




## Article

# Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review

Tariq Emad Ali <sup>†</sup>, Yung-Wey Chong <sup>\*,†</sup> and Selvakumar Manickam <sup>†</sup>

National Advanced IPv6 Centre, Universiti Sains Malaysia, Gelugor 11800, Penang, Malaysia

\* Correspondence: chong@usm.my

† These authors contributed equally to this work.

**Abstract:** The recent advancements in security approaches have significantly increased the ability to identify and mitigate any type of threat or attack in any network infrastructure, such as a software-defined network (SDN), and protect the internet security architecture against a variety of threats or attacks. Machine learning (ML) and deep learning (DL) are among the most popular techniques for preventing distributed denial-of-service (DDoS) attacks on any kind of network. The objective of this systematic review is to identify, evaluate, and discuss new efforts on ML/DL-based DDoS attack detection strategies in SDN networks. To reach our objective, we conducted a systematic review in which we looked for publications that used ML/DL approaches to identify DDoS attacks in SDN networks between 2018 and the beginning of November 2022. To search the contemporary literature, we have extensively utilized a number of digital libraries (including IEEE, ACM, Springer, and other digital libraries) and one academic search engine (Google Scholar). We have analyzed the relevant studies and categorized the results of the SLR into five areas: (i) The different types of DDoS attack detection in ML/DL approaches; (ii) the methodologies, strengths, and weaknesses of existing ML/DL approaches for DDoS attacks detection; (iii) benchmarked datasets and classes of attacks in datasets used in the existing literature; (iv) the preprocessing strategies, hyperparameter values, experimental setups, and performance metrics used in the existing literature; and (v) current research gaps and promising future directions.

**Keywords:** machine learning; deep learning; distributed denial-of-service; datasets



**Citation:** Ali, T.E.; Chong, Y.-W.; Manickam, S. Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review. *Appl. Sci.* **2023**, *13*, 3183. <https://doi.org/10.3390/app13053183>

Academic Editor: Luis Javier Garcia Villalba

Received: 19 January 2023

Revised: 23 February 2023

Accepted: 26 February 2023

Published: 2 March 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

With the increasing demand for high-quality multimedia content, software-defined networking (SDN) has been proposed as the future of internet architecture. In this network paradigm, the control plane (which is the brains of the network) and the data plane (which is the muscle) are decoupled [1]. SDN models include SDN controllers, as well as southbound and northbound APIs. This architecture provides a programmable and centralized network that can dynamically provision services [2]. OpenFlow (OF) is a standard and open protocol used in SDN that explains how a centralized controller configures and governs the control layer in the network. The data in SDN is kept in Mac tables and routing tables and is handled by various sophisticated switching and routing protocols. These tables are utilized to create the forwarding plane in traditional networks [3].

Today's society relies heavily on the internet, which is essential for economic transactions, education, and communication. However, along with its many benefits, the internet has experienced an increase in criminal activity, such as hacking, spreading false information, and denial-of-service (DoS) attacks. A DoS attack occurs when a legitimate service, system or network is made inaccessible to its intended users. A DDoS attack, a subcategory of DoS attacks, involves an attacker breaching multiple computing systems in order to disrupt a specific target's regular traffic [4].

Defending against DoS and DDoS attacks is more challenging in SDN than in traditional networks. These types of attacks have become significant threats to computer networks, causing a decline in network performance by consuming available resources and disabling services. An effective DoS/DDoS attack intentionally depletes resources and prevents hosts from accessing the targeted service. In SDNs, a DoS/DDoS attack can overwhelm the control plane, data plane, or control plane bandwidth, potentially bringing down the entire network. An attack on the data plane may consume all of the OpenFlow switch's limited flow table RAM, resulting in the discarding of packets and the inability to install newly received flow rules. DoS/DDoS attacks on the data plane may also involve the generation of a large number of new flows that do not correspond to flow table entries. These packets are buffered by the switch, and if the buffer fills up, the entire packet is sent to the controller rather than just the headers through packet-in messages. This can cause delays in installing new flow rules and higher communication bandwidth use [5].

The primary distinction between DoS and DDoS assaults is that DoS utilizes many internet connections to take the victim's computer network offline, whereas DDoS assaults use a network of devices controlled by the attacker. DDoS assaults are more challenging to detect and trace because they are launched from various locations, and the attack volume used is enormous. DDoS assaults are carried out differently from DoS attacks, which are often carried out via a script or a DoS tool like Low-Orbit Ion Cannon. Types of DoS attacks include buffer overflows, ICMP floods, teardrop attacks, and flooding assaults, whereas types of DDoS attacks include volumetric attacks, fragmentation attacks, application layer attacks, and protocol attacks. DDoS assaults are more destructive than DoS attacks because they involve several systems, making it more challenging for security teams and products to pinpoint the source of the attack [6].

The aforementioned examples highlight the requirement for a reliable approach to identifying DDoS assaults. DDoS assaults may be detected using a variety of approaches, including statistical analysis, ML/DL, etc. Among these, deep learning approaches are the most effective at identifying DDoS assaults. The following are shortcomings of the alternative approaches that have been studied to date:

- **Statistical Methods Limitations:** The limitations of various DDoS detection approaches have been studied, including statistical and machine learning (ML) methods. Statistical methods are based on past network flow information, which may not accurately describe current network traffic due to evolving hostile network flows. Such techniques rely heavily on user-defined criteria, which need to be able to change dynamically in order to keep up with changes in the network. Statistical techniques such as entropy and correlation require a significant amount of computational effort, making them unsuitable for real-time detection [7]. ML methods work effectively on a small amount of data and determine the statistical properties of attacks before classifying or valuing them. However, they require routine model updates to reflect changes in attack patterns, and certain algorithms can take a very long time to test [8].
- **Machine Learning (ML) Limitations:** Even when applying ML principles to a tiny quantity of data, it can function quite effectively. The ML first determines the assault's statistical properties before classifying or valuing them. Additionally, it requires routine model updates in order to reflect changes in attack patterns [9]. ML techniques address this problem by decomposing it into manageable subproblems, addressing those subproblems, and then providing the full solution. ML algorithms typically require a short amount of time to train and a considerably longer amount of time to test [10].

DL techniques can effectively identify DDoS attacks, as the data can be classified and the features extracted using DL algorithms, unlike in ML which needs to extract the features in different algorithms before inserting them into the model. In today's security environment, a detection system that can handle data unavailability is a necessity. Although labels for valid traffic are frequently accessible, labels for malicious traffic are less common. DL methods are capable of extracting information from incomplete data [11], and are appropriate for recognizing low-rate assaults. To recognize low-rate assaults,

historical data are necessary, which DL techniques use to discover long-term relationships of temporal patterns [12]. As a result, in circumstances where such data are available DL techniques can be very helpful. During the training phase, DL methods perform intricate mathematical operations across a variety of hidden layers and parameters [13]. Quantum computing has shown great promise in a variety of fields, including artificial intelligence (AI), cybersecurity, and medical research. Quantum computing can help AIs to solve more complicated issues by speeding up computation. It can be used with both SML and DL models for quick training or other enhancements. By addressing complicated issues that need vast datasets and are demanding to process, quantum computing can enhance the capabilities of deep learning [14,15].

Compared to other review studies in the literature, Table 1 illustrates that the majority of these studies have not provided a comprehensive evaluation of the preparation techniques, benefits, and types of attacks used in the analyzed datasets. In contrast, our systematic study presents an extensive review of various deep-learning techniques for detecting DDoS attacks. Through this research, we have identified a gap in the literature, namely, that a comprehensive evaluation of deep learning methods for DDoS detection remains lacking. Our study contributes to addressing this gap by providing a comprehensive review and analysis of the strengths and weaknesses of different deep learning approaches for detecting DDoS attacks. As such, our review provides valuable insights into the current state-of-the-art in DDoS attack detection using deep learning techniques.

**Table 1.** Comparison of various research papers in detail (✓ = Yes; × = No).

Review Article	Ferrag et al. [16]	Aleesa et al. [17]	Gamage et al. [18]	Ahmad et al. [19]	Ahmad et al. [20]	This Article
Focused	Cyber security intrusion detection	IDS	NID	IDS	IoT security	DDoS
ML/DL	DL	DL	DL	ML/DL	ML/DL	ML/DL
Systematic study	×	✓	DL	✓	✓	✓
Taxonomy	×	✓	✓	✓	×	✓
Preprocessing strategy	×	×	×	×	×	✓
Types of attack used in existing literature from the datasets	×	✓	×	×	×	✓
Strengths	×	×	✓	×	×	✓
Weaknesses	×	×	✓	✓	✓	✓
Research gaps	×	×	×	×	×	✓

We reviewed DDoS assaults detection systems based on DL techniques in this research using the SLR protocol, and offer the following findings:

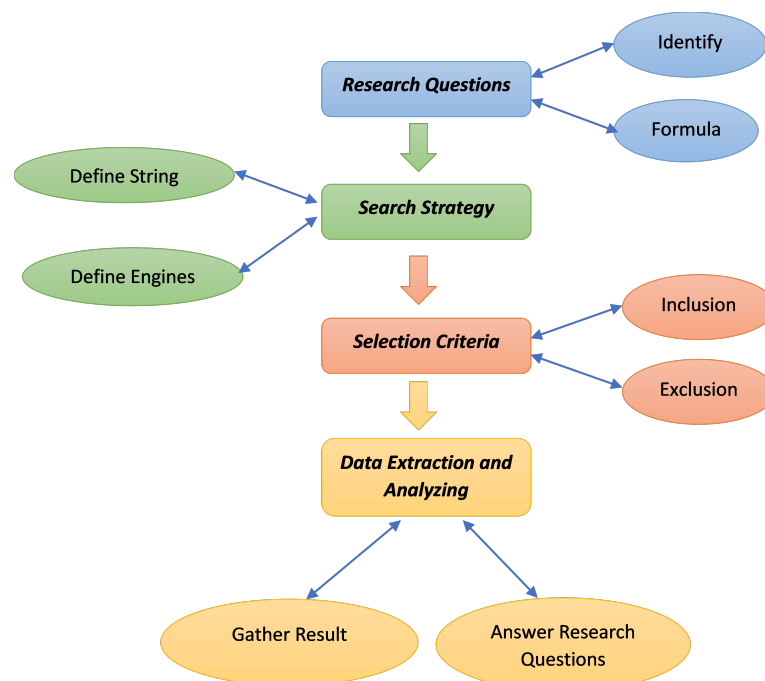
- Based on common criteria, modern DDoS attack detection technologies involving deep learning algorithms have been identified and grouped.
- The methodology, benefits, and drawbacks of current ML/DL systems for detecting DDoS assaults have been outlined.
- The different kinds of assaults in the datasets utilized in recent studies as well as the accessible DDoS benchmarked datasets have been compiled.
- The core of our review was focused on data pre-processing techniques, hyperparameter adjustments, testing configurations, and the quality measures used by current ML/DL systems for DDoS attack detection.
- The main purpose of the study was to identify areas for future research in this field and to highlight current research gaps.

The remainder of this review is structured as follows: the SLR protocol is explained in Section 2; Section 3 discusses current ML/DL methods that have been employed in the literature for detection of DDoS assaults; in Section 4, the methodology, advantages, and disadvantages of different studies are discussed; the available benchmarked DDoS datasets and classes of attacks in the datasets commonly used in the literature are described in Section 5; preprocessing techniques and hyperparameters are described in Section 6;

in Section 7, the research gaps in the current literature are shown; finally, in Section 8, our conclusions are explained and future prospects are explored.

## 2. Systematic Literature Review (SLR) Protocol

This paper presents a systematic literature review (SLR) conducted between 2018 and 2022 focusing on detection of DDoS attacks using DL methods. The SLR method used in this study adheres to the recommendations made in [21], providing a comprehensive approach to understanding the literature on the subject. Unlike previous review papers, this study includes an analysis of the preparation techniques, advantages, and different types of attacks used in various datasets. The output of the SLR is a collection of research publications organized according to the taxonomy of the DL techniques utilized. By identifying research limitations in the body of literature, this study offers exciting new options for future research. Overall, this paper presents a rigorous and novel approach to a systematic review of DDoS attack detection techniques. The research protocol summary is shown in Figure 1, and is described in detail below.



