ICT Express 11 (2025) 181–215

Contents lists available at ScienceDirect

ICT Express

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

Deep learning-driven methods for network-based intrusion detection systems: A systematic review

Ramya Chinnasamy [a], Malliga Subramanian [a], Sathishkumar Veerappampalayam Easwaramoorthy [b], Jaehyuk Cho [c,*]

[a] Department of Computer Science and Engineering, Kongu Engineering College, India
[b] Department of Data Science and Artificial Intelligence, Sunway University, Jalan Universiti, Bandar Sunway, 47500 Petaling Jaya, Selangor, Malaysia
[c] Department of Software Engineering and Division of Electronics and Information Engineering, Jeonbuk National University, Jeonju-Si, 54896, Republic of Korea

ARTICLE INFO

Keywords:
Deep Learning
Network-based Intrusion Detection Systems (NIDS)
Cyber Security
Systematic Review

ABSTRACT

This paper presents a systematic review of deep learning (DL) techniques for Network-based Intrusion Detection Systems (NIDS) based on Preferred Reporting Items for Systematic Reviews and Meta-Analyses: (PRISMA2020) guidelines. It explores recent advancements in data preparation, DL architectures, and performance evaluation metrics for NIDS. The review provides insights into various datasets and tools used in the field, highlighting the effectiveness of DL in improving NIDS performance. Additionally, it discusses the applications of NIDS across different industries and identifies emerging research trends, offering a comprehensive resource for researchers and practitioners in cybersecurity.

1. Introduction

The computer and network security, simply cyber security has been a trending topic forever and a day. Every day, systems, and networks of various organizations like business, medicine, science, engineering, and education experienced versatile cyber-attacks [1]. Accordingly, for all the information technology organizations, information protection is crucial and must be given emphasis. For instance, any security system must provide confidentiality, integrity, and availability [1]. Correspondingly, IDS is one solution to improve cybersecurity. Intrusion detection (ID) is a method of surveilling the computer systems and networks for all the occurrence of events and investigating them for any computer system or network security policy violations. An IDS is a tool that identifies intrusion [1]. In 1980, Jim Anderson initially introduced the concept of IDS [2]. Afterwards, stack of IDS products has been evolved and perfected to meet the demands of network security. Nevertheless, the network size and the applications per node has experienced an enormous growth due to the technological advancements over the last decade [3]. Consequently, a massive amount of essential data has been generated and passed between multiple network nodes. Admittedly, massive number of fresh breach either variation of an old breach or brand-new exploit in recent days make security of these data a big challenge [3]. Since data is dynamic during transfer, many IDS on the market are ineffective in detecting different types of attacks. In the end, precise, effective, and accurate IDS remain essential to increase the security of network nodes. Section 3 gives elaborate details about various kinds of IDS along with advantages and disadvantages of each type. Researchers have explored the application of artificial intelligence (AI) techniques such as deep learning (DL) and machine learning (ML) to address the security concern of IDS [4]. ML and DL methods learn useful information from large-scale data and becoming popular in the last decade due to the advancements in processing unit like powerful graphical processing unit (GPU) [4,5].

1.1. Why deep learning for IDS?

The deep learning (DL) models learns complex features automatically owing to its deep structure. Besides, the deep learning models extract features automatically from raw data and minimize the need for manual feature engineering [5]. Because of this trait, deep learning has become a powerful approach in the development of IDS. The DL algorithms learn intricate and subtle patterns with in the data and make them suitable for identifying complex attack vectors which may be overlooked by traditional methods [6]. The adaptability of the deep

Peer review under responsibility of The Korean Institute of Communications and Information Sciences.
* Corresponding author.
E-mail address: chojh@jbnu.ac.kr (J. Cho).

https://doi.org/10.1016/j.icte.2025.01.005
Received 29 June 2024; Received in revised form 2 December 2024; Accepted 12 January 2025
Available online 13 January 2025

learning models makes them learning continuously from the new and evolving threats and can detect novel attacks without extensive retraining. Since, the DL models can handle large-scale datasets effectively, it is suitable for the processing of large-scale data encountered in the network flow of the contemporary IDS [7]. With the end-to-end learning capabilities, deep learning simplifies the IDS development process and enhances the overall performance of intrusion detection systems [8].

1.2. Why NIDS?

Among various IDS approaches discussed in Section 3, the Network based Intrusion Detection System (NIDS) has unique capabilities and advantages over other methods like Host-based Intrusion Detection System (HIDS). It operates at the network level comprising of data link layer and network layer, and provides a comprehensive view of the traffic flow in the entire network [5]. Because of this network wide perspective, it can detect attacks that span multiple hosts or different network segment. In addition, the NIDS monitors the network traffic in real-time and can detect pattern that indicates a malicious activity and provides immediate response to prevent or mitigate the any damage [9,10]. Lately, the cyber threats are dynamic and evolving, the real-time analysis capability of NIDS makes it more applicable for these cyber-attacks. Furthermore, the centralized management and control of NIDS allows the cyber security team to monitor and analyse the network traffic from a single point of control. It can be deployed in various network segments which allows scalability when the network grows [11]. Over all, the characteristics such as the network wide coverage, real-time detection, centralized management and reduction of host overhead and scalability of NIDS makes it as an essential component of comprehensive network security strategy [8].

1.3. Systematic review

According to Preferred Reporting Items for Systematic Reviews and Meta-Analyses: (PRISMA)2009 [12] statement, "A systematic review is a review of a clearly formulated question that uses systematic and explicit methods to identify, select, and critically appraise relevant research, and to collect and analyze data from the studies that are included in the review" [13]. The process of finishing a systematic review is iterative. The process is significantly influenced by the scope and quality of the included studies, often requiring reviewers to adjust their original review protocol as they progress. Additionally, any changes to the original review protocol must be clearly reported and explained without simply naming it as appropriate or inappropriate. The conduct and reporting of systematic reviews are closely linked, making it challenging to separate the two, as inadequate reporting, such as failing to assess risk of bias, can indicate poor conduct [12]. A checklist for reporting a systematic review or meta-analysis includes items covering the title, abstract, introduction, methods, results, discussion, and funding, ensuring comprehensive and transparent reporting of the review process and findings [12].For reporting systematic reviews, there is an updated guidelines named as PRISMA2020 statement [14]. It includes a flow diagram template which provides a structured method for documenting the selection process of studies included in a systematic review. It begins with identifying records from databases, registers, and other sources, followed by removing duplicates and marking ineligible records with automation tools. The diagram then proceeds to the screening stage, where records are assessed for eligibility, and irrelevant studies are excluded. Reports are sought for retrieval, and those not retrieved are documented alongside the reasons for their exclusion [14].

This paper provides a systematic review based on PRISMA 2020 [14] guidelines about the latest data preparation techniques and DL-based classification methods for the development of effective NIDS. Furthermore, the basic proposal is to impart recent information about various datasets, data preparation techniques, DL based classification and performance assessment for the development of an effective NIDS along with various applications. The investigation for this research is three-fold.

1.4. Article selection and analysis process

A systematic study has been conducted to select latest articles centered on DL-driven NIDS that are proclaimed from January 2018 to July2024.

(i) Each article has been reviewed thoroughly and the components such as datasets, data preparation techniques, learning and classification methods, performance metrics, applications, implementation tools and future trends has been noted.
(ii) Based on this inspection, the state-of-the-art trends of using DL methods for NIDS have been highlighted along with the components mention in the previous step and meticulously documented.

1.5. Novelty of this review

There are various review papers on the literature that provide details of implementation of IDS. Table 1 displays a comprehensive comparison of this article with other review articles. This article differs from other articles in the following aspects

(i) This review provides novel insights into effective preprocessing techniques, feature engineering, and data augmentation methods that are specifically beneficial for network intrusion detection.
(ii) This review includes various applications of NIDS such as Internet of Things (IoT), Industrial Internet of Things (IIoT), Internet of Medical Things (IMoT), Cloud computing, Smart cities, In-vehicular networks and Industrial Control System (ICS) which highlights the adaptability and impact of deep learning methods to develop NIDS across different industries and environments, providing insights into specific challenges and practical implementations.
(iii) It includes various implementation tools which provides practical guidance for building effective NIDS, covering software platforms, machine learning libraries, data management solutions, and evaluation tools.
(iv) It employs a systematic review process using PRISMA2020 guidelines which follow a systematic and methodical approach for gathering, evaluating, and synthesizing research on DL-based NIDS while other articles centered on universal IDS or without systematic approach.
(v) This study focused on articles published from 2018 to July 2024. So, it gives recent trends and information on DL-driven NIDS.

1.6. Structure and section descriptions

This paper is organized as below

(i) The research methodology used in this review is given in Section 2.
(ii) The definition and detailed explanation of various categories of IDS with literature is given Section 3.
(iii) Section 4 discusses and analyzes various deep learning techniques used in the construction of NIDS.
(iv) Various data Preparation techniques such as data cleaning, data augmentation and feature selection techniques for the construction of effective NIDS are identified and analysed in Section 5.
(v) Different evaluation metrics used to analyse the performance of NIDS is given in Section 6.
(vi) Section 7 provides comprehensive explanation of versatile data sets used to train, validate and test the NIDS model.

Table 1
Analogy with similar methods.

Survey Study	Year	Systematic Review	Data Preparation Techniques	NIDS Centered	AI Methods		Implementation tools	Applications
					ML	DL		
Saide et al. [4]	2024	×	×	×	✓	✓	×	✓
Abdulganiyu [5]	2024	✓	×	✓	✓	✓	×	×
Musleh et al. [6]	2023	×	×	×	✓	×	×	✓
Abdulganiyu et el. [8]	2023	✓	×	✓	×	✓	×	×
Halbouni et al. [7]	2022	×	×	×	✓	✓	×	×
Binbusayyis et al. [15]	2022	×	×	×	✓	×	×	✓
Jan et al [9]	2021	×	×	×	×	✓	×	×
Thakkar & Lohiya [10]	2021	×	×	×	✓	✓	×	✓
Sunanda and Jagath [11]	2020	×	×	×	×	✓	×	×
Gamage & Samarabandu [11]	2020	×	×	✓	×	✓	×	✓
Khraisat et al. [16]	2019	×	×	×	✓	×	×	×
Lateef et al [17]	2019	×	×	×	×	✓	×	×
Mishra et al. [18]	2018	×	×	×	✓	×	✓	×
Sultana et al [19]	2018	×	×	✓	✓	×	×	×
This Article	2024	✓	✓	✓	×	✓	✓	✓

(vii) Numerous tools for implementing NIDS are explicated in Section 8.
(viii) Section 9 presents a comprehensive discussion of the findings, offering an in-depth analysis and interpretation of the results derived from the study.
(ix) Future scope and research challenges is given in Section 10.
(x) Section 11 includes the comprehensive findings and discussion.
(xi) Eventually, Section 12 concludes this survey. The abbreviations are encapsulated in Table 6.

1.7. Key contributions

The main contributions of this paper are summarized as follows:

(i) The review presents detailed analysis of various DL techniques used in NIDS, including architectures like Auto Encoder (AE), Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Restricted Boltzmann Machine (RBM), Deep Belief Network (DBN), Multilayer Perceptron (MLP) and ensemble of these methods. It discusses how these techniques are applied to intrusion detection.
(ii) It discusses about various data preparation strategies used for the development of effective NIDS
(iii) It consolidates and analyses the performance evaluation measures used by different researchers, such as accuracy, precision, recall, F1-score, detection rate and false alarm rate. This helps in understanding the effectiveness of different DL-based NIDS.
(iv) The study details the many datasets that were used to test, validate, and train the IDS models. It also offers insights into the data sources and how relevant they are to the efficacy of DL techniques in detecting intrusions.
(v) In addition, this paper provides practical tools and platforms used for implementing DL-driven NIDS, giving valuable information for researchers and practitioners looking for creating and developing these systems. It also mentions the application environment where the NIDS is applied.

2. Research methodology

This review utilized PRISMA2020 [14] guidelines for performing systematic review. This section outlines the comprehensive process followed in conducting the systematic review. Fig. 1 outlines the process of conducting a systematic review. It begins with the formulation of research questions, which guide the focus of the review [5]. Next is the selection of bibliographic databases to search for relevant studies, followed by conducting a scoping search to explore available literature and refine the review's scope [14]. Based on this, a detailed search strategy is planned, identifying keywords and criteria for inclusion and exclusion. Then, titles and abstracts are screened to identify relevant studies, after which the full papers are obtained for a more in-depth review. The quality of the studies is verified, ensuring that only high-quality research is included. Finally, data from the selected papers is extracted and analyzed to synthesize findings and address the research questions.

2.1. Research questions (RQ)

The first step is to clearly define the research questions (RQ) that the review seeks to answer. The RQs for this review are

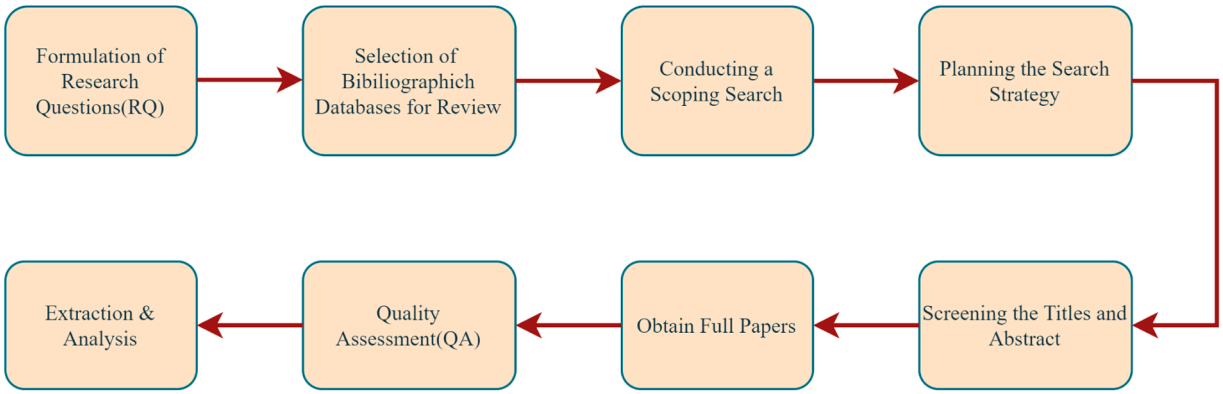

Fig. 1. Process of conducting a systematic review.

(i) What are the recent advancements in deep learning methodologies for NIDS, and how effectively do these approaches address the evolving challenges in intrusion detection, such as adaptability to novel attack patterns and scalability across various environments?
(ii) What are the different datasets available for the development and evaluation of NIDS? How do these datasets support the development of effective intrusion detection models, and what are their limitations in terms of diversity, realism, and representation of evolving attack patterns in dynamic environments?
(iii) What are the predominantly used data preparation techniques such as encoding of categorical features, normalization, data augmentation techniques for minority attack classes and feature selection techniques including various optimization algorithms for the development of NIDS? Which data preparation techniques enhance the efficacy of NIDS, and what are their impacts on model robustness and detection accuracy?
(iv) What are the strengths and limitations of current evaluation metrics for NIDS, and in what contexts are specific metrics most appropriate for assessing system reliability and robustness?
(v) What are the perspectives on tools and frameworks used in implementing DL-based NIDS, particularly their impact on scalability, ease of use, and performance optimization in real-world deployments?
(vi) What are the applications of NIDS in various domains, and how effective are deep learning methods in addressing the unique challenges of these domains?

2.2. Selection of bibliographic databases for review

This step involves choosing the databases that will be used to search for relevant literature. A Clarivate Web of Science and Scopus are selected as databases to search records and reports. These databases were selected due to their extensive coverage of high-quality, peer-reviewed publications across multiple disciplines, particularly in computer science and network security.

2.3. Conducting a scoping search

To ensure a comprehensive exploration of the literature on network-based intrusion detection, a scoping search was conducted using databases: Web of Science (WoS) and Scopus. The scoping search aimed to capture a wide range of studies, providing a preliminary overview of the existing research landscape. This process allowed for the identification of key trends, gaps, and the overall volume of literature available on network-based intrusion detection. The results from this scoping search informed the refinement of the search strategy and helped ensure that subsequent stages of the review were both exhaustive and focused [14].

2.4. Search strategy

To ensure a thorough and structured review of the literature on NIDS, a detailed search strategy was formulated following the scoping search. This strategy was designed to capture a broad range of studies relevant to the research questions, with an emphasis on high-quality, peer-reviewed papers. It involves the following steps

(i) The key concepts are identified. The key concepts are Network-based intrusion detection, deep learning techniques, data preparation techniques and applications of NIDS.
(ii) List of Keywords are identified based on the key concepts.
(iii) Using Search Operators and Combining Terms:
(iv) Use detect* to capture detection, detecting, detector, etc. Combine related keywords with OR to capture more relevant results. The applied limits are shown in Table 2 to get more appropriate result.

Table 2
Applied Limits.

Date range	Language	Publication/document type
2018-July2024	English	Journal articles, Conference Proceedings

(v) A pilot search was conducted to test this search string in the selected databases WoS and Scopus. The results were reviewed and additional relevant keywords and subject headings were identified. The final list of identified keywords is shown in Table 3.
(vi) The search was conducted in WoS and Scopus based on the above search strategy and list of records with titles and abstracts are exported to Endnote. The search operators along with Boolean operators for the Scopus database is shown in Table 3. The same combination of search operators within ""instead of {} were used while searching in WoS. In our search, there were 349 records from Scopus and 441 records from WoS exported to Endnote. The identified records were undergone rigorous refinement based on PRISMA2020 guidelines as shown in Fig. 2. The Endnote identified 222 records as duplicates and removed the duplicates subsequently. So, there were 568 records available for screening in the next phase of the PRISMA2020 guidelines.

2.5. Screening the titles and abstract

The goal of the Screening the titles and abstracts phase of a systematic review, is to quickly filter through a large number of studies to identify those that are potentially relevant based on their titles and abstracts. This step involves applying initial inclusion and exclusion criteria to ensure that only studies aligned with the research objectives are retained for further evaluation. The inclusion and exclusion criteria for screening the titles and abstract phase is given in Table 4. A total of 568 records were screened. Following the initial identification, 206 records were excluded based on the exclusion criteria. The remaining 362 studies were deemed potentially relevant and moved forward for a more in-depth full-text review to assess eligibility.

2.6. Obtaining full papers

This phase refers to the step where full-text reports of studies identified during the titles and abstracts screening phase are retrieved and assessed for further evaluation. At this stage, all 362 reports were successfully retrieved, indicating no access issues or unavailability of full-text documents. These studies were thoroughly assessed for eligibility based on the predefined inclusion and exclusion criteria given in Table 4. In total, 362 reports underwent this detailed evaluation. However, 275 reports were excluded from the review by following exclusion criteria, including reports that contains machine learning techniques also (94 studies), and failure to meet quality assessment criteria (181 studies). Following this phase, the remaining 87 studies met the eligibility criteria and were included for further review and analysis.

2.7. Quality assessment (QA)

The quality of included studies was assessed to ensure the reliability and robustness of the results [14]. The quality assessment criteria were developed based on key factors related to research design, methodology, dataset usage, and result reporting. Each selected study was evaluated using the following criteria.

(i) QA1: Does the study clearly state its research questions or objectives, and is the problem addressed well-defined?

Table 3
Search operators and combining terms for Scopus Database.

Network-based intrusion detection	AND	Deep learning techniques	AND	Data preparation techniques	Applications of NIDS
{NIDS} OR {Network intrusion detect*}OR {Network security monitoring} OR {Network anomaly detect*} OR {Network attack detect*} OR {Anomaly detect*}OR{Misuse detect*}OR {Hybrid detect*)		{Deep learning} OR {Artificial Intelligence} OR{AI}OR {Auto Encod*} OR {AE} OR {Deep Neural Network*} OR{DNN} OR {Convolutional Neural Network*} OR{CNN}OR {Recurrent Neural Network*} OR{RNN}OR {Deep Belief Network*} OR{DBN} OR {Multilayer Percept*} OR {MLP}OR {Long Short Term Memory} OR {LSTM} OR {Restricted Boltzmann Machine*} OR {RBM}		{Data preparation} OR {Data preprocessing} OR {Data cleaning} OR {Data transformation} OR {Feature selection} OR {Feature extraction} OR {Data normalization} OR {Data encoding} OR {Data augmentation} OR {Data imputation} OR {Dataset merging	{IoT}OR {Industrial control system*} OR {ICS}OR{Cloud security}OR {Smart grid security} OR{Health care}OR {IMoT}OR {In-Vehicle*} OR{Cyber Physical System*}

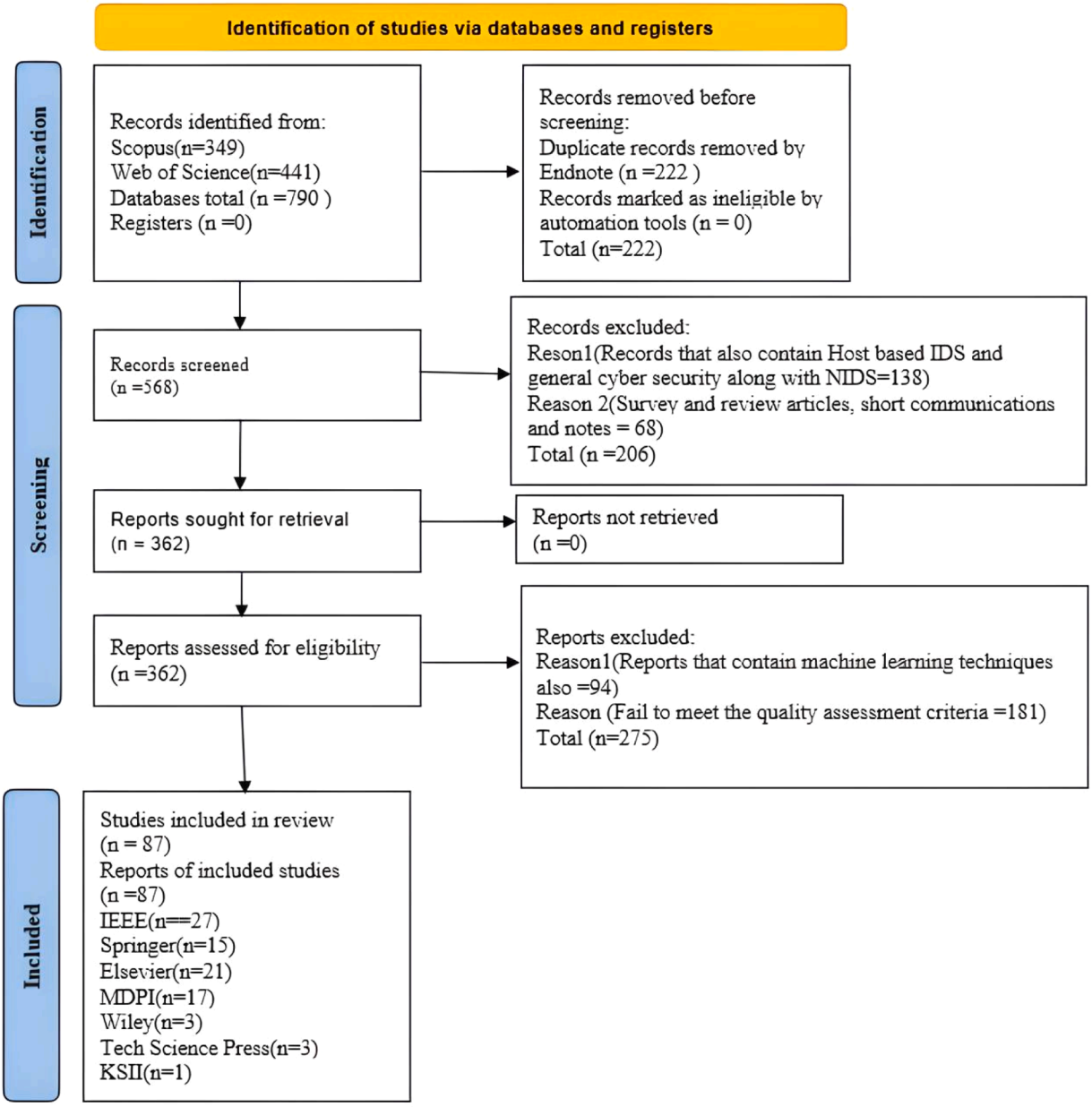

Fig. 2. Flow chart for the essential publication selection.

(ii) QA2: Does it provide sufficient detail on the methodology, including data processing, feature selection, model evaluation, and performance assessment?

(iii) QA3: Does the study utilize relevant and recognized benchmark datasets for NIDS research?

Table 4
Inclusion and exclusion criteria.

Phase	Inclusion Criteria	Exclusion Criteria
Screening the Titles and Abstracts	Title/abstract mentions NIDS. Published in peer-reviewed journals and conferences.	Records mention on topics unrelated to NIDS (e.g., host-based IDS, general cybersecurity). Clearly a review or survey article or short communications or notes
Obtain Full Papers	The study that uses only deep learning methods. Articles with full text availability	Study that uses machine learning techniques also. Articles with inaccessible full texts.

(iv) QA4: Are the study's experimental results meaningful, and does the study include a discussion on the real-world application of the proposed model?
(v) QA5: Does the study contribute new insights, techniques, or innovations to the field of NIDS research?

Scoring System and Selection Criteria:
Each QA criterion was assigned 1 point if fulfilled, for a maximum possible score of 5 points. Each paper is assessed using the QA criteria, with two critical factors determining its qualification as a high-quality paper suitable for inclusion in the study.

C1: The paper must score a total of 3 or more points out of 5.
C2: The paper is required to satisfy the criteria outlined in QA2, QA3, and QA4.

Total of 181 studies were excluded because they did not meet the quality assessment criteria.

2.8. Distribution of publications

There are 87 journal articles selected from 790 records upon following PRISMA 2020 guidelines for analysis and synthesis of results. Table 5 gives distribution of articles over various databases. The selection criteria for the systematic literature review show that IEEE has the highest selection percentage at 31.03%. Elsevier follows with 24.14%, and MDPI accounts for 19.54% of the selections. Springer holds 17.24% of the total, while Tech Science Press and Wiley each have a selection percentage of 3.45%. KSII has the lowest selection percentage at 1.15%.

These percentages reflect the distribution of publications across different journals for the review. Fig. 3 delineates the selected publications over the years from 2018 to 2024. Fig. 4 shows the distribution of various DL methods over the years from 2018 to 2024.There are 5 articles selected in the year 2018 followed by 11 articles in 2019 followed by 19 articles in 2020 and 17 articles in 2021 followed by 10 articles in 2022 and 17 articles in 2023 and finally 8 articles in 2024.

