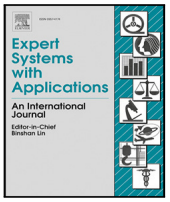
Expert Systems With Applications 256 (2024) 124982

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



Anomaly and intrusion detection using deep learning for software-defined networks: A survey

Vitor Gabriel da Silva Ruffo [a], Daniel Matheus Brandão Lent [b], Mateus Komarchesqui [a], Vinícius Ferreira Schiavon [a], Marcos Vinicius Oliveira de Assis [c], Luiz Fernando Carvalho [d], Mario Lemes Proença Jr. [a,*]

[a] Computer Science Department, State University of Londrina, Londrina, Paraná, Brazil
[b] Electrical Engineering Department, State University of Londrina, Londrina, Paraná, Brazil
[c] Engineering and Exact Department, Federal University of Paraná, Paraná, Brazil
[d] Federal Technological University of Paraná, Paraná, Brazil

ARTICLE INFO

Keywords:
Literature review
NIDS
SDN
Datasets
Deep learning
Hyperparameters

ABSTRACT

Software-Defined Networks (SDN) represent an adaptable paradigm for dealing with network users' dynamic demands. Confidentiality, integrity, and availability are fundamental pillars for the security of the networks, which are often targeted by cyberattacks. The scientific community has been recently exploring deep learning to implement Network Intrusion Detection Systems (NIDS) against network attacks. In this survey, we aim to present an empirical literature review on state-of-the-art NIDS based on deep learning for defending SDNs. The essential steps to develop such systems are carefully examined: benchmark datasets, data preprocessing, deep learning modeling, hyperparameter tuning, and performance evaluation. There has been a growing trend in published works since 2021, underpinning the importance of the research field, which is still active and under investigation. We support the development of the area by discussing the identified open issues and future research directions.

1. Introduction

Computer networks have been as essential as energy and water services for the current society. Broadband, low-latency networks such as Ethernet and 5G over the TCP/IP architecture enable numerous applications. Examples include communication, cloud computing, distance education, digital banking, online shopping, on-demand entertainment, AI assistance, autonomous vehicles, and Internet of Things. The networks' serviceability has caused the number of connected devices to grow exponentially in recent decades. Consequently, the volume of network traffic and stored personal user information has drastically increased (Gupta, Jindal, & Bedi, 2021; Imrana, Xiang, Ali, & Abdul-Rauf, 2021; Javed, Khayat, Elghariani, & Ghafoor, 2023).

Traditional networks lack adaptability and strive to meet users' increasingly complex and dynamic requirements. Conversely, Software-Defined Networks (SDN) emerged as a flexible paradigm for simplifying network management. Network control is decoupled from the data plane and moved to a central entity, the controller. This mechanism controls the whole network, facilitating network orchestration and policy enforcement (Polat, Türkoˇglu, Polat, & Şengür, 2022; Sattari et al., 2022). Service providers like Google and Microsoft have been exploring the SDN's advantages over the traditional paradigm to implement their global wide area networks (Zhang et al., 2023).

Malicious agents aim to disrupt network services' confidentiality, integrity, or availability to achieve their goals (Aydın, Orman, & Aydın, 2022; Gupta, Tripathi, & Grover, 2022; Mustapha et al., 2023). Cyberattacks may compromise the services' functioning, potentially leading to money and reputation losses and harming human lives (Friha et al., 2022; Hidalgo et al., 2022).

Network Intrusion Detection System (NIDS) is among the most popular security solutions that have been studied to identify cyberattacks on network services promptly (Friha et al., 2023; Qazi, Imran, Haider, Shoaib, & Razzak, 2022; Udas, Karim, & Roy, 2022). NIDS are positioned strategically on a network, gathering and analyzing all incoming traffic. Network administrators are notified to counter the problem whenever a traffic anomaly is identified. Some solutions

* Corresponding author.
E-mail addresses: vitor.ruffo19cc@uel.br (V.G. da Silva Ruffo), matheus.daniel@uel.br (D.M. Brandão Lent), mateus.komarchesqui@uel.br (M. Komarchesqui), vinicius.schiavon@uel.br (V.F. Schiavon), marcos.assis@ufpr.br (M.V.O. de Assis), luizfcarvalho@utfpr.edu.br (L.F. Carvalho), proenca@uel.br (M.L. Proença Jr.).

https://doi.org/10.1016/j.eswa.2024.124982
Received 11 May 2024; Received in revised form 30 June 2024; Accepted 1 August 2024
Available online 5 August 2024

even include a mitigation module to attenuate the detected anomalies automatically (Illy & Kaddoum, 2023; Myneni, Chowdhary, Huang, & Alshamrani, 2022).

The scientific community has been studying novel strategies to implement efficient and robust NIDS. Deep Learning (DL) is a state-of-the-art solution that is applied to solve intrusion detection. DL represents a class of mathematical models developed to identify complex patterns in high-dimensional data (Hairab, Said Elsayed, Jurcut, & Azer, 2022; Yousuf & Mir, 2022). Deep Learning models are suitable for handling network traffic since they can effectively process high volumes of data. Examples include the Long Short-Term Memory (LSTM) (Tayfour et al., 2023) and Gated Recurrent Unit (GRU) (Assis, Carvalho, Lloret, & Proença, 2021) networks, which can discover long-term dependencies in sequential data. Also, Convolutional Neural Network (CNN) (Fox & Boppana, 2023) models are usually applied to identify spatial patterns in the processed input. Autoencoder (AE) (Fouladi, Ermiş, & Anarim, 2022) networks compress data to a fundamental low-dimensional representation. Generative Adversarial Networks (GAN) (Shu, Zhou, Zhang, Du, & Guizani, 2021) efficiently capture training data statistical distribution.

This survey article proposes a comprehensive, empirical review of state-of-the-art deep learning-based intrusion detection for SDNs. None of the reviewed related works simultaneously addressed benchmark datasets, data preprocessing, deep learning modeling, hyperparameter tuning, and performance evaluation. The present study aims to fill this research gap by discussing these critical topics often employed jointly in developing such systems. Combined with a holistic view of the current intrusion detection scenario, an up-to-date taxonomy of deep learning model applications, public datasets, and dataset generation tools is conducted. We support innovation in the study field by pinpointing the research directions that require further investigation.

The contributions of this work are six-fold:

- Present the most used datasets, attacks, and tools for developing and evaluating intrusion detection systems to inform researchers of standard and underexplored data sources;
- List the preprocessing steps commonly applied to NIDS input data and their frequency for newcomers' awareness;
- Introduce an up-to-date application taxonomy of state-of-the-art deep learning models, highlighting the established and superficially explored ones;
- Discuss how state-of-the-art works usually optimize deep learning models hyperparameters;
- Quantify the most popular performance metrics among the scientific community to benchmark and compare intrusion detection systems;
- Identify the knowledge gaps in the area, pointing out future research directions.

The remainder of this survey article is structured as follows: Section 2 reviews the related works; Section 3 briefly introduces the background theory; Section 4 discusses the methodology applied to conduct this paper's research; Section 5 summarizes the found results; Section 6 points out the open issues and future research directions within the area of study; and Section 7 elaborates on the conclusions of the work.

2. Related works

In this section, we review relevant studies from leading journals that survey intrusion detection using Machine and Deep Learning. The general outline and particular insights are discussed for each work. We conclude by pointing out how our work builds upon them, providing an up-to-date, thorough view of the state of the art.

Yang et al. (2022) proposed a systematic and comprehensive literature that surveyed 119 highly cited papers on anomaly-based network intrusion detection approaches. The authors covered the application of IDS in the so-called Internet (Traditional Networks), Internet of Things (IoT), Software-Defined Networks, and Industrial Control Networks (ICN). This study highlighted that 70% of the analyzed papers apply to conventional networks rather than a specific application domain (e.g., SDN, and IoT), indicating that security in traditional networks is an important research topic. It is also pointed out that the lack of datasets or the confidentiality of industrial data are key factors limiting the development of IDSs in SDN and ICN, respectively. In addition, preprocessing data proved necessary for building high-performance detectors, and traditional supervised machine learning is the most applied technology. They also present the datasets and performance metrics usually utilized for experimentation.

de Souza, Westphall, Machado, Loffi, Westphall, and Geronimo (2022) conducted a systematic literature review that addresses 108 studies on intrusion detection and mitigation in fog computing and IoT-based environments. The authors categorize IDSs according to their detection analysis approach as either signature-based or behavior-based and deployment strategy, distinguishing between Network-based IDS and Host-based IDS. Due to the resource constraints of IoT devices, implementing monitoring systems in the fog is considered one of the most promising strategies in this study. The Machine Learning techniques evaluated were divided into supervised, unsupervised, semi-supervised, ensemble, and reinforcement learning. Besides that, collaborative IDSs approaches, post-detection countermeasures, evaluation strategies, and leading public datasets were also pointed out.

Nuaimi, Fourati, and Hamed (2023) published a systematic literature review that presents novel learning-based approaches for intrusion detection in the Industrial Internet of Things (IIoT) in papers between 2017 and 2022. The authors employ the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyzes) methodology to facilitate reporting systematic reviews and meta-analyses. This work surveys the IIoT domain in the selected articles, provides a brief background on different systems and architectures for such domain, and exposes the addressed intrusion detection methods, datasets, metrics, and attacks. The security threats are organized into five categories based on the security aspect they compromise. Also, the reviewed ML/DL models are separated into supervised, unsupervised, and reinforcement learning, and the IDS's placement (centralized, distributed, or hybrid) and approach (signature or anomaly-based) are considered. This survey concluded that deep learning models are prone to be better than machine learning at detecting network attacks and suggests that Federated Learning and Blockchain should be explored further. Finally, it is stated that most IIoT devices are battery-powered; therefore, power consumption must be taken into account.

M. and Sethuraman (2023) present a comprehensive survey of Machine and Deep Learning-based malware detection approaches applied to Sandboxing, Android, iOS, Windows, IoT, APT, and Ransomware. The authors noted that most malicious software exploitation targets the Windows operating system, which is classified into two areas: bulk automated for widespread malware and sophisticated specialized malware. According to this work, these threats can be detected statically by examining the executable file of the potentially malicious software or dynamically by evaluating behaviors at executing the malicious program in a virtual sandboxed environment. It is inferred that static analysis techniques cannot correctly handle malware with obfuscation but provide a better detection for familiar malware families. Hence, the dynamic or a hybrid of both approaches is employed for more efficient results.

Taheri, Ahmed, and Arslan (2023) surveyed the utilization of Deep Learning algorithms for the security of Software-Defined Networks. The authors separate Deep Learning into supervised discriminative learning, unsupervised generative learning, and hybrid deep learning, like transfer and reinforcement learning. This work also analyses the architectural vulnerabilities of the SDN, organizing which type of attack thrives in each plane. It provides an in-depth analysis of the

datasets used throughout security papers and reviews the commonly applied performance metrics. Finally, it is pointed out that the need for large-scale, high-quality datasets majorly affects the overall performance of Deep Learning models that have notoriously lengthy and resource-intensive training phases.

Abdulganiyu, Ait Tchakoucht, and Saheed (2023) compare the signature, anomaly, and hybrid-based Netword Intrusion Detection System approaches. The authors analyze every Machine and Deep Learning model applied to each approach, collecting the most used evaluation metrics and datasets. Thus, it is pointed out that anomaly-based systems are the majority of research in NIDS. They presented that most papers evaluate the proposed systems with simulated datasets, cover only a fraction of the system, use biased parameters, and report questionable results, which may compromise the application of such systems in the real world.

Bilot, Madhoun, Agha, and Zouaoui (2023) surveyed the different types of graph-based learning employed in network and host intrusion detection systems, reviewing current methods and datasets. The authors also evaluate the robustness of IDS based on Graph Neural Network (GNN) against adversarial attacks. Even though GNN is relatively neglected in cybersecurity, it holds great promise, especially compared to traditional ML and most DL models. This method can learn and detect complex attacks and transcend obfuscation strategies, proving its strong suitability for security systems.

Melis, Sadi, Berardi, Callegati, and Prandini (2023) conducted a systematic literature review of offensive and defensive security solutions for Software-Defined Networks. In this work, the authors gathered 466 relevant publications and submitted them to quantitative and qualitative analysis to aggregate the literature. As expected, the number of paper distributions has increased from 2015 to 2021. The correlations between the articles' keywords and research questions highlight the connected topics and present open challenges. This analysis evidenced the focus on anti-DoS techniques, the absence of an exhaustive threat model, and the need for more variety of technologies.

Sabeel, Heydari, El-Khatib, and Elgazzar (2024) surveyed recent techniques for detecting atypical, polymorphic, and unknown network attacks using Deep Learning. The authors separated the detection methods into misuse, anomaly, hybrid, adversarial generation, transfer learning, and reinforcement-based approaches, presenting background on the reviewed attacks and the often-used datasets. They concluded that many implementations rely on supervised learning, which is restricted to a network traffic distribution and may perform poorly in practical scenarios. Only a few labeled and quality network security datasets are available, and most training data have many features that demand an attribute extraction phase, making the IDS more complex. It is presented that a good evaluation metric selection is paramount to avoid biased results alongside explainability techniques.

Ozkan-Okay, Akin, Aslan, Kosunalp, Iliev, Stoyanov, and Beloev (2024) reviews Machine Learning, Deep Learning, Reinforcement Learning, and AI tools as ChatGPT devoted to cybersecurity, presenting each approach categorized into different learning algorithms, models, and applications. This work addresses malware and intrusion detection, vulnerability assessment, data quality, interpretability, and adversarial attacks. The authors point out that voluminous datasets are fundamental for Machine and Deep Learning techniques to thrive against threats, and their performance in real-world scenarios can be affected by poor training. Conversely, Reinforcement Learning, especially combined with other strategies, has shown significant potential for cybersecurity defense and should be further explored. Finally, ChatGPT-like tools hold great promise as valuable resources for enhancing security.

Table 1 summarizes the related works based on their coverage of fundamental topics within deep learning-based intrusion detection. Note that we only considered a specific topic to be covered by a work when the authors discussed it comprehensively. Many studies presented an analysis of both datasets and evaluation metrics. Only one work conducted a detailed review of preprocessing methods, and none discussed hyperparameter tuning. Despite the critical importance of these subjects, no reviewed study simultaneously addressed the most used benchmark datasets, data preprocessing methods, hyperparameter tuning techniques, and performance evaluation metrics. We bridge this gap by conducting an updated, in-depth review of the state of the art, embracing all elementary topics.

3. Background

3.1. Deep learning

Deep Learning represents a subset of Machine Learning. It comprises mathematical models that are made of multiple layers of abstraction. These models mimic the human brain and can find complex patterns in large volumes of data without human intervention (Elsayed, Hamada, Abdalla, & Elsaid, 2023; Shaji, Jain, Muthalagu, & Pawar, 2023).

3.1.1. Models

Fig. 1 illustrates examples of common Deep Learning models, including Deep Neural Network (DNN), CNN, AE, GAN, LSTM, GRU, and Graph Neural Network (GNN). These models are based on the concept of collections of artificial neurons that collectively calculate a non-linear function of the input. The DNN (Cil, Yildiz, & Buldu, 2021), or Multilayer perception (MLP), is the most basic neural network within DL, comprising a stack of layers of artificial neurons. Each layer progressively calculates higher-level patterns from input data, producing a corresponding output.

CNN (Nguyen & Le, 2023) is a neural network specialized to discover spatial patterns in data through convolutional and pooling operations. It is usually less computationally complex than DNN since its internal architecture utilizes fewer connections between neurons, reducing the trainable parameters.

The AE (Ding, Kou, and Wu, 2022) is a neural network with a structure similar to DNN aimed at compressing and decompressing data accurately. It is composed of two internal subnetworks, called encoder and decoder. The former is responsible for calculating a low-dimensional representation of input data. The latter takes the codified input and reconstructs it as precisely as possible.

The GAN (Siniosoglou, Radoglou-Grammatikis, Efstathopoulos, Fouliras, & Sarigiannidis, 2021) model is formed by two inner networks with opposite training objectives: the discriminator and the generator. The discriminative one is a binary classifier that aims to correctly distinguish between samples coming from inside and outside its training set. The generative strives to create synthetic samples that resemble the training set, fooling the discriminator into misclassifying them. A well-trained GAN model's generator can produce convincing fake samples that follow the training data distribution.

