 doi:10.20944/preprints202201.0465.v1

Genetic Algorithm-based Optimal Deep Neural Network for Detecting Network Intrusions

Sourav Adhikary[1], Md. Musfique Anwar[2], Mohammad Jabed Morshed Chowdhury[3], and Iqbal H. Sarker[1*]

[1] Department of Computer Science and Engineering,
Chittagong University of Engineering & Technology, Chattogram-4349, Bangladesh
[2] Jahangirnagar University, Dhaka, Bangladesh
[3] La Trobe University, Australia.
*Correspondence: souravadhikary.joy@gmail.com, iqbal@cuet.ac.bd

Abstract. Computer network attacks are evolving in parallel with the evolution of hardware and neural network architecture. Despite major advancements in Network Intrusion Detection System (NIDS) technology, most implementations still depend on signature-based intrusion detection systems, which can't identify unknown attacks. Deep learning can help NIDS to detect novel threats since it has a strong generalization ability. The deep neural network's architecture has a significant impact on the model's results. We propose a `genetic algorithm` based model to find the `optimal number of hidden layers` and the `number of neurons` in each layer of the `deep neural network (DNN) architecture` for the `network intrusion detection` binary classification problem. Experimental results demonstrate that the proposed DNN architecture shows better performance than classical machine learning algorithms at a lower computational cost.

Keywords: genetic algorithm · deep neural network · hidden layer · optimal architecture · intrusion detection.

1 Introduction

Network intrusion detection detects any unauthorized access to the computer network. Intrusion detection systems (NIDS) are classified into two types: signature-based and anomaly-based [9]. Signature-based intrusion detection uses pattern matching techniques of known attacks. With increasing cyberattacks, new unknown attacks are emerging. A signature-based system uses pattern matching technique that can not detect unknown attacks. An anomaly-based NIDS uses machine learning techniques to analyze and learn the normal network behavior of a system. Then it can be used to detect unknown attacks by analyzing the deviations from normal traffic behavior.

With a good generalization ability, deep learning can enable NIDS to detect unknown attacks. One of the main topics of NIDS research in recent years has been the application of machine learning and deep learning techniques [7][8].

 doi:10.20944/preprints202201.0465.v1

Deep learning approaches show better results than conventional machine learning algorithms [5][6]. Most deep learning architectures in the literature are selected using the trial and error method which is prone to error.

To overcome the neural network architecture selection problem, we used the genetic algorithm to determine the optimum number of hidden layers and neurons in each hidden layer for the intrusion detection problem. The fitness function is a linear function that combines loss, parameter, and specificity. The search for neural network architecture is a time-consuming process. To find the optimal deep neural network architectures, the genetic algorithm is used. The best neural architecture found in the search process is used to evaluate performance and compared with other classical machine learning algorithms. The proposed deep neural network showed promising results compared with previously deployed machine learning and deep learning algorithms. the contributions of this research work are as follows:

- Finding the optimal number of hidden layers and number of neurons in each layer for intrusion detection problem using genetic algorithm.
- To design an optimal deep learning model for detecting network intrusions.

The rest of the paper is organized as follows. The related work on intrusion detection and neural architecture search is discussed in Section 2. Section 3 gives an outline of the suggested solution. The experimental findings are presented in Section 4. Section 5 analyzes the limitations of the suggested solution and finishes with future research prospects.

2 Related Works

Research on cybersecurity gains popularity in recent years. Many researchers use machine learning techniques for anomaly-based network intrusion detection [14][20] [7]. Using publicly available datasets, Vinayakumar et al. [15] conducted a systematic study of DNN and other machine learning algorithms. From a limited collection of architecture search spaces, the authors selected the DNN topology by trial and error method. In [12], the authors proposed a convolutional neural network for intrusion detection problems. A comparative study of various settings is used to find the hyperparameters. A systematic study of seven deep learning strategies was proposed by M.A. Ferrag et al. [5] with the deep neural network performing best. In order to choose both subsets of features and hyperparameters in one operation, the authors proposed a double algorithm based on particle swarm optimization (PSO) [3]. A deep neural network-based network intrusion detection system needs to be trained regularly with new data from network traffic which is computationally costly [1]. With the optimal architecture of the deep neural network, computational costs can be lessened. The architecture of the deep neural network greatly influences the performance of the model. As mentioned above many researchers use the trial and error method to find an optimal network architecture that needs extensive knowledge of deep learning