2.9. Utilized tools

(i) draw.io and Lucid chart are employed for the drawing of figures and diagrams in this study.

Table 5
The proportions of journals selected in this review.

Journal	Percentage of Selection
IEEE	31.03%
Elsevier	24.14%
Springer	17.24%
MDPI	19.54%
Tech Science Press	3.45%
Wiley	3.45%
KSII	1.15%

Table 6
Acronyms and their expansions used in this study.

Acronym	Expansion
"IDS	Intrusion Detection System
NIDS	Network based Intrusion Detection System
HIDS	Host based Intrusion Detection System
NBAIDS	Network Behavioral Analysis Intrusion Detection System
WIDS	Wireless Intrusion Detection System
AI	Artificial Intelligence
DL	Deep Learning
ML	Machine Learning
AE	Auto Encoder
SDCAE	Stacking Dilated Convolutional Auto Encoders
NDAE	Nonsymmetric Deep Auto Encoder
SAE	Stacked Auto Encoder
DNN	Deep Neural Network
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
SLSTM	Stacked Long Short-Term Memory
Bi LSTM	Bidirectional Long Short-Term Memory
RBM	Restricted Boltzmann Machine
DBN	Deep Belief Network
MLP	Multi-Layer Perceptron
CBOA	Chaotic Butterfly Optimization Algorithm
RF	Random Forest
PSO	Particle Swarm Optimization
SMO	Sequential Minimum Optimization
NIST	National Institute of Standard and Technology
DDoS	Distributed Denial of Service
STL	Self-Taught Learning
MAPE-K	Monitor-Analyze-Plan-Execute over a shared Knowledge
DSAE	Deep Sparse Auto Encoder
PTDAE	Pretraining approach with Deep Autoencoder
HPO	Hyper Parameter Optimization
GAN	Generative Adversarial Networks
DMA	Direct Memory Access
WDLSTM	Weight Dropped Long Short-Term Memory
SDN	software defined networking
GRU-RNN	Gated Recurrent Unit *Re*-Current Neural Network
SFSDT	Sequence Forward Selection (SFS) and Decision Tree (DT)
Bi-LSTM	Bidirectional Long Short-Term Memory
GRU	Gated Recurrent Unit
FCN	Fully Connected Network"
GAN	Generative Adversarial Network
SMOTE	Synthetic Minority Over-sampling Technique
ADASYN	Adaptive Synthetic Sampling
ET	Extra Tree
CVAE	Conditional Variational Autoencoder
XGboost	Extreme Gradient Boosting
IGAN	Iterative Generative Adversarial Network
WCGAN-GP	Wasserstein Conditional Generative Adversarial Network with Gradient Penalty
GA	Genetic Algorithms
PCA	Principal Component Analysis
ACGAN	Auxiliary Classifier GAN
NSL-KDD	Network Security Laboratory - Knowledge Discovery and Data Mining
KDD	Knowledge Discovery and Data Mining
CIC-IDS	Canadian Institute for Cybersecurity - Intrusion Detection System
UNSW-NB15	University of New South Wales - Network-Based 2015
AWID	Aegean Wi-Fi Intrusion Dataset
Bot-IoT	Botnet Internet of Things Dataset
ICS	Industrial Control System
IoT	Internet of Things
IMoT	Internet of Medical Things
IIoT	Industrial Internet of Things

(ii) Fotor tool is used for image enhancement.
(iii) Endnote is used for reference management and removal of duplicates.

3. Intrusion detection system

This section describes the basics of IDS and its categorization depending on its implementation and detection. According to NIST, any

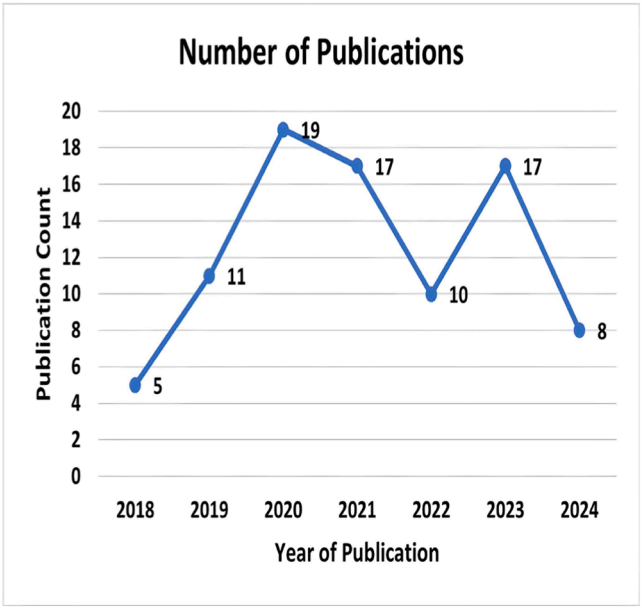

Fig. 3. Distribution of selected articles over the year with DL techniques.

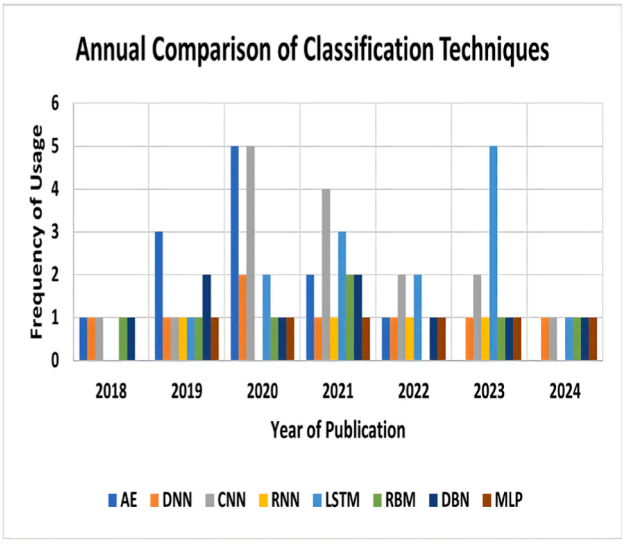

Fig. 4. Distribution of selected articles over the year with DL techniques.

illegal access to network and system data that undermines its availability, integrity, and confidentiality is referred to as an intrusion [3]. Subsequently, an IDS is a software that continuously surveils the network and computer system and identify any possible intrusion [5]. The IDS generates and sends alerts when any intrusion occurs in the host and the network to the concerned authority [4]. The IDS is classified according to its deployment and detection methods as shown in Fig. 5.

3.1. Deployment-driven ids

There are four components of IDS based on how they monitor the events, the types of events they monitor and their deployment methods. They are (i)Network-based IDS (NIDS) surveil wired network traffic for specific network partition or devices and examine the protocol and network activity to identify any dubious activity. It is generally deployed at the boundary between networks [5,8]. (ii)Host-based IDS(HIDS) monitors the events within a single host and analyses the events for any doubtful activity. HIDS are generally deployed on critical hosts such as servers with sensitive information or publicly accessible servers [5,7]. (iii)Wireless IDS(WIDS) monitor wireless network traffic and investigate wireless protocols for any dubious activity. It can normally be deployed within the range of wireless network to monitor the traffic [6]. (iv) Network behavioral analysis IDS(NBAIDS) inspect network traffic to pick out threats that generate aberrant traffic flow. It is commonly deployed with in the internal network of an organization [9].

3.2. Detection-driven ids

The IDS is divided into three types based on how they detect the intrusion. They are(i) Signature-Based detection: a signature is a pattern that is analogous to a known threat. In signature-based detection, the observed event is compared with the signature pattern. If it matches, then it classifies the observed event as threat. It is powerful in identifying known threats whereas it is inefficient in detecting unknown or new threat [20]. (ii)Anomaly-based detection: is a method of comparing the observed event with descriptions of what action is considered normal and measure the significant deviations. It has profiles of normal behavior of all the objects of the system and network [5]. The anomaly-based IDS compares the characteristics of current event with the threshold of the profile. If it exceeds the threshold, the corresponding authority is notified [15]. (iii)Stateful-Protocol Analysis: is a method of comparing observed protocol event with predefined benign protocol activity. The universal protocol profile is usually given by protocol vendors [9]. Table 7 gives the comprehensive comparison of various types of IDS along with advantages, limitations, example tools and use cases.

Overall, NIDS offers more advantages over other methods such as

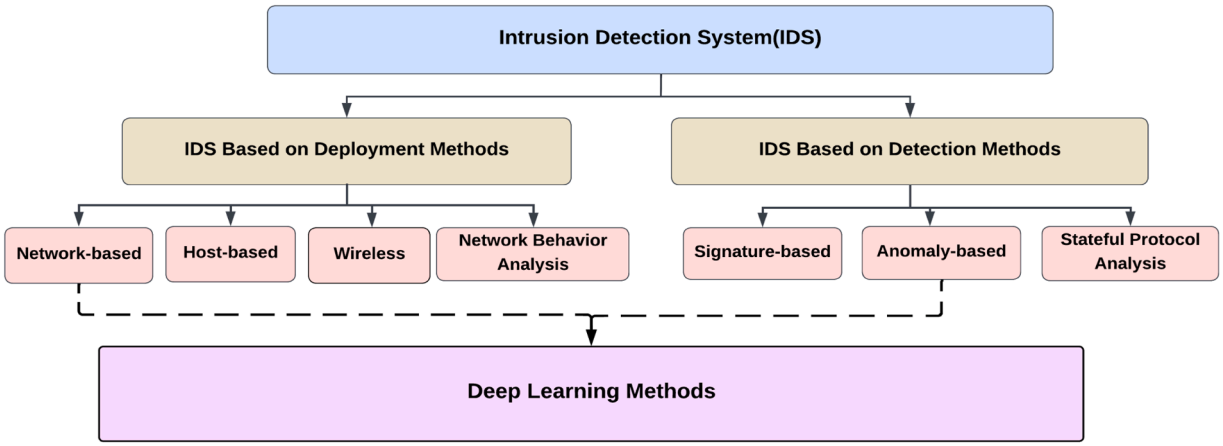

Fig. 5. IDS categorization with the applied methods for the study.

Table 7
A detailed analysis of IDS types: Advantages, challenges, tools, and use cases.

Category	Type of IDS/Detection	Advantages	Limitations	Examples	Application Areas
Deployment-driven IDS	Network-based IDS (NIDS) [5]	(i)Real-time detection. (ii) Scalable for large networks. (iii)Effective for detecting network-level attacks.	(i)Limited visibility into host-level activities. (ii)Challenges with encrypted traffic.	Snort, Bro/Zeek, Suricata	(i)Large organizational networks. (ii)Detecting DDoS attacks. (iii)Monitoring public-facing servers.
	Host-based IDS (HIDS) [7]	(i) Detailed host-level monitoring. (ii) Complements NIDS for a layered defence.	(i)Cannot monitor network-level threats. (ii) Limited scalability in large systems.	OSSEC, Tripwire, AIDE	(i) Protecting critical servers. (ii)Monitoring endpoints with sensitive data. (iii)Securing database systems.
	Wireless IDS (WIDS) [6]	(i) Specialized for wireless threats. (ii)Essential for IoT and mobile networks.	(i) Limited range of deployment. (ii)Requires hardware integration for coverage.	AirMagnet, Kismet	(i)Securing IoT environments. (ii) Detecting rogue access points. (iii) Protecting Wi-Fi networks in enterprises.
	Network Behavioral Analysis IDS (NBAIDS) [3]	(i)Effective for zero-day attacks. (ii)Identifies unusual traffic patterns such as DDoS attacks.	(i)High computational overhead. (ii)May generate false positives for dynamic traffic patterns.	Cisco Stealthwatch, Darktrace, Flowmon	(i)Identifying insider threats. (ii)Monitoring advanced persistent threats (APTs). (iii)Detecting abnormal network flows.
Detection-driven IDS	Signature-based Detection [20]	(i)High accuracy for known threats. (ii)Low false positives.	(i) Ineffective for zero-day or novel attacks. (ii)Requires constant signature updates.	Snort (Signature mode), Symantec IDS	(i) Identifying known malware. (ii)Monitoring standard attack patterns. (iii) Protecting legacy systems.
	Anomaly-based Detection [17]	(i)Detects unknown attacks. (ii)Monitors the entire system for deviations.	(i)High false alarm rate. (ii)Difficult to define normal behavior accurately.	Bro/Zeek (anomaly mode), Splunk (with ML add-ons)	(i) Detecting zero-day attacks. (ii)Monitoring new and emerging threats. (iii)Securing dynamic network environments.
	Stateful Protocol Analysis [15]	(i) Effective for protocol-specific threats. (ii)Useful in industrial networks.	(i)Requires predefined protocol standards. (ii) Limited to protocol-level analysis.	McAfee NSP, Palo Alto Networks NGFW	(i)Industrial control systems (ICS). (ii)Securing SCADA networks. (iii) Monitoring compliance with protocol standards.

real time detection, wide range of network level attack detection, compliance with regulatory requirements, scalability, complementarity with host-based detection and ability to detect network attacks in packet level [17]. Hence, this paper focusses on the design and development of NIDS.

The sophistication and frequency of cyber threats have increased in recent years, resulting in significant changes to the cybersecurity landscape [3]. Although conventional IDS have long been essential tools for protecting network infrastructures from malicious activity, they are finding it difficult to keep up with the ever-changing threat landscape [6]. In response to these challenges, there has been a growing interest in applying Artificial Intelligence (AI) methods for the development of IDS. AI techniques such as machine learning and deep learning, have shown remarkable capabilities in various domains, ranging from image recognition to natural language processing [17]. Application of AI methods in the development of IDS yields better accuracy, adaptability and efficiency. The AI methods make the IDS to learn autonomously from data, identify complex attack patterns, and adapt to emerging threats in real-time [19].

Among the several AI approaches for the development of IDS, deep learning is particularly powerful because of its unmatched capacity to identify complex patterns and relationships within large and detailed datasets [5]. Unlike conventional AI methods, which may rely on predefined rules, deep learning IDS has advanced neural network architectures to autonomously learn representations directly from raw network data. This inherent capability makes deep learning IDS to overcome the limitations of traditional approaches and offers several advantages in the development of IDS [8,7].

This paper aims to investigate the application of deep learning methods for the development of NIDS by exploring the fundamental ideas, working methods, and advantages of incorporating deep learning techniques into NIDS. Through an in-depth review of existing literature, case studies, and practical implementations, this paper elucidates the deep learning methods for the construction of NIDS to detect, mitigate and respond to wide range of cyber threats.

4. Deep learning methods for the development of nids

This section provides the most common properties of DL techniques for building effective NIDS. Furthermore, DL methods are categorized into supervised and unsupervised learning [9]. In supervised learning, the fruitful information is selected from the labeled data while the fruitful information is extracted from unlabeled data [7]. Fig. 6 outlined the general process of DL based NIDS. It has three major steps. (i) Data preprocessing or data preparation step (ii) Training step (iii) Validation and testing step. In all the suggested solutions, the data set is preprocessed to convert into a form that is appropriate for the AI algorithm [11]. This stage typically involves data cleaning, selection, integration, and data reduction [15]. The preprocessed data is roughly divided into two portions, the training and the testing dataset. Generally, 80% of the original dataset is given to the training step to generate a model. The remaining 20% is fed to the testing step to test the model for its performance. The upcoming section provides detailed sketch of various DL approaches for NIDS. Besides, Fig. 8 portrays the categorization of DL-Driven techniques for NIDS. “A subset of machine learning called deep learning” uses several layers to obtain deep structure. Since DL

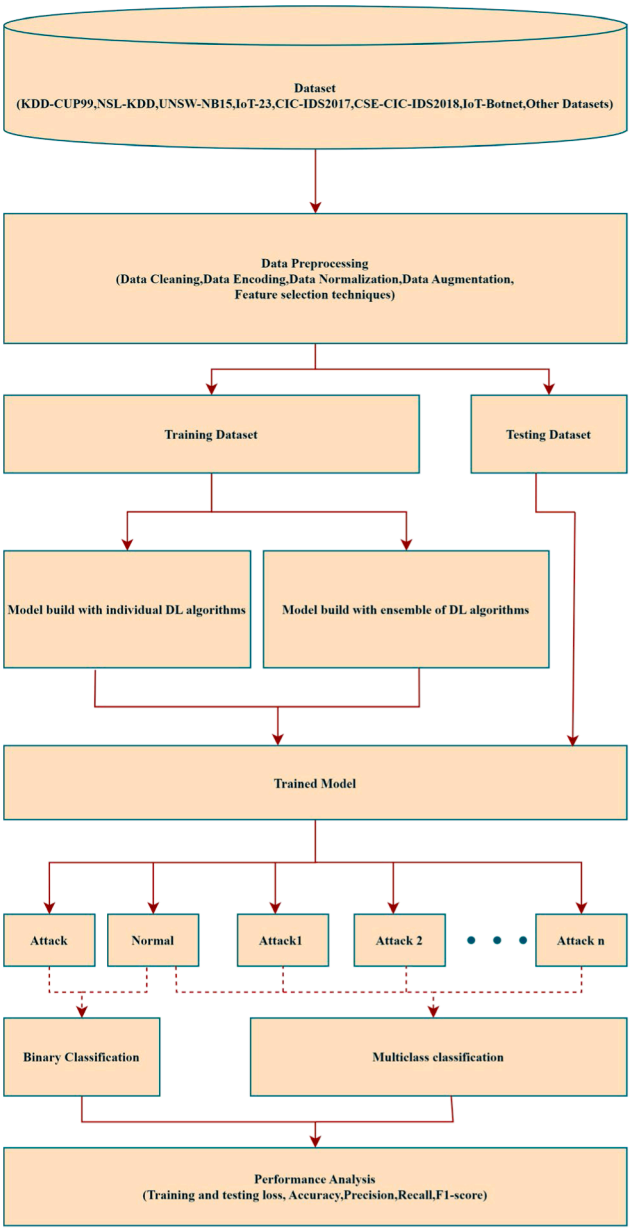

Fig. 6. General methodology of DL-driven NIDS.

approaches pick up the essential traits on their own, they are more effective [21]. This part provides details of the DL methodologies followed in various DL-driven NIDS solutions.

4.1. Auto encoder (AE)

Autoencoder (AE) is a famous DL technique that pertains to the group of neural networks with unsupervised learning. The primary goal of an autoencoder is to encode the input into a compressed representation and then decode it back to the original input. The encoder and the decoder are the two primary components of AE [21] as shown in Fig. 7. The mathematical formulation of encoding [21] is

$$z = f(y) = \sigma\left(w_y^T\right) \tag{1}$$

The mathematical formulation of decoding [21] is

$$y' = g(z) = \sigma\left(W^T z + b'\right)^T \tag{2}$$

In an autoencoder, a bottleneck has been introduced in the network to force the creation of a compressed representation of the original input. If the input features are related, then this compression is easier and the network learn the inherent structure. The decoder on the other hand tries to reconstruct the original input by using the learned inherent structure.

Shone et al. [22] devised a variation of auto encoder scheme called non-symmetric deep auto-encoder that employs robust unsupervised learning of features. Further, NDAE minimize human intervention and improve detection accuracy. He et al. [23] employed a variant of autoencoder named stacked auto encoder (SAE) that minimizes the feature dimension and lessen the memory for covariant matrix. Since SAE selects high level features automatically to identify a breach, detection rate is higher. Muhammed et al. [24] designed an IDS with the encapsulation of SAE with two latent layers and deep neural networks (DNN) with multiple layers to identify intrusion. Also, the system outperformed in comparison with benchmark systems. AE that possesses non-negativity constraints used to select less frequent features [25]. Besides, it reduces load given to the edge classifier and issue the load to IoT and other edge devices. The result of sparse auto encoder is fed to feedforward autoencoder to provide an additional layer of data mutation [26]. "Self-Taught Learning (STL) and MAPE-K are encapsulated to make a scalable, autonomous and self-adaptive misuse IDS". In addition, this methodology avoids the constant necessity of replenishing the training set manually. Dutta et al. [27] experimented a method of using deep sparse autoencoder in data preprocessing for feature extraction. Furthermore, stacking grouping approach is used for classification.

Tang et al. [28] experimented an ensemble of stacked autoencoder with DNN and attention mechanisms. The SAE after training, can extract features on it's own and can set the weights of DNN important layers to improve the accuracy and efficiency of the detection. The combination of Stacked sparse autoencoder with DNN was suggested by Bharadwaj et al. [29] to detect Distributed Denial of Service (DDos) efficiently. The advancement results in small reconstruction error, avert blowing and vanishing gradients and evade overfitting with smaller networks. A principle called stacked generalization is employed in this study in both the network and host. Various DL methods along with their advantages and disadvantages is given in Table 8.

Amalgamation of Deep Neural Network and pretraining approach with deep autoencoder (PTDAE) along with hyperparameter optimization (HPO) is examined to ameliorate detection accuracy of IDS [30]. HPO is tuned automatically that could reveal the crucial parameters for DL-driven IDS. Furthermore, the efficiency of DL-driven IDS is improved with the immaculate feature extraction. Lee and Park [31] explicated a methodology of integrating generative adversarial networks (GAN) and AE to resolve data imbalance problem and to increase detection accuracy. Random forest (RF) is employed to classify the data to evaluate the performance of Auto Encoder Conditional GAN(AE-CGAN). Ieracitano et al. [43] developed an IDS using autoencoder by grouping statistical methods and data analytics to acquire better and firmly correlated features. The suggested model attains low false alarm rate and high detection accuracy.

In summary, autoencoders have proven to be highly effective in NIDS, particularly for anomaly detection. By compressing high-dimensional network traffic data, they can learn representations of normal behavior and identify deviations that indicate potential security threats. Their application in IoT networks has demonstrated significant advantages, especially in detecting anomalies in highly dynamic environments such as smart homes and industrial IoT. Furthermore, in edge computing scenarios, autoencoders enhance efficiency by detecting local anomalies at the edge, thereby offloading detection tasks from centralized systems before transmitting data to the cloud. These capabilities highlight the versatility and strength of autoencoders in modern NIDS, especially in environments requiring unsupervised learning and real-time responsiveness.

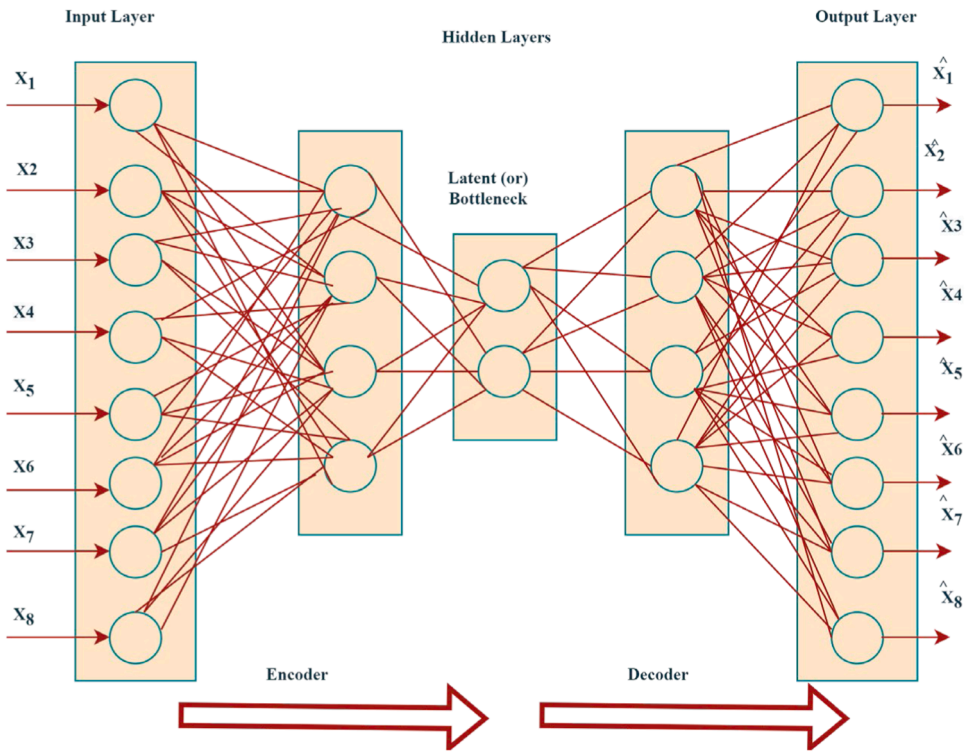

Fig. 7. General Block diagram of autoencoder.

4.2. Deep neural networks

A deep neural network (DNN) is a type of artificial neural network (ANN) characterized by multiple layers of nodes between the input and output layers as shown in Fig. 9. These intermediate layers, known as hidden layers, allow the network to learn complex representations of the input data by progressively extracting higher-level features [21]. The term "deep" refers to the depth of the network, meaning the number of hidden layers it contains [21]. DNNs are designed to model complex patterns and relationships in data. Deep Neural Network (DNN) is a deep learning method in which various layers are employed from input to output. Their efficiency and high accuracy make them essential in various deep learning applications, including object detection, language translation, and transfer learning with models like YOLO, BERT, and RESNET [8].

The mathematical formulation of DNN [43] for weighted sum is given in Eq. 3.

$$z = x.w + b \tag{3}$$

Priya et al. [43] designed an IDS to identify and classify the unprecedented breach using deep neural networks. Hyper selection method is used to precondition, optimize, and tune the network parameters. When comparing with traditional machine learning approaches, DNN yields accuracy improvement of 15% and time complexity reduction of 32%. Application of deep neural network without any alteration and free from applying any hyperparameter tuning is the major benefit of the proposed approach. The problem of poor performance in IDS in dynamic environment is averted using drift detection method [44]. Vinayakumar et al. [45] utilized DNN to design an adaptable and practical IDS to identify and categorize unprecedented attacks. In this DNN model, conceptual and complete data features are learned by passing them to many hidden layers of DNN. The experimental results showed that DNN outperforms all the classical machine learning based detection.

Awotunde et al. [32] developed a NIDS with “deep feedforward neural network and rule-based feature selection”. The trained model is used to analyse the incoming TCP/IP packets. Naseer et al. [33] developed an anomaly-based NIDS using various deep neural network structures. The model was trained and evaluated with NSL-KDD dataset. The results showed that deep learning models outperform conventional machine learning techniques in key classification metrics, indicating their suitability for real-world anomaly detection applications. Alzughaibi and Khediri [46] introduced an IDS model that enhances IDS performance in cloud environments by developing two deep neural network (DNN) models using multi-layer perceptron (MLP) with back-propagation (BP) and particle swarm optimization (PSO). It achieved accuracies of 98.97% for binary classification and 98.41% for multi-class classification on the CSE-CIC-IDS2018 dataset. He et al. [47] introduced NIDS-Vis, an innovative black-box algorithm that visualizes the decision boundaries of DNN-based NIDSs, providing insights into their geometry and the impact on performance and adversarial robustness. Through this visualization, the study uncovers a tradeoff between performance and robustness. To address this, two novel training techniques—feature space partition and distributional loss function—are proposed to enhance the generalized adversarial robustness of DNN-based NIDS without significantly compromising performance.

