**Securing IoT Servers: Shallow vs. Deep Neural Network Architectures**

Enhancing IoT Server Security with Neural Networks: Shallow vs. Deep Architecture Analysis

Niranjan, M. G. N. W., Meegammana

Shilpa Sayura Foundation, niranjan.meegammana@gmail.com

Harinda,  Fernando

Sri Lanka Institute of Information Technology, harinda.f@sliit.lk

This study investigates the effectiveness of Shallow and Deep neural network architectures in detecting attacks on IoT application servers. It compares four models created using a Shallow Neural Network (NN) model with a single hidden layer of 512 neurons to a Deep NN model with 7 hidden layers ranging from 256 to 4 neurons. It uses a balanced UNSW-NB15 dataset, employing 20 and 40 features. The study found that the Deep model, which utilized 40 features, consistently outperformed other models, achieving an accuracy of 98.37%. Despite slightly longer prediction times and higher resource usage, the deep models prove suitable for high-scale IoT applications, while the shallow model remains appropriate for resource-constrained IoT servers. This research contributes to enhancing IoT security by employing NN solutions and proposes integrating the Deep model into next-generation firewall systems to protect higher-value IoT servers.  Future work involves exploring hybrid NN architectures for defending edge servers.

**CCS CONCEPTS •** Security and privacy~Intrusion detection systems • Computing methodologies~Artificial intelligence • Networks~Cyber-physical networks

**Additional Keywords and Phrases:** IoT Security, Application Servers, Shallow and Deep Architecture

1. **Introduction**

Application servers deployed as central nodes in IoT (Internet of Things) environments, such as healthcare facilities and smart cities, play a pivotal role in delivering critical services in the Industry 4.0 Digital Economy (Khang et al., 2024). The rapid proliferation of IoT systems for data aggregation and communication has created an internet-based massive attack surface and a prime target for malicious actors. The inherent vulnerabilities in IoT caused by the heterogeneity of devices and platforms, resource constraints, lack of security standards, and physical exposure, allow threat actors to exploit IoT networks to gain unauthorized access, disrupt services, steal sensitive information, and abuse devices to launch large-scale attacks (Alwarafy,2020). These attacks can disrupt critical services posing significant risks to businesses and economies, and potentially harming lives. Conventional security measures struggle to mitigate sophisticated attacks targeting IoT infrastructure different from traditional networks. Hence, IoT applications across sectors demand novel and robust security measures against escalating threats (Moustafa, 2022).