None of the previously described models can remember information about past processed samples. Conversely, the LSTM (Sahu et al., 2022) and GRU (Ahmad, Wan, & Ahmad, 2023) networks are improved versions of the traditional Recurrent Neural Network (RNN), capable of learning long-term temporal dependencies between data instances. They are composed of neurons that feedback on their own output. This design architecture allows the neurons to retain information about previously processed data, which can be leveraged when processing new incoming samples. These recurrent models generally demand more computing requirements than the previous networks, since they utilize neurons with a more complex internal architecture.

The GNN (Caville, Lo, Layeghy, & Portmann, 2022) uses a graph's topological structure to aggregate neighboring node features. Each node encapsulates its intrinsic properties and the context provided by its neighbors. Through a process known as message passing, each node iteratively exchanges information with its neighbors, and the aggregated information is used to update its representation. The resulting node embeddings are n-dimensional vector representations that capture topological and node-specific information. This process can be extended to edges and entire graphs.

Table 1
Related surveys on deep learning-based intrusion detection.

Reference	Benchmark datasets	Data preprocessing	Hyperparameter tuning	Performance evaluation
Yang et al. (2022)	✓	✓	✗	✓
de Souza et al. (2022)	✓	✗	✗	✗
Nuaimi et al. (2023)	✓	✗	✗	✓
M. and Sethuraman (2023)	✗	✗	✗	✗
Taheri et al. (2023)	✓	✗	✗	✓
Abdulganiyu et al. (2023)	✓	✗	✗	✓
Bilot et al. (2023)	✓	✗	✗	✗
Melis et al. (2023)	✗	✗	✗	✗
Sabeel et al. (2024)	✓	✗	✗	✓
Ozkan-Okay et al. (2024)	✗	✗	✗	✗
Our work (2024)	✓	✓	✓	✓

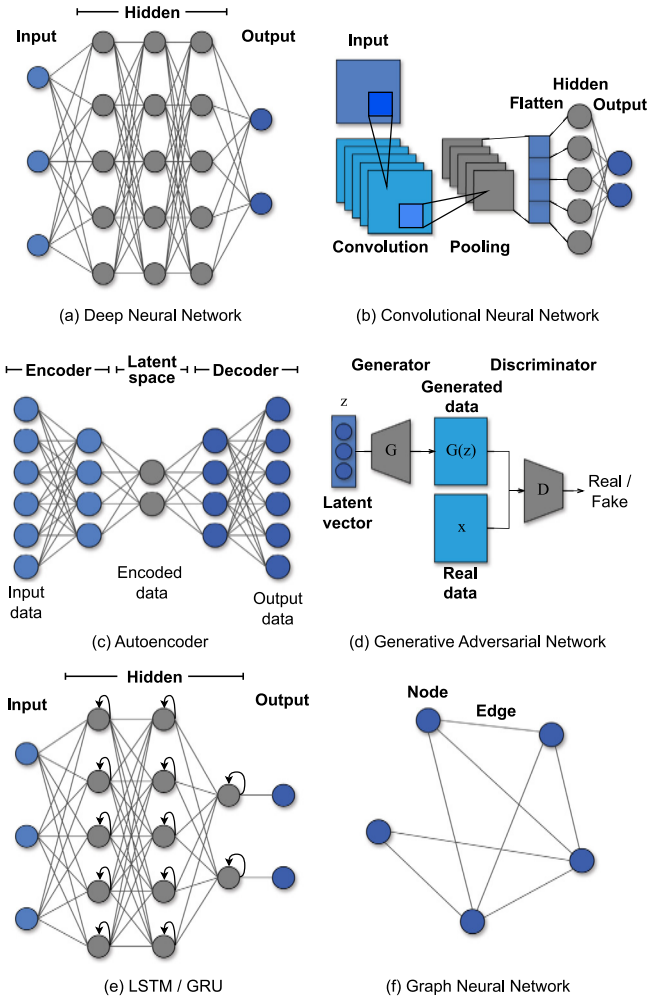


Fig. 1. Common deep learning models: (a) Deep Neural Network, (b) Convolutional Neural Network, (c) Autoencoder, (d) Generative Adversarial Network, (e) LSTM/GRU, and (f) Graph Neural Network.

3.1.2. Learning paradigms

DL models can be categorized as supervised, unsupervised, federated, or reinforcement learning based on how the learning process occurs. A supervised model is trained on labeled data to approximate a function that accurately maps input samples to their correct labels (Fouladi et al., 2022; Nadeem, Goh, Aun, & Ponnusamy, 2023). Labels are continuous or discrete values associated with each data sample that provide information about them. An illustrative example of supervised learning is categorizing cat and dog images. Each data sample comprises a cat or dog picture and a discrete label indicating the depicted animal. By training on labeled data first, a CNN classifier can learn to accurately assign labels to unseen cat and dog images, effectively discerning the type of animal presented in each analyzed input.

In real-world scenarios, it is common for data to be unlabeled, as labeling a dataset is very costly (Yang, Song, King, & Xu, 2023). Unsupervised learning-based models are utilized to find underlying, hidden structures in unlabeled data (Van Engelen & Hoos, 2020). For instance, the GAN model is widely applied to learn the statistical distribution of an unlabeled dataset. The trained network can generate new synthetic data samples not in the training set but fitting its distribution.

Federated learning is an implementation paradigm that may be applied over supervised or unsupervised models. It addresses data privacy and distribution challenges in deep learning. Unlike a traditional centralized approach, where data is aggregated on a single server for training, federated learning enables model training directly on decentralized data sources, like mobile devices or edge servers (Duy et al., 2021). Each device maintains control over its data, and only model updates, rather than raw data, are exchanged with a central server. This decentralized approach to training allows for collaborative model learning across distributed devices while mitigating privacy risks associated with centralized data collection (Houda, Hafid, & Khoukhi, 2023).

Reinforcement learning introduces a dynamic interaction between an agent and its environment (Shukla, Maheshwary, Subramanian, Shilpa, & Varma, 2023). The model does not rely on labeled or unlabeled data. Instead, it learns through trial and error, taking actions in an environment and receiving rewards or penalties based on the consequences of its actions. This feedback loop enables the agent to improve its decision-making policy iteratively, ultimately maximizing cumulative rewards over time (Kim et al., 2022).

3.2. Software-defined networks

The Software-Defined Network paradigm removes the network control decision from the forwarding devices, moving it to a logical central entity, the controller. Control rules and network policies are defined and updated only on the controller, which installs them into the network switches and routers automatically. This design approach allows for a centralized view, programmability, and simplified management of the whole network (Kumar et al., 2023; Zhou, Zheng, Jia, & Shu, 2023).

Fig. 2 illustrates the SDN architecture and network planes. The data plane represents the forwarding devices responsible for directing incoming data to the correct destination. The controller resides in the control plane. It utilizes the southbound API to send control rules to the forwarders or to gather traffic statistics from them. The application plane comprises the software needed for managing the network. The administrators utilize the northbound API provided by the controller to specify and automate the desired network policies utilizing a high-level programming language (Lent et al., 2022; Liu et al., 2022).

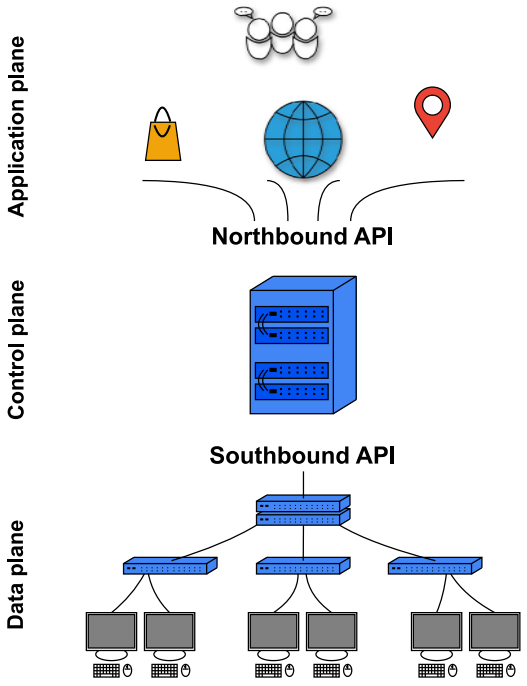


Fig. 2. SDN architecture and network planes.

As traditional networks grow, new devices and rules need to be introduced. However, the introduction process is hindered by different factors, such as manufacturer software and compatibility between devices. The management functionalities are limited to the forwarding devices' software, which can become expensive to change for larger networks. Software-defined networks do not have this problem since they frequently utilize an open protocol to control each device. For this reason, management systems such as intrusion detection and mitigation can be attached to a network without requiring major reconfiguration, which would not always be possible for traditional ones, such as the Internet, Internet of Things, and Industrial Control System networks. Besides, the controller can request traffic flow data from its switches, which can feed an intrusion detection module through the northbound interface. Similarly, a mitigation module can quickly implement new forwarding rules through the controller to every device in time to avoid significant consequences from attacks (Choobdar, Naderan, & Naderan, 2021).

SDN's centralized nature makes the controller a point where malicious attacks may focus. For example, an attacker may conduct a Distributed Denial of Service (DDoS) attack to take the controller down, jeopardizing the whole network operation and affecting the services' availability (Long & Jinsong, 2022). The SDN can also be targeted by other types of attacks, such as scanning, to obtain information about the target network and allow the execution of other attacks; hijacking, to gain total control over a network element; tampering, to compromise the integrity of network data; man-in-the-middle, to listen to the communication between the controller and the data plane, hazarding their confidentiality (Li, Meng, & Kwok, 2016), (Nisar et al., 2020). The scientific community has been studying and implementing efficient protection mechanisms to safeguard the confidentiality, integrity, and availability of SDN services (Assis et al., 2021).

3.3. Intrusion detection

The definition of anomaly is a core concept in intrusion detection. Different works may have contrasting considerations on anomalies and build their systems accordingly. For example, Siniosoglou et al. (2021) define anomaly as any malicious activity, such as denial of service attacks, since they are deliberately harmful. This definition excludes legitimate abnormal events like flash crowds. Some works consider any deviation from the statistical baseline observed from legitimate traffic as an anomaly (Fouladi et al., 2022; Novaes, Carvalho, Lloret, & Proença, 2021; Zavrak & Iskefiyeli, 2023).

Efficient detection of attacks is crucial to defend computer networks and maintain their services online, supporting availability, confidentiality, and integrity. Network Intrusion Detection Systems are state-of-the-art solutions that have been studied and developed to counter such threats (Huang, Ye, Hu, & Wu, 2023; Sood et al., 2023; Xue & Jing, 2023). These systems collect and analyze traffic data periodically. The NIDS generates a warning alarm for the administrators if any uncommon behavior is identified.

There are two common design strategies for implementing NIDS: signature-based and anomaly-based. The first one keeps a database of known attack signatures. The system analyzes incoming traffic, identifying when an observed pattern matches a stored signature. In this case, an anomaly is detected and reported. This approach usually uses supervised learning to extract signatures of each type of anomaly. The main drawback of this method is that it cannot recognize zero-day attacks whose signatures are unknown, augmenting its false negative rate (Hairab et al., 2022). The anomaly-based design strategy, or anomaly detection, builds a model representing legitimate traffic behavior. Incoming traffic is compared with the normality model, and if their difference exceeds a tolerance threshold, an alarm is raised. This design allows for the detection of unknown attacks using unsupervised or semi-supervised learning. Its principal disadvantage is that it may interpret normal behavior variations as anomalies, increasing its false alarm rate (Gupta, Jindal, & Bedi, 2022).

The anomaly score is a concept some authors apply when implementing anomaly-based systems. It represents the degree of anomalousness of a traffic sample evaluated by the IDS. This value can be obtained through the raw output of a deep learning model or calculation of some loss function, for example. Hu et al. (2022) take the output of a DNN and calculate its Euclidian distance to a fixed point, obtaining the anomaly score of the evaluated sample. Shu et al. (2021) define the anomaly score as a linear combination of a Bidirectional GAN's reconstruction and discriminating loss. Intrusion Detection Systems based on anomaly scores usually calculate a threshold representing an upper bound for tolerating anomalous behavior. The respective traffic sample is classified as malicious whenever the anomaly score surpasses the defined threshold. Its value can be defined in several ways, like using mean, standard deviation, and exhaustive search. Garcia et al. (2021) implement an anomaly score function based on an autoencoder's loss. The tolerance threshold is given by the mean plus standard deviation of normal instances' reconstruction errors. Lent et al. (2022) define the anomaly score using fuzzy logic. A Gaussian membership function is used to calculate the anomaly level of a sample. The decision threshold is defined using trial and error on attack validation data.

One of the main challenges in developing an IDS for SDN is not overloading the network's controller (Janabi, Kanakis, & Johnson, 2022). Using an elevated number of features for intrusion detection may increase resource usage and congestion on the southbound channel during data collection. Higher memory consumption and longer processing times can cause bottlenecks in the controller, rendering overhead, and crashes when applied to large-scale networks. Another frequently addressed challenge is the lack of purely SDN datasets and the fact that they may not accurately reflect the breadth of real-world security threats (Duy, Khoa, Hien, Hoang, & Pham, 2023).

Deep Learning models are the current leading solution for building intrusion detection systems due to their ability to process massive amounts of traffic, which is essential for the ever-increasing speeds of modern SDN networks. Their capacity to automatically capture intricate structures within the data is used to solve many computational problems within intrusion detection, including pattern extraction (Hnamte & Hussain, 2023), classification (Sivanesan & Archana,

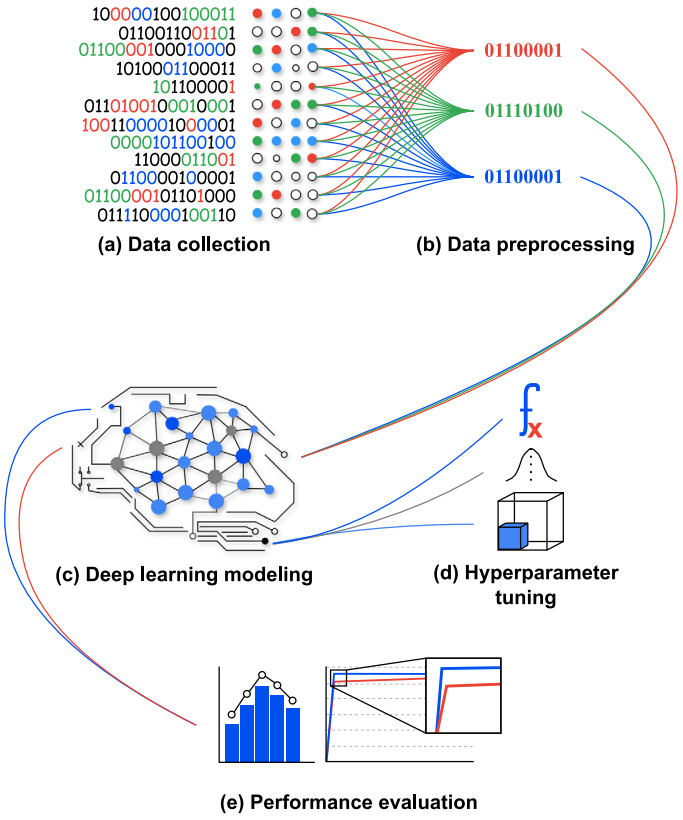


Fig. 3. Deep learning-based NIDS development common framework.

2023), regression (Yeom, Choi, & Kim, 2022), synthetic data generation (H., Rao, & Prasad, 2023), and data distribution learning (Duan, Fu, & Wang, 2023). Selecting and optimizing features tend to mitigate the risk of controller overload without relinquishing the detection ability of the model. Fig. 3 shows the steps generally conducted during DL-based NIDS development, including data collection and preprocessing, deep learning modeling, hyperparameter tuning, and performance evaluation.

