Article

JOURNAL OF UNIVERSITY OF BABYLON
For Pure and Applied Sciences (JUBPAS)

Vol.32; No.1.| 2024

Hyper-Parameters Consequence on the Performance of Deep Learning Algorithm for Intrusion Detection System

Hadeel Qasem Gheni [1*], Wathiq L. Al-Yaseen [2]

[1]Department of Software, Information Technology College, University of Babylon, wsci.hadeel.qasem@uobabylon.edu.iq, Babylon, Iraq.

[2]Karbala Technical Institute, Al-Furat Al-Awsat Technical University, wathiq@atu.edu.iq, Karbala, Iraq

***Corresponding author email:** wsci.hadeel.qasem@uobabylon.edu.iq; mobile: 07729199917

تأثير المعلمات الفائقة على أداء خوارزمية التعلم العميق لنظام كشف التسلل

هديل قاسم غني [1*], واثق لفتة الياسين [2]

1 كلية تكنولوجيا المعلومات، جامعة بابل، wsci.hadeel.qasem@uobabylon.edu.iq ، بابل، العراق

2 معهد كربلاء التقني، جامعة الفرات الأوسط التقنية، wathiq@atu.edu.iq ، كربلاء، العراق

Accepted: 11/2/2024 **Published:** 31/3/2024

ABSTRACT

Background: The increasing reliance on Internet-based applications has led to a rise in the number of hackers, posing a significant threat to the security and confidentiality of digital resources. The performance of the Deep Learning (DL) model is greatly impacted by the design of DL architectures, which typically need expert knowledge. The intrusion detection results can be enhanced by the model by adjusting certain hyper-parameters. The right selection for the best hyper-parameters of the model helps to improve the detection performance.

Materials and Methods: This paper studied the effect of some hyper-parameters used in the classification algorithm for improving the performance of intrusion detection models. Restricted Boltzmann Machine (RBM) and Multilayer Perceptron (MLP) are used to identify the best values in terms of the number of hidden nodes, the number of epochs, and the size of batches used to train the UNSW-NB15 dataset to enhance the accuracy and reduce the implementation time for detecting the attack.

Results: Different numbers of hidden neurons, epochs, and batch size were used to prove which hyper-parameter had the most effect on the performance in terms of accuracy and implementation time to detect attacks.

Conclusion: Building a model and refining it with appropriate hyper-parameter values leads to better design of the model with high performance. The hyper-parameters must be chosen in a way that enhances the performance of the model in terms of increasing accuracy and reducing the implementation time.

Keywords: Intrusion Detection System, RBM, MLP, UNSW-NB15, Parameters Efficacy, Deep Learning.

ISSN: 2312-8135 | Print ISSN: 1992-0652

info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

مجلة جامعة بابل للعلوم الصرفة والتطبيقية مجلة جامعة بابل للعلوم الصرفة والتطبيقية

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

INTRODUCTION

The uses of Internet-based applications are flourishing due to the dependence of daily human activities on the Internet and computer networks [1]. This leads to a steadily growing number of hackers [2], and thus, threatens the safety and confidentiality of digital resources [3]. Any attempts to enter a network without authorization and obtain unauthorized data that endangers network security are known as intrusions.

All possible security risks must be located in order to develop an efficient network protection plan, and the best collection of technologies to defend against those threats must then be chosen [4]. It is not possible to prevent all attempts that aim to exploit the security gaps in networks and systems.

One of the very influential factors in the learning process is parameter and hyper-parameter [5]. Parameters (such as weights and biases) are internal to the model, either estimated or learned from training data, whereas hyper-parameters (such as the number of hidden layers, the number of hidden neurons in each layer, the activation function, the learning rate, the number of epoch, and batch size) are considered external to the model since the model is unable to alter its values while being trained or learned.

A hyper-parameter is a variable in machine learning and deep learning that is set before training and remains unchanged at the end of the training process, as well as used to define or represent the model [6]. The values of hyper-parameters govern the learning process and establish the model parameter values that a learning algorithm ultimately learns.

As a result, choosing the appropriate hyper-parameter values is crucial since doing so will have a direct effect on the model's performance after it is trained [7].

The extraction of valuable patterns and models from large datasets is known as Data Mining which extracts features from network traffic and is used to differentiate between genuine traffic and malicious traffic [8]. Since it was created primarily for representational learning and classification, the Restricted Boltzmann Machine (RBM) has recently piqued the interest of the artificial intelligence and machine learning field [9].

The remaining parts of the essay are arranged as follows. Section 2 went over the IDS challenges. The material and methods are presented in Section 3. Section 4 displays the proposed model. In Section 5, the results and discussions. In Section 6, the conclusion is introduced.

THE CHALLENGES OF INTRUSION DETECTION SYSTEM

Despite the extensive research on IDSs, there are still many crucial issues to be resolved [10]. Intrusion detection systems suffer from a number of challenges, including low detection rates, false alarm rates, response time, and unbalanced datasets [11].

