SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

Comparing Machine Learning Algorithms for Intrusion Detection Systems

By Lawrence Liu

AUTHOR BIOGRAPHY

Lawrence Liu is a junior at Head-Royce School in California, USA. He enjoys exploring cutting-edge innovations and turning ideas into practical solutions. Lawrence believes that technology can make a meaningful difference for the community.

ABSTRACT

With the advancement of distributed computing systems, cyberattacks are continually evolving, making cybersecurity research crucial for protecting organizations and individuals from network breaches. An Intrusion Detection System (IDS) is designed to monitor software and hardware activities for potential threats. In this study, machine learning (ML) techniques, including both shallow models and deep learning models, are employed to detect a comprehensive set of attack type, including probe, user-to-root (U2R), remote-to-local (R2L), denial of service (DoS), fuzzer, analysis, backdoor, exploit, generic, reconnaissance, shellcode, and worm attacks. Experiments are conducted on three benchmark datasets: KDD99, NSL-KDD, and UNSW-NB15. Among the models tested, Random Forest demonstrated the best performance in terms of accuracy and processing time. Although the deep learning models–Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU)–achieved very high accuracy, their training and prediction times were significantly longer than those of Random Forest. A notable finding was the variation in model performance across the datasets. High accuracy was observed with KDD99 and NSL-KDD, whereas UNSW-NB15 proved more challenging, particularly for multiclass models. This highlights the need for further advancements in IDS models to better address these complex cyber threats.

Keywords: *Intrusion Detection System (IDS), Machine Learning (ML), KDD99, NSL-KDD, UNSW-NB15, k-Nearest Neighbors (kNN), Random Forest, Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU)*

SCHOLARLY REVIEW

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

INTRODUCTION

Given the increasing volume and complexity of cyberattacks in today's world, the importance of robust cybersecurity procedures has never been greater. Cyberattacks involve attempts by malicious actors to breach devices, networks, or systems, posing severe threats to individuals and organizations. The primary goal of cybersecurity is to safeguard private data, systems, and applications from potential damage or destruction (Microsoft Security, 2025). It is projected that cybercrime costs will reach $9.22 trillion by 2024 and could rise to $13.82 trillion by 2028 (Fleck, 2024).

An Intrusion Detection System (IDS) is a crucial network security application designed to protect computer users from cyberattacks (Khamis, 2020). IDS operates by monitoring network traffic for potential threats and malicious activities, which can indicate possible breaches. Intrusion detection can be broadly categorized into two main types: Host-Based IDS (H-IDS) and Network-Based IDS (N-IDS). H-IDS focuses on traffic from individual systems or hosts, comparing it against predefined profiles of normal host activity to detect intrusions. It relies on the host's internal logs, operating systems, and databases to identify threats, but is limited in its ability to detect network-wide threats. In contrast, N-IDS monitors network traffic in real-time, analyzing packet-level data to detect suspicious activities without relying on the host's operating system (Liu & Lang, 2019).

In terms of detection accuracy, IDS encounters several challenges. Researchers have explored various methodologies to reduce both false positives and false negatives. False positives occur when non-attack traffic is incorrectly classified as an attack, leading to unnecessary alarms. False negatives, on the other hand, involve failing to detect actual threats, which can result in significant harm to individuals and organizations.

IDS can also be categorized into Signature-Based IDS (S-IDS) and Anomaly-Based IDS (A-IDS) (Liu & Lang, 2019). S-IDS looks for network packets against known attack signatures from the database; it produces exhaustive reports with low false positives but does a poor job of detecting new threats and variants. In contrast, A-IDS monitors network traffic against a predefined "normal" baseline using machine learning (ML) techniques. It has the advantage of identifying novel attack patterns, although it has a slightly higher false positive rate. Many organizations combine both S-IDS and A-IDS techniques to provide maximum accuracy in detection. The ability to detect new types of attacks will become increasingly important as different threats to cybersecurity continue to evolve with new technologies and devices.

ML has recently become a valuable tool for enhancing IDS, enabling the systems to study data and make informed decisions based on recognized attack patterns. Compared to the traditional methods, ML-based IDS can generalize from past incidents and anticipate new, unknown threats; therefore, it is imperative for advanced systems facing diverse and evolving attacks. This paper applies various ML algorithms to intrusion detection using three benchmark datasets: KDD99, NSL-KDD, and UNSW-NB15. Each dataset presents different dimensions of challenges and opportunities for developing efficient IDS models. This study developed both binary and multiclass classification models using four different ML algorithms: k-Nearest Neighbors

SCHOLARLY REVIEW

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

(kNN), Random Forest, Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU).