4. Methodology

The objective of this research is to empirically review the state of the art of deep learning-based anomaly and intrusion detection on Software-Defined Networks. To meet this goal, we must answer important research questions regarding the common steps applied during NIDS development:

- **RQ1** - Which datasets and attacks are considered in the benchmarking experiments?
- **RQ2** - What preprocessing steps are usually taken before the anomaly detection phase?
- **RQ3** - How are deep learning models applied to detect anomalies and intrusions?
- **RQ4** - How are the models' hyperparameters optimized?
- **RQ5** - Which performance metrics are commonly calculated to evaluate and compare the models?
- **RQ6** - What are the open issues and possible future directions within the study area?

Fig. 4 illustrates the step-by-step sequence of the research methodology to conduct the proposed empirical literature review and answer the previous questions. The first step toward reviewing an area of study is the selection of adequate research article sources (Database Selection). There are plenty of online scientific databases indexing hundreds of thousands of articles. We defined some minimum criteria for a database to be eligible for our work: online accessibility through our institutions' agreements, including high-impacting journals that are aligned with our work scope, having the predominance of papers in the English language, supporting the search by boolean logic and the application of filters, and allowing the exportation of search results into .csv or .bib file formats.

We specify the search scope by defining a boolean query string with the most critical keywords (Querying and Filtering): "Deep Learning" AND (Intrusion OR Anomaly) AND Detection AND ("Software-Defined Network" OR "Software-Defined Networking" OR SDN). All selected databases are queried with the string. Besides we also apply some filtering to refine the search results: only full-length papers published in journals between 2021 and 2024 are considered since we are interested in the state-of-the-art. Although we have the consciousness that several relevant papers were published before 2021, this is an ever-changing research area, and we focus on providing a complete guide of its advances and challenges in recent years. We also filter out any article representing a survey or a review. The search results metadata are downloaded in .csv file format and organized using software for data tabulation.

The retrieved articles pass through a manual selection stage, which aims to analyze if they fit the work scope (Article Selection). We check if the papers propose an SDN intrusion detection solution utilizing a deep learning model. The checking is an essential step toward further result refinement, as the returned articles do not necessarily match our search goals. To illustrate, despite filtering out surveys and conference papers and excluding articles that do not consider deep learning, we still get those works returned from our queries to the databases. We also selected some pertinent works for other network environments despite focusing on Software-Defined Networks. We include them since they presented innovative approaches that could be straightforwardly adapted to the SDN paradigm.

After the data is collected and cleansed, we must thoroughly read all selected articles to answer the six research questions introduced previously (Article Reading). Each question is tackled and answered separately. Finally, we use the insights extracted from the reviewed works to write about the current state of the art (Result Writing). The novelties in the concerned area are presented through distinct sections dedicated exclusively to answering each proposed research question. We chose this organization scheme so that one could skip through the text and read only the desired insights. The remainder of this work showcases the results of the conducted literature review.

5. Results

In this section, we review the state of the art on Software-Defined Network anomaly and intrusion detection using deep learning. The following sections present the latest trends within the study area regarding benchmarking datasets, preprocessing methods, deep learning models, hyperparameter optimization, and performance metrics.

5.1. Metadata

Firstly, before introducing the findings, we display the metadata relating to the collected articles using our research methodology. We utilized our database selection requirements to pick the most relevant data sources for the proposed work: IEEE Xplore, Science Direct, ACM Digital Library, and Springer Link. The databases were accessed and queried on September 20th, 2023. The search query and the applied filters retrieved 557, 414, 155, and 350 articles from the data sources, totaling 1476 works.

After the article selection step, we kept only the works that fit our research scope. The number of selected articles was 105, corresponding to approximately 7% of the original retrieved results. From the selected

Fig. 4. Research methodology steps.

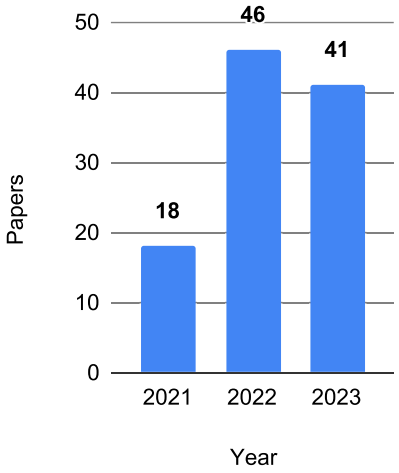


Fig. 5. Selected articles distribution per year.

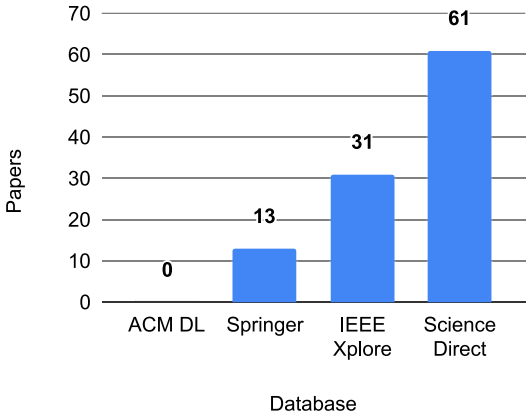


Fig. 6. Selected articles distribution per database.

works, 35 (33%) represent approaches not initially meant for the SDN environment. These solutions easily adapt to the Software-Defined Network paradigm since they are based on IP flow data. Thus, we considered them in this work.

Fig. 5 shows the distribution of the selected articles per year. There has been a growing trend in published papers in the research field since 2021, reinforcing that deep learning-based intrusion detection for SDNs is still an evolving area of research. The true amount of work available in 2023 surpasses the previous year, as our literature review did not consider papers from October to December 2023. We also illustrate the number of selected papers per database in Fig. 6. Almost 60% of the papers come from Science Direct. In contrast, no paper indexed by ACM DL has been selected, indicating that the former is a valuable data source for the field. Fig. 7 shows the distribution of the selected articles per publication journal. This graph lets us see which journals publish the most in the area, helping researchers find relevant periodicals for their publications. Remarkably, almost 30% of the papers come from either IEEE Access or Computer & Security. Note that we omitted the journals with two or fewer published papers for ease of visualization.

5.2. Datasets and attacks

In this section, we answer the first research question (RQ1) by providing an extensive overview of the datasets and attacks utilized in the reviewed works. Table 2 presents the usage of public datasets for developing state-of-the-art NIDS solutions. Most papers apply more than one dataset in their experiments, so the total number of works in the table exceeds the amount of reviewed papers. The five most used datasets are, respectively, CIC-IDS2017 (Sharafaldin, Lashkari, Ghorbani, et al., 2018), NSL-KDD (Tavallaee, Bagheri, Lu, & Ghorbani, 2009), CIC-DDoS2019 (Sharafaldin, Lashkari, Hakak, & Ghorbani, 2019), CSE-CIC-IDS2018 (Sharafaldin et al., 2018), and KDD Cup 1999 (Hettich & Bay, 1999). There are 36 unique datasets, but their utilization distribution is highly skewed since most were utilized only twice or once in the analyzed literature studies. The top ten datasets represent solely 28% of all identified data sources, but they account for 76% of the total data usage.

Table 2
Public datasets usage.

#	Name	Works
1	CIC-IDS2017	24
2	NSL-KDD	22
3	CIC-DDoS2019	21
4	CSE-CIC-IDS2018	14
5	KDD Cup 1999	11
6	InSDN	9
7	UNSW-NB15	9
8	ISCXIDS2012	5
9	Bot-IoT	5
10	MIT LL DARPA	4
11	CIC-DoS2017	3
12	N-BaIoT	3
13	Hogzilla	3
14	MAWI	3
15	UTSA-2021	2
16	ToN-IoT	2
17	CTU-13	2
18	WUSTL-IIOT-2018	2
19	CIDDS-001	2
20	WorldCup 1998	2
21	ISCX2016	1
22	KDD 2019	1
23	Orion	1
24	USTC-TFC16	1
25	Mendeley Data	1
26	MQTTset	1
27	CSIC 2010	1
28	CESA	1
29	Edge-IIoTset	1
30	CAIDA DDoS 2007	1
31	WSN-DS	1
32	SDN-IoT	1
33	IRAD	1
34	SPEAR Project	1
35	Kyoto 2006+	1
36	NITIDS	1

Fig. 8 presents a taxonomy of the found public datasets. Firstly, we classify them by the type of networking environment they represent,

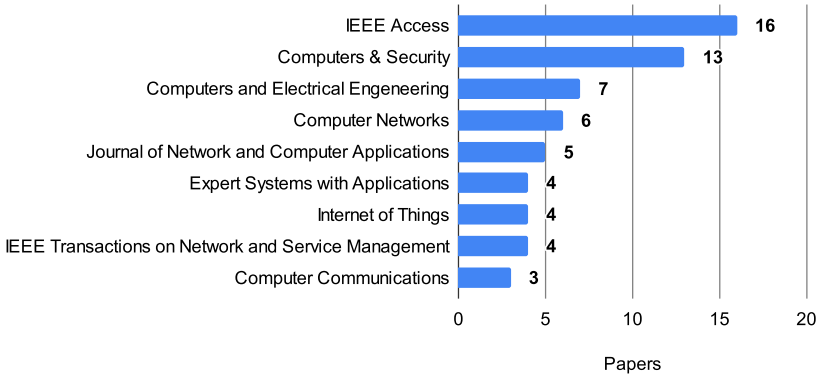


Fig. 7. Selected articles distribution per journal.

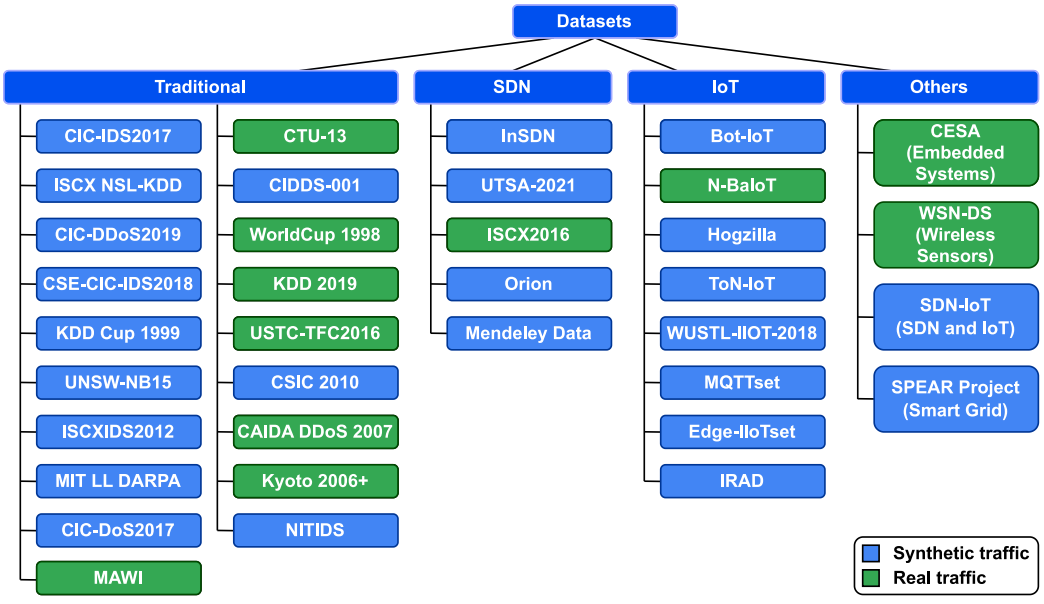


Fig. 8. Public datasets taxonomy.

including traditional, SDN, and IoT. The green-colored datasets are composed of traffic data from real networks. The blue-colored ones comprise synthetic traffic data generated in a controlled environment. Only 30% of the found datasets contain real data, while the remaining 70% simulate or emulate networking systems to collect artificial data. We emphasize that the ten most used datasets listed in Table 2 are all synthetic. Also, popular datasets among scientists, namely NSL-KDD and KDD Cup 1999, are not representative of modern networks since they contain outdated traffic patterns and attacks. Traffic data captured on real, contemporary networks is essential for a realistic assessment of NIDS performance. Thus, future research should focus on collecting and utilizing real-world, up-to-date datasets to develop more reliable intrusion detection systems.

Fig. 9 illustrates a word cloud of the network attacks found in the analyzed public datasets. The bigger and bolder an attack is, the greater its frequency on the data sources. The DoS and DDoS attacks stand out, highlighting the scientific community's effort in studying network availability threats. Botnet attacks are also a problem that captures scientists' attention. The popularization of the IoT paradigm is likely facilitating the construction of botnets to launch distributed attacks. Port scanning may disclose network vulnerabilities that can be explored by other types of attacks, explaining its frequent appearance in benchmark datasets. A multitude of network services are available through websites. Malicious agents are constantly updating web attacks to exploit such systems, so it is coherent to investigate these kinds of traffic anomalies. Computers have grown exponentially in processing and storing capacity in the last decades. The accessibility to powerful computing is probably enabling the execution of brute force attacks with a greater success rate, thus requiring the scientists' cautious examination.

We recognized many tools some works utilize to generate synthetic datasets for their experiments. Table 3 lists the eleven different tools found in the literature. We also classify them according to how they support dataset creation, as illustrated in Fig. 10. The most popular tool is Mininet (Bob Lantz, 2024), a network emulator applied in ten distinct works. An emulator is a tool that aims to replicate the inner workings of a real system, in this case, a network. MaxiNet (Wette et al., 2014) is an alternative emulator utilized by only one study. hping3 (Oliveira, 2024) is used by four works to generate traffic since it allows for the sending of custom packets. Network simulators, as GNS3 (SolarWinds Worldwide, 2024) and OMNeT++ (Ltd., 2024b) are utilized by two works each. A simulation tool tries to reproduce the general behavior of a real system, in this case, a network. There are also tools for implementing virtual SDN controllers, for example, Ryu (Community, 2024), POX (McCauley, 2024), and ONOS (Foundation, 2024). We also found tools for traffic replaying and analysis in the reviewed works: Tcpreplay (Fred Klassen, 2024) and DNS-STATS (Ltd., 2024a).

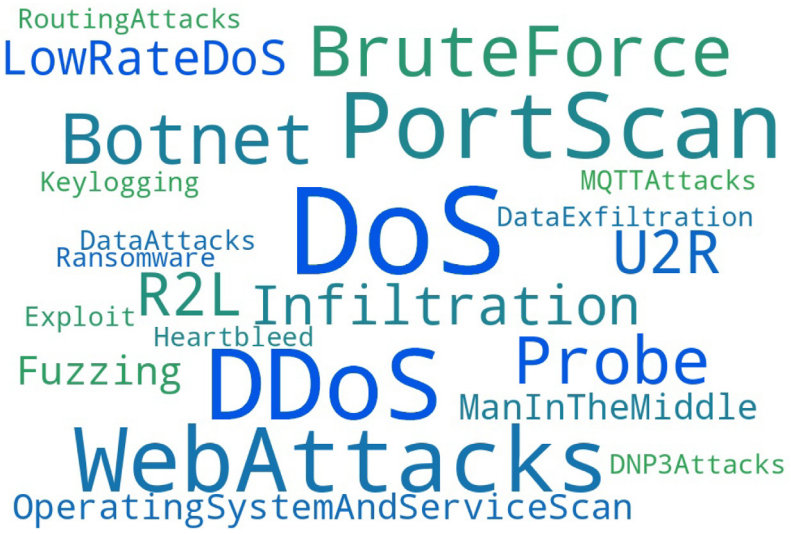


Fig. 9. Attacks word cloud.

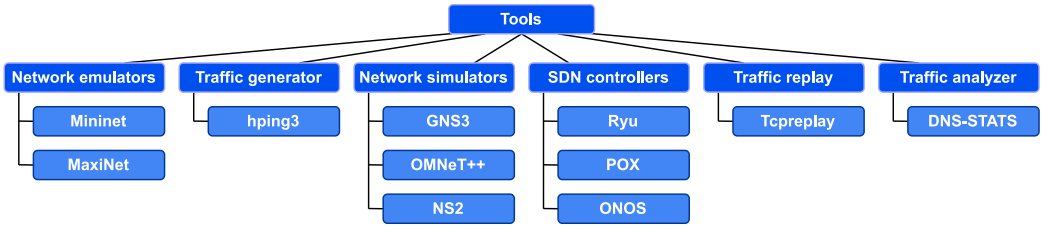


Fig. 10. Taxonomy of dataset generation supporting tools.

Table 3
Usage of dataset generation supporting tools.