In conclusion, DNN have emerged as a highly effective tool in the design and implementation of NIDS [43]. Their ability to handle complex data patterns and large-scale datasets allows them to outperform traditional machine learning methods, as demonstrated by studies that report improvements in both accuracy and time efficiency [32,45]. DNNs have been successfully applied in a variety of real-world contexts, including cloud environments where they are used to detect sophisticated threats such as advanced persistent threats (APTs) and Distributed Denial-of-Service (DDoS) attacks [33,43]. They have also shown great potential in critical infrastructure networks, where the ability to process high-dimensional data and adapt to dynamic environments makes them well-suited for anomaly detection in systems like power grids and healthcare [44,47]. Furthermore, the advancements in DNN models, including the use of hyperparameter optimization, rule-based feature selection, and adversarial robustness techniques, highlight their applicability in modern, large-scale, and complex network environments.

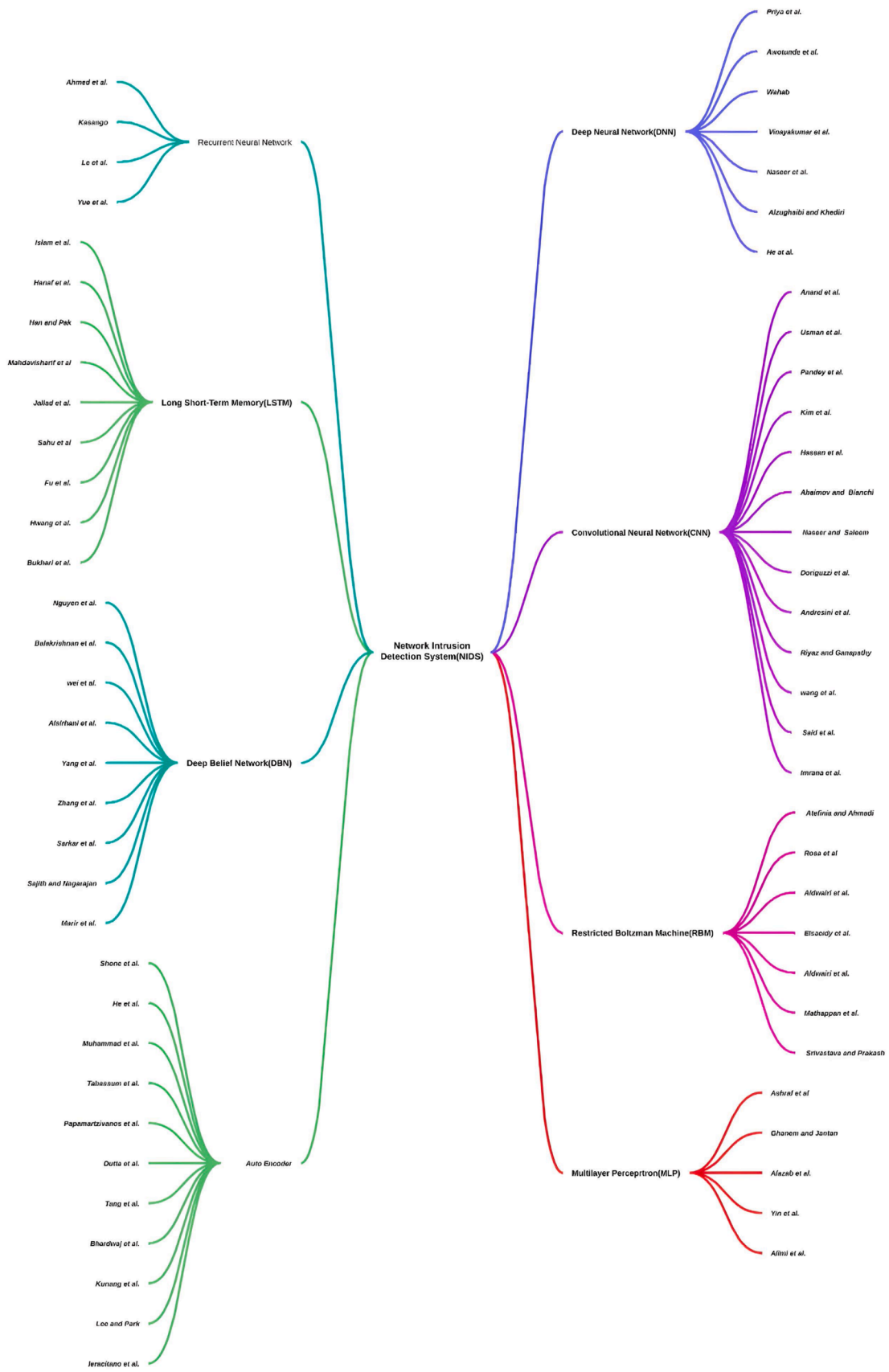

Fig. 8. Categorization of Surveyed DL Algorithms.

Table 8
Articles with advantages and disadvantages of DL methods.

Reference	DL method	Advantages	Disadvantages
[30]	AE	(i) It significantly enhances attack classification accuracy and outperforms previous approaches in multiclass classification. (ii)The use of automatic hyperparameter tuning efficiently identifies the optimal configuration of hyperparameters, such as learning rate, number of layers, and neurons, leading to improved model performance	(i) The hyperparameter optimization process can be time-consuming, especially when optimizing a large number of hyperparameters simultaneously. (ii) The effectiveness of the model is heavily dependent on the chosen feature extraction methods and hyperparameter configurations.
[31]	AE	(i)The AE-CGAN model effectively addresses the data imbalance issue in network intrusion detection systems by utilizing the conditional GAN to oversample rare classes.	(i) It requires significant computational resources and time for training, which may be a limitation for real-time intrusion detection applications in resource-constrained environments.
[32]	DNN	(i)The framework can handle large-scale data in real time using distributed deep learning models. (ii)DNNs exceed classical machine learning classifiers in detecting cyberattacks across various datasets and IDS types.	(i)Training complex DNNs requires significant computational resources (ii)The framework does not provide in-depth analysis of malware structures or characteristics.
[33]	DNN	(i)Capable of detecting new and existing attacks in IIoT networks with a strong focus on TCP/IP packets. (ii)Achieves a high identification rate and low false positive rate .	(i) Limited Real-World Testing (ii)Current model is limited to specific protocols and needs extension to accommodate a broader range of protocols .
[34]	CNN	(i) Distributes data among multiple levels of edge devices to handle large volumes of multimedia data, minimizing packet drop and buffer overflow issues. (ii) Utilizes Least-Squares Support-Vector Machines (LS-SVMs) to anonymize and secure data during transmission, addressing privacy concerns.	(i)The multi-level architecture introduces additional complexity in system design and maintenance, which may increase deployment and operational challenges. (ii)As the number of edge devices grows, managing the distributed data and ensuring efficient communication between levels could become increasingly challenging.
[35]	CNN	(i)The AI-IDS uses an optimal CNN-LSTM model and SFL to accurately detect sophisticated web attacks, including unknown and obfuscated patterns, which legacy systems might miss. (ii)Implemented in a high-performance computing environment with Docker-based modular design, the system can scale effectively and adapt to evolving attack patterns.	(i)The system initially struggles with false positives and requires continuous re-validation of suspicious events, which may delay detection and require ongoing manual oversight. (ii)The implementation and maintenance of the AI-IDS involve complex configurations and significant computational resources
[36]	LSTM	(i)Requires only 25.8% of the memory compared to existing NIDSs, allowing support for a greater number of concurrent sessions. (ii)Achieves high detection performance comparable to or better than existing methods without detection delay.	(i)The system assumes packets are received in order, which may not always be the case in real networks. (ii)High space complexity for sorting if many packets arrive out of order, which could impact performance.
[37]	LSTM	(i)The packet-level classification significantly reduces detection time by eliminating flow pre-processing, enabling real-time analysis. (ii)The approach leverages deep learning techniques like LSTM and word embedding to effectively capture semantic and temporal packet features.	(i)The model may struggle with packets lacking sufficient contextual information, potentially reducing classification accuracy. (ii)Implementing and training LSTM models requires substantial computational resources and expertise.
[38]	RBM	RBMs can effectively classify normal and anomalous network traffic, with the ability to recognize novel attack patterns by learning from high-dimensional data.	The computational complexity of scaling RBMs to deeper architectures (e.g., Deep Restricted Boltzmann Machines) can be a barrier to real-time application, requiring significant computational resources.
[39]	RNN	(i)The framework utilizes RNNs (LSTM, GRU, Simple RNN) and XGBoost for feature selection, leading to higher test accuracy (TAC) and validation accuracy (VAC). (ii)The XGBoost-based feature selection method reduces feature space, focusing on the most relevant attributes and improving classification accuracy and efficiency.	(i)The use of advanced models increases computational complexity, which may require significant computational resources and longer training times. (ii)The system may face difficulties in handling imbalanced datasets and accurately classifying minority classes without further optimization.
[40]	DBN	(i)The joint optimization algorithm improves average classification accuracy by up to 14.80% and reduces detection time by at least 24.69%. (ii)Combines PSO with fish swarm behavior and genetic algorithms to effectively optimize DBN network structures, leading to improved detection speed and accuracy.	(i)The method may be constrained by the maximum number of hidden layers, potentially impacting training time and model performance. (ii)The increase in average training time by 6.9% may affect practical deployment and efficiency, especially with larger datasets or more complex models.
[41]	DBN	(i)It efficiently process and analyze large-scale network traffic data, improving detection performance in large environments. (ii)The DBN effectively reduces data dimensionality and extracts informative features, enhancing the predictive accuracy	(i)Implementing and managing a distributed system with deep learning and ensemble can be complex, requiring significant computational resources and expertise. (ii)The distributed nature of the system may raise privacy and data security concerns,.
[42]	MLP	(i)The IGRF-RFE method effectively balances speed and accuracy by combining filter and wrapper approaches, leading to improved anomaly detection. (ii)It reduces the feature set size significantly while maintaining or improving model performance, making it suitable for large datasets.	(i)The method may have higher computational complexity compared to standalone feature selection methods due to the dual-phase process. (ii)The approach may still require optimization for feature selection when applied to highly imbalanced datasets without additional resampling techniques.

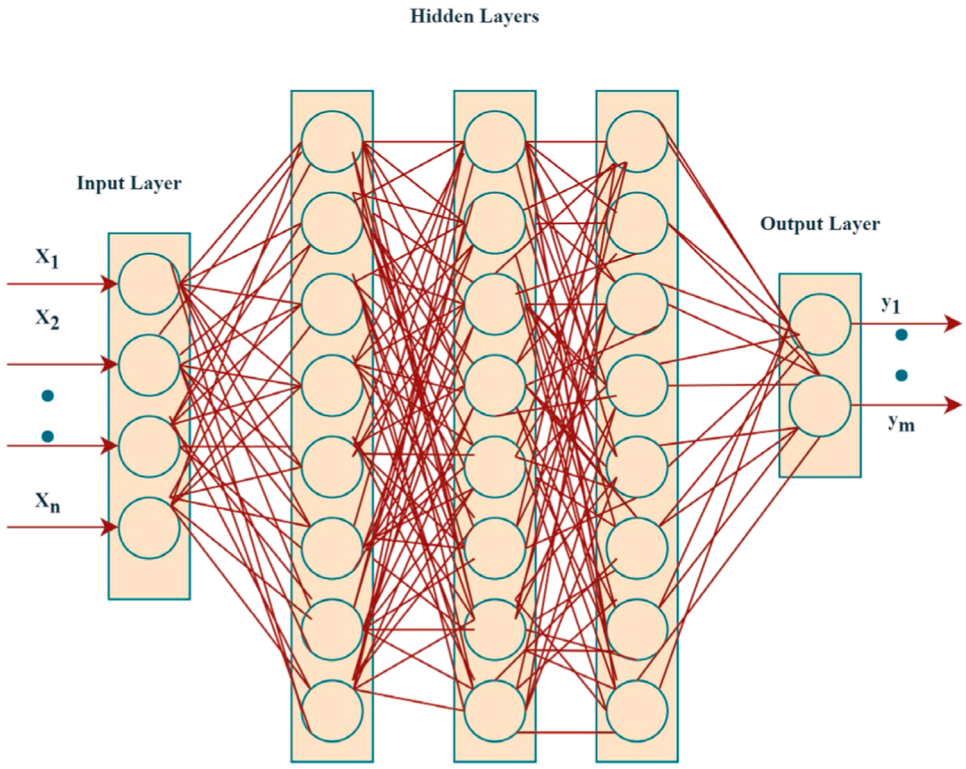

Fig. 9. General block diagram of DNN.

These capabilities demonstrate the scalability, adaptability, and practical significance of DNNs in enhancing the security of real-world networks [47].

4.3. Convolutional neural network (CNN)

Convolution neural network is a kind of deep neural network designed to process and analyze grid-like data structures [48]. CNNs are composed of multiple layers, including convolutional layers, pooling layers, and fully connected layers as shown in Fig. 10. The convolutional layers apply filters to the input data, capturing spatial hierarchies and local patterns, which are crucial for tasks like image recognition and classification [49]. Pooling layers reduce the spatial dimensions, making the network more computationally efficient and less prone to overfitting [49]. The fully connected layers at the end of the network combine these features to produce the final output, making CNNs highly effective for complex tasks [48].

Anand et al. [48] established a deep learning approach named CNN-DMA to discover any malicious intrusion by using convolutional neural network with three layers (dropout, dense, flat). The suggested model is 99% accurate. The data are preprocessed to send only fruitful input to the CNN classifier that discovers any malicious compromise in the network [25]. Besides false alarm rate is reduced significantly. Usman et al. [49] established "an anomaly driven IDS by using the CNN model". The suggested model any possible malicious breach and aberrant network traffic. Pandey et al. [34] applied the combination of multichannel autoencoder and CNN "to classify the network traffic as the normal and" aberrant traffic. The CNN studies the probable relationship among channel to differentiate between normal and breach traffic. The experimental results showed that the false positive rate is decreased, and the accuracy of detection is increased with the application of ensemble model. Kim et al. [50] developed AI driven IDS by encapsulating CNN and LSTM model to improve the detection accuracy. The system is flexible and scalable and is developed using docker images. It helps to imprint and meliorate snort rules for signature driven IDS. Hasan et al. [35] experimented a "hybrid deep learning model with an ensemble of CNN and weight-dropped long short-term memory (WDLSTM)". CNN extracted the useful features from the training data. The long-term dependencies are retained with the help of WDLSTM. Abaimov and Binachi [51] conceived of an attack detection mechanism named CODDLE to detect code injection attack originated in web. It employs a customized preprocessing step that transform the SQL/XSS symbols to type/value pairs and apply the preprocessed input to convolutional deep neural network to improve the efficiency of the model. Naseer and Saleem [52] developed an IDS based on deep convolutional neural network (DCNN). The experimental results showed that DCNN driven IDS provides good results for anomaly detection.

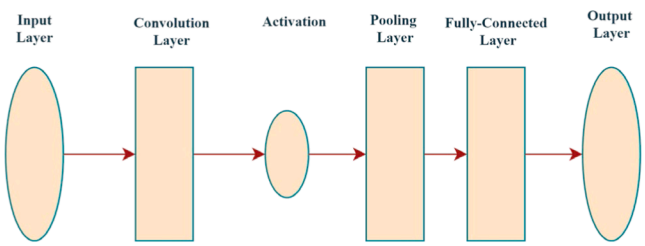

Fig. 10. General block diagram of CNN.

Doriguzzi et al. [53] created a thin DDoS identification system named LUCID using CNN with less processing burden. A preprocessing mechanism called dataset-agnostic has been employed to generate traffic monitoring for online breach identification. Closest group-based convolutional neural network has been developed to find any traces of malicious activity in a network [54].The CNN is trained using two-dimensional image representation of network traffic. "Riyaz and Ganapathy [55] developed a feature identification algorithm named conditional random field and linear correlation coefficient-based algorithm to identify the relevant features". The selected features are given as input to the CNN algorithm to effectively detect any malicious compromise. Wang et al. [56] "developed an IDS using a deep

multi-scale convolutional neural network" that reduces false alarm rate and improve the convergence speed. Said et al. [57] developed a hybrid intrusion detection system that combines Convolutional Neural Networks (CNN) and bidirectional Long Short-Term Memory (BiLSTM) networks, utilizing both binary and multiclass classification approaches to address the security threat in software defined network(SDN). The effectiveness of this model was validated using popular datasets like UNSW-NB15 and NSL-KDD, as well as the InSDN dataset specifically designed for SDN, demonstrating high accuracy and reduced training time. Imrana et al. [58] introduced a novel method, CNN-GRU-FF, which employs a double-layer feature extraction and fusion technique with a modified focal loss function to address the class imbalance problem prevalent in IDS datasets. The effectiveness of the proposed model is demonstrated using the NSL-KDD and UNSW-NB15 datasets, achieving high detection rates of 98.22% and 99.68% respectively, while maintaining low false alarm rates.

In summary, CNN have proven to be highly effective in network-based intrusion detection systems due to their ability to process and analyze complex, grid-like data structures, such as network traffic patterns [51]. CNNs have been successfully applied in various real-world contexts, including anomaly detection in IoT networks, cloud infrastructures, and software-defined networks (SDN) [58]. Their capacity to reduce false positive rates and enhance detection accuracy makes them a valuable tool in identifying malicious network activities such as Distributed Denial-of-Service (DDoS) attacks and code injection threats [52]. The flexibility of CNNs, especially when combined with other deep learning techniques like Long Short-Term Memory (LSTM) or ensemble methods, has made them ideal for improving real-time detection in dynamic environments [53]. These models have been implemented in industrial applications such as traffic monitoring in critical infrastructure, web-based attack detection, and advanced threat detection in cloud and edge computing environments [54]. As NIDS continue to evolve, CNNs will play an increasingly important role in efficiently analyzing vast volumes of network data while maintaining scalability and high detection performance [56].

4.4. Recurrent neural network (RNN)

RNN algorithm is one of the deep learning approaches and has internal memory to remember input and is perfectly suited for learning from sequence data [21]. Unlike traditional feedforward neural networks, RNNs incorporate loops as shown in Fig. 11, that allow information to persist, enabling them to capture dependencies across time steps. This unique capability allows RNNs to model temporal dynamics and context, which is crucial for tasks requiring sequence prediction or understanding [59].

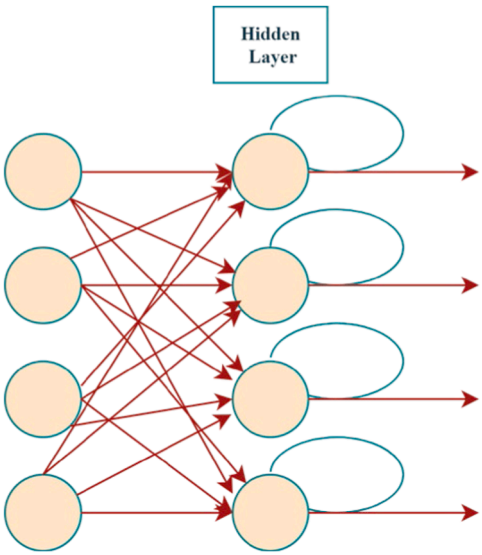

Fig. 11. General block diagram of RNN.

Ahmad et al. [59] experimented various hybrid approaches for intrusion detection. They grouped auto encoder with bidirectional recurrent neural network. Experimental results revealed that the performance of this hybrid classifier is poor when comparing to other hybrid models. Kasango [39] "suggested an IDS using Recurrent Neural Network (RNN)". Experiments have been conducted on ensembled model and the results expressed that RNN outperforms all other techniques Le et al. [60] designed an IDS to avert the problem of low detection accuracy in finding "User-to-Root (U2R) and Remote-to-Local (R2L) attacks. Firstly, they employed SFSDT feature selection algorithm to select the optimal feature". Secondly, various RNN models like traditional RNN, "Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU)" were used to build IDS. Finally, the proposed system is tested with various data sets to obtain performance measures. Yue et al. [61] developed ensemble intrusion detection method tailored for the Train Ethernet Consist Network (ECN), targeting specific attacks like IP Scan, Port Scan, Denial of Service (DoS), and Man in the Middle (MITM). The method involves extracting 34 features from protocol contents to form a specialized dataset, which is then optimized through a data imaging method and temporal sequence building. Utilizing a combination of convolutional and recurrent neural networks, including LeNet-5, AlexNet, VGGNet, SimpleRNN, LSTM, and GRU, the approach employs a dynamic weight matrix voting method to integrate these classifiers, achieving a high detection accuracy of 0.975 [61].

In summary, RNNs have demonstrated significant potential in NIDS due to their ability to capture temporal dependencies and model sequential data [59]. Their internal memory allows them to retain information over time, making them highly effective for detecting complex patterns in evolving network traffic. RNN-based models have been successfully applied in real-world areas such as detecting advanced attacks like User-to-Root (U2R) and Remote-to-Local (R2L) intrusions, as well as identifying malicious activities like IP scanning, Denial of Service (DoS) attacks, and Man-in-the-Middle (MITM) attacks [61]. These models are particularly useful in dynamic environments, including train networks and industrial control systems, where they can analyze real-time traffic and recognize subtle variations that might indicate security breaches [60]. Furthermore, when combined with other deep learning techniques, such as CNNs, LSTM, and GRU architectures, RNNs have enhanced intrusion detection accuracy, offering robust solutions for both enterprise and critical infrastructure networks [39].

4.5. Long short-term memory (LSTM)

A specific type of recurrent neural networks is called LSTM. It is designed to overcome the limitations of standard RNNs in learning long-term dependencies. LSTM networks achieve this by using a unique architecture that includes memory cells and gating mechanisms, which regulate the flow of information and enable the network to retain important information over extended sequences [62]. These gates—input, forget, and output as shown in Fig. 12 allow LSTMs to effectively manage the problem of vanishing gradients. With its feedback connection, it can be applied in mining complex data types such as sequence da ta and visual data. It is further divided into stacked LSTM and Bidirectional LSTM [62].

Islam ed al. [62] developed an IDS using "LSTM, stacked LSTM and Bi-LSTM. Experimentation on the NSL-KDD, IoTDevNet" and DS2OS dataset reveals that BiLSTM exhibits highest performance when comparing to all other DL models. Hanaf et al. [63] suggested an enhancement to Golden Jackal Optimization Algorithm (IGJO) and used IGJO for feature selection and designed IDS models using LSTM for attack categorization. The models are trained both with normal and adversarial samples. Han and Pak [64] applied LSTM with two-stage for effective categorization of incoming network flow. It achieves high

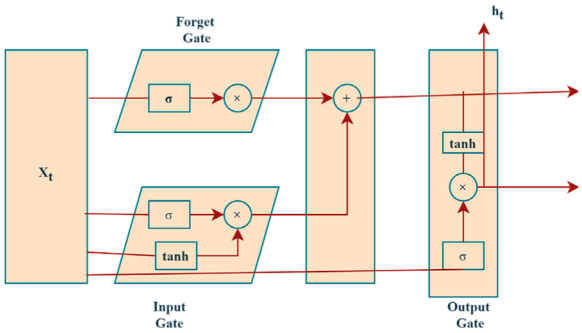

Fig. 12. General Block diagram of LSTM.

detection rate with less memory usage. Fu et al. [36] designed three IDS models using LSTM, CNN and GRU.

They produced attack samples using Fast Gradient Sign Method (FGSM). The models are trained both with normal and adversarial samples. Mahdavisharif et al. [65] designed a particular architecture of LSTM called Big data aware deep learning IDS that increases detection accuracy while lowering the false alarm rate. Jallad et al. [66] designed LSTM based IDS to achieve good performance as it can act with sequences of events and context. To overcome the intrusion categorization problem, Sahu et al. [67] proposed an IDS based on LSTM and Fully Connected Network(FCN).This model can accurately categorize multi-class attack strategies. Hwang et al. [68] developed a model with word embedding technique to get packet semantics and LSTM to study the temporal relation among packet header fields. Bukhari et al. [37] constructed a novel Intrusion Detection System (IDS) based on a Stacked Convolutional Neural Network and Bidirectional Long Short Term Memory (SCNN-Bi-LSTM) model to address the growing cybersecurity challenges in Wireless Sensor Networks (WSNs). The model was specifically designed to detect and categorize various Denial of Service (DoS) attacks using the WSN-DS and CIC-IDS-2017 datasets. It demonstrated superior performance compared to traditional Artificial Deep Neural Network (ADNN) models, achieving approximately 99.9% precision and recall, and significantly reducing false positives and negatives.

In summary, LSTM networks have become a powerful tool in NIDS due to their ability to capture long-term dependencies and model sequential data effectively [63]. By incorporating memory cells and gating mechanisms, LSTMs overcome the limitations of standard RNNs and can manage complex patterns in dynamic network traffic [62]. LSTM-based models have been applied across various real-world scenarios, including wireless sensor networks, IoT devices, and cloud infrastructures. These models demonstrate high detection accuracy, particularly in identifying sophisticated attacks like Denial of Service (DoS), multi-class attack strategies, and adversarial threats [65]. Stacked and bidirectional LSTM variants have proven especially useful in enhancing performance, reducing false positives, and improving generalization in real-time detection environments [64,65]. Furthermore, LSTMs, when combined with other techniques like convolutional networks or integrated with advanced optimization algorithms, show exceptional promise in protecting modern networks from emerging cyber threats [67].

4.6. Restricted boltzmann machine (RBM)

RBM is a generative, unsupervised, and probabilistic deep learning algorithm that can accelerate log likelihood function [21]. They consist of a visible layer and a hidden layer as shown in Fig. 13, with symmetrical connections between the layers but no connections within a layer, making them restricted [69]. RBMs learn to model the distribution of input data by adjusting the weights between visible and hidden units through contrastive divergence [70]. This structure allows RBMs to

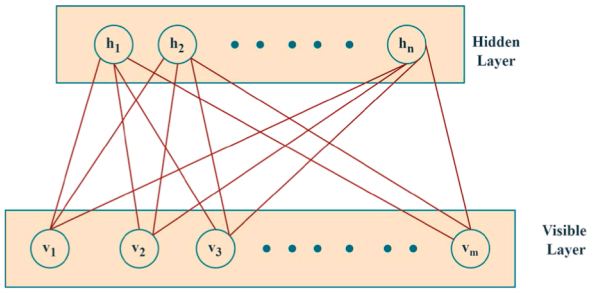

Fig. 13. General block diagram of RBM.

capture complex patterns in data, making them useful as building blocks in more advanced architectures like Deep Belief Networks (DBNs) [69].