 doi:10.20944/preprints202201.0465.v1

and is also prone to error. In [19], different network architecture search methods were discussed. An algorithm for optimizing a multilayered artificial neural network is proposed which prunes neurons from the hidden layers as much as possible while retaining the same error rate [17]. Starting from an overly large network, pruning algorithms remove unimportant neurons before the optimum network emerges [16]. In [2], Bergstra et al. showed trials on a grid are less effective for hyper-parameter optimization than trials chosen at random. In [11], the authors used a genetic algorithm to find an optimal network topology and showed comparative analysis with other network architecture search algorithms. A hybrid genetic algorithm with a stochastic layer showed promising results in affordable computation cost by [6]. In [4], a modified evolutionary strategy was used to find optimal architecture for retinal vessel segmentation.
In the world of automation, it is important to automate the architecture search process. Deep neural network architecture search using the trial-and-error method used in most literature requires extensive expertise in the field and may not yield the best results at a reasonable computation cost. A genetic algorithm is used in this study to automate the optimum network architecture search procedure, which is both reliable and fast.

3 Methodology

The performance of a deep learning model which has three or more hidden layers depends on the network architectures and other parameters such as activation function used in each layer and optimization algorithm. Genetic algorithm is used to find the optimal neural network parameters. Our approach allows us to find the number of hidden layers and the number of possible neurons in each hidden layer. A combination of all those components creates a multidimensional search space of all possible neural network architectures. Genetic algorithms is one of the most utilized approaches in the studies of the evolution of neural network architectures [19]. It is a population-based search algorithm that can search over complex and multidimensional search space and reach global optimum or near optimum solution reliably and fast.

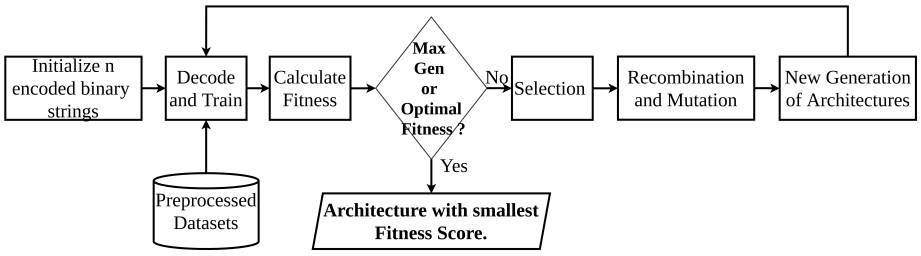

Fig. 1. Neural network architecture search using Genetic Algorithm.

 doi:10.20944/preprints202201.0465.v1

The block diagram of the neural network architecture search using the genetic algorithm is shown in Fig. 1. Encoded binary strings that represent the neural network architectures are initialized randomly. Each string is decoded to build a neural topology and trained using a prepossessed data set. The algorithm evaluates the model and calculates fitness using a fitness function. The goal of the algorithm is to minimize the fitness value. From the fitness value of the population of architectures, a certain number of chromosome strings is selected for further process. Those strings which are not selected are discarded for the population pool. Selected binary strings are recombined and mutated to generate a new set of binary strings that represent neural architecture. This new generation of chromosome strings are used to build new models. The whole process is repeated until the maximum generation is reached or a satisfactory fitness value is achieved.

3.1 Random Set of Neural Architectures

Representing the candidate solutions is the most important issue to consider in the evolution of architecture tasks. The genetic algorithm is utilized in this study to determine the number of hidden layers and the number of neurons in each hidden layer. This information is encoded into a binary string using direct binary encoding. 50 encoded binary strings that represent the neural network architectures are initialized randomly. The neural network architectures have the 5 highest possible hidden layers and 1024 maximum possible neurons. Each chromosome is a binary string of length 50 which represents the hidden layer architecture. The maximum possible hidden layer consists of 5 hidden layers having 1023 neurons in each layer. The minimum possible architecture consists of 3 hidden layers having 32 neurons in each layer.

3.2 Build and Train

Back Propagation is used to train each neural network on the training dataset and performance is evaluated using a test dataset. For optimization, Adam is used as an optimizer with the default parameter. Binary cross-entropy is used as a loss function. Activation function relu is used for inner layers and sigmoid activation function is used for the output layer. Each architecture is trained for 90 epochs with a batch size of 512.