#	Tools	Works
1	Mininet	10
2	hping3	4
3	GNS3	2
4	Ryu	2
5	Tcpreplay	2
6	OMNeT++	2
7	POX	1
8	ONOS	1
9	MaxiNet	1
10	NS2	1
11	DNS-STATS	1

Table 4
Preprocessing methods summary.

Subtopic	Description	Works
Data Normalization	Scale data to fit into certain interval.	49
Data Cleaning	Remove invalid data (e.g., NaN and infinite values).	28
Remove Duplicate Data	Remove data that may bias the model during training.	9
Data Sampling	Balancing a dataset by creating or removing entries.	5
Data Encoding	Conversion of categorical data into numerical data.	37
Feature Extraction	The processing of data to create new features.	20
Feature Selection	Statistical analysis to reduce the number of features used by the model.	37

5.3. Preprocessing methods

In this section, we answer the second research question (RQ2) by compiling the methods used by the analyzed articles to prepare data before feeding it into their models. All procedures made before the main model receives the data are considered preprocessing methods. This excludes approaches that, for instance, apply the anomaly detection method as the feature selection itself. Table 4 summarizes each method for a quick overview.

5.3.1. Data normalization

Data normalization is the process of transforming values from each feature to match a certain scale. This is especially important to neural networks due to how neurons work (Lent et al., 2022). Each feature's value is multiplied by its weight and added to the other features' results. Any feature that has values extremely high or excessively low may overtake the neuron's "attention" for themselves for several epochs until their weight value adapts to it. This hinders the model's convergence and delays training.

Min–max scaling and z-score were the most mentioned methods among the collected articles. The former formula scales the values to a given range, usually 0 and 1 (Duy et al., 2023, 2021). This maintains the values' proportion, so newer data may still be mapped to numbers outside this interval if they are greater or smaller than the initial ones (Sayed, Le-Khac, Azer, & Jurcut, 2022). For this reason, min–max normalization is vulnerable to outliers since the result would have normal data grouped in a smaller interval while leaving outliers in the edges.

Part of these problems can be mitigated with the z-score, a standardization technique (ElSayed, Le-Khac, Albahar, & Jurcut, 2021). In this method, the resulting value represents how many standard deviations from the set's mean the value is Sayed et al. (2022). This way, outliers

become immediately clear and can be treated as such without affecting the mapping significantly.

5.3.2. Data cleaning

Data cleaning is a necessary step in data pre-processing in certain datasets to avoid the malfunction of deep learning models. Invalid data, e.g., NaN and infinite values, are replaced or have their whole entries removed (Duy et al., 2023).

Among the analyzed articles, the data cleaning process was mentioned for a variety of datasets, namely CICIDS2017 (Yungaicela-Naula, Vargas-Rosales, & Perez-Diaz, 2021), CICIDS2018 (Duy et al., 2023), InSDN (Hnamte & Hussain, 2023), and CICDDoS2019 (O. Lopes, Zou, Abdulqadder, Akbar, Li, Ruambo, & Pereira, 2023). The process was also mentioned when the works used datasets generated for their own study to clean collecting errors, as discussed by Nadeem et al. (2023).

5.3.3. Remove duplicate data

Despite being not as crucial when compared with the previous procedures, the removal of duplicate data can be impactful on the performance of a deep learning model. If there are multiple copies of an entry during training, it is possible that a bias is created toward that type of data (O. Lopes et al., 2023). Note that this would only impact a model if there is a lack of entries or the redundancy is considerably high.

Another and more problematic issue caused by duplicate data happens during the split between training and testing, when the redundant data may end in separate groups. This way, the model will be tested with data seen during training, causing an inflation on the resulting metrics.

5.3.4. Data sampling

Data sampling plays an important role in deep learning training since a considerable amount of data is necessary for this type of model to become capable of generalizing a class or a problem. For this reason, developers may upsample minority classes or downsample majority ones (O. Lopes et al., 2023). Among the analyzed works, it was common not to have the sampling process explained, except for Friha et al. (2023) that specified SMOTE.

5.3.5. Data encoding

Dada encoding is fundamental for some features in deep learning models. Categorical or qualitative data are incompatible with most deep learning models since they require numerical input. In addition, the numerical values need to have a meaning of quantity to impact the model's training (Lent et al., 2022). Many analyzed articles mentioned the encoding of features as IP addresses, protocols, and labels in their system's preprocessing stage.

Another similar process employed is called one-hot encoding, where instead of simply transforming a categorical value into a number, a binary vector is created. This vector comprises zeros except for a single element with the value 1 representing the former category. This is frequently applied to labels, but it can also be used on features (Shaji et al., 2023). Some works handle categorical data by extracting entropies from the features. These were considered a form of feature extraction and are mentioned in the next section.

5.3.6. Feature extraction

Due to the nature of the problem, anomaly detection systems require traffic data to discover anomalies. The SDN paradigm makes collecting this data significantly simple since the controller can request flow data from the whole network and group it in a single place. Some analyzed works use features from the dumped flows directly in their deep learning models. It is also possible to extract features from this data to either improve training by reducing the number of features or to add information that would not be available in its current form.

There are systems that do not use flows or packets directly to detect anomalies. Instead, they may digest traffic by grouping those elements, for example, by seconds (Janabi et al., 2022) or flow tables (Fouladi et al., 2022). The extracted features vary by system, either due to how the system works or due to some feature selection algorithm.

We identified that entropy is a frequently extracted feature. It was used to represent IP addresses and ports on the work presented by Novaes et al. (2021) and other features on Fouladi et al. (2022)'s study. Entropy is a measure of the randomness of a variable. The entropy measure changes based on the probability of each possible value that the variable can assume. If there are multiple values with none standing out, the entropy may increase. Otherwise, it may decrease if an element is significantly more common than the others.

It is important to highlight that deep learning models can be responsible for extracting abstract features. Choobdar et al. (2021) use an autoencoder to extract features from data before sending it to the detection module. Some works may use the main deep learning module to select and extract features automatically, but they were not considered preprocessing and will be mentioned in Section 5.4.

5.3.7. Feature selection

Some network anomaly detection datasets can have more than eighty features that describe the network's traffic. Several of those features do not contribute to anomaly detection and tend not to be used by the deep learning models. Those features may be discarded to save memory and even lower training times. Less data means less processing, avoiding extra training epochs for the network to lower the relevance of those features.

Some algorithms can be used to find a correlation between a set of features and the dataset labels, identifying the most relevant ones. Some examples of algorithms are the Principal component analysis (PCA) used by Cherian and Varma (2023); XGBoost, used by Zainudin, Ahakonye, Akter, Kim, and Lee (2023); Random forest, used by Sayed et al. (2022); and Autoencoders by Choobdar et al. (2021).

5.4. Deep learning-based intrusion detection

In this section, we answer the third research question (RQ3). Table 5 lists deep learning models and their usage in the reviewed works. Note that some papers implement multiple models, so the total number of uses surpasses the amount of selected works. CNN and LSTM are the two most applied models, both with over 30 uses. Following, the DNN and GRU networks were applied 15 and 11 times, respectively. There is a high concentration of usage for these four models. They reflect only 11% of all applied models but represent 61% of the total deep learning utilization. Despite this, we draw attention to the scientific community's efforts to test new deep learning networks and improve IDS performance since there are more than 30 different models under experimentation. Researchers are implementing multiple Autoencoder variations, such as VAE (Bårli, Yazidi, Viedma, & Haugerud, 2021), SSAE (Long & Jinsong, 2022), DAE[1] (Sood et al., 2023), and DAE[2] (Lopes et al., 2022). There is an interest in neural networks based on graph data structure, including GCN (Ding & Li, 2022), HGCN (Phu et al., 2023), and ST-GCN (Wang et al., 2023). Generative networks as GAN (Siniosoglou et al., 2021), WGAN (Mustapha et al., 2023), WGAN-GP (Duy et al., 2023), Bi-GAN (Shu et al., 2021), and Bi-CGAN (H. et al., 2023), are also being analyzed for IDS development.

Fig. 11 presents a taxonomy of how the listed deep learning models are usually applied during the development of an intrusion detection system. The reviewed works are classified using a tree-like diagram of four levels. Each tree level indicates a different system characteristic: the learning approach, the system architecture, the type of problem solved by the deep learning model, and the applied model itself. A complete path in the tree describes a deep learning application pattern in the literature. For instance, it is possible to visualize that the

Table 5
Deep learning models usage.

#	Acronym	Model	Works
1	CNN	Convolutional Neural Network	36
2	LSTM	Long Short-Term Memory	31
3	DNN	Deep Neural Network	15
4	GRU	Gated Recurrent Unit Networks	11
5	AE	Autoencoder	6
6	RNN[1]	Recurrent Neural Network	6
7	GAN	Generative Adversarial Network	4
8	LSTM-AE	Long Short-Term Memory Autoencoder	4
9	SAE	Stacked Autoencoder	4
10	Bi-LSTM	Bidirectional Long Short-Term Memory	3
11	GCN	Graph Convolutional Network	3
12	WGAN	Wasserstein Generative Adversarial Network	3
13	WGAN-GP	Wasserstein Generative Adversarial Network with Gradient Penalty	2
14	DAE[1]	Deep Autoencoder	2
15	Bi-GRU	Bidirectional Gated Recurrent Unit	2
16	AM	Attention Mechanism	2
17	Bi-CGAN	Bidirectional Cross Generative Adversarial Network	1
18	Bi-GAN	Bidirectional Generative Adversarial Network	1
19	DAE[2]	Denoising Autoencoder	1
20	DBN	Deep Belief Network	1
21	DDPG	Deep Deterministic Policy Gradient	1
22	DDQN	Double Deep Q-Network	1
23	DMN	Deep Maxout Network	1
24	SE	Stacking Ensemble	1
25	SSAE	Stacked Sparse Autoencoder	1
26	ST-GCN	Spatio-Temporal Graph Convolutional Network	1
27	TST	Time Series Transformer	1
28	VAE	Variational Autoencoder	1
29	RNN[2]	Replicator Neural Network	1
30	RNN[3]	Recursive Neural Network	1
31	PNN	Probabilistic Neural Network	1
32	HGCN	Hyper Graph Convolutional Network	1
33	GNN	Graph Neural Network	1
34	GC-LSTM	Graph Convolutional Long Short-Term Memory	1

autoencoder is applied across many scenarios: supervised and unsupervised approaches; centralized or decentralized architectures; and feature extraction and data reconstruction problems.

Fig. 12 shows how the selected works are distributed according to their learning approach, system architecture, and the type of solved problem. All three distributions are highly skewed, as the evaluated works tend to follow similar design decisions. Firstly, the leftmost graph (a) reveals that 96 out of 105 works are dependent on labeled data for developing their system. Implementing these solutions in a real environment may require a high investment in data labeling. Systems independent of data labels are proposed by 6 works. Notably, only 3 works use the reinforcement learning paradigm for their solutions. Secondly, regarding the architecture (b), we observe that approximately 85% of the systems are built to run on a central server, while the remaining 15% follow a distributed design. Thirdly, the majority of the selected papers use deep learning to solve feature extraction, classification, or both. There are 9 works using data reconstruction to build their IDS. Few solutions are implemented to solve regression, data generation, and Markov decision process.

The skewness in these distributions highlights many possible future research possibilities: unsupervised and reinforcement-based systems, distributed IDS architectures, deep learning modeling for solving regression, data generation and reconstruction, and Markov decision process. We encourage researchers to investigate these uncommon design choices to promote further development in the intrusion detection area. Some innovative works explore unusual models and approaches for implementing NIDS. Zavrak and Iskefiyeli (2023) train unsupervised Replicator Neural Network and LSTM-based encoder–decoder for reconstructing normal traffic data. Every sample whose reconstruction error exceeds a threshold is inferred anomalous. Shu et al. (2021) proposed an unsupervised NIDS installed on distributed SDN controllers to avoid computational overhead. The system applies Bi-GAN to calculate an anomaly score based on a linear combination of reconstruction error and discrimination loss. Friha et al. (2022) introduces a federated learning IDS which coordinates collaborating edge nodes to build a global traffic classification model. Shukla et al. (2023) describes a system based on Recursive Neural Network for solving the Markov Decision Process. Yeom et al. (2022) utilize LSTM to implement a regression model which predicts future normal traffic volume. The following sections dive deeper into the proposed organization taxonomy by discussing the selected works.

5.4.1. Supervised learning

Every IDS dependent on labeled data in any stage of development was classified as supervised. For instance, Janabi et al. (2022) and Imrana et al. (2021) works were considered supervised. The former utilized labeled data to build a CNN model that maps input records to benign or anomalous classes. The latter applied a Bi-LSTM to extract patterns from input data, which is then fed to a regular, fully connected neural network to produce the corresponding classification. Kaur and Kakkar (2022) created a supervised attack detection system that applies the Deep Maxout Network model to classify traffic as normal or anomalous.

Fig. 11 and Fig. 12 reveal a broad exploration of supervised approaches in the literature. These may be implemented centrally, where the data and the code are contained in a single server. Hairab et al. (2022) and El-Ghamry, Darwish, and Hassanien (2023) designed their IDS to be installed on a central computer responsible for receiving and processing all incoming traffic data. Both systems are based on a CNN model implementing binary classification. Novaes et al. (2021) discussed a centralized IDS based on the GAN model to detect DDoS attacks. Their system explores the discriminator network to perform binary classification on traffic samples. There are also supervised approaches based on a decentralized architecture. Illy and Kaddoum (2023) introduced an intrusion detection solution that may be deployed in local servers of different network segments. The IDS utilizes DNN, CNN, RNN, and LSTM for traffic classification. Illy, Kaddoum, de

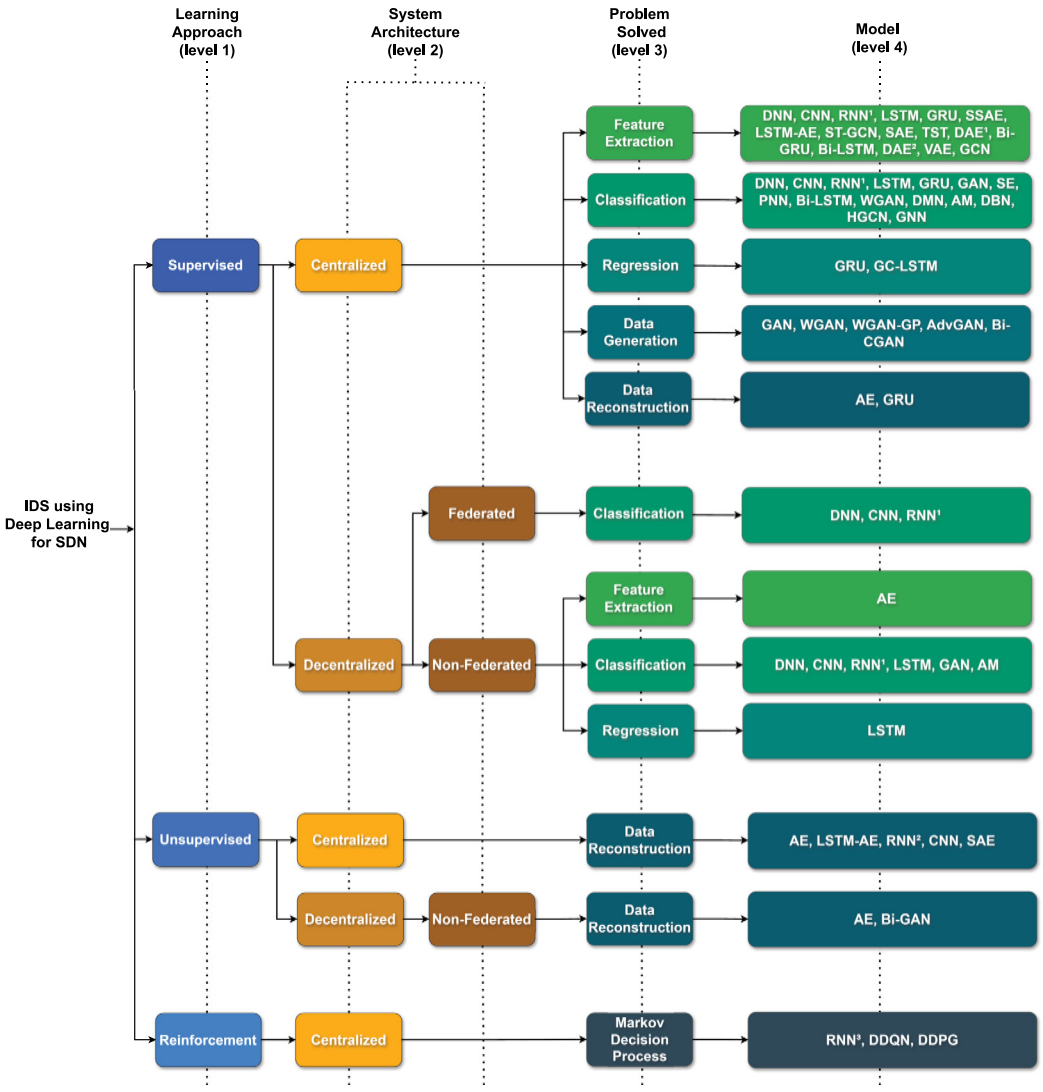


Fig. 11. Deep learning application taxonomy.