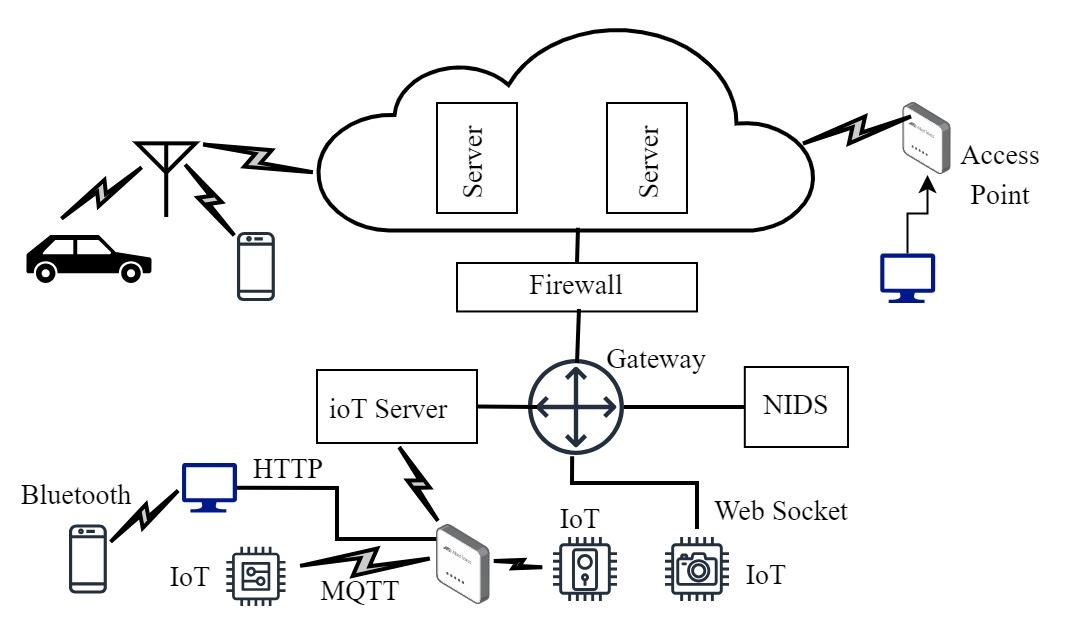


Figure 1: An IoT Environment

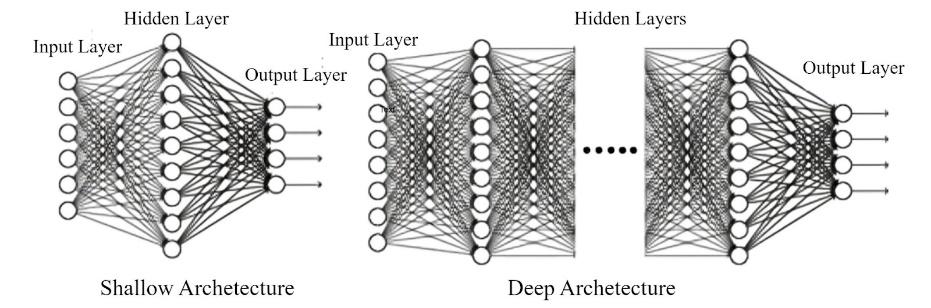
NNs offer capabilities of real-time network traffic analysis for identifying suspicious activities to help detect network attacks on IoT servers. Hence, NNs have become increasingly prominent in cybersecurity research. Studies have shown the effectiveness of NNs in detecting patterns of DDoS attacks, malware propagation, unauthorized access, injection attacks, and phishing attempts within network traffic. Hence, the adaptability, scalability, automation, and rapid response capabilities of NNs in protecting IoT environments have become an important area of research (Sagu et al., 2020). This investigation examines Shallow and Deep NN architectures to provide valuable insights into IoT server security, particularly within critical infrastructure. The subsequent sections of this paper include a literature review on NN-based IoT attack detection, methodology detailing research steps, results and discussion, conclusion on key findings stating future directions.

1. **Literature Review**

IoT application servers enable mission-critical service delivery. They also have broadened the online attack surface, endangering data security, privacy, and financial stability including human lives. The traditional security measures lack the capabilities to address IoT security's complexity and real-time nature, necessitating novel mechanisms. Research highlights the neural network potential of NNs protecting IoT environments (Moustafa et al., 2022).

* 1. **Neural Networks (NN)**

NNs consisting of interconnected nodes, allow information to flow from the input layer through hidden layers to the output layer. Neurons receiving inputs perform computations applying activation function to the weighted sum of inputs and generate output signals for the next layer, while learning from data through iterative processing, computing loss, updating weights, and ultimately producing an output through prediction neurons (Goodfellow et al., 2016). NNs are broadly categorized into Shallow and Deep architectures based on their depth, determined by the number of layers and neurons as shown in Figure 2.

  
Figure 2: Shallow and Deep Models (Sarker, 2022)

Shallow models use a single hidden layer with a higher number of neurons to reduce model complexity while learning complex patterns, but they struggle with hierarchical data representations. Deep models offer enhanced performance using more hidden layers to handle complex problems by capturing intricate patterns and hierarchical representations in data. Nonetheless, their computational complexity demands higher resources (Sarker, 2022; Bianchini and Scarselli 2014). Hyperparameters tuning is pre-configuring important NN settings before the training. They significantly influence the model architecture and learning process (Chollet,2018).