3.3 Fitness Function

The fitness function calculates the fitness of each candidate solution. A good intrusion detection model should detect intrusion accurately with less error rate and less computation cost. In this study, the goal of the genetic algorithm is to minimize the score calculated by the fitness function described below.

$$F = \frac{(W_l \times loss) + \frac{W_s}{specificity} + (W_p \times Norm_{parameter})}{(W_l + W_s + W_p)} \quad (1)$$

where,

$$Norm_{parameter} = \frac{(p - p_{min})}{(p_{max} - p_{min})} \tag{2}$$

Parameter is the number of weights and bias in a deep artificial neural network. In equation 2, p_{max} and p_{min} represent the parameter of maximum and minimum possible neural network architecture of our proposed solution respectively. The overall objective of using p_{min} and p_{max} is to normalize the complexity of each topology tested to the interval (0,1) so that it can be smoothly incorporated into the fitness function.

$$parameter = \sum_{l=1}^{l=Layer} (Neurons_l \times Neurons_{l-1} + Bias_l) \tag{3}$$

For loss binary cross-entropy function is used. Binary cross-entropy loss is defined as

$$loss = -(y \times log(y^{pred}) + (1 - y)log(1 - y^{pred})) \tag{4}$$

where y is the true binary value and y^{pred} is the predicted value.
Specificity is the metric that evaluates a model's ability to predict the true negatives of each available category. In our proposed solution, *specificity* specifies the attack class detection rate to all attacks. This fitness function tries to maximize the specificity value to get a good attack class detection rate.

$$Specificity = TN/(TN + FP) \tag{5}$$

W_l, W_s, W_p, are three user-defined constants that represent the weight value of the loss, normalized parameter, and specificity to have more control over fitness function. The free parameters W_l, W_s, and W_p of the fitness function defined in the previous section are important to converge into optimal fitness. Due to computational constraints, all three free parameters are set as $W_l = 1, W_s = 2$, and $W_p = 0.01$. These three values are used to control the impact of loss, specificity and parameter over fitness value. As specificity value represents the ability to detect attack class, specificity value should have the most impact on fitness value.

3.4 Selection

The selection process as shown in Fig. 2 uses a hybrid method using the rank selection and the tournament selection method. The algorithm selects several candidate solutions with the lowest fitness value. In the tournament selection method as described in Goldberg [18], the algorithm randomly selects two architectures and compares their fitness value. Architectures with the lowest fitness value are selected for recombination. This hybrid method ensures the diversity and effectiveness of the neural network architecture population pool.

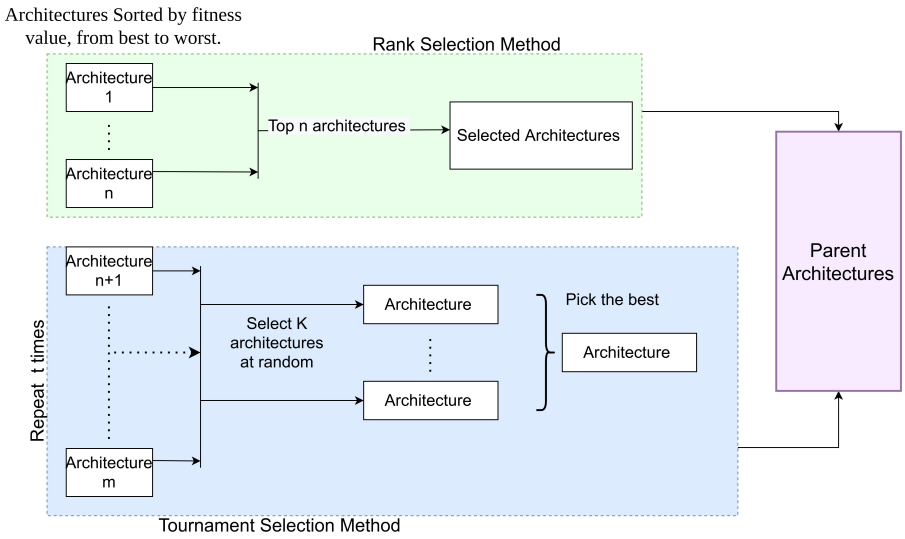

Fig. 2. Selection process of the best architecture

3.5 Recombination

The selected architecture candidates are combined to get all possible groups of architecture candidates of size two called parents. From the pool of parents, 15 parents are selected randomly for recombination. For recombination, a two-point crossover is used as prescribed in [13]. In a two-point crossover, two randomly chosen crossover points are used to exchange a string segment that falls between the two points. The recombination process generates 10 new chromosomes. The two point crossover recombination process is shown in Fig. 3.