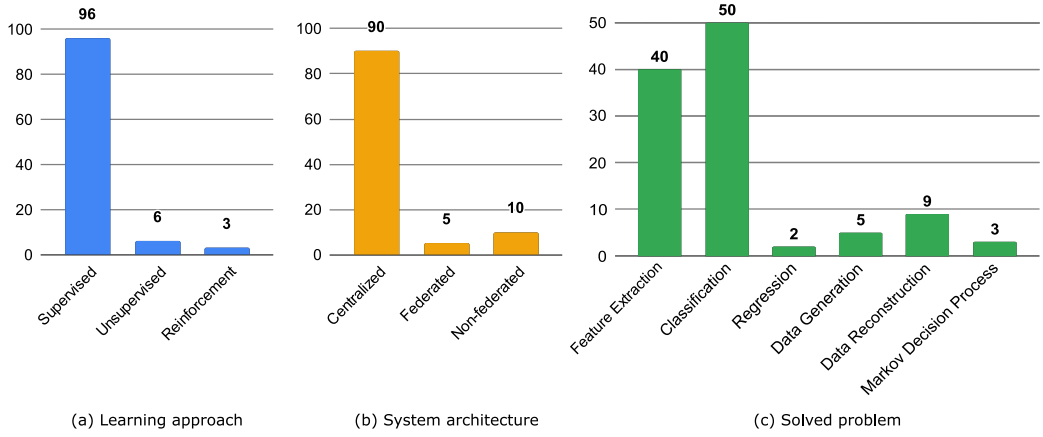


Fig. 12. Selected works distribution according to their learning approach (a), system architecture (b), and solved problem (c).

Araujo-Filho, Kaur, and Garg (2022) proposed a collaborative intrusion detection and prevention system that may be installed on multiple sub-networks to react to raising anomalies promptly. Their system employs DNN, RNN, and CNN for binary and multiclass classification. Ferrag, Friha, Hamouda, Maglaras, and Janicke (2022) describe a decentralized system that uses the federated learning paradigm to develop multiple DNN models based on different devices' local data. These models are trained to perform data classification on their available data. They are later shared and aggregated to build a global model for the whole network.

The analyzed supervised works implemented their detection systems by applying deep learning to solve specific mathematical problems. The most common type of solved problem is feature extraction. It involves reducing input dimensionality by calculating more expressive data characteristics. Reducing the number of features (input dimensions) allows for more straightforward pattern finding in complex data. Data classification is another issue commonly solved when building intrusion detection mechanisms. It represents the challenge of deriving a function to map an input to a correct discrete output. Feature extraction and classification are usually solved in conjunction. To illustrate, Cherian and Varma (2023), Soltani, Siavoshani, and Jahangir (2021), Tayfour et al. (2023) developed intrusion detection systems that use LSTM to extract temporal features from traffic data. The reduced input is processed by a fully connected neural network, which solves classification and outputs the traffic state (benign or malign). Kumar et al. (2023) built a Bi-GRU network to calculate temporal features from incoming input. The extracted patterns pass through dense layers, producing the corresponding sample classification.

The regression problem is similar to classification, as it corresponds to the derivation of a function to map the input to a corresponding continuous output. Lent et al. (2022) utilized GRU to solve regression and learn a non-linear function that inputs past network traffic and outputs the expected traffic for the moment of analysis. The system issues an anomaly alarm if the observed traffic in the network patterns does not match the GRU prediction.

The scientific community also addresses the topic of data generation. Duy et al. (2021) and Duy et al. (2023) apply variations of the Generative Adversarial Network model, e.g., WGAN, WGAN-GP, and AdvGAN, to synthesize adversarial attacks and prepare their IDS to be robust against them. Ding et al. (2022), H. et al. (2023) develop GAN and Bi-CGAN models to oversample known attack classes with few data instances, aiming to improve the performance of classification models.

Data reconstruction concerns the problem of accurately compressing and decompressing data. The most common deep learning model used to solve reconstruction is the autoencoder. For data that fit the model training set, the mean squared error between original and reconstructed (decompressed) samples should be small. Meanwhile, the error may be significant for outliers, as the model cannot precisely reconstruct them. Sarıkaya, Kılıç, and Demirci (2023) built an IDS that reconstructs legit and anomalous traffic data using an autoencoder. Afterward, an ensemble of machine learning models classify the input based on the magnitude of its reconstruction error.

5.4.2. Unsupervised learning

Intrusion detection systems that do not rely on labeled data for their configuration were considered unsupervised. The unsupervised paradigm is not as widely adopted as the supervised one. Fig. 11 shows that these systems solely solve the data reconstruction problem, independent of having a centralized or decentralized architecture. We should point out that no analyzed work approached unsupervised federated learning.

Only a few deep learning models have been explored for implementing unsupervised systems. AE and its variations, like Stacked AE and LSTM-based AE, are common for solving data reconstruction and building unsupervised IDS. For example, Fouladi et al. (2022) and Garcia et al. (2021) described detection systems using autoencoder to reconstruct benign traffic samples. If the error of a sample reconstruction surpasses a predefined threshold value, the input is inferred anomalous. Duan et al. (2023) developed a centralized intrusion detection system based on a Stacked AE. The model learns to reconstruct benign samples with low error while attributing high error to outliers. Fu, Duan, Wang, and Li (2022) devised an LSTM-based autoencoder for compressing and decompressing data sequences. Their approach relies on a threshold for separating benign and anomalous input. Nonetheless, other models including Bi-GAN (Shu et al., 2021) and RNN[3] (Zavrak & Iskefiyeli, 2023) have also been experimented.

Table 6
Hyperparameter tuning summary.

Method Class	Works
Grid Search	Ravi, Chaganti, and Alazab (2022), Tsogbaatar et al. (2021), Liu et al. (2022), Ferrag et al. (2022).
Population-based Metaheuristics	El-Ghamry et al. (2023), Mansour (2022), Sivanesan and Archana (2023), Ahmad et al. (2023), Vatambeti et al. (2023).
Bayesian Optimization	Presekal, Ştefanov, Rajkumar, and Palensky (2023).
Genetic Algorithm	Song et al. (2023).
Taguchi Method	Dinh and Park (2021).

5.4.3. Reinforcement learning

The reinforcement learning paradigm was rarely applied in the selected works for this survey. The works that relied on this learning approach were all implemented centrally to solve the Markov Decision Process (MDP) problem. Phan and Bauschert (2022) proposed an ML-based intrusion detection system associated with an adaptive intrusion response system based on a Double Deep Q-Network (DDQN). This system was developed using reinforcement learning to obtain the optimal intrusion response policy for malicious activity. The optimization problem was modeled as an MDP, where DDQN is intended to address the slow convergence of Q-learning due to large state space.

Kim et al. (2022) developed a framework that dynamically allocates traffic inspection resources. It employs a deep deterministic policy gradient algorithm to learn an optimal resource allocation policy, adapting to changing network conditions and malicious flow occurrences by modeling the problem as an MDP. It includes a DRL-based network traffic inspection mechanism and an address shuffling-based moving target defense technique for proactive intrusion prevention.

Shukla et al. (2023) implemented a Recursive Neural Network strategy to monitor traffic flow and improve detection performance. This framework aims to teach users how to match traffic flows effectively. Enhances transparency and proactively protects the SDN data plane from overload. Applying a learned traffic flow matching control policy optimizes traffic data acquisition for real-time abnormality detection.

Reinforcement learning's ability to learn and adapt over time makes it a promising direction for future research in enhancing network security and resilience against evolving cyber threats.

5.5. Hyperparameter optimization

In this section, we answer the fourth research question (RQ4) regarding hyperparameter tuning. This represents an optimization problem that aims to calculate a deep learning model's hyperparameters to improve its results in a certain task. It is present in the development of every deep learning model since there are unlimited combinations of parameters to be chosen for a neural network. Most of the study works omit this process since it often does not contribute to the description of the presented process nor to its final result. The lack of innovation in the tuning process contributes to its disregard. Table 6 summarizes the optimization methods and respective works mentioned in this section.

Most of the works that mention the tuning process describe it briefly in two main ways: Random and empirical search, and grid search (Ferrag et al., 2022; Liu et al., 2022; Ravi et al., 2022; Tsogbaatar et al., 2021). The former consists of testing different values and altering them based on performance, while the latter establishes a range for each hyperparameter and tests each combination. Those methods, just like most of these optimizations, will result in good enough combinations but not the optimal ones. It is important to understand how it is unrealistic to find the optimal values since there are too many parameters to optimize that influence each other. In addition, the random nature of

neural network training generates different results on each trial, which can favor or disfavor certain hyperparameter combinations, even for small amounts.

Some works present some optimization techniques based on metaheuristics, which are a way of avoiding exhaustive combination trials. El-Ghamry et al. (2023) employed particle swarm optimization to optimize some CNN parameters: learning rate, dropout rate, and others specific to the proposed model. Other uncommon metaheuristics are the chicken swarm optimization used by Mansour (2022), the firefly optimization applied by Sivanesan and Archana (2023), the slime mould optimization used by Ahmad et al. (2023), and the prairie dog optimization utilized by Vatambeti et al. (2023). All mentioned works apply population-based techniques that simulate certain group behaviors in order to maximize a metric.

A different class of optimization is used by Presekal et al. (2023). The authors apply Bayesian optimization. This method is intended to optimize functions that are costly to evaluate. This fits the deep learning hyperparameter optimization problem since it is necessary for a neural network to be trained to evaluate each combination, thus requiring significant computational processing.

A genetic heuristic was employed by Song et al. (2023). It is called gene expression programming, a combination of genetic programming with genetic algorithms to make a global search optimization. This algorithm works with chromosome encoding to build the hyperparameter combination for a CNN network.

Most works do not present the tuning process but may mention the final hyperparameter combination to improve replicability. The parameters frequently mentioned include learning rate, number of epochs, batch size, number of neurons, activation functions, and the optimizer algorithm. Those are core parameters of a deep learning model, which explains their presence.

5.6. Performance metrics

In this section, we answer the fifth research question (RQ5). The study explores the metrics commonly employed in the literature to gauge the effectiveness of proposed deep learning models for anomaly and intrusion detection in Software-Defined Networks. Understanding these metrics is essential for assessing the performance of such models accurately. By examining the metrics utilized in existing studies, we aim to gain insights into the evaluation criteria adopted by researchers to measure the efficiency of their proposed detection systems.

An extensive analysis of the 105 selected papers showed that the 10 most used performance metrics are Recall, Accuracy, F1-score, Precision, False Positive Rate (FPR), ROC curve, True Negative Rate, Confusion matrix, Area under the ROC curve and training time, as can be seen in Fig. 13.

As most of these metrics are based on primitive metrics defined by the confusion matrix, we will address this topic first. A confusion matrix provides a clear breakdown of the IDS's correct and incorrect classifications. It comprises four key variables, as illustrated in Fig. 14:

True Positive (TP): The IDS correctly identifies abnormal network traffic as anomalous.
True Negative (TN): The IDS correctly identifies legitimate network traffic as normal.
False Positive (FP): The IDS incorrectly identifies normal traffic as anomalous.
False Negative (FN): The IDS incorrectly identifies anomalous traffic as normal.

For each classification made by the IDS, the corresponding variable is incremented accordingly. The matrix's rows represent all samples classified as anomalous or normal, while the columns represent the actual anomalous and normal samples, respectively. Correct classifications, including both true positives and true negatives, are represented along the main diagonal, while incorrect classifications, encompassing false positives and false negatives, are situated along the secondary diagonal. This structured representation provides a comprehensive overview of the IDS's performance in distinguishing between normal and anomalous network traffic.

With the components of the confusion matrix defined, we can address the definitions of the remaining metrics outlined above. First, we will discuss the more commonly used metrics, which appeared in more than 70% of the evaluated papers.

The Accuracy (Eq. (1)) quantifies the proportion of all samples correctly classified by the IDS. It addresses the fundamental question: What percentage of all samples were accurately classified? Accuracy may not be suitable for scenarios with significant class imbalances, where the model might favor the most prevalent classes, thus failing to detect rarer ones. Consequently, a poorly performing model, for instance, one predicting all samples as the most common class, could achieve high accuracy. Despite its drawbacks, it still remains one of the most used metrics in this area, being applied in 86% of the evaluated papers, as shown in Fig. 13.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

Recall (Eq. (2)), also known as Sensitivity, True Positive Rate (TPR), and Detection Rate (DR), quantifies the proportion of actual anomalous samples correctly identified as anomalies by the IDS. It addresses the question: What percentage of all anomalous samples in the network traffic can be detected by the IDS? This metric provides insights into model errors since higher recall indicates fewer false negatives. It was the most frequently used metric on the evaluated papers, being applied in 87.6% of them.

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

Precision (Eq. (3)), also known as Positive Predictive Value (PPV) and Attack Predictive Value (APV), measures the proportion of anomaly predictions that are correct. It answers the question: What percentage of the samples identified as anomalies by the IDS are truly anomalous? This metric indicates the level of false positives: higher precision implies fewer false positives.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

The F1-score (Eq. (4)), also referred to as F-measure or F-score, represents the harmonic mean between Precision and Recall. It provides a balanced measure when both Precision and Recall are either high or low. Thus, if one of them is high while the other is low, the F1-score will be moderate. Unlike accuracy, the F1-score is a preferable metric for scenarios with data imbalance.

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4}$$

Now, we will discuss some less commonly used metrics, which appeared in 10% to 30% of the evaluated papers. Although less common, they are well-known in the literature.

The False Positive Rate (FPR) (Eq. (5)), also known as False Alarm Rate (FAR) and Fall-out, calculates the percentage of normal samples incorrectly classified as anomalies. It answers the question: What fraction of all normal samples were erroneously labeled as anomalies?

$$FPR = \frac{FP}{FP + TN} \tag{5}$$

The Receiver Operating Characteristic (ROC) curve offers a graphical representation of an IDS's balance between True Positive Rate (TPR), and False Positive Rate (FPR), for various thresholds. Fig. 15 illustrates an example ROC curve, with the x-axis denoting FPR values and the y-axis representing TPR values. Each point on this curve corresponds to the TPR and FPR for a specific threshold. A favorable threshold choice yields high TPR coupled with low FPR, indicating minimal false positives and false negatives. The ROC curve empowers modelers to select the threshold that optimally balances false negatives and false positives for the IDS's particular use case.