Atefinia and Ahmadi [71] proposed a multi architectural IDS model with a stack of restricted Boltzmann machine, feed forward module and two recurrent modules. Rosa et al. [69] developed an IDS using restricted Boltzmann machine as a projection algorithm to effectively detect malicious attempts. The RBM is used to identify unforeseen attacks effectively [70].They used balanced training set to lessen any bias during learning. Elsaeidy et al. [38] proposed an intrusion detection framework for smart cities using Restricted Boltzmann Machines (RBMs) to effectively detect Distributed Denial of Service (DDoS) attacks. RBMs are utilized for their ability to learn high-level features from raw data generated by smart meters and sensors in an unsupervised manner, enhancing the accuracy of the detection process. The methodology, tested on a smart water distribution plant dataset, demonstrates high efficiency and superior performance compared to models without the feature learning step. Aldwairi et el. [72] conducted an experimental research that explored the impact of feature selection on the performance of anomaly network intrusion detection systems, particularly focusing on Restricted Boltzmann Machines (RBMs). The research showed that accurate feature selection can significantly influence the accuracy of classifiers, with changes in accuracy remaining under 3% across all algorithms. Mathappan et al. [73] The paper proposes a Hybrid Intelligent Intrusion Detection System (IDS) to detect various Wi-Fi attacks, including impersonation, injection, and flooding, using Deep Learning and Machine Learning algorithms. The system leverages Deep Belief Networks (DBN) with stacked Restricted Boltzmann Machines (RBM) for feature representation and dimensionality reduction, and applies optimization strategies to enhance classification performance. A two-pronged approach combining Restricted Boltzmann Machines (RBMs) with Chaotic Ant Optimization (CAO) to enhance the performance and reliability of Wireless Sensor Networks (WSNs) is proposed [74]. By optimizing confidence levels for each sensor node, this method addresses challenges in intrusion detection and energy optimization, surpassing state-of-the-art techniques.

In summary, RBMs have shown great potential in the realm of NIDS due to their ability to capture complex patterns in high-dimensional data [38]. As a generative and unsupervised learning model, RBMs are highly effective at modeling the distribution of network traffic data, which makes them useful in identifying anomalies and unforeseen attacks. Real-world applications of RBMs in NIDS include their use in smart city infrastructures to detect DDoS attacks by learning high-level features from raw sensor data [70]. Furthermore, RBMs have been integrated into more advanced architectures, such as Deep Belief Networks (DBNs) and hybrid systems, where they contribute to feature representation and dimensionality reduction, enhancing detection accuracy [72]. Their role in optimizing wireless sensor networks (WSNs) and improving energy efficiency further demonstrates their versatility in addressing modern cybersecurity challenges [69]. By combining RBMs with techniques like feature selection and optimization strategies, these models offer robust solutions for intrusion detection in complex, real-time network environments [38].

4.7. Deep belief network (DBN)

Deep belief network is a generative graphical model that multiple layers with values and there exists a relationship between layers, often implemented using stacked Restricted Boltzmann Machines (RBMs) as shown in Fig. 14 [21]. Each layer in a DBN learns to capture complex, hierarchical features from the data, with lower layers focusing on more fundamental patterns and higher layers representing more abstract concepts [75]. DBNs are pre-trained in a layer-by-layer manner using unsupervised learning, followed by fine-tuning through supervised learning, which improves their ability to generalize from data [76]. This combination of unsupervised pre-training and supervised fine-tuning allows DBNs to achieve high performance [77].

Nguyen et al. [75] planned IDS model in which the sensor devices acquire the data cyber physical system and DBN is used to detect any malicious threat. Alsirhani et al. [76] developed an anomaly detection technique where African vulture optimization is used to select the optimal features. Those features train the DBN. Balakrishnan et al. [77] defined a security network with improved DBN model. The smart IDS scrutinizes malicious threat inside the network and defend against it. Modified density peak clustering algorithm (MDPCA) and deep belief networks (DBNs)" are joined to form an effective IDS [78]. Wei et al. [79] conceived a model that enhance the DBN network structure so that the classification performance is enhanced. This model employed joint optimization algorithms to boost the DBN network structure.

Zhang et al. [40] designed a network attack detection method that combines flow calculations with deep learning, utilizing a real-time detection algorithm based on sliding window (SW) stream processing and a classification algorithm combining deep belief networks and support vector machines (DBN-SVM). The approach improves classification accuracy and real-time detection efficiency, outperforming traditional machine learning methods. Sarkar et al. [80] proposed an Intrusion Detection System (IDS) for cloud environments using an Improved Squirrel Search Algorithm (ISSA) for feature selection and a Modified Deep Belief Network (MDBN) for anomaly detection. The ISSA effectively reduces the dataset's dimensionality, while the MDBN handles both binary and multi-class classifications, improving performance on unbalanced datasets. The proposed ISSA-MDBN model achieves significant reductions in training and testing times and outperforms recent approaches, with a high accuracy of 99.8% and a low false alarm rate of 0.02% on the UNSW-NB15 dataset. An intrusion detection system that integrates Deep Belief Networks with Particle Swarm Optimization to classify intrusions into categories such as Normal, Probe, DoS, U2R, and R2L, using the DARPA 1999 dataset is proposed [81]. The proposed system outperforms other models like ANFIS, HHO, and Fuzzy GNP, achieving a high accuracy of 96.5%. Marir et al. [82] presented a novel distributed approach for detecting abnormal behavior in large-scale networks by combining deep feature extraction with multi-layer ensemble support vector machines (SVMs). The approach involves non-linear dimensionality reduction using distributed deep belief networks on large-scale network traffic data, followed by classification with an ensemble of SVMs constructed through the iterative reduce paradigm on Spark.

In conclusion, DBNs offer a robust solution for NIDS due to their ability to model complex, hierarchical features and learn from data through both unsupervised pre-training and supervised fine-tuning [76]. By leveraging stacked Restricted Boltzmann Machines (RBMs), DBNs effectively detect malicious activities and potential security threats in dynamic network environments. Real-world applications of DBNs include their use in cyber-physical systems and cloud environments, where they are employed to identify and defend against a wide range of network attacks [82]. DBNs have also been integrated with optimization algorithms, such as African vulture optimization and particle swarm optimization, to enhance feature selection and classification accuracy, demonstrating superior performance in anomaly detection [40]. Moreover, DBN-based models, when combined with other techniques like support vector machines (SVMs) and clustering algorithms, excel in large-scale network traffic analysis, ensuring high accuracy and real-time detection [79]. These applications highlight the versatility of DBNs in addressing modern cybersecurity challenges across diverse networked environments [81].

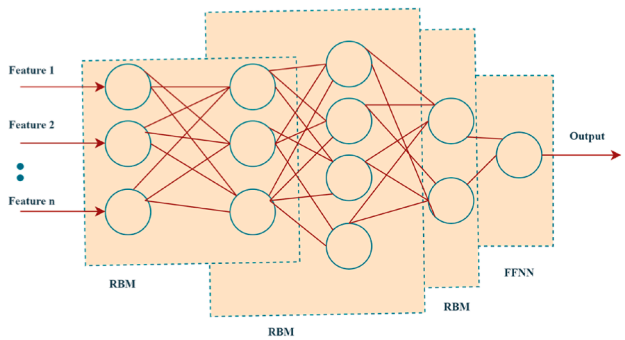

Fig. 14. General block diagram of DBF.

4.8. Multi-Layer perceptron (MLP)

A Multilayer perceptron is a supervised deep learning approach with neurons of multiple layers [21]. The MLP uses backpropagation technique for learning and constructing the model. It consists of an input layer, one or more hidden layers, and an output layer as shown in Fig. 15, making it a foundational model in deep learning. Each neuron in an MLP applies a weighted sum of its inputs followed by a non-linear activation function, enabling the network to learn complex patterns and relationships in data [21]. MLPs are fully connected, meaning every neuron in one layer is connected to every neuron in the next, which allows them to approximate any continuous function given sufficient data and layers [21]. They are widely used for tasks such as classification, regression, and function approximation, making them versatile tools in both research and practical applications [3]. Despite their simplicity compared to more advanced architectures, MLPs remain a critical component in the study and development of neural networks [21].

Ashraf et al. [41] devised a model that encapsulate multilayer perceptron (MLP) and random Forest (RF) to improve the intrusion detection performance. The results proved that the suggested model detects R2L and U2R threat types with improved accuracy when comparing to the benchmark techniques. Ghanem and Jantan [83] devised a method that clustered enhanced Bat algorithm with Multilayer Perceptron to increase the accuracy. Nine optimization algorithms have been used to train the EBATMLP-IDS frame work which improves the detection accuracy substantially. Alazab et al. [84] introduced an Intrusion Detection System (IDS) that utilizes the Harris Hawks Optimization (HHO) algorithm to optimize the learning parameters of a Multilayer Perceptron (MLP), aiming to reduce intrusion detection errors. Implemented using the EvoloPy NN framework, the HHO-MLP model has been evaluated against other evolutionary algorithms like BOA, GOA, and BOW using the KDD dataset, demonstrating better performance. Yin et al. [85]

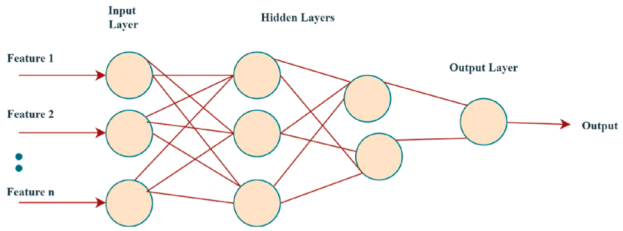

Fig. 15. General block diagram of MLP.

introduced IGRF-RFE, a hybrid feature selection method that combines filter and wrapper techniques to enhance multi-class network anomaly detection. In the first phase, the method uses information gain (IG) and random forest (RF) to narrow down the feature subset, effectively managing less important features. In the second phase, a recursive feature elimination (RFE) approach is applied to further refine feature dimensions, resulting in improved MLP classification accuracy. A hybrid Support Vector Machine and Multilayer Perceptron Neural Network (SVMNN) algorithm is designed to detect cyber intrusions in power systems [42]. It uses modified IEEE Garver 6-bus and Nigerian 24-bus test systems to simulate attack scenarios and evaluate the algorithm's performance on real-time data. The hybrid SVMNN algorithm achieved a detection accuracy of 99.6%, outperforming existing methods in addressing cyber threats to power system networks.

In conclusion, MLP models remain a foundational approach in network-based intrusion detection systems, owing to their ability to learn complex patterns through fully connected layers and back-propagation [41]. Despite the rise of more advanced architectures, MLPs continue to demonstrate their effectiveness in detecting network anomalies, including R2L and U2R threats. Real-world applications of MLPs in NIDS include hybrid approaches that integrate MLP with optimization algorithms such as HHO, Bat Algorithm, and Random Forest, which enhance the model's accuracy and reduce false positives [85]. MLP-based models have also been successfully applied in power systems to detect cyber intrusions in real-time, highlighting their versatility across different domains. When combined with feature selection techniques and hybrid models, MLPs further improve classification accuracy and intrusion detection performance, making them valuable tools for safeguarding modern network infrastructures [83].

4.9. Ensemble methods

Ensemble deep learning techniques uses multiple neural network models to enhance the performance and accuracy of Network Intrusion Detection Systems (NIDS) by combining their individual predictions [61]. These techniques integrate diverse architectures, such as CNN, RNN and DBN, to capture a broader spectrum of attack patterns and anomalies in network traffic [86]. The fusion of multiple deep learning models in ensemble systems helps to utilize the strengths of different learning algorithms, resulting in improved generalization and adaptability to evolving cyber threats and network environments [61]. Implementing ensemble techniques in deep learning-based NIDS not only increases detection performance but also enhances the system's ability to handle class imbalance and detect previously unseen attack vectors, contributing to a more comprehensive security solution [41].

5. Data preparation

Data Preparation also known as data preprocessing is the process of cleaning, transforming, and organizing raw data into a structured format that is suitable for analysis, modeling, or visualization [21]. It is a crucial step in the data analysis process, including for Network Intrusion Detection Systems (NIDS) [86].In the context of NIDS, the most commonly used data preparation activities are data cleaning, data encoding, data normalization, data augmentation and feature selection as shown in Fig. 16.

5.1. Dataset merging

Normally, in NIDS environment, the dataset is very large in size [54]. So, it is stored in more than one file. For building a model, the datasets must be merged and should form a single data frame. This is the primitive step in the development of NIDS [31].

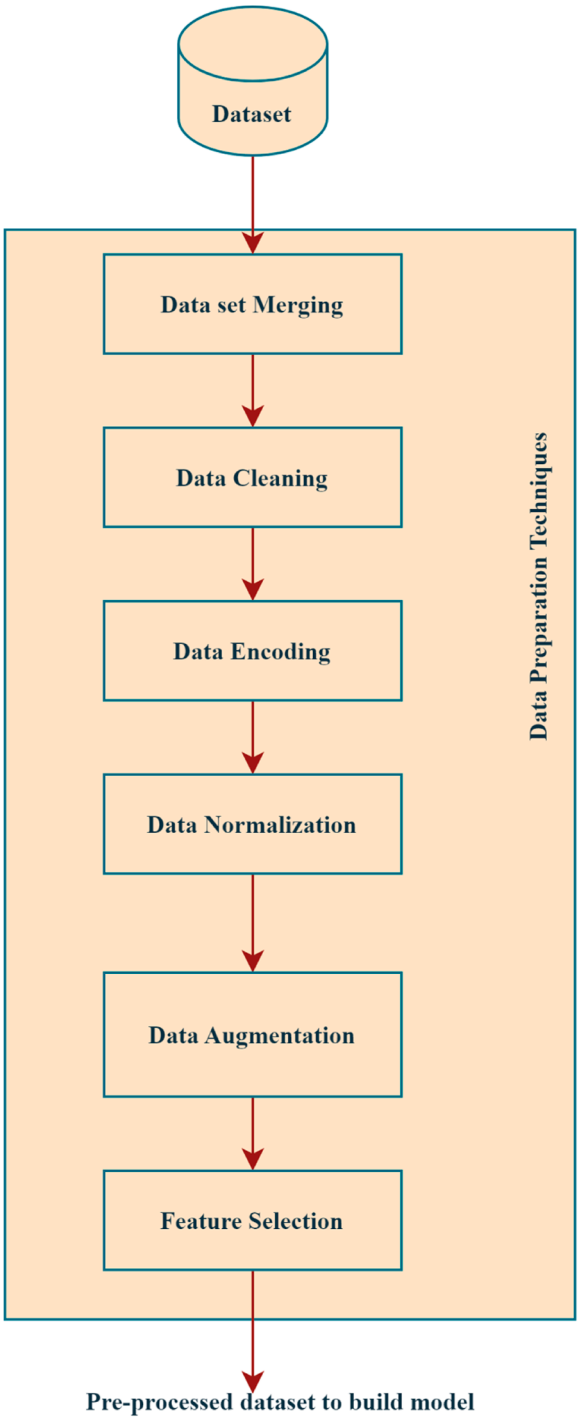

Fig. 16. General data preparation techniques for NIDS.

5.2. Data cleaning

Data cleaning is a process of identifying and correcting errors, inconsistencies, and inaccuracies in a dataset [21]. It includes activities such as removing duplicates, handling missing values, identifying and fixing errors such as typos, incorrect entries, and inconsistencies in data formats, and handling outliers [44,59,71].

5.3. Data encoding

Data encoding is the process of converting categorical or textual data into numerical formats that can be used by machine learning algorithms

[21]. In (NIDS), data encoding is important for converting categorical network traffic features into numerical formats so that it can be processed by machine learning algorithms [33,46]. NIDS often deal with various types of network data, including protocol types, service types, IP addresses, and more, which are typically categorical in nature [51]. Encoding these features correctly is essential for building effective models that can detect intrusions or anomalies [55,58]. The most common encoding methods employed in the construction of NIDS are

(i) Label Encoding converts categorical data into integer labels. Each unique category is assigned a unique integer value. The Label encoding may be used for encoding various attack categories. [87]
(ii) One-hot encoding [88] converts categorical data into binary vectors where each category is represented by a vector with a single 1 and the rest 0s.

5.4. Data normalization

Data normalization is a preprocessing technique used to scale numerical data into a specific range, typically between 0 and 1, or -1 and 1 [21]. This process ensures that different features contribute equally to the analysis or machine learning model, preventing features with larger ranges from dominating those with smaller ranges [37,68]. In NIDS, data normalization is essential for ensuring that features with varying scales contribute equally to the model's learning process. Among the normalization techniques, Min-Max normalization and Z-Score normalization are the most commonly used normalization techniques for NIDS [70].

(i) Min-Max Normalization:

Min-Max Normalization scales the data to a fixed range, usually between 0 and 1, or sometimes between -1 and 1 [47,84]. The formula used for Min-Max normalization is given Eq. 4 [10].

$$X' = \frac{X - X_{Min}}{X_{Max} - X_{Min}} \tag{4}$$

Where X' is the normalized value
X is the original value
X_{Min} is the minimum value of the feature
X_{Max} is the maximum value of the feature

(ii) Z-Score Normalization:

Z-Score Normalization, transforms the data into a distribution with a mean of 0 and a standard deviation of 1 [21]. The formula of the Z-score normalization is given in Eq. 5.

$$Z = \frac{X - \mu}{\sigma} \tag{5}$$

Where, Z is the standardized value,
X is the original value
μ is the mean of the feature
σ is the standard deviation of the feature

5.5. Data augmentation

Data augmentation is a technique used to artificially increase the size and diversity of a dataset by creating modified versions of existing data [89]. The robustness and generalization capabilities of machine learning models is improved by using data augmentation techniques particularly when dealing with limited or imbalanced data [21]. In the context of Network Intrusion Detection Systems (NIDS), data augmentation is used to generate synthetic data that mimics real network traffic, enhancing the ability of models to detect various types of intrusions [90]. The most commonly used data augmentation techniques for the development of effective NIDS are Generative Adversarial Networks (GAN) and it's variations, Synthetic Minority Over-sampling Technique (SMOTE), data resampling and ADASYN [21].

Srivastava et al. [89] proposed an NIDS that applies a Wasserstein Conditional Generative Adversarial Network with Gradient Penalty (WCGAN-GP) to address the class imbalance problem by generating synthetic samples for minority classes in the NSL-KDD and UNSW-NB15 datasets. The WCGAN-GP model effectively generates high-quality, realistic samples, ensuring better training for machine learning models, particularly in recognizing minority class instances. This approach helps maintain a balanced data distribution, improving the generalization capability of the learned models. Alabrah [90] applied the ADASYN method to generate synthetic samples to address class imbalance by calculating an imbalance ratio based on k-nearest neighbors and creating new data points using the original minority class samples and their neighbors in the UNSW-NB15 dataset. This process enhances the relevance of synthetic data to the original dataset, helping balance class distributions. Abdelkhalek and Mashaly [91] developed an IDS which addresses the class imbalance in the NSL-KDD dataset to enhance both accuracy and detection rates of minority classes by applying a data preprocessing method that combines ADASYN for oversampling minority classes and TomekLinks for under sampling to remove redundant samples. Liu et al. [92] proposed a NIDS that addresses the challenges in anomaly detection for gas turbines, focusing on class imbalance, inter-class overlap, and complex temporal properties. To overcome these issues, a novel attention-based hybrid re-sampling scheme is proposed, combining clustering-based under-sampling with a DA-SMOTE-ED data augmentation method that generates high-quality synthetic data while preserving inter-class separability. The resulting balanced training set improves the performance of the self-attention classification network for detecting anomalies in gas turbine monitoring data. Andresini et al. [93] applied an Auxiliary Classifier GAN (ACGAN) to generate synthetic images of artificial attacks, balancing the training set for a 2D CNN. The ACGAN architecture, which employs labels for better data augmentation quality, involves a two-player minimax game between the generator and discriminator networks. The trained 2D CNN then leverages these augmented images to improve its ability to distinguish between attack and normal network flow images by capturing spatial patterns in the data. The IGAN approach addresses class imbalance by enhancing minority classes in datasets like UNSW-NB15 and CICIDS 2017 using an improved GAN model. It employs an imbalanced data filter, a discriminator with multilayer perceptron, and a generator with convolutional layers to balance the dataset effectively. The IGAN model generates synthetic samples for minority classes, improving the classifier's performance by integrating these samples into the training data, thus mitigating the effects of class imbalance [94]. The anonymized traffic trace generator utilizes a Conditional Variational Autoencoder (CVAE) to create sequences of benign and attack packets, with both encoding and decoding modules built using Dense Neural Networks. The CVAE model, optimized with a combination of reconstruction loss, Kullback-Leibler divergence, and weight regularization, generates Gaussian-distributed latent representations to produce synthetic traffic data [95]. The proposed XIDINTFL-VAE framework integrates Class-Wise Focal Loss (CWFL) and Variational AutoEncoder (VAE) with XGBoost to enhance the detection of minority class intrusions in imbalanced network traffic, addressing limitations of traditional methods like SMOTE. Evaluated on NSL-KDD and CSE-CIC-IDS2018 datasets, it achieves superior performance with 99.67% precision and 94.74% F1 score, demonstrating reduced false positives and improved detection rates for real-world applications [96].

Overall,Data augmentation is a critical technique in NIDS for handling class imbalance and improving model robustness [89]. Various methods offer unique benefits for different use cases. Generative Adversarial Networks (GANs) are highly effective for generating synthetic attack samples in imbalanced datasets, improving the detection of

rare intrusions, especially in IoT environments or cloud infrastructure where realistic data simulation is essential [89]. Wasserstein Conditional GAN with Gradient Penalty (WCGAN-GP) enhances synthetic data quality, particularly for minority classes in datasets like NSL-KDD and UNSW-NB15, making it ideal for environments such as healthcare or banking, where accurate detection is critical. SMOTE and ADASYN are commonly used for oversampling minority classes, with SMOTE being effective for simplifying model training in traditional NIDS setups and ADASYN offering a more focused approach by targeting hard-to-classify instances, which is particularly useful in real-time NIDS or cloud-based systems [90,91]. Resampling techniques like Tomek Links clean noisy, overlapping data, improving detection precision in environments like smart grids or corporate networks, while attention-based hybrid resampling (e.g., DA-SMOTE-ED) helps improve temporal detection capabilities in sensitive setups like industrial IoT or critical infrastructure. ACGAN and CVAE [94,95] further enhance synthetic data generation by incorporating labels and creating anonymized traffic, making them valuable in privacy-sensitive sectors such as healthcare and finance, or in complex systems like smart cities or vehicular networks. Together, these techniques address different aspects of NIDS challenges, improving detection rates, handling evolving threats, and ensuring the system's adaptability across diverse, real-world network environments [95].

5.6. Feature selection

Feature selection is the process of choosing a subset of relevant features from a larger set to improve model performance and reduce computational complexity [24]. It aims to retain the most informative features while eliminating redundant or irrelevant ones, thus enhancing model accuracy and interpretability [33,79]. There various types of feature selection methods that can be applied to NIDS such as optimization algorithm, wrapper methods, genetic algorithms and embedded methods [97].

Kasongo and sun [97] utilized Extra Trees (ET) algorithm to rank feature importance, which informs the selection of relevant features. The refined feature set is then used to train a feed-forward deep neural network (FFDNN), which undergoes forward and backpropagation to enhance the model's performance. Srivatsava et al. [89] utilized Genetic Algorithms (GAs) based feature selection by iteratively evolving populations of feature subsets, enhancing model performance and computational efficiency. Unlike methods such as PCA or Autoencoders, GAs effectively handles non-linear distributions and evaluate feature relevance using a novel fitness function that combines MeritS (feature correlation with the target) and G-mean (a metric for class imbalance).Liu et al. [98] proposed a feature selection technique using embedded models, specifically Random Forest (RF) and eXtreme Gradient Boosting (XGBoost), to reduce less impactful features in the dataset. By selecting only the most relevant features, the NIDS achieves faster computation times and improved anomaly detection performance. Nugyen et al. [99] introduced a special feature selection process named chaotic butterfly optimization algorithm (CBOA) which a modification of original butterfly algorithm to speed up the convergence. The optimization algorithm that is inspired by the foraging nature of Spider monkeys and have Fission-Fusion social structure are called spider monkey optimization. Khare et al. [100] utilized spider monkey optimization (SMO) algorithm for dimensionality reduction of dataset and the reduced set is fed as input to DNN for training. Firefly algorithm is a nature inspired metaheuristic optimization algorithm and is based on the flashing character of fireflies. Kavousi et al. [101] defined an IDS that utilizes firefly optimization to generate a category of generators in a feasible region. Donkol et al [86]. has developed an IDS that uses particle swarm optimization (PSO) for effective feature selection that can reduce the training time. Accuracy, precision, recall and error rate were the performance metrics. The system is applied on Bot_dataset, CIC-IDS2017, NSL-KDD, CIC-IDS2018 and UNSW-NB15 to test the effectiveness of the system.

Almutairi et al. [102] developed an IDS for cyber-physical systems by using ensemble of deep learning methods with Quantum Dwarf Mongoose Optimization (QDMO) for feature selection. Deep auto encoder, deep belief network and convolution residual network are ensembled to improve the efficiency of the IDS. Hussen et al. [103] proposed a framework for cyber security that uses optimized deep learning techniques. A technique named Hyper parallel optimization (HPO) had been utilized to improve stability and efficiency of big data framework. Deore & Bhosale [104] designed an IDS with the ensemble of LSTM and CNN for classification. Chimp chicken swarm optimization (ChCSO) which is a hybridization of chimp and chicken swarm optimization algorithm has been employed for feature selection. Alamro et al. [105] suggested an IDS for IoT health care systems by utilizing immutable and decentralized behavior of block chain technique. Subset of feature selection is made by utilizing ant lion optimizer (ALO) techniques. The detection rate is improved by tuning the hyper parameter of deep learning algorithm using the hyper pollination optimization algorithm. An IDS has been proposed for IoT environment using hybrid deep learning methods [106]. Honey badger optimization Algorithm (HBO) has been utilized for the efficient feature selection. The suggested model performs better in comparison with benchmark algorithms [107].