3.6 Mutation

In mutation, a random binary string of the chromosome is complemented as shown in Fig.4. A new generation is evolved by mutating the candidate strings from the recombination process. After 82 generations, the new best architecture learning rate slows down, and the network architecture search is terminated due to computational constraints. So, the total number of architecture evaluated is 1690 from architecture search space size of 1.023×10^{15}.

4 Results And Discussion

4.1 Dataset

CSE-CIC-IDS-2018 dataset collected by the Canadian Institute for Cybersecurity is used to develop a deep learning model for intrusion detection [10]. The

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 January 2022 doi:10.20944/preprints202201.0465.v1

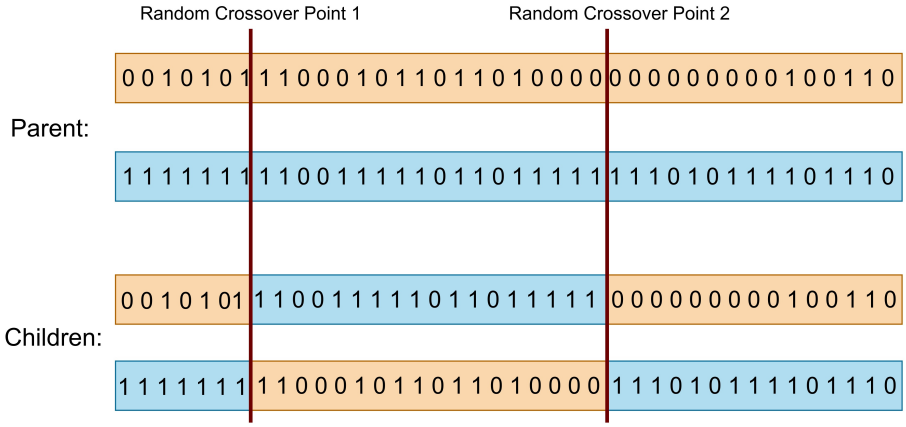

Fig. 3. Two-point crossover of parent architectures

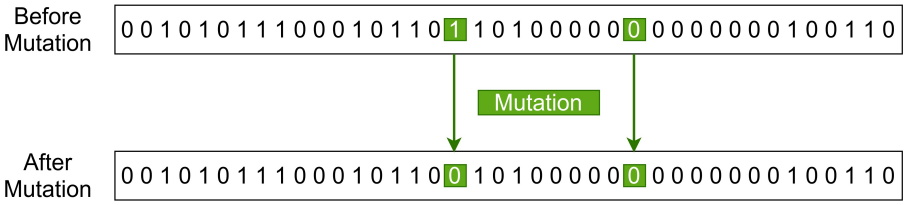

Fig. 4. Mutation process of selected architecture

dataset has 80 network traffic features. It comprises seven distinct attacks such as brute-force, heartbleed, botnet, dos, DDoS, web assaults, and infiltration of the network within. Data is normalized to improve the generalization ability of DNN.

Table 1. Training and testing CSE-CIC-IDS-2018 dataset

Class	Train	Test
Benign	1592052	957820
Attack	807948	478910

4.2 Results of optimal DNN search using Genetic Algorithm

Fig. 5 shows the smallest fitness score of each generation getting smaller over generations. The slope of the curve shows that the genetic algorithm has learned

to find the neural network architecture of the smallest fitness value. We introduced random architecture in every generation which causes the many ups and downs of the curve. Here, 0 is the initial generation. The individuals in the population are getting better and better during the evolution process. After 82 generations the learning rate of the evolution algorithm became very low. Due to computation hardware constraints, we stopped at the 82nd generation. The fitness score of the 82nd nd generation is the smallest and we take the individuals as the output of the neural network architecture search. With the relu activation function and neurons 172,182,512,38,74, the optimal architecture has five fully connected hidden layers. For binary classification, one neuron in the output layer has the sigmoid activation function.