* 1. **Similar Work**

Aversano et al. (2021) systematically reviewed 69 studies focusing on NN approaches to enhance security in IoT environments. They analyze various attack types on IoT, NN architectures, and datasets, highlighting notable NN applications. They also provide a structured taxonomy to highlight research gaps and drawbacks in using NN methods for IoT security. One notable research gap identified is the absence of discussion model biases stemming from inadequate representation of real-world IoT scenarios in the data. Bianchini and Scarselli (2014) explore the theoretical comparison between Shallow and Deep architectures, arguing that the Deep NNs using multiple hidden layers outperform Shallow ones using a single hidden layer, due to their enhanced capability to represent complex functions for classification. Their paper introduces a topological complexity-based metric to compare the expressive power of Shallow and Deep architectures employing common activation functions. However, experimental validation with both Shallow and Deep models is necessary to support their findings. Sagu et al. (2020) highlight the escalating security challenges in the IoT landscape, outlining common threats and the shortcomings of conventional security measures in safeguarding resource-constrained IoT devices. They present theoretical aspects of NNs and underscore the potential for employing IoT security solutions, comparing them with traditional machine learning approaches. However, the research falls short in providing comparative results between NNs and classical machine learning methods in the IoT security context.

Choudhary and Kesswani (2020) utilized the KDD-Cup '99, NSL-KDD, and UNSW-NB15 datasets for network attack direction. Although the authors achieved a higher classification accuracy with the UNSW-NB15 dataset, they overlooked data balancing and hyperparameter tuning which significantly influence the final model performance. Ali et al. (2023) conducted a comprehensive review of 143 studies on NN approaches to address IoT security challenges. They identified various security IoT requirements, examined NN approaches to address them, and discussed various mechanisms in studies by comparing their performances. Their review highlights the significant diversity in security requirements across studies and sectors, reflecting the concerns on the evolving landscape of IoT security that require attention. Aggarwal (2023) compared the efficacy of Shallow and Deep neural network architectures to present distinct trade-offs in terms of model complexity, performance, and computational efficiency. Consequently, they state that employing them in the real world depends on specific task requirements, data complexity, and available computational resources.

* 1. **Research Gap**

Aversano et al. (2021), Sagu et al. (2020), and Ali et al. (2023) highlight the limitations of existing IoT network attack detection methods and propose NN approaches. However, their studies often overlook crucial steps in the Machine Learning (ML) pipeline essential for model optimization and performance improvement. Choudhary and Kesswani (2020) also acknowledged the significance of the UNSW-NB15 dataset, however, this study effectively addressed the data imbalance issue their study encountered. Sarker (2022), Bianchini et al., (2014), and Aggarwal (2023) delve into theoretical aspects of Shallow and Deep models, but their studies lack real-world experimentation of IoT attack detection. This study aims to bridge these gaps by systematically building and investigating optimized Shallow and Deep NN architectures to safeguard IoT application servers from network attacks.

**Methodology**

The study following the Machine Learning (ML) pipeline, a sequence of interconnected steps designed to produce higher-quality NNs (Hapke and Nelson, 2020), used a supervised learning approach to train, validate, and test  Shallow and Deep NNs employing a labeled dataset to make predictions on new inputs.

**Data Preparation**

The study utilized a subset of the UNSW-NB15 dataset, in which instances were classified as either attack or benign (David, 2019). The dataset was chosen due to its significant relevance to the attacks on IoT addressed by the study. After cleaning and removing categorical variables, we balanced the label feature using random undersampling to avoid model bias toward the benign class (Thabtah et al., 2020), which is an ethical consideration (Sutaria, 2022). Ultimately reducing the dataset to 186,000 instances, the study created two input datasets for model experimentation. The first input dataset consisted of 20 significant features, aimed to reduce model complexity for low-resource IoT environments. The second input dataset focusing on high-resource IoT environments comprised 40 numeric features. Subsequently, all data underwent scaling using the min-max scalar technique, bringing features within a common range of 0 to 1. Finally, the datasets were divided into three sets, training, testing, and validation (Goodfellow et al., 2016) in a 90:5:5 ratio for model development.

* 1. **Model Development**

The study designed two primary NN models. The Shallow NN model consisted of a single hidden layer with 512 neurons, aimed to capture complex patterns and relationships within data while maintaining simplicity and computational efficiency as shown in Figure 3.

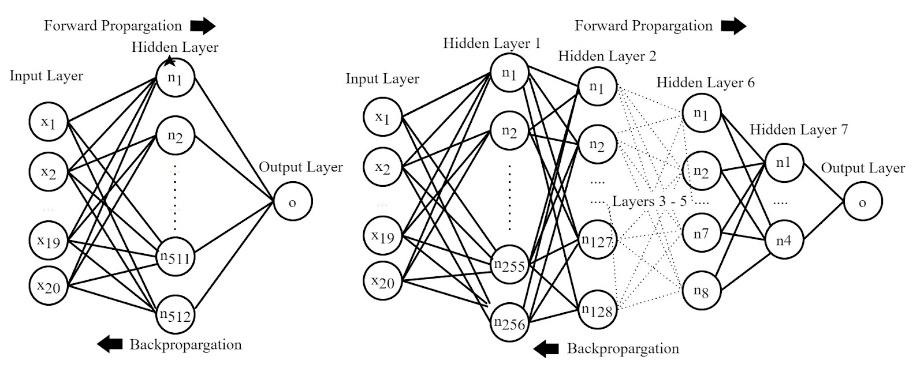


Figure 3: Primary Shallow and Deep ANN Model Designs

The Deep model comprises 7 hidden layers employing 256, 128, 64, 32, 16, 8, and 4 neurons respectively, allowing hierarchical feature extraction to capture intricate patterns and nuances in the data as shown in Figure 3. Ultimately, Shallow20, Deep20, Shallow40, and Deep40 experimental models as shown in Figure 4 were trained by applying 20 and 40 feature datasets to the primary Shallow and Deep architectures, to establish benchmarks across different model configurations.

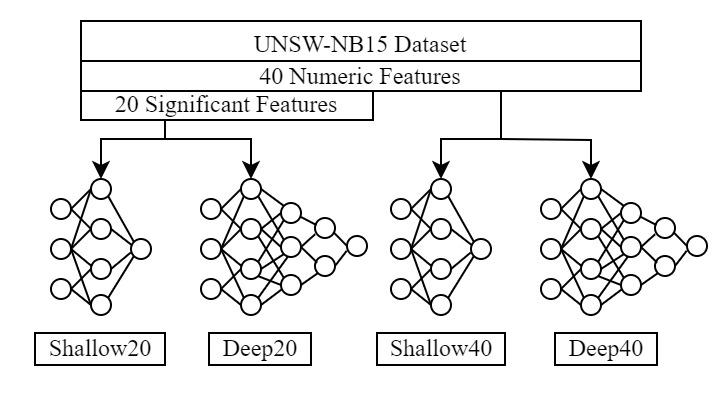


Figure 4: Architectures of Four Experimental Models

The hyperparameter tuning process enhances model performance by exploring various hyperparameter combinations to select the optimal configuration for the experimental model training. The study used Keras Tuner’s random search algorithm using 50 trials, employing 20% of the training set for internal validation during each tuning epoch, to monitor potential overfitting by experimental models (Chollet, 2018). Subsequently, we trained the four experimental models using their best hyperparameters, employing 20 and 40 feature training and validation sets. Each model was saved at the epoch reaching the minimum validation loss to control overfitting while enhancing model generalization (Goodfellow et al., 2016). After developing the models on Google Colab using Python, the study created four optimized executables using PyInstaller and executed them in an isolated environment to train, build, and test each model (Moustafa et al., 2022).

* 1. **Model Evaluation and Selection**

Each experimental model underwent rigorous testing and validation to assess its effectiveness in generalizing to new, unseen data, and unbiased evaluation was conducted to select the best-performing model using performance metrics derived from the confusion matrix, including accuracy, precision, recall, F1 score, and ROC-AUC curve. These metrics collectively offer a comprehensive evaluation of each model's ability to generalize, detect patterns, and make accurate predictions in classification tasks. The resource utilization by models during training and testing was evaluated by measuring the training and prediction times, including their CPU and memory usage. This assessment was crucial for evaluating real-time model performance and scalability (Moustafa et al., 2022). Analyzing the experimental model performance metrics and resource usage helped identify areas for model improvement and make informed decisions about deploying them within IoT environments to protect application servers.

1. **Results and Discussion**

The four experimental models initially showed relatively high accuracies. They further improved to a range of 0.94 to 0.98 after hyperparameter tuning, raising concerns of potential overfitting. Hence, requires close monitoring of the minimum validation loss during training. Table 1 shows the optimized model architectures after hyperparameter tuning. In a comparison of model complexities, theShallow20 model has the fewest (11,265) model parameters, followed by Deep20 (21,008), Shallow40 (49,313), and Deep40 (54,433).

Table 1: Optimized Architectures of the Models after Hyperparameter Tuning

| Configuration | Shallow20 | Shallow40 | Deep20 | Deep40 |
| --- | --- | --- | --- | --- |
| Model Parameters | 11265 | 49313 | 21008 | 54433 |
| Data Inputs | 20 | 40 | 20 | 40 |
| Hidden layers | 1 | 1 | 7 | 7 |
| Neurons per layer | 512 | 512 | 256, 128, 64, 32, 16, 8, 4 | 256, 128, 64, 32, 16, 8, 4 |
| activation\_layer1 | Relu | Tanh | tanh | ReLU |
| activation\_layer2 | - | - | tanh | ReLU |
| activation\_layer3 | - | - | ReLU | leaky\_relu |
| activation\_layer4 | - | - | ReLU | ReLU |
| activation\_layer4 | - | - | ReLU | ReLU |
| activation\_layer6 | - | - | ReLU | ReLU |
| activation\_layer7 | - | - | leaky\_relu | leaky\_relu |
| optimizer | Rmsprop | Adam | Adam | Adam |
| learning\_rate | 0.001 | 0.001 | 0.001 | 0.001 |
| weight\_initializer | he\_normal | he\_normal | he\_normal | glorot\_uniform |
| batch\_size | 256 | 64 | 8 | 16 |
| Output function | Sigmoid | Sigmoid | Sigmoid | Sigmoid |

* 1. **Training of Final Models**

Figure 5 shows the changes in accuracy and loss during training. The early stopping’s patience set to 50 epochs, allowed the training to halt when the validation loss was no longer improving.

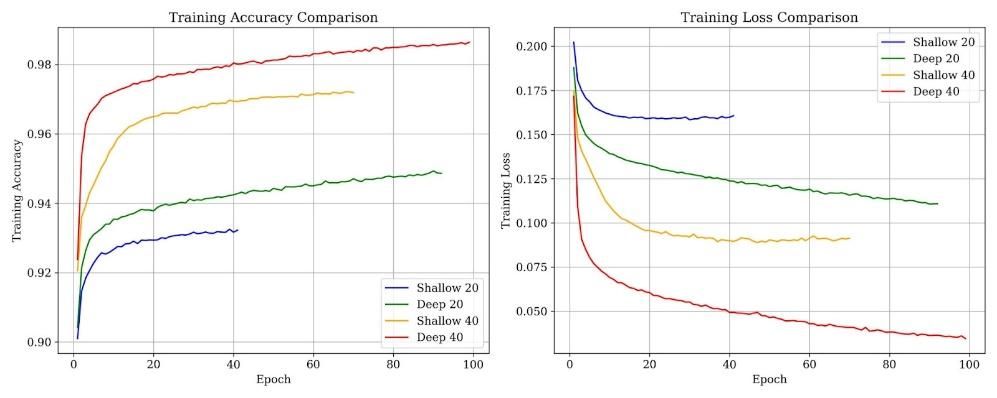


Figure 5: Changes in Accuracy and Loss of Four Models during Training

Table 4: Key Performance Metrics of Four Experimental Models

| Performance Data | Shallow20 | Deep20 | Shallow40 | Deep40 |
| --- | --- | --- | --- | --- |
| Training Time (S) | 289 | 902 | 512 | 962 |
| Maximum Validation Accuracy | 0.93 | 0.94 | 0.97 | 0.98 |
| End Epoc | 10 | 61 | 39 | 68 |
| Minimum Validation Loss | 0.166 | 0.130 | 0.088 | 0.057 |
| Prediction Time | 0.55 | 0.56 | 0.57 | 0.59 |
| Test Accuracy | 0.93 | 0.94 | 0.97 | 0.98 |
| Precision | 0.94 | 0.96 | 0.97 | 0.98 |
| Recall | 0.92 | 0.92 | 0.97 | 0.98 |
| F1 Score | 0.93 | 0.94 | 0.97 | 0.98 |
| ROC AUC Score | 0.986 | 0.990 | 0.996 | 0.998 |
| Average Precision | 0.987 | 0.991 | 0.996 | 0.998 |

* 1. **Model Comparison**

As shown in Table 4, the four experimental models achieved relatively high accuracies between 0.93 and 0.98, alongside high precision, recall, F1 score, and ROC AUC scores, indicating their strong predictive capabilities. Among them, the Deep40 model demonstrated superior performance across multiple metrics, achieving the highest test accuracy of 0.98, validation accuracy of 0.98, precision of 0.981, F1 score of 0.98, ROC AUC score of 0.998, and average precision of 0.998, while recording the minimum validation loss of 0.057. Hence, the Deep40 model consistently outperformed the other models, indicating its robustness in making accurate predictions and its exceptional ability to generalize to unseen data. Exhibiting the highest number of true positives (4572) and true negatives (4537), the Deep40 model demonstrates its effectiveness in correctly classifying both positive and negative instances. Moreover, showing the lowest number of false positives (78) and false negatives (113), showcases its accuracy in making predictions with high sensitivity and specificity.

* 1. **Resource Utilization**

The Shallow20 model with simpler architecture and lower computational complexity, achieved faster training time. This factor is insignificant if the model training time is not a critical factor for the task. The difference of prediction times between the Shallow20 (0.55 seconds) and the Deep40 (0.59 seconds) model may appear small. However, even minor differences can be significant in real-time circumstances where predictions are made continuously or in batches.

Table 4: Key Performance Metrics of Four Experimental Models

| Performance Data | Shallow20 | Deep20 | Shallow40 | Deep40 |
| --- | --- | --- | --- | --- |
| Model Size (Kb) | 110 | 190 | 648 | 708 |
| Prediction Time (s) | 0.55 | 0.56 | 0.57 | 0.59 |
| CPU Usage % | 0.25 | 0.26 | 0.30 | 0.32 |
| Memory Usage (GB) | 0.016 | 0.017 | 0.019 | 0.021 |

As shown in Table 4, the single layer Shallow20 model exhibited lower CPU and memory usage compared to deep models with multiple layers and more parameters. The Shallow20 model requires less RAM for storing and processing model parameters and intermediate computations, contributing to their lower memory consumption. During model inference, Shallow20 model offers advantages for deployment in resource-constrained environments and time-sensitive real-time applications, where timely responses are necessary to mitigate security threats effectively. The Deep40 model exhibits superior performance and better overall model quality, particularly effective for classification tasks in well-resourced IoT servers. By recognizing these differences and trade-offs, researchers and practitioners can make informed decisions regarding model selection and deployment strategies for enhancing security mechanisms in IoT systems.

1. **Conclusion**

This research focused on enhancing IoT server security using Shallow and Deep models employing two datasets followed a systematic approach involving iterative improvements. The investigation revealed that despite their higher computational demands, Deep NNs demonstrate superior accuracy and discriminative capabilities for detecting network attacks on IoT servers. Conversely, Shallow models, with lower resource requirements, remained suitable for low-resource IoT environments, although with slightly reduced accuracy. Furthermore, the analysis showed the significance of feature selection, data balancing, and hyperparameter tuning for optimizing model performance. In conclusion, the study underscores the practical implications of employing NNs architectures to enhance IoT server security. The choice between Shallow and Deep models depends on specific application requirements, considering trade-offs in computational resources and model complexity, inference speed, interpretability, scalability, and cost. This research significantly advances IoT security by demonstrating the effectiveness of Shallow and Deep NN architectures in detecting attacks on IoT application servers, proposing a practical deployment strategy by integrating the Deep40 model into next-generation firewall systems to protect higher-value IoT servers, and suggesting future directions for exploring hybrid models in defending edge environments.