**Figure 1.** Survey protocol overview.

### 2.1. Research Questions

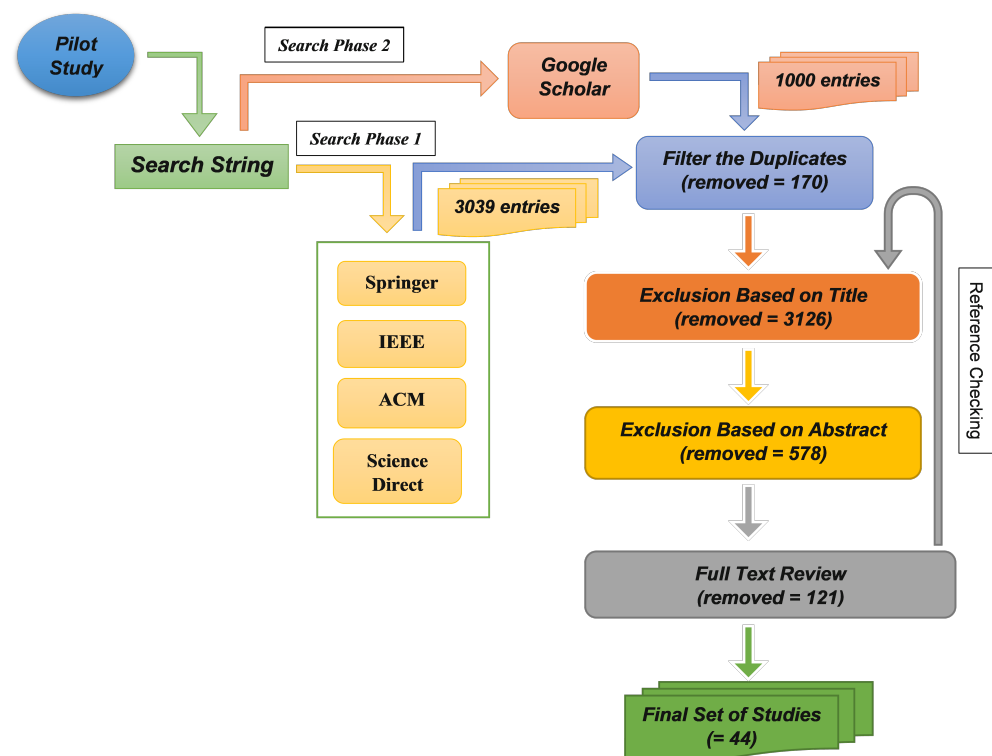
The main objective of a systematic review is to address research questions by analyzing data extracted from previous studies. The research questions addressed in the present work include:

- **RQ1:** What are the most recent DL techniques for detecting DDoS attacks, and how can they be classified?
- **RQ2:** What are the current methodologies, advantages, and disadvantages of DL methods for detecting DDoS attacks?
- **RQ3:** What types of attacks are included in the datasets used in current research, and what benchmarked DDoS datasets are available?
- **RQ4:** What preprocessing techniques, hyperparameter settings, experimental configurations, and performance metrics are used by current DL algorithms for DDoS attack detection?
- **RQ5:** What are the research gaps in the published literature?

## 2.2. Search Strategy

An effective search strategy is essential for any systematic survey. In this study, a carefully selected set of databases was used to mine the relevant literature. Two search phases were conducted between 2018 and 2022. The first phase searched four databases: ACM, IEEE Explore, Springer, and Science Direct. The second phase added Google Scholar in order to ensure that all relevant material was included. To refine the search string, pilot research was conducted. From the search results, ten highly referenced and relevant articles were selected.

One such search term that was used in several digital libraries with little alteration was (DDoS attack detection using DL approaches OR DDoS attack detection using ML approaches OR Detection of DDoS attacks using DL OR Detection of DDoS attacks using ML). Using “filtering choices”, we were able to improve the outcomes from the selected digital libraries. Figure 2 shows the flow of the various phases of the survey protocol.



**Figure 2.** Systematic Literature Review method.

## 2.3. Study Selection Criteria

The main objective of the research selection process was to identify relevant literature addressing the defined research questions while excluding any irrelevant material. To this end, inclusion and exclusion criteria were applied; these encompassed research papers that built upon earlier relevant studies. In stage 1, we took the first 1000 items from the second search phase and combined them with the 3039 entries from the first search phase to create 4039 entries. In stage 2, 170 duplicate entries were eliminated. After stage 2, articles were removed in accordance with their titles (3126), abstracts (581), and complete texts (118), respectively. In the end, (44) research articles were chosen. Studies that were unrelated to established research topics were eliminated using the inclusion and exclusion criteria. The following definitions describe the inclusion/exclusion criteria:

**Inclusion criteria:**

- All publications that present a novel method for ML/DL-based DDoS attack detection
- Research that exclusively pays attention to ML/DL techniques
- Studies involving related topics while differing in crucial elements are incorporated as separate primary studies
- Research that responds to the study's questions
- Research building on earlier relevant research
- Articles released between 2018 and 2022.

**Exclusion criteria:**

- Articles not written in English
- Research unrelated to this study's topic
- Review papers, editorials, discussions, data articles, brief communications, software publications, encyclopedias, posters, abstracts, tutorials, works-in-progress, keynotes, and invited talks
- Articles that do not provide a sufficient amount of information
- Duplications of other research.

*2.4. Reference Checking*

The references from the (32) studies that were retained after scanning the whole manuscripts were evaluated to make sure that no significant work had been missed. The (76) papers that contributed to their conclusions were then evaluated more thoroughly based on the title, abstract, and full article using the same inclusion and exclusion criteria as previously. Articles based on titles (11), abstracts (51), and entire articles (12) were removed in the next rounds. Of the papers found via reference checking, (74) entries were removed, resulting in only two additional papers.

*2.5. Data Extraction*

After examining the entire manuscripts, pertinent information was collected based on our research questions. The collected information from each study was used to complete a templated form. The title, technique, datasets used, number of features, recognition of attack and genuine classes, preprocessing techniques, testing configuration for enhancement of the model, evaluation methods, advantages and disadvantages of the model, and a summary were all used to critically evaluate the final set of articles in order to condense the answers to our research questions. The fields used for data extraction are detailed in Table 2.

**Table 2.** Fields used for data extraction.

Field	Description
Title	Provides the study paper's title
The approach used	Lists the various ML/DL-related methods that were employed in the article.
Datasets	Lists the various datasets that were utilized in the study for the analysis.
The number of features	List of the datasets' chosen features.
Identification of attack and legitimate classifications	Names of the attacks used in the article
Preprocessing strategy	Explains how the data were preprocessed before the model was trained.
Model setup and performance optimization for experiments	Describes how experiments were carried out and lists the model parameter values leading to the best performance.
Performance metrics	Provides the findings when using different measures for comparison of one model to another.
Strengths	Describes the model's positive attributes.
Weaknesses	Lists the model's shortcomings.
Summary	A succinct description of the fields mentioned above



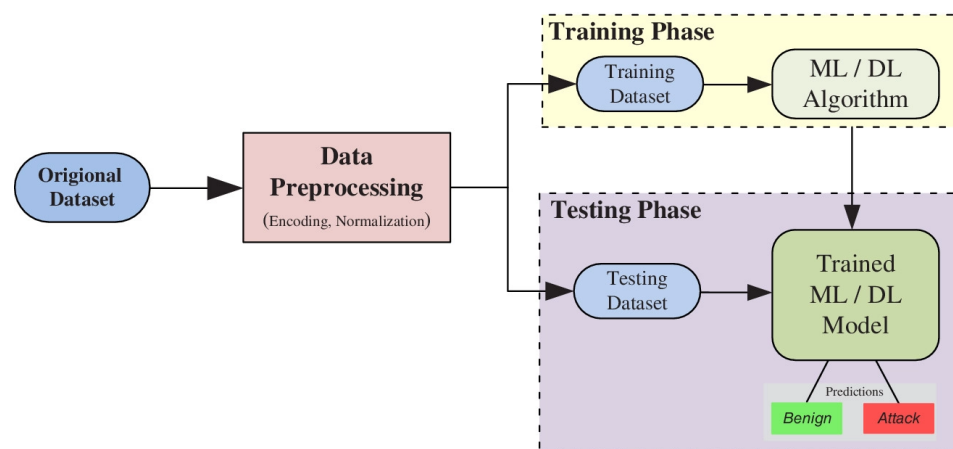
### 3. Most Up-to-Date ML/DL Techniques for Detecting DDoS Attacks

The field of ML is a subfield of artificial intelligence (AI) that encompasses all techniques and algorithms that allow computers to automatically learn from big datasets by applying mathematical models. Decision Tree (DT), K-Nearest Neighbor (KNN), Artificial Neural Network (ANN), Support Vector Machine (SVM), K-Means Clustering, Fast Learning Networks, Ensemble Methods, and others are the most popular ML methods used for DDoS detection in SDN (sometimes called Shallow Learning). The brief explanations of each category are as follows:

- **Decision Tree (DT):** a fundamental supervised ML method that leverages a set of rules to classify and predict data using regression. The model is structured as a tree with nodes, branches, and leaves, where each node represents a feature or characteristic. Each leaf on the branch denotes a possible outcome or a class label, and the branch itself signifies a decision or a rule. The DT algorithm automatically selects the optimal attributes for tree construction and performs pruning to eliminate unnecessary branches and prevent overfitting [22].
- **K-Nearest Neighbor (KNN):** the K-Nearest Neighbor (KNN) algorithm is a simple supervised ML method that uses the concept of “feature similarity” to classify a given data sample. By determining a sample’s identity based on its neighbors and how far away it is from them, KNN can effectively determine the class of a data sample. The value of the KNN algorithm’s  $k$  parameter can have an impact on its performance, and selecting a  $k$  value that is too small or too large can lead to overfitting or incorrect categorization of the sample case. To improve the detection rate of attacks in the minority class, researchers using the most recent benchmark dataset, CSE-CIC-IDS2018, have applied the Synthetic Minority Oversampling Technique (SMOTE) to overcome the dataset imbalance issue when evaluating the performance of various ML algorithms, including KNN [23].
- **Support Vector Machine (SVM):** Support Vector Machine (SVM) is a supervised ML method that uses the max-margin separation hyperplane in  $n$ -dimensional feature space as its foundation. It can be used to solve both linear and nonlinear issues, employing kernel functions to address the latter. The goal of SVM is to first translate a low-dimensional input vector into a high-dimensional feature space using the kernel function, then to use the support vectors create an optimal maximum marginal hyperplane that serves as a decision boundary. By correctly identifying the benign and harmful classes, the SVM method can be used to identify DDoS attacks with greater efficiency and accuracy [24].
- **K-Mean Clustering:** the goal behind clustering is to group together sets of data that are very similar in order to divide the data into meaningful clusters or groups. One popular iterative ML technique that learns without supervision is K-Mean clustering. Here,  $K$  denotes a dataset’s total number of centroids (cluster centers). Distance is typically measured when allocating specific data points to a cluster. The main goal is to decrease the total distance between each data point and its associated centroid within a cluster [25].
- **Artificial Neural Network (ANN):** the functioning of the human nervous system serves as the inspiration for the supervised ML algorithm known as ANN. It consists of neurons (nodes), which are processing units, and the connections that link them together. The organization of these nodes includes an input layer, several hidden levels, and an output layer. A backpropagation algorithm is employed by ANNs as a learning method. The capacity of the ANN approach to perform nonlinear modeling by learning from larger datasets is its key benefit. However, the fundamental difficulty with training ANN models is the lengthy procedure required, as its complexity can hinder learning and result in less than ideal results [26].
- **Ensemble methods:** the main idea behind ensemble techniques is to learn in an ensemble fashion in order to benefit from the use of multiple classifiers. Each classifier has its own advantages and disadvantages; for example, they may be good at spotting

a certain kind of attack and bad at spotting other kinds. By training several classifiers, ensemble techniques can combine several weak classifiers to create a single stronger classifier, which is typically selected using a voting mechanism [27].

DL is a type of ML used in AI that has the ability to learn from both supervised and unstructured data [28]. DL models are known as Deep Neural Networks or Deep Neural Learning, as the technology makes use of multi-layer networks. Neurons connect the levels and stand in for the mathematical calculations behind learning processes [29]. As seen in Figure 3, the three main processes that make up most ML/DL methods are: (i) the data preparation phase, (ii) the training phase, and (iii) the testing phase. The dataset is initially preprocessed for each of the suggested solutions in order to convert it into a form that the algorithm can use. Typically, this phase involves normalization and coding. The dataset may need to be cleaned, which occurs during this step if necessary. Duplicate entries and entries with missing data are removed. The training dataset and testing dataset are created by randomly dividing the preprocessed data into two halves. Typically, nearly all (80%) of the initial dataset size is typically made up of the training dataset, with the remaining amount (20%) constituting the testing dataset. In the subsequent training phase, the ML or DL algorithm is taught using the training dataset. The proportion of the dataset that is used and the complexity of the model being trained affect how long it takes the algorithm to learn. Due to their intricate and sophisticated structure, DL models often require a longer training period than ML models. After training, models are tested using the testing dataset, with performance being assessed based on the predictions made by the model. In the case of DDoS detection models, this takes the form of network traffic instances being classified as either benign (normal) or attack instances.



**Figure 3.** Methodology for generalized machine learning/deep learning-based DDoS detection systems.

DL techniques can be divided into five groups: hybrid learning, semi-supervised learning, supervised instance learning, and supervised sequence learning. A succinct summary of each category is provided below:

**Supervised instance learning:** Supervised Instance 1 = Learning uses the flow of instances [18]. For training purposes, it makes use of labeled instances. The most popular techniques in this area are:

- **Deep Neural n = Networks (DNN):** a fundamental DL structure that allows the model to learn at multiple levels. It comprises several hidden layers, along with input and output layers. DNNs are used to simulate complex nonlinear functions. The addition of more hidden layers improves the model's abstraction level, expanding its potential. For classification purposes, the output layer consists of one fully connected layer and a softmax classifier. The Rectified Linear Unit (ReLU) function is commonly used as the activation function for the hidden layer [30,31].



- **Convolutional Neural Network (CNN):** a CNN is a DL structure that is well suited for image and signal data. All CNNs have an input layer, a stack of convolutional and pooling layers for feature extraction, a fully connected layer, and a softmax classifier in the classification layer. CNNs have achieved great progress in the realm of computer vision, and can perform supervised feature extraction and classification functions for DDoS detection tasks [32].

**Supervised sequence learning:** in supervised sequence learning, a series of flows are used; when learning from a set of inputs, this form of model keeps track of the prior input states in its memory. The most popular models of this kind include:

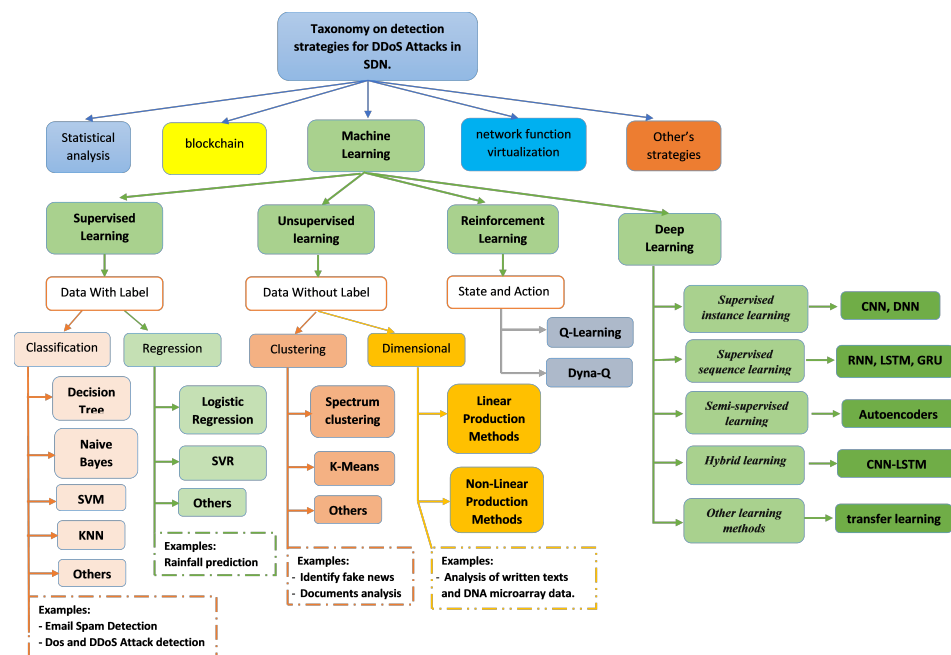
- **Recurrent Neural Networks (RNN):** RNNs were developed to improve upon the capabilities of traditional feed-forward neural networks and model sequence data. Input, hidden, and output units make up an RNN, with the hidden units acting as the memory elements. In order reach a decision, each RNN unit considers both the current input and the results of prior inputs. RNNs are commonly used in a wide range of fields, such as semantic comprehension, handwriting prediction, voice processing, and human activity identification [33]. RNNs can be used for feature extraction and supervised categorization in DDoS detection. However, RNNs can only manage sequences up to a certain length before running into short-term memory problems [34].
- **Long Short-Term Memory (LSTM):** LSTM is a DL structure that has successfully addressed the challenges of RNNs. An LSTM network is composed of different memory cells or blocks. The following cell receives both the hidden state and the cell state through three mechanisms known as gates, specifically, forget, input, and output gates [35]. The memory blocks may choose which data to recall or ignore. A forget gate eliminates information from the current input that the LSTM no longer requires [36]. The output gate is responsible for extracting pertinent data from the current input and processing it as an output. Finally, the input gate is responsible for adding inputs to the cell state [37].

**Semi-supervised learning:** semi-supervised learning involves using unlabeled data in the pre-training stage of the algorithm. This approach trains a model using both labeled and unlabeled data. In this case, the features are extracted using an autoencoder and classification is performed using various deep or shallow machine learning models. AutoEncoding (AE) is a common deep-learning method that belongs to the unsupervised neural network family. By learning the best features, AE aims to match the output to the input as closely as possible. Although the dimensions of the hidden layers are often smaller than those of the input layer, an autoencoder has input and output layers of the same dimension. Symmetric encoder–decoder operation is a key aspect of AE. Stacked AE, Sparse AE, and Variational AE are three different versions of AE [13].

**Hybrid learning:** a combination of any two other methods, such as shallow machine learning, supervised deep learning, or unsupervised deep learning, is known as a hybrid learning method. Researchers have commonly employed CNN–LSTM [38–40]), LSTM–Bayes [41], RNN–AE [42], and other hybrid models.

**Other learning methods:** this group includes transfer learning, in which a pre-trained model from a repository is used in a transfer learning technique [13]. In these cases, researchers use deep learning techniques train models on one attack domain before applying them to another.

This section has provided a thorough summary of the most popular ML and DL algorithms for DDoS detection systems. Figure 4 illustrates the taxonomy of current ML/DL-based DDoS detection approaches.



**Figure 4.** Taxonomy of machine learning-based and deep learning-based DDoS detection systems.

#### 4. Methodologies, Strengths, and Weaknesses

The specifics of the most popular ML and DL algorithms used to create an effective DDoS detection model are described in this section, along with basic techniques for AI-based DDoS detection. Both supervised and unsupervised methods are used in ML and DL. In supervised algorithms, data need to be labeled prior to use. Unsupervised algorithms, on the other hand, use unlabeled data to extract important characteristics and details. The methodologies, advantages, and disadvantages of studies using these approaches are summarized in Table 3.

**Table 3.** Methodologies, strengths, and weaknesses of different studies using ML/DL for DDoS detection.

Reference	Methodology	Strengths	Weaknesses
Shen et al. [43]	Used BAT algorithm with Ensemble Method for Optimization.	When used in an ensemble setting, several ELMs showed good performance.	Used outdated datasets, including Kyoto, NSL-KDD, and KDDCup99. Additionally, the model's detection accuracy for the U2R attack class was lower.
Shone et al. [44]	Utilized RF with Non-Symmetric Deep Auto Encoder	Presented a non-symmetric deep AE and RF classifier-based DDoS detection, which lowers the model complexity	The model was examined with outdated datasets, and its performance on the minority R2L and U2R classes was on the low side.
Ali et al. [45]	Employed a Particle Swarm Algorithm and a Fast Learning Network	The suggested model outperformed other FLN-based models using several additional optimization strategies	Outdated dataset used. Additionally, fewer training data led to a lower detection rate.
Yan et al. [46]	Utilized SVM and a Sparse Auto Encoder	SVM was effectively used as a classifier with SSAE for feature extraction to identify DDoS attacks	The model was evaluated using an outdated dataset; while the model's detection rates for U2R and R2L attacks were respectable, they were lower than for the other attack classes in the dataset.
Naseer et al. [47]	Comparison of several ML/DL-based IDS models	Used a GPU-integrated testbed to compare ML/DL-based DDoS detection methods	The flaws were evaluated using an earlier dataset called NSL-KDD.
Al-QatfM et al. [48]	A self-taught learning model was employed using an autoencoder and SVM	The effective notion of self-taught learning based on Sparse AE and SVM was proposed as a DDoS detection model	An older dataset called NSL-KDD was used. Additionally, no outcomes were provided regarding the model's effectiveness against minority attack classes

Table 3. Cont.

Reference	Methodology	Strengths	Weaknesses
Marir et al. [49]	Used SVM and Deep Belief Network	DBN was utilized to extract features, which were then passed on to an ensemble SVM before being predicted through a voting method	Model complexity and training require a longer time with more deep layers.
Yao et al. [50]	K-Means Clustering and Random Forest-based multilevel model	The clustering idea was applied in combination with RF to offer a multilayer intrusion detection model; the model performed better than average in identifying assaults	Used the outdated KDDCup99 dataset to test the model.
Gao et al. [51]	Employed a voting system and ensemble ML techniques	Used an adaptive ensemble model combining many basic classifiers such as DT, RF, KNN, and DNN, and used an adaptive voting mechanism to select the best classifier	An earlier dataset called NSL-KDD was used to test the model; the results on weaker attack classes were insufficient.
Karatas et al. [52]	Comparison of the performance of several ML algorithms by first lowering the dataset imbalance ratio using SMOTE	SMOTE progressively increased the detection rate for the minority attack classes	Longer execution times
Sabeel et al. [53]	Two ML models (DNN and LSTM) were proposed for the detection of DDoS assaults	The performance of these models significantly improved, with DNN and LSTM accuracy levels of 98.72% and 96.15%, respectively. AUC values for the DNN and LSTM were 0.987 and 0.989, respectively	The authors did not use real-time detection, and only binary class classification was carried out.
Vir. et al. [54]	Identifying DDoS assaults in the cloud utilizing DT, KNN, NB, and DNN algorithms	The DNN classifier outperformed DT, KNN, and NB in terms of accuracy and precision, achieving 96% on the cloud dataset	Employed an outdated dataset; there was no information provided regarding the LAN or the cloud dataset.
Asad et al. [55]	Developed a DNN architecture	The proposed DeepDetect model outscored other strategies and produced an F1-score of 0.99. Additionally, the AUC value was very near to 1, demonstrating great accuracy	This strategy was only tested against DDoS assaults at the application layer.
Mural. et al. [56]	Identified sluggish DoS assaults on HTTP and suggested a data flow-based DNN algorithm	The model could categorize assaults with a 99.61% overall accuracy	Only slow HTTP DoS attacks were assessed using this method; the CICIDS2017 dataset was used.
Sbai et al. [57]	Identified DF or UDPFL attacks in MANETs using the CICDDoS2019 dataset and suggested a DL model DNN using two hidden layers and six epochs	Recall = 1, Precision = 0.99, F1-score = 0.99, and Accuracy = 0.99	This study focused solely on DF or DPFL attacks, and used the CICDDoS2019 dataset.
Amaizu et al. [58]	Combined two DNN models with distinct designs and a PCC feature extraction approach for DDoS attack detection in 5G and B5G scenarios	99.66% accuracy rate and a 0.011 loss; all models except a CNN ensemble were outperformed by the suggested framework	Because of its complicated structure, the suggested model's performance in a real-time environments may suffer as a result of longer detection times.
Cil et al. [59]	Suggested DL model mechanisms for feature extraction and classification built into the model framework	Nearly 100% accuracy for DatasetA. Additionally, when using DatasetB the model correctly categorized DDoS assaults with a 95% accuracy rate	For multiclass classification, the suggested model performed less accurately.
Hasan et al. [60]	Suggested a Deep CNN model	The proposed technique performed better than three other ML methods	The dataset utilized included a limited number of cases and excluded several forms of traffic.
Amma et al. [61]	Combined FCNN with Vector VCNN	The suggested technique outperformed basic classifiers and the most sophisticated attack detection system in terms of high accuracy, reduced false alarms, and enhanced detection rate	Utilized an outdated dataset and omitted their trials for identifying unidentified assaults
Chen et al. [62]	Suggested a Multi-channel CNN architecture for DDoS assaults	The MCCNN performed better on the constrained dataset	The outcomes of multi-class and single-class classification models were not significantly different; the complexity of the multi-class model makes it unsuitable for validation in real-time circumstances.

Table 3. Cont.