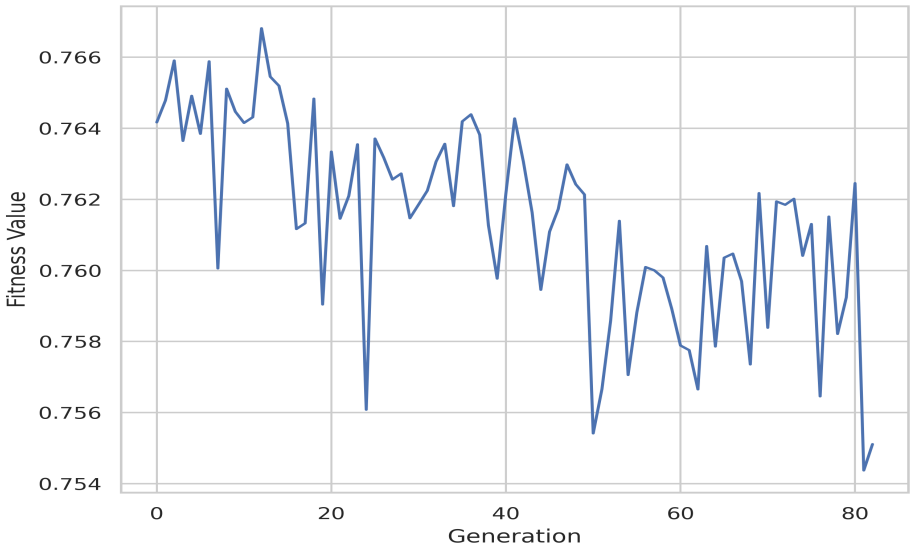

Fig. 5. Min Fitness scores over generations

4.3 Results of Proposed Deep Learning Model

The degree of convergence of the training set loss is a crucial criterion for determining whether a model is effective. If it continues to plummet, it is a better training model. Fig.6 shows the training time loss and validation curve of the selected network architectures keep decreasing which indicates an effective model. The selected architecture is trained for 300 epochs to get the best results. The attack class has been defined as the negative class in the preprocessing step. A high specificity score represents the high attack class detecting ability. High specificity is crucial for intrusion detection problems. The test results of various DNN hidden layer architectures are shown in Table 2. 174,182,512,38,74,64

 doi:10.20944/preprints202201.0465.v1

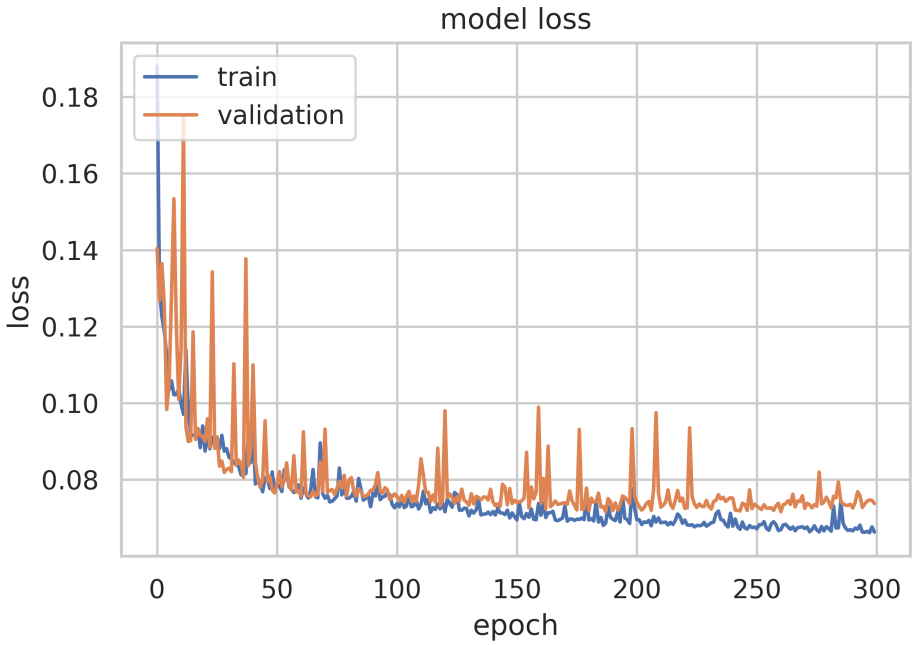

Fig. 6. Training loss of the proposed architecture.

are the number of neurons used for the different layers of hidden layer architecture respectively. In terms of accuracy and f1 score architectures of hidden layer 3,4, and 5 performed better. The highest specificity score was found in layer 5 architecture.