**REFERENCES**Aggarwal, Charu C. 2023. *Neural Networks and Deep Learning*. Springer Nature.

Alwarafy, Abdulmalik, Khaled A. Al-Thelaya, Mohamed Abdallah, Jens Schneider, and Mounir Hamdi. 2020. “A Survey on Security and Privacy Issues in Edge Computing-Assisted Internet of Things.” *IEEE Internet of Things Journal*, 1–1. https://doi.org/10.1109/jiot.2020.3015432.

Ali, Yasir, Habib Ullah Khan, and Muhammad Khalid. 2023. “Engineering the Advances of the Artificial Neural Networks (ANNs) for the Security Requirements of Internet of Things: A Systematic Review.” *Journal of Big Data* 10 (1). https://doi.org/10.1186/s40537-023-00805-5.

Aversano, Lerina, Mario Luca Bernardi, Marta Cimitile, and Riccardo Pecori. 2021. “A Systematic Review on Deep Learning Approaches for IoT Security.” *Computer Science Review* 40 (May): 100389. https://doi.org/10.1016/j.cosrev.2021.100389.

Bianchini, Monica, and Franco Scarselli. 2014. “On the Complexity of Neural Network Classifiers: A Comparison between Shallow and Deep Architectures.” *IEEE Transactions on Neural Networks and Learning Systems* 25 (8): 1553–65. https://doi.org/10.1109/tnnls.2013.2293637.

Chollet, François. 2018. *Deep Learning with Python*. Shelter Island (New York, Estados Unidos): Manning, Cop.

Choudhary, Sarika, and Nishtha Kesswani. 2020. “Analysis of KDD-Cup’99, NSL-KDD and UNSW-NB15 Datasets Using Deep Learning in IoT.” *Procedia Computer Science* 167: 1561–73. https://doi.org/10.1016/j.procs.2020.03.367.

David, Wells. 2019. “UNSW-NB 15 Dataset.” Www.kaggle.com. 2019. https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

Hapke, Hannes, and Catherine Nelson. 2020. *Building Machine Learning Pipelines : Automating Model Life Cycles with Tensorflow*. Beijing ; Boston ; Farnham ; Sebastopol ; Tokyo O’reilly.

Khang, Alex, Vugar Abdullayev, Vladimir Hahanov, and Vrushank Shah. (2024) 2024. *Advanced IoT Technologies and Applications in the Industry 4.0 Digital Economy*. CRC Press.

Moustafa, Nour, Mohamed Abdel-Basset, and Reda Mohamed. 2022. *Deep Learning Approaches for Security Threats in IoT Environments*. Wiley-IEEE Press.

Sagu, Amit, Nasib Singh Gill, and Preeti Gulia. 2020. “Artificial Neural Network for the Internet of Things Security.” *International Journal of Engineering Trends and Technology* 68 (11): 129–36. https://doi.org/10.14445/22315381/ijett-v68i11p218.

Sarker, Iqbal H. 2022. “AI-Based Modeling: Techniques, Applications and Research Issues towards Automation, Intelligent and Smart Systems.” *SN Computer Science* 3 (2). https://doi.org/10.1007/s42979-022-01043-x.

Sutaria, Niral. 2022. “Bias and Ethical Concerns in Machine Learning.” ISACA. August 29, 2022. https://www.isaca.org/resources/isaca-journal/issues/2022/volume-4/bias-and-ethical-concerns-in-machine-learning.

Thabtah, Fadi, Suhel Hammoud, Firuz Kamalov, and Amanda Gonsalves. 2020. “Data Imbalance in Classification: Experimental Evaluation.” *Information Sciences* 513 (March): 429–41. https://doi.org/10.1016/j.ins.2019.11.004.