DLINE JOURNALS

Journal of Information Security Research

Print ISSN: 0976-4143
Online ISSN: 0976-4151

JISR 2024; 15 (2)
https://doi.org/10.6025/jisr/2024/15/2/59-71

Impact of Neural Networks on Improving Cloud Computing Security with AI-powered Smart Intrusion Detection

Mohsin Ali[1], Abdul Razaque[1], Damelya M. Yeskendirova[1], Talgat A. Nurlybayev[1], Nessibeli Y. Askarbekova[1] and Zarina A. Kashaganova[1]
[1]International Information Technology University
Manas St. 34/1, Almaty, 050000, Kazakhstan

Received: 4 December 2023
Revised: 16 February 2024
Accepted: 9 March 2024

ABSTRACT

This work introduces a highly effective method for identifying harmful network activity by leveraging the power of artificial neural networks. These networks, particularly adept at analyzing deep packet inspection-based systems for detecting intrusions, have been the cornerstone of our research. The method was rigorously tested using a diverse range of harmless network data, including images, dynamic link library files, logs, music files, word processing documents, and malicious shell code files from the exploit and vulnerability database, exploitdb. The proposed artificial neural network design successfully distinguished between harmless and harmful network traffic. The developed neural network consistently achieved a 99% accuracy rate, a 0.99 average area under the receiver operator characteristic curve (AUC-ROC), and a false positive rate of less than 2% across various 10-fold cross-validation tests. These results underscore our classification method's reliability, accuracy, and effectiveness. This innovative strategy for identifying harmful network traffic holds significant potential for enhancing intrusion detection systems in both traditional network traffic analysis and cyber-physical systems analysis, such as smart grids. Furthermore, we present a new intrusion detection system (IDS) that combines a multilayer perceptron (MLP) network with an artificial bee colony (ABC) and fuzzy clustering algorithms. The MLP network distinguishes between standard and irregular network data packets, while the ABC method fine-tunes the connections and prejudices of the MLP during its training phase. We used the CloudSim model and the NSL-KDD data set to validate our method, evaluating it with criteria including mean absolute error (MAE), root mean square error (RMSE), and the kappa coefficient. Our findings demonstrate that our proposed technique surpasses current leading methods in the field.

Keywords: Intrusion detection, artificial neural networks, deep packet inspection, malware traffic analysis, multilayer perceptron (MLP), artificial bee colony optimization, fuzzy clustering, classification technique

1. Introduction

In today's rapidly evolving computing landscape, network intrusion detection systems (NIDS) occupy a pivotal role in safeguarding modern computing infrastructures. These systems are indispensable for monitoring and identifying unwarranted and malicious network traffic, which encompasses a spectrum of threats, including unauthorized system access and the vulnerabilities stemming from improperly configured systems. The prevailing methodology adopted by most commercial NIDS relies on signature-based detection mechanisms, where predefined rules are deployed to discern undesirable network traffic patterns by scrutinizing the behavioral patterns exhibited by the traffic. While this signature-based approach delivers commendable efficacy against known threats, its limitations become evident when confronting unknown attack vectors or when previously recognized attacks undergo modifications to elude these predefined rules [1-3]. Furthermore, the realm of signature-based detection within NIDS grapples incessantly with a recurring challenge in real-world scenarios, which is the issue of false positives. This challenge is particularly pronounced when dealing with the detection of malicious shellcode—an impactful threat vector that empowers attackers to attain unauthorized command-line access to both traditional computer systems and cyber-physical systems, such as smart grid infrastructures. Discerning between the patterns of shellcode and benign network traffic can be an intricate endeavor, often culminating in the frequent occurrence of false positives [4-7].

To emphasize this challenge, let's explore the scenario of a network security consultant collaborating with a multinational financial institution in US. In this capacity, they were tasked with utilizing network intrusion detection tools, specifically Sguil and Snort, from which Snort is primarily responsible for detecting and alerting on network intrusions, while Sguil is used to manage and analyze the alerts generated by Snort and other security data to facilitate the incident response process. These tools are commonly used together in network security operations sourced from the Debian-based Linux distribution Security Onion. During their tenure, they observed that the signatures engineered to detect shellcode frequently triggered false positives by erroneously flagging non-shellcode binaries, including dynamic link libraries (DLLs), and even innocent image files in JPG format. The prevalence of these false positives reached a threshold where the signatures had to be deactivated, rendering them ineffectual. Incidences of false positives, especially concerning shellcode and signature-based systems, are widespread, with even Microsoft dedicating extensive discourse to this issue in their patent addressing methods to detect malicious shellcode with reduced false positives in memory [8-9]. It is noteworthy that shellcode often serves as the payload in system penetration tools, offering attackers augmented access and leverage [10].

1.1. Contributions of the paper

This paper makes significant contributions to addressing the aforementioned challenges:

- **Innovative non-signature-based detection:** Our research pioneers a novel approach for detecting malicious shellcode that breaks free from the constraints of conventional signature-based methods. Instead, we harness the power of artificial neural networks (ANNs) to achieve this goal.

- **Exceptional detection accuracy:** The results we present in this paper underscore the exceptional capability of our innovative classification approach to accurately detect shellcode while minimizing false positives. Our approach substantially improves the precision of shellcode detection.

- **Thorough validation and evaluation:** We rigorously validate our approach through repeated 10-fold cross-validation, a robust method for assessing its performance.