Reference	Methodologies	Strengths	Weaknesses
Shaaban et al. [63]	Used two datasets to test the suggested model against classification algorithms including DT, SVM, KNN, and NN	Suggested model scored well and provided 99% accuracy on both datasets	In this method, the data were transformed into a matrix by extending one column. As a result, this may influence how the model learns.
Haider et al. [64]	Suggested a deep CNN framework for the detection of DDoS assaults in SDN	Ensemble CNN technique performed better than the currently used rival approaches, attaining a collective 99.45% accuracy rate	This strategy requires longer training and testing periods. As a result, the mitigating mechanism may be impacted, meaning that assaults can cause more damage.
Wang et al. [65]	Suggested an information entropy and DL technique to identify DDoS assaults in an SDN context	The CNN outperformed alternatives in terms of precision, accuracy, F1-score, and recall, with an accuracy rate of 98.98%	The model Required a longer to perform time detection
Kim et al. [66]	Created a CNN-based model to identify DoS attacks	The CNN model is better able to recognize unique DoS assaults with similar features. Additionally, CNN kernel size had no discernible effect on either binary or multiclass categorization	Longer time detection
Doriguzzi et al. [67]	DDoS assaults were detected using the LUCID approach	LUCID's performance was better in terms of accuracy	The padding might interfere with CNN's ability to learn patterns. Additionally, there were compromises between accuracy and the amount of memory required. The preprocessing time was not been computed for real-time situations
de Assis et al. [68]	Suggested an SDN defense mechanism	Overall findings demonstrated effectiveness of the CNN in identifying DDoS assaults for each of the test cases	When using the CICDDoS 2019 dataset, the model exhibited worse accuracy.
Hussain et al. [69]	Suggested a technique for converting three-channel picture formats from non-image network data	The suggested strategy obtained 99.92% accuracy in binary-class classification	The preparation time for transforming non-images to images was not calculated. Additionally, the processing used to convert the original $60 \times 60 \times 3$ dimensions into the $224 \times 224 \times 3$ dimensions used as the input for the ResNet model was been specified
Li C et al. [70]	Suggested a deep learning approach	DDoS assault detection was 98% accurate	Long time required for detection
Priy. et al. [71]	A DL-based approach was developed	For all the test cases, it was noted that the LSTM model displayed 98.88% accuracy. The module was able to prevent an attacked packet from reaching the cloud server via the SDN OF switch	Only DDoS assaults at the network or transport level were examined; real-time feasibility analyses of the proposed model were not conducted.
Liang et al. [72]	A four-layered architectural model with two LSTM layers was suggested	Experimental findings demonstrated that the LSTM-based technique performed better than other approaches	When the flow consisted of short packets, N was padded with fictitious packets. These padding settings have the potential to degrade performance, and have an impact on how the suggested model learns.
Shu. et al. [73]	Two methods, a hybrid-based IDS and a DL model based on LSTM, were proposed for DoS/DDoS assaults detection	The LSTM-based model reached an accuracy of 99.19%	A long time was required for detection
Assis et al. [37]	Proposed a defense mechanism against DDoS and intrusion assaults in an SDN environment	The average results for the accuracy, recall, precision, and f1-score were 99.94% and 97.09%, respectively	The model used a complex framework
Catak et al. [7]	Cpmbo a deep ANN and AE model	The best F1 values with the ReLu activation function were obtained (0.8985). For the activation functions of softplus, softsign, relu, and tanh, the overall accuracy, and precision are close to 99%	The activation functions are the sole thing this article focuses on
Ali et al. [74]	Proposed a deep AE for feature learning and the MKL framework for detection model learning and classification	The proposed approach was found to be more accurate than alternative approaches	Used an outdated dataset

Table 3. Cont.

Reference	Methodologies	Strengths	Weaknesses
Yang et al. [75]	A five-layered AE model was developed for efficient unsupervised DDoS detection	AE-D3F achieved nearly 100% DR with less than 0.5% FPR, although the RE threshold value must be specified. This method compensates for the lack of labeled attack data by training the model using only regular traffic	Used an outdated dataset
Kasim et al. [76]	Suggested a hybrid AE-SVM technique	In terms of quick anomaly identification and low false-positive rate, the AE-SVM method fared better than other approaches	Compared to the other two datasets, the suggested model's accuracy on the NSL-KDD dataset was lower
Bhardwaj et al. [77]	Combined a stacked sparse AE to learn features with a DNN for categorizing network data	The results indicated an accuracy of 98.92%. The suggested approach worked well to address issues with feature learning and overfitting, as the AE was trained with random training data samples to perform feature learning and overfitting was avoided by employing the sparsity parameter	Performed an offline study rather than evaluating the most recent datasets. Additionally, the suggested model could not compute the detection time.
Moha. et al. [41]	Combined the LSTM and Bayes techniques	The results revealed that, the performance indicators decreased only slightly with the new data, and the outcomes were positive	Assaults that are unsuited for real-time applications may take the LSTM-BA longer to identify. Comparing the suggested model to the current DeepDefense approach, the accuracy was only improved by 0.16%. IP addresses were transformed into actual vectors using feature hashing, and the preprocessing time was not computed using the BOW.
RoopakM et al. [39]	Employed multi-objective optimization, namely, the NSGA approach	F1-score value of 99.36% and high accuracy of 99.03%. Additionally, The outcomes demonstrated that the suggested model outperformed earlier studies. When compared to previous DL approaches, the training time was cut by an astounding eleven times	The majority of the cutting-edge methods used in this article did not use the CICIDS2017 dataset; therefore, the analogy appears inappropriate.
Elsa et al. [42]	Combined AE and RNN to produce DDoS-Net for identifying DDoS assaults in SDNs	The results indicated that DDoS-Net performed better than six traditional ML techniques (DT, NB, RF, SVM, Booster, and LR) in terms of accuracy, recall, precision, and F1-score. The proposed method obtained 99% accuracy and an AUC of 98.8	The dataset used offline analysis, and multi-class classification was not carried out
Nugraha et al. [40]	A DL-based strategy was proposed to identify sluggish DDoS assaults in SDNs using a CNN-LSTM model	The suggested model performed better than other approaches, obtaining more than 99.5 percent across all performance criteria	The dataset used offline analysis
He et al. [78]	A strategy based on DTL was proposed to identify DDoS attacks	A 20.8% improvement was achieved on detection of the 8LANN network in the target domain. The DTN technique with fine-tuning avoided the decline in detection performance brought on by using a small sample of DDoS attacks	For model evaluation, only one attack is used in both the source domain and the target domain.
Chen et al. [79]	Suggested a minimax gradient-based deep reinforcement learning technique	When compared to cutting-edge algorithms, the suggested policy-based GPDS algorithm outperformed them in terms of anti-jamming performance	

## 5. Available Benchmarked DDoS Datasets and Classes of Attacks in Datasets

The datasets and attack class types utilized by the studies that were examined for DDoS attack detection are listed in Table 4. eight datasets (KDD Cup99, Kyoto 2006+, NSL-KDD, UNSW-NB15, CIC-IDS2017, CSE-CIC-IDS2018, SCX2012, and CICDDoS2019) were utilized across the majority of studies. Following is a description of these datasets.

**KDD Cup99:** One of the most well-known and often used datasets for IDS is KDD Cup99. It contains about five million and two million recordings for training and testing, respectively. Each recording has 41 distinct characteristics or properties, and is classified as an attack or as normal data. Four categories of attacks are established for the recordings, namely, Denial of Service (DoS), Probe, Remote to Local (R2L), and User to Root (U2R) [80].



**Table 4.** The most current research on ML/DL for DDoS attack detection, showing techniques, datasets, and types of attacks.

Ref.	Year	Approach	Dataset	Classes-of-Attacks
Shen et al. [43]	2018	BAT, Ensembl	KC, NK, NY	DoS, Probe, R2L, U2R
Shone et al. [44]	2018	RF, DAE	KC, NK	DoS, Probe, R2L, U2R
Ali et al. [45]	2018	FLN	KC	DoS, Probe, R2L, U2R
Yan et al. [46]	2018	SVM, SAE	NK	DoS, Probe, R2L, U2R
Naseer et al. [47]	2018	GPU	NK	DoS, Probe, R2L, U2R
Al-QatfM et al. [48]	2018	SAE, SVM	NK	DoS, Probe, R2L, U2R
Marir et al. [49]	2018	SVM, DBN	NK, UN, C7	DoS, Probe, R2L, DDoS, Web
Yao et al. [50]	2018	KMC, RF	KC	DoS, Probe, R2L, U2R
Gao et al. [51]	2019	ensemble	NK	DoS, Probe, R2L, U2R
Karatas et al. [52]	2020	KNN, RF, DT	C8	HeartBleed, DoS, Botnet, DDoS
Sabeel et al. [53]	2019	DNN, LSTM	C7, A9	Benign, DoS GoldenEye, DoS, DDoS.
Virupakshar et al. [54]	2020	DT, KNN, NB, DNN	KC	DoS, Probe, R2L, U2R
Asad et al. [55]	2020	DNN	C7	Botnet, DoS, DDoS, Web
Muraleedharan et al. [56]	2020	DNN	C7	Botnet, DoS, DDoS, Web
Sbai et al. [57]	2020	DNN	C9	Data flooding or UDP flooding attack
Amaizu et al. [58]	2021	DNN	C9	UDP, SYN, DNS, Benign
Cil et al. [59]	2021	DNN	C9	DDoS attacks
Hasan et al. [60]	2018	CNN	OBS	DDoS
Amma et al. [61]	2019	CNN	NK	DoS
Chen et al. [62]	2019	CNN	C7, K9	DoS, Probe, R2L, U2R, DDoS
Shaaban et al. [63]	2019	CNN	NK	DoS, Probe, R2L, U2R
Haider et al. [64]	2020	RNN, LSTM	C7	Botnet, DoS, DDoS, Web
Wang et al. [65]	2020	Entropy, CNN	C7	Botnet, DoS, DDoS, Web
Kim et al. [66]	2020	CNN	KC, C8	Botnet, DoS, DDoS, Web
Doriguzzi et al. [67]	2020	CNN	I2, C7, C8	Botnet, DoS, DDoS, Web
de Assis et al. [68]	2020	CNN	C9	DDoS attacks
Hussain et al. [69]	2020	CNN	C9	Syn, TFTP, DNS, LDAP, UDP
Li C et al. [70]	2018	LSTM, CNN, GRU	I2	DDoS attack
Priyadarshini et al. [71]	2019	LSTM	I2	DDoS attack
Liang et al. [72]	2019	LSTM	C7	Botnet, DoS, DDoS
ShurmanM et al. [73]	2020	LSTM	C9	MSSQL, SSDP, CharGen, LDAP, NTP
Assis et al. [37]	2021	GRU	C8, C9	DDoS attacks
Catak et al. [7]	2019	AE, ANN	U5, KC	DoS, Probe, R2L, U2R
Ali et al. [74]	2019	AE, MKL	I2, U5	Fuzzers, Backdoors, Analysis, DoS
Yang et al. [75]	2020	AE	U7	(HTTP, Hulk and Slowloris) attack
Kasim et al. [76]	2020	AE, SVM	C7, NSL, KDD	(HTTP, Hulk and Slowloris) attack
Bhardwaj et al. [77]	2020	AE, DNN	C7, NSL, KDD	(HTTP, Hulk and Slowloris) attack
Premkumar et al. [81]	2020	RBF	Generated dataset	DDoS
RoopakM et al. [38]	2019	MLP, CNN, LSTM	C7	DoS, Probe, R2L, U2R
Mohammad et al. [41]	2019	LST, BN	I2	DDoS
RoopakM et al. [39]	2020	CNN, LSTM	C7	DDoS
ElsayedMS et al. [42]	2020	RNN, AE	C9	DDoS
Nugraha et al. [40]	2020	CNN, LSTM	generated	DDoS
He et al. [78]	2020	LANN	generated	DDoS
Chen et al. [79]	2022	reinforcement	generated	DDoS

**Kyoto 2006+:** this dataset was produced using network traffic statistics that Kyoto University collected through the use of honeypots, darknet sensors, email servers, web crawlers, and other network security mechanisms. The most recent dataset covers traffic statistics for the years 2006 to 2015. Each entry consists of 24 analytical attributes, among which fourteen are taken directly from the KDD Cup99 dataset and the remaining ten are extra features.

**NSL-KDD:** this dataset consists of the KDD Cup99 dataset improved and amended by eliminating a number of its fundamental problems. This dataset has 41 features, and the attacks are split into four groups, as stated in KDD Cup99.131 [82].

**UNSW-NB15:** the Australian Center for Cyber-Security produced this dataset. Using Bro-IDS, Argus tools, and a number of newly created methods, almost two million records were retrieved with a total of 49 characteristics. Worms, Shellcode, Reconnaissance, Port Scans, Generic, Backdoor, DoS, Exploits, and Fuzzers are among the attack types included in this dataset.



**CIC-IDS2017:** the Canadian Institute of Cyber-Security (CIC) produced this dataset in 2017. New actual assaults and typical flows are both included. CICFlowMeter uses data from records, source and destination IP addresses, protocols, and assaults to assess the network traffic. CICIDS2017 includes typical attack cases such as Brute Force Attacks, HeartBleed Attacks, Botnets, Distributed DoS (DDoS), Denial of Service (DoS), Web Attacks, and Infiltration Attacks [83].

**CSE-CIC-IDS2018:** in 2018, the Communications Security Establishment (CSE) and the CIC collaborated to generate this dataset by creating user profiles that include an abstract depiction of many occurrences. All of these profiles were then integrated with a special set of characteristics to create the dataset. Brute Force, Heartbleed, Botnet, DoS, DDoS, Web assaults, and network penetration from within are just a few of the seven various attack scenarios that are covered by this dataset [84].

**ISCX2012:** this dataset, which contains full-packet network data, was developed in 2012 by Ali Shiravi et al. [20] It covers seven days from June 11 to June 17, 2010, with network activity including both legitimate and malicious traffic. A few examples of malicious traffic include Distributed Denial of Service, HTTP Denial of Service, and Brute Force SSH. This dataset was produced in a simulated network context, and contains labeled and unbalanced data. Two broad profiles, one that describes attack activity and the other that describes typical user scenarios, are utilized in the ISCX dataset [85].

**CICDDoS2019:** Sharafaldin et al. [86] created the CICDDoS2019 dataset (2019). More than 80 traffic characteristics were taken from the original information by using the CICFlowMeter-V3 program to extract the features. The CICDDoS2019 contains typical DDoS assaults that are safe and current. This dataset, which was created using actual traffic, contains a variety of DDoS assaults created utilizing TCP/UDP protocols [87].

## 6. Preprocessing Techniques, Hyperparameter Settings, Experimental Configurations, and Performance Metrics

Table 4 lists the preprocessing techniques, hyperparameter settings, test configurations, and performance metrics employed by the current ML/DL algorithms for DDoS attack detection. At the beginning, the data are preprocessed. Preprocessing the data is essential, as it changes raw data into a structure that improves the model's capacity for learning [88]. Table 5 in this research provides an overview of preprocessing techniques employed in the body of the literature.

**Table 5.** The most recent studies on DDoS attack detection using ML/DL.

Ref.	Preprocessing Strategies	Hyperparameter Values	Experimental Setups	Performance Metrics
Shen et al. [43]		ELM with a BAT		Accuracy = 99.3%, Sensitivity = 99%, Specificity = 99%, Precision = 99%, F1-score = 99%, FPR = 1%, FNR = 1%
Shone et al. [44]		NDAEs with RF	Built-in TensorFlow	98.81%, saving time and improving accuracy by up to 5%.
Naseer et al. [47]		CNN, AE, and RNN		Accuracy for DCNN and LSTM of 85% and 89%, respectively.
Al-QatfM et al. [48]		AE technique		Improved SVM Classification accuracy and time
Marir et al. [49]	Multi-layer ensemble SVM		Hadoop cluster	Enhanced IDS performance.
Yao et al. [50]	MSML			MSML was superior to other current intrusion detection algorithms in terms of accuracy, F1-score, and capacity.

Table 5. Cont.

Ref.	Preprocessing Strategies	Hyperparameter Values	Experimental Setups	Performance Metrics
Gao et al. [51]		Ensemble learning		Accuracy = 85.2%, Precision = 86.5%, Recall = 85.2%, F1-score = 84.9%
Karatas et al. [52]		Six different ML models		Accuracy between 4.01% and 30.59%
Sabeel et al. [53]	DNN/LSTM	Input layer = 25 pixels, dense layer = 60 neurons, dropout rate = 0.2, batch size = 0.0001, learning rate = 0.0001	TensorFlow, Keras 1.1.0	TPR = 0.998, Accuracy = 98.72%, Precision = 0.949, F1-score = 0.974, and AUC = 0.987
Virupakshar et al. [54]			Two computers with dual-core processors	Recall = 0.91, F1-score = 0.91, Support = 2140
Asad et al. [55]	Cost-sensitive learning and min-max scaling	Seven hidden layers, input layer = 66 neurons, output layer = 5 neurons, epochs = 300, learning rate = 0.001	Intel Xeon E5 v2	AUC = 1, F1-score = 0.99, Accuracy = 98%
Muraleedharan et al. [56]		Eighty neurons for input, five neurons in the output layer, four hidden layers, Adam optimizer	SciKit and Keras API	Precision: Benign = 0.99, Slowloris = 1.00, Slowhttptest = 0.99, Hulk = 1.00, GoldenEye = 1.00, Accuracy = 99.61%.
Sbai et al. [57]				Recall = 1, F1-score = 0, Accuracy = 0.99997, Precision = 0.99
Amaizu et al. [58]	Min-max	Two Dropout Layers, 0.001 Learning Rate, 50 Epochs, and ReLu	Four PCs, one fire-wall, two switches, and one server	Recall = 99.30%, Precision = 99.52%, F1-score = 99.99%, Accuracy = 99.66%.
Cil et al. [59]	Min-max	Three hidden layers, 50 units of neurons each, and sigmoid	Intel Core i7-7700	Dataset 1: F1-Score = 0.9998, Accuracy = 0.9997, Precision = 0.9999, Recall = 0.9998; Dataset2: F1-Score = 0.8721, Accuracy = 0.9457, Precision = 0.8049, Recall = 0.9515.
Hasan et al. [60]		Two convolutional layers, a max-pooling layer, a fully connected layer (250 neurons), and SoftMax		F1-score = 99%, FPR = 1%, FNR = 1%, Accuracy = 99%, Sensitivity = 99%, Specificity = 99%, Precision = 99%
Amma et al. [61]	Min-max	Max pooling size = 2, filter size = 3, two hidden layers, output layer with 11-9-7-6 nodes, and ReLu		Normal = 99.3% accurate; Back = 97.8% accurate; Neptune = 99.1% accurate; Smurf = 99.2% accurate; Teardrop = 83.3% accurate; Others = 87.1% accurate.
Chen et al. [62]		Used a method of progressive training to train an MC-CNN		Accuracy: C7 = 98.87%, KU (two classes) = 99.18%, KC (five classes) = 98.54%.
Shaaban et al. [63]	Padding (8 and 41) was transformed into $3 \times 3$ and $6 \times 7$ matrices, respectively	CNN model with softmax function, and ReLu function	Tenserflow and Keras	Dataset 1: Accuracy = 0.9933, Loss = 0.0067; Dataset 2: Accuracy = 0.9924, Loss = 0.0076 (NSL-KDD)
Haider et al. [64]	Z-score	Two dense FC layers, one layer to flatten, three 2D CLs, two max PLs, and three 2D CLs with ReLu	Intel Core i7-6700	F1-score = 99.61%, Accuracy = 99.45%, Precision = 99.57%, Recall = 99.64%, Testing = 0.061 min, Training = 39.52 min, CPU Usage = 6.025.
Wang et al. [65]	Image are created by turning each byte in a packet into a pixel	Two FC layers, two PL levels, and two CL layers; the limit for entropy values = 100 packets/s	Intel Core i5, 7300HQ processor, POX controller, one server, and six switches with Hping3 and the TensorFlow framework	Precision = 98.99%, Recall = 98.96%, F1-score = 98.97%, Accuracy = 98.98%, Training time = 72.81 s, AUC = 0.949

Table 5. Cont.

Ref.	Preprocessing Strategies	Hyperparameter Values	Experimental Setups	Performance Metrics
Kim et al. [66]		117 features and pictures, with 13 and 9 pixels and the kernel size set to 2 and 3, respectively	Python and Tensor-Flow	KC accuracy = 99%, CSE8 = 91.5%
Doriguzzi et al. [67]	Min-max	n = 100, t = 100, k = 64, h = 3, m = 98, batch size = 2048, LR = 0.01, Adam optimizer, and sigmoid	Two 16-core Intel Xeon Silver 4110 CPUs with Tensor-Flow and Python	Accuracy = 0.9888, FPR = 0.0179, Precision = 0.9827, Recall = 0.9952, F1-score = 0.9889
de Assis et al. [68]	Shannon Entropy	CNN model consisting of three layers: a Flatten layer, a 0.5 Dropout layer, and an FC layer with ten neurons, two Conv1D layers, and MaxPooling1D layers with sigmoid and 1000 epochs	Intel Core i7	SDN data on average Accuracy, Precision, Recall, and F-measure (95.4%, 93.3%, 2.4%, 92.8%) for CI-CDDoS 2019.
Hussain et al. [69]	Min-max	Ten CLs and eight PLs in a ResNet18 model, learning rate = 0.0001, momentum = 0.9, epochs = 10 and 50 with SGD optimizer		F1-measure = 86, Precision = 87%, Recall = 86%, Accuracy = 87.06% for multiclass and 99.99% for binary
Li C et al. [70]	BOW and a 3D matrix used to convert a 2D feature matrix	Input, forward recursive, reverse recursive, FC hidden, and output layers making up a DL model.	128 GB of RAM and two NVIDIA K80 GPUs with Keras and Ubuntu	Accuracy = 98%
Liang et al. [72]	Examined a sub-sequence of n-packet flows	Two LSTM layers, with a series of ten packets extracted from each flow		F1-score = 0.9991, Precision = 0.9995, Recall = 0.9997
ShurmanM et al. [73]	RF	Three LSTM layers with 128 neurons each and sigmoid function, three dropout layers, and a dense layer with tanh function		Accuracy = 99.19%
Assis et al. [37]	MD5	A dropout layer with a dropout rate of 0.5 and an FC layer with ten neurons and sigmoid function	Intel Core i7, Keras and Sklearn software	Average metrics for accuracy, precision, recall, and F-measure values on the C8 dataset: 97.1%, 99.4%, 94.7%, and 97%, respectively. Valid flow classification rate: 99.7%.
Catak et al. [7]	Normalization	Three hidden layers, an output layer with 28, 19, 9, 19, and 28 units, and sigmoid activation function. The input layer has five hidden layers with 28, 500, 800, 1000, 800, and 500 units	NVIDIA Quadro, Python, Keras, TensorFlow, and SciKit-learn libraries	F1-score, Accuracy, Precision, and Recall values: 0.8985, 0.9744, 0.8924, and 0.9053, respectively.
Ali et al. [74]	Discretized features are those that are not numerical	Nine MSDAs with different number of layers L = (1, 3, 5, 7, 9, 11)	NVIDIA Tesla V100 with MATLAB	Average accuracy on Datasets D1 through D16, = 93%; Accuracy on Dataset D2 = 97%.
Yang et al. [75]	Flow split into several sub-flows based on a threshold value of 10 milliseconds	One input layer, three hidden layers, and one output layer, with 27, 24, 16, 24, and 27 neurons in each layer, respectively		Exp1: DR = 98.32%, FPR = 0.38%; U7: DR = 94.10%, FPR = 1.88%; SYNT: DR = 100%; FPR = 100%; Exp2: DR = 94.14%, FPR = 1.91%.
Kasim et al. [76]	Min-max	25 hidden neurons, 82 hidden input neurons and 82 hidden output neurons, 0.3 learning rate, 0.2 momenta, with 25 input nodes and two output nodes in the SVM. Learning rate = 0.01 and iterations = 1000.	Intel Core (TM) i7-2760 QM and Python with Keras, Scapy, TensorFlow, and SciKit libraries, and the Rest API	Training Time = 2.03 s, Testing Time = 21 milliseconds, C7 Accuracy = 99.90%. For the created DDoS assaults, Accuracy = 99.1% and AUC = 0.9988. For the NSL-KDD test, Accuracy = 96.36%
Bhardwaj et al. [77]	Min-max	Two encoding layers with 70 and 50 neurons each, a coding layer with 25 neurons, two decoding levels with 25 neurons each, and ReLu activation in each layer. Utilized the Adadelta optimizer and sigmoid activation in the output layer	Intel(R) Core-i7 CPU	NSL-KDD: Accuracy = 98.43%, Precision = 99.22%, Recall = 97.12%, F1-score = 98.57%. C7: Accuracy = 98.92%, Precision = 97.45%, Recall = 98.97%, F1-score = 98.35%

Table 5. Cont.

Ref.	Preprocessing Strategies	Hyperparameter Values	Experimental Setups	Performance Metrics
RoopakM et al. [38]		A 1D CNN layer with ReLu function, LSTM layer with Adam optimizer, dropout layer with a rate of 0.5, FC layer, and dense layer with sigmoid function make up the CNN–LSTM model	Intel Core-i7 with Keras, TensorFlow, and MATLAB	Precision = 97.41%, Recall = 99.1%, Accuracy = 97.36%
Moh. et al. [41]	Feature hashing and BOW	Two hidden FC layers of 256 neurons each with ReLU activation function and one neuron with sigmoid activation function make up the LSTM module.	GPU NVIDIA GTX 1050	Recall = 97.6%, Accuracy = 98.15%, Precision = 98.42%, TNR = 98.4%, FPR = 1.6%, F1-Score = 98.05%
RoopakM et al. [39]	Min-max	Maxpooling, LSTM, and dropout layers with Relu activation function after a 1D CNN; learning rate = 0.001, batch size = 256, epochs = 100, and dropout rate = 0.2%.	NVIDIA Tesla V100 GPUs with TensorFlow and Keras software	Accuracy = 99.03%, Recall = 99.35%, Precision = 99.26%, F1-score = 99.36%, Training Time = 15,313.10 s
Elsay et al. [42]	Min-max	Four RNN hidden layers in the RNN-AE; the channel counts for the encoder phase are 64, 32, 16, and 8, with two softmax functions		Accuracy for identification of attacks was 0.99%, whereas for benign cases it was 1.00. On the F1 scale, attack cases scored 0.99 and benign cases scored 0.99%. AUC = 98.8
Premkumar et al. [81]			Constant Bit Rate (CBR) application, 200 nodes, 500 s of simulation duration, and 5% to 20% of the regular nodes as attacking nodes.	Attack rate between 5% and 15%, detection rate between 86% and 99%, and false alarm rate of 15%
Nugraha et al. [40]	Min-Max	Flatten, maxpool, and dropout layers. Following the LSTM layer, there is an FC dense layer with a ReLu function, a dropout layer, and the last dense layer with a sigmoid function. Epochs = 50, training = 0.0005, dropout rate = 0.3, kernel size = 5, and the CNN filter is set to 64	Python	Accuracy = 99.998%, Precision = 99.989%, Specificity = 99.997%, Recall = 100%, F1-score = 99.994%
He et al. [78]		Eight FC layers in 8LANN. Pooling layers and the ReLu function are applied after each layer, with the eighth layer being the exception. 500 batches, a cross-entropy loss function, an SGD optimizer, and a training rate of 0.001 were utilized in this experiment.	Ubuntu 16.04 with NVIDIA RTX 2080Ti	Accuracy = 87.8% and Transferability = 19.65.
Chen et al. [79]	Game with Post-Decision State (GPDS)	Addresses the MDP optimization problem by offering a unique policy-based multi-user reinforcement learning game technique	Tensorflow and Intel(R)	According to the experimental findings, the suggested GPDS outperforms SOTA algorithms in terms of anti-jamming performance.

Hyperparameters are crucial because they directly affect how ML training algorithms behave. Prior to training the model, certain hyperparameter values must be chosen, which calls for specialized expertise and experience. There are two approaches for hyperparameter tuning, namely, manual search and automated search techniques. In a manual search, values for the hyperparameters are chosen manually. The automated search technique is similar to grid search, however, the grid search approach is more expensive. Another approach, known as a random search, has been introduced to address the grid search issue. Examples of hyperparameters include the number of epochs, batch size, learning rate, training algorithm, amount of layers, amount of neurons in each layer, etc.

An experimental setup includes information about the program, dataset, physical hardware, and other aspects of the experimentation process. As the training and testing

timeframes rely on the hardware setup, it is particularly crucial. Due to the complexity of ML/DL algorithms, suitable hardware configurations are needed.

The performance indicators are the most popular measures defined in this section. For binary classification, the typical performance measurements are accuracy, recall, precision, F1-score, AUC, etc.

The confusion matrix is described as the overview of outcomes foreseen by the categorization model. It includes (True Positive TP), True Negative (TN), False Positive (FP), and False Negative (FN) [89].

The true positive rate (TPR) is determined following Equation (1). Additionally, it may be known as the recall or sensitivity [90], and ought to be as high as possible.

$$TPR = \frac{TP}{(TP + FN)} \quad (1)$$

Precision is determined following Equation (2) by checking how many of the positive classes that the model adequately predicted are really positive [91].

$$\text{Precision} = \frac{TP}{(TP + FP)} \quad (2)$$

Following Equation (3), accuracy is defined as the percentage of true predictions made by the model across all classes. The highest level is preferable. Its formula is as follows [91]:

$$\text{Accuracy} = \frac{TP + TN}{(Total)} \quad (3)$$

The FPR or False Positive Rate is shown in Equation (4) [90]; it measures the percentage of negative occurrences that the model incorrectly forecast as positive.

$$FPR = \frac{FP}{(TN + FP)} \quad (4)$$

The percentage of positive cases incorrectly anticipated as negative cases is known as the false negative rate (FNR), and is determined as shown in Equation (5) [90].

$$FNR = \frac{FN}{(TP + FN)} \quad (5)$$

The TNR or True Negative Rate is shown in Equation (6); Specificity is another name for it. It is described as the percentage of adverse events accurately foreseen as adverse [90].

$$TNR = \frac{TN}{(TN + FP)} \quad (6)$$

It is challenging to compare two models if one has great recall and low accuracy or vice versa. The F1-score is therefore used to compare them. It is employed to simultaneously assess memory and precision [92]. Equation (7) is used to compute the F1-score:

$$\text{F1-Score} = \frac{2 * \text{Recall} * \text{Precision}}{(\text{Recall} + \text{Precision})} \quad (7)$$

Efficiency at different threshold levels for classification problems is known as the AUC–ROC curve. A model makes more accurate predictions if the AUC is near 1 [93].

## 7. Research Gaps in the Existing Literature

The research gaps detailed below were identified through our thorough assessment of the literature.

- Insufficiently large datasets: due to the potential loss of reputation or money, the majority of victim organizations are reluctant to disclose information regarding attacks undertaken against them. Furthermore, there are no complete databases in the public domain that include all traffic kinds, including genuine, low rate, high rate, and flash traffic [37,39,40,42,53–73,75–77,81]. Therefore, experimental settings are necessary to provide extensive datasets for the thorough validation of DDoS detection methodologies.
- Access to skewed datasets: occurrences of DDoS attacks are typically highly skewed in comparison to genuine events in the datasets currently available [37,39,40,53–56,60–67,70–72,75,77]. Therefore, a large number of cases in each class are required in order to execute deep learning techniques effectively. For effective study in this area, good enhancement strategies are needed to provide a large enough volume of all forms of traffic.
- Demand for high-quality preprocessed data: the caliber of the preprocessed data affects how accurate the resulting deep learning model is. As a result, effective preprocessing methods are needed for effective DL model training [61–63,65–70,72,75].
- Binary categorization: the majority of the currently available literature [37,39,40,42,53,54,57,63–65,67,68,70–73,75–78,81] concentrates on binary categorization of DDoS assaults rather than multi-class classification.
- Insufficient effort on unknown data or zero-day attacks: when the instruction and assessment datasets contain the same traits or patterns, ML models are able to function well. However, ML-based algorithms are unable to accurately detect unknown threats in real-life situations, where attacks may be launched using novel patterns. As a result, these models must be frequently updated in order to account for novel and untested assaults [53].
- Using an offline dataset for evaluation: the majority of the research we reviewed used offline datasets to assess deep learning models [37,39,42,55–59,61–67,69,70,72,73,75,77,78,81]. The implementation of these models in actual networks remains a work in progress. Real-time evaluation of models would be highly beneficial for adequate verification.
- No deployment of automated real-time defense models: most DDoS assaults overwhelm the target site in a relatively short amount of time, and network managers are often unable to automatically identify and fight back against these attacks. The primary cause of this is that defense strategies themselves become susceptible to DDoS assaults based on floods. Therefore, high-speed and computationally efficient DDoS solutions are needed in order to automatically stop those attacks.

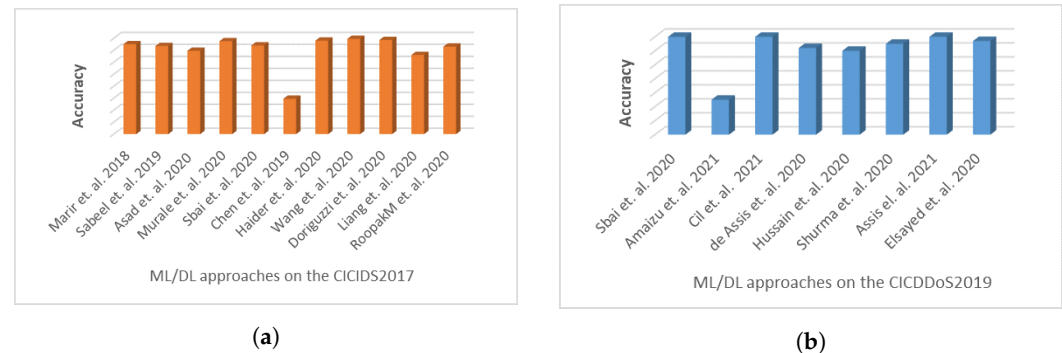
## 8. Conclusions and Future Directions

It may be quite difficult to distinguish between DDoS assaults with various rates and patterns and normal traffic. Over the years, many effective ML/DL methods for DDoS attack detection have been suggested by different researchers. Sadly, however, the applicability of these techniques is severely constrained due to attackers constantly changing their attack tactics. Findings involving the SLR protocol are evaluated and drawn from in this review in order to assess the state-of-the-art DDoS assault detection systems based on ML/DL approaches. The literature has been summarized in Section 4 in accordance with the suggested taxonomy for DDoS attack detection using ML/DL techniques, with each study's respective advantages and disadvantages listed. The accuracy rate reported in much of the literature is over 99%. Because the majority of these studies assessed their models using offline data analysis for evaluation and comparison, certain metrics for performance may vary in a real-world or production settings. In particular, we note that existing papers have generally not employed the same DS or assessment techniques, making comparisons between their results difficult.

With respect to the datasets most often used in the literature, 29% of studies utilized the current well-known research dataset CICIDS2017, 23% used the CICDDoS2019 dataset, 18% used the ISCX2012 dataset, 10% used the NSL-KDD dataset, 10% used the KDDCUP99



dataset, 5% used the UNSW-NB15 dataset, 3% used the CSE-CIC-IDS2018 dataset, and 2% used the Kyoto 2006 dataset. Figure 5a displays the efficacy of the investigated ML/DL-based DDoS attack detection techniques on the CICIDS2017 dataset. It can be seen that methods relying on CNN [64,67], DNN [56], AE-SVM [76], and CNN-LSTM [39] have all been able to achieve accuracy better than 99%. Figure 5b shows how well the investigated ML/DL-based DDoS attack detection methods performed on the CICDoS2019 dataset. Techniques relying on CNN-based ResNet [69], LSTM [73], DNN [57–59], and GRU [37] all demonstrated accuracy over than 99%.

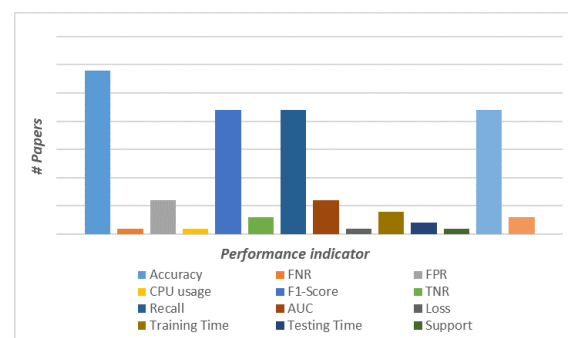


**Figure 5.** Accuracy of the studied ML/DL approaches on (a) the CICIDS2017 dataset [38,49,53,55–57,62,64,65,67,72] and (b) the CICDDoS2019 dataset [37,42,53,58,59,68,69,73].

On the ISCX2012 dataset, the LSTM, CNN, and LSTM-Bayes techniques all showed accuracy lower than 98.8% [70,71]. On the NSL-KDD dataset, only the CNN technique in [63] demonstrated accuracy above 99%, and it required complex calculations to achieve this.

We can conclude from this study that the most commonly used preprocessing techniques are BOW, Z-score normalization, one-hot encoding, and min-max normalization.

Another conclusion of our review relates to performance metrics. Of the reviewed studies, 29 employed accuracy measurements for evaluating their techniques, compared to 22 studies each using the precision, recall, and F1-score metrics and six studies each using the FPR and AUC metrics. These findings are shown in Figure 6; it can be seen that the majority of papers did not report the testing/training time for their methodologies, despite the fact that these measurements are crucial for system implementation in real-world or production settings.



**Figure 6.** The proportion of ML/DL methods that utilized different performance indicators.

With respect to future research directions, our discoveries on ML/DL techniques for DDoS attack detection point towards the following paths for further study:

- **Lack of actual implementation of ML/DL systems:** most research focusing on analysis of these models has neglected the crucial need to evaluate the performance of these models in the real-time situations where DDoS attacks actually occur. There remains a pressing need for ML/DL models that have been verified using real-world scenarios.

- **ML/DL models rely on dynamically and frequent updating:** models that can be dynamically and routinely updated in accordance with new types of attacks are a necessity due to constant and rapid changes in attack patterns, and is a crucial element in today's world of quickly developing new technologies that carry with them more sophisticated threats. However, no such DL models are available in the literature.
- **Requirement for lightweight ML/DL models:** lightweight models are necessary for networks such as the Internet of Things, MANETS, and wireless sensor networks, as these have limited computational power and memory and are highly susceptible to security threats. In the future, it is expected to become increasingly necessary to develop effective and portable DL models for these contexts.
- **Need for appropriate datasets:** the current datasets lack diversity in terms of the types of attacks and quality of the data recordings they contain, leading to biased detection systems that are unable to identify all types of attacks. It is essential to have sufficient datasets in order to ensure accurate and effective detection models.

To close, addressing these areas of research is important in order to realize significant advances in this field and bridge the gaps that currently exist in the literature.

**Author Contributions:** Conceptualization, T.E.A. and Y.-W.C.; Methodology, T.E.A. and Y.-W.C.; Software, T.E.A. and Y.-W.C.; Validation, T.E.A. and Y.-W.C.; Resources, T.E.A. and Y.-W.C.; Writing—original draft, T.E.A.; Supervision, Y.-W.C. and S.M. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by Publication Fund under Research Creativity and Management Office, Universiti Sains Malaysia and Universiti Sains Malaysia (USM) external grant (Grant Number: 304/PNAV/650958/U154).

**Institutional Review Board Statement:** This article does not contain any studies involving human participants or animals performed by any of the authors.

**Informed Consent Statement:** Informed consent was obtained from all individual participants included in the study.

**Conflicts of Interest:** The authors declare that they have no conflict of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

AE	Auto Encoder
ANN	Artificial Neural Network
BOW	Bag of Word
CIC	Canadian Institute of Cyber security
CL	Convolutional Layer
CNN	Convolutional Neural Network
DDoS	Distributed Denial of Service
DL	Deep Learning
DNN	Deep Neural Network
DoS	Denial of Service
DT	Decision Tree
FNR	False Negative Rate
FPR	False Positive Rate
GPU	Graphics Processing Unit
IDS	Intrusion Detection System
IoT	Internet of Things
IP	Internet Protocol
KNN	k-Nearest Neighbours
LR	Logistic Regression
LSTM	Long short-term memory

LUCID	Lightweight Usable CNN in DDoS Detection
MKL	Multiple Kernel Learning
ML	Machine Learning
MLP	Multilayer Perceptron
MSE	Mean Squared Error
NB	Naïve Bayes
NID	Network Intrusion Detection
NN	Neural Network
PDR	Packet Delivery Ratio
RF	Random Forest
RNN	Recurrent Neural Network
SDN	Software Defined Network
SLR	Systematic Literature Review
SML	Shallow Machine Learning
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TL	Transfer Learning
TNR	True Negative Rate
TPR	True Positive Rate
UDP	User Datagram Protocol
WSN	Wireless Sensor Network
DL	Deep Neural Network
ELM	Extreme Learning Machine
FAR	False Alarm Rate
FCN	Fully Connected Network
FN	False Negative
FP	False Positive
TN	True Negative
TP	True Positive
AI	Artificial Intelligence
KC	KDD Cup
KY	Kyoto
NK	NSL-KDD
UN	UNSW-NB15
C7	CIC-IDS2017
C8	CSE-CIC-IDS2018
I2	ISCX2012
C9	CICDDoS2019
FNR	False negative rate
FPR	False Positive rate

## References

1. Ali, T.; Morad, A.; Abdala, M. Load balance in data center sdn networks. *Int. J. Electr. Comput. Eng.* **2018**, *8*, 3086–3092.
2. Ali, T.; Morad, A.; Abdala, M. SDN Implementation in Data Center Network. *J. Commun.* **2019**, 223–228. [\[CrossRef\]](#)
3. Ali, T.; Morad, A.; Abdala, M. Traffic management inside software-defined data center networking. *Bull. Electr. Eng. Inform.* **2020**, *9*, 2045–2054. [\[CrossRef\]](#)
4. Cybersecurity and Infrastructure Security Agency. Available online: <https://www.cisa.gov/uscert/ncas/tips/ST04-015> (accessed on 20 November 2021).
5. Eliyan, L.F.; Di Pietro, R. DoS and DDoS attacks in Software Defined Networks: A survey of existing solutions and research challenges. *Future Gener. Comput. Syst.* **2021**, *122*, 149–171. [\[CrossRef\]](#)
6. Cryptocurrency Exchange EXMO Has Been Knocked Offline by a “Massive” DDoS Attack. Available online: <https://portswigger.net/daily-swig/uk-cryptocurrency-exchange-exmo-knocked-offline-by-massive-ddos-attack> (accessed on 1 November 2021).
7. Catak, F.O.; Mustacoglu, A.F. Distributed denial of service attack detection using autoencoder and deep neural networks. *J. Intell. Fuzzy Syst.* **2019**, *37*, 3969–3979. [\[CrossRef\]](#)

8. Li, Y.; Lu, Y. LSTM-BA: DDoS detection approach combining LSTM and bayes. In Proceedings of the 2019 7th International Conference on Advanced Cloud and Big Data (CBD), Suzhou, China, 21–22 September 2019; pp. 180–185.
9. Yuan, X.; Li, C.; Li, X. DeepDefense: Identifying DDoS attack via deep learning. In Proceedings of the 2017 IEEE International Conference on Smart Computing (SMARTCOMP), Hong Kong, China, 29–31 May 2017.
10. Xin, Y.; Kong, L.; Liu, Z.; Chen, Y.; Li, Y.; Zhu, H.; Gao, M.; Hou, H.; Wang, C. Machine learning and deep learning methods for cybersecurity. *IEEE Access* **2018**, *6*, 35365–35381. [\[CrossRef\]](#)
11. Van N.T.; Thinh, T.N.; Sach, L.T. An anomaly-based network intrusion detection system using deep learning. In Proceedings of the 2017 International Conference on System Science and Engineering (ICSSE), Ho Chi Minh City, Vietnam, 21–23 July 2017; pp. 210–214.
12. Vinayakumar, R.; Soman, K.P.; Poornachandran, P. Applying convolutional neural network for network intrusion detection. In Proceedings of the 2017 International Conference on Advances in Computing, Communications and Informatics, (ICACCI), Udupi, India, 13–16 September 2017; pp. 1222–1228.
13. Aldweesh, A.; Derhab, A.; Emam, A.Z. Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. *Knowl. Based Syst.* **2020**, *189*, 105124. [\[CrossRef\]](#)
14. Wang, Y.; Lou, X.; Fan, Z.; Wang, S.; Huang, G. Verifiable multi-dimensional (t, n) threshold quantum secret sharing based on a quantum walk. *Int. J. Theor. Phys.* **2022**, *61*, 24. [\[CrossRef\]](#)
15. Trending News about Artificial Intelligence. Summary: In-Depth Guide to Quantum Artificial Intelligence. Available online: <https://www.ai-summary.com/summary-in-depth-guide-to-quantum-artificial-intelligence/> (accessed on 22 January 2022).
16. Ferrag, M.A.; Maglaras, L.; Moschoyiannis, S.; Janicke, H. Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *J. Inf. Secur. Appl.* **2020**, *50*, 102419. [\[CrossRef\]](#)
17. Aleesa, A.M.; Zaidan, B.B.; Zaidan, A.A.; Sahar, N.M. Review of intrusion detection systems based on deep learning techniques: Coherent taxonomy, challenges, motivations, recommendations, substantial analysis and future directions. *Neural Comput. Appl.* **2020**, *32*, 9827–9858. [\[CrossRef\]](#)
18. Gamage, S.; Samarabandu, J. Deep learning methods in network intrusion detection: A survey and an objective comparison. *J. Netw. Comput. Appl.* **2020**, *169*, 102767. [\[CrossRef\]](#)
19. Ahmad, Z.; Khan, A.S.; Shiang, C.W.; Abdullah, J.; Ahmad, F. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4150. [\[CrossRef\]](#)
20. Ahmad, R.; Alsmadi, I. Machine learning approaches to IoT security: A systematic literature review. *Internet Things* **2021**, *14*, 100365. [\[CrossRef\]](#)
21. Keele, S. *Guidelines for Performing Systematic Literature Reviews in Software Engineering*; Technical Report, Ver. 2.3 EBSE Technical Report; Keele University and Durham University Joint Report EBSE: Newcastle, UK, 2007.
22. Costa, V.G.; Pedreira, C.E. Recent advances in decision trees: An updated survey. *Artif. Intell. Rev.* **2022**, 1–36. [\[CrossRef\]](#)
23. Zhang, Y.; Cao, G.; Wang, B.; Li, X. A novel ensemble method for k-nearest neighbor. *Pattern Recogn.* **2019**, *85*, 13–25. [\[CrossRef\]](#)
24. Yılmaz, A.; Küçüker, A.; Bayrak, G.; Ertekin, D.; Shafie-Khah, M.; Guerrero, J.M. An improved automated PQD classification method for distributed generators with hybrid SVM-based approach using un-decimated wavelet transform. *Int. J. Electr. Power Energy Syst.* **2022**, *136*, 107763. [\[CrossRef\]](#)
25. Ren, H.; Lu, H. Compositional coding capsule network with k-means routing for text classification. *Pattern Recognit. Lett.* **2022**, *160*, 1–8. [\[CrossRef\]](#)
26. Gopi, R.; Sathiyamoorthi, V.; Selvakumar, S.; Manikandan, R.; Chatterjee, P.; Jhanjhi, N.Z.; Luhach, A.K. Enhanced method of ANN based model for detection of DDoS attacks on multimedia internet of things. *Multimed. Tools Appl.* **2022**, *81*, 26739–26757. [\[CrossRef\]](#)
27. Zeinalpour, A.; Ahmed, H.A. Addressing the Effectiveness of DDoS-Attack Detection Methods Based on the Clustering Method Using an Ensemble Method. *Electronics* **2022**, *11*, 2736. [\[CrossRef\]](#)
28. AI vs. Machine Learning vs. Deep Learning: Know the Differences. Available online: <https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/ai-vs-machine-learning-vs-deeplearning#:~:text=Machine%20Learning%20is%20a%20subset,algorithms%20to%20train%20a%20model> (accessed on 1 January 2023).
29. Bachouch, A.; Huré, C.; Langrené, N.; Pham, H. Deep neural networks algorithms for stochastic control problems on finite horizon: Numerical applications. *Methodol. Comput. Appl. Probab.* **2022**, *24*, 143–178. [\[CrossRef\]](#)
30. Sellami, A.; Tabbone, S. Deep neural networks-based relevant latent representation learning for hyperspectral image classification. *Pattern Recognit.* **2022**, *121*, 108224. [\[CrossRef\]](#)
31. Machine Learning Mastery. A Gentle Introduction to the Rectified Linear Unit (ReLU). Available online: <https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/#:~:text=The%20rectified%20linear%20activation%20function,otherwise%2C%20it%20will%20output%20zero> (accessed on 1 January 2023).
32. Santos, C.F.G.D.; Papa, J.P. Avoiding overfitting: A survey on regularization methods for convolutional neural networks. *ACM Comput. Surv. CSUR* **2022**, *54*, 213. [\[CrossRef\]](#)
33. Yadav, S.P.; Zaidi, S.; Mishra, A.; Yadav, V. Survey on machine learning in speech emotion recognition and vision systems using a recurrent neural network (RNN). *Arch. Comput. Methods Eng.* **2022**, *29*, 1753–1770. [\[CrossRef\]](#)
34. Types of Neural Networks and Definition of Neural Networks. Available online: <https://www.mygreatlearning.com/blog/types-of-neural-networks> (accessed on 25 November 2022).