IDS generates a lot of false positives (identify a normal action as harmful) and false negatives (miss a malicious activity), which can be difficult for network managers to manage, and thus, may

become less trusting of the alerts as a result, their defense levels may drop, or they may have too much work to do to identify actual assaults [12]. The nature of the intrusion datasets is rather unbalanced [13]. The classification algorithms are easily skewed in favor of the dominant class due to the disparity of class variables [14].

Response time is the time required for the system to react to a specific event. In intrusion detection, it is the time required for the system to issue an alert when malicious activity is detected. Extensive times of responses to user requests are associated with higher attack rates [15], the classifier with a quick response time is preferred over the classifier with a slow response time [16].

The process of creating a dataset suitable for testing intrusion detection systems is expensive and requires a lot of accuracy and time, as it is necessary to adjust the real work environment to explore all possibilities of attack. It is challenging to evaluate, compare, and apply a system capable of detecting new attacks in the field of network intrusion detection, especially when anomaly-based intrusion detection is used. It is imperative that these systems are tested and evaluated before they are implemented in any real environment, using real addressable traffic with a comprehensive set of attacks. This is a major challenge given the scarcity of such data sets.

MATERIALS AND METHODS

Intrusion Detection Dataset

The classifier will make mistakes and lose accuracy since the previous datasets did not contain any contemporary attacks and the distribution of benchmark training and testing datasets varies with respect to the types of data. As time passes, stealth and espionage attacks become increasingly commonplace activities [17]. Consequently, in 2015, a new dataset known as UNSW_NB15 was created by combining a range of typical network traffic with recent attack occurrences in an artificial environment at the University of New South Wales Cyber Security Lab. [18].

The UNSW_NB15 dataset contains genuine activities of typical traffic in addition to nine classifications of contemporary attack types and 49 features composed of these various categories [19]. The nine categories of attacks on networks in the UNSW-NB15 dataset are DoS, Shellcode, Backdoor, Fuzzers, Reconnaissance, Worms, Analysis, Exploits, and Generic [20].

Two datasets, one for training and the other for testing, are included in the UNSW_NB15 dataset. There are 175,342 connection records in the training dataset and 82,332 connection records in the testing dataset. Table 1 displays the various attack types and how they are distributed across training and testing sets in the UNSW_NB15.

Overall, out of 49 features, there are 28 features as integer type, 10 features as float type, 6 features as nominal type, 3 features as binary type, and 2 features as timestamp type.

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

Table 1. Primary Classes for UNSW_NB15 Dataset [21]

Classes	Training Dataset		Testing Dataset	
	No. Patterns	**Per.%**	**No. Patterns**	**Per.%**
DoS	12,264	6.994%	4,089	4.966%
Backdoor	1,746	0.996%	583	0.708%
Analysis	2,000	1.141%	677	0.822%
Fuzzers	18,184	10.371%	6,062	7.363%
Generic	40,000	22.813%	18,871	22.921%
Exploits	33,393	19.045%	11,132	13.521%
Reconnaissance	10,491	5.983%	3,496	4.246%
Shell Code	1,133	0.646%	378	0.459%
Worms	130	0.074%	44	0.053%
Normal	56,000	31.938%	37,000	44.940%
Total	**175,341**	**100%**	**82,332**	**100%**

Restricted Boltzmann Machine

The visible layer and hidden layer comprise the two layers of the stochastic network of neurons that is known as the Restricted Boltzmann Machine (RBM) [22], While the hidden layer attempts to extract features from the visible layer in order to reflect a probabilistic distribution of the data, the visible layer displays the data that has been gathered [23].

The foundation of deep belief networks is RBM, which is a two-layer, shallow neural net [24]. The visible, or input, layer is the first layer of the RBM, and the hidden layer is the second. Nodes from different layers are connected to each other, but no two nodes from the same layer are linked, this is the reason why the network is referred to as restricted. The information can move in both directions due to the symmetric and bidirectional connections between the levels [25]. The RBM architecture is shown in Figure 1.

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

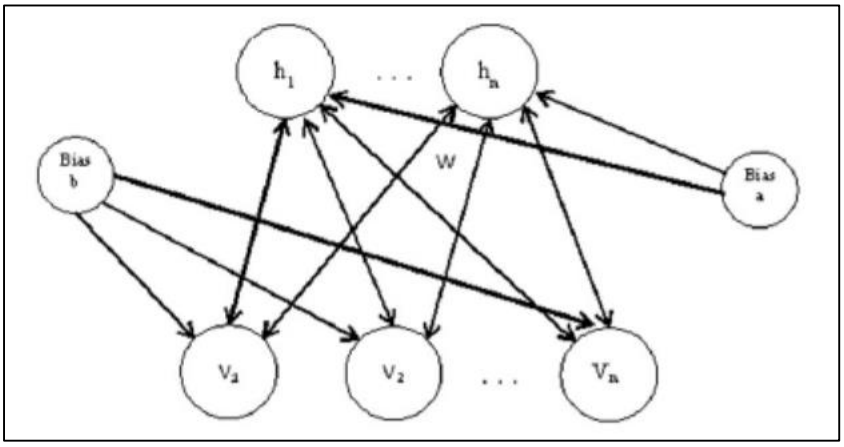

Figure (1): Restricted Boltzmann Machine [21]

To generate the node's output, each input V is multiplied by a separate weight W. The resulting products are then summed, added to a bias, and then fed to an activation function, the outcome reaches the hidden layers. Individual weights and an overall bias are merged with backward-passed data to reconstruct the input when data reaches the visible layers, as explained in Equation (1) and Equation (2) [26], respectively.

$$P(h_j = 1|v) = \sigma(c_j + \sum_i w_{ij} v_i) \quad (1)$$

$$P(v_j = 1|h) = \sigma(b_j + \sum_i w_{ij} h_i) \quad (2)$$

Where: v is the visible neuron, h is the hidden neuron, b is the bias for the visible neurons, c is the bias for the hidden neurons, and w is the weight.

Multilayer Perceptron Algorithm

MLP is a forward-moving neural network that feeds information from the input layer to the output layer [27]. The fundamental framework of an artificial neural network (ANN) is made up of the three layers of MLP [28]. The data to be processed must first be received by the input layer, which then forwards it to the hidden layer, which carries out the actual computations, and finally to the output layer, which provides the desired outcome for a prediction or classification [29]. Each layer's neurons are connected to one another by weights, and a bias is used to set the threshold at which a neuron will become active [30]. MLPs are intended to approximate any continuous function, and they can be used to tackle problems that cannot be solved linearly [31]. A basic MLP architecture is depicted in Figure 2.

Before the training starts, the weights are randomly assigned. The input data (X1, X2) and the predicted result (Y) make up the training set of data. The following equation is used to calculate the output:

$$Y=WX+B \quad (3)$$

where W is the weight and B is the bias.

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

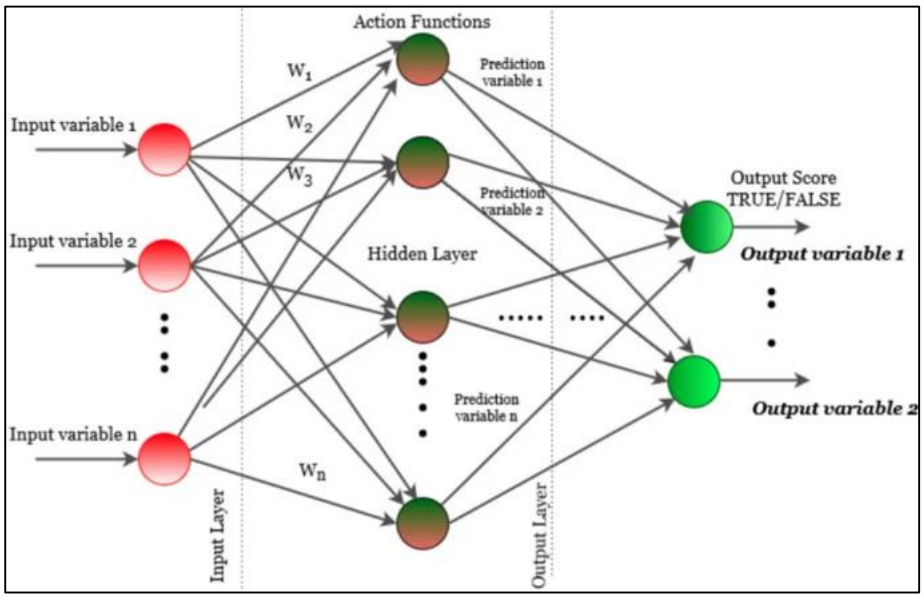

Figure (2): MLP architecture [32]

The Proposed Model

This paper aims to measure the extent to which the hyper-performance of an intrusion detection system is affected when changing the hyper-parameters that make up the classification algorithm, especially the number of neurons in the hidden layer. For this task, we chose the RBM and the MLP algorithms.

The UNSW-NB15 training and testing datasets go through several steps to prepare them for subsequent stages. First, both datasets will undergo the preprocessing stage, which includes examining the data to ensure that it is free of missing values, then, converting the categorical values into numerical values by coding process so that the algorithm can handle these values, and finally making the values in the same scale (between 0 and 1) through the min-max normalization equation as follows.

$$V_{new} = (V - V_{min}) / (V_{max} - V_{min}) \quad (3)$$

Where: V_{new} is the new value after scaling, V is the present value, V_{min} is the min value in the feature, and V_{max} is the max value in the feature.

Both datasets will enter the preprocessing stage, then the training dataset will enter the stage of changing the hyper-parameters and training by the RBM and MLP algorithms, and finally, it will be tested using the test dataset. The obtained accuracy and required execution time will be calculated. Figure 3 explains the proposed model.

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

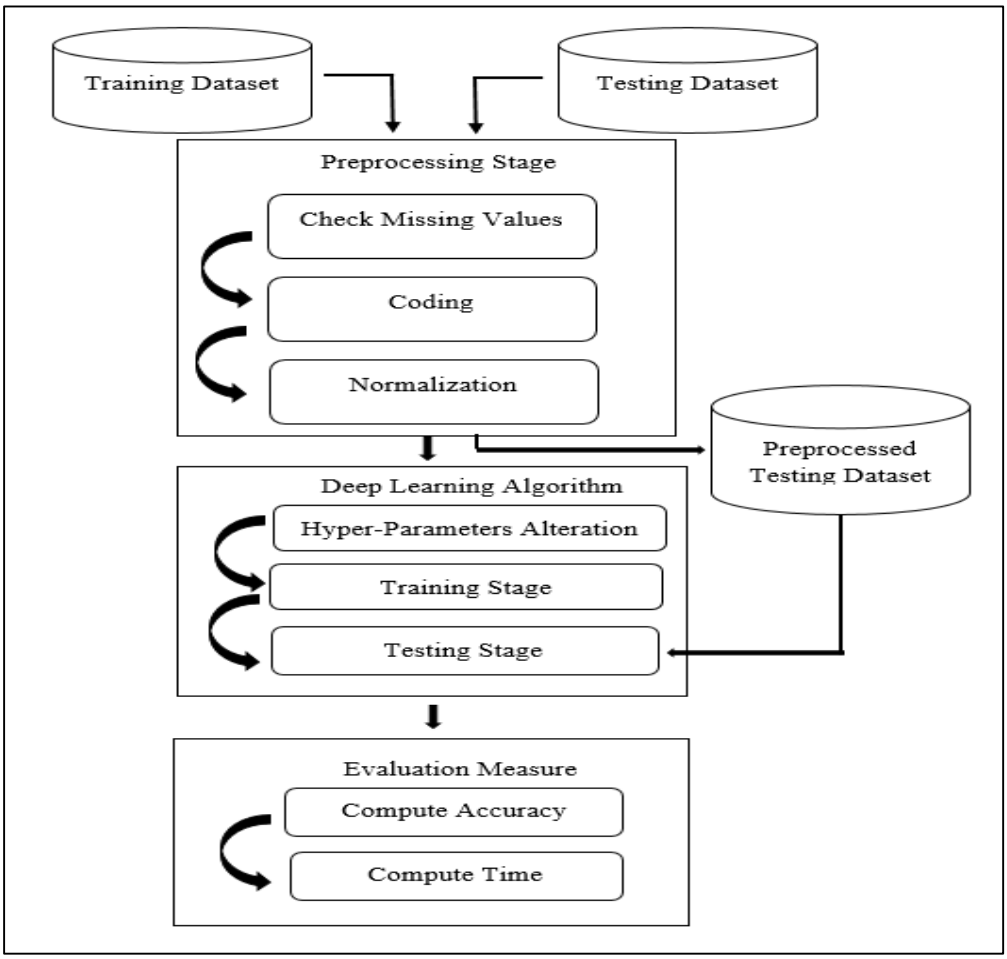

Figure (3): The Proposed System Architecture

RESULTS AND DISCUSSION

First of all, the primary hyper-parameters used in the classification algorithm must be defined, including epoch, batch size, and hidden neurons. The number of epochs is a gradient descent hyper-parameter that regulates how many full iterations are across the training dataset [33]. The batch size is a gradient descent hyper-parameter that regulates how much training data must be processed before the internal model parameters are changed [34]. Input, output, and hidden layers with hidden neurons comprise the typical architecture of a neural network, where Determining the optimal number of hidden neurons necessitates trial and error during training in order to estimate the generalization error [35]. Problems related to overfitting or underfitting may arise from the random selection of the number of hidden neurons [36].

Not every one of the UNSW-NB15 features, nevertheless, is necessary for the target labels. The existence of total duration (feature named: dur) renders two input features, Start time and Last time, superfluous. Also, some features lack the necessary data to detect intrusions, so they eliminated such as source-IP-address, source-port-number, destination-IP-address, and destination-port-number. Two target features are found, one containing only 0,1 representing

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

normal and attack and used for binary classification, and the other containing the types of attack and used for multi-classification.

To train the UNSW-NB15 dataset, the RBM and MLP algorithms are used, which consists of 42 input neurons equal to the number of features in the dataset and a different number of hidden neurons that will control the accuracy of the system (Multiples of 2 to 512), the number of epoch used are 1, 10, 100, the batch size is 32, 64, the learning rate is 0.1, the weights are randomly selected, and the sigmoid is the activation function of the model.

The primary goal of the paper is to explain how choosing different numbers of hidden neurons with epoch number, and batch size will affect the model's performance. Table 2 and Table 3 explain the accuracy and time of the system that is obtained when using different numbers of hidden neurons, taking into account the number of epochs and batch size for the binary classification and the multi-classification with the RBM algorithm, respectively, while Table 4 and Table 5 explain the accuracy with the MLP algorithm, respectively.

