applied sciences

MDPI

Review

A Study of Network Intrusion Detection Systems Using Artificial Intelligence/Machine Learning

Patrick Vanin [1], Thomas Newe [1,2,*], Lubna Luxmi Dhirani [1,2], Eoin O'Connell [1,2], Donna O'Shea [2,3], Brian Lee [2,4] and Muzaffar Rao [1,2]

[1] Department of Electronic and Computer Engineering, University of Limerick, V94 T9PX Limerick, Ireland
[2] Confirm—SFI Centre for Smart Manufacturing, Park Point, Dublin Rd, Castletroy, V94 C928 Limerick, Ireland
[3] Department of Computer Sciences, Munster Technological University (MTU), T12 P928 Cork, Ireland
[4] Software Research Institute, Technological University of the Shannon, Midlands Midwest, N37 HD68 Athlone, Ireland
* Correspondence: thomas.newe@ul.ie

Abstract: The rapid growth of the Internet and communications has resulted in a huge increase in transmitted data. These data are coveted by attackers and they continuously create novel attacks to steal or corrupt these data. The growth of these attacks is an issue for the security of our systems and represents one of the biggest challenges for intrusion detection. An intrusion detection system (IDS) is a tool that helps to detect intrusions by inspecting the network traffic. Although many researchers have studied and created new IDS solutions, IDS still needs improving in order to have good detection accuracy while reducing false alarm rates. In addition, many IDS struggle to detect zero-day attacks. Recently, machine learning algorithms have become popular with researchers to detect network intrusion in an efficient manner and with high accuracy. This paper presents the concept of IDS and provides a taxonomy of machine learning methods. The main metrics used to assess an IDS are presented and a review of recent IDS using machine learning is provided where the strengths and weaknesses of each solution is outlined. Then, details of the different datasets used in the studies are provided and the accuracy of the results from the reviewed work is discussed. Finally, observations, research challenges and future trends are discussed.

Keywords: Intrusion Detection Systems (IDS); machine learning; network security; Intrusion Prevention Systems (IPS); deep learning algorithms

check for updates

Citation: Vanin, P.; Newe, T.; Dhirani, L.L.; O'Connell, E.; O'Shea, D.; Lee, B.; Rao, M. A Study of Network Intrusion Detection Systems Using Artificial Intelligence/Machine Learning. *Appl. Sci.* **2022**, *12*, 11752. https://doi.org/10.3390/app122211752

Academic Editor: Gianluca Lax

Received: 25 October 2022
Accepted: 15 November 2022
Published: 18 November 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 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

The concept of IDS started in 1980 with James Anderson's paper, Computer Security Threat Monitoring and Surveillance [1]. James was a member of the Defense Science Board Task Force on Computer Security at the U.S. Air Force. In his paper, James explained how analyzing audit trails could be useful to detect unauthorized or malicious activities. For instance, analyzing the log of the files provides important information to determine if there is an abnormal use. The paper formed the basis of what will be known as Host Intrusion Detection System (HIDS). Soon after the first IDS was built to detect threats that were in a list of known attacks.

At the end of the 1980's, many systems administrators started to use intrusion detection systems. However, they were vulnerable and unable to detect zero-day attacks, and they used a lot of resources to constantly scan the network [2].

In the 1990's, a new method of detection was explored to face the increasing number of attacks. This method was anomaly detection and it looked for unusual behaviour/activity in a system to raise an alarm. Nevertheless, the inconsistent nature of the networks, between the 1990s and the 2000s, caused a lot of false alarms. Thus, many administrators stopped using IDS due to their unreliability [2].

Fortunately, thanks to, on the one hand, the improvement made to the network and the processing power of the network's components, and on the other hand, the advent of machine learning, IDS is now a key component of the security of many systems. Most machine learning techniques have been evaluated in the development of IDS. However, the use of deep learning algorithms has not been sufficiently explored. With deep learning, many more possibilities can be achieved to tackle the issue of false alarm rates and lack of high accuracy.

The first goal of this paper is to present the key concepts of IDS and machine learning. The second goal of the paper is to provide the recent trends and observations on recent machine learning intrusion detection systems. We review related published work, their advantages, disadvantages, datasets, techniques, and evaluation metrics. We discuss what has been done and what could be done to improve the IDS using new machine learning techniques. Finally, future trends and research challenges are outlined.

The paper is organized as follows: Section 2 describes the concept of IDS. Section 3 explains the different metrics used to assess IDSs. Section 4 provides the basic understanding of machine learning, Section 5 provides datasets description and Section 6 reviews the relevant papers used in the study. Evaluation metrics are discussed in Section 7. Observations, research challenges and future trends are provided in Section 8. Finally, Section 9 concludes this paper.

2. Intrusion Detection Systems (IDS)

An IDS is a tool, either a hardware device or software, that raises an alarm when it detects malicious activity on the network. IDSs are the "watch-eye" of the network. It is an important component of the security architecture of modern networks. It allows early stages of attacks to be detected and thus, gives the opportunity to mitigate against them. In addition, they allow detection of a variety of attacks, e.g.: Denial of Service (DoS), Man in the Middle (MitM), etc. IDSs can monitor and log any, and all, network traffic as specified. Since IDSs can provide details on attacks as they occur, it helps security administrators to understand what has happened. Therefore, in the case of future similar attacks the security of the system can be configured to detect and prevent attacks of this type. There are two types of IDS, Network Intrusion Detection System (NIDS) and Host Intrusion Detection System (HIDS) [3]:

- Network Intrusion Detection Systems are IDSs placed in the network at strategic points. NIDS analyses the overall traffic of the network to detect if there are malicious activities in the network. NIDS helps to detect attacks from your own hosts and is a key component of the security of most organization networks.
- Host Intrusion Detection Systems are IDSs that are in all client computers (hosts) of the network. Contrary to NIDS, HIDS analyses the traffic of a single host as well as its activities, if it detects abnormal behaviour, it will raise an alarm.

There are three different detection types for IDS: Misuse Detection, Anomaly Detection, and Hybrid detection [3]:

- Misuse detection, also known as signature detection, searches for known patterns of intrusion in the network or in the host. Each attack has a specific signature, for instance, it can be the payload of the packet, the source IP address, or a specific header. The IDS can raise an alarm if it detects an attack that has one of the signatures listed in the list of known signatures of the IDS. The advantage of this approach is its high accuracy to detect known attacks. However, its weakness is that it is inefficient against unknown or zero-day (never seen before attack) patterns.
- Anomaly detection defines a normal state of the network or the host, called a baseline, and any deviation from this baseline is reported as a potential attack. For instance, anomaly-based IDS can create a baseline based on the common network traffic such as the services provided by each host, the services used by each host and the volume of activity during the day. Thus, if an attacker accesses an internal resource at midnight, and if in the baseline there should be almost no activity at midnight, then the IDS will

raise an alarm. The advantage of anomaly detection is its flexibility to find unknown intrusion attacks. However, in most cases it is difficult to precisely define what the baseline of a network is, thus, the false detection rate of these techniques can be high.
- Hybrid detection combines both of the aforementioned detections. Generally, they have a lower false detection rate than anomaly techniques and can discover new attacks.

Nowadays, IDS by themselves are not enough for companies that want to protect themselves from attacks. IDS are more and more being replaced by Intrusion Prevention Systems (IPS). An IPS is similar to an IDS but with active components to stop attacks before they are successful. Usually, an IPS consists of a firewall with IDS rules. Contrary to IDS, IPSs are placed inline, this means that an IPS will continuously scan the traffic as the traffic passes through it. Thus, an IPS needs to be fast and have high computing capacities to avoid causing latency issues in a network which can affect network performance for its users.

One of the main disadvantages of many IPS is false positive attack detection. With an IDS, a false positive can be an inconvenience but for an IPS it can cause DoS, as legitimate traffic will be blocked. In addition, since IPSs, and especially Network Intrusion Prevention Systems (NIPSs), form single point of failure in the network, they need to be highly stable and robust against attacks.

3. IDS Performance Evaluation

To evaluate IDS's, which use machine learning methods, we use performance metrics. A confusion matrix is a table which describe the performance of a classification model, also called a "classifier", on a set of test data for which the true values are known. The matrix compares the actual target values with those predicted by the machine learning model.

To describe a confusion matrix the following terms are used:

- True Positive (TP): An attack sample has been correctly identified as an attack.
- True Negative (TN): A normal sample has been correctly identified as normal traffic.
- False Positive (FP): A normal sample has been incorrectly identified as an attack.
- False Negative (FN): An attack sample has been incorrectly identified as normal traffic

IDS should have a low false positive rate to avoid false alarms in the system, as this creates network disturbances. In addition, it is necessary to have a low false negative rate to avoid undetected attacks entering your network.

When using the aforementioned terms and matrix from Table 1, we can evaluate the various metrics that are frequently used to assess the performance of an IDS. A list of the main metrics that are used by researchers follows here:

Table 1. Metrics used to assess the performance of IDS.

		Predicted Class	
		Normal	**Attack**
Actual class	Normal	TN	FP
	Attack	FN	TP

3.1. Precision

Corresponds to the ratio of correctly predicted attack samples to all the predicted attack samples.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

3.2. Recall

Corresponds to the ratio of correctly predicted attack samples to all the samples that correspond to an attack. This metric is also known as the Detection Rate.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

3.3. False Alarm Rate

Corresponds to the ratio of wrongly predicted attacks to all the samples that are normal. This metric is also known as the False Positive Rate.

$$\text{False Alarm Rate} = \frac{\text{FP}}{\text{TN} + \text{FP}}$$

3.4. True Negative Rate

Corresponds to the ratio of all correctly predicted normal samples to all the samples that are normal.

$$\text{True Negative Rate} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

3.5. Accuracy

Corresponds to the ratio of correctly identified classes to all the samples. This metric is often used to measure the efficiency of an IDS when the dataset is balanced.

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

3.6. F-Measure

Corresponds to the harmonic mean of the Precision and Recall. It is used to provide a better evaluation of the system by showing the gap between the two metrics to see if the solution is balanced. This metric is also known as the F-Score or the F1-Score.

$$\text{F} - \text{Measure} = 2 \times \frac{(\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})}$$

Since the attack detection rate and false alarm rate are often opposed to each other, evaluation of IDSs is also performed using Receiver Operating Characteristics (ROC) analysis. A ROC curve, as shown in Figure 1, represents the trade-off between attack detection rate and false alarm rate. The closer the ROC curve is to the top left corner the more effective the IDS is [4]. As shown in Figure 1, ROC Curves can also be used to compare different IDS using the same dataset.