Saheed and Arowolo [108] proposed an IDS that detects intrusion in Internet of Medical Things (IoMT). Particle swarm optimization (PSO) has been employed for efficient feature selection and deep RNN has been utilized for classification. Lahasan and Samma [109] developed an IDS with deep autoencoder technique and utilized "Reinforcement Learning-based Memetic Particle Swarm Optimization (RLMPSO)" for feature selection. Many optimization algorithms such as Modified genetic algorithm [110], Ant Colony Optimization [111], Fruitfly optimization [112], Butterfly Optimization Algorithm [113], Emperor penguin colony optimization [114], Horse herd optimization [115] and cuckoo search optimization [116] has been applied for efficient feature selection in the construction of an effective IDS. Fig. 17 shows the comparison of performances of various hybrid and ensemble algorithms with optimization and deep learning techniques.

To summarize, Data preparation is a critical step in constructing NIDS, involving data cleaning, encoding, normalization, augmentation, and feature selection [24,46,80]. Data cleaning ensures the accuracy and consistency of datasets, while data encoding transforms categorical data into numerical formats suitable for machine learning algorithms [51,62]. Data normalization scale features to ensure balanced contributions during model training, and data augmentation expands the dataset by generating synthetic samples, improving model robustness

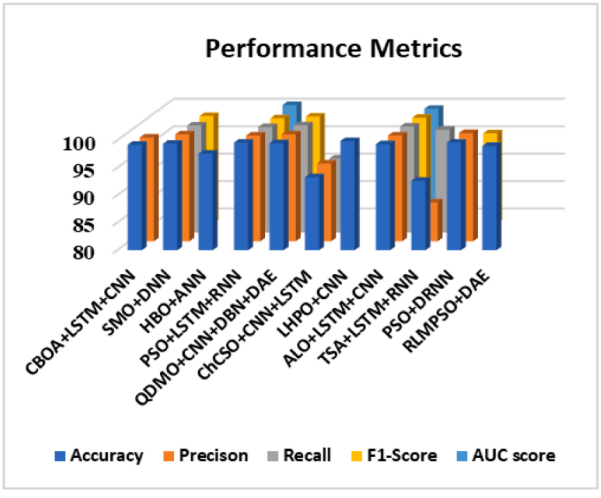

Fig. 17. Comparison of performance for various DL algorithm with feature selection.

[89]. Feature selection techniques, including various optimization algorithms and embedded methods, enhance model performance by selecting the most relevant features and reducing computational complexity [99,108,110]. These steps collectively improve the efficiency and accuracy of NIDS in detecting cyber threats. Table 9 presents the Applications, Benefits, and Use Cases of various data preparation techniques.

6. Performance assessment metrics

This section discusses the commonly used performance metrics for evaluating an IDS. These metrics are crucial for analyzing the efficacy of proposed methodologies across diverse scenarios [43]. It also provides an analytical discussion on the applicability of each metric, factors contributing to high metric values, and the implications of these values in real-world contexts [55,71,110]. Confusion matrix is the basis for all the performance assessment metrics that is a two-dimensional matrix between the actual and anticipated class [10] and incorporate the below details.

(i) True Positive (TP): The instances of data that are rightly classified as an attack.
(ii) False Positive (FP): The instances of data that are incorrectly classified as an attack.
(iii) False Negative (FN): The instances of data that are incorrectly classified as regular instances.
(iv) True Negative (TN): The instances of data that are rightly predicted as regular.

Table 10 depicts a confusion matrix where the diagonal elements represent the correct prediction and the nondiagonal element represents the incorrect prediction [42]. By using the confusion matrix, various performance measures have been proposed. They are,

Table 9
Data Preparation Techniques in NIDS: Applications, Benefits, and Use Cases.

References	Data Preparation Technique	Appropriate Scenarios	Applicable Areas	Significance
[29,31,32, 35,50, 54]	Dataset Merging	(i)When datasets are stored across multiple files. (ii) During the initial step of model building.	(i) Large-scale NIDS datasets (e.g., CIC-IDS2017, NSL-KDD).	(i)Ensures consistency and completeness for model training and evaluation.
[22–116]	Data Cleaning	(i) When datasets contain missing values, duplicates, or outliers.	(i)Any dataset with noise, such as logs from heterogeneous environments.	(i) Improves dataset quality, ensuring accurate analysis and robust model training.
[22–116]	Data Encoding	(i)When datasets include non-numeric features like protocols or attack types.	(i) Protocol types (e.g., TCP, UDP), service types, and attack categories in NIDS datasets.	(i)Makes categorical data usable for machine learning models. (ii)Facilitates better model performance.
[22–116]	Data Normalization	(i)When numerical features have varying scales.	(i)Features like packet size, traffic volume, or time intervals in NIDS datasets.	(i)Prevents large-scale features from dominating smaller ones, ensuring balanced model training.
[89–96]	Data Augmentation	(i)When datasets are imbalanced (e.g., minority attack classes).	(i)Imbalanced datasets like UNSW-NB15 or CIC-IDS2017. (ii) IoT or healthcare datasets with limited samples.	(i)Addresses class imbalance, improving model performance in detecting rare attacks.
[97–116]	Feature Selection	(i)When datasets contain redundant or irrelevant features.	(i) High-dimensional datasets (e.g., CIC-IDS2017, AWID,H23Q).	(i)Reduces overfitting. (ii) Enhances interpretability (iii)Improves computational efficiency.

6.1. Accuracy

Accuracy refers to the proportion of correctly identified instances out of the total instances evaluated [22]. The accuracy can be calculated with the equation

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

While it is a primary metric, it is most effective for balanced datasets where attack and normal traffic proportions are similar [44,71]. For imbalanced datasets, accuracy can be misleading, as high accuracy may mask poor detection of minority class attacks [26]. Therefore, in scenarios such as large-scale public cloud networks, accuracy can provide a general overview but must be complemented with other metrics like recall and precision [63,77,83].

6.2. Precision

The ratio of rightly anticipated attacks against all the instances that are anticipated as attacks is called precision [48]. It can be calculated with the following equation.

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

It is particularly important in environments like healthcare or finance, where false positives can disrupt operations or lead to significant costs [48,78]. High precision indicates fewer false positives, but achieving this may reduce recall, which can be detrimental in critical systems [22,44].

6.3. Recall

The ratio of number of instances of rightly predicted attack with all the instances of attack is called recall or detection rate [78]. It be calculated with the below equation.

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

It is crucial in high-stakes environments like Industrial Control Systems (ICS) or smart grids, where missing an attack can result in catastrophic consequences [25,48,78]. High recall is desirable in these contexts, but it must be balanced against precision to avoid

Table 10
Confusion matrix.

Actual class	Anticipated class Breach Regular	
Breach	True Negative (TN)	False Positive (FP)
Regular	False Negative (FN)	True Positive (TP)

overwhelming operators with false alarms [69,76,84].

6.4. False alarm rate

The ratio of incorrectly anticipated attack instances with all the instances that are normal is called False Alarm Rate [83]. It is calculated with the below equation.

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

It is vital in scenarios with high traffic volumes, such as smart cities or large enterprise networks, where excessive alerts can overwhelm security teams [23,32]. Minimizing FAR is essential for maintaining operational efficiency, and methods like feature selection and data preprocessing can contribute to achieving this [78,83].

6.5. F-Measure

It is a statistical technique of computing accuracy by considering recall and precision [34].

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

It is ideal for imbalanced datasets where both metrics are critical [39,50]. In IoT environments, where rare attack types like botnets need to be detected, the F1-Score balances the trade-offs between detecting actual attacks and minimizing false positives [25,48].

6.6. Detection rate

Detection rate essentially reflects the proportion of correct positive class predictions relative to the total number of predictions made [26].

$$\text{Detection Rate} = \frac{TP}{(TP + FP + TN + FN)}$$

It is particularly relevant in military or government networks, where undetected intrusions could have severe consequences [26,75]. High detection rates are often prioritized in these systems, but achieving them requires robust feature selection and model tuning [32,83].

Performance metrics are measured by applying the suggested methodologies on the selected benchmark datasets.

6.7. Understanding metric implications

Attributes Influencing Higher Metrics: High metric values depend on factors such as robust feature selection [97–116], balanced datasets [89–96], advanced preprocessing [22–109], and optimized model architecture [22–109]. For instance, methods like SMOTE and class-specific loss functions (e.g., Focal Loss) can improve recall and F1-Score in imbalanced datasets [92].

When Higher Metrics May Not Indicate Better Performance: In real-world scenarios, higher precision may lead to missed attacks (low recall), while high recall might overwhelm operators with false positives [36,69,71]. Metrics must be evaluated in the context of the IDS application to ensure a balance between detection capabilities and operational practicality [63,76,78].

Preferred Metrics for Specific Scenarios: Table 11 has shown each metric to specific use cases, illustrating when and why a particular

Table 11
Articles with their applied performance metrics and datasets.

Article	ACC	PRE	REC	F1	FAR	DER	Dataset
[22]	97.85%	99.99%	97.85%	98.15%	2.15%		KDD Cup99
							NSL-KDD
[23]	97.83%				2.35%		Real connected
							Healthcare Systems (CHS)
[24]	99.90%						KDDCup99, NSL-KDD
[26]	77.99%					73.37%	KDDCup99, NSL-KDD
[25]	99.90%	99.70%	98.90%	99.10%			KDDCup99,NSL-KDD and Real Network Traffic
[44]	98.40%	97.40%	97.30%	97.50%			Real World Dataset
[45]	99%						KDD CUP99, NSL-KDD,CICIDS2017, UNSW-NB15, Kyoto, WSN-DS
[32]	99%				1%	99%	NSL-KDD
							UNSW-NB15
[48]	99%	95.80%	98.90%				Malimg
[49]	97%						Real-world dataset
[34]	95%			98%			KDD CUP99, CICIDS2017, UNSW-NB15
[50]	98.07%	97.06%	99.22%	98.13%			CSIC-2010,CIC-IDS2017
[59]	100%						KDD CUP99,NSL-KDD and UNSW-NB15
[39]				90.16%			NSL-KDD and UNSW-NB15
[60]	92%						NSL-KDD and ISCX
[61]	97.50%						Real-world dataset with ECN testbed
[62]	99.81%						NSL-KDD, IoTDevNet, DS2OS, IoTID20, IoT Botnet
[63]	98.21%	96.42%	97.68%	97.05%			NSL-KDD and CICIDS2017
[64]	96.09%	97.01%	96.48%	96.74%			ISCXIDS2012, CIC-IDS2017,
[36]	98%	98.63%	98.62%	98.60%			CSE-CIC-IDS2018.
[71]	99.98%	100%	100%	100%			CSE-CIC-IDS2018
[69]	82.59%	84.59%	84.36%	82.58%			NSL-KDD
[70]	80%						ISCX
[38]				91.54%			Smart Water Distribution Plant Dataset
[75]	98.45%	96.12%	98.02%			98.95%	NSL-KDD 2015, CIDDS-001, and ISIC
[76]	98.99%	96.97%	96.97%	96.97%			NSL-KDD
[78]	82.08%	70.51%	97.27%	81.75%	2.62%		NSL-KDD
							UNSW-NB15
[41]	99.99%						KDD-CUP 99, NSL-KDD, and STIN
[83]	97.08%				0.07%	99.78%	KDD Cup 99, NSL-KDD, ISCX2012, and UNSW-NB15
[84]	93.17%		95.25%				KDD-CUP99 and UNSW-NB15
[85]	84.24%						UNSW-NB15

ACC- Accuracy REC-Recall FAR - False Alarm Rate
PRE-Precision DER- Detection Rate F1 -F1-Score

metric is critical.

Fig. 18 presents a bar chart representation of the various performance measures and their frequency of usage in the proposed algorithms that are applied in this study. Clearly, the chart showed that accuracy is the main performance indicator that is used by more than 90% of the researchers. The least used indicator is the False Alarm rate. Table 11 elucidates various performance measures along with the dataset used by the researchers. Table 12 illustrates various metrics along with their use cases and application scenarios.

7. Datasets

Several benchmark datasets used for training and evaluating DL models in order to calculate performance metrics are explained in this section. Fig. 19 outlines various datasets used with their frequency of usage. The datasets are,

7.1. NSL-KDD

From the earlier dataset KDD-CUP99, NSL-KDD was created. This is an optimal dataset used by many researchers in the field of intrusion detection and prevention system in the last decade. This dataset has four attack types and 41 versatile features [117]. From Fig. 19, it is particularly striking that the most widely used dataset in this study is NSL-KDD. Fig. 20 describes the attack distribution of train and test data of NSL-KDD.

7.2. KDD-CUP99

KDD-CUP99 with its two million testing and five million training cases, has been the most widely used IDS dataset for the past 20 years. It is a superset of NSL-KDD with 41 features and four breach types [118]. The attack types are User to Root(U2R), Denial of Service (DoS), Remote to Local(R2L) and Probe.Fig. 21 outlines the attack distribution of train and test data of KDD cup dataset.

7.3. UNSW-NB15

The UNSW-NB15 data set was created in 2015 by the Australian Center for Cyber Security [119]. It has around two million instances with 49 features. It has around two million instances with 49 features. The features can be extracted, and models can be implemented and tested using the latest implementation tools such as Pytorch 1.0, Python using Tensorflow framework, Keras 2.0, MATLAB and Talos library. For instance, this dataset has nine attack types namely Backdoors, Exploits, DoS, Generic, Fuzzers, Reconnaissance Port scans, worms, Shellcode, worms. Fig. 22 describes the attack distribution of UNSW-NB15 dataset.

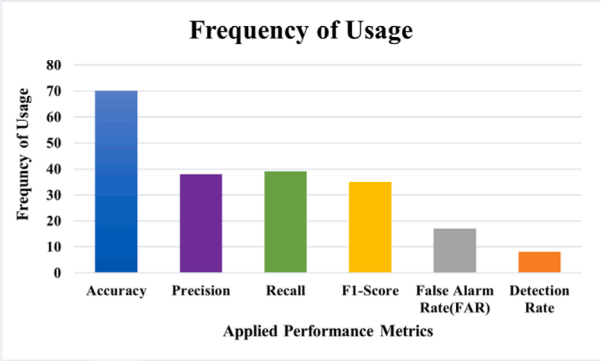

Fig. 18. Frequency of occurrence of performance matrix in the reviewed articles.

Table 12
Various metrics with their application scenarios and use cases.

Reference	Metric	Application Scenario	Use Case
[23,33, 34,61]	Accuracy	When the dataset is balanced, with equal proportions of attacks and benign traffic.	Ideal for environments like public cloud services or corporate networks where attacks are frequent and varied.
[22,48, 50,69]	Precision	When false positives (benign traffic misclassified as attacks) need to be minimized.	Important in financial institutions or healthcare systems, where false alarms can disrupt operations or incur high costs.
[25,36, 44,71]	Recall	When detecting all attacks is crucial, even at the cost of raising more false alarms.	Critical in industrial control systems (ICS) or smart grids, where missing attacks can lead to severe consequences.
[22,23, 32,78]	False Alarm Rate (FAR)	When minimizing false positives is paramount to avoid overwhelming security personnel.	Useful in large enterprise networks or smart cities, where a high volume of traffic can lead to excessive alerts.
[34,38, 39,44]	F1-Score	When both precision and recall are important, especially in imbalanced datasets.	Effective in IoT environments with rare but critical attack types like IoT botnets. Balances detecting actual attacks with minimizing false positives.
[26,32, 75,83]	Detection Rate (DER)	When the focus is on identifying a high proportion of intrusions, regardless of false positives.	Relevant in military or government networks, where missing an attack could have severe or catastrophic outcomes.

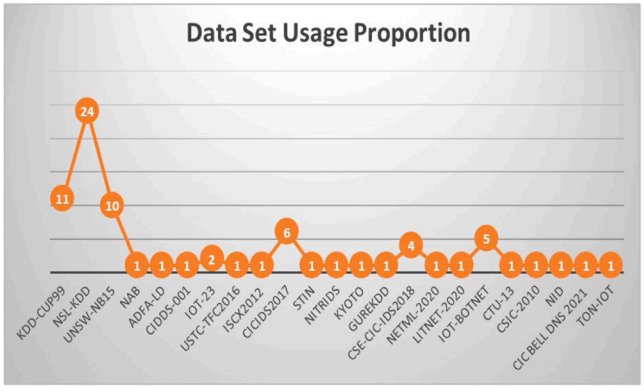

Fig. 19. Proportion of datasets used.

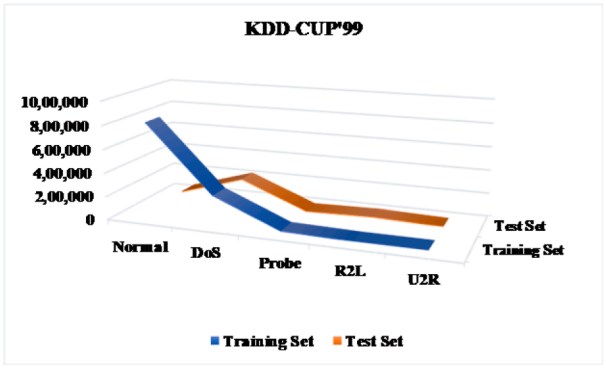

Fig. 20. KDD-CUP'99 attack distribution.

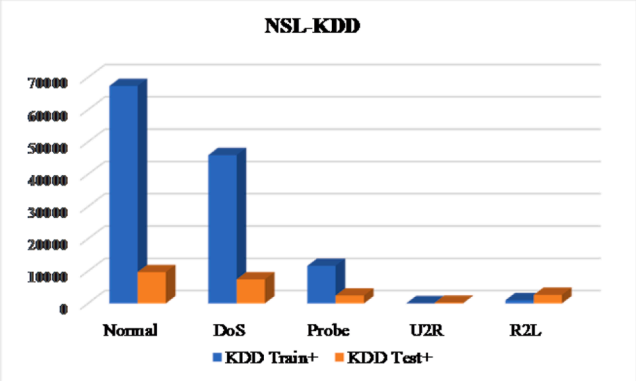

Fig. 21. NSL-KDD attack distribution.

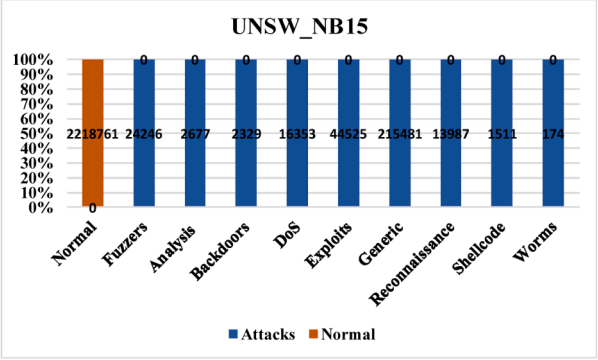

Fig. 22. UNSW-NB15 attack distribution.

7.4. IoT-23

The most recent dataset, IoT-23, was made available by the Czech Republic's Stratosphere Laboratory in January 2020. It consists of twenty captures from infected IoT devices and three from benign IoT devices utilizing a variety of IoT network flow [120] that can be applied on any IDS system for IoT.It has 15 attack types as shown in Fig. 23.

7.5. CIC-IDS2017

Canadian Institute for Cyber Security has created the dataset CICIDS2017. It has higher than 80 network traffic features, and have eight attack types such as 'Brute Force SSH, DDoS, Infiltration, Brute Force FTP, DoS, Heartbleed, Web Attack and Botnet ' [121]. It is the only dataset that matches all the 11 indicators namely 'Complete Network configuration, Complete Traffic, Labelled Dataset, Complete Interaction, Complete Capture, Available Protocols, Attack Diversity, Heterogeneity, Feature Set and Metadata for a robust benchmark dataset' [122]. Fig. 24 describes the attack distribution of CIC-IDS2017 dataset.

7.6. CSE-CIC-IDS2018

Canadian Institute for Cyber Security created the dataset CSE-CIC-IDS2018 [123]. It has seven distinct breach types namely Brute-force, infiltration of the network from inside, Botnet, Heartbleed, DoS, DDoS, and Web attacks. It utilized CICFlowMeter-V3 for measuring the network flow and has 80 features. Fig. 25 describes the attack distribution of CIC-IDS2018.

7.7. Bot-IoT

Cyber Range Lab of UNSW Canberra has measured real-time network domain and created IoT-Botnet [124]. It has around 72 million data instances with five attacks namely 'DDoS, Data exfiltration attacks, DoS, OS and Service Scan and Keylogging' [125]. Additionally, Dos and DDoS are also arranged depending on the protocol used. Fig. 26 describes the attack distribution of Bot_IoT.

7.8. AWID dataset

A well-known dataset used for studies in network security and intrusion detection is the Aegean Wi-Fi Intrusion Dataset(AWID) dataset [126]. It contains a diverse collection of network traffic data captured from various wireless network environments. The dataset includes both normal and attack traffic, making it suitable for training and evaluating IDS. It has 253 features, 36,913,503 instances and 13 types of attack as shown in Fig. 27. It is mainly used for detecting wireless attacks.

7.9. Other datasets

Many datasets such as Numenta anomaly benchmark (NAB) [127], ADFA-LD [128], CIDDS-001 [129], USTC-TFC2016 [130], ISCX2012 [131], STIN [132], NITRIDS [133], Kyoto [134], GureKDD [135], NetML-2020 [136], LITNET-2020 [137], CTU-13 [138], CSIC-2010 [139], CIC Bell DNS 2021 [140] and ToN-IoT [141] has been used by researchers in different domain to create an effective IDS.Table 13 gives the details of various important datasets with the total number of features, number of instances and different attacks.

Table 14 presents several datasets commonly used in NIDS, each with unique strengths, use cases, and limitations. NSL-KDD is an improved version of KDD Cup 99, useful for initial benchmarking but lacks representation of modern threats like IoT-based attacks. KDD Cup 99, while historically significant, is outdated and no longer reflective of current

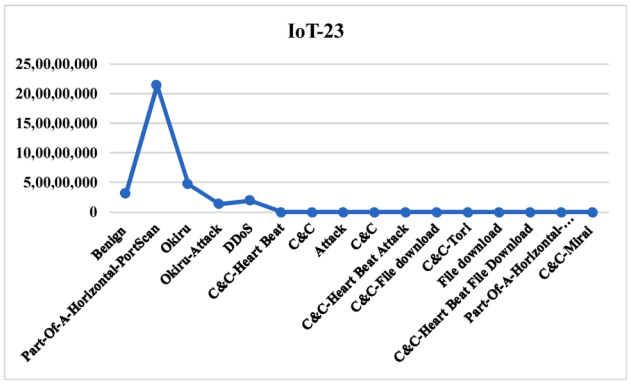

Fig. 23. IoT-23 attack distribution.

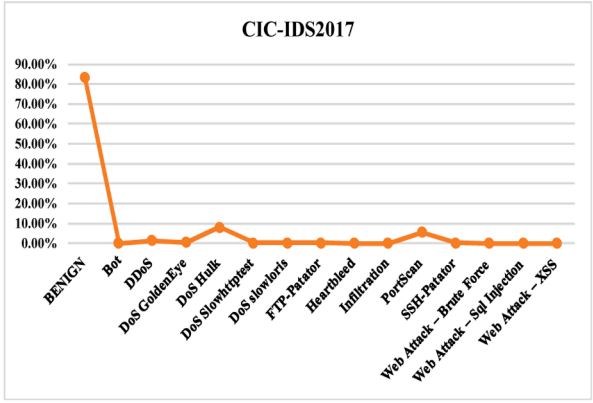

Fig. 24. CIC-IDS2017 attack distribution.

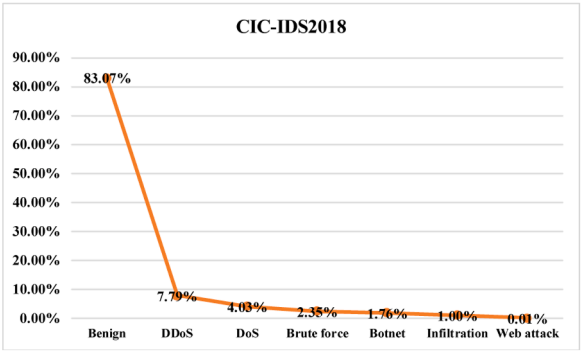

Fig. 25. CIC-IDS2018 attack distribution.

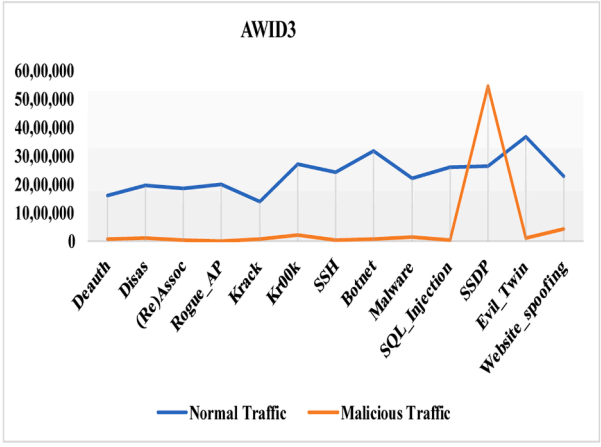

Fig. 26. Bot_IoT attack distribution.

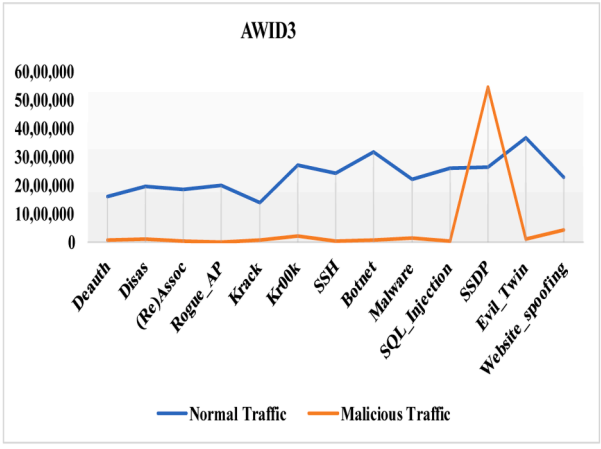

Fig. 27. AWID attack distribution.

cyberattack strategies. UNSW-NB15 offers a modern set of diverse attack types, making it suitable for critical infrastructures but contains synthetic data, limiting its real-world applicability. IoT-23 focuses on IoT traffic and botnet detection, ideal for smart cities and industrial IoT, though not applicable to traditional enterprise networks. CIC-IDS 2017 and CIC-IDS 2018 are well-suited for enterprise and cloud environments, capturing a wide range of modern attacks, but both datasets are synthetic, limiting their real-time adaptability. Bot-IoT is designed to capture IoT traffic, especially botnet attacks, and is effective in IoT ecosystems like healthcare or manufacturing, but it is IoT-specific and may not generalize well to other attack types. AWID focuses on wireless traffic and is ideal for detecting Wi-Fi network attacks, such as impersonation and injection, but it is limited to wireless environments and does not cover a broader spectrum of cyber threats.