Table 2. Hyper-parameters Effect on Accuracy and Implementation Time for Binary Classification with RBM Algorithm

No. Hidden Neuron	Epoch 1				Epoch 10				Epoch 100			
	Batch 32		Batch 64		Batch 32		Batch 64		Batch 32		Batch 64	
	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s
2	62.39	0.010	62.39	0.009	56.24	0.010	56.27	0.013	56.23	0.009	56.20	0.035
4	64.20	0.011	73.24	0.012	69.5	0.014	61.46	0.015	56.23	0.015	56.29	0.013
8	77.99	0.018	76.25	0.019	78.30	0.025	**78.76**	**0.026**	56.12	0.018	61.56	0.024
16	78.52	0.034	77.97	0.032	78.10	0.035	78.53	0.033	61.74	0.040	63.48	0.053
32	78.53	0.063	78.53	0.050	78.54	0.056	78.61	0.053	61.35	0.058	61.73	0.072
64	78.53	0.085	78.53	0.087	78.45	0.096	78.14	0.105	61.66	0.109	77.48	0.114
128	78.51	0.172	78.53	0.162	77.99	0.173	78.03	0.218	77.02	0.202	75.98	0.215
256	78.52	0.305	78.53	0.332	78.64	0.346	78.55	0.383	73.97	0.410	75.51	0.441
512	78.54	0.603	78.52	0.750	78.60	0.811	78.53	0.738	78.49	0.744	78.61	0.935

Table 3. Hyper-parameters Effect on Accuracy and Implementation Time for Multi-Classification with RBM Algorithm

No. Hidden Neuron	Epoch 1				Epoch 10				Epoch 100			
	Batch 32		Batch 64		Batch 32		Batch 64		Batch 32		Batch 64	
	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s
2	48.32	0.036	50.93	0.023	55.28	0.021	55.30	0.020	55.28	0.021	55.28	0.023
4	61.97	0.024	56.82	0.021	62.88	0.026	62.80	0.033	60.86	0.024	57.29	0.028
8	59.34	0.028	55.38	0.035	63.06	0.034	**65.09**	**0.035**	60.86	0.033	60.89	0.042
16	55.75	0.044	55.19	0.048	59.97	0.055	59.11	0.049	60.87	0.055	61.97	0.066
32	54.44	0.074	53.90	0.080	60.31	0.071	54.84	0.078	61.72	0.099	64.24	0.087
64	55.51	0.119	54.40	0.117	60.34	0.131	57.33	0.127	62.08	0.148	62.44	0.128
128	54.94	0.218	54.97	0.217	55.60	0.237	56.52	0.229	62.50	0.255	62.57	0.244
256	55.62	0.405	54.03	0.388	56.64	0.453	57.13	0.459	61.41	0.500	58.43	0.426
512	56.24	0.701	55.61	0.694	55.71	0.909	55.28	0.846	64.54	0.822	57.27	0.786

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

Table 4. Hyper-parameters Effect on Accuracy and Implementation Time for Binary Classification with MLP Algorithm

No. Hidden Neuron	Epoch 1				Epoch 10				Epoch 100			
	Batch 32		Batch 64		Batch 32		Batch 64		Batch 32		Batch 64	
	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s
2	78.94	0.008	74.66	0.008	78.64	0.008	79.38	0.009	78.52	0.010	79.16	0.011
4	79.65	0.010	80.33	0.010	80.09	0.012	79.50	0.012	81.19	0.010	76.55	0.012
8	80.10	0.013	80.55	0.016	81.17	0.014	80.01	0.013	78.95	0.016	78.78	0.014
16	80.20	0.022	79.01	0.20	80.75	0.023	80.59	0.026	81.35	0.023	82.31	0.023
32	81.24	0.028	80.60	0.046	83.43	0.031	80.66	0.030	86.55	0.030	80.71	0.030
64	81.39	0.043	81.81	0.065	80.81	0.050	81.47	0.048	82.22	0.044	81.30	0.045
128	80.46	0.072	81.06	0.085	85.07	0.077	85.01	0.076	81.37	0.079	82.60	0.083
256	81.19	0.133	81.90	0.132	87.54	0.160	82.74	0.129	**88.12**	**0.150**	84.14	0.139
512	82.13	0.273	82.83	0.241	84.86	0.304	85.66	0.276	83.01	0.261	83.68	0.244

Table 5. Hyper-parameters Effect on Accuracy and Implementation Time for Multi Classification with MLP Algorithm