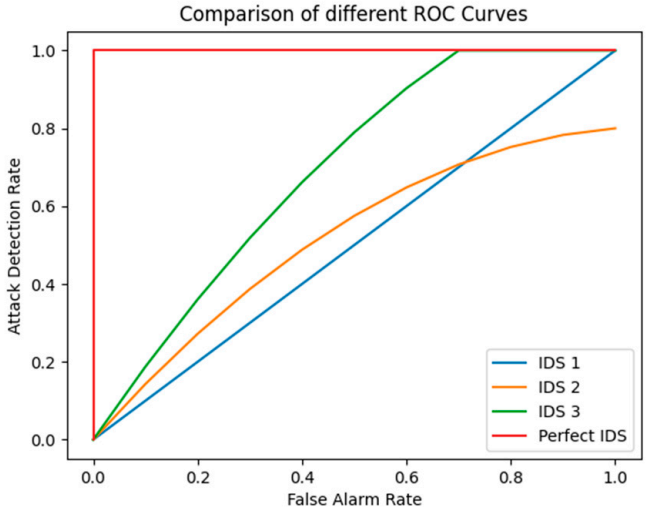

Figure 1. ROC Curves for different IDSs.

IDSs can also be evaluated according to their time performance. The time performance corresponds to the total time that the IDS needs to detect an intrusion. This time is composed of the processing time and the propagation time. The processing time is the

time needed by the IDS to process the information to detect an attack. The processing speed of the IDS needs to be as fast as possible, if not, real-time processing of intrusion is not feasible. The propagation time is the time needed to propagate the information to the security analyst or the Security Operation Centre (SOC). Both times, processing time and propagation time need to be as short as possible to allow security analysts to have enough time to react to an attack in real-time [4].

4. Machine Learning

Machine learning is closely linked to Artificial Intelligence (AI) technology. It trains an algorithm to find regular patterns in a dataset. This training results in a model that can be used to predict or automate things. For IDSs, machine learning can be used to detect either known attacks or unknown attacks if the model has been sufficiently trained.

As shown in Figure 2, there are three main types of machine learning methods: supervised, unsupervised and semi-supervised machine learning [5]. These methods are discussed further in this section.

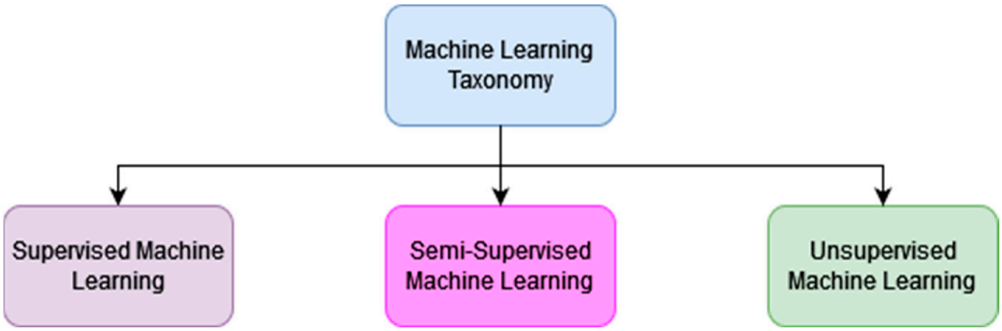

Figure 2. Main types of machine learning methods.

4.1. Supervised Machine Learning

Supervised machine learning uses labelled data to generate a function that maps an input to an output. The function is constructed from labelled training data.

Supervised learning can be categorized into two types of models, classification and regression:

- Classification models are used to put data into specific categories. From a dataset it recognizes the category it belongs to based on its features. The model will be trained on labelled input and output data to understand what the features of the input data are so as to correctly classify it. Classification models are very useful to detect attacks. For instance, for traffic coming into our network, a well-trained classification model can classify the traffic as normal traffic or as abnormal traffic. If the model performs well, the abnormal traffic can then be classified into subcategories of well-known attacks such as DoS, phishing, worms, port scan, etc. Common classification algorithms are decision tree, k-nearest neighbour, support vector machine, random forest, and neural network [6].
- Regression models are used to predict continuous outcomes. The model is trained to understand the relationship between independent variables and a dependent variable. Regression is used to find patterns and relationships in datasets that can then be applied to a new dataset. Regression models are mainly used for forecasting the evolution of market prices or predicting trends [7]. Common regression algorithms are linear regression, logistic regression, decision tree, random forest, and support vector machine.

One of the main advantages of supervised learning is to use previous experiences to produce outputs. In addition, previous results can be used to improve the algorithm by optimizing the performance criteria to reach a precise model.

Supervised learning is used to solve many computational problems. However, the model needs precise and good input during the training phase to produce good outputs. In addition, this training requires a lot of computation time. Finally, it can be tough to classify big data, thus, if the dataset is too big, unsupervised learning is often a better choice.

4.2. Unsupervised Machine Learning

Unsupervised machine learning is used with unlabeled data. As suggested by its name, unsupervised machine learning is not supervised by the user to improve the model. The model will improve by itself as it will discover patterns and information from the dataset it was given. Usually, the algorithm will group the different data into categories that have the same similarities or by differences. Unsupervised learning is useful for big data analysis. As shown in Figure 3, unsupervised learning can be categorized as three types of problems clustering, association, and dimensionality reduction:

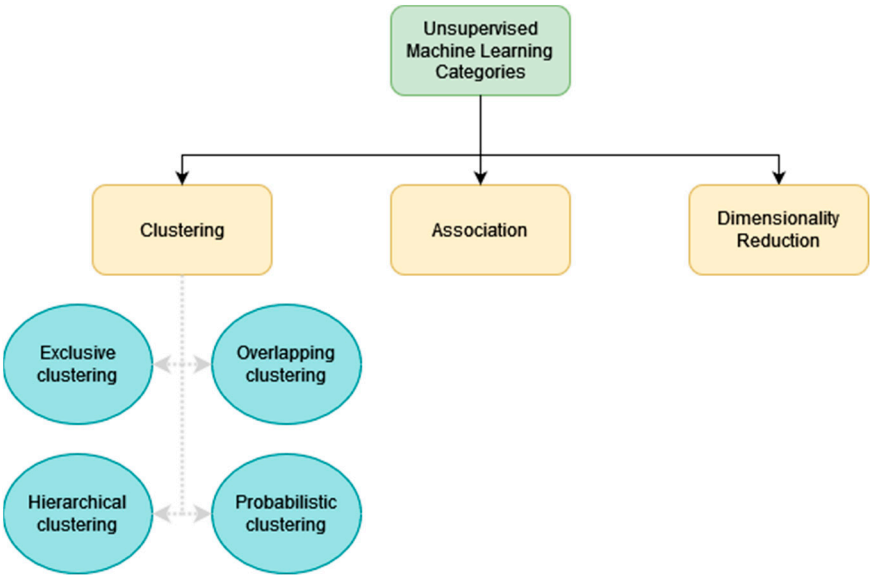

Figure 3. Unsupervised Machine Learning Categories.

- Clustering is a technique which groups unlabeled data according to their common or uncommon features [8]. At the end of this process, the unlabeled data will be sorted into different data groups. Clustering algorithms can be subcategorized as exclusive, overlapping, hierarchical and probabilistic [9]. With exclusive clustering, data that are grouped can only belong to one cluster only. Overlapping clustering allows data to belong to multiple clusters to have a richer model when data can belong to different categories. For instance, overlapping clustering is required for video analysis, since video can have multiple categories [10]. Hierarchical clustering is when the different isolated clusters are merged iteratively based on the similarity of the clusters until only one cluster is left. Different methods can be used to measure the similarities between two different clusters. The similarities between clusters are often measured as the distance between the clusters or their data. One of the most common metrics used is the Euclidean distance. The Manhattan distance is another metric often used [9]. With hierarchical clustering, it is possible to use the differences instead of the similarities to merge the clusters into a single cluster. It is known as Divisive clustering. With probabilistic clustering data points are put into clusters based on the probability that they belong to this cluster. Common clustering algorithms of the different clustering types are K-means, Fuzzy K-means, and Gaussian Mixture.
- Association is a method that finds relationships between the input data. Association is often used for marketing purposes to find the relationship between products and

thus, to propose other products based on customer purchases. A common association algorithm is the Apriori algorithm [11].

- Dimensionality reduction is a method used to reduce the number of features in a dataset to make it easier to process. Nowadays, due to the growing size of the dataset that is used, it is common to use a dimensionality reduction method before applying a specific machine learning algorithm to a dataset. There are two main methods to reduce the dimensionality of a dataset. The first one is feature elimination. It consists of removing features that will not be useful for the prediction that we want to make. The second one is feature extraction. Feature extraction will create the same number of features that already exist in the dataset. These new features are a combination of the old features. These features are independent and ordered by importance. Thus, we can remove the least important new features. The advantage of this approach is that we reduce the dataset while keeping the important part of each original feature. One of the most used methods for feature extraction is Principal Component Analysis (PCA) [12].

One of the main advantages of unsupervised learning is that it does not require human intervention. In addition, unsupervised learning helps to discover links or differences in large datasets quicker than manual analysis. Thus, unsupervised learning can be useful to find an unknown anomaly in large traffic if used to improve IDSs.

However, due to the high amount of data required to train the model, it requires high computational power and a lot of time. In addition, unsupervised learning has a higher risk of inaccurate results compared to supervised learning because it learns by itself. Thus, human intervention is often needed at the end of the training to verify whether the output variables are correct or not. This verification is also time-consuming.

4.3. Semi-Supervised Machine Learning

Finally, semi-supervised learning is the combination of both types of machine learning; supervised and unsupervised. With semi-supervised learning, only a part of the dataset is labelled. All the previous methods and techniques can be applied to this dataset.

One of the main advantages of semi-supervised learning is that it allows using techniques and algorithms from both machine learning types. Thus, new machine learning algorithms can be created to reach a better accuracy. In addition, semi-supervised learning is less time consuming since it does not need to use an entire set of labelled data. However, semi-supervised learning also has the disadvantages of both above techniques.

Table 2 summarises these ML methods.

Table 2. Summary of ML Methods—Advantages and Disadvantages.

ML Methods	Frequently Used Algorithms	Advantages	Disadvantages
Supervised ML Based on Known + unknown = Known (Labelled data used to generate a function that maps an input to an output).	SVM, Random Forest, Linear and Logistic Regression [6,7]	Uses labelled/trained data sets and can work with larger data sets. Works well in predictive environment/use-cases.	Takes a longer time to compute.
Unsupervised ML Based on Known + unknown = Unknown (Uses unlabeled data)	K-means, Apriori, Principal Component Analysis [9–11]	Uses unlabeled data sets, works well in analytical environment/use-cases	Takes less time to compute
Semi-Supervised ML (Mixed labelled and unlabelled)	Q-Learning, Deep Learning (DQN)	Is a combination of supervised and unsupervised learning and enables the predictive and analytical aspects of analysing data.	Time factor and computing complexity may depend on the combination of algorithms used

5. Datasets

To train and test their models, the researchers used datasets. Following is a discussion of the most known and used datasets for Intrusion Detection System training and testing.

5.1. KDDcup99

The KDDcup99 dataset has been one of the most widely used datasets to assess IDS. It is based on the DARPA'98 dataset. The KDDcup99 contains approximately 4,900,000 samples. Each sample has 41 features and is labelled as Normal or Attack. The attack samples are classified into four categories: Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L), and Probe. There are three different datasets for KDDcup99, the first one is the whole dataset, the second corresponds to 10% of the whole dataset the third one is a test dataset which contains 311,029 samples. One of the main disadvantages of this dataset is that it is imbalanced, i.e., many samples are like each other for major classes such as DoS and Probe whereas for R2L and U2R there are few. Depending on which part of the dataset is used some classes might be completely absent [13].

5.2. Kyoto 2006

This dataset was created by deploying honeypots, darknet sensors, email servers, web crawlers, and other network security measures outside and inside Kyoto University to collect various types of traffic. Based on the 41 features from the KDDcup99 dataset, they extracted 14 statistical features. In addition, they also extracted 10 additional features to form the dataset, thus, each sample has 24 features. The most recent version of the Kyoto dataset includes traffic from 2006 to 2015 [14].

5.3. NSL-KDD

This dataset was created to fix the main issue of the KDDcup99 dataset. It was proposed in 2009 by Tavallaee et al. [13]. It keeps the four attack categories of the KDDcup99. The NSL-KDD proposes two files, a training set, and a testing set. The training set is made of 21 different attacks and has 126,620 instances. The testing set is made of 37 different attacks and has 22,850 instances [14].

5.4. UNSW-NB15

This dataset was created by the Australian Centre for Cyber Security. It was created to generate traffic which is a hybrid of normal activities and attack behaviours. This dataset has nine types of attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms. The UNSW proposes two files, a training set, and a testing set. These files contain records from different types of traffic, attacks and normal, from the original dataset. The original dataset has a number of records of 2,540,044 while, the training set has 175,341 records, and the testing set has 82,332 records [15].

5.5. CICIDS2017

This dataset was created by the Canadian Institute for Cybersecurity (CIC) in 2017. This dataset was built using real-world traffic containing both normal and recent attack samples. The results were analyzed based on the time stamp, source, and destination IP, protocols, and attacks using CICFlowMeter. In addition, they implemented common attacks such as Brute Force FTP, Brute Force SSH, Denial of Service (DoS), HeartBleed, Web Attack, Infiltration, Botnet and Distributed Denial of Service (DDoS) [16].

A summary of the different datasets is given below in Table 3.

Table 3. Datasets summary.

Dataset	Year	Attack Types	Attacks
KDDcup99	1999	4	Probe, DoS, U2R, R2L
Kyoto 2006	2006	2	Known attacks, Unknown attacks
NSL-KDD	2009	4	Probe, DoS, U2R, R2L
UNSW-NB15	2015	7	Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms
CICIDS2017	2017	7	Brute Force, DoS, HeartBleed, Web attack, Infiltration, Botnet, DDoS

6. Literature Review

The solution proposed by Lirim et al. [17] used a Convolutional Neural Network (CNN) with a multi-layer perceptron for its model. Multi-layer perceptron can be considered as a fully connected network where a neuron corresponds to one layer and is connected to all neurons in the next layers. For neural networks, a CNN is composed of an input layer, hidden layers, and an output layer. Contrary to a traditional neural network, in one of its hidden layers, CNN uses a mathematical operation called a convolution instead of using a multiplication matrix. In CNN, the input is a tensor made of different parameters such as the number of inputs, the input height, width, and channels. The convolutional layer convolves the input and forwards the result to the next layer. However, the tensor size can grow tremendously after multiple convolutions. To tackle this issue, Lirim et al. [17] use padding to reduce the tensor dimension. They trained their model by optimizing the hyperparameters until a decrease in performance is met. Their final model uses ten classes (nine for attacks and one for normal traffic) and is made of multiple dual convolutional layers followed by a pooling and a dropout layer to avoid oversize. However, their model presents a class imbalance between the upper and bottom class which needs the use of bootstrapping to solve the issue. They tested their model on the pre-partitioned UNSW-NB15 dataset and on a user-defined dataset which corresponds to 30% of the whole dataset. They obtained, respectively an accuracy of 94.4% and 95.6% for both datasets.

Lin et al. [18] proposed another IDS based on CNN. Their solution is composed of two parts. The first one is offline training using CNN, where in their model, they start with an input layer of 9×9 and reduce it through successive convolutional layers and a maximum pooling layer to reach an output layer of 1×1. The second part of their system is the online detection phase, where they use Suricata, an open-source IDS, to catch the traffic. Then, the packets are pre-processed, and the trained model is used on the network traffic to produce the outcome of the detection. To test their model, they used the CICIDS2017 dataset. They tested it on the feature dataset and the raw traffic dataset. They obtained, respectively an accuracy of 96.55% and 99.56%, showing that their model is better with raw traffic than with an extracted feature set.

Rohit et al. [19] proposed an ensemble approach to detect intrusion. They perform three tests to show how their approach proposed better results. They first performed normalization on the KDD Cup99 dataset, then, they use a correlation method to perform feature selection. The feature selection used information gain as a decision factor, and finally, they use an ensemble approach combining three algorithms: Naïve Bayes, PART, and Adaptive Boost. The result of each algorithm is then compared, and the average of the results or most voting results is used to decide the outcome. In addition, they use the bagging method to reduce the variance error. They obtained an accuracy of 99.9732% on the KDD Cup99 dataset using their solution.

Al-Yaseen et al. [20], proposed a new model for intrusion detection systems, using a hybrid multi-level model combining SVM (Support Vector Machine) and ELM (Extreme Learning Machine). In their model there are five levels, the first level distinguishes the traffic into DoS or Other. The second level distinguishes the previous unknown traffic

into Probe or Other. The third distinguishes the previous unknown traffic into User to Root attack (U2R) or Other, and, the fourth level distinguishes the previous unknown traffic into Remote to Local attacks (R2L) or Other. Finally, the previous unknown traffic is distinguished between normal or unknown traffic in the fifth level. R2L and U2R are placed at the bottom level because they are similar to normal connections. At each level, a classifier is used. Their model is composed of 4 SVM classifiers at levels 1, 3, 4, and 5 and of 1 ELM classifier at level 2. They choose to use an ELM classifier to detect Probe because ELM has shown better results than SVM. After pre-processing the training set from the KDD dataset, they performed a modified K-means for feature extraction to have the 5 different categories that their solution can detect. Using their solution, they obtained an accuracy of 95.75%, which is slightly better than if they only used multi-level SVM (95.57%). In addition their hybrid model has a lower false alarm rate, 1.87%, compared to multi-level SVM at 2.17%.

Kanimozhi et al. [21], proposed a solution using oppositional tunicate fuzzy C-mean for detecting cloud intrusions. In their model, they first pre-processed the data and performed a normalization to have two datasets, one for training and one for testing. They performed a feature selection using logistic regression to keep the more relevant features, and they used the OPTSA and FCM clustering model. The dataset is split into C clusters using the fuzzy C-means algorithm. Once the data is clustered, they performed a cluster expansion and integration to reduce redundant clusters. They tested their solution on different datasets such as CICIDS2017 and obtained an accuracy of 80%.

Yiping et al. [22] created an intrusion detection system for wireless networks based on the random forest algorithm. They first created a signal detection model to catch the important features of signals, then, they created the model to detect malicious nonlinear scrambling intrusion signals. An improved random forest algorithm was used to extract the spectral features of the malicious signal, and then, optimal detection of malicious traffic in a wireless network was performed using a reinforcement learning method and static feature fusion. They obtained a mean accuracy of 96.93%.

Jabez et al. [23] created a system using an outlier detection approach to detect unknown attacks. The outlier detection approach is based on identifying data points that are isolated from clustered points. This approach uses the neighbourhood outlier factor to detect points that are not close to each other. They trialled their solution on the KDDcup99 datasets. In addition, the main advantage of their solution is its execution time which is significantly better compared to other solutions such as back propagation neural network which requires a lot of computing resources.

Kurniawan et al. [24] proposed an improved solution of Naïve Bayes for intrusion detection systems. The Naïve Bayes algorithm is based on the Bayes equation:

$$P(H|U) = \frac{P(U|H) * P(H)}{P(U)}$$

where:

- U is the data with an unknown class
- H is the hypothesis class of U
- $P()$ is the Probability

The Naïve Bayes algorithm has an issue when one of the probabilities is 0 [25]. This results in its low accuracy when used. In their solution, Kurniawan et al. proposed two modifications to the Naïve Bayes algorithm. The first one is removing each variable that has a probability of 0. The second modification is to change the multiplication operation by an addition operation when the probability is 0. In their solution, they first realized a feature selection using the correlation-based features selection (CFS). Thus, the number of features was reduced from 41 to 10 features. They tested their two modifications on the NSL-KDD dataset, the second modification showed promising results with an accuracy of 89.33%.

Gu et al. [26] proposed a new solution to improve IDS using SVM. Their solution used Naïve Bayes algorithm to perform feature selection. Then, they trained the model with the transformed data from the feature selection. They tested their solution on the UNSW-NB15 and the CICIDS2017 datasets. Compared to using only the SVM classifier, the use of Naïve Bayes for feature extraction before using the SVM classifier shows better results. Indeed, they obtained an accuracy of 93.75% on the UNSW-NB15 dataset and an accuracy of 98.92% on the CICIDS2017 dataset. However, their solution only shows if there is an intrusion, it cannot be used to detect what kind of attack is in operation.

Pan et al. [27] conceived a solution to detect intrusion in wireless networks. Their solution was based in the cloud to have the maximum efficiency in terms of computational power. They used sink nodes based in the fog to lessen the burden on the cloud computing section. In order to have a solution as light as possible, they used a combination of Polymorphic Mutation (PM) and Compact SCA (CSCA), as CSCA helps to reduce the computing load by reducing the density of the data by using probability. They added Polymorphic Mutation to reduce the loss of precision when using CSCA. They used PM-CSCA to optimize the parameters of KNN algorithms to have the best configuration. They tested their solution on the NSL-KDD and UNSW-NB15 datasets. They, respectively obtained an accuracy of 99.327% and 98.27%.

Xiao et al. [28], proposed a solution based on CNN. They first performed feature extraction using both Principal Component Analysis (PCA) and Auto-Encoder (AE). Auto-Encoder is a dimension reduction method using several hidden layers of neural networks to remove insignificant data. Then, they transformed the dimension of the data from one into a two-dimensional matrix and forwarded it to the CNN model to train it. The model is trained and improved using back propagation algorithms. They tested their model on the KDDcup99 and obtained an overall accuracy of 94%. They compared their model with DNN and RNN models and got slightly better results. However, their model has a low detection rate of U2R and R2L which are not represented enough in the dataset.

Zhang et al. [29], proposed a multi-layer model to detect attacks. Their solution combined two machine learning techniques: CNN and GcForest. The GcForest is a random forest technique which generates a cascade structure of decision trees. Their model is composed of two main parts. In the first part, they run a CNN algorithm to detect different kinds of attacks and normal traffic from the input data. Their CNN algorithm is an improved model of GoogLeNet called GoogLeNetNP. The second part consists of using a deep forest model to create more subclasses of the attacks. This second layer improves the precision of their solution by classifying the abnormal classes into N-1 subclasses. The second layer uses the cascade principle of gcForest but instead of the random forest, it uses XGBoost. XGBoost is like a random forest, however, the construction of the trees is done one after another until the objective function is optimized. They tested their solution on a combination of the UNSW-NB15 and CICIDS2017 datasets. They obtained an overall accuracy of 99.24% which is better compared to the algorithms used singularly.

Yu et al. [30], proposed an IDS model based on Few-Shot Learning (FSL). FSL is a deep learning method that can learn from a small amount of data. In their solution they used two embedding models, CNN and DNN, to perform feature extraction. Those models help to reduce the dimension of the input data without losing important information. They tested their model on the UNSW-NB15 and NSL-KDD datasets. Their solution obtained, respectively an accuracy of 92.34% and 92%.

Gao et al. [31], proposed an ensemble machine learning IDS. They used the Principal Component Analysis method for feature extraction. After different tests on the NSL-KDD datasets, their ensemble algorithm is combining Decision Tree, Random Forest, KNN, DNN and MultiTree. The results of the ensemble algorithm are made by a majority vote using weights for each algorithm to have better accuracy. They obtained an accuracy of 85.2% which is better than the accuracy if they were only using one algorithm. However, their model lacks efficiency when analyzing attacks that are not in large quantity.

Marir et al. [32], proposed a solution using a Deep Belief Network (DBN) and an ensemble method composed of multiple SVMs. A DBN is a succession of unsupervised networks such as Restricted Boltzmann Machines (RBM). An RBM is composed of an input and a hidden layer where the nodes are connected to the previous and next layers but are not connected within their layer. DBN uses an unsupervised pre-training based on the greedy layer-wise structure. Then, they use a supervised fine-tuning approach to learn the important features. In their solution, they use DBN for feature extraction. Then, the extracted features are forwarded to the multi-layer ensemble SVM. The output is generated by a voting algorithm. They tested their solution on KDDcup99, NSL-KDD, UNSW-NB15, and CICIDS2017 datasets. They, respectively obtained a precision of 94.76%, 97.27%, 90.47% and 90.40%. However, it was shown that when more layers are used their solution is more time consuming.

Wei et al. [33], improved the performance of DBN for IDS by using an optimizing algorithm. To optimize their model, they used a combination of Particle Swarm Optimization (PSO), Artificial Fish Swam Algorithm (AFSA), and Genetic Algorithm (GA). The PSO is first optimized using AFSA. Then, GA is used to find the global optimal solution of the initial particle search. The optimal solution is then used in the DBN model to improve its accuracy. They tested their solution on the NSL-KDD dataset and obtained an accuracy of 82.36%.

Vinayakumar et al. [34], proposed an IDS based on Deep Neural Network (DNN). Their DNN architecture is made of an input layer, five hidden layers and an output layer. Their solution is scalable, and it is possible to use between one and five hidden layers in the DNN models. They used the Apache Spark computing platform. Their solution can work in both cases, HIDS and NIDS. For NIDS, they tested their solution on KDDcup99, NSL-KDD, Kyoto, UNSW-NB15 and CICIDS2017 datasets. They, respectively obtained an overall accuracy of 93%, 79.42%, 87.78%, 76.48% and 94.5% when combining the accuracy for each number of DNN layers.

Shone et al. [35] proposed a solution combining Non-symmetric Deep Auto-Encoder (NDAE) and Random Forest. Usually, an auto-encoder uses the symmetric scheme from encoder-decoder, however, in their solution, they only used the encoding phase. It reduces the computational time without impacting too much on the accuracy of the IDS. To handle complex datasets, they choose to stack their NDAE. However, they discovered that using only NDAE was not enough to have an accurate classification. Therefore, they added Random Forest as their classifier after performing feature extraction using two NDAE with three hidden layers each. They tested their solution on the KDDcup99 and NSL-KDD datasets and compared it to a DBN solution. They obtained, respectively a total accuracy of 97.85% and 85.42%. However, their solution struggles to detect small classes such as R2L and U2R.

Yan et al. [36], showed the impact of feature extraction using a Stacked Sparse Auto-Encoder (SSAE) to improve IDS. A sparse auto-encoder is an autoencoder which uses a sparsity penalty, usually, the penalty is activated when hidden nodes are used. Thus, using a sparse auto-encoder reduces the number of hidden nodes used. Stacked sparse auto-encoder is the addition of further sparse auto-encoders. It allows for reducing the dimension of the input data without losing significant information. To optimize their SSAE they used the error back propagation method, and, to test their SSAE model they used the NSL-KDD dataset. They used different classifiers with and without their SSAE model to show how much the use of SSAE for feature extraction improves the accuracy. The best accuracy was obtained when the SSAE and SVM classifiers were combined. They reached an overall accuracy of 99.35%. One of the main advantages of using their solution is the large time reduction for training and testing, approximately a tenth of the time of other solutions is needed. However, the detection rate for R2L and U2R is lower compared to the other classes.

Khan et al. [37], proposed a two-stage deep learning model (TSDL) to improve IDS. In the first stage, they classify the traffic as normal or abnormal with a probability value.

In the second stage they used this value as an additional feature to train the classifier, they used a DNN approach for both stages, where they used a Deep stacked auto-encoder (DSAE) for feature extraction and Soft-max as a classifier. Soft-max is often used in a neural network for multi-class classification problems. They tested their solution on the KDDcup99 and UNSW-NB15 datasets. They, respectively obtained an overall accuracy of 99.996% and 89.134%.

Andresini et al. [38], proposed a solution combining an unsupervised approach with two auto-encoders and a supervised stage to build the datasets. They trained the two auto-encoders separately using normal and attack traffic. Then, the auto-encoders reconstruct those samples and add them to the dataset that is used to train the model. The dataset goes through a one-dimension CNN. This is done to see the impact of one channel on the other to have a better distinction between the two classes: normal and attack. Finally, they used a Soft-max classifier to identify if the data was an attack or normal. They tested their model on KDDcup99, UNSW-NB15 and CICIDS2017 datasets. They, respectively obtained an overall accuracy of 92.49%, 93.40% and 97.90%. One of the drawbacks of their solution is that it does not provide details about the different types of attacks.

Ali et al. [39], proposed a model using Fast Learning Network (FLN) based on particle swarm optimization (PSO). They used PSO to improve the accuracy of FLN which can be inefficient due to the weights used in the neural network. They tested their solution on the KDDcup99 dataset against other FLN solutions. They obtained a better accuracy to detect the different classes than the other solutions. They achieved an overall accuracy of 89.23%. However, their overall accuracy is decreased by their low accuracy when identifying one of the small classes of attack (R2L).

Dong et al. [40], proposed a hybrid solution combining clustering with SVM. In their solution, they first used K-means clustering to process the data and divided it into different subsets. Then, they used SVM on each of those subsets. They tested their solution on the NSL-KDD datasets and they obtained an overall accuracy of 99.45%. In addition, compared to other methods their solution improved the detection rate. Their solution also requires less time processing compared to SVM algorithms using different parameters. However, the authors provided no information concerning the accuracy of each attack classification.

Wisanwanichthan et al. [41], proposed a Double-Layered Hybrid Approach (DLHA). In their solution, they first create two groups in the NSL-KDD dataset. The first one contains all classes and the second one contains only the U2R, R2L and normal classes. They created these two groups to have better accuracy of the U2R and R2L classes which are often the weakness of most of the IDS solutions that we have seen. Then, they performed feature extraction in both groups. They first used Intersectional Correlated Feature Selection (ICFS). In ICFS, the Pearson Correlation Coefficient (PCC) is used to select important features between two random variables. PCC can determine how much two variables vary from each other. Once ICFS is done, they performed Principal Component Analysis (PCA) to reduce the dimension of the data. Finally, to have a ratio of 1:1 between attacks and normal data in the second group they randomly choose the same amount of data as R2L and U2R combined. Then, they used those two groups to train their model which is composed of a first layer using Naïve Bayes classifier and a second layer using SVM. The first layer is used only to detect DoS and Probe. If the outcome is not one of those two classes, then the data goes through the second layer to detect if it is a R2L, U2R or Normal data. They tested their solution on the NSL-KDD dataset. They obtained an overall accuracy of 93.11% and detection of 96.67% for R2L and 100% for U2R classes. Their solution outperformed other solutions when identifying the small classes, however, contrary to other efficient solutions, their accuracy for the large classes was not as good.

Elhefnawy et al. [42] proposed a Hybrid Nested Genetic-Fuzzy Algorithm (HNGFA) to detect attacks. They first performed feature selection using Naïve Bayes. Major features and Minor features are split into two groups. Their model is composed of two genetic-fuzzy algorithms. The first one is the Outer Genetic-Fuzzy Algorithm (OGFA) and the second one is the Inner Genetic-Fuzzy Algorithm (IGFA). Each of these algorithms used two nested

genetic algorithms. The outer one is used for the fuzzy sets and the inner one is used for the fuzzy rules. The OGFA is used for classifying data with major features, whereas the IGFA is used for classifying data with minor features. The two genetic-fuzzy algorithms interact with each other to develop new solutions to have better accuracy. The goal is to make the interaction between the best results of the OGFA with weak results from the IGFA to have the best model possible. They tested their solution on the KDDcup99 and UNSW-NB15 datasets and they obtained an overall accuracy of 98.19% and 80.54%, respectively. In addition, their solution got a good accuracy for detecting small classes such as R2L and U2R. However, due to the complexity of their model, the training time is high.

A summary of this study is provided in Table 4. In addition, the strengths, and weaknesses of each solution are presented in Table 5.

Table 4. Summary of Literature Review (Feature extraction method and Classifier).

Authors	Year	Feature Extraction Method Used	Classifier Used	Attack Detected
Jabez et al. [23]	2015	NA	Outlier detection	Network attacks
Al-Yaseen et al. [20]	2017	Modified K-means	Multi-level Hybrid SVM and ELM	DoS, User to Root (U2R) and Remote to Local (R2L) attacks
Rohit et al. [19]	2018	Correlation method	Ensemble method (Naïve Bayes, PART, and Adaptive Boost)	DoS, Probe, U2R, R2L
Marir et al. [32]	2018	Deep Belief Network	Multi-layer ensemble method SVM	DoS, U2R, R2, Probe, Fuzzers, Analysis, Backdoors, Exploits, Generic, Reconnaissance, Shellcode, Worms, Brute Force FTP, Brute Force SSH, Heartbleed, Web Attack, Infiltration, Botnet and DDoS
Shone et al. [35]	2018	Non-symmetric Deep Auto-Encoder (NDAE)	Random Forest	DoS (back, land, Neptune), Probe (ipsweep, nmap, portsweep, satan), R2L (ftp_write, guess_password, imap, multihop, phf, spy, warezclient, warezmaster), U2R (loadmodule, buffer_overflow, rootkit, perl)
Yan et al. [36]	2018	Stacked Sparse Auto-Encoder (SSAE)	SVM	DoS, Probe, R2L, U2R
Ali et al. [39]	2018	NA	Fast Learning Network improved using PSO	DoS, U2R, R2L and Probing
Xiao et al. [28]	2019	PCA and Auto-Encoder	CNN	DoS, U2R (illegal Access from remote machines), R2L (illegal access to local super user privileges), probe (supervisory and detection).
Zhang et al. [29]	2019	NA	CNN with improved gcForest	DoS, Exploits, Generic, Reconnaissance, Virus and Web attacks
Gao et al. [31]	2019	PCA	Ensemble method (DT, RF, KNN, DNN and MultiTree)	DoS (SYN flood), Probe (port scanning), R2L (guessing password), U2R (buffer overflow attacks)

Table 4. *Cont.*

Authors	Year	Feature Extraction Method Used	Classifier Used	Attack Detected
Wei et al. [33]	2019	NA	DBN improved using optimizing algorithm (PSO-AFSA-GA)	Analysed 39 types of attacks that fall under the following categorises: Probe (scan and probe), DoS, U2R (illegal access to local superuser) and R2L (unauthorized remote access)
Vinayakumar et al. [34]	2019	NA	DNN with scalable hidden layers	Normal, DoS, Probe, R2L, U2R
Khan et al. [37]	2019	Deep Stack Auto-Encoder (DSAE)	Soft-max	Normal, DoS, Probe, R2L, U2R (22 different categories of attacked tested, i.e., analysis, backdoor, exploits, fuzzers, generic, reconnaissance, shellcode, worm)
Dong et al. [40]	2019	NA	K-mean clustering with SVM	-
Lin et al. [18]	2020	NA	CNN	FTP Brute Force, SSH Brute Force, DoS (slowloris, slowtptest, Hulk), Web attacks (web brute force, XSS, SQL injection), penetration attacks (infiltration Dropbox download)
Yu et al. [30]	2020	Embedded function using CNN and DNN	Few-Shot Learning	DoS (Teardrop, Smurf), Probe (Satan, Portsweep, saint), U2R (Rootkit, Buffer_overflow, Loadmodule) and R2L (Xsnoop, Httptunnel). Other attack types tested were: normal, generic, fuzzers, reconnaissance, shellcode, worms, backdoor and exploits)
Andresini et al. [38]	2020	Dual Auto-Encoder	Soft-max	-
Elhefnawy et al. [42]	2020	Naïve Bayes	Hybrid Nested Genetic Fuzzy Algorithm	Probe, DoS, U2R, and R2L. Other attack types tested were: normal, generic, fuzzers, reconnaissance, shellcode, worms, backdoor and exploits)
Lirim et al. [17]	2021	NA	CNN with multi-layer perceptron	DoS, DDoS, PortScan, Web Attack, Heartbleed, Benign, Infiltration, Brute Force, SSH, FTP
Kanimozhi et al. [21]	2021	Logistic Regression	Oppositional tunicate fuzzy C-mean	-
Kurniawan et al. [24]	2021	Correlation-based features selection	Modified Naïve Bayes	Normal, DoS, Probe, R2L, U2R
Gu et al. [26]	2021	Naïve Bayes	SVM	-
Pan et al. [27]	2021	NA	KNN using PM-CSCA for optimization	DoS, Sniffing (Probe), U2R and R2L

Table 4. *Cont.*

Authors	Year	Feature Extraction Method Used	Classifier Used	Attack Detected
Wisanwanichthan et al. [41]	2021	ICFS and PCA	Naïve Bayes and SVM	Employed Double Layered Hybrid Approach (DLHA) for detecting DoS, Probe, R2L and U2R
Yiping et al. [22]	2022	NA	Improved random forest algorithm	Wireless network attacks

Table 5. Summary of Literature Review (IDS model advantages and disadvantages).

Authors	Advantages	Disadvantages
Jabez et al. [23] (2015)	An outlier detection approach for IDS. The neighborhood outlier factor is used to detect isolated points. Compared to back propagation neural network the execution time is significantly better.	Old dataset KDDcup99 was used, and no information was given on the accuracy for the different types of attacks.
Al-Yaseen et al. [20] (2017)	A Hybrid multi-level model merging SVM and ELM. The model is divided into 5 levels, with each level responsible to detect one categories of attack. In addition, they used Modified K-means for feature extraction. They achieved a better overall accuracy compared to multi-level SVM and ELM.	Old dataset used for the model. The accuracy for R2L and U2R attacks is very low.
Rohit et al. [19] (2018)	An ensemble approach combining three algorithms: Naïve Bayes, PART, and Adaptive Boost with an average voting result to decide the outcome. Feature extraction methods are used.	The dataset used, KDDcup99, is very old. There is no description on how well the model can classified the different categories of attacks.
Marir et al. [32] (2018)	Use of a multi-layer ensemble SVM to detect abnormal traffic. DBN was used for feature extraction before forwarding the result to the ensemble SVM and the voting algorithm. They tested their solution on old and recent datasets KDDcup99, NSL-KDD, UNSW-NB15, and CICIDS2017.	When more layers are used, the model is more time consuming. Lack of information on the different attacks classes that are identified.
Shone et al. [35] (2018)	A Non-symmetric Deep Auto-Encoder (NDAE) and random forest model to detect abnormal behavior. They used two NDAE that they stacked for feature extraction. Compared to a DBN solution they obtained good accuracy.	Their solution struggles to detect small classes such as R2L and U2R on old datasets.
Yan et al. [36] (2018)	A NIDS using Stacked Sparse Auto-Encoder (SSAE) for feature extraction and SVM as classifier. Their solution reduces significantly the time required for training and testing	Old dataset NSL-KDD is used, and their model has a low detection rate of U2R and R2L attacks.
Ali et al. [39] (2018)	Use of Fast Learning Network (FLN) and particle swarm optimization to detect abnormal traffic.	Model is tested on old KDDcup99 dataset and has low accuracy when identifying one of the small classes.
Xiao et al. [28] (2019)	CNN model for IDS using PCA and Auto-Encoder for feature extraction.	Old dataset KDDcup99 is used, and their model has a low detection rate of U2R and R2L attacks.
Zhang et al. [29] (2019)	A multi-layer model combining random forest technique with CNN. They achieved an outstanding accuracy of 99.24% on a combination of two recent datasets UNSW-NB15 and CICIDS2017 compared to the single algorithms. The accuracy on each attack class is given.	Lack of feature extraction method.

Table 5. *Cont.*

Authors	Advantages	Disadvantages
Gao et al. [31] (2019)	An ensemble machine learning IDS combining Decision Tree, Random Forest, KNN, DNN and MultiTree. Weights are used to have a better accuracy during the voting process. PCA was used for feature extraction.	Model lacks in efficiency when analyzing attack that are not in large quantity. Old-fashioned dataset NSL-KDD was used.
Wei et al. [33] (2019)	An improved DBN model using a combination of Particle Swarm, Fish Swarm and Genetic algorithm.	Model is tested on old NSL-KDD dataset. Lack of feature extraction method.
Vinayakumar et al. [34] (2019)	A hybrid IDS using DNN. They used old and new datasets to test their model.	Lack of feature extraction method. In addition, the model is complex and has low detection rate for weaker attack classes.
Khan et al. [37] (2019)	A two-stage deep learning model (TSDL) using a DNN approach to detect abnormal traffic. The first stage is used to classify the traffic with a given probability. The second stage used this probability as an additional feature to improve the classification results.	They tested their solution on KDDcup99 and UNSW-NB15 datasets. However, even if they achieved a 99.996% on the old dataset there is a 10% gap compared to the accuracy with the recent dataset. Thus, the model still needs to be improved to reduce this gap.
Dong et al. [40] (2019)	Hybrid solution combining clustering with SVM. K-means is used to divide the data in different subsets and SVM method is performed on each of those subsets. Their solution requires less time processing compared to SVM algorithms using different parameters.	The solution uses old dataset, and no information was provided for the accuracy of each class of attack.
Lin et al. [18] (2020)	A NIDS using CNN which achieves excellent results on one of the most recent datasets, the CICIDS2017.	There are no accuracy details on each kind of attacks. No feature extraction methods.
Yu et al. [30] (2020)	IDS based on Few-Shot Learning which gives good accuracy results using only a small portion of the datasets. DNN and CNN were used to perform feature extraction. Their solution was tested on an old dataset NSL-KDD and a recent dataset UNSW-NB15.	Although the detection rates for U2R and R2L is not bad compared to other methods, there is still a lot of room for improvement.
Andresini et al. [38] (2020)	An IDS based on two auto-encoders for feature extraction and using a soft-max classifier to detect intrusions. The model is tested on an old and a recent dataset.	The solution does not provided details about the different types of attacks.
Elhefnawy et al. [42] (2020)	A Hybrid Nested Genetic-Fuzzy Algorithm (HNGFA) to detect attacks. Feature selection is performed using Naïve Bayes. The model combines two genetic-fuzzy algorithms that use the results of each other to improve the accuracy. Also, they tested their solution on an old and a recent dataset.	The model is complex and requires a high training time.
Lirim et al. [17] (2012)	A NIDS based on CNN with multi-layer perceptron that achieves good accuracy on recent dataset.	The model has a class imbalance between the upper and bottom class. Lack of feature reduction methods.
Kanimozhi et al. [21] (2021)	An innovative method using oppositional tunicate fuzzy C-mean for cloud intrusion. Logistic regression is used for feature selection to improve the accuracy. The model is tested on a recent dataset, the CICIDS2017.	The proposed model is complex and only achieved an overall accuracy of 80%.
Kurniawan et al. [24] (2021)	A NIDS using a modified Naïve Bayes model. Feature selection is used to improve the model.	Old dataset NSL-KDD used. No detailed accuracy for the different classes of attacks is given.

Table 5. *Cont.*

Authors	Advantages	Disadvantages
Gu et al. [26] (2021)	An SVM model using Naïve Bayes for feature selection with great accuracy results on two recent datasets UNSW-NB15 and CICIDS2017.	Their solution does not identify what kind of attack is used.
Pan et al. [27] (2021)	KNN model using PM-CSCA for optimization that achieves great accuracy on an old dataset NSL-KDD and a recent dataset UNSW-NB15. Their solution is used for wireless network and is based on the cloud to reduce the time and speed required by the algorithm.	No identification of the attacks classes is provided.
Wisanwanichthan et al. [41] (2021)	An IDS using a Double-Layered Hybrid Approach (DLHA). They first used Intersectional Correlated Feature Selection (ICFS) and PCA to select important features. They then split the NSL-KDD dataset into two groups, one with all the classes and one with only the U2R, R2L and normal classes. These two groups are used to train the model which is composed of a first layer using Naïve Bayes classifier and a second layer using SVM. Their solution achieved outstanding accuracy for R2L and U2R with respective accuracy of 96.67% and 100%.	Although their solution outperformed other solutions when identifying small classes, the accuracy for large classes could be improved.
Yiping et al. [22] (2022)	A random forest model for a wireless network.	The mean accuracy obtained is 96.93%. No known datasets were used.

7. Evaluation Metrics

The datasets used in the different papers analyzed in Section 6, the main outcome for their solution is given in Table 6. The evaluation metrics used in the literature review (Section 6) to assess their solution is given in Table 7.

Table 6. Outcome/Accuracy.

Authors	Database Used	Outcome/Accuracy
Jabez et al. [23]	KDDcup99	NA
Al-Yaseen et al. [20]	KDDcup99	95.75%
Rohit et al. [19]	KDDcup99	99.97%
Marir et al. [32]	KDDcup99 NSL-KDD UNSW-NB15 CICIDS2017	94.76% (KDDcup99) 97.27% (NSL-KDD) 90.47% (UNSW-NB15) 90.40% (CICIDS2017)
Shone et al. [35]	KDDcup99 NSL-KDD	97.85% (KDDcup99) 85.42% (NSL-KDD)
Yan et al. [36]	NSL-KDD	99.35%
Ali et al. [39]	KDDcup99	89.23%
Xiao et al. [28]	KDDcup99	94%
Zhang et al. [29]	Combination of UNSW-NB15 and CICIDS2017	99.24%
Gao et al. [31]	NSL-KDD	85.20%
Wei et al. [33]	NSL-KDD	82.36%

Table 6. *Cont.*

Authors	Database Used	Outcome/Accuracy
Vinayakumar et al. [34]	KDDcup99 NSL-KDD Kyoto UNSW-NB15 CICIDS2017	93% (KDDcup99) 79.42% (NSL-KDD) 87.78 % (Kyoto) 76.48% (UNSW-NB15) 94.5% (CICIDS2017)
Khan et al. [37]	KDDcup99 UNSW-NB15	99.996% (KDDcup99) 89.134% (UNSW-NB15)
Dong et al. [40]	NSL-KDD	99.45%
Lin et al. [18]	CICIDS2017	99.56%
Yu et al. [30]	NSL-KDD UNSW-NB15	92.34% (NSL-KDD) 92% (UNSW-NB15)
Andresini et al. [38]	KDDcup99 UNSW-NB15 CICIDS2017	92.49% (KDDcup99) 93.40% (UNSW-NB15) 97.90% (CICIDS2017)
Elhefnawy et al. [42]	KDDcup99 UNSW-NB15	98.19% (KDDcup99) 80.54% (UNSW-NB15)
Lirim et al. [17]	UNSW-NB15	95.60%
Kanimozhi et al. [21]	CICIDS2017	80%
Kurniawan et al. [24]	NSL-KDD	89.33%
Gu et al. [26]	CICIDS2017 UNSW-NB15	98.92% (CICIDS2017) 93.75% (UNSW-NB15)
Pan et al. [27]	NSL-KDD UNSW-NB15	99.33% (NSL-KDD) 98.27% (UNSW-NB15)
Wisanwanichthan et al. [41]	NSL-KDD	93.11%
Yiping et al. [22]	NA	96.93%

Table 7. Evaluation Metrics.

Authors	Evaluation Metrics						
	Accuracy	Precision	Detection Rate	F-Measure	False Alarm Rate	ROC Curves	Others
Jabez et al. [23]							✓
Al-Yaseen et al. [20]	✓		✓		✓	✓	
Rohit et al. [19]	✓	✓	✓				
Marir et al. [32]		✓	✓	✓		✓	✓
Shone et al. [35]	✓	✓	✓	✓	✓		✓
Yan et al. [36]	✓		✓		✓		✓
Ali et al. [39]	✓						
Xiao et al. [28]	✓		✓		✓		✓
Zhang et al. [29]	✓		✓		✓		
Gao et al. [31]	✓	✓	✓	✓			
Wei et al. [33]	✓		✓		✓		✓
Vinayakumar et al. [34]	✓	✓	✓	✓	✓	✓	
Khan et al. [37]	✓	✓	✓	✓	✓		
Dong et al. [40]	✓		✓		✓		

Table 7. *Cont.*

Authors	Evaluation Metrics						
	Accuracy	**Precision**	**Detection Rate**	**F-Measure**	**False Alarm Rate**	**ROC Curves**	**Others**
Lin et al. [18]	✓						
Yu et al. [30]	✓	✓	✓	✓	✓		
Andresini et al. [38]	✓			✓			
Elhefnawy et al. [42]	✓	✓	✓	✓	✓		
Lirim et al. [17]	✓		✓				
Kanimozhi et al. [21]	✓		✓		✓		
Kurniawan et al. [24]	✓	✓	✓				
Gu et al. [26]	✓		✓		✓		
Pan et al. [27]	✓		✓		✓	✓	
Wisanwanichthan et al. [41]	✓	✓	✓	✓	✓		
Yiping et al. [22]	✓					✓	

Metrics included in previously published papers are marked as ✓.

8. Discussion

8.1. Observations

This study shows that the impact of machine learning is significant to improve the efficiency of IDSs, and that the quality of the dataset is an important factor that determines how efficient an IDS will be. Using well-constructed datasets is necessary and in many of the research papers reviewed here, they used labelled data to improve the training of the model. However, nowadays the size of datasets is growing and previous research shows that machine learning models are often not suitable when the dataset size grows too large. Thus, deep learning models, such as CNN are being adopted more and more by researchers to develop new solutions. These new methods learn and extract useful features from raw datasets to make NIDS efficient against zero-day attacks. In addition, NIDS need to be trained frequently with up-to-date data from real networks. However, these new solutions have a cost. They require more powerful computing resources and more time processing to train an efficient model.

Table 5 shows the main advantages and weaknesses from the different reviewed solutions. The first observation that can be done is that traditional machine learning techniques such as clustering are less popular compared to deep learning techniques. Indeed, it was shown that in most cases using deep learning techniques such as neural networks significantly improved the accuracy of the IDS. In addition, to tackle the need for more computing power, the researchers use GPUs and cloud-based platforms as they help to implement more powerful deep learning methods. One of the main drawbacks that was observed in this study, is that most of the existing solutions used old datasets such as KDDcup99 and NSL-KDD to test their model. Those datasets can be out of date and not accurate with regard to real-world network traffic. In addition, it was observed that for some solutions, when recent data sets were used the accuracy of the system decreased compared to older datasets where the accuracy was excellent. Another point of concern related to most of the solutions studied is their inefficiency to detect specific attack classes that have fewer samples in their dataset. This is mainly due to a class imbalance in the dataset, which results in a lower detection rate for those classes. This is a serious issue as those minor classes could be zero-day attacks. We also observed that there is a trade-off between the complexity of the model and the number of layers used in deep learning models. The more layers used in the algorithm the more complex the model will be, and therefore, the model will require more computing resources and time. Efficient filtering

and selection of the important features for the model training helps to decrease the amount of time and resources needed.

Figure 4 shows the number of times each classifier was used in the literature review given in Section 6. It can clearly be seen that the researchers are using more and more deep learning techniques such as CNN or DNN. We can also see that many researchers are using SVM as their classifier because it has great performance when detecting minor classes. Thus, they often combine it with other machine learning methods in a multi-layer manner. Additionally, ensemble methods are frequently used as it allows the use of different methods together to improve the efficiency of the IDS. The advent of these new powerful techniques is possible, thanks to the improvement in GPU utilization and cloud computing platforms. Finally, old traditional machine learning methods such as K nearest neighbourhood, K-mean clustering, and genetic algorithms are less used as compared to CNN and DNN.

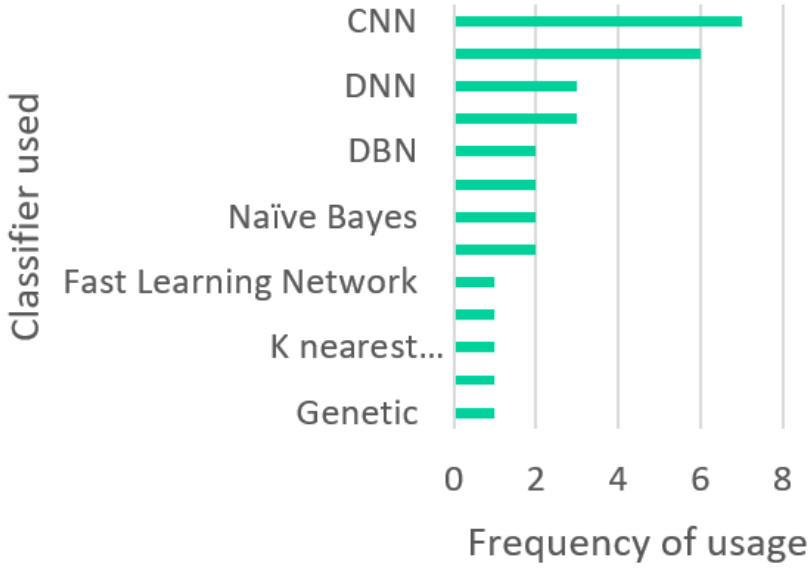

Figure 4. Frequency of Classifier used (Section 6).

In Section 6, it is shown that more and more researchers are using feature extraction in their solutions. Indeed, as shown in Figure 5, 60% of the reviewed articles in Section 6 used feature extraction. It was clearly shown that feature extraction greatly helps to improve the accuracy of the system. In addition, it helps to improve the structure of the dataset so that the amount of computing resources needed is decreased. In most cases, the feature extraction methods were based on neural network techniques such as Auto-Encoder techniques. However, many solutions also used traditional machine learning methods for feature extraction such as Naïve Bayes or Correlation method. Thus, it was shown that old machine learning techniques and new deep learning methods can be combined, with one of them used for feature selection and the other one used for classifying traffic.

The analysis of the metrics used to assess the various solutions studied is shown in Figure 6. The most used metrics are the Accuracy and the Detection rate (Recall). Those metrics are indeed the most important to assess the quality of a solution. Thus, they should always be used when assessing the efficiency of an IDS. Nevertheless, we consider that the F-measure should also be used more when assessing an IDS because it shows if the solution is efficient to detect samples that are indeed attacks even from minor classes.

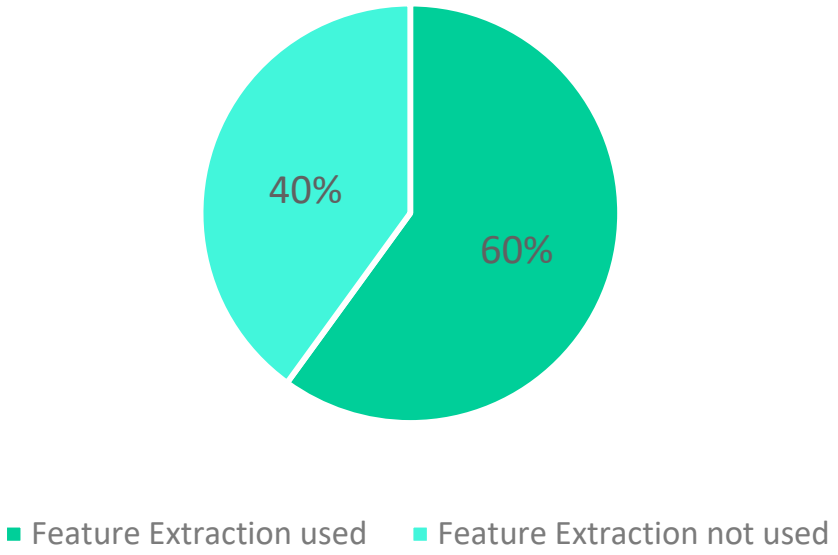

Figure 5. Feature extraction distribution (from Section 6).

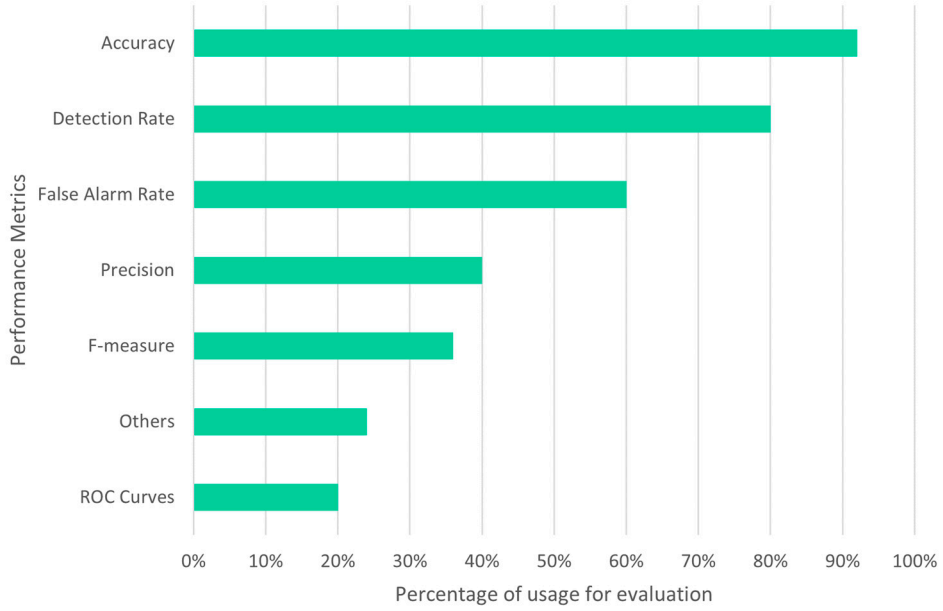

Figure 6. Metrics used (Section 6).

Datasets are a key component of IDSs. The analysis of which public datasets is used for testing is shown in Figure 7. It is shown that the KDDcup99 and NSL-DD were used 56% of the time for testing. Those datasets are old but still very popular with researchers since many studies have been made using them. However, the network structure and architecture have changed compared to 20 years ago. Indeed, nowadays IoT devices and wireless device usage has significantly increased. Thus, the amount of data exchanged in the world is incomparable with what existed 20 years ago, in addition, many novel and powerful attacks have since been created. If a new IDS solution is trained on an old dataset only, it is likely that the solution will not be efficient when utilised in a real-world environment. Thus, this study shows the need of using recent datasets to have better trained models that will be efficient in real-world modern networks.

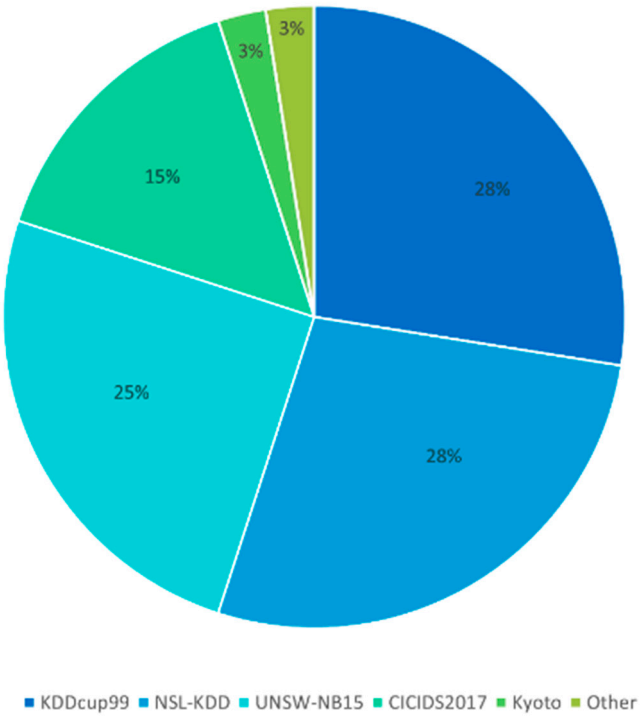

Figure 7. Public datasets percentage of use (Section 6).

8.2. Research Challenges

This subSection highlights the main research challenges for IDS.

8.2.1. Unavailability of Recent Dataset

The study highlighted the lack of recent datasets that represent new attacks that are used against our networks. Most of the solutions studied were not able to detect zero-day attacks because the models were not sufficiently trained with enough diverse attacks. To have a powerful and efficient IDS, it is needed to train it with a large dataset that includes the most recent attacks. The dataset needs to have both old and new attacks so that the IDS would be able to learn the important features of each attack to detect them. Therefore, one of the challenges for researchers is to have an up-to-date dataset that would contain enough samples of all types of attacks. In addition, the dataset should be frequently updated with new intrusions samples from the real-world environment.

8.2.2. Lower Detection Accuracy for Minor Classes

It was also shown in the study, that most of the solutions available are inefficient in detecting minor classes even if they have a very good overall accuracy to detect abnormal behaviour. This problem comes from imbalanced datasets. This results in an accuracy that is lower for minor classes than for major classes. One way to solve the problem is by using an up-to-date dataset that contains enough instances of minor classes. However, it was shown that such a dataset still needs to be made. Another solution would be to use feature extraction methods, so that the datasets can be split into two parts, one with the major classes and one with the minor classes. Then, using a multi-level structure and different machine learning techniques the model can be trained to have good results to detect minor classes.

8.2.3. Low Performance in a Real-World Environment

Another challenge for IDS is their lack of testing in real-world environments. Indeed, most of the studied solutions were tested using available datasets and those datasets were old and are not reflective of modern network traffic. In addition, none of the solutions

were tested using real-world data. Thus, it is not certain if any of these solutions will perform well in a real-world environment. Therefore, one of the main challenges going forward is to ensure that future solutions will be tested in a real-world environment to check their efficacy.

8.2.4. Resources Consumed by Complex Models

As shown in this study in Section 6, most IDSs are very complex and require a lot of computing resources and time. This requirement may affect the efficiency of IDS in a real environment. The use of multi-core GPUs can be a solution to reduce the time required, however, this solution is costly. Therefore, the algorithms developed need to use feature extraction to select the most important features to monitor to speed up the processing. However, as shown in Figure 5, 40% of the solutions reviewed were not using feature selection. Future solutions should try to find new feature extraction methods to reduce the processing of their algorithm.

8.2.5. IDS for IoT

There is a need to develop IDS specifically for IoT. One of the main challenges with IDS for IoT is that IoT devices generally use wireless networks. Most of these devices contain sensor nodes and they collect a tremendous amount of data. However, most of these nodes have limited computing power. Thus, to implement IDS for IoT, the IDS needs to be lightweight, require less computing power and utilize a smaller amount of data to detect threats. However, most of the reviewed papers in this study proposed solutions for wired networks, therefore, one of the biggest challenges for researchers will be to develop lightweight IDS that would have a high detection rate in a wireless environment.

8.3. Future Trends

8.3.1. Efficient NIDS

Recent studies show the limitation of NIDS to detect zero-day attacks. New methods are proposed to improve the accuracy of NIDS, but no solution has been proved to be enough. Future research might be oriented toward the development of an up-to-date dataset that would represent the real environment and propose enough instances of each attack category. Another solution might be to develop a framework that would combine the most recent deep learning methods and use its results to improve itself by learning new features each time a new instance is encountered. In addition, to face the large computing resources that will be required for this framework, separating the NIDS into two parts can be a solution. One part that captures the traffic and analyses it and another part that will continuously train the model with the up-to-date datasets would improve the efficiency of how the resources are used in the NIDS. In addition, the second part which requires the biggest number of resources can be in the cloud to benefit from its powerful computing resources.

8.3.2. Solution to Complex Models

Deep learning methods are finding more and more used in recent research and this results in a growth of the complexity of the models. One solution to this growth problem is to use a cloud platform for its computing power. Another solution is to use a high-performance GPU platform. However, these solutions are expensive. Therefore, the use of new feature extraction methods to optimize the dataset that will be used during the training process can be an interesting compromise solution. Using such methods will reduce the complexity of the model while optimizing the accuracy of the solution.

8.3.3. Detecting Encrypted Traffic

Few solutions have proposed a way to detect encrypted traffic. However, nowadays almost everything is encrypted, and hackers are also using encryption to forward attacks. Thus, there is a need to find an efficient way to detect abnormal traffic from encrypted

traffic. As discussed in Section 6, all the solutions were tested using public datasets. However, those datasets are not properly representing our networks of today where a significant amount of the data is encrypted. New solutions to extract important features from encrypted traffic are required to detect abnormal instances.

8.3.4. Use of Feature Extraction

The use of feature extraction in IDS solutions is fairly recent. However, as explained, feature extraction is one of the best solutions that we can use to reduce the complexity of the model. As discussed, only 60% of the reviewed papers in this work used feature extraction, and in most cases the methods used are old such as Principal Component Analysis. Future research should be directed towards deep learning methods for feature extraction and then using simple machine learning techniques to decrease the computing resources needed for their solution.

9. Conclusions

This paper provides a thorough study of Intrusion Detection Systems and how they could be improved using machine learning.

Firstly, the concept of Intrusion Detection Systems was presented. There are three main types of IDS: Network Intrusion Detection System, Host Intrusion Detection System, and a Hybrid Intrusion Detection System. In addition, each type of IDS can either detect attacks by using a recorded signature or by comparing the behavior of the network with a baseline of the normal traffic or both. Then, the different metrics used to assess Intrusion Detection System by various researchers are presented. The most important metrics are the Accuracy, the Detection Rate (Recall) and the F-Measure. A general overview of what is machine learning, and a global taxonomy is also discussed. There are three types of machine learning techniques: Supervised, Semi-supervised and Unsupervised. Most of the machine learning techniques studied fall into one of these categories. A comprehensive review of recently published papers using machine learning for IDS was also discussed. Based on this study, the recent trends show that deep learning methods are more and more used to detect attacks. However, this increases the complexity of the models which requires more computing resources. In addition, it was shown that more and more solutions are using feature extraction with Auto-Encoder as one of the techniques used.

This study also shows that 56% of the proposed IDSs were tested using KDDcup99 and NSL-KDD which are both old datasets. Therefore, these datasets by themselves are not enough to verify the effectiveness of their solution, and, an up-to-date dataset must be created. This new dataset should provide enough instances of minor classes, which have a very low detection rate, and recent attack instances in a real environment. Finally, this work highlights the main issue for IDS is their complexity and their low accuracy for minor classes. In addition, future research trends are presented with the idea presented of a NIDS framework that would continuously be improved using cloud computing.

Future work proposed includes the development of a solution, that can address encrypted network traffic by extracting traffic features to facilitate abnormal instance detection while decreasing required computing resources.

Author Contributions: Conceptualization, P.V., T.N., L.L.D. and M.R.; methodology, P.V. and T.N.; validation, P.V., T.N., L.L.D. and M.R.; investigation, P.V.; resources, T.N., E.O., D.O. and B.L.; data curation, P.V., L.L.D. and M.R.; writing—original draft preparation, P.V. and M.R.; writing—review and editing, T.N., L.L.D. and M.R.; supervision, M.R. and T.N.; funding acquisition, T.N., E.O., D.O. and B.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported, in part, by Science Foundation Ireland grant number 16/RC/3918 to the CONFIRM Science Foundation Ireland Research Centre for Smart Manufacturing and co-funded under the European Regional Development Fund. This work additionally received support from the Higher Education Authority (HEA) under the Human Capital Initiative-Pillar 3 project, Cyberskills.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Acknowledgments: Thank you to the Department of Electronic and Computer Engineering at the University of Limerick for supporting this work as part of the Masters in Information and Network Security programme.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Anderson, P. Computer Security Threat Monitoring and Surveillance. 1980. Available online: https://csrc.nist.gov/csrc/media/publications/conference-paper/1998/10/08/proceedings-of-the-21st-nissc-1998/documents/early-cs-papers/ande80.pdf (accessed on 19 May 2022).
2. ThreatStack. The History of Intrusion Detection Systems (IDS)—Part 1. Available online: https://www.threatstack.com/blog/the-history-of-intrusion-detection-systems-ids-part-1 (accessed on 19 May 2022).
3. Checkpoint. What Is an Intrusion Detection System? Available online: https://www.checkpoint.com/cyber-hub/network-security/what-is-an-intrusion-detection-system-ids/ (accessed on 19 May 2022).
4. Sabahi, F.; Movaghar, A. Intrusion Detection: A Survey. In Proceedings of the 2008 Third International Conference on Systems and Networks Communications, Sliema, Malta, 26–31 October 2008; pp. 23–26. [CrossRef]
5. IBM Cloud Education. Machine Learning. Available online: https://www.ibm.com/cloud/learn/machine-learning (accessed on 19 May 2022).
6. IBM Cloud Education. Supervised Learning. Available online: https://www.ibm.com/cloud/learn/supervised-learning (accessed on 19 May 2022).
7. Seldon. Machine Learning Regression Explained. Available online: https://www.seldon.io/machine-learning-regression-explained (accessed on 19 May 2022).
8. Terence, S. All Machine Learning Models Explained in 6 Minutes. Available online: https://www.ibm.com/cloud/learn/unsupervised-learning (accessed on 19 May 2022).
9. IBM Cloud Education. Unsupervised Learning. Available online: https://www.ibm.com/cloud/learn/unsupervised-learning (accessed on 19 May 2022).
10. Ben n'cir, C.-E.; Cleuziou, G.; Nadia, E. Overview of Overlapping Partitional Clustering Methods. In *Partitional Clustering Algorithms*; Springer: Cham, Switzerland, 2015; pp. 245–275.
11. Joos, K. The Apriori Algorithm. Available online: https://towardsdatascience.com/the-apriori-algorithm-5da3db9aea95 (accessed on 22 June 2022).
12. Matt, B. A One-Stop Shop for Principal Component Analysis. Available online: https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c (accessed on 28 June 2022).
13. Ashiku, L.; Dagli, C. Network Intrusion Detection System using Deep Learning. *Procedia Comput. Sci.* **2021**, *185*, 239–247. [CrossRef]
14. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6. [CrossRef]
15. Protic, D. Review of KDD Cup '99, NSL-KDD and Kyoto 2006+ Datasets. *Vojnoteh. Glas.* **2018**, *66*, 580–596. [CrossRef]
16. Moustafa, N. The UNSW-NB15 Dataset. Available online: https://research.unsw.edu.au/projects/unsw-nb15-dataset (accessed on 28 February 2022).
17. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A. *Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization*; Canadian Institute for Cybersecurity (CIC): Fredericton, NB, Canada, 2018; pp. 108–116.
18. Chen, L.; Kuang, X.; Xu, A.; Suo, S.; Yang, Y. A Novel Network Intrusion Detection System Based on CNN. In Proceedings of the 2020 Eighth International Conference on Advanced Cloud and Big Data (CBD), Taiyuan, China, 5–6 December 2020; pp. 243–247. [CrossRef]
19. Gautam, R.K.S.; Doegar, E.A. An Ensemble Approach for Intrusion Detection System Using Machine Learning Algorithms. In Proceedings of the 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 11–12 January 2018; pp. 14–15. [CrossRef]
20. Al-Yaseen, W.L.; Othman, Z.A.; Nazri, M.Z.A. Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system. *Expert Syst. Appl.* **2017**, *67*, 296–303. [CrossRef]
21. Kanimozhi, P.; Victoire, T.A.A. Oppositional tunicate fuzzy C-means algorithm and logistic regression for intrusion detection on cloud. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6624. [CrossRef]
22. Chen, Y.; Yuan, F. Dynamic detection of malicious intrusion in wireless network based on improved random forest algorithm. In Proceedings of the 2022 IEEE Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC), Dalian, China, 14–16 April 2022; pp. 27–32. [CrossRef]
23. Jabez, J.; Muthukumar, B. Intrusion Detection System (IDS): Anomaly Detection Using Outlier Detection Approach. *Procedia Comput. Sci.* **2015**, *48*, 338–346. [CrossRef]

24. Kurniawan, Y.; Razi, F.; Nofiyati, N.; Wijayanto, B.; Hidayat, M. Naive Bayes modification for intrusion detection system classification with zero probability. *Bull. Electr. Eng. Inform.* **2021**, *10*, 2751–2758. [CrossRef]
25. Chauhan, N. Naïve Bayes Algorithm: Everything You Need to Know. Available online: https://www.kdnuggets.com/2020/06/naive-bayes-algorithm-everything.html#:~{}:text=One%20of%20the%20disadvantages%20of,all%20the%20probabilities%20are%20multiplied (accessed on 9 September 2022).
26. Gu, J.; Lu, S. An effective intrusion detection approach using SVM with naïve Bayes feature embedding. *Comput. Secur.* **2021**, *103*, 102158. [CrossRef]
27. Pan, J.-S.; Fan, F.; Chu, S.C.; Zhao, H.; Liu, G. A Lightweight Intelligent Intrusion Detection Model for Wireless Sensor Networks. *Secur. Commun. Networks* **2021**, *2021*, 1–15. [CrossRef]
28. Xiao, Y.; Xing, C.; Zhang, T.; Zhao, Z. An Intrusion Detection Model Based on Feature Reduction and Convolutional Neural Networks. *IEEE Access* **2019**, *7*, 42210–42219. [CrossRef]
29. Zhang, X.; Chen, J.; Zhou, Y.; Han, L.; Lin, J. A Multiple-Layer Representation Learning Model for Network-Based Attack Detection. *IEEE Access* **2019**, *7*, 91992–92008. [CrossRef]
30. Yu, Y.; Bian, N. An Intrusion Detection Method Using Few-Shot Learning. *IEEE Access* **2020**, *8*, 49730–49740. [CrossRef]
31. 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]
32. 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]
33. Wei, P.; Li, Y.; Zhang, Z.; Hu, T.; Li, Z.; Liu, D. An Optimization Method for Intrusion Detection Classification Model Based on Deep Belief Network. *IEEE Access* **2019**, *7*, 87593–87605. [CrossRef]
34. Vinayakumar, R.; Alazab, M.; Soman, K.P.; Poornachandran, P.; Al-Nemrat, A.; Venkatraman, S. Deep Learning Approach for Intelligent Intrusion Detection System. *IEEE Access* **2019**, *7*, 41525–41550. [CrossRef]
35. 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]
36. Yan, B.; Han, G. Effective Feature Extraction via Stacked Sparse Autoencoder to Improve Intrusion Detection System. *IEEE Access* **2018**, *6*, 41238–41248. [CrossRef]
37. Khan, F.A.; Gumaei, A.; Derhab, A.; Hussain, A. A Novel Two-Stage Deep Learning Model for Efficient Network Intrusion Detection. *IEEE Access* **2019**, *7*, 30373–30385. [CrossRef]
38. Andresini, G.; Appice, A.; Mauro, N.D.; Loglisci, C.; Malerba, D. Multi-Channel Deep Feature Learning for Intrusion Detection. *IEEE Access* **2020**, *8*, 53346–53359. [CrossRef]
39. Ali, M.H.; Mohammed, B.A.D.A.; 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]
40. Liang, D.; Liu, Q.; Zhao, B.; Zhu, Z.; Liu, D. A Clustering-SVM Ensemble Method for Intrusion Detection System. In Proceedings of the 2019 8th International Symposium on Next Generation Electronics (ISNE), Zhengzhou, China, 9–10 October 2019; pp. 1–3. [CrossRef]
41. Wisanwanichthan, T.; Thammawichai, M. A Double-Layered Hybrid Approach for Network Intrusion Detection System Using Combined Naive Bayes and SVM. *IEEE Access* **2021**, *9*, 138432–138450. [CrossRef]
42. Elhefnawy, R.; Abounaser, H.; Badr, A. A Hybrid Nested Genetic-Fuzzy Algorithm Framework for Intrusion Detection and Attacks. *IEEE Access* **2020**, *8*, 98218–98233. [CrossRef]