Table 2. Test results of DNN hidden layer architectures

No of Hidden Layer	Accuracy	F1 Score	Specificity	Parameter
1 Layer	0.926	0.960	0.734	13761
2 Layer	0.946	0.966	0.782	45257
3 Layer	0.967	0.982	0.912	139283
4 Layer	0.968	0.982	0.905	158303
5 Layer	0.967	0.981	0.945	161225
6 Layer	0.963	0.978	0.881	166015

The proposed automated method successfully finds the best architecture within a reasonable time frame. The 5 hidden layer architecture has the largest Area Under the Curve as shown in figure 7.

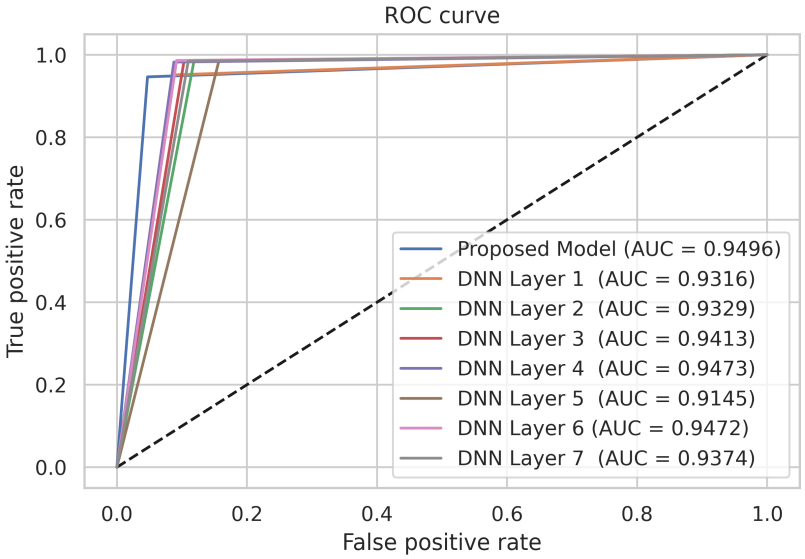

Fig. 7. ROC curves of DNN layers

The automated search process of deep neural network architecture using the genetic algorithm found the optimal number of hidden layers and number of neurons in each layer. The performance of the proposed deep neural network model is compared with five classical machine learning algorithms shown in table 3 and figure 8.

Table 3. The comparison of machine learning algorithms

Algorithms	**Accuracy**	**Precision**	**Recall**	**F1 Value**	**Specificity**
Logistic Regression	0.86	0.86	0.98	0.91	***0.52***
KNeighbors	0.89	0.95	0.99	0.96	***0.84***
Decision Tree	0.90	0.95	0.95	0.95	***0.86***
Gaussian Naive Bayes	0.86	0.87	0.94	0.90	***0.56***
Random Forest	0.91	0.95	0.99	0.97	***0.85***
Deep Neural Network	**0.97**	**0.99**	**0.98**	**0.98**	***0.95***

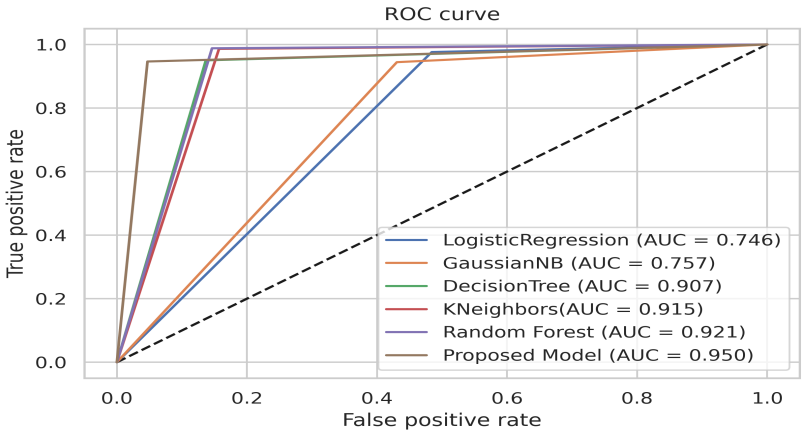

Fig. 8. ROC curves of classical machine learning algorithm and proposed model

All the models shown in Table 3 have been trained and tested on the same set of test and train datasets. Our proposed model outperforms the classical machine learning models in terms of specificity and f1 score.