No. Hidden Neuron	Epoch 1				Epoch 10				Epoch 100			
	Batch 32		Batch 64		Batch 32		Batch 64		Batch 32		Batch 64	
	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s	Acc.	T/s
2	56.13	0.023	53.31	0.024	51.88	0.029	50.40	0.027	51.01	0.029	55.76	0.029
4	60.08	0.026	58.03	0.028	51.31	0.032	61.16	0.031	54.15	0.031	58.41	0.030
8	67.80	0.034	66.07	0.032	63.39	0.035	58.91	0.026	58.13	0.035	54.02	0.031
16	68.41	0.036	67.32	0.041	66.59	0.042	65.35	0.031	67.65	0.042	63.42	0.046
32	69.38	0.047	70.72	0.047	68.55	0.058	70.01	0.049	68.60	0.051	70.40	0.058
64	69.86	0.059	69.93	0.059	70.01	0.063	71.14	0.064	73.43	0.067	65.32	0.095
128	69.57	0.087	69.98	0.086	71.00	0.113	71.31	0.094	66.15	0.110	70.03	0.131
256	72.46	0.137	70.06	0.131	70.37	0.158	70.64	0.164	72.66	0.194	65.43	0.210
512	70.15	0.265	69.67	0.262	70.11	0.287	72.01	0.277	**74.01**	**0.317**	72.24	0.299

The tables above demonstrate the importance of the hyper-parameters and their impact on the model's efficiency. For example, in the case of binary classification with the RBM algorithm, we find that the accuracy increased from 56 to 78 when changing the hyper-parameters required for training, whereas, in the case of binary classification with the MLP algorithm, we find that the accuracy increased from 78 to 88. On the other hand, increasing the number of hidden neurons or increasing the number of epochs or batch size does not mean increasing the efficiency of the system.

Incorrect selection of hyper-parameters may lead to system failure, for example, increasing the number of epochs may lead to overfitting, which means that the classification algorithm gives accurate predictions for the training data, but fails to give correct predictions for new data.

The best accuracy obtained in the case of binary classification with the RBM algorithm is when the number of epochs is 10, the batch size is 64, and the number of hidden neurons is 8, as shown

in Table 2 in bold. As for multiple classification, the best accuracy can also be obtained when the hyper-parameters have the same values as in binary classification, as shown in Table 3 in bold.

For the MLP algorithm, the best accuracy obtained in the case of binary classification is when the number of epochs is 100, the batch size is 32, and the number of hidden neurons is 256, as shown in Table 4 in bold. As for multi-classification, the best accuracy can also be obtained when the number of epochs is 100, the batch size is 32, and the number of hidden neurons is 512, as shown in Table 5 in bold.

CONCLUSION

The research studied the importance of some hyper-parameters used in the classification algorithm in the behavior of the intrusion detection system. Among these hyper-parameters are the batch size, epoch numbers, and the number of hidden neurons. The hyper-parameters must be chosen in a way that enhances the performance of the system in terms of increasing accuracy and reducing the implementation time.

Building a model and refining it with appropriate hyper-parameter values leads to better design of the model with high performance, where determining the right number of hidden neurons with the least amount of error and maximum accuracy is one of the main issues in neural network architecture. Overfitting, a condition in which neural networks overestimate the difficulty of the target problem is brought on by an abundance of hidden neurons.

Using more hidden neurons may lead to an increase in the accuracy but also an increase in the time required for implementation, this is a negative factor as time is very important in the process of detecting attacks, also, in some cases, increasing the number of hidden neurons does not mean an increase in accuracy. Choosing the right number of hidden neurons, epochs, and the size of batches leads to an enhancement in the model performance, therefore, hyper-parameters should be chosen that give a balance between increasing accuracy and reducing the implementation time.

In the future, there will be studies on other classification algorithms and datasets to measure the importance of hyper-parameters on the performance of the intrusion detection system.

DATA AVAILABILITY STATEMENT

The UNSW-NB15 dataset can be found at https://research.unsw.edu.au/projects/unsw-nb15-dataset.

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

Conflict of interests.

There is no conflict interest

References

[1] M. G. Raman, N. Somu, K. Kirthivasan, R. Liscano, and V. S. Sriram, "An efficient intrusion detection system based on Hypergraph-Genetic algorithm for parameter optimization and feature selection in support vector machine", *Knowledge-Based Systems*, vol. 134, pp. 1-12, 2017. https://doi.org/10.1016/j.knosys.2017.07.005

[2] A. A. Salih, and M. B. Abdulrazaq, "Combining best features selec-tion using three classifiers in intrusion detection system.", *in 2019 International Conference on Advanced Science and Engineering (ICOASE), IEEE*, 2019. https://doi: 10.1109/ICOASE.2019.8723671

[3] N. Acharya, and S. Singh, "An IWD-based feature selection method for intrusion detection system.", *Soft Computing*, vol. 22, pp. 4407-4416, 2018. https://doi.org/10.1007/s00500-017-2635-2

[4] A. A. Ramaki, A. Rasoolzadegan, and A. G. Bafghi, "A systematic mapping study on intrusion alert analysis in intrusion detection systems.", *ACM Computing Surveys (CSUR)*, vol. 51, no.3, pp. 1-41, 2018. https://doi.org/10.1145/3184898

[5] Y. N. Kunang, S. Nurmaini, D. Stiawan, and B. Y. Suprapto, "Attack classification of an intrusion detection system using deep learning and hyperparameter optimization". *Journal of Information Security and Applications*, vol. 58, pp. 102804, 2021. https://doi.org/10.1016/j.jisa.2021.102804

[6] R. Wazirali, "An Improved Intrusion Detection System Based on KNN Hyperparameter Tuning and Cross-Validation", *Arab J Sci Eng* vol. 45, pp. 10859–10873, 2020. https://doi.org/10.1007/s13369-020-04907-7.

[7] Q. Wang, H. Jiang, J. Ren, H. Liu, X. Wang, and B. Zhang, "An intrusion detection algorithm based on joint symmetric uncertainty and hyperparameter optimized fusion neural network". *Expert Systems with Applications*, vol. 244, pp. 123014, 2024. https://doi.org/10.1016/j.eswa.2023.123014

[8] S. A. Alasadi, W. S. Bhaya, "Anomaly Detection System for Internet Traffic based on TF-IDF and BFR Clustering Algorithms.", *International Journal of Engineering & Technology*, vol. 8, pp. 131-137, 2019.

[9] N. Zhang, S. Ding, J. Zhang, and Y. Xue, "An overview on restricted Boltzmann machines.", *Neurocomputing*, vol. 275, pp. 1186-1199, 2018.

[10] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges", *Cybersecurity*, vol. 2, no.1, pp. 1-22, 2019. https://doi.org/10.1186/s42400-019-0038-7

[11] M. Aljanabi, M. A. Ismail, and A. H. Ali, "Intrusion detection systems, issues, challenges, and needs", *International Journal of Computational Intelligence Systems*, vol. 14, no.1, pp. 560-571, 2021. https://doi.org/10.2991/ijcis.d.210105.001

[12] S. Duque, and M. N. bin Omar, "Using data mining algorithms for developing a model for intrusion detection system (IDS)", *Procedia Computer Science*, vol. 61, pp. 46-51,2015. https://doi.org/10.1016/j.procs.2015.09.145

[13] D. A. Cieslak, N. V. Chawla, and A. Striegel, "Combating imbalance in network intrusion datasets", in *IEEE Int. Conf. Granular Comput.*, 2006.

[14] A. A. A. Alsameraee, and M. K. Ibrahem, "Toward constructing a balanced intrusion detection dataset based on CICIDS2017", *Samarra Journal of Pure and Applied Science*. vol. 2, no.3, pp. 132-142, 2020. https://doi.org/10.54153/sjpas.2020.v2i3.86

[15] P. T. Dinh, and M. Park, "R-EDoS: robust economic denial of sustainability detection in an SDN-based cloud through stochastic recurrent neural network", in *IEEE Access*. 9, pp. 35057-35074, 2021. https://doi: 10.1109/ACCESS.2021.3061601.

[16] A. Verma, and V. Ranga, "Machine learning based intrusion detection systems for IoT applications", *Wireless Personal Communications*, vol. 111, pp. 2287-2310, 2020. https://doi.org/10.1007/s11277-019-06986-8

[17] D. Jing, H.-B. Chen, "SVM based network intrusion detection for the UNSW-NB15 dataset", in *2019 IEEE 13th International Conference on ASIC (ASICON), IEEE*, pp. 1–4, 2019. https:// doi: 10.1109/ASICON47005.2019.8983598.

[18] N. Moustafa, J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set) ", in *2015 Military Communications and Information Systems Conference, MilCIS 2015 - Proceedings, Institute of Electrical and Electronics Engineers Inc*., pp. 1-6, 2015. https:// doi: 10.1109/MilCIS.2015.7348942.

[19] L. Zhiqiang, G. Mohi-Ud-Din, L. Bing, L. Jianchao, Z. Ye, L. Zhijun, "Modeling network intrusion detection system using feed-forward neural network using unsw-nb15 dataset", in *2019 IEEE 7th International Conference on Smart Energy Grid Engineering (SEGE), IEEE*, pp. 299–303, 2019. https:// doi: 10.1109/SEGE.2019.8859773.

[20] S.M. Kasongo, Y. Sun, "Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset", *Journal of Big Data*, vol. 7, pp.1-20, 2020. https://doi.org/10.1186/s40537-020-00379-6

[21] A. Divekar, M. Parekh, V. Savla, R. Mishra, M. Shirole, "Benchmarking datasets for Anomaly-based Network Intrusion Detection: KDD CUP 99 alternatives", in *2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS)*, pp. 1-8, 2018. https://doi: 10.1109/CCCS.2018.8586840

[22] G. O. Coli, S. Aina, S. D. Okegbile, A. R. Lawal, and A. I. Oluwaranti, "DDoS Attacks Detection in the IoT Using Deep Gaussian-Bernoulli Restricted Boltzmann Machine", *Modern Applied Science*, vol. 16, no. 2, pp. 1-12, 2022. https://doi: 10.5539/mas.v16n2p12

[23] Y. Imamverdiyev, and F. Abdullayeva, "Deep learning method for denial of service attack detection based on restricted boltzmann machine" *Big data*, vol. 6, no. 2, pp. 159-169, 2018. https://doi.org/10.1089/big.2018.0023

[24] M. A. Cueto, J. Morton, and B. Sturmfels, "Geometry of the restricted Boltzmann machine", *Algebraic Methods in Statistics and Probability*, vol. 516, pp. 135-153, 2010.

[25] N. T. Van, and T. N. Thinh, "An anomaly-based network intrusion detection system using deep learning", in *2017 International Conference on System Science and Engineering (ICSSE), IEEE*, pp. 210-214, 2017. https://doi: 10.1109/ICSSE.2017.8030867

[26] A. Gouveia, and M. Correia, "A systematic approach for the application of restricted Boltzmann machines in network intrusion detection.", in *International Work-Conference on Artificial Neural Networks, Cham: Springer International Publishing*, pp. 432-446. 2017. https://doi.org/10.1007/978-3-319-59153-7_38

[27] P. Raj, P. E. David, "The digital twin paradigm for smarter systems and environments: The industry use cases", Academic Press (2020).

[28] J. Noh, T. Badloe, C. Lee, J. Yun, S. So, and J. Rho, "Inverse design meets nanophotonics: From computational optimization to artificial neural network", *Intelligent Nanotechnology*, vol. 3, no.32, 2023. https://doi.org/10.1016/B978-0-323-85796-3.00001-9

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

[29] S. Abinaya, M. K. Devi, "Enhancing crop productivity through autoencoder-based disease detection and context-aware remedy recommendation system", in *Application of Machine Learning in Agriculture, Academic Press*, pp. 239-262, 2022.
[30] R. Rajamanickam, D. Baskaran, "Neural network model for biological waste management systems", in *Current Trends and Advances in Computer-Aided Intelligent Environmental Data Engineering, Academic Press*, pp. 393-415, 2022.
[31] T. Menzies, E. Kocagüneli, L. Minku, F. Peters, B. Turhan, "Using goals in model-based reasoning", *Sharing Data and Models in Software Engineering*, vol. 1, pp. 321-353, 2015.
[32] J. Naskath, G. Sivakamasundari, and A.A.S. Begum, "A Study on Different Deep Learning Algorithms Used in Deep Neural Nets: MLP SOM and DBN", *Wireless Pers Commun*, vol. 128, pp. 2913–2936, 2023. https://doi.org/10.1007/s11277-022-10079-4
[33] J. Brownlee, "What is the Difference Between a Batch and an Epoch in a Neural Network", *Machine Learning Mastery 20*, 2018.
[34] D. R. Wilson, and T. R. Martinez, "The general inefficiency of batch training for gradient descent learning", *Neural networks*, vol. 16, no. 10, pp. 1429-1451, 2003.
[35] J. Ke, and X. Liu, "Empirical analysis of optimal hidden neurons in neural network modeling for stock prediction" in *2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, IEEE*, vol. 2, pp. 828-832, 2008.
[36] K. G. Sheela, and S. N. Deepa, "Review on methods to fix number of hidden neurons in neural networks", *Mathematical problems in engineering*, vol. 2013, 2013.

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com

الخلاصة

المقدمة: أدى الاعتماد المتزايد على التطبيقات المستندة إلى الإنترنت إلى ارتفاع عدد المتسللين، مما يشكل تهديدا كبيرا لأمن وسرية الموارد الرقمية. يتأثر أداء نموذج التعلم العميق (DL) بشكل كبير بتصميم بنيات التعلم العميق، والتي تحتاج عادةً إلى معرفة الخبراء. يمكن تعزيز نتائج كشف التسلل بواسطة النموذج عن طريق ضبط معلمات معينة. يساعد الاختيار الصحيح لأفضل معلمات النموذج على تحسين أداء الكشف.

طرق العمل: درس هذا البحث أهمية بعض المعلمات الفائقة المستخدمة في خوارزمية التصنيف لتحسين أداء نماذج كشف التسلل. يتم استخدام آلة بولتزمان المقيدة (RBM) وخوارزمية MLP لتحديد أفضل المعلمات لتدريب مجموعة بيانات UNSW–NB15 وتعزيز الدقة ووقت التنفيذ لاكتشاف الهجوم.

النتائج: تم استخدام أعداد مختلفة من الخلايا العصبية المخفية والدورات وحجم الدفعة لإثبات المعلمة الفائقة التي لها التأثير الأكبر على الأداء من حيث الدقة والوقت لاكتشاف الهجمات.

الاستنتاجات: يؤدي بناء نموذج وتحسينه باستخدام قيم المعلمات الفائقة المناسبة إلى تصميم أفضل للنموذج بأداء عالٍ. يجب اختيار المعلمات الفائقة بطريقة تعزز أداء النموذج من حيث زيادة الدقة وتقليل زمن التنفيذ.

الكلمات المفتاحية: نظام كشف التسلل, خوارزمية بولتزمان المقيدة, خوارزمية MLP, مجموعة بيانات UNSW–NB15, كفاءة المعلمات، التعلم العميق.

ISSN: 2312-8135 | Print ISSN: 1992-0652
info@journalofbabylon.com | jub@itnet.uobabylon.edu.iq | www.journalofbabylon.com