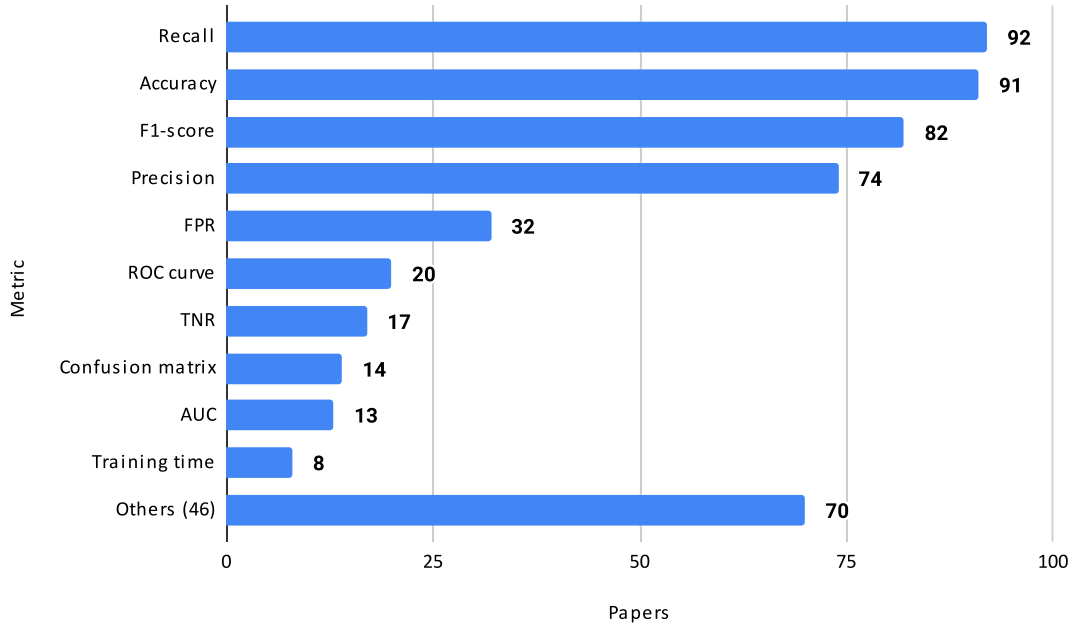


Fig. 13. Most used performance metrics.

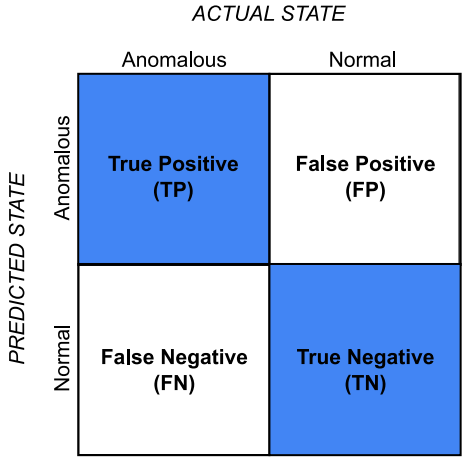


Fig. 14. Confusion matrix.

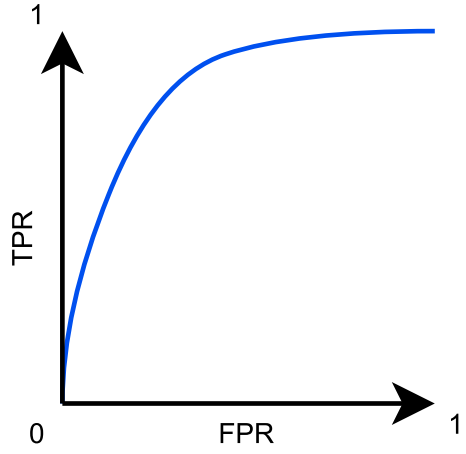


Fig. 15. ROC curve example.

AUC (Area Under the Curve) serves as a metric summarizing the information provided by the ROC curve. It gauges the model's proficiency in distinguishing between positive and negative classes by measuring the area beneath the ROC curve. Consequently, a higher AUC implies superior IDS performance, irrespective of the chosen threshold.

True Negative Rate (TNR) (Eq. (6)), also known as Specificity and Normal Predictive Value (NPV), quantifies the proportion of normal samples correctly classified by the IDS as normal. It addresses the question: What percentage of all normal samples in network traffic could be correctly identified by the IDS? TNR complements the False Positive Rate (FPR) and, therefore, achieving high TNR and low FPR is desirable for IDS to reduce the amount of false alarms generated.

$$TNR = \frac{TN}{TN + FP} \tag{6}$$

Some authors also employed the training time, which was the most frequent computing performance-related metric. Some authors also measured the testing time, the inference time, and the convergence time. Another approach was to evaluate detection and mitigation times and the computational complexity of the proposed method.

The remaining observed metrics rarely were applied to the studies, representing more specific approaches, such as G mean, Bookmaker Informedness (BM), and mitigation rate. From the remaining metrics, we will discuss four of them, since they are more common in general classification research problems.

The first is the Matthews Correlation Coefficient (MCC) (Eq. (7)), which is considered a robust metric as it takes into account all four cells of the confusion matrix in its computation. Therefore, it only yields a favorable result if the model performs well across all four matrix cells: high TP and TN, and low FP and FN. This property makes MCC a preferred metric for scenarios with imbalanced class distributions. Its results range between −1 and 1, with −1 representing the poorest model performance and 1 indicating the best. An MCC of zero represents that the model's prediction is no better than a random one.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{7}$$

Secondly, the False Negative Rate (FNR) (Eq. (8)) quantifies the proportion of actual anomalous samples that were incorrectly classified as normal by the IDS. It complements the recall by addressing the

question: What percentage of all anomalous samples in network traffic could not be detected by the IDS?

$$FNR = 1 - Recall = \frac{FN}{FN + TP} \tag{8}$$

The third one is the False Discovery Rate (FDR) (Eq. (9)), which is the complement of Precision, measuring the proportion of incorrect anomaly predictions. FDR answers the question: What percentage of the samples flagged as anomalies by the IDS were, in fact, not anomalous? Thus, a desirable performance goal for FNR is to achieve a low value, indicating that the IDS accurately identifies the majority of anomalous samples present in the network traffic.

$$FDR = 1 - Precision = \frac{FP}{FP + TP} \tag{9}$$

Finally, the fourth metric is the False Omission Rate (FOR) (Eq. (10)), quantifying the proportion of actual anomalous samples that were incorrectly classified as normal by the IDS. It addresses the question: What percentage of the samples flagged as normal by the IDS were, in fact, anomalous?

$$FOR = 1 - TNR = \frac{FN}{FN + TN} \tag{10}$$

6. Open issues and future directions

Recall, Accuracy, F1-score, and Precision were the most used performance metrics from the 105 analyzed papers, appearing in more than 70% of the selected studies. These were employed majorly in measuring the performance of CNN, LSTM, DNN, and GRU models. These four Deep Learning approaches reflect only 11% of all applied models but represent 61% of the total deep learning utilization. The five most used datasets are synthetic and represent traditional networks, including CIC-IDS2017, NSL-KDD, CIC-DDoS2019, CSE-CIC-IDS2018, and KDD Cup 1999. Data preprocessing and hyperparameter tuning tend to limit the versatility of a model, often making it highly specialized for a specific application. The lack of SDN datasets with real data, a significant number of days, and diverse and updated attacks are limitations found in this research. The best choice of model, data, and metrics is fundamentally conditioned on the target application, and this survey aims to highlight the trends in the field.

In this section, we answer the final research question (RQ6). We compiled important open issues found while conducting the analysis of the selected works, indicating the possible future research directions.

6.1. Datasets

The effectiveness of intrusion detection systems hinges critically on the quality and relevance of the datasets used for training and testing these systems. There is a pronounced reliance on a limited number of public datasets, such as CIC-IDS2017, NSL-KDD, CIC-DDoS2019, CSE-CIC-IDS2018, and KDD Cup 1999. While these datasets have been invaluable to the research community, they may not accurately reflect the breadth of real-world network security challenges.

The first issue with this concentrated use of specific datasets is the risk of model overfitting. IDS models trained on these datasets might perform exceptionally well in a lab environment, but they may fail to generalize to different or more recent network behaviors. This discrepancy arises because these datasets, which are often several years old, do not contain the latest attack signatures or reflect contemporary network traffic conditions. The landscape of cyber threats is dynamic, with new vulnerabilities and attack methodologies continually emerging. This rapid evolution renders existing datasets quickly obsolete. The absence of contemporary threats impairs the ability of IDSs to identify and mitigate current risks effectively.

Another critical aspect is the representation of legitimate network behavior in these datasets. Many of them emphasize attack scenarios, providing an imbalanced view that can lead to high false-positive rates in deployed IDS. A dataset predominantly consisting of attack vectors, without a corresponding diversity of normal behaviors, limits the system's ability to learn and distinguish between benign and malicious activities under real operational conditions.

There is an urgent need for regular updates to existing datasets and the development of new ones that mirror the current network environments. These updates should include recent attack types and tactics, ensuring that IDS can respond to the latest threats. Additionally, these datasets should reflect the varied and legitimate uses of network resources in contemporary settings, encompassing different network architectures, traffic, and user behavior. Besides, researchers should evaluate their solutions on available datasets containing real traffic data, as presented in Fig. 8 (e.g., CTU-13, ISCX2016, and N-BaIoT). The UGR'16 (Maciá-Fernández, Camacho, Magán-Carrión, García-Teodoro, & Therón, 2018) is a valuable dataset not used in the reviewed works. It comprises five months of real legitimate traffic collected from a Spanish ISP, which can be used in NIDS benchmarking.

6.2. Deep learning

Deep learning techniques have become popular for attack detection. The usage of DL models is highly concentrated in four traditional methods: CNN, LSTM, DNN, and GRU. The scientific community has been exploring more than 30 unique models, as listed in Table 5. However, further investigation of their applicability in different anomaly detection contexts is still an open issue.

The majority of the reviewed works apply deep learning to solve feature extraction or classification problems. A more in-depth analysis of solving regression, data generation, reconstruction, and the Markov decision process for implementing NIDS needs to be conducted.

Based on the reviewed works, we recommend a more in-depth exploration of Graph Convolutional Network models, such as vanilla GCN, ST-GCN, HGCN, and GC-LSTM. Also, Time Series Transformer and Attention Mechanism are propitious deep learning solutions rarely applied in intrusion detection. These models present promising results but are still not widely evaluated.

6.2.1. Unsupervised learning

We identified that unsupervised learning models are not commonly used. Training such models can be complex. Unlike supervised learning, which relies on labeled data, unsupervised learning has to learn from the underlying data patterns and structures without explicit guidance. As a result, developing and validating unsupervised models requires more advanced modeling and evaluation techniques, which can be costly.

Another challenge in developing unsupervised models is the need for suitable and representative datasets for training. Unsupervised learning requires diverse and complex datasets, which are not always feasible or available. Unsupervised learning models tend to create complex and latent representations, which make it difficult to understand the patterns learned. This lack of understanding can limit the confidence and adoption of these techniques in cybersecurity environments.

Our analysis have shown a significant concentration of work exploring Autoencoder and Stacked Autoencoder networks in unsupervised learning for detecting attacks. Other model variations, such as VAE, SSAE, DAE^1, and DAE^2, should be further evaluated and compared. Although autoencoders are powerful at reconstructing data, more unsupervised deep learning models must be tested to promote innovation and progress. For example, generative adversarial network variations are still not sufficiently explored in this context, leaving vast territory for future research: WGAN, WGAN-GP, and Bi-GAN.

6.2.2. Reinforcement learning

We also noticed that reinforcement learning has low adoption in detecting attacks on computer networks. Although reinforcement learning has been widely explored in other areas, e.g., games and robotics, its application in cybersecurity is still incipient. This may be due to the complexity of the attack detection environment, which may not be well suited to the reward model used in reinforcement learning, or to the difficulty of adequately simulating the network environment to train reinforcement learning agents. Considering their optimistic results in the reviewed works, we recommend researchers to conduct more experimentation on reinforcement learning models, such as Deep Q-Network, Recursive Neural Network, and Policy Gradient.

6.2.3. Decentralized architecture

Implementing centralized NIDS is a clear trend, as most analyzed works apply this architecture design. Decentralized systems are a promising area of research since NIDS may leverage parallel computing to scale and cope with the tremendous amount of traffic produced by modern networks. Scalability is a fundamental characteristic of a NIDS, considering the complexity of current networks. Nonetheless, it is a neglected study branch, as many authors need to discuss their system's adaption to high volumes of input data. We encourage researchers to design their solutions to run on multiple nodes, as this architecture may be more suitable to real-world scenarios.

6.3. Explainable artificial intelligence

In the past decade, significant progress has been made in Artificial Intelligence (AI), resulting in the adoption of algorithms to solve various problems. This success has led to increased model complexity and the use of black-box AI models that lack transparency. Many of the decisions taken by these models do not clearly express the reasons that led to that decision being made. In response to this challenge, Explainable AI (XAI) has been proposed to enhance the transparency of AI and facilitate its adoption in critical domains.

Despite its rapid growth and increasing acceptance, XAI remains an emerging field that lacks formality and agreed-upon definitions (Linardatos, Papastefanopoulos, & Kotsiantis, 2021). Among several studies dealing with XAI, the work by Roscher, Bohn, Duarte, and Garcke (2020) stands out. They discuss the requirements for using ML for scientific discovery and organize them into three core elements: transparency, interpretability, and explainability. An ML model is transparent if its construction process (including methods for model structure choices and fitting the parameters) can all be well described and motivated. In general, interpretability refers to the ability to make sense of an ML model obtained. The definition of interpretability takes into account the data and the model created. Thus, interpretability methods try to explain which input data was responsible for the model's prediction. Finally, explainability presents the relationship between the data, the model, and the user. Explainability is the ability to explain the model's operation, making its behavior more intelligible to humans.

The potential of using XAI is twofold. Firstly, it can justify algorithmic decisions, enabling users to understand and improve the model's performance. Secondly, XAI can contribute to knowledge discovery by revealing learned patterns (Nauta et al., 2023). XAI algorithms must be extensively validated to ensure their effectiveness and usefulness in achieving this potential. The increase in studies looking at the interpretability of deep learning models confirms the room for improvement in this area, not only by improving the model training pipeline but also by highlighting the flaws and how much they are lacking in performance. In general, XAI has few explored aspects and the potential to be unlocked in future work. We suggest new research to study how XAI methods, such as SHAP and LIME, may be integrated into NIDS solutions to assist in explaining their inference process.

7. Conclusion

We presented a thorough, empirical literature review on deep learning-based NIDS for the Software-Defined Network environment. This survey builds upon previous studies by meticulously analyzing each common step employed in cutting-edge intrusion detection solutions. These steps encompass benchmark datasets, data preprocessing, deep learning modeling, hyperparameter tuning, and performance evaluation.

Following the proposed methodology, we selected 105 final papers to form the basis of our review. The five most used datasets for developing, testing, and comparing NIDS are CIC-IDS2017, NSL-KDD, CIC-DDoS2019, CSE-CIC-IDS2018, and KDD Cup 1999. The most common attacks found in the data sources are DoS, DDoS, Botnet, Portscan, WebAttacks, and Brute force, highlighting the importance of studying such threats. Preprocessing is an essential step before feeding data to a machine learning model. We discussed the methods the reviewed works usually apply to data: normalization, cleaning, duplicate deletion, over and under-sampling, and feature engineering. The most used deep learning models are CNN, LSTM, DNN, and GRU, among more than 30 unique explored models. These are utilized to build NIDS following different implementation paradigms. The DL networks solve feature extraction, classification, regression, data generation, reconstruction, or Markov decision processes. They may function centrally or in a decentralized manner. Also, the systems can be supervised, unsupervised, or reinforcement-based. Most works did not describe their deep learning hyperparameter optimization process. Nonetheless, some hyperparameter search approaches are used by some studies, including empirical, random, and grid-based ones. The most commonly used metrics to evaluate the deep learning-based NIDS results are Recall, Accuracy, F1-score, Precision, False Positive Rate, and ROC curve. Many works still utilize accuracy in imbalanced datasets despite the consensus that it is not a trustworthy metric in such a context.

There has been an increasing number of published works in the regarded research field. The scientific community has tried to develop better solutions for the intrusion detection problem. Nevertheless, we identified some critical issues that remain open and require further investigation. More real-world, updated datasets are needed for reliable NIDS evaluation. Unsupervised learning has not been significantly explored in the literature. Since collecting labeled data in real environments is challenging, this solution paradigm is more practical and cheaper than the supervised one. Novel unsupervised NIDS studies may experiment with alternative deep learning models to solve problems other than reconstruction error. Our research revealed that reinforcement learning is an implementation paradigm rarely utilized in state-of-the-art NIDS. The examination of such solutions remains a future direction to be studied. Scalable, decentralized NIDS is also a promising future research topic. XAI is an evolving area that aims to provide insight into the inner workings of black-box machine learning models. Scientists may apply this solution to deep learning-based NIDS to understand their inference process, allowing for debugging and improvement.