Table 13
Description of various datasets.

Dataset	Total number of Features	Total number of Instances	Total number of Attacks
AWID	254	36,913,503	13
CIC-IDS2017	79	2,830,743	14
CIC-IDS2018	80	16,000,000	7
KDD-CUP 99	41	7,000,000	4
NSL-KDD	41	148,517	4
UNSW-NB15	49	2,540,044	9
IoT-23	21	325,308,048	21
Bot-IoT	45	72,000,000	10

8. Implementation tools for dl algorithms

Researchers have utilized various implementation to tools to train and test the deep learning models. Table 15 delineates different implementation tools used by researchers along with the description, appropriate scenarios, use cases and advantages. The major development environments are,

8.1. Python programming language

Python instructions are succinct and legible. The humongous libraries and framework, flexibility and platform independence characteristics make python a perfect tool for are,

(i) NumPy that performs complex mathematical task for machine and deep learning. [142]
(ii) Keras provides an environment that is simpler to run innovative fresh ideas [143].
(iii) TensorFlow delivers vast set of ready to use deep learning models [144] .
(iv) Pandas is built with NumPy as the base and yields powerful functions for deep learning [145].
(v) Matplotlib is cross-platform, classical charting library used for data visualization [146].
(vi) PyTorch is a self- differentiation library utilized to develop neural network [147].
(vii) Scikit-learn, predictive analysis tool with large utilities convenient for deep learning [148].
(viii) SciPy has various modules for linear algebra and optimization and is suitable for deep learning [149].
(ix) Theano is a library with module for quick numeric computation [150].

8.2. MATLAB

MATLAB is a quantitative analytic programming environment for data analysis, generate algorithms and develop models. with its huge tool-box, it is much faster than python for engineering applications [151]. MATLAB delivers superior algorithms and visualization tools to build and analyze AI models.

8.3. R programming language

R is an opensource software for statistical analysis and computing. It works on multiple platforms. It has many tools for data acquisition and cleaning [152].

Table 14
Datasets Used in NIDS: Strengths, Use Cases, and Limitations.

Dataset	Strengths	Use Case	Limitations
NSL-KDD	Improved version of KDD cup 99, reduces redundancy and duplicate records.	Suitable for initial benchmarking of NIDS algorithms. Educational tool for learning intrusion detection.	Outdated; lacks representation of modern attacks (e.g., APTs, IoT-based attacks). Not reflective of modern traffic.
KDD Cup 99	Historically significant; Broad range of attack and normal traffic for basic intrusion detection.	Ideal for teaching, early-stage NIDS testing, and baseline comparisons.	Extremely outdated; does not reflect modern attack strategies (e.g., ransomware, phishing, IoT-based threats).
UNSW-NB15	Modern dataset with diverse attack types (e.g., backdoors, exploits, shellcode).	Useful in high-security environments like critical infrastructure or ICS. Suitable for testing robust NIDS.	Synthetic data; limited representation of modern, dynamic attacks like multi-stage or obfuscated malware.
IoT-23	Focused on IoT traffic; Addresses vulnerabilities and botnet attacks in IoT networks.	Perfect for smart cities, industrial IoT, and smart homes. Helps detect IoT-specific threats like botnets.	Limited to IoT networks; not applicable to traditional NIDS in enterprise or cloud environments.
CIC-IDS 2017		Best for enterprise networks and cloud-based systems. Suitable for both traditional and deep learning models.	Synthetic dataset; lacks variability and complexity of live network traffic. May not fully replicate real-time conditions.
CIC-IDS 2018	Captures a broad spectrum of up-to-date attack types (DDoS, brute force, botnet). Provides realistic, labelled traffic.	Suitable for modern enterprise environments or cloud-based services. Useful for evaluating NIDS across various attack vectors.	Complex and large; requires significant computational resources. Still a synthetic dataset, limiting real-world adaptability.
	Captures IoT traffic with a focus on botnet attacks.	Relevant for IoT ecosystems (e.g., healthcare, manufacturing, automotive). Effective for botnet detection.	IoT-specific dataset; less applicable to enterprise networks. May not generalize to other attack patterns.
	Focused on wireless traffic, capturing Wi-Fi network attacks like impersonation and injection.	Best suited for NIDS targeting wireless networks in public spaces or corporate Wi-Fi networks. Helps address Wi-Fi-specific vulnerabilities.	Limited to Wi-Fi-based traffic; does not cover a wide range of modern cyberattacks found in wired or cloud environments.

To summarize, the choice of implementation tools plays a critical role in the development and deployment of deep learning models for NIDS [27,94]. Python, with its extensive libraries such as TensorFlow, Keras, and PyTorch, is ideal for rapid prototyping and building scalable models [31,57,87,89],while MATLAB excels in statistical analysis and visualization for engineering-focused applications [24,67,76,83,102]. R is a preferred choice for exploratory data analysis and visualization, especially in academic research contexts [32]. Tools like Scikit-learn and NumPy simplify data preprocessing, making them invaluable for integrating traditional machine learning techniques with deep learning models [27,31,71,85]. Each tool has its unique strengths, and selecting the appropriate one depends on the specific requirements of the NIDS, such as scalability, computational complexity, or the need for real-time deployment. Together, these tools empower researchers to tackle diverse challenges in network intrusion detection effectively, enabling better adaptability and performance.

9. Applications of NIDS

This section gives the overview of various applications of NIDS. Deep learning-based Network Intrusion Detection Systems (NIDS) have been applied across a wide array of domains, reflecting the versatility and robustness of these techniques in various environments. Table 16 provides the details of application environment in which the NIDS is applied. The most common application environment of NIDS are Internet of Things (IoT), Industrial Internet of Things (IIoT), Internet of Medical Things (IMoT), Industrial Control System (ICS), Cloud security and In-vehicle security.

9.1. Healthcare

The growing interconnectivity of medical devices, often termed as the Internet of Medical Things (IoMT), poses significant security risks. Intrusion detection systems are critical in protecting patient data and ensuring the safe operation of healthcare systems [48]. The proposed NIDS [23], utilizing a stacked autoencoder (SAE), is designed to enhance the security of Connected Healthcare Systems (CHSs) by effectively detecting and mitigating both internal and external threats, such as data tampering, eavesdropping, and replay attacks. Nguyen et al. [75] employed advanced techniques such as MSC-based encryption and ResNet 101-based classification, and ensures both high detection accuracy and secure handling of sensitive medical data. The proposed multimodal NIDS [98] is particularly effective in securing healthcare environments, offering a detection rate of 99.59% and demonstrating the benefits of combining multimodal data fusion with advanced machine learning techniques. The proposed Blockchain-Assisted IoT Healthcare System (BHS-ALOHDL) [104] is effective in enhancing the security and reliability of IoT healthcare systems, demonstrating superior performance on benchmark datasets through optimized hyperparameter tuning using the Flower Pollination Algorithm.

9.2. IoT (Internet of things)

The most common application area of NIDS is IoT. Given the vast number of connected devices and the potential vulnerabilities, intrusion detection systems must be highly adaptive and capable of learning from a wide variety of attack patterns. [98,100,106], and [108]. The proposed IDS framework rigorously evaluates multiple deep learning classifiers across diverse datasets, including IoT-specific environments, to address the limitations of single-dataset models and ensure robust, unbiased intrusion detection [59]. By utilizing adversarial training techniques like FGSM, the study demonstrates significant enhancements in the resilience of GRU and LSTM models against adversarial examples, making them more effective for securing IoT networks [36]. The study demonstrates that the enhanced DBN-based IDS outperforms traditional detection algorithms, offering a more robust defense against unauthorized access and malicious activities in IoT networks [77]. Ullah et al. [87] introduced a deep convolutional neural network (DCNN)-based intrusion detection system (IDS) specifically designed for Internet of Things (IoT) networks, aiming to enhance performance while reducing computational demands. The proposed system introduced a wireless Intrusion Detection System (IDS) proved adaptable for both wired and wireless networks [96].The NIDS proposed by various researchers [97,99,105,108] proved to be successful in IoT environment.

9.3. IMoT (Internet of medical things)

IMoT is a specialized field within IoT that focuses on medical devices. Priya et al. [43] highlighted the use of a hybrid PCA-GWO based DNN Classifier Model for effectively managing and classifying cyberattacks

Table 15
Comparison of Implementation Tools for Deep Learning in NIDS Applications.

References	Tools	Description	Appropriate Scenarios	Application Areas	Key Advantages
[27,30,45,54–57,63,71,84,85,87,89,92,94]	Python	A versatile programming language with extensive libraries and frameworks for DL model development.	(i) For general-purpose DL model development. (ii)When flexibility and ease of use are priorities.	(i)Building custom NIDS. (ii) Integrating DL models into real-world systems. (iii)Experimenting with novel algorithms.	(i)Rich ecosystem (NumPy, TensorFlow, PyTorch, Keras). (ii)Platform-independent. (iii)Ideal for rapid prototyping.
[2,27,30,31,46,55,57,62,68,85,87,89,91,94,95]	TensorFlow	An open-source framework providing ready-to-use DL models and tools for scalable DL development.	(i)When working with large-scale or distributed models. (ii) For building and deploying production-grade systems.	(i)Training high-performance models. (ii) Developing cloud-based NIDS solutions. (iii)Edge device integration.	(i) Scalable. (ii)Provides pre-built DL models. (iii)Excellent for GPU/TPU acceleration.
[27,28,30,39,48,52,57,62,66,68,87,89,91,94,95]	Keras	A high-level API for building DL models, offering simplicity and rapid experimentation.	(i)When ease of experimentation is required. (ii)For beginners developing DL models.	(i) Prototyping NIDS models. (ii)Experimenting with new architectures.	(i)User-friendly. (ii)Supports multiple backends (TensorFlow, Theano). (iii) Encourages quick innovation.
[92]	PyTorch	A flexible framework for DL, focusing on dynamic computation graphs and ease of debugging.	(i) For research-focused projects. (ii) When iterative model development and debugging are critical.	(i) Developing advanced NIDS architectures (e.g., attention mechanisms). (ii)Academic research projects.	(i)Dynamic computation graphs. (ii)Strong community support. (iii) Ideal for experimentation
[27,31,39,45,71,85]	Scikit-learn	A machine learning library for predictive analytics, including tools for pre-processing and classification.	(i)For integrating traditional ML techniques with DL models. (ii)When lightweight tools are sufficient.	(i)Data pre-processing for NIDS. (ii)Feature engineering and selection.	(i)Easy integration with Python. (ii)Lightweight and efficient. (iii)Broad ML utilities.
[24,67,76,83,102]	MATLAB	A quantitative analysis environment with extensive visualization and data analysis capabilities.	(i)When complex statistical analysis is required. (ii) For engineering applications with specialized toolboxes.	(i) Statistical analysis for NIDS datasets. (ii)Academic environments requiring robust tools.	(i)High precision. (ii)Superior visualization. (iii)Optimized for engineering problems.
[32]	R	An open-source software for statistical analysis and visualization.	(i)For statistical analysis-heavy projects. (ii)When working with smaller datasets.	(i)Exploratory data analysis for NIDS. (ii)Visualization of attack trends in datasets.	(i)Excellent for statistical modeling. (ii)Easy data visualization. (iii)Cross-platform support.
[52]	Theano	A Python library for efficient numerical computation in DL.	(i)When computational speed is critical. (ii)For GPU-accelerated training of smaller models.	(i)Developing lightweight NIDS. (ii) Integrating DL in resource-constrained environments.	(i)Optimized for GPU usage. (ii)Integrates well with Keras.

within the IoMT environment.The findings [107] can be applied to enhance the security of IoMT systems by implementing robust intrusion detection mechanisms that quickly identify and respond to cyber threats. The proposed DRNN and SML-based IDS can be integrated into healthcare networks to protect sensitive medical data from unauthorized access and potential cyberattacks [43].

9.4. Industrial control systems (ICS)

Network Intrusion Detection Systems (NIDS) in Industrial Control Systems (ICS) are essential for monitoring network traffic and identifying potential threats in real-time. They help protect critical infrastructure, such as power plants and manufacturing systems, by detecting and responding to cyberattacks, thereby preventing disruptions, operational failures, and financial losses. The proposed deep learning-based intrusion detection paradigm can be applied to enhance the security of IIoT systems by accurately identifying and mitigating network attacks in real-time [32]. The proposed hybrid SVMNN algorithm can be applied to enhance the security of power grids by accurately predicting and detecting cyber intrusions that could disrupt optimal power flow operations [42]. The proposed DA-SMOTE-ED method can be applied to improve anomaly detection in gas turbines by effectively balancing imbalanced datasets and reducing the impact of inter-class overlap, leading to more accurate detection of faults [92].

9.5. Smart cities and in-vehicle

NIDS can be deployed to secure the vast and interconnected networks of smart cities, protecting critical infrastructure such as traffic management systems, utilities, and public safety networks from cyber threats. In modern vehicles, NIDS can be integrated into in-vehicle networks to detect potential cyberattacks targeting critical systems like engine control, braking, and infotainment systems. The proposed intrusion detection [38] framework based on Restricted Boltzmann Machines (RBMs) enhances the security of smart city infrastructures by accurately detecting Distributed Denial of Service (DDoS) attacks. This methodology is particularly effective in analyzing data from smart city applications, such as water distribution systems, and can be extended to other critical smart city services to prevent and mitigate cyber threats. The proposed model enhances vehicle cybersecurity by accurately detecting and mitigating message flooding attacks on the Controller Area Network (CAN) bus [100]. It a powerful tool for securing in-vehicle communication systems, ensuring reliable and safe vehicle operation within smart city environments.

Based on the findings presented in this section, several methods within DL based NIDS show great potential.

(i) Ensemble learning techniques, which combine multiple neural network architectures, demonstrate a strong ability to enhance detection performance and generalization across diverse environments, making them particularly well-suited for dynamic settings like IoT and smart cities [36,38,77].

Table 16
Articles with Applications.

References	Application Area	When to Use	Where to Apply	Key Benefits
[23,48,75, 99,105]	Healthcare	(i)When dealing with sensitive patient data. (ii)To prevent unauthorized access and cyberattacks.	(i) Connected Healthcare Systems (CHS). (ii)IoMT environments like hospitals and telemedicine setups.	(i) Protects patient privacy. (ii)Ensures reliable operation of medical devices. (iii) High detection accuracy.
[36,59,77, 87,97, 98,100, 106, 109]	IoT (Internet of Things)	(i)When handling diverse datasets with potential vulnerabilities. (ii)To secure wireless networks.	(i) Smart homes. (ii)IoT-specific networks in enterprises or manufacturing units.	(i)Adapts to a wide variety of attack patterns. (ii)Enhances resilience against adversarial attacks.
[43,108]	IMoT (Internet of Medical Things)	(i)When focusing on medical device-specific security. (ii)To protect life-critical systems.	(i) IoMT-specific networks (e.g., infusion pumps, wearable devices).	(i) Offers specialized protection for medical IoT. (ii) Ensures safety and functionality of devices.
[32,42,92, 102]	Industrial Control Systems (ICS)	(i)When monitoring industrial networks for real-time anomaly detection. (ii)For critical systems.	(i)Power grids, gas turbines, manufacturing systems. (i)Industrial IoT (IIoT) environments.	(i) Prevents operational failures. (ii) Ensures continuity of critical services.
[38]	Smart Cities	(i)When managing large-scale interconnected infrastructures. (ii)To prevent disruptions in services.	(i) Traffic management systems, public safety networks, utility services.	(i) Protects critical infrastructure. (ii) Prevents cyberattacks in essential public services
[101]	In-vehicle Security	(i)When securing CAN buses or in-vehicle networks. (ii)To ensure vehicle safety and reliability.	(i)Autonomous vehicles. (ii)Smart transportation systems within smart cities.	(i) Enhances vehicle cybersecurity. (ii)Prevents message flooding and spoofing attacks.

(ii) The integration of advanced optimization algorithms into NIDS can significantly improve feature selection and model accuracy, particularly in resource-constrained environments like healthcare and industrial control systems [42,48,75,92].

(iii) The use of multimodal data fusion techniques in NIDS, which leverage various data sources for improved anomaly detection, will likely become increasingly relevant as networks grow in complexity and interconnectedness. These methods not only address current challenges in cyber threat detection but also provide a foundation for building more adaptive, efficient, and resilient security solutions in an ever-evolving landscape of cyber threats [99].

10. Future trends

This part discusses the possible future enhancements to the proposed frameworks that are drawn in this review. Table 17 consolidates the possible future trends. The notable future trends are

10.1. Real-Time detection and response

One of the most critical future trends is the development of real-time NIDS that can process and analyze network data with minimal latency [46] . Traditional NIDS often suffer from delays due to the need to accumulate and preprocess data flows. However, there is a growing interest in packet-level analysis, which can drastically reduce detection time. Future NIDS will be able to detect malicious activities as they occur, enabling quicker responses to cyber threats. The integration of these various deep learning models with advanced hardware, such as Field-Programmable Gate Arrays (FPGAs) and Graphics Processing Units (GPUs), will further enhance real-time capabilities [55].

Perspective: The future of real-time detection will involve fully autonomous NIDS that not only detect but also respond to threats instantly. This could involve collaboration with software-defined networking (SDN) frameworks, where automated threat mitigation actions can be triggered in real time.

10.2. Enhancing adaptability and robustness

One of the primary challenges in NIDS is the detection of zero-day attacks, which are previously unknown vulnerabilities exploited by attackers [23]. Future research will likely focus on enhancing the adaptability and robustness of detection models to respond to these threats effectively. This involves developing algorithms that can learn and adapt to new attack patterns in real-time, ensuring that NIDS remain resilient against novel and evolving threats [77].

Perspective:In the future, we foresee the development of self-healing networks that can not only detect zero-day attacks but also learn from them, enabling NIDS to evolve continuously and become more resilient over time. This approach will help organizations stay ahead of emerging threats in an increasingly interconnected world.

10.3. Optimizing detection efficiency

The efficiency of detection mechanisms without overwhelming system resources is a critical concern. Future work will focus on optimizing the depth of models such as Stacked Autoencoders (SAEs) to enhance detection accuracy while minimizing the system workload [22,29]. Additionally, research will explore the relationship between feature selection and model depth across different environments to optimize detection schemes further [74]. This optimization is crucial for maintaining high accuracy without compromising the speed and efficiency necessary for real-time detection [89].

Perspective: The future holds the promise of lightweight and energy-efficient NIDS, particularly for resource-constrained environments such as IoT and edge computing. These systems will leverage energy-efficient algorithms to ensure high detection performance without overwhelming the underlying infrastructure.

10.4. Real-Time deployment and performance enhancement

Deploying NIDS in real-time environments, particularly in cloud computing contexts, will be a significant focus [46]. Future work will involve evaluating the proposed IDS models in actual conditions to assess their detection efficiency and effectiveness [35]. Furthermore, exploring alternative metaheuristic optimization algorithms will be crucial in enhancing the performance and accuracy of these models. This approach aims to refine the models to operate effectively in dynamic, large-scale network environments [55].

Perspective: Future deployment scenarios will include edge-based NIDS, where security systems are distributed closer to data sources, such as IoT devices and smart sensors. This will enable faster detection

Table 17
Articles with their issues and challenges with future trends.

Article	DL Method	Issues/Challenges	Future trends
[22]	AE	(i) It suffers from declining detection accuracy, especially when confronted with sophisticated or novel attack patterns. (ii) The ability of these systems to adapt to evolving threats, such as zero-day attacks, remains a critical concern, limiting their effectiveness in real-world applications.	(i)Enhancing the adaptability and robustness of detection models, particularly in the context of zero-day attacks. (ii)Integrating real-world network traffic data into model evaluations will become increasingly important, ensuring that these systems are capable of sustaining high performance in diverse and complex network environments
[23]	AE	(i)The gap in detection capability between lightweight intrusion detection and deep learning methods remains a challenge. (ii)Overhead reduction is limited, and improper depth selection in the SAE can negatively impact detection performance.	(i)To improve detection efficiency without increasing the system workload and optimizing the depth of the stacked autoencoder (SAE) to enhance detection accuracy. (ii)Research will also explore the relationship between feature selection and SAE depth across different environments to further optimize the detection scheme.
[46]	DNN	(i)The models were tested on a specific dataset (CSE-CIC-IDS-2018) and not yet implemented in a real cloud environment, potentially limiting the practical applicability of the results. (ii)It focused on with specific optimization techniques (BP and PSO), which may not generalize to other approaches.	(i)Future work will involve deploying the proposed intrusion detection systems (IDSs) in a real-time cloud computing environment to evaluate their detection efficiency under actual conditions. (ii)Exploring alternative metaheuristic optimization algorithms will be pursued to further enhance the performance and accuracy of the models.
[47]	DNN	(i)The experiments primarily focus on basic hyperparameters and architectures, leaving out potential improvements from more advanced techniques such as batch normalization and dropout. (ii)The experiments are based on a single IoT dataset and feature-level attacks, which may not fully represent real-world network traffic or packet-level attacks, potentially limiting generalizability.	(i)Future work will explore the impact of advanced layers and techniques like batch normalization, dropout, and various NIDS architectures on decision boundary robustness and performance. (ii)This research will extend to diverse network datasets and packet-level adversarial attacks to enhance the practical applicability and generalizability of the proposed defenses in real-world NIDS environments.
[55]	CNN	(i)Implementing and training the system can be complex due to the combination of CRF, LCC, and CNN. (ii)The system may require significant computational resources, impacting deployment on resource-limited devices.	(i)To explore the integration of intelligent agents to improve decision-making processes and enhance data communication speed in the proposed intrusion detection system. (ii)Further research may focus on optimizing the model's efficiency in real-time applications while maintaining high detection accuracy.
[58]	CNN	(i)The reliance on specific datasets (NSL-KDD and UNSW-NB15) may limit generalization to other datasets or real-world scenarios. (ii)The complexity of the model may increase computational requirements, impacting deployment on resource-constrained systems.	(i)Refining the feature fusion process and feature augmentation to enhance the model's operational efficiency. (ii)Exploring data construction standards for various dimensions and optimizing feature selection techniques will be pursued to develop a more robust and effective NIDS model.
[60]	RNN	(i) Integrating the SFSDT feature selection with various RNN models increases the complexity of the system, requiring more sophisticated implementation and fine-tuning.	(i)To explore applying the SFSDT feature selection model and LSTM-based IDS framework to other types of network attacks and larger, more diverse datasets to enhance generalization and performance. ii)Further research will focus on optimizing the framework for real-time big data processing and investigating its applicability in different domains beyond network security.
[61]	RNN	(i) The ensemble approach with multiple base classifiers increases computational complexity and may require significant processing power. (ii)The study focuses on specific attack types, and additional research is needed to cover emerging threats and optimize detection speed.	(i)Expanding the proposed ensemble IDS method to detect emerging and more sophisticated network attacks on the train Ethernet Consist Network (ECN). (ii)Efforts will be made to optimize the computational speed of the method while maintaining high detection accuracy, ensuring it can effectively operate in real-time environments.
[36]	LSTM	(i)The system assumes packets are received in order, which may not always be the case in real networks. (ii)High space complexity for sorting if many packets arrive out of order, which could impact performance.	(i)To address the challenges of out-of-order packet reception and packet loss to enhance the robustness of the proposed NIDS in real-world network environments. (ii)Efforts will be made to optimize the system further for large-scale deployments, ensuring high detection accuracy and minimal memory usage even under increasing network traffic.
[37]	LSTM	(i)The model may struggle with packets lacking sufficient contextual information, potentially reducing classification accuracy and implementing and training LSTM models requires substantial computational resources and expertise.	(i) To optimize packet-based classification methods to enhance detection accuracy and further reduce processing time, making real-time intrusion detection more feasible.
[38]	RBM	The computational complexity of scaling RBMs to deeper architectures (e.g., Deep Restricted Boltzmann Machines) can be a barrier to real-time application, requiring significant computational resources.	(i)To explore the potential of Deep Restricted Boltzmann Machines (DRBMs) for improved accuracy and novel attack detection in A-NIDS, despite their computational complexity. (ii)Advancements in high-performance computing and Adiabatic Quantum Computers (AQC) may enable real-time

(continued on next page)

Table 17 (*continued*)

Article	DL Method	Issues/Challenges	Future trends
			applications of DRBMs, offering a promising direction for cybersecurity.
[75]	MLP	(i)The integration of multiple technologies (DBN, MSC, blockchain, ResNet) makes the system complex to implement and maintain. (ii)The use of multiple advanced techniques could introduce performance overhead and increased computational requirements.	(i)To incorporate advanced hyperparameter tuning techniques and dynamic learning rate schedulers to further enhance the performance and efficiency of the proposed CPS model in healthcare. (ii)To expand the model to support real-time applications and diverse healthcare datasets could improve its adaptability and robustness in varying clinical environments.

and response times, improving overall system resilience and minimizing the risks posed by delays in centralized processing.

10.5. Advanced techniques and architectures

Exploring the impact of advanced deep learning techniques, such as batch normalization, dropout, and diverse NIDS architectures, on decision boundary robustness and overall performance is another critical area of research [60]. Future studies will extend these investigations to diverse network datasets and packet-level adversarial attacks to improve the practical applicability and generalizability of the proposed defenses in real-world NIDS environments [76]. These advancements aim to create more resilient and adaptable systems capable of defending against sophisticated attacks [77].

Perspective: The future will see the adoption of deep learning ensembles and advanced hybrid models that combine multiple neural networks to improve robustness and generalization. These architectures will form the backbone of next-generation NIDS, capable of defending against increasingly sophisticated cyberattacks in real-time.

10.6. Intelligent agents and feature optimization

The integration of intelligent agents into NIDS is a promising direction for improving decision-making processes and enhancing data communication speeds [55]. Future research may focus on optimizing these models for real-time applications while maintaining high detection accuracy. Additionally, refining the feature fusion process and exploring data construction standards will be crucial in developing more robust and effective NIDS models [58]. Optimizing feature selection techniques will also play a vital role in enhancing the operational efficiency and accuracy of these systems [69,80].

Perspective: The future of NIDS could involve autonomous security agents powered by AI that continuously monitor and optimize network performance. These agents will work in tandem with human operators, learning from real-world data and adjusting models in real time to ensure optimal performance.

10.7. Expanding applications and generalization

Future work will explore applying advanced models, such as Deep Restricted Boltzmann Machines (DRBMs), to improve accuracy and detect novel attacks in Anomaly-based NIDS (A-NIDS) [38]. Despite their computational complexity, advancements in high-performance computing and Adiabatic Quantum Computers (AQC) may enable real-time applications of these models [38,81]. This research will also extend to various domains beyond network security, such as healthcare and industrial control systems, to improve the generalization and applicability of NIDS across different sectors.

Perspective: NIDS applications will continue to expand beyond traditional networks into specialized sectors such as smart cities, autonomous vehicles, and critical infrastructure. As computational power increases, these systems will become more sophisticated, detecting and mitigating threats in real time across multiple environments.

Hyper parameter tuning methods with learning rate schedulers can be employed to improve the performance [75]. The performance of the suggested framework will be analyzed for multiclass problems [43,99,153]. The proposed system will be analyzed with real-time data and power-time optimization [62]. The swarm-NN strategy is improved for categorization of attack signature in IoMT environment [62,154] .The proposed framework will be applied on various real-world datasets to measure the utility and scalability of the model [35]. The incremental models will be retrained with fresh attacks to make the system suitable for multiclass scenario [25]. The model can be extended to detect any malicious intrusions on real-time steaming data [155]. The SPEED model will be extended to edge nodes and mobile devices [49].The framework will be elongated to accustom various protocol [32]. Efficient methods are investigated for the performance improvement in IDS. Also, AE with various latent layers are investigated for the effective feature extraction [28]. The robustness of the framework will be verified with complex attacks and using adversarial examples [36]. The complexity of the framework will be reduced by applying less complex models such as GRU deep learning algorithms [65]. The proposed model will be tested with latest dataset.in addition, improvement of detection rate in reduced dataset classes [26]. Various deep learning models will be combined to form hybrid models to improve the accuracy and detection rate [30].

To recapitulate, the major noted future trends are(i) Application of the framework to real-time data (ii) Construct hybrid models (iii) Enhancement of models to multiclass problems and(iv) Improve the models to increase the detection rate, accuracy and to reduce complexity.

11. Findings and discussion

This section provides detailed discussion of the findings acquired from this survey. From this survey, many inferences have been made.

Research Question 1(RQ1) This review outlines the use of DL in NIDS. It is particularly noted that DL models can learn complex features automatically, making them suitable for identifying intricate attack patterns that traditional methods might overlook [22]. DL models' adaptability allows them to detect new and evolving threats without extensive retraining, which is crucial given the dynamic nature of cyber threats [32]. The paper highlighted various DL architectures such as Auto Encoders (AE), Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Restricted Boltzmann Machine (RBM), Deep Belief Network (DBN), and Multilayer Perceptron (MLP) [22–110]. Fig. 28 shows various DL methodologies and their proportion of usage in this study. Clearly, it is inferred from Fig. 28 that CNN is the predominantly used methods with around 21% of articles uses CNN followed by LSTM with around 19%. AE also shows significance with about 15%. DNN shows moderate popularity with about 10% of selection There is a balanced used of RBM and DBN. Recurrent Neural Networks (RNN) and Multilayer Perceptron (MLP) have lower percentages of selection compared to other techniques.

For instance, CNNs are particularly effective in feature extraction from raw network traffic data, enabling the identification of intricate attack patterns that may not be immediately apparent through traditional methods [34,35,48–58]. LSTM networks, on the other hand, excel in handling sequential data, making them ideal for detecting temporal patterns and anomalies in network traffic that unfold over time [36,37,

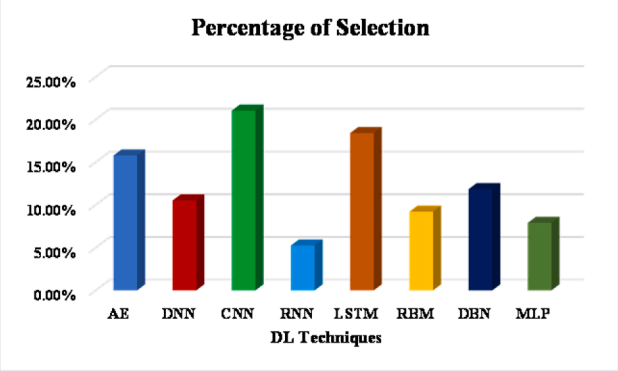

Fig. 28. Proportion of usage of various deep learning algorithms used in this study.

62–68]. Auto Encoders are highlighted for their ability to perform dimensionality reduction and anomaly detection, which are critical in identifying deviations from normal network behavior that may indicate an intrusion [22–31,43]. The review also noted that while DNNs are versatile and widely applicable, they may not be as specialized as CNNs or LSTMs in certain NIDS applications [32,33,43–47]. Nevertheless, DNNs' deep hierarchical structure allows for the modeling of complex relationships within the data, making them valuable in scenarios where a more generalized approach is needed [33]. It is observed that, there is always a scope to the combine DL models and make hybrid models or to construct variation of any DL techniques to improve performance of the proposed system [89].

(RQ2) The use of appropriate dataset decides the effectiveness of the proposed system. The DL methods learn fruitful features and patterns from large raw datasets. The zero-day attacks can be identified by training the NIDS frequently with the fresh data extracted by inspecting the network flow [22]. The detection accuracy depends on how frequently the model is trained. It is observed that most of the models have been tested with age-old datasets like KDD-CUP99 and its successor NSL-KDD [24–26,28,29,30]. Also, it is inferred that some models produce outstanding results with the older datasets and comparatively low results for newer dataset [96]. Furthermore, the class imbalance problem due to few breach samples, can affect the detection rate directly and more attention must be given in this area. From Fig. 19, it is inferred that NSL-KDD is the most commonly used data set in this study followed by KDD-CUP99 and UNSW_NB15 due to it's long existence and general applicability.

(RQ3) In addition to discussing the DL techniques, this paper places significant emphasis on the importance of data preparation strategies in the development of effective NIDS. Data preparation is a crucial step in the machine learning pipeline, as the quality and relevance of the data directly impact the performance of the DL models [22–116]. The paper outlines various techniques for data preparation, including data cleaning, data augmentation, and feature selection. Data cleaning involves removing noise and irrelevant information from the dataset, ensuring that the models are trained on high-quality data [22–116]. Data augmentation is used to artificially increase the size of the training dataset by creating modified versions of the existing data, which helps in improving the model's generalization capabilities [89–96]. Feature selection is another critical aspect of data preparation, as it involves identifying the most relevant features that contribute to the detection of intrusions, thereby reducing the dimensionality of the dataset and improving model efficiency [97–116]. It is inferred that the data augmentation technique can be applied only to minority classes also. Over the last decade, several optimization algorithms have been utilized for feature selection in NIDS.

(RQ4) The review also explores various performance evaluation metrics used to assess DL-based NIDS, including accuracy, precision, recall, F1-score, detection rate, and false alarm rate. These metrics provide a holistic understanding of the models' effectiveness, guiding researchers in optimizing their systems. From Fig. 29, it is noted that, the detection accuracy is the predominantly used metrics to measure the performance of the proposed system followed by precision [22–116]. So, the accuracy and precision must be calculated as a basic measure of performance of the system.

(RQ5) It is noticed that various implementation tools and libraries are used to construct and test any system [35]. This review discusses the tools and platforms commonly used for implementing DL-based NIDS. These include machine learning libraries like TensorFlow and PyTorch, data management tools, and various platforms that facilitate model development, training, and testing [56,57,63,71]. Further, it is deduced that, the computational resources for building, training, and testing the model is high. The deeper the methodology, more complex it could be and the computational resource requirements are higher [45]. Thanks to the advent of GPU, the higher computational requirements can be met. Since, the DL strategies are not matured, there is always a room for improvement in most of the models.

Python with its massive libraries lead the race of implementation platform and tools for DL techniques. Moreover, MATLAB can be utilized for scientific applications.

(RQ6) The review addresses the application of DL-based NIDS across various domains, including the Internet of Things (IoT), Industrial Internet of Things (IIoT), Internet of Medical Things (IMoT), cloud computing, fog computing, in-vehicular networks, and Industrial Control Systems (ICS) [23,59,43,32,38,101]. Each domain presents unique challenges, and the paper illustrates how DL techniques are adapted to meet these challenges, thereby demonstrating the versatility and impact of NIDS across different network environments.

11.1. Challenges/Issues

While deep learning methods offer significant advancements in detecting complex and evolving cyber threats, they also present challenges that must be carefully considered, especially in the context of critical applications like intrusion detection.

The inherent stochastic nature of deep learning models can lead to variability in performance, requiring extensive training and fine-tuning to achieve consistent results [22]. Additionally, deep learning models often operate as "black boxes," making it difficult to interpret their decision-making processes, which is crucial for ensuring trust and transparency in security-critical environments [46]. The computational complexity of deep learning models can also result in resource-intensive deployments, potentially limiting real-time applicability in high-throughput networks [55].

Moreover, deep learning models are susceptible to adversarial attacks, where small, intentional perturbations in the input data can lead to misclassification [47]. This vulnerability poses a significant risk in

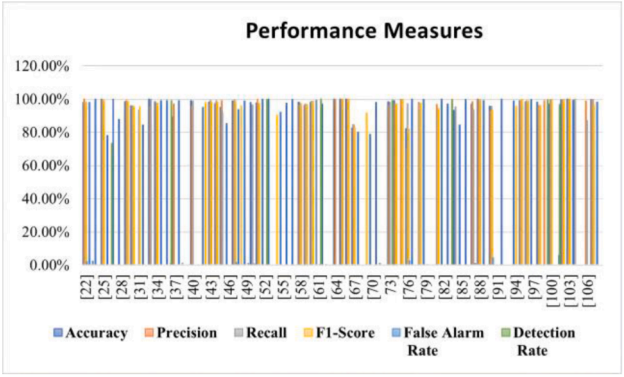

Fig. 29. Performance measures used by various articles.

intrusion detection, where false negatives can have severe consequences [65]. We have expanded the discussion in our paper to reflect these challenges and potential mitigation strategies in future trends, which are summarized in the Table 17.

To tackle the challenges presented by various DL methods in NIDS, it is essential to recognize the fundamental characteristics of each method and how these contribute to particular difficulties.

(i) For AE, the primary challenge is their limited performance when dealing with complex and subtle attack patterns, such as Advanced Persistent Threats (APTs). This arises because autoencoders focus on compressing input data and reconstructing it, making it difficult to capture sophisticated attack behaviors that closely mimic normal traffic. The size of the latent space, or bottleneck layer, is crucial—too small and important information is lost, too large and the compression is ineffective [22–31].
(ii) DNNs, while powerful, face challenges related to overfitting and computational complexity. DNNs contain many layers and parameters, which can result in overfitting, especially when dealing with imbalanced datasets—a common scenario in NIDS where attacks are rarer than benign traffic. Additionally, DNNs require significant computational resources for training, especially as the number of hidden layers and neurons increases, making them less efficient for real-time detection [32,33,43–47].
(iii) CNNs, though highly effective for spatial data like images, struggle when applied to network traffic data, which is temporal in nature. CNNs are not well-suited for capturing sequential dependencies, which are essential in detecting network intrusions. To overcome this, CNNs often require heavy preprocessing of data into a structured format, but this adds complexity and limits their ability to detect patterns that emerge over time [34,35,48–58].
(iv) RNNs, designed for sequential data, face the vanishing gradient problem, which reduces their ability to retain information over long sequences. This makes it harder for RNNs to detect long-term dependencies in network traffic, an issue further exacerbated by the need for careful tuning of learning rates and optimizers to mitigate gradient issues [39,59–61].
(v) LSTM networks, a type of RNN, address some of the vanishing gradient issues but introduce the challenge of high training times due to their complex internal structure. LSTMs are ideal for capturing long-term dependencies in network traffic but require significant computational resources, and their numerous hyperparameters (e.g., memory cell size and gating mechanisms) make them difficult to tune effectively [36,37,62–68].
(vi) RBMs struggle with capturing complex patterns in network traffic. Their reliance on contrastive divergence for training limits their effectiveness when dealing with high-dimensional traffic data. The number of hidden units and the learning rate must be finely adjusted to balance computational efficiency and model performance, which can be difficult to achieve [38,69–74].
(vii) DBNs, which are built using RBMs, face the challenge of complex layer-by-layer training. DBNs require unsupervised pre-training followed by supervised fine-tuning, which is computationally expensive and prone to suboptimal performance if not properly configured. The number of RBM layers and units per layer directly impacts the learning capacity, adding to the training complexity. MLPs, although simpler than some other DL methods, have limitations in their generalization ability when applied to complex scenarios with high-dimensional network data [40,75–82].
(viii) MLPs are fully connected, which makes them prone to overfitting, especially in noisy or imbalanced datasets. Proper regularization techniques, such as dropout, are required, along with careful tuning of hidden layer sizes, to maintain performance without overfitting [41,42,83–85].
(ix) Finally, Ensemble Methods, which combine multiple models, offer high accuracy but at the cost of increased computational complexity and training time. While combining models such as CNNs, RNNs, and DNNs can improve detection rates by capturing different aspects of network traffic, these methods require extensive tuning to balance model diversity and performance, making them resource-intensive and difficult to implement in real-time NIDS.

In conclusion, the challenges faced by these DL methods in NIDS are largely due to the nature of network traffic data, which is often high-dimensional, imbalanced, and temporal. Configuring these models effectively requires careful tuning of hyperparameters, balancing model complexity with computational efficiency, and adapting the models to the specific characteristics of the network data.

11.2. Challenges and considerations in applying deep learning for critical systems

We find out that deploying deep learning-based network intrusion detection systems (NIDS) in critical infrastructures, such as power grids, healthcare systems, and financial networks, presents unique challenges and considerations [32,42]. Critical systems demand the highest levels of reliability, robustness, and real-time performance. While deep learning models can significantly enhance the detection of sophisticated and evolving threats, their application in such environments must be approached with caution [92]. One key challenge is the "black box" nature of deep learning models, which can hinder the transparency and explainability required for critical decision-making processes [102]. This lack of interpretability may lead to hesitancy in adopting these models in environments where understanding the rationale behind each decision is crucial [92].

Moreover, the resource-intensive nature of deep learning algorithms, both in terms of computational power and data requirements, can pose challenges for real-time intrusion detection in critical systems, where latency and efficiency are paramount [99]. Additionally, the risk of adversarial attacks targeting the deep learning model itself can undermine the security of the system [105].

Despite these challenges, deep learning holds immense potential for enhancing the security of critical systems [108]. To ensure their safe and effective deployment, ongoing research is needed to improve the interpretability, efficiency, and resilience of these models.

11.3. Issues with the datasets used for deep learning training for NIDS

It is essential to acknowledge that while the datasets discussed in Section 7, are foundational for developing and evaluating models, they also present certain challenges that impact their suitability and generalizability.

(i) One major issue is the age and relevance of some datasets, such as NSL-KDD and KDD-CUP99, which are derived from older network environments [23,70]. Although these datasets have been extensively used and are well-understood, they may not accurately reflect modern network traffic and attack vectors, leading to models that might be less effective against contemporary threats. The changing landscape of cybersecurity, with more sophisticated and varied attack techniques, necessitates datasets that are updated and representative of current network conditions [24,26].
(ii) Another significant challenge is the class imbalance present in many datasets, where the number of instances of normal traffic vastly outweighs the instances of attacks, or certain attack types are underrepresented [89–96]. This imbalance can lead to models that are biased toward predicting the majority class, resulting in higher false negative rates for the minority class, which in the

case of intrusion detection, could mean missing critical attacks [89–96].

(iii) The lack of diversity in some datasets also limits their applicability. For instance, datasets like UNSW-NB15 and CIC-IDS2017 are more recent and include a wider variety of attack types, but they still may not cover the full spectrum of potential threats, especially in specialized environments like IoT or industrial control systems [31,46]. This lack of diversity can lead to overfitting, where models perform well on specific datasets but struggle in real-world scenarios with different attack patterns [50].

(iv) Furthermore, many datasets, including Bot-IoT and IoT-23, focus on specific environments or types of devices, which can limit their generalizability across different network settings [63]. The specific features or attack types prevalent in these datasets may not be representative of broader network environments, which could result in models that perform well only in the context for which they were trained [87].

(v) Finally, data quality issues, such as noise, redundant features, or missing values, can adversely affect the performance of deep learning models [46,71]. High-quality preprocessing and feature selection techniques are required to mitigate these issues, but they add complexity to the model development process [97–116].

In conclusion, while the datasets discussed are invaluable for the development of NIDS, their suitability must be critically evaluated concerning the specific application and environment in which the model is to be deployed. Future work should focus on developing more representative, balanced, and diverse datasets to improve the robustness and generalizability of deep learning models in intrusion detection.

11.4. Significance of less complex models to multi-class problems in NIDS

In the context of Network Intrusion Detection Systems (NIDS), the complexity of deep learning models, while often correlated with high performance, can pose challenges in terms of computational overhead, interpretability, and deployment feasibility, especially when addressing multi-class classification problems [22,27,47]. The need for less complex models becomes even more critical in environments where real-time analysis and resource efficiency are paramount, such as in edge computing or low-latency networks [98].

The application of deep learning in NIDS typically involves complex architectures like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), or hybrid models that combine multiple techniques [35,39]. While these models can achieve high accuracy, their complexity may hinder their applicability in real-world scenarios where computational resources are limited [98]. Moreover, the interpretability of these models is often low, making it difficult for security analysts to understand the decision-making process, which is crucial for trust and regulatory compliance in critical systems.

11.5. Potential approaches to less-complex models

(i) Dimensionality Reduction Techniques: One approach to reducing complexity is the application of dimensionality reduction techniques such as Principal Component Analysis (PCA) or t-Distributed Stochastic Neighbor Embedding (t-SNE) before feeding data into the model. This can help in reducing the feature space, thus simplifying the model without significantly compromising performance [100].

(ii) Model Pruning and Quantization: Techniques like model pruning, where unnecessary weights are removed, and quantization, which reduces the precision of the weights, can be employed to create lighter versions of complex models. These methods can significantly reduce the model size and inference time, making them more suitable for real-time multi-class NIDS [104] .

(iii) Shallow Neural Networks: For multi-class problems, shallow neural networks can be explored. These models have fewer layers and parameters, making them less complex and faster to train. Although they may not capture as intricate patterns as deeper networks, they can be effective with well-engineered features and are easier to interpret [68].

(iv) Optimization of Hyperparameters: Automated hyperparameter optimization methods like Bayesian Optimization or Grid Search can be used to tune models, potentially finding simpler configurations that offer good performance. This approach ensures that the model is neither over-engineered nor unnecessarily complex [96].

In summary, while deep learning offers powerful tools for multi-class NIDS, the complexity of these models can limit their practical applicability. By exploring and applying techniques that reduce model complexity, such as dimensionality reduction, model pruning and shallow networks, one can develop more efficient and deployable NIDS that maintain high performance.

12. Conclusion

This survey has systematically reviewed the contemporary landscape of Network Intrusion Detection Systems (NIDS), emphasizing the integration of Deep Learning (DL) methodologies. By following the PRISMA 2020 guidelines, a comprehensive synthesis of the current research on NIDS was performed, covering key aspects such as data preparation techniques, DL models, performance metrics, datasets, and implementation tools.

The review highlights the growing preference for Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNN) due to their superior performance in handling sequential data and complex patterns in network traffic. Data preparation techniques, especially those focusing on normalization and augmentation, have been shown to be critical in enhancing model accuracy and robustness, addressing the variability and scale of network datasets.

Performance metrics such as accuracy, precision, and recall remain central to evaluating NIDS, with a notable emphasis on reducing false alarm rates—a persistent challenge in the field. The analysis of various datasets revealed that while NSL-KDD, KDD-CUP99, and UNSW-NB15 are predominantly used, there is an emerging trend towards employing more diverse and realistic datasets to better simulate real-world scenarios.

Moreover, the application of NIDS across domains like IoT, healthcare, and smart cities underscores the versatility and necessity of these systems in safeguarding modern infrastructures. The review also identified the trend towards developing less complex, yet effective, models suitable for multi-class problems—an area ripe for future research.

In conclusion, this survey not only consolidates the vast research on DL-based NIDS but also identifies key areas for future exploration. By synthesizing the contributions, challenges, and trends in this domain, the paper provides a valuable reference for researchers aiming to develop more effective, scalable, and adaptable NIDS solutions. The continuous evolution of network environments demands equally dynamic and innovative approaches in NIDS, with DL at the forefront of this advancement.

CRediT authorship contribution statement

Ramya Chinnasamy: Writing – original draft, Methodology, Conceptualization. **Malliga Subramanian:** Writing – review & editing, Writing – original draft, Validation, Conceptualization. **Sathishkumar Veerappampalayam Easwaramoorthy:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Jaehyuk Cho:** Writing – review & editing,

Writing – original draft, Visualization, Supervision, Funding acquisition.

Declaration of competing interests

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

Acknowledgement

This work was supported the Korea Environmental Industry & Technology Institute(KEITI), with a grant funded by the Korea government, Ministry of Environment(The development of IoT-based technology for collecting and managing big data on environmental hazards and health effects), under Grant RE202101551.

References

[1] E. Aroms, NIST Special Publication 800-94 Guide to Intrusion Detection and Prevention Systems (IDPS), CreateSpace, 2012.
[2] J.P. Anderson, "Computer security threat monitoring and surveillance," Technical Report, James P. Anderson Company, 1980.
[3] R. Chinnasamy, S. Malliga, N. Sengupta, Deep learning-driven intrusion detection systems for smart cities-a systematic study, in: 6th Smart Cities Symposium (SCS 2022), 2022, pp. 79–84.
[4] M. Saied, S. Guirguis, M. Madbouly, Review of artificial intelligence for enhancing intrusion detection in the internet of things, Eng. Appl. Artif. Intell. 127 (2024) 107231.
[5] O.H. Abdulganiyu, T.A. Tchakoucht, Y.K. Saheed, Towards an efficient model for network intrusion detection system (IDS): systematic literature review, Wireless Net. 30 (2024) 453–482.
[6] D. Musleh, M. Alotaibi, F. Alhaidari, A. Rahman, R.M. Mohammad, Intrusion detection system using feature extraction with machine learning algorithms in IoT, J. Sens. Actuat. Net. 12 (2023) 29.
[7] A. Halbouni, T.S. Gunawan, M.H. Habaebi, M. Halbouni, M. Kartiwi, R. Ahmad, Machine learning and deep learning approaches for cybersecurity: A review, IEEe Access. 10 (2022) 19572–19585.
[8] O.H. Abdulganiyu, T.Ait Tchakoucht, Y.K. Saheed, A systematic literature review for network intrusion detection system (IDS), Int. J. Inf. Secur. 22 (2023) 1125–1162.
[9] J. Lansky, S. Ali, M. Mohammadi, M.K. Majeed, S.H.T. Karim, S. Rashidi, et al., Deep Learning-Based Intrusion Detection Systems: A Systematic Review, IEEe Access. 9 (2021) 101574–101599.
[10] A. Thakkar, R. Lohiya, A review on machine learning and deep learning perspectives of IDS for IoT: recent updates, security issues, and challenges, Arch. Comput. Meth. Eng. 28 (2021) 3211–3243.
[11] S. Gamage, J. Samarabandu, Deep learning methods in network intrusion detection: A survey and an objective comparison, J. Net. Comput. Appl. 169 (2020) 102767.
[12] D. Moher, A. Liberati, J. Tetzlaff, D.G. Altman, t. PRISMA Group*, Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement, Ann. Intern. Med. 151 (2009) 264–269.
[13] J.P. Higgins, S. Green, Cochrane Handbook For Systematic Reviews of Interventions, 2008.
[14] M.J. Page, J.E. McKenzie, P.M. Bossuyt, I. Boutron, T.C. Hoffmann, C.D. Mulrow, et al., The PRISMA 2020 statement: an updated guideline for reporting systematic reviews, BMJ 372 (2021).
[15] A. Binbusayyis, H. Alaskar, T. Vaiyapuri, M. Dinesh, An investigation and comparison of machine learning approaches for intrusion detection in IoMT network, J. Supercomput. 78 (2022) 17403–17422.
[16] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, Survey of intrusion detection systems: techniques, datasets and challenges, Cybersecur. (Singap) 2 (2019) 1–22.
[17] A.A.A. Lateef, S. Al-Janabi, B. Al-Khateeb, Survey on intrusion detection systems based on deep learning, Period. Eng. Nat. Sci. 7 (2019) 1074–1095.
[18] P. Mishra, V. Varadharajan, U. Tupakula, E.S. Pilli, A detailed investigation and analysis of using machine learning techniques for intrusion detection, IEEE Commun. Survey. Tutor. 21 (2018) 686–728.
[19] N. Sultana, N. Chilamkurti, W. Peng, R. Alhadad, Survey on SDN based network intrusion detection system using machine learning approaches, Peer. Peer. Netw. Appl. 12 (2018) 493–501.
[20] R. Chinnasamy, M. Subramanian, Detection of Malicious Activities by Smart Signature-Based IDS. Artificial Intelligence for Intrusion Detection Systems, Chapman and Hall/CRC, 2023, pp. 63–78.
[21] J. Han, J. Pei, H. Tong, Data mining: Concepts and Techniques, Morgan kaufmann, 2022.
[22] N. Shone, T.N. Ngoc, V.D. Phai, Q. Shi, A deep learning approach to network intrusion detection, IEEe Trans. Emerg. Top. Comput. Intell. 2 (2018) 41–50.
[23] D. He, Q. Qiao, Y. Gao, J. Zheng, S. Chan, J. Li, et al., Intrusion detection based on stacked autoencoder for connected healthcare systems, IEEE Network 33 (2019) 64–69.
[24] G. Muhammad, M.S. Hossain, S. Garg, Stacked autoencoder-based intrusion detection system to combat financial fraudulent, IEEe Internet. Things. J. (2020).
[25] A. Tabassum, A. Erbad, A. Mohamed, M. Guizani, Privacy-preserving distributed IDS using incremental learning for IoT health systems, IEEe Access. 9 (2021) 14271–14283.
[26] D. Papamartzivanos, F.G. Mármol, G. Kambourakis, Introducing deep learning self-adaptive misuse network intrusion detection systems, IEEe Access. 7 (2019) 13546–13560.
[27] V. Dutta, M. Choraś, M. Pawlicki, R. Kozik, A deep learning ensemble for network anomaly and cyber-attack detection, Sensors 20 (2020) 4583.
[28] C. Tang, N. Luktarhan, Y. Zhao, SAAE-DNN: Deep learning method on intrusion detection, Symmetry. (Basel) 12 (2020) 1695.
[29] A. Bhardwaj, V. Mangat, R. Vig, Hyperband tuned deep neural network with well posed stacked sparse autoencoder for detection of DDoS attacks in cloud, IEEe Access. 8 (2020) 181916–181929.
[30] Y.N. Kunang, S. Nurmaini, D. Stiawan, B.Y. Suprapto, Attack classification of an intrusion detection system using deep learning and hyperparameter optimization, J. Inform. Secur. Appl. 58 (2021) 102804.
[31] J. Lee, K. Park, AE-CGAN model based high performance network intrusion detection system, Appl. Sci. 9 (2019) 4221.
[32] J.B. Awotunde, C. Chakraborty, A.E. Adeniyi, Intrusion detection in industrial internet of things network-based on deep learning model with rule-based feature selection, Wireless Commun. Mobile Comput. 2021 (2021) 1–17.
[33] S. Naseer, Y. Saleem, S. Khalid, M.K. Bashir, J. Han, M.M. Iqbal, et al., Enhanced network anomaly detection based on deep neural networks, IEEe Access. 6 (2018) 48231–48246.
[34] J.K. Pandey, S. Kumar, M. Lamin, S. Gupta, R.K. Dubey, F. Sammy, A Metaheuristic autoencoder deep learning model for intrusion detector system, Math. Probl. Eng. 2022 (2022).
[35] M.M. Hassan, A. Gumaei, A. Alsanad, M. Alrubaian, G. Fortino, A hybrid deep learning model for efficient intrusion detection in big data environment, Inform. Sci. 513 (2020) 386–396.
[36] X. Fu, N. Zhou, L. Jiao, H. Li, J. Zhang, The robust deep learning–based schemes for intrusion detection in internet of things environments, Ann. Telecommun. 76 (2021) 273–285.
[37] S.M.S. Bukhari, M.H. Zafar, M. Abou Houran, S.K.R. Moosavi, M. Mansoor, M. Muaaz, et al., Secure and privacy-preserving intrusion detection in wireless sensor networks: Federated learning with SCNN-Bi-LSTM for enhanced reliability, Ad. Hoc. Netw. 155 (2024) 103407.
[38] A. Elsaeidy, K.S. Munasinghe, D. Sharma, A. Jamalipour, Intrusion detection in smart cities using Restricted Boltzmann Machines, J. Net. Comput. Appl. 135 (2019) 76–83.
[39] S.M. Kasongo, A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework, Comput. Commun. 199 (2023) 113–125.
[40] H. Zhang, Y. Li, Z. Lv, A.K. Sangaiah, T. Huang, A real-time and ubiquitous network attack detection based on deep belief network and support vector machine, IEEE/CAA J. Autom. Sinica 7 (2020) 790–799.
[41] I. Ashraf, M. Narra, M. Umer, R. Majeed, S. Sadiq, F. Javaid, et al., A deep learning-based smart framework for cyber-physical and satellite system security threats detection, Electronics. (Basel) 11 (2022) 667.
[42] O.A. Alimi, K. Ouahada, A.M. Abu-Mahfouz, Real time security assessment of the power system using a hybrid support vector machine and multilayer perceptron neural network algorithms, Sustainability. 11 (2019) 3586.
[43] S.P. RM, P.K.R. Maddikunta, M. Parimala, S. Koppu, T.R. Gadekallu, C. L. Chowdhary, et al., An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture, Comput. Commun. 160 (2020) 139–149.
[44] O.A. Wahab, Intrusion detection in the iot under data and concept drifts: Online deep learning approach, IEEe Internet. Things. J. 9 (2022) 19706–19716.
[45] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, A. Al-Nemrat, S. Venkatraman, Deep learning approach for intelligent intrusion detection system, IEEe Access. 7 (2019) 41525–41550.
[46] S. Alzughaibi, S. El Khediri, A cloud intrusion detection systems based on dnn using backpropagation and pso on the cse-cic-ids2018 dataset, Appl. Sci. 13 (2023) 2276.
[47] K. He, D.D. Kim, M.R. Asghar, NIDS-Vis: Improving the generalized adversarial robustness of network intrusion detection system, Comput. Secur. (2024) 104028.
[48] A. Anand, S. Rani, D. Anand, H.M. Aljahdali, D. Kerr, An efficient CNN-based deep learning model to detect malware attacks (CNN-DMA) in 5G-IoT healthcare applications, Sensors 21 (2021) 6346.
[49] M. Usman, M.A. Jan, A. Jolfaei, SPEED: a deep learning assisted privacy-preserved framework for intelligent transportation systems, IEEE Trans. Intell. Transp. Syst. 22 (2020) 4376–4384.
[50] A. Kim, M. Park, D.H. Lee, AI-IDS: Application of deep learning to real-time Web intrusion detection, IEEe Access. 8 (2020) 70245–70261.
[51] S. Abaimov, G. Bianchi, CODDLE: Code-injection detection with deep learning, IEEe Access. 7 (2019) 128617–128627.
[52] S. Naseer, Y. Saleem, Enhanced network intrusion detection using deep convolutional neural networks, KSII Trans. Int. Inf. Syst. (TIIS) 12 (2018) 5159–5178.
[53] R. Doriguzzi-Corin, S. Millar, S. Scott-Hayward, J. Martinez-del-Rincon, D. Siracusa, LUCID: A practical, lightweight deep learning solution for DDoS attack detection, IEEE Trans. Net. Serv. Manag. 17 (2020) 876–889.
[54] G. Andresini, A. Appice, D. Malerba, Nearest cluster-based intrusion detection through convolutional neural networks, Knowl. Based. Syst. 216 (2021) 106798.

[55] B. Riyaz, S. Ganapathy, A deep learning approach for effective intrusion detection in wireless networks using CNN, Soft. comput. 24 (2020) 17265–17278.
[56] X. Wang, S. Yin, H. Li, J. Wang, L. Teng, A network intrusion detection method based on deep multi-scale convolutional neural network, Int. J. Wirel. Inf. Netw. 27 (2020) 503–517.
[57] R. Ben Said, Z. Sabir, I. Askerzade, CNN-BiLSTM: A Hybrid Deep Learning Approach for Network Intrusion Detection System in Software-Defined Networking With Hybrid Feature Selection, IEEe Access. 11 (2023) 138732–138747.
[58] Y. Imrana, Y. Xiang, L. Ali, A. Noor, K. Sarpong, M.A. Abdullah, CNN-GRU-FF: a double-layer feature fusion-based network intrusion detection system using convolutional neural network and gated recurrent units, Complex. Intell. Systems. (2024) 1–18.
[59] R. Ahmad, I. Alsmadi, W. Alhamdani, L.a. Tawalbeh, A comprehensive deep learning benchmark for IoT IDS, Comput. Secur. 114 (2022) 102588.
[60] T.-T.-H. Le, Y. Kim, H. Kim, Network intrusion detection based on novel feature selection model and various recurrent neural networks, Appl. Sci. 9 (2019) 1392.
[61] C. Yue, L. Wang, D. Wang, R. Duo, X. Nie, An ensemble intrusion detection method for train ethernet consist network based on CNN and RNN, IEEe Access. 9 (2021) 59527–59539.
[62] N. Islam, F. Farhin, I. Sultana, M.S. Kaiser, M.S. Rahman, M. Mahmud, et al., Towards Machine Learning Based Intrusion Detection in IoT Networks, Comput. Mater. Contin. 69 (2021).
[63] A.V. Hanafi, A. Ghaffari, H. Rezaei, A. Valipour, B. arasteh, Intrusion detection in Internet of things using improved binary golden jackal optimization algorithm and LSTM, Cluster. Comput. (2023) 1–18.
[64] J. Han, W. Pak, High performance network intrusion detection system using two-stage LSTM and incremental created hybrid features, Electronics. (Basel) 12 (2023) 956.
[65] M. Mahdavisharif, S. Jamali, R. Fotohi, Big data-aware intrusion detection system in communication networks: a deep learning approach, J. Grid. Comput. 19 (2021) 1–28.
[66] K. Al Jallad, M. Aljnidi, M.S. Desouki, Anomaly detection optimization using big data and deep learning to reduce false-positive, J. Big. Data 7 (2020) 1–12.
[67] S.K. Sahu, D.P. Mohapatra, J.K. Rout, K.S. Sahoo, Q.-V. Pham, N.-N. Dao, A LSTM-FCNN based multi-class intrusion detection using scalable framework, Comput. Electric. Eng. 99 (2022) 107720.
[68] R.-H. Hwang, M.-C. Peng, V.-L. Nguyen, Y.-L. Chang, An LSTM-based deep learning approach for classifying malicious traffic at the packet level, Appl. Sci. 9 (2019) 3414.
[69] G.H.d. Rosa, M. Roder, D.F. Santos, K.A. Costa, Enhancing anomaly detection through restricted boltzmann machine features projection, Int. J. Inf. Technol. 13 (2021) 49–57.
[70] T. Aldwairi, D. Perera, M.A. Novotny, An evaluation of the performance of Restricted Boltzmann Machines as a model for anomaly network intrusion detection, Comput. Netw. 144 (2018) 111–119.
[71] R. Atefinia, M. Ahmadi, Network intrusion detection using multi-architectural modular deep neural network, J. Supercomput. 77 (2021) 3571–3593.
[72] T. Aldwairi, D. Perera, M.A. Novotny, Measuring the impact of accurate feature selection on the performance of RBM in comparison to state of the art machine learning algorithms, Electronics. (Basel) 9 (2020) 1167.
[73] N. Mathappan, S. Elavarasan, S. Sehar, Hybrid intelligent intrusion detection system for multiple Wi-Fi attacks in wireless networks using stacked restricted Boltzmann machine and deep belief networks, Concurr. Comput. Pract. Exper. 35 (2023) e7769.
[74] J. Srivastava, J. Prakash, Deep learning-enabled energy optimization and intrusion detection for wireless sensor networks, OPSEARCH (2024) 1–38.
[75] G.N. Nguyen, N.H. Le Viet, M. Elhoseny, K. Shankar, B. Gupta, A.A. Abd El-Latif, Secure blockchain enabled Cyber–physical systems in healthcare using deep belief network with ResNet model, J. Parallel. Distrib. Comput. 153 (2021) 150–160.
[76] A. Alsirhani, M.M. Alshahrani, A.M. Hassan, A.I. Taloba, R.M. Abd El-Aziz, A. H. Samak, Implementation of African vulture optimization algorithm based on deep learning for cybersecurity intrusion detection, Alexand. Eng. J. 79 (2023) 105–115.
[77] N. Balakrishnan, A. Rajendran, D. Pelusi, V. Ponnusamy, Deep Belief Network enhanced intrusion detection system to prevent security breach in the Internet of Things, Int. things 14 (2021) 100112.
[78] Y. Yang, K. Zheng, C. Wu, X. Niu, Y. Yang, Building an effective intrusion detection system using the modified density peak clustering algorithm and deep belief networks, Appl. Sci. 9 (2019) 238.
[79] P. Wei, Y. Li, Z. Zhang, T. Hu, Z. Li, D. Liu, An optimization method for intrusion detection classification model based on deep belief network, IEEe Access. 7 (2019) 87593–87605.
[80] N. Sarkar, P.K. Keserwani, M.C. Govil, A better and fast cloud intrusion detection system using improved squirrel search algorithm and modified deep belief network, Cluster. Comput. 27 (2024) 1699–1718.
[81] P. Sajith, G. Nagarajan, Intrusion detection system using deep belief network & particle swarm optimization, Wirel. Pers. Commun. 125 (2022) 1385–1403.
[82] N. Marir, H. Wang, G. Feng, B. Li, M. Jia, Distributed abnormal behavior detection approach based on deep belief network and ensemble SVM using spark, IEEe Access. 6 (2018) 59657–59671.
[83] W.A. Ghanem, A. Jantan, A new approach for intrusion detection system based on training multilayer perceptron by using enhanced Bat algorithm, Neural Comput. Appl. 32 (2020) 11665–11698.
[84] M. Alazab, R.A. Khurma, P.A. Castillo, B. Abu-Salih, A. Martín, D. Camacho, An effective networks intrusion detection approach based on hybrid Harris Hawks and multi-layer perceptron, Egypt. Inform. J. 25 (2024) 100423.
[85] Y. Yin, J. Jang-Jaccard, W. Xu, A. Singh, J. Zhu, F. Sabrina, et al., IGRF-RFE: a hybrid feature selection method for MLP-based network intrusion detection on UNSW-NB15 dataset, J. Big. Data 10 (2023) 15.
[86] A.A.E.-B. Donkol, A.G. Hafez, A.I. Hussein, M.M. Mabrook, Optimization of Intrusion Detection Using Likely Point PSO and Enhanced LSTM-RNN Hybrid Technique in Communication Networks, IEEe Access. 11 (2023) 9469–9482.
[87] S. Ullah, J. Ahmad, M.A. Khan, E.H. Alkhammash, M. Hadjouni, Y.Y. Ghadi, et al., A new intrusion detection system for the internet of things via deep convolutional neural network and feature engineering, Sensors 22 (2022) 3607.
[88] J. Figueiredo, C. Serrão, A.M. de Almeida, Deep learning model transposition for network intrusion detection systems, Electronics. (Basel) 12 (2023) 293.
[89] A. Srivastava, D. Sinha, V. Kumar, WCGAN-GP based synthetic attack data generation with GA based feature selection for IDS, Comput. Secur. 134 (2023) 103432.
[90] A. Alabrah, Scientific Elegance in NIDS: Unveiling Cardinality Reduction, Box-Cox Transformation, and ADASYN for Enhanced Intrusion Detection, Comput. Mater. Contin. 79 (2024).
[91] A. Abdelkhalek, M. Mashaly, Addressing the class imbalance problem in network intrusion detection systems using data resampling and deep learning, J. Supercomput. 79 (2023) 10611–10644.
[92] D. Liu, S. Zhong, L. Lin, M. Zhao, X. Fu, X. Liu, Deep attention SMOTE: Data augmentation with a learnable interpolation factor for imbalanced anomaly detection of gas turbines, Comput. Ind. 151 (2023) 103972.
[93] G. Andresini, A. Appice, L. De Rose, D. Malerba, GAN augmentation to deal with imbalance in imaging-based intrusion detection, Future Gener. Comput. Syst. 123 (2021) 108–127.
[94] Y.N. Rao, K.Suresh Babu, An imbalanced generative adversarial network-based approach for network intrusion detection in an imbalanced dataset, Sensors 23 (2023) 550.
[95] G. Aceto, F. Giampaolo, C. Guida, S. Izzo, A. Pescapè, F. Piccialli, et al., Synthetic and privacy-preserving traffic trace generation using generative AI models for training Network Intrusion Detection Systems, J. Net. Comput. Appl. (2024) 103926.
[96] O.H. Abdulganiyu, T.A. Tchakoucht, Y.K. Saheed, H.A. Ahmed, XIDINTFL-VAE: XGBoost-based intrusion detection of imbalance network traffic via class-wise focal loss variational autoencoder, J. Supercomput. 81 (2025) 1–38.
[97] S.M. Kasongo, Y. Sun, A deep learning method with wrapper based feature extraction for wireless intrusion detection system, Comput. Secur. 92 (2020) 101752.
[98] Z. Liu, N. Thapa, A. Shaver, K. Roy, M. Siddula, X. Yuan, et al., Using embedded feature selection and CNN for classification on CCD-INID-V1—A new IoT dataset, Sensors 21 (2021) 4834.
[99] P.T. Nguyen, V.D.B. Huynh, K.D. Vo, P.T. Phan, M. Elhoseny, D.-N. Le, Deep Learning Based Optimal Multimodal Fusion Framework for Intrusion Detection Systems for Healthcare Data, Comput. Mater. Contin. 66 (2021).
[100] N. Khare, P. Devan, C.L. Chowdhary, S. Bhattacharya, G. Singh, S. Singh, et al., Smo-dnn: Spider monkey optimization and deep neural network hybrid classifier model for intrusion detection, Electronics. (Basel) 9 (2020) 692.
[101] A. Kavousi-Fard, M. Dabbaghjamanesh, T. Jin, W. Su, M. Roustaei, An evolutionary deep learning-based anomaly detection model for securing vehicles, IEEE Trans. Intell. Transp. Syst. 22 (2020) 4478–4486.
[102] L. Almutairi, R. Daniel, S. Khasimbee, E.L. Lydia, S. Acharya, H. Kim, Quantum Dwarf Mongoose Optimization with Ensemble Deep Learning Based Intrusion Detection in Cyber-Physical Systems, IEEe Access. (2023).
[103] N. Hussen, S.M. Elghamrawy, M. Salem, A.I. El-Desouky, A Fully Streaming Big Data Framework for Cyber Security based on Optimized Deep Learning Algorithm, IEEe Access. (2023).
[104] B. Deore, S. Bhosale, Hybrid optimization enabled robust CNN-LSTM technique for network intrusion detection, IEEe Access. 10 (2022) 65611–65622.
[105] H. Alamro, R. Marzouk, N. Alruwais, N. Negm, S.S. Aljameel, M. Khalid, et al., Modelling of Blockchain Assisted Intrusion Detection on IoT Healthcare System using Ant Lion Optimizer with Hybrid Deep Learning, IEEe Access. (2023).
[106] F. Taher, M. Elhoseny, M.K. Hassan, I.M. El-Hasnony, A Novel Tunicate Swarm Algorithm With Hybrid Deep Learning Enabled Attack Detection for Secure IoT Environment, IEEe Access. 10 (2022) 127192–127204.
[107] R. Chinnasamy, M. Subramanian, N. Sengupta, Designing of Intrusion Detection System Using an Ensemble of Artificial Neural Network and Honey Badger Optimization Algorithm, in: 2023 International Conference on IT Innovation and Knowledge Discovery (ITIKD), 2023, pp. 1–6.
[108] Y.K. Saheed, M.O. Arowolo, Efficient cyber attack detection on the internet of medical things-smart environment based on deep recurrent neural network and machine learning algorithms, IEEe Access. 9 (2021) 161546–161554.
[109] B. Lahasan, H. Samma, Optimized deep autoencoder model for internet of things intruder detection, IEEe Access. 10 (2022) 8434–8448.
[110] Y.K. Saheed, O.H. Abdulganiyu, T.A. Tchakoucht, Modified genetic algorithm and fine-tuned long short-term memory network for intrusion detection in the internet of things networks with edge capabilities, Appl. Soft. Comput. 155 (2024) 111434.
[111] M.H. Aghdam, P. Kabiri, Feature selection for intrusion detection system using ant colony optimization, Int. J. Netw. Secur. 18 (2016) 420–432.
[112] R. Sekhar, K. Sasirekha, P. Raja, K. Thangavel, A novel GPU based intrusion detection system using deep autoencoder with Fruitfly optimization, Appl. Sci.SN. Appl. Sci. 3 (2021) 594.

[113] Y. Li, S.-m. Ghoreishi, A. Issakhov, Improving the Accuracy of Network Intrusion Detection System in Medical IoT Systems through Butterfly Optimization Algorithm, Wirel. Pers. Commun. 126 (2022) 1999–2017.
[114] M. Alweshah, A. Hammouri, S. Alkhalaileh, O. Alzubi, Intrusion detection for the internet of things (IoT) based on the emperor penguin colony optimization algorithm, J. Ambient. Intell. Humaniz. Comput. (2022) 1–18.
[115] A. Hosseinalipour, R. Ghanbarzadeh, A novel approach for spam detection using horse herd optimization algorithm, Neural Comput. Appl. 34 (2022) 13091–13105.
[116] M. Imran, S. Khan, H. Hlavacs, F.A. Khan, S. Anwar, Intrusion detection in networks using cuckoo search optimization, Soft. comput. (2022) 1–13.
[117] S. Choudhary, N. Kesswani, Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 datasets using deep learning in IoT, Procedia Comput. Sci. 167 (2020) 1561–1573.
[118] K. Siddique, Z. Akhtar, F.A. Khan, Y. Kim, KDD cup 99 data sets: A perspective on the role of data sets in network intrusion detection research, Computer. (Long. Beach. Calif) 52 (2019) 41–51.
[119] N. Moustafa, J. Slay, The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set, Inf. Secur. J. A Glob. Persp. 25 (2016) 18–31.
[120] A. Parmisano, S. Garcia, M.J. Erquiaga, A labeled dataset with malicious and benign iot network traffic, Stratosph. Lab. Praha, Czech Republic (2020).
[121] D. Stiawan, M.Y.B. Idris, A.M. Bamhdi, R. Budiarto, CICIDS-2017 dataset feature analysis with information gain for anomaly detection, IEEe Access. 8 (2020) 132911–132921.
[122] A. Gharib, I. Sharafaldin, A.H. Lashkari, A.A. Ghorbani, An evaluation framework for intrusion detection dataset, in: 2016 International conference on information science and security (ICISS), 2016, pp. 1–6.
[123] J.L. Leevy, T.M. Khoshgoftaar, A survey and analysis of intrusion detection models based on cse-cic-ids2018 big data, J. Big. Data 7 (2020) 1–19.
[124] N. Koroniotis, N. Moustafa, E. Sitnikova, B. Turnbull, Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset, Future Gener. Comput. Syst. 100 (2019) 779–796.
[125] N. Koroniotis, Designing an Effective Network Forensic Framework For the Investigation of Botnets in the Internet of Things, UNSW Sydney, 2020.
[126] E. Chatzoglou, G. Kambourakis, C. Kolias, Empirical evaluation of attacks against IEEE 802.11 enterprise networks: The AWID3 dataset, IEEe Access. 9 (2021) 34188–34205.
[127] A. Lavin, S. Ahmad, Evaluating real-time anomaly detection algorithms–the Numenta anomaly benchmark, in: 2015 IEEE 14th international conference on machine learning and applications (ICMLA), 2015, pp. 38–44.
[128] B. Borisaniya, D. Patel, Evaluation of modified vector space representation using adfa-ld and adfa-wd datasets, J. Inform. Secur. 6 (2015) 250–264.
[129] A. Verma, V. Ranga, Statistical analysis of CIDDS-001 dataset for network intrusion detection systems using distance-based machine learning, Procedia Comput. Sci. 125 (2018) 709–716.
[130] B. Wang, Y. Su, M. Zhang, J. Nie, A deep hierarchical network for packet-level malicious traffic detection, IEEe Access. 8 (2020) 201728–201740.
[131] M. Ghurab, G. Gaphari, F. Alshami, R. Alshamy, S. Othman, A detailed analysis of benchmark datasets for network intrusion detection system, Asian J. Res. Comput. Sci. 7 (2021) 14–33.
[132] E.T. Meyer, Socio-technical interaction networks: A discussion of the strengths, weaknesses and future of Kling's STIN model, in: in *Social Informatics: An Information Society for all? In Remembrance of Rob Kling: Proceedings of the Seventh International Conference on Human Choice and Computers (HCC7), IFIP TC 9*, Maribor, Slovenia, 2006, pp. 37–48. September 21–23, 2006 7.
[133] Y. Zhang, C. Ling, A strategy to apply machine learning to small datasets in materials science, NPJ. Comput. Mater. 4 (2018) 25.
[134] M.M. Najafabadi, T.M. Khoshgoftaar, N. Seliya, Evaluating feature selection methods for network intrusion detection with kyoto data, Int. J. Reliab. Qual. Saf. Eng. 23 (2016) 1650001.
[135] S. Al-Riyami, F. Coenen, A. Lisitsa, A re-evaluation of intrusion detection accuracy: Alternative evaluation strategy, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 2018, pp. 2195–2197.
[136] J. Ha, H. Roh, Experimental evaluation of malware family classification methods from sequential information of tls-encrypted traffic, Electronics. (Basel) 10 (2021) 3180.
[137] R. Damasevicius, A. Venckauskas, S. Grigaliunas, J. Toldinas, N. Morkevicius, T. Aleliunas, et al., LITNET-2020: An annotated real-world network flow dataset for network intrusion detection, Electronics. (Basel) 9 (2020) 800.
[138] M.H. Haghighat, J. Li, Intrusion detection system using voting-based neural network, Tsinghua Sci. Technol. 26 (2021) 484–495.
[139] J. Li, H. Zhang, Z. Wei, The weighted word2vec paragraph vectors for anomaly detection over HTTP traffic, IEEe Access. 8 (2020) 141787–141798.
[140] R. Zhou, X. Wang, J. Yang, W. Zhang, S. Zhang, Characterizing Network Anomaly Traffic with Euclidean Distance-Based Multiscale Fuzzy Entropy, Secur. Commun. Net. 2021 (2021) 5560185.
[141] N. Moustafa, M. Ahmed, S. Ahmed, Data analytics-enabled intrusion detection: Evaluations of ToN_IoT linux datasets, in: 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2020, pp. 727–735.
[142] A. Nicolae, "Deep Learning Framework From Scratch Using Numpy," *arXiv preprint arXiv:2011.08461,* 2020.
[143] A. Gulli, S. Pal, Deep Learning With Keras, Packt Publishing Ltd, 2017.
[144] B. Pang, E. Nijkamp, Y.N. Wu, Deep learning with tensorflow: A review, J. Edu Behav. Stat. 45 (2020) 227–248.
[145] W. McKinney, pandas: a foundational Python library for data analysis and statistics, Python High Perf. Sci. Comput. 14 (2011) 1–9.
[146] P. Barrett, J. Hunter, J.T. Miller, J.-C. Hsu, P. Greenfield, matplotlib–A Portable Python Plotting Package. Astronomical Data Analysis Software and Systems XIV, 2005, p. 91.
[147] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, et al., Pytorch: An imperative style, high-performance deep learning library, Adv. Neural Inf. Process. Syst. 32 (2019).
[148] O. Kramer, O. Kramer, Scikit-learn, Mach. Learn. Evolut. Strateg. (2016) 45–53.
[149] P. Virtanen, R. Gommers, T.E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, et al., SciPy 1.0: fundamental algorithms for scientific computing in Python, Nat. Methods 17 (2020) 261–272.
[150] R. Al-Rfou, G. Alain, A. Almahairi, C. Angermueller, D. Bahdanau, N. Ballas, et al., "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints,* p. arXiv: 1605.02688, 2016.
[151] M. Paluszek, S. Thomas, Practical Matlab deep learning, A Project Based Appr. Michael Paluszek Stephanie Thomas (2020).
[152] L. Pace, Beginning R: An introduction to Statistical Programming, Apress, 2012.
[153] Z. Ahmad, A. Shahid Khan, K. Nisar, I. Haider, R. Hassan, M.R. Haque, et al., Anomaly Detection Using Deep Neural Network for IoT Architecture, Appl. Sci. 11 (2021) 7050.
[154] S. Nandy, M. Adhikari, M.A. Khan, V.G. Menon, S. Verma, An intrusion detection mechanism for secured IoMT framework based on swarm-neural network, IEEe J. Biomed. Health Inform. 26 (2021) 1969–1976.
[155] M. Zeeshan, Q. Riaz, M.A. Bilal, M.K. Shahzad, H. Jabeen, S.A. Haider, et al., Protocol-based deep intrusion detection for dos and ddos attacks using unsw-nb15 and bot-iot data-sets, IEEe Access. 10 (2021) 2269–2283.