot environments. Future Generation Computer Systems **96**, 481–489 (2019)
49. Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Shabtai, A., Breitenbacher, D., Elovici, Y.: N-baiot—network-based detection of IoT botnet attacks using deep autoencoders. IEEE Pervasive Computing **17**(3), 12–22 (2018)
50. Öke, G., Loukas, G.: A denial of service detector based on maximum likelihood detection and the random neural network. The Computer Journal **50**(6), 717–727 (2007)
51. Ostinato Traffic Generator for Network Engineers. [online], available: https://ostinato.org/ [Accessed: 2020-09-08]
52. Qureshi, A., Larijani, H., Ahmad, J., Mtetwa, N.: A novel random neural network based approach for intrusion detection systems. In: 2018 10th Computer Science and Electronic Engineering Conference, CEEC 2018, University of Essex, Colchester, UK, September 19-21, 2018. pp. 50–55. IEEE (2018), https://doi.org/10.1109/CEEC.2018.8674228
53. Radhakrishnan, K., Larijani, H.: Evaluating perceived voice quality on packet networks using different random neural network architectures. Perform. Evaluation **68**(4), 347–360 (2011), https://doi.org/10.1016/j.peva.2011.01.001
54. Rubino, G., Tirilly, P., Varela, M.: Evaluating users' satisfaction in packet networks using random neural networks. In: Kollias, S.D., Stafylopatis, A., Duch, W., Oja, E. (eds.) Artificial Neural Networks - ICANN 2006, 16th International Conference, Athens, Greece, September 10-14, 2006. Proceedings, Part I. Lecture Notes in Computer Science, vol. 4131, pp. 303–312. Springer (2006), https://doi.org/10.1007/11840817_32
55. Saeed, A., Ahmadinia, A., Javed, A., Larijani, H.: Intelligent intrusion detection in low-power IoTs. ACM Transactions on Internet Technology (TOIT) **16**(4), 1–25 (2016)

56. Shon, T., Kovah, X., Moon, J.: Applying genetic algorithm for classifying anomalous tcp/ip packets. Neurocomputing **69**(16), 2429 – 2433 (2006)
57. Timotheou, S.: The random neural network: A survey. The Computer Journal **53**(3), 251–267 (2010)
58. Timotheou, S.: A novel weight initialization method for the random neural network. Neurocomputing **73**(1-3), 160–168 (2009)
59. Xiang, C., Yong, P.C., Meng, L.S.: Design of multiple-level hybrid classifier for intrusion detection system using bayesian clustering and decision trees. Pattern Recognition Letters **29**(7), 918 – 924 (2008)
60. 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)
61. Yin, Y.: Deep learning with the random neural network and its applications. ArXiv **abs/1810.08653** (2018)