CRediT authorship contribution statement

Vitor Gabriel da Silva Ruffo: Conceptualization, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing. **Daniel Matheus Brandão Lent:** Conceptualization, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing. **Mateus Komarchesqui:** Conceptualization, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing. **Vinícius Ferreira Schiavon:** Conceptualization, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing. **Marcos Vinicius Oliveira de Assis:** Conceptualization, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing. **Luiz Fernando Carvalho:**

Conceptualization, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing. **Mario Lemes Proença Jr.:** Conceptualization, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by CAPES, Brazil, due to the concession of scholarships and by the National Council for Scientific and Technological Development (CNPq) of Brazil under Grant of Project 306397/2022-6.

References

Abdulganiyu, O. H., Ait Tchakoucht, T., & Saheed, Y. K. (2023). A systematic literature review for network intrusion detection system (IDS). *International Journal of Information Security*, *22*(5), 1125–1162. http://dx.doi.org/10.1007/s10207-023-00682-2.

Ahmad, I., Wan, Z., & Ahmad, A. (2023). A big data analytics for DDOS attack detection using optimized ensemble framework in internet of things. *Internet of Things*, *23*, Article 100825. http://dx.doi.org/10.1016/j.iot.2023.100825.

Assis, M. V., Carvalho, L. F., Lloret, J., & Proença, M. L. (2021). A GRU deep learning system against attacks in software defined networks. *Journal of Network and Computer Applications*, *177*, Article 102942. http://dx.doi.org/10.1016/j.jnca.2020.102942.

Aydın, H., Orman, Z., & Aydın, M. A. (2022). A long short-term memory (LSTM)-based distributed denial of service (DDoS) detection and defense system design in public cloud network environment. *Computers & Security*, *118*, Article 102725. http://dx.doi.org/10.1016/j.cose.2022.102725.

Bilot, T., Madhoun, N. E., Agha, K. A., & Zouaoui, A. (2023). Graph neural networks for intrusion detection: A survey. *IEEE Access*, *11*, 49114–49139. http://dx.doi.org/10.1109/ACCESS.2023.3275789.

Bob Lantz, M. C. (2024). Mininet: An instant virtual network on your laptop (or other PC). http://mininet.org/. (Last Access 6 May 2024).

Brandao Lent, D. M., Novaes, M. P., Carvalho, L. F., Lloret, J., Rodrigues, J. J. P. C., & Proenca, M. L. (2022). A gated recurrent unit deep learning model to detect and mitigate distributed denial of service and portscan attacks. *IEEE Access*, *10*, 73229–73242. http://dx.doi.org/10.1109/access.2022.3190008.

Bårli, E. M., Yazidi, A., Viedma, E. H., & Haugerud, H. (2021). Dos and DDoS mitigation using variational autoencoders. *Computer Networks*, *199*, Article 108399. http://dx.doi.org/10.1016/j.comnet.2021.108399.

Caville, E., Lo, W. W., Layeghy, S., & Portmann, M. (2022). Anomal-E: A self-supervised network intrusion detection system based on graph neural networks. *Knowledge-Based Systems*, *258*, Article 110030. http://dx.doi.org/10.1016/j.knosys.2022.110030.

Cherian, M., & Varma, S. L. (2023). Secure SDN–IoT framework for ddos attack detection using deep learning and counter based approach. *Journal of Network and Systems Management*, *31*(3), http://dx.doi.org/10.1007/s10922-023-09749-w.

Choobdar, P., Naderan, M., & Naderan, M. (2021). Detection and multi-class classification of intrusion in software defined networks using stacked auto-encoders and CICIDS2017 dataset. *Wireless Personal Communications*, *123*(1), 437–471. http://dx.doi.org/10.1007/s11277-021-09139-y.

Cil, A. E., Yildiz, K., & Buldu, A. (2021). Detection of ddos attacks with feed forward based deep neural network model. *Expert Systems with Applications*, *169*, Article 114520. http://dx.doi.org/10.1016/j.eswa.2020.114520.

Community, R. S. F. (2024). Ryu SDN framework. https://ryu-sdn.org/. (Last Access 6 May 2024).

de Souza, C. A., Westphall, C. B., Machado, R. B., Loffi, L., Westphall, C. M., & Geronimo, G. A. (2022). Intrusion detection and prevention in fog based IoT environments: A systematic literature review. *Computer Networks*, *214*, Article 109154. http://dx.doi.org/10.1016/j.comnet.2022.109154.

Ding, S., Kou, L., & Wu, T. (2022). A GAN-based intrusion detection model for 5G enabled future metaverse. *Mobile Networks and Applications*, *27*(6), 2596–2610. http://dx.doi.org/10.1007/s11036-022-02075-6.

Ding, Q., & Li, J. (2022). Anogla: An efficient scheme to improve network anomaly detection. *Journal of Information Security and Applications*, *66*, Article 103149. http://dx.doi.org/10.1016/j.jisa.2022.103149.

Dinh, P. T., & Park, M. (2021). R-EDoS: Robust economic denial of sustainability detection in an SDN-based cloud through stochastic recurrent neural network. *IEEE Access*, *9*, 35057–35074. http://dx.doi.org/10.1109/access.2021.3061601.

Duan, X., Fu, Y., & Wang, K. (2023). Network traffic anomaly detection method based on multi-scale residual classifier. *Computer Communications*, *198*, 206–216. http://dx.doi.org/10.1016/j.comcom.2022.10.024.

Duy, P. T., Khoa, N. H., Hien, D. T. T., Hoang, H. D., & Pham, V. H. (2023). Investigating on the robustness of flow-based intrusion detection system against adversarial samples using generative adversarial networks. *Journal of Information Security and Applications*, *74*, Article 103472. http://dx.doi.org/10.1016/j.jisa.2023.103472.

Duy, P. T., Tien, L. K., Khoa, N. H., Hien, D. T. T., Nguyen, A. G. T., & Pham, V.-H. (2021). DIGFuPAS: Deceive IDS with GAN and function-preserving on adversarial samples in SDN-enabled networks. *Computers & Security*, *109*, Article 102367. http://dx.doi.org/10.1016/j.cose.2021.102367.

El-Ghamry, A., Darwish, A., & Hassanien, A. E. (2023). An optimized CNN-based intrusion detection system for reducing risks in smart farming. *Internet of Things*, *22*, Article 100709. http://dx.doi.org/10.1016/j.iot.2023.100709.

Elsayed, R. A., Hamada, R. A., Abdalla, M. I., & Elsaid, S. A. (2023). Securing IoT and SDN systems using deep-learning based automatic intrusion detection. *Ain Shams Engineering Journal*, *14*(10), Article 102211. http://dx.doi.org/10.1016/j.asej.2023.102211.

ElSayed, M. S., Le-Khac, N. A., Albahar, M. A., & Jurcut, A. (2021). A novel hybrid model for intrusion detection systems in SDNs based on CNN and a new regularization technique. *Journal of Network and Computer Applications*, *191*, Article 103160. http://dx.doi.org/10.1016/j.jnca.2021.103160.

Ferrag, M. A., Friha, O., Hamouda, D., Maglaras, L., & Janicke, H. (2022). Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. *IEEE Access*, *10*, 40281–40306. http://dx.doi.org/10.1109/access.2022.3165809.

Fouladi, R. F., Ermiş, O., & Anarim, E. (2022). A DDoS attack detection and countermeasure scheme based on DWT and auto-encoder neural network for SDN. *Computer Networks*, *214*, Article 109140. http://dx.doi.org/10.1016/j.comnet.2022.109140.

Foundation, O. N. (2024). Open network operating system (ONOS) SDN controller for SDN/NFV solutions. https://opennetworking.org/onos/. (Last Access 6 May 2024).

Fox, G. T., & Boppana, R. V. (2023). On early detection of anomalous network flows. *IEEE Access*, *11*, 68588–68603. http://dx.doi.org/10.1109/access.2023.3291686.

Fred Klassen, A. (2024). Tcpreplay - pcap editing and replaying utilities. https://tcpreplay.appneta.com/. (Last Access 6 May 2024).

Friha, O., Ferrag, M. A., Benbouzid, M., Berghout, T., Kantarci, B., & Choo, K. K. R. (2023). 2DF-IDS: Decentralized and differentially private federated learning-based intrusion detection system for industrial IoT. *Computers & Security*, *127*, Article 103097. http://dx.doi.org/10.1016/j.cose.2023.103097.

Friha, O., Ferrag, M. A., Shu, L., Maglaras, L., Choo, K. K. R., & Nafaa, M. (2022). FELIDS: Federated learning-based intrusion detection system for agricultural internet of things. *Journal of Parallel and Distributed Computing*, *165*, 17–31. http://dx.doi.org/10.1016/j.jpdc.2022.03.003.

Fu, Y., Duan, X., Wang, K., & Li, B. (2022). Low-rate denial of service attack detection method based on time-frequency characteristics. *Journal of Cloud Computing*, *11*(1), http://dx.doi.org/10.1186/s13677-022-00308-3.

Garcia, N., Alcaniz, T., González-Vidal, A., Bernabe, J. B., Rivera, D., & Skarmeta, A. (2021). Distributed real-time SlowDoS attacks detection over encrypted traffic using artificial intelligence. *Journal of Network and Computer Applications*, *173*, Article 102871. http://dx.doi.org/10.1016/j.jnca.2020.102871.

Gupta, N., Jindal, V., & Bedi, P. (2021). LIO-IDS: Handling class imbalance using LSTM and improved one-vs-one technique in intrusion detection system. *Computer Networks*, *192*, Article 108076. http://dx.doi.org/10.1016/j.comnet.2021.108076.

Gupta, N., Jindal, V., & Bedi, P. (2022). CSE-IDS: Using cost-sensitive deep learning and ensemble algorithms to handle class imbalance in network-based intrusion detection systems. *Computers & Security*, *112*, Article 102499. http://dx.doi.org/10.1016/j.cose.2021.102499.

Gupta, S. K., Tripathi, M., & Grover, J. (2022). Hybrid optimization and deep learning based intrusion detection system. *Computers & Electrical Engineering*, *100*, Article 107876. http://dx.doi.org/10.1016/j.compeleceng.2022.107876.

H., S. C., Rao, K. V., & Prasad, M. H. M. K. (2023). Deep neural network empowered bi-directional cross GAN in context of classifying DDoS over flash crowd event on web server. *Multimedia Tools and Applications*, *82*(24), 37303–37326. http://dx.doi.org/10.1007/s11042-023-15030-8.

Hairab, B. I., Said Elsayed, M., Jurcut, A. D., & Azer, M. A. (2022). Anomaly detection based on CNN and regularization techniques against zero-day attacks in IoT networks. *IEEE Access*, *10*, 98427–98440. http://dx.doi.org/10.1109/access.2022.3206367.

Hettich, S., & Bay, S. D. (1999). *KDD Cup 1999 Data*. University of California, Department of Information and Computer Science, https://kdd.ics.uci.edu/. (Last Access 6 May 2024).

Hidalgo, C., Vaca, M., Nowak, M. P., Frölich, P., Reed, M., Al-Naday, M., et al. (2022). Detection, control and mitigation system for secure vehicular communication. *Vehicular Communications*, *34*, Article 100425. http://dx.doi.org/10.1016/j.vehcom.2021.100425.

Hnamte, V., & Hussain, J. (2023). An efficient DDoS attack detection mechanism in SDN environment. *International Journal of Information Technology*, *15*(5), 2623–2636. http://dx.doi.org/10.1007/s41870-023-01332-5.

Houda, Z. A. E., Hafid, A. S., & Khoukhi, L. (2023). MiTFed: A privacy preserving collaborative network attack mitigation framework based on federated learning using SDN and blockchain. *IEEE Transactions on Network Science and Engineering*, *10*(4), 1985–2001. http://dx.doi.org/10.1109/tnse.2023.3237367.

Hu, B., Bi, Y., Zhi, M., Zhang, K., Yan, F., Zhang, Q., et al. (2022). A deep one-class intrusion detection scheme in software-defined industrial networks. *IEEE Transactions on Industrial Informatics*, *18*(6), 4286–4296. http://dx.doi.org/10.1109/tii.2021.3133300.

Huang, H., Ye, P., Hu, M., & Wu, J. (2023). A multi-point collaborative ddos defense mechanism for IIoT environment. *Digital Communications and Networks*, *9*(2), 590–601. http://dx.doi.org/10.1016/j.dcan.2022.04.008.

Illy, P., & Kaddoum, G. (2023). A collaborative DNN-based low-latency IDPS for mission-critical smart factory networks. *IEEE Access*, *11*, 96317–96329. http://dx.doi.org/10.1109/access.2023.3311822.

Illy, P., Kaddoum, G., de Araujo-Filho, P. F., Kaur, K., & Garg, S. (2022). A hybrid multistage DNN-based collaborative IDPS for high-risk smart factory networks. *IEEE Transactions on Network and Service Management*, *19*(4), 4273–4283. http://dx.doi.org/10.1109/tnsm.2022.3202801.

Imrana, Y., Xiang, Y., Ali, L., & Abdul-Rauf, Z. (2021). A bidirectional LSTM deep learning approach for intrusion detection. *Expert Systems with Applications*, *185*, Article 115524. http://dx.doi.org/10.1016/j.eswa.2021.115524.

Janabi, A. H., Kanakis, T., & Johnson, M. (2022). Convolutional neural network based algorithm for early warning proactive system security in software defined networks. *IEEE Access*, *10*, 14301–14310. http://dx.doi.org/10.1109/ACCESS.2022.3148134.

Javed, Y., Khayat, M. A., Elghariani, A. A., & Ghafoor, A. (2023). PRISM: A hierarchical intrusion detection architecture for large-scale cyber networks. *IEEE Transactions on Dependable and Secure Computing*, *20*(6), 5070–5086. http://dx.doi.org/10.1109/TDSC.2023.3240315.

Kaur, G., & Kakkar, D. (2022). Hybrid optimization enabled trust-based secure routing with deep learning-based attack detection in VANET. *Ad Hoc Networks*, *136*, Article 102961. http://dx.doi.org/10.1016/j.adhoc.2022.102961.

Kim, S., Yoon, S., Cho, J. H., Kim, D. S., Moore, T. J., Free-Nelson, F., et al. (2022). DIVERGENCE: Deep reinforcement learning-based adaptive traffic inspection and moving target defense countermeasure framework. *IEEE Transactions on Network and Service Management*, *19*(4), 4834–4846. http://dx.doi.org/10.1109/tnsm.2021.3139928.

Kumar, P., Kumar, R., Aljuhani, A., Javeed, D., Jolfaei, A., & Islam, A. K. M. N. (2023). Digital twin-driven SDN for smart grid: A deep learning integrated blockchain for cybersecurity. *Solar Energy*, *263*, Article 111921. http://dx.doi.org/10.1016/j.solener.2023.111921.

Li, W., Meng, W., & Kwok, L. F. (2016). A survey on OpenFlow-based software defined networks: Security challenges and countermeasures. *Journal of Network and Computer Applications*, *68*, 126–139. http://dx.doi.org/10.1016/j.jnca.2016.04.011.

Linardatos, P., Papastefanopoulos, V., & Kotsiantis, S. (2021). Explainable AI: A review of machine learning interpretability methods. *Entropy*, *23*(1), http://dx.doi.org/10.3390/e23010018.

Liu, Y., Zhi, T., Shen, M., Wang, L., Li, Y., & Wan, M. (2022). Software-defined ddos detection with information entropy analysis and optimized deep learning. *Future Generation Computer Systems*, *129*, 99–114. http://dx.doi.org/10.1016/j.future.2021.11.009.

Long, Z., & Jinsong, W. (2022). A hybrid method of entropy and SSAE-SVM based DDoS detection and mitigation mechanism in SDN. *Computers & Security*, *115*, Article 102604. http://dx.doi.org/10.1016/j.cose.2022.102604.

Lopes, I. O., Zou, D., Abdulqadder, I. H., Ruambo, F. A., Yuan, B., & Jin, H. (2022). Effective network intrusion detection via representation learning: A denoising AutoEncoder approach. *Computer Communications*, *194*, 55–65. http://dx.doi.org/10.1016/j.comcom.2022.07.027.