5 Conclusion

In this study, the deep neural network model is developed by determining the optimal number of hidden layers and neurons in each layer for intrusion detection binary classification problems. The number of hidden layers and neurons in each layer is encoded in binary strings which are used with a genetic algorithm to find the optimal architectures. The proposed deep neural network architecture outperforms previously deployed machine learning in terms of attack classification ability and computational cost, to the best of our knowledge. However, using a genetic algorithm to find the optimal architecture is computationally expensive and time-intensive. This study only focused on the binary classification of network intrusion detection problems. By extending the search space on advanced hardware, the genetic algorithm's performance may improve. One of the interesting directions for future research is to use the genetic algorithm to design deep neural network architecture for other classification problems.

References

1. Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., Ahmad, F.: Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies **32**(1), e4150 (2021)

2. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. Journal of machine learning research **13**(2) (2012)
3. Elmasry, W., Akbulut, A., Zaim, A.H.: Evolving deep learning architectures for network intrusion detection using a double pso metaheuristic. Computer Networks **168**, 107042 (2020)
4. Fan, Z., Wei, J., Zhu, G., Mo, J., Li, W.: Evolutionary neural architecture search for retinal vessel segmentation. arXiv e-prints pp. arXiv–2001 (2020)
5. Ferrag, M.A., Maglaras, L., Janicke, H., Smith, R.: Deep learning techniques for cyber security intrusion detection: A detailed analysis. In: 6th International Symposium for ICS & SCADA Cyber Security Research 2019 6. pp. 126–136 (2019)
6. Kapanova, K., Dimov, I., Sellier, J.: A genetic approach to automatic neural network architecture optimization. Neural Computing and Applications **29**(5), 1481–1492 (2018)
7. Sarker, I.H.: Cyberlearning: Effectiveness analysis of machine learning security modeling to detect cyber-anomalies and multi-attacks. Internet of Things **14**, 100393 (2021)
8. Sarker, I.H.: Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. SN Computer Science **2**(6), 1–20 (2021)
9. Sarker, I.H., Furhad, M.H., Nowrozy, R.: Ai-driven cybersecurity: an overview, security intelligence modeling and research directions. SN Computer Science **2**(3), 1–18 (2021)
10. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: ICISSp. pp. 108–116 (2018)
11. Stathakis, D.: How many hidden layers and nodes? International Journal of Remote Sensing **30**(8), 2133–2147 (2009)
12. Tao, W., Zhang, W., Hu, C., Hu, C.: A network intrusion detection model based on convolutional neural network. In: International Conference on Security with Intelligent Computing and Big-data Services. pp. 771–783. Springer (2018)
13. Thierens, D., Goldberg, D.: Convergence models of genetic algorithm selection schemes. In: International Conference on Parallel Problem Solving from Nature. pp. 119–129. Springer (1994)
14. Tsai, C.F., Hsu, Y.F., Lin, C.Y., Lin, W.Y.: Intrusion detection by machine learning: A review. expert systems with applications **36**(10), 11994–12000 (2009)
15. Vinayakumar, R., Alazab, M., Soman, K., Poornachandran, P., Al-Nemrat, A., Venkatraman, S.: Deep learning approach for intelligent intrusion detection system. IEEE Access **7**, 41525–41550 (2019)
16. Wagarachchi, M., Karunananda, A.: Optimization of artificial neural network architecture using neuroplasticity. International Journal of Artificial Intelligence **15**(1), 112–125 (2017)
17. Wagarachchi, N.M.: Mathematical modelling of hidden layer architecture in artificial neural networks. Ph.D. thesis (2019)
18. Whitley, D.: A genetic algorithm tutorial. Statistics and computing **4**(2), 65–85 (1994)
19. Wistuba, M., Rawat, A., Pedapati, T.: A survey on neural architecture search. CoRR **abs/1905.01392** (2019), `http://arxiv.org/abs/1905.01392`
20. Zaman, M., Lung, C.H.: Evaluation of machine learning techniques for network intrusion detection. In: NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium. pp. 1–5. IEEE (2018)