Furthermore, we comprehensively evaluate the efficacy of our approach in generating false positive alerts, using an extensive dataset comprising typical network traffic file contents.

Remarkably, our approach yields a false positive rate of less than 2%.

- Integration of ANNs with ABC and fuzzy clustering: The paper proposes the integration of ANNs with ABC and fuzzy clustering algorithms, suggesting that this combination can improve the performance of IDS.

- **Role of fuzzy clustering:** Fuzzy clustering is employed to generate homogeneous subsets of training data, with the aim of accelerating training by segmenting the dataset into uniform subsets.

- **Role of ABC:** ABC algorithm is used to expedite the determination of optimal values for linkage weights and biases within ANNs.

- **Innovative hybrid method:** The paper pioneers a novel approach that harmonizes ANNs, ABC, and fuzzy clustering to enhance intrusion detection within cloud computing environments. This suggests a novel integration of these techniques for improving the efficiency of intrusion detection systems in the context of cloud computing.

- **Error mitigation:** The research effectively mitigates instances of misclassification, reducing mean absolute error (MAE) and root mean squared error (RMSE) within cloudbased IDS. This indicates a focus on improving the accuracy of intrusion detection and reducing errors in classification.

- **Enhanced kappa statistic:** The proposed approach elevates the efficacy of the kappa statistic, presenting an efficient cloud-based IDS technique proficient in precise instance classification. This implies an improvement in the evaluation metrics used to assess the performance of intrusion detection systems.

The cloud computing offers cost-effective, Internet-based access to IT resources, spanning operating systems, storage, and network infrastructure, hardware, and software applications [11-12]. Users pay based on resource consumption, aligning with a utility-based model. This paradigm introduces services like SaaS, IaaS, PaaS, and EaaS [13-14]. In web applications, particularly in online retail and auctions, network security is critical. Intrusion detection systems (IDS) serve as frontline defenses, detecting anomalies and recognized attack patterns.

Cloud-based intrusion detection presents challenges, leading to diverse algorithm deployments rooted in evolutionary and meta-heuristic methods [15].

1.2 Paper organization

The paper systematically explores an AI-based intrusion detection system employing artificial neural networks and artificial bee colony optimization. It commences with an introduction in section 1 outlining the significance, challenges, contributions, and paper structure. Section 2 covers background and related work, focusing on intrusion detection systems and artificial neural networks. Section 3 details the methodology, emphasizing nonsignature-based detection and rigorous validation. In section 4, we represent experimental setups results and evaluations. The research concludes in Section 5, summarizing contributions and suggesting potential future directions.

2. Background and Related Work

This section delves into the literature addressing intrusion detection systems, artificial neural networks, and artificial bee colony optimization. The discussed publications, selected based on criteria including publication dates between 2019 and 2023, inclusion in review and research papers exclusively in reputable journals, and keywords ‘IDS’ and ‘ANN’ for cloud security, were sourced from reputable platforms such as ScienceDirect, Wiley Online Library, and Google Scholar.

In a study by Varun and Ashokkumar [16], a deep neural network with game theory for cloud security (GT-CSDNN) was developed. This innovative approach incorporates both defender and attacker strategies using game theory principles. The deep neural network leverages this game theory framework to classify normal data and detect attacks effectively. The GT-CSDNN’s performance was assessed using the CICIDS—2017 dataset. The collected data underwent normalization, and an improved whale algorithm (IWA) was employed to identify optimal points for attacks and normal data. Shyla et al [17] proposed a novel intrusion detection system (IDS) based on the integration of leader-oriented K-means clustering (LKM) and an optimal fuzzy logic (FL) system. Initially, input data was clustered using LKM, and these cluster datasets were then processed by the FL system (FLS). Abnormal and normal data were scrutinized by the FLS, and FLS training was facilitated through the grey wolf optimizer (GWO) to optimize rules. Mishra et al. [18] introduced a VM Introspection (VMI)-related security model for finegrained monitoring of virtual machines (VMs) to identify well-known attacks and their variations. VMGuard utilizes the software breakpoint injection approach to trap program execution. Employing text mining techniques, VMGuard combines the 'Bag of n-grams (BonG)' method with term frequency-inverse document frequency (TF-IDF) for feature selection and extraction from attack and normal traces. The random forest (RF) method is then used to generate generic behavior profiles for different intrusion categories.

Aoudni et al. [19] presented HMM_TDL, a deep learning approach designed to prevent and detect attacks in the cloud environment. This method operates in three states, incorporating a hidden markov model (HMM) for attack detection, and it sends hyper-alerts to the database for immediate assault prevention. In an anomaly-based NIDS was developed to analyze and monitor network traffic targeting a cloud platform, the study focused on providing network administrators with insights into such traffic to enable the blocking and dropping of intrusive network connections. Binary particle swarm optimization (BPSO) was employed to select the most relevant network features, while standard PSO (SPSO) fine-tuned support vector machine (SVM) control parameters. Velliangiri and Premalatha [20] evaluated a radial basis function neural network (RBF-NN) detector for detecting distributed denial of service (DDoS) attacks. This study aimed to simplify network configuration, which can often be overly complex or inadequate, by utilizing the bat algorithm (BA) for automated RBF-NN configuration. Sathiyadhas and Soosai [21] introduced an effective dragonfly-improved invasive weed optimizer-based shepard CNN (DIIWO-based ShCNN) for intrusion detection and attack mitigation in cloud environments. Additionally, they proposed a sailfish dolphin optimizer- based deep RNN (SFDO-based deep RNN) for anomaly identification within the cloud framework, leveraging the combined SFDO technique formed by sailfish optimizer (SFO) and the dolphin echolocation (DE) technique. They also discussed the utility of the ChicWhale technique for virtual machine (VM) migration and cloud data management.

Geetha and Deepa [22] proposed a fisher kernel-based PCA combined with a grey wolf optimizer-based weight-dropped bi-directional LSTM (FKPCA-GWO WDBiLSTM) for IDS. This involves fisher kernel for dimensionality reduction and WDBiLSTM network for preserving long-term dependencies. Another approach [23] introduced a DL technique using CNN and RNN for IDS in cloud security, effectively preventing unauthorized traffic from accessing the cloud server, the IDS plays a vital role in identifying malicious attempts to compromise system operations, categorized into host-based (HIDS) and network-based (NIDS) systems. The HIDS operates on individual devices, while NIDS identifies compromises during network transit. NIDS includes signature-based and anomaly-based systems, with an emerging trend integrating IDS alerts with SIEM systems for a comprehensive security perspective [24]. The ANNs, inspired by biological neurons, process inputs through artificial neurons in hidden layers, adapting weights and biases via learning rules like gradient descent and backpropagation [25]. This self-adaptive capability allows ANNs to capture complex relationships without prior knowledge, making them valuable for diverse classification tasks [26]. In applications such as concealed weapons detection, Internet traffic prediction, and signature verification, ANNs' "black-box" adaptability proves advantageous [27]. ANNs excel in handling high-dimensional datasets, overcoming challenges of traditional techniques like decision trees [28]. In computer security, ANNs analyze software design flaws, detect viruses, and identify network attacks [29]. Despite their wide application, exploration of ANNs in shellcode detection remains limited.

3. Proposed Methods

3.1. Artificial Neural Network Design

In this section, we delve into the design and configuration of the artificial neural network utilized in our research. The primary objective was to process the byte-level data from the network traffic dataset efficiently. To achieve this, the data was converted into integer values, a process represented by the equation:

$$\text{Integer\_Value}=\text{Byte\_to\_Integer}(\text{Byte\_Data}) \qquad (1)$$

This conversion was performed meticulously, with a keen emphasis on avoiding “magic numbers” often present at the start of files, as these could potentially mislead the classifier.

The numerical results are critical, especially when dealing with the design of obfuscated shellcode. The patterns found in shellcode often bear a striking resemblance to those in benign network traffic, making precision in classification essential. To prepare the data for the ANN, we extracted 1000 bytes of contiguous data, which served as input to the network. To ensure consistent input size, zero padding was applied where necessary. The process of extracting contiguous data is described by the equation:

$$\text{Contiguous\_Data}=\text{Extract\_Bytes}(\text{Byte\_Data},1000) \quad (2)$$

While the initial exploration of the data revealed distinct patterns within different file types. It is noteworthy that there was considerable variability among files of the same class. In our experiments, the ANN was implemented using the MATLAB (R2022a) robotics system tool box and neural network toolbox. The optimal structure of the ANN was identified through an exhaustive grid search process. The best structure, determined in terms of classification accuracy, was identified as a multi-layer perceptron with two hidden layers. Each hidden layer consisted of 30 hidden neurons. This optimal configuration of the ANN can be succinctly expressed as:

$$\text{MLP}(\text{Input},\text{Hidden\_Layers},\text{Output}) \qquad (3)$$

The structure optimization process involved repeated 10-fold cross-validation to rigorously evaluate various classifier designs, ensuring robustness. An overview of the final optimized classifier design is presented in Figure 1. For the training of the neural network, we employed the resilient backpropagation learning strategy, which utilized a default learning rate of 0.01 and conducted training for a maximum of 1000 epochs. Additionally, the feed-forward neural network was utilized to establish the initial values of the weights.

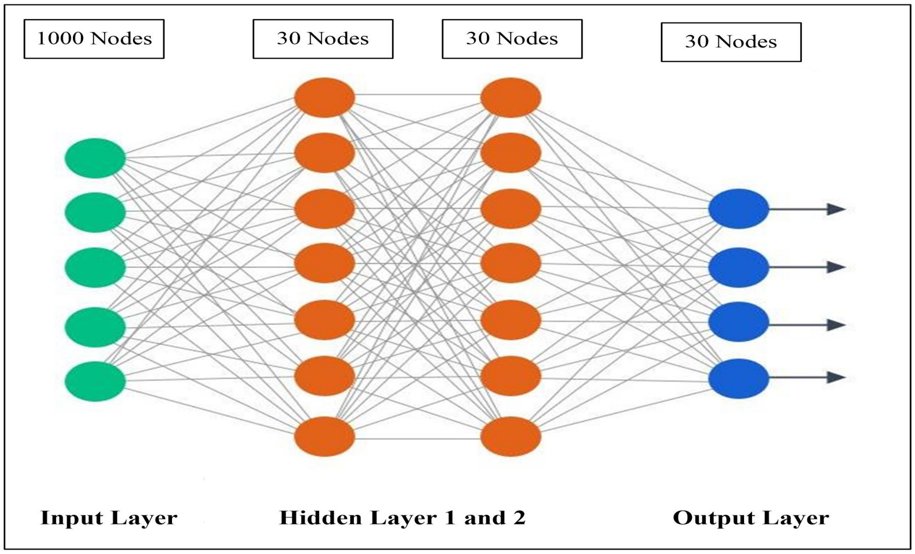


Figure 1. Architectural configuration of the artificial neural network

3.2. Integrated Methodology for Intrusion Detection System

In this section, we outline a comprehensive intrusion detection system methodology that leverages the synergies of fuzzy clustering, an ABC algorithm, and MLP network. Our IDS system is meticulously designed to efficiently detect intrusions within the intricate landscape of network traffic. This methodology is organized into three pivotal phases: training, validation, and testing, each contributing significantly to the optimization of our IDS architecture.

Our journey begins with the training phase, where we embark on the task of extracting and fine-tuning fuzzy rules. These rules are instrumental in achieving precise intrusion detection. We do so by harnessing the potential of the training dataset, with the ultimate aim of attaining exceptional accuracy when applied to the test dataset. The validation phase assumes paramount importance as it acts as a litmus test for the system's overall performance. During this phase, we meticulously scrutinize the validation dataset, continuously monitoring error rates. The training process concludes when we observe a consistent upward trend in errors, signifying the need for further refinement. In the testing phase, we put our trained model to the ultimate test, deploying it for intrusion detection by processing the test data. The dataset is systematically divided into three distinct subsets: training (TR), validation (VA), and testing (TE). At the core of our methodology lies the concept of clustering based on similarity measures, a pivotal step in unveiling the natural groupings or clusters within multidimensional data. The fundamental goal of fuzzy clustering is to categorize the dataset into clusters based on key attributes, specifically emphasizing homogeneity within clusters and heterogeneity between clusters. This strategic approach ensures that data residing within separate clusters exhibit maximum dissimilarity, enhancing our system's ability to distinguish between benign and malicious traffic.

The Fuzzy C-Means (FCM) clustering algorithm plays a central role in our methodology for partitioning the dataset into clusters. The FCM equation is represented as follows:

The steps for k-means clustering, a vital component of FCM, encompass selecting the desired number of clusters (k), initializing starting values, classifying data points, recalculating centroids, and ascertaining convergence.

$$U_{ij}=(\sum_{k=1}^{c}(\frac{x_i-c_j}{x_{i-c_k}})\,\frac{2}{m-1})^{-1}$$

This equation forms the basis of our FCM clustering process, allowing us to effectively group network traffic data into clusters that optimize our subsequent IDS elements.

MLP-based IDS model: Subsequent to clustering, the training set (TR) is subdivided into k subsets (TRk). Our IDS model relies on a multilayer perceptron augmented with a backpropagation algorithm. The BP learning rule employs the steepest descent method to fine- tune network weights and threshold values, minimizing the error sum of squares.

$$F(x) = \frac{1}{1+e^{-x}} \qquad (5)$$

The ABC algorithm assumes a pivotal role in optimizing node weights by quantitatively fine- tuning linkage weights and biases. This algorithm emulates the intelligent foraging behavior of honeybee swarms, comprising employed bees, onlookers, and scouts. Employed bees discover food sources by observing hive dances, onlooker bees identify sources, and scout bees explore new sources. The optimization process ensures that the nectar quantity of a food source corresponds to the quality (fitness) of the associated solution. It plays a crucial role in determining the output of each neuron within the MLP.

$$\Delta\, w_{ij}\,(n) = \eta\delta x_i\,(n) \qquad (6)$$

Where: $\Delta\, w_{ij}$ (n) *denotes the change in the weight connecting neuron, i to neuron and j at iteration,* η *is the learning rate,* δj *(n) represents the error signal at neuron j at iteration,* x_i *is the input to neuron i. This equation guides the weight adjustment process during network traini ng. The ABC algorithm continues to iteratively refine the linkage weights and biases of the MLP network until the output model achieves an acceptable error rate during the testing phase using the TE dataset. The optimization process can be summarized as follows:*

$$f_{ij}^{t+1}= f_{ij}^{t} + \phi \cdot \ (f_{ij}^{t} - f_{kj}^{t}). \, rand\ (\ ) \qquad (7)$$

Where f_{ij}^{t+1} *the updated position of the employed bee is,* f_{ij}^{t} *is the current position of the employed bee.* f_{kj}^{t} *is the position of the food source indicated by the onlooker bee,* Φ *is a scaling factor and rand ( ) generates a random number between 0 and 1. This iterative process ensures the optimal*configuration of the IDS model for effective intrusion detection.iterations of repeated 10-fold cross-validation.

4. Experimental setup and results

This section provides experimental configuration and results.

4.1. Simulation framework and hardware

In this section, we detail the experimental setup employed to evaluate and validate our proposed IDS using CloudSim version 5.0. The setup encompasses the simulation environment, dataset selection, and relevant considerations. To rigorously assess the performance of our IDS, we harnessed the capabilities of CloudSim version 5.0, a versatile and extendable simulation framework. The simulations were executed on a high-performance computing system equipped with a robust Core-i9 2900 CPU and 18 GB of DDR3 RAM. CloudSim's comprehensive modeling and simulation features for cloud computing infrastructures and services made it the natural choice for our research, ensuring the reliability and accuracy of our evaluations. For the crucial tasks of training and testing our IDS, we opted for the NSL-KDD dataset, an enhanced iteration of the KDD'99 dataset, recognized for its relevance in intrusion detection research. The dataset boasts a substantial repository of five million connection records, with approximately two million meticulously reserved for exclusive testing purposes. Within this dataset, we meticulously examined 55 distinctive features extracted from each connection record. These features are accompanied by labels that categorize connection status into two classes as the normal or indicative of a specific attack type. These features span various data types, including continuous, discrete, and symbolic variables.

4.2. Result and evaluation

In Section 3.1, we applied the artificial neural network classifier to a network traffic dataset encompasses both benign and malicious files. To ensure the classifier's robustness and gen

Metric	*Mean*	*Standard Deviation*
Accuracy	*0.99*	*0.01*
Precision	*0.98*	*0.01*
Sensitivity (Recall)	*0.94*	*0.03*

Table 1. Metrics for classifier performance

Metrics	*Value*
Average AUROC	*0.99*
Standard Deviation AUROC	*0.01*
Maximum AURO	*1.00*
Minimum AUROC	*0.84*

Table 2. Metrics for AUROC

eralization to unseen data, we employed repeated 10-fold cross-validation. Table 1 presents the mean (highlighted in bold) and standard deviation of accuracy, precision, and sensitivity across 1000 iterations of repeated 10-fold cross-validation.

Figure 2 showcases a receiver-operator characteristics curve derived from data collected during all 1000 iterations of the repeated 10-fold cross-validation process. ROC curves provide a valuable analysis of the trade-off between a classifier's sensitivity and specificity at different classification thresholds. The area under the ROC curves (AUROC), reported in Table 2, and quantifies the overall discrimination capability of a classification model. A higher AUROC signifies superior differentiation between distinct classes. In Fig. 2, represents the average ROC curve across all 1000 iterations, while the shaded grey area illustrates the range of ROC curves produced during these iterations. The dashed line serves as a benchmark, representing the performance of a random classifier that assigns file classes arbitrarily.

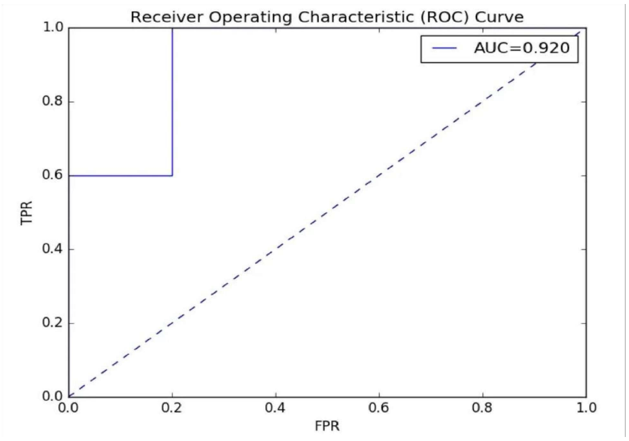


Figure 2. ROC curve for detection of malicious file content

Metrics	***Value***
Average AUROC	*0.99*
Standard Deviation AUROC	*0.01*
Maximum AURO	*1.00*
Minimum AUROC	*0.84*

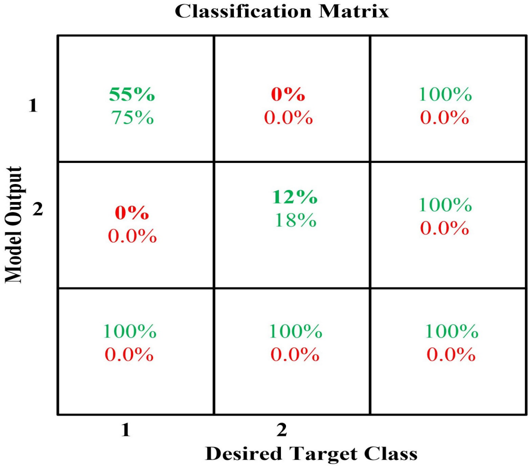
Table 2.s Metrics for AUROC

4.2.1. Classifier performance on unseen test data

In Figure 3, we present the performance of one of our best-trained artificial neural network designs on a completely unseen test dataset. Remarkably, this dataset was entirely excluded from both the training process and the cross-validation procedure. The results are striking: the best-performing classifier correctly identified 100% of malicious file contents in the test dataset, demonstrating a remarkable absence of false positives.

4.2.2. Evaluating false positives

Assessing the performance of the best-trained classifier in terms of flagging false positives on an extensive dataset of candidate network traffic data is critical. An overabundance of false positives renders a network intrusion detection system ineffective, as genuine mali-



	1	2	
1	55% 75%	0% 0.0%	100% 0.0%
2	0% 0.0%	12% 18%	100% 0.0%
	100% 0.0%	100% 0.0%	100% 0.0%

Figure 3. Performance evaluation on unseen test data: confusion plot matrix

cious code becomes obscured by misidentified benign traffic. To test this, we processed data from 500,000 random files, encompassing various types such as text files, log files, compressed and uncompressed music, executable, office documents, and miscellaneous file data. This dataset was prepared to match the format expected by our artificial neural network. The classifier was then applied to this benign data. Across this large-scale dataset, the classifier misidentified a mere 10,500 samples, accounting for approximately 1.8% of all data samples. This comprehensive evaluation demonstrates the robustness and accuracy of our intrusion detection system, particularly in the critical task of minimizing false positives while effectively identifying malicious file content.

4.2.3. Evaluation methodology and metrics for intrusion detection system assessment

In this section, we provide a comprehensive overview of the evaluation methodology employed to assess the effectiveness of our intrusion detection system. Subsequently, we present the results obtained using various evaluation metrics, each serving a unique purpose in assessing the system's performance. The mean absolute error calculates the average magnitude of errors between our system's predictions and actual outcomes, offering valuable insights into prediction accuracy. Root mean squared error: RMSE quantifies differences between predicted and observed values, effectively measuring the spread of prediction errors for a deeper understanding of accuracy. Kappa statistic: This statistic assesses the level of agreement between observed classifier accuracy and expected accuracy, considering random agreement between classifiers for a more realistic evaluation compared to traditional accuracy measures.

4.2.4. Performance evaluation of IDS: comparative analysis and metrics

The graphical representation in Figure 4(a) presents a performance comparison of four different IDS: NNID, FC-ANN, SRF, and the proposed method. The numbers in the figure represent accuracy percentages or some performance metric. For instance, the proposed method achieved an impressive 99.01% accuracy, indicating its ability to correctly classify instances or detect intrusions. This suggests that the 'proposed' method outperforms the other three methods (NNID, FC-ANN, and SRF) in terms of accuracy. Figure 4(b) illustrates a comparative analysis of four different Intrusion Detection Systems (IDS): NNID, FC-ANN, SRF, and the proposed method. The figures represent performance metrics, and the proposed method achieved an impressive value of 0.97, indicating a high level of precision and performance. In contrast, the other three methods (NNID, FC-ANN, and SRF) show different values, reflecting their respective performance levels.

Figure 4(c) provides a comparison of the average absolute deviation values for different intrusion detection systems, including NNID, FC-ANN, SRF, and our proposed method. Average absolute deviation measures the average magnitude of errors between the predictions made by each ID and the actual outcomes. Smaller values in this table indicate that the IDS predictions are closer to the actual outcomes, signifying higher prediction accuracy. As we move from NNID to our proposed method, there is a decreasing trend in the average absolute deviation values. This trend suggests that our proposed method exhibits improved accuracy and precision in predicting outcomes compared to the other IDSs listed in Figure 4(c), and Figure 4(d) presents RMSE values for different intrusion detection systems, including NNID, FC-ANN, SRF, and our proposed method. RMSE measures the spread of prediction errors in the IDS performance evaluation. Lower RMSE values indicate higher accuracy and precision in predicting outcomes. As the IDS moves from NNID to our proposed method, there is a decreasing trend in RMSE values, showcasing the improved accuracy of our approach compared to the others.

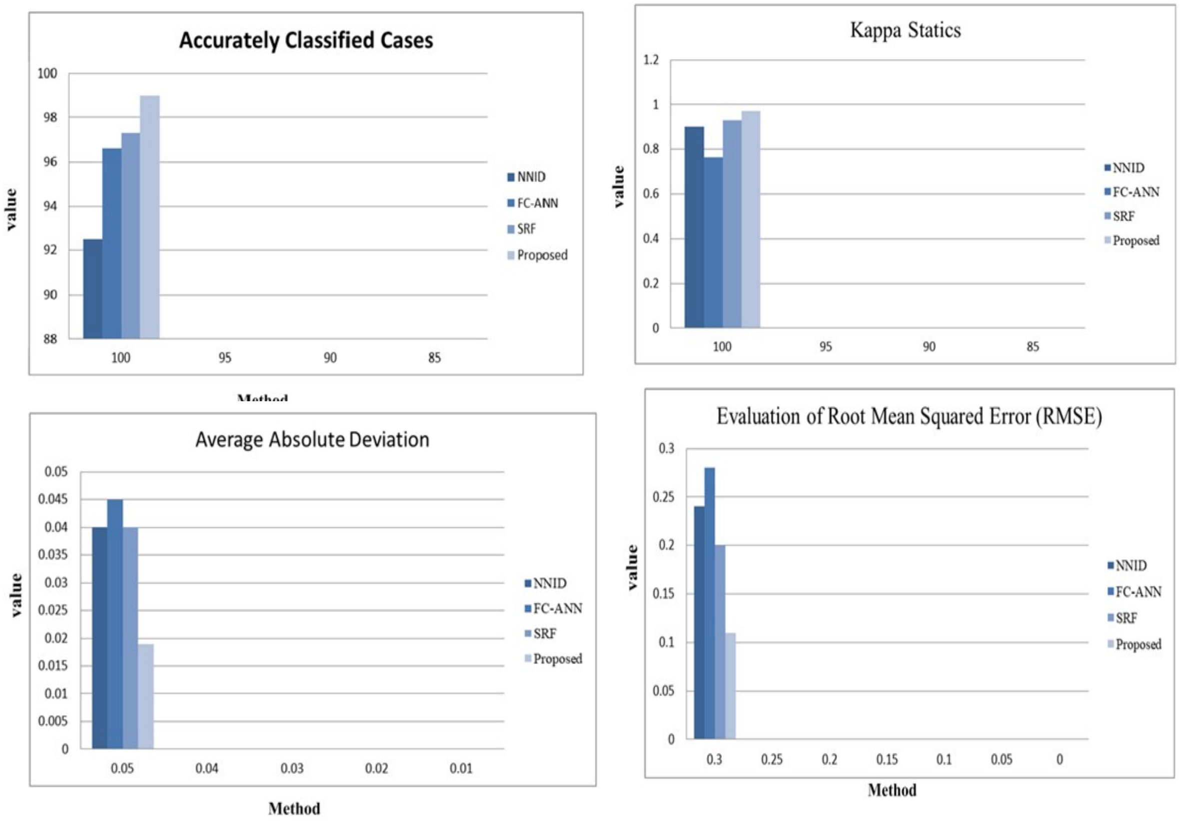


Figure 4 (a) Accurately classified cases comparison. (b) Kappa statistics analysis. (c) Average absolute deviations. (d) Evaluation of root mean squared error (RMSE)

The graphical representation in Figure 4 provides a holistic view of our IDS's performance compared to establish IDSs, including fuzzy clustering, artificial neural networks, network-node intrusion detection, and the selection of relevant features (SRF). Our proposed method consistently outperforms these IDSs across various evaluation criteria. Notably, we observe a significant 2.5% improvement in correctly classified instances, encompassing true positives and false positives. Furthermore, our method exhibits higher accuracy, as evidenced by a Kappa statistic value surpassing SRF by 0.0471. In summary, our approach achieves enhancements ranging from 2.571% to 7.865% in correctly classified instances in Figure 4(a) and attains Kappa statistic values ranging from 0.0471 to 0.208 in Figure 4(b). Moreover, our method yields lower MAE and RMSE values, as evidenced in Figure 4(c) and 4(d), respectively.

5. Conclusion and Future Directions

The research presented in this paper showcases remarkable IDS that represents a substantial improvement over traditional signature-based detection methods. Our IDS leverages an artificial neural network classifier for detecting shellcode patterns within network traffic, achieving outstanding results. The ANN-based classifier not only demonstrates perfect sensitivity by identifying all instances of shellcode but also excels in precision, minimizing false positives, a critical concern in real-world network intrusion systems. Furthermore, our approach underwent rigorous evaluation; including testing on an extensive dataset containing 500,000 samples of benign network traffic file content. Impressively, our method achieved a false positive rate of less than 2%, addressing the challenge of minimizing false positives. This achievement is significant because high false positive rates can severely impact signal-to-noise ratios, rendering intrusion detection systems ineffective.

Whereas, our paper focuses on offline shellcode pattern detection; our ongoing efforts aim to seamlessly integrate this method into online network intrusion detection systems. Future work entails real-time testing on network data, complemented by enhancements to optimize its performance in live network traffic monitoring. Additionally, we envision applying this intelligent approach to other facets of network security, including the detection of cross-site scripting attacks and SQL injection attacks targeting web applications. These domains hold significant promise for future research and development. Furthermore, we introduced new IDS that combine artificial neural networks, artificial bee colony algorithms, and fuzzy clustering.

This ID effectively discriminates between normal and abnormal network traffic packets, producing distinct training subsets through fuzzy clustering. The ABC algorithm plays a vital role in updating linkage weights and biases for MLP training. Our simulations employ CloudSim software and the NSL-KDD dataset, and we evaluate the IDS using various criteria, including MAE, RMSE, and the kappa statistic, showcasing its superiority over similar IDS systems. The integration of meta-heuristic methods and genetic algorithms in our approach holds promise for future enhancements in intrusion detection systems.

References

[1] Talaei Khoei, Tala, and Naima Kaabouch. (2023). A Comparative Analysis of Supervised and Unsupervised Models for Detecting Attacks on the Intrusion Detection Systems. Information 14, no. 2: 103.

[2] Debicha, Islam, Richard Bauwens, Thibault Debatty, Jean-Michel Dricot, Tayeb Kenaza, and Wim Mees (2023). TAD: Transfer learning-based multi-adversarial detection of evasion attacks against network intrusion detection systems. Future Generation Computer Systems 138: 185-197.

[3] Almiani, Muder, Alia AbuGhazleh, Amer Al-Rahayfeh, Saleh Atiewi, and Abdul Razaque (2020). Deep recurrent neural network for IoT intrusion detection system. Simulation Modelling Practice and Theory 101: 102031.

[4] Linka, Kevin, and Ellen Kuhl (2023). A new family of Constitutive Artificial Neural Networks towards automated model discovery. Computer Methods in Applied Mechanics and Engineering 403: 115731.

[5] Ali, Mohsin, Nurgul Nalgozhina, Olzhas Tasmagambetov, and Yerzhan N. Seitkulov (2022). Homomorphic Encryption with Enhanced Elliptic Curve for Mobile Wireless Network Security.

[6] Khan, Muhammad Ashfaq (2021). HCRNNIDS: Hybrid convolutional recurrent neural network-based network intrusion detection system. Processes 9, no. 5: 834.

[7] Alamleh, Amneh, O. S. Albahri, A. A. Zaidan, A. H. Alamoodi, A. S. Albahri, B. B. Zaidan, Sarah Qahtan et al. (2023). Multi-attribute decision-making for intrusion detection systems: A systematic review. International Journal of Information Technology & Decision Making 22, no. 01: 589-636.

[8] Razaque, Abdul, Bandar Alotaibi, Munif Alotaibi, Fathi Amsaad, Ansagan Manasov, Salim

Hariri, Banu B. Yergaliyeva, and Aziz Alotaibi (2021). Blockchain-enabled deep recurrent neural network model for clickbait detection. IEEE Access 10: 3144-3163.

[9] Capuano, Nicola, Giuseppe Fenza, Vincenzo Loia, and Claudio Stanzione (2022). Explainable artificial intelligence in cybersecurity: A survey. IEEE Access 10: 93575-93600.

[10] Ajmal, Abdul Basit, Shawal Khan, Masoom Alam, Abolfazl Mehbodniya, Julian Webber, and Abdul Waheed (2023). Towards Effective Evaluation of Cyber Defense: Threat Based Adversary Emulation Approach. IEEE Access.

[11] Razaque, Abdul, Bandar Alotaibi, Munif Alotaibi, Shujaat Hussain, Aziz Alotaibi, and Vladimir Jotsov (2022). Clickbait detection using deep recurrent neural network. Applied Sciences 12, no. 1: 504.

[12] He, Ke, Dan Dongseong Kim, and Muhammad Rizwan Asghar (2023). Adversarial machine learning for network intrusion detection systems: a comprehensive survey. IEEE Communications Surveys & Tutorials.

[13] Balla, Asaad, Mohamed Hadi Habaebi, Elfatih AA Elsheikh, Md Rafiqul Islam, and F. M. Suliman (2023). The Effect of Dataset Imbalance on the Performance of SCADA Intrusion Detection Systems. Sensors 23, no. 2: 758.

[14] Folino, Francesco, Gianluigi Folino, Massimo Guarascio, Francesco Sergio Pisani, and Luigi Pontieri (2021). On learning effective ensembles of deep neural networks for intrusion detection. Information Fusion 72: 48-69.

[15] Srilatha, Doddi, and Gopal K. Shyam (2021). Cloud-based intrusion detection using kernel fuzzy clustering and optimal type-2 fuzzy neural network. Cluster Computing 24, no. 3: 2657-2672.

[16] Varun, P., Ashokkumar, K. (2022). Intrusion detection system in cloud security using deep convolutional network. Appl. Math. Inf. Sci., 16, 581–588.

[17] Shyla, S.I., Sujatha, S.S. (2020). Cloud security: LKM and optimal fuzzy system for intrusion detection in cloud environment. J. Intell. Syst., 29, 1626–1642.

[18] Mishra, P.; Varadharajan, V.; Pilli, E.S.; Tupakula, U. (2018). Vmguard: A vmi-based security architecture for intrusion detection in cloud environment. IEEE Trans. Cloud Comput., 8, 957–971.

[19] Aoudni, Y., Donald, C., Farouk, A., Sahay, K.B., Babu, D.V., Tripathi, V., Dhabliya, D. (2022). Cloud security based attack detection using transductive learning integrated with Hidden Markov Model. Pattern Recognit. Lett., 157, 16–26.

[20] Velliangiri, S., Premalatha, J. (2019). Intrusion detection of distributed denial of service attack in cloud. Clust. Comput., 22, 10615–10623.

[21] Sathiyadhas, S. S., Soosai Antony, M. C. V. (2022). A network intrusion detection system in cloud computing environment using dragonfly improved invasive weed optimization integrated Shepard convolutional neural network. International Journal of Adaptive Control and Signal Processing, 36, 1060–1076.

[22] Geetha, T. V., Deepa, A. J. (2022). A FKPCA-GWO WDBILSTM classifier for intrusion

detection system in cloud environments. Knowledge-Based Systems, 253, 109557.

[23] Ayachi, Y., Mellah, Y., Saber, M., Rahmoun, N., Kerrakchou, I., Bouchentouf, T. (2022). A survey and analysis of intrusion detection models based on information security and object technology-cloud intrusion dataset. IAES International Journal of Artificial Intelligence, 11(4), 1607.

[24] Zhang, B., Zhu, J., Su, H. (2023). Toward the third generation artificial intelligence. Science China Information Sciences, 66(2), 121101.

[25] Amitay, Y., Bussi, Y., Feinstein, B., Bagon, S., Milo, I., Keren, L. (2023). CellSighter: a neural network to classify cells in highly multiplexed images. Nature Communications, 14(1), 4302.

[26] Jospin, L. V., Laga, H., Boussaid, F., Buntine, W., Bennamoun, M. (2022). Hands-on Bayesian neural networks—A tutorial for deep learning users. IEEE Computational Intelligence Magazine, 17(2), 29-48.

[27] Dasgupta, D., Akhtar, Z., Sen, S. (2022). Machine learning in cybersecurity: a comprehensive survey. The Journal of Defense Modeling and Simulation, 19(1), 57-106.

[28] Al-A'araji, N. H., Al-Mamory, S. O., Al-Shakarchi, A. H. (2021). Classification and clustering based ensemble techniques for intrusion detection systems: A survey. In Journal of Physics: Conference Series, 1818(1), 012106. IOP Publishing.

[29] Aslan, Ö., Aktuð, S. S., Ozkan-Okay, M., Yilmaz, A. A., Akin, E. (2023). A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions. Electronics, 12(6), 1333.