Ltd., S. I. T. (2024a). DNS-STATS. https://dns-stats.org/. (Last Access 6 May 2024).

Ltd., O. (2024b). OMNeT++ discrete event simulator. https://omnetpp.org/. (Last Access 6 May 2024).

M., G., & Sethuraman, S. C. (2023). A comprehensive survey on deep learning based malware detection techniques. *Computer Science Review*, *47*, Article 100529. http://dx.doi.org/10.1016/j.cosrev.2022.100529.

Maciá-Fernández, G., Camacho, J., Magán-Carrión, R., García-Teodoro, P., & Therón, R. (2018). UGR '16: A new dataset for the evaluation of cyclostationarity-based network IDSs. *Computers & Security*, *73*, 411–424. http://dx.doi.org/10.1016/j.cose.2017.11.004.

Mansour, R. F. (2022). Blockchain assisted clustering with intrusion detection system for industrial internet of things environment. *Expert Systems with Applications*, *207*, Article 117995. http://dx.doi.org/10.1016/j.eswa.2022.117995.

McCauley, M. (2024). The POX network software platform. https://github.com/noxrepo/pox. (Last Access 6 May 2024).

Melis, A., Sadi, A. A., Berardi, D., Callegati, F., & Prandini, M. (2023). A systematic literature review of offensive and defensive security solutions with software defined network. *IEEE Access*, *11*, 93431–93463. http://dx.doi.org/10.1109/ACCESS.2023.3276238.

Mustapha, A., Khatoun, R., Zeadally, S., Chbib, F., Fadlallah, A., Fahs, W., et al. (2023). Detecting ddos attacks using adversarial neural network. *Computers & Security*, *127*, Article 103117. http://dx.doi.org/10.1016/j.cose.2023.103117.

Myneni, S., Chowdhary, A., Huang, D., & Alshamrani, A. (2022). SmartDefense: A distributed deep defense against DDoS attacks with edge computing. *Computer Networks*, *209*, Article 108874. http://dx.doi.org/10.1016/j.comnet.2022.108874.

Nadeem, M. W., Goh, H. G., Aun, Y., & Ponnusamy, V. (2023). Detecting and mitigating botnet attacks in software-defined networks using deep learning techniques. *IEEE Access*, *11*, 49153–49171. http://dx.doi.org/10.1109/access.2023.3277397.

Nauta, M., Trienes, J., Pathak, S., Nguyen, E., Peters, M., Schmitt, Y., et al. (2023). From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable AI. *ACM Computing Surveys*, *55*(13s), http://dx.doi.org/10.1145/3583558.

Nguyen, X. H., & Le, K. H. (2023). Robust detection of unknown DoS/DDoS attacks in IoT networks using a hybrid learning model. *Internet of Things*, *23*, Article 100851. http://dx.doi.org/10.1016/j.iot.2023.100851.

Nisar, K., Jimson, E. R., Hijazi, M. H. A., Welch, I., Hassan, R., Aman, A. H. M., et al. (2020). A survey on the architecture, application, and security of software defined networking: Challenges and open issues. *Internet of Things*, *12*, Article 100289. http://dx.doi.org/10.1016/j.iot.2020.100289.

Novaes, M. P., Carvalho, L. F., Lloret, J., & Proença, M. L. (2021). Adversarial deep learning approach detection and defense against DDoS attacks in SDN environments. *Future Generation Computer Systems*, *125*, 156–167. http://dx.doi.org/10.1016/j.future.2021.06.047.

Nuaimi, M., Fourati, L. C., & Hamed, B. B. (2023). Intelligent approaches toward intrusion detection systems for industrial internet of things: A systematic comprehensive review. *Journal of Network and Computer Applications*, *215*, Article 103637. http://dx.doi.org/10.1016/j.jnca.2023.103637.

O. Lopes, I., Zou, D., Abdulqadder, I. H., Akbar, S., Li, Z., Ruambo, F., et al. (2023). Network intrusion detection based on the temporal convolutional model. *Computers & Security*, *135*, Article 103465. http://dx.doi.org/10.1016/j.cose.2023.103465.

Oliveira, M. d. (2024). hping3 | kali linux tools. https://www.kali.org/tools/hping3/. (Last Access 6 May 2024).

Ozkan-Okay, M., Akin, E., Aslan, Ö., Kosunalp, S., Iliev, T., Stoyanov, I., et al. (2024). A comprehensive survey: Evaluating the efficiency of artificial intelligence and machine learning techniques on cyber security solutions. *IEEE Access*, *12*, 12229–12256. http://dx.doi.org/10.1109/ACCESS.2024.3355547.

Phan, T. V., & Bauschert, T. (2022). DeepAir: Deep reinforcement learning for adaptive intrusion response in software-defined networks. *IEEE Transactions on Network and Service Management*, *19*(3), 2207–2218. http://dx.doi.org/10.1109/tnsm.2022.3158468.

Phu, A. T., Li, B., Ullah, F., Ul Huque, T., Naha, R., Babar, M. A., et al. (2023). Defending SDN against packet injection attacks using deep learning. *Computer Networks*, *234*, Article 109935. http://dx.doi.org/10.1016/j.comnet.2023.109935.

Polat, H., Türkoˇ glu, M., Polat, O., & Şengür, A. (2022). A novel approach for accurate detection of the DDoS attacks in SDN-based SCADA systems based on deep recurrent neural networks. *Expert Systems with Applications*, *197*, Article 116748. http://dx.doi.org/10.1016/j.eswa.2022.116748.

Presekal, A., Ştefanov, A., Rajkumar, V. S., & Palensky, P. (2023). Attack graph model for cyber-physical power systems using hybrid deep learning. *IEEE Transactions on Smart Grid*, *14*(5), 4007–4020. http://dx.doi.org/10.1109/tsg.2023.3237011.

Qazi, E. u.-H., Imran, M., Haider, N., Shoaib, M., & Razzak, I. (2022). An intelligent and efficient network intrusion detection system using deep learning. *Computers & Electrical Engineering*, *99*, Article 107764. http://dx.doi.org/10.1016/j.compeleceng.2022.107764.

Ravi, V., Chaganti, R., & Alazab, M. (2022). Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system. *Computers & Electrical Engineering*, *102*, Article 108156. http://dx.doi.org/10.1016/j.compeleceng.2022.108156.

Roscher, R., Bohn, B., Duarte, M. F., & Garcke, J. (2020). Explainable machine learning for scientific insights and discoveries. *IEEE Access*, *8*, 42200–42216. http://dx.doi.org/10.1109/ACCESS.2020.2976199.

Sabeel, U., Heydari, S. S., El-Khatib, K., & Elgazzar, K. (2024). Unknown, atypical and polymorphic network intrusion detection: A systematic survey. *IEEE Transactions on Network and Service Management*, *21*(1), 1190–1212. http://dx.doi.org/10.1109/TNSM.2023.3298533.

Sahu, S. K., Mohapatra, D. P., Rout, J. K., Sahoo, K. S., Pham, Q. V., & Dao, N. N. (2022). A LSTM-FCNN based multi-class intrusion detection using scalable framework. *Computers & Electrical Engineering*, *99*, Article 107720. http://dx.doi.org/10.1016/j.compeleceng.2022.107720.

Sarıkaya, A., Kılıç, B. G., & Demirci, M. (2023). RAIDS: Robust autoencoder-based intrusion detection system model against adversarial attacks. *Computers & Security*, *135*, Article 103483. http://dx.doi.org/10.1016/j.cose.2023.103483.

Sattari, F., Farooqi, A. H., Qadir, Z., Raza, B., Nazari, H., & Almutiry, M. (2022). A hybrid deep learning approach for bottleneck detection in IoT. *IEEE Access*, *10*, 77039–77053. http://dx.doi.org/10.1109/access.2022.3188635.

Sayed, M. S. E., Le-Khac, N. A., Azer, M. A., & Jurcut, A. D. (2022). A flow-based anomaly detection approach with feature selection method against ddos attacks in SDNs. *IEEE Transactions on Cognitive Communications and Networking*, *8*(4), 1862–1880. http://dx.doi.org/10.1109/tccn.2022.3186331.

Shaji, N. S., Jain, T., Muthalagu, R., & Pawar, P. M. (2023). Deep-discovery: Anomaly discovery in software-defined networks using artificial neural networks. *Computers & Security*, *132*, Article 103320. http://dx.doi.org/10.1016/j.cose.2023.103320.

Sharafaldin, I., Lashkari, A. H., Ghorbani, A. A., et al. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, *1*, 108–116. http://dx.doi.org/10.5220/0006639801080116.

Sharafaldin, I., Lashkari, A. H., Hakak, S., & Ghorbani, A. A. (2019). Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy. http://dx.doi.org/10.1109/CCST.2019.8888419.

Shu, J., Zhou, L., Zhang, W., Du, X., & Guizani, M. (2021). Collaborative intrusion detection for VANETs: A deep learning-based distributed SDN approach. *IEEE Transactions on Intelligent Transportation Systems*, *22*(7), 4519–4530. http://dx.doi.org/10.1109/tits.2020.3027390.

Shukla, P. K., Maheshwary, P., Subramanian, E., Shilpa, V. J., & Varma, P. R. K. (2023). Traffic flow monitoring in software-defined network using modified recursive learning. *Physical Communication*, *57*, Article 101997. http://dx.doi.org/10.1016/j.phycom.2022.101997.

Siniosoglou, I., Radoglou-Grammatikis, P., Efstathopoulos, G., Fouliras, P., & Sarigiannidis, P. (2021). A unified deep learning anomaly detection and classification approach for smart grid environments. *IEEE Transactions on Network and Service Management*, *18*(2), 1137–1151. http://dx.doi.org/10.1109/tnsm.2021.3078381.

Sivanesan, N., & Archana, K. S. (2023). Detecting distributed denial of service (DDoS) in SD-IoT environment with enhanced firefly algorithm and convolution neural network. *Optical and Quantum Electronics*, *55*(5), http://dx.doi.org/10.1007/s11082-023-04553-x.

SolarWinds Worldwide, L. (2024). GNS3 | the software that empowers network professionals. https://www.gns3.com/. (Last Access 6 May 2024).

Soltani, M., Siavoshani, M. J., & Jahangir, A. H. (2021). A content-based deep intrusion detection system. *International Journal of Information Security*, *21*(3), 547–562. http://dx.doi.org/10.1007/s10207-021-00567-2.

Song, D., Yuan, X., Li, Q., Zhang, J., Sun, M., Fu, X., et al. (2023). Intrusion detection model using gene expression programming to optimize parameters of convolutional neural network for energy internet. *Applied Soft Computing*, *134*, Article 109960. http://dx.doi.org/10.1016/j.asoc.2022.109960.

Sood, K., Nosouhi, M. R., Nguyen, D. D. N., Jiang, F., Chowdhury, M., & Doss, R. (2023). Intrusion detection scheme with dimensionality reduction in next generation networks. *IEEE Transactions on Information Forensics and Security*, *18*, 965–979. http://dx.doi.org/10.1109/tifs.2022.3233777.

Taheri, R., Ahmed, H., & Arslan, E. (2023). Deep learning for the security of software-defined networks: A review. *Cluster Computing*, *26*(5), 3089–3112. http://dx.doi.org/10.1007/s10586-023-04069-9.

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications*. http://dx.doi.org/10.1109/CISDA.2009.5356528.

Tayfour, O. E., Mubarakali, A., Tayfour, A. E., Marsono, M. N., Hassan, E., & Abdelrahman, A. M. (2023). Adapting deep learning-LSTM method using optimized dataset in SDN controller for secure IoT. *Soft Computing*, http://dx.doi.org/10.1007/s00500-023-08348-w.

Tsogbaatar, E., Bhuyan, M. H., Taenaka, Y., Fall, D., Gonchigsumlaa, K., Elmroth, E., et al. (2021). Del-IoT: A deep ensemble learning approach to uncover anomalies in IoT. *Internet of Things*, *14*, Article 100391. http://dx.doi.org/10.1016/j.iot.2021.100391.

Udas, P. B., Karim, M. E., & Roy, K. S. (2022). SPIDER: A shallow PCA based network intrusion detection system with enhanced recurrent neural networks. *Journal of King Saud University - Computer and Information Sciences*, *34*(10), 10246–10272. http://dx.doi.org/10.1016/j.jksuci.2022.10.019.

Van Engelen, J. E., & Hoos, H. H. (2020). A survey on semi-supervised learning. *Machine Learning*, *109*(2), 373–440. http://dx.doi.org/10.1007/s10994-019-05855-6.

Vatambeti, R., Venkatesh, D., Mamidisetti, G., Damera, V. K., Manohar, M., & Yadav, N. S. (2023). Prediction of ddos attacks in agriculture 4.0 with the help of prairie dog optimization algorithm with IDSNet. *Scientific Reports*, *13*(1), http://dx.doi.org/10.1038/s41598-023-42678-x.

Wang, K., Cui, Y., Qian, Q., Chen, Y., Guo, C., & Shen, G. (2023). USAGE: Uncertain flow graph and spatio-temporal graph convolutional network-based saturation attack detection method. *Journal of Network and Computer Applications*, *219*, Article 103722. http://dx.doi.org/10.1016/j.jnca.2023.103722.

Wette, P., Dräxler, M., Schwabe, A., Wallaschek, F., Zahraee, M. H., & Karl, H. (2014). MaxiNet: Distributed emulation of software-defined networks. In *2014 IFIP networking conference*. http://dx.doi.org/10.1109/IFIPNetworking.2014.6857078.

Xue, H., & Jing, B. (2023). SDN attack identification model based on CNN algorithm. *IEEE Access*, *11*, 87652–87666. http://dx.doi.org/10.1109/access.2023.3296798.

Yang, Z., Liu, X., Li, T., Wu, D., Wang, J., Zhao, Y., et al. (2022). A systematic literature review of methods and datasets for anomaly-based network intrusion detection. *Computers & Security*, *116*, Article 102675. http://dx.doi.org/10.1016/j.cose.2022.102675.

Yang, X., Song, Z., King, I., & Xu, Z. (2023). A survey on deep semi-supervised learning. *IEEE Transactions on Knowledge and Data Engineering*, *35*(9), 8934–8954. http://dx.doi.org/10.1109/TKDE.2022.3220219.

Yeom, S., Choi, C., & Kim, K. (2022). LSTM-based collaborative source-side DDoS attack detection. *IEEE Access*, *10*, 44033–44045. http://dx.doi.org/10.1109/access.2022.3169616.

Yousuf, O., & Mir, R. N. (2022). DDoS attack detection in internet of things using recurrent neural network. *Computers & Electrical Engineering*, *101*, Article 108034. http://dx.doi.org/10.1016/j.compeleceng.2022.108034.

Yungaicela-Naula, N. M., Vargas-Rosales, C., & Perez-Diaz, J. A. (2021). SDN-based architecture for transport and application layer DDoS attack detection by using machine and deep learning. *IEEE Access*, *9*, 108495–108512. http://dx.doi.org/10.1109/access.2021.3101650.

Zainudin, A., Ahakonye, L. A. C., Akter, R., Kim, D. S., & Lee, J. M. (2023). An efficient hybrid-DNN for DDoS detection and classification in software-defined IIoT networks. *IEEE Internet of Things Journal*, *10*(10), 8491–8504. http://dx.doi.org/10.1109/jiot.2022.3196942.

Zavrak, S., & Iskefiyeli, M. (2023). Flow-based intrusion detection on software-defined networks: A multivariate time series anomaly detection approach. *Neural Computing and Applications*, *35*(16), 12175–12193. http://dx.doi.org/10.1007/s00521-023-08376-5.

Zhang, P., He, F., Zhang, H., Hu, J., Huang, X., Wang, J., et al. (2023). Real-time malicious traffic detection with online isolation forest over sd-wan. *IEEE Transactions on Information Forensics and Security*, *18*, 2076–2090. http://dx.doi.org/10.1109/TIFS.2023.3262121.

Zhou, H., Zheng, Y., Jia, X., & Shu, J. (2023). Collaborative prediction and detection of DDoS attacks in edge computing: A deep learning-based approach with distributed SDN. *Computer Networks*, *225*, Article 109642. http://dx.doi.org/10.1016/j.comnet.2023.109642.