35. Mehedi, M.A.A.; Khosravi, M.; Yazdan, M.M.S.; Shabanian, H. Exploring Temporal Dynamics of River Discharge using Univariate Long Short-Term Memory (LSTM) Recurrent Neural Network at East Branch of Delaware River. *Hydrology* **2022**, *9*, 202. [CrossRef]
36. Recurrent Neural Networks and LSTM Explained. Available online: <https://purnasaigudikandula.medium.com/recurrent-neural-networks-and-lstm-explained-7f51c7f6bbb9> (accessed on 25 November 2022).
37. Assis, M.V.; Carvalho, L.F.; Lloret, J.; Proença, M.L. A GRU deep learning system against attacks in software defined networks. *J. Netw. Comput. Appl.* **2021**, *177*, 102942. [CrossRef]
38. Roopak, M.; Tian, G.Y.; Chambers, J. Deep learning models for cyber security in IoT networks. In Proceedings of the 2019 IEEE 9th Annual Computing and Communication Workshop and Conference, (CCWC), Las Vegas, NV, USA, 7–9 January 2019; pp. 452–457.
39. Roopak, M.; Tian, G.Y.; Chambers, J. An intrusion detection system against DDoS attacks in IoT networks. In Proceedings of the 2020 10th annual Computing and Communication Workshop and Conference, (CCWC), Las Vegas, NV, USA, 6–8 January 2020; pp. 562–567.
40. Nugraha, B.; Murthy, R.N. Deep learning-based slow DDoS attack detection in SDN-based networks. In Proceedings of the 2020 IEEE conference on Network Function Virtualization and Software Defined Networks, (NFV-SDN), Leganes, Spain, 10–12 November 2020; pp. 51–56.
41. Mohammad, H.; Slimane, S. IoT-NETZ: Practical spoofing attack mitigation approach in SDWN network. In Proceedings of the 2020 Seventh International Conference on Software Defined Systems (SDS), Paris, France, 20–23 April 2020; pp. 5–13.
42. Elsayed, M.S.; Le-Khac, N.A.; Dev, S.; Jurcut, A.D. DDoSNet: A deep learning model for detecting network attacks. In Proceedings of the 21st IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Cork, Ireland, 31 August–3 September 2020; pp. 391–396.
43. Shen, Y.; Zheng, K.; Wu, C.; Zhang, M.; Niu, X.; Yang, Y. An ensemble method based on selection using bat algorithm for intrusion detection. *Comput. J.* **2018**, *61*, 526–538. [CrossRef]
44. Shone, N.; Ngoc, T.N.; Phai, V.D.; Shi, Q. A deep learning approach to network intrusion detection. *IEEE Trans. Emerg. Top. Comput. Intell.* **2018**, *2*, 41–50. [CrossRef]
45. Ali, M.H.; Al Mohammed, B.A.D.; Ismail, A.; Zolkipli, M.F. A new intrusion detection system based on fast learning network and particle swarm optimization. *IEEE Access* **2018**, *6*, 20255–20261. [CrossRef]
46. Yan, B.; Han, G. Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system. *IEEE Access* **2018**, *6*, 41238–41248. [CrossRef]
47. Naseer, S.; Saleem, Y.; Khalid, S.; Bashir, M.K.; Han, J.; Iqbal, M.M.; Han, K. Enhanced network anomaly detection based on deep neural networks. *IEEE Access* **2018**, *6*, 48231–48246. [CrossRef]
48. Al-Qatf, M.; Lasheng, Y.; Al-Habib, M.; Al-Sabahi, K. Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. *IEEE Access* **2018**, *6*, 52843–52856. [CrossRef]
49. Marir, N.; Wang, H.; Feng, G.; Li, B.; Jia, M. Distributed abnormal behavior detection approach based on deep belief network and ensemble svm using spark. *IEEE Access* **2018**, *6*, 59657–59671. [CrossRef]
50. Yao, H.; Fu, D.; Zhang, P.; Li, M.; Liu, Y. MSML: A novel multilevel semi-supervised machine learning framework for intrusion detection system. *IEEE IoT J.* **2018**, *6*, 1949–1959. [CrossRef]
51. Gao, X.; Shan, C.; Hu, C.; Niu, Z.; Liu, Z. An adaptive ensemble machine learning model for intrusion detection. *IEEE Access* **2019**, *7*, 82512–82521. [CrossRef]
52. Karatas, G.; Demir, O.; Sahingoz, O.K. Increasing the performance of machine learning-based IDSs on an imbalanced and up-to-date dataset. *IEEE Access* **2020**, *8*, 32150–32162. [CrossRef]
53. Sabeel, U.; Heydari, S.S.; Mohanka, H.; Bendhaou, Y.; Elgazzar, K.; El-Khatib, K. Evaluation of deep learning in detecting unknown network attacks. In Proceedings of the 2019 International Conference on Smart Applications, Communications and Networking (SmartNets), Sharm El Sheikh, Egypt, 17–19 December 2019.
54. Virupakshar, K.B.; Asundi, M.; Channal, K.; Shettar, P.; Patil, S.; Narayan, D.G. Distributed Denial of Service (DDoS) attacks detection system for OpenStack-based Private Cloud. *Procedia Comput. Sci.* **2020**, *167*, 2297–2307. [CrossRef]
55. Asad, M.; Asim, M.; Javed, T.; Beg, M.O.; Mujtaba, H.; Abbas, S. Deep- Detect: Detection of Distributed Denial of Service attacks using deep learning. *Comput. J.* **2020**, *63*, 983–994. [CrossRef]
56. Muraleedharan, N.; Janet, B. A deep learning based HTTP slow DoS classification approach using flow data. *ICT Express* **2020**, *7*, 210–214.
57. Sbair, O.; El Boukhari, M. Data flooding intrusion detection system for manets using deep learning approach. In Proceedings of the SITA'20: Proceedings of the 13th International Conference on Intelligent Systems: Theories and Applications, Rabat, Morocco, 23–24 September 2020; pp. 281–286.
58. Amaizu, G.C.; Nwakanma, C.I.; Bhardwaj, S.; Lee, J.M.; Kim, D.S. Composite and efficient DDoS attack detection framework for B5G networks. *Comput. Netw.* **2021**, *188*, 107871. [CrossRef]
59. Cil, A.E.; Yildiz, K.; Buldu, A. Detection of DDoS attacks with feed forward based deep neural network model. *Expert Syst. Appl.* **2021**, *169*, 114520. [CrossRef]
60. Hasan, M.Z.; Hasan, K.M.Z.; Sattar, A. Burst header packet flood detection in optical burst switching network using deep learning model. *Procedia Comput. Sci.* **2018**, *143*, 970–977. [CrossRef]



61. Amma, N.G.B.; Subramanian, S. VCDDeepFL: Vector Convolutional Deep Feature Learning approach for identification of known and unknown Denial of Service Attacks. In Proceedings of the IEEE Region 10 Annual International Conference, TENCON, Jeju, Republic of Korea, 28–31 October 2018; pp. 640–645.
62. Chen, J.; Yang, Y.; Hu, K.; Zheng, H.; Wang, Z. DADMCNN: DDoS attack detection via multi-channel CNN. In Proceedings of the ICMMLC '19: Proceedings of the 2019 11th International Conference on Machine Learning and Computing, Zhuhai, China, 22–24 February 2019; pp. 484–488.
63. Shaaban, A.R.; Abd-Elwanis, E.; Hussein, M. DDoS attack detection and classification via Convolutional Neural Network (CNN). In Proceedings of the 2019 IEEE 9th International Conference on Intelligent Computing and Information Systems (ICICIS), Cairo, Egypt, 8–10 December 2019; pp. 233–238.
64. Haider, S.; Akhunzada, A.; Mustafa, I.; Patel, T.B.; Fernandez, A.; Choo, K.K.R.; Iqbal, J. A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks. *IEEE Access* **2020**, *8*, 53972–53983. [\[CrossRef\]](#)
65. Wang, L.; Liu, Y. A DDoS attack detection method based on information entropy and deep learning in SDN. In Proceedings of the 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference, (ITNEC), Chongqing, China, 12–14 June 2020; pp. 1084–1088.
66. Kim, J.; Kim, J.; Kim, H.; Shim, M.; Choi, E. CNN-based network intrusion detection against Denial-of-Service attacks. *Electronics* **2020**, *9*, 916. [\[CrossRef\]](#)
67. Doriguzzi-Corin, R.; Millar, S.; Scott-Hayward, S.; Martinez-Del-Rincon, J.; Siracusa, D. Lucid: A practical, lightweight deep learning solution for DDoS attack detection. *IEEE Trans. Netw. Serv. Manag.* **2020**, *17*, 876–889. [\[CrossRef\]](#)
68. de Assis, M.V.; Carvalho, L.F.; Rodrigues, J.J.; Lloret, J.; Proença, M.L. Near real-time security system applied to SDN environments in IoT networks using convolutional neural network. *Comput. Electr. Eng.* **2020**, *86*, 106738. [\[CrossRef\]](#)
69. Hussain, F.; Ghazanfar, S.; Al-Khawarizmi, A.; Husnain, M.; Fayyaz, U.U.; Shahzad, F.; Al-Khawarizmi, G.A.S. IoTDoS and DDoS attack detection using ResNet. In Proceedings of the 2020 IEEE 23rd International Multitopic Conference (INMIC), Bahawalpur, Pakistan, 5–7 November 2020.
70. Li, C.; Wu, Y.; Yuan, X.; Sun, Z.; Wang, W.; Li, X.; Gong, L. Detection and defense of DDoS attack-based on deep learning in OpenFlowbased SDN. *Int. J. Commun. Syst.* **2018**, *31*, e3497. [\[CrossRef\]](#)
71. Priyadarshini, R.; Barik, R.K. A deep learning based intelligent framework to mitigate DDoS attack in fog environment. *J. King Saud Univ. Comput. Inf. Sci.* **2019**, *34*, 825–831. [\[CrossRef\]](#)
72. Liang, X.; Znati, T. A long short-term memory enabled framework for DDoS detection. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 9–13 December 2019.
73. Shurman, M.; Khrais, R.; Yateem, A. DoS and DDoS attack detection using deep learning and IDS. *Int. Arab. J. Inf. Technol.* **2020**, *17*, 655–661. [\[CrossRef\]](#)
74. Ali, S.; Li, Y. Learning multilevel auto-encoders for DDoS attack detection in smart grid network. *IEEE Access* **2019**, *7*, 108647–108659. [\[CrossRef\]](#)
75. Yang, K.; Zhang, J.; Xu, Y.; Chao, J. DDoS attacks detection with AutoEncoder. In Proceedings of the IEEE/IFIP Network Operations and Management Symposium 2020: Management in the Age of Softwarization and Artificial Intelligence (NOMS), Budapest, Hungary, 20–24 April 2020.
76. Kasim, O. An efficient and robust deep learning based network anomaly detection against distributed denial of service attacks. *Comput. Netw.* **2020**, *180*, 107390. [\[CrossRef\]](#)
77. Bhardwaj, A.; Mangat, V.; Vig, R. Hyperband tuned deep neural network with well posed stacked sparse AutoEncoder for detection of DDoS attacks in Cloud. *IEEE Access* **2020**, *8*, 181916–181929. [\[CrossRef\]](#)
78. He, J.; Tan, Y.; Guo, W.; Xian, M. A small sample DDoS attack detection method based on deep transfer learning. In Proceedings of the 2020 International Conference on Computer Communication and Network Security (CCNS), Xi'an, China, 21–23 August 2020; pp. 47–50.
79. Chen, M.; Liu, W.; Zhang, N.; Li, J.; Ren, Y.; Yi, M.; Liu, A. GPDS: A multi-agent deep reinforcement learning game for anti-jamming secure computing in MEC network. *Expert Syst. Appl.* **2022**, *210*, 118394. [\[CrossRef\]](#)
80. Computer Network Intrusion Detection. Available online: <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html> (accessed on 25 October 2022).
81. Premkumar, M.; Sundararajan, T.V. DLDM: Deep learning-based defense mechanism for denial of service attacks in wireless sensor networks. *Microprocess. Microsyst.* **2020**, *79*, 103278. [\[CrossRef\]](#)
82. Canadian Institute for Cybersecurity. Available online: <https://www.unb.ca/cic/datasets/nsl.html> (accessed on 1 October 2022).
83. Canadian Institute for Cybersecurity. Available online: <https://www.unb.ca/cic/datasets/ids-2017.html> (accessed on 1 October 2022).
84. Canadian Institute for Cybersecurity. Available online: <https://www.unb.ca/cic/datasets/ids-2018.html> (accessed on 1 October 2022).
85. Canadian Institute for Cybersecurity. Available online: <https://www.unb.ca/cic/datasets/ids.html> (accessed on 1 October 2022).
86. Sharafaldin, I.; Lashkari, A.H.; Hakak, S.; Ghorbani, A.A. Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy. In Proceedings of the International Carnahan Conference on Security Technology (ICCST), Chennai, India, 1–3 October 2019.



87. Canadian Institute for Cybersecurity. Available online: <https://www.unb.ca/cic/datasets/ddos-2019.html> (accessed on 1 October 2022).
88. Holzinger, A. Big data calls formachine learning. *Encycl. Biomed. Eng.* **2019**, *3*, 258–264.
89. Metrics to Evaluate your Machine Learning Algorithm. Available online: <https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234> (accessed on 1 October 2022).
90. Amanullah, M.A.; Habeeb, R.A.A.; Nasaruddin, F.H.; Gani, A.; Ahmed, E.; Nainar, A.S.M.; Akim, N.M.; Imran, M. Deep learning and big data technologies for IoT security. *Comput. Commun.* **2020**, *151*, 495–517. [CrossRef]
91. Understanding Confusion Matrix. Available online: <https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62> (accessed on 1 October 2022).
92. Machine Learning Mastery. Available online: <https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/> (accessed on 1 October 2022).
93. Understanding AUC—ROC Curve. Available online: <https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5> (accessed on 1 October 2022).

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.