METHODS

Machine Learning (ML) Models

As a method for intrusion detection, ML can be employed to differentiate between normal and suspicious network traffic, making it a promising approach for identifying various types of attacks, including previously unknown patterns (Liu & Lang, 2019). Furthermore, certain ML algorithms can achieve high accuracy in predictions while maintaining a reasonable computational cost.

ML models include both shallow models and deep learning models. Shallow models mainly include Artificial Neural Network (ANN), Support Vector Machine (SVM), K-Nearest Neighbors (kNN), Naive Bayes Classification, Logistic Regression (LR), Decision Tree, and Clustering. The deep learning models mainly include Autoencoder, Restricted Boltzmann Machine (RBM), Deep Brief Network (DBN), Deep Neural Network (DNN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), and Generative Adversarial Network (GAN) (Liu & Lang, 2019).

In this paper, two shallow models and two deep learning models are chosen to make the predictions and to be compared.

Shallow Models

k-Nearest Neighbors (kNN) is one of the simplest ML algorithms. It operates as follows: First, identify K similar instances, where instances have similar attributes. Then, take the majority class among those K instances and give this class to the new instance. In the kNN regression, it predicts the new instance by taking the average of the values from these K neighbors. It is important to note that a smaller value of K can increase the model complexity and risk of overfitting (Chen, 2018).

Random Forest comprises numerous decision trees, where each individual tree is trained using a random subset of the dataset and at each split, a random subset of features is considered (Géron, 2019). This randomness introduces variability among the individual trees, which helps to reduce overfitting and improve overall prediction accuracy. Random Forest makes predictions by combining the average predictions from all trees in the forest (Kumar et al., 2024).

Deep Learning Models

Recurrent Neural Networks (RNN) are designed to handle sequential data, processing inputs of varying lengths, unlike traditional neural networks that require fixed-size inputs (Wang, 2021). RNN can be applied in many fields, including analyzing time series data. For example, RNN can be used to forecast the stock

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

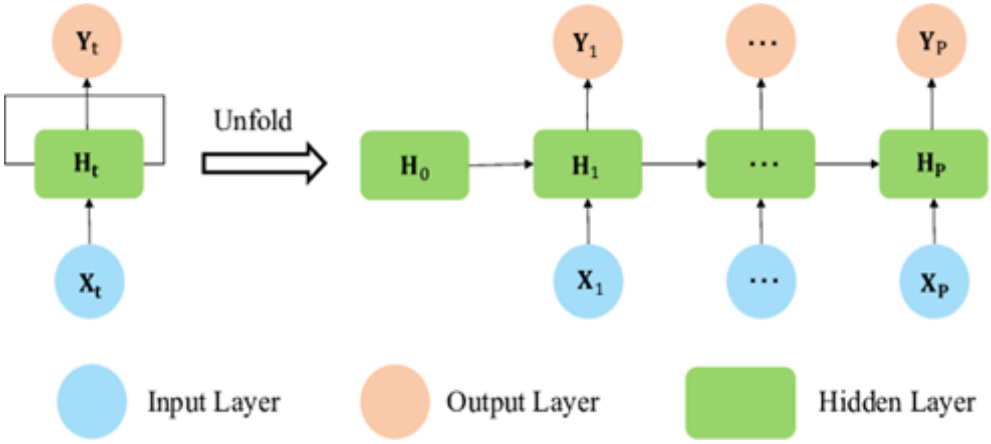
FALL 2025

price. It is also widely used in natural language processing tasks, including automatic speech-to-text translation.

In RNN, each recurrent neuron has an input X_t and its own previous time-step output, Y_{t-1} (Géron, 2019). Each recurrent neuron is connected to two sets of weights: one for inputs X_t and another for the output from the previous time step, Y_{t-1}. We can informally think about any recurrent neuron's output at time step t as some sort of memory, since it depends on all the previous inputs.

Figure 1

The Structure of RNN



Note. This figure is reprinted from "Getting started with AI: Building an RNN from scratch and practicing resilience. Medium." by A. Choudhry, 2023, *Medium*.

However, when all the information eventually passes through all of the neurons, exponential decay probably occurs. Thus, sometimes after the information was provided, the state of an RNN might potentially forget the very first items. Specialized cells for long-term memory have been developed and have been very successful. The most popular of these cells include the Long Short-Term Memory and the Gated Recurrent Unit.

Long Short-Term Memory (LSTM) cells were introduced in 1997 by Sepp Hochreiter and Ju¨rgen Schmidhuber (Géron, 2019). LSTM cells are designed to learn to recognize important inputs, therefore internal memory states are able to store them for long periods of time, later retrieving the information and using it. In the LSTM, the forget gate, input gate, and output gate work together to process the information, including short-term state H_t and long-term state C_t. The LSTM can learn what to store in the long-term state, what to throw away, and what to read from it. Specifically, as the long-term state C_{t-1} traverses the LTSM network, the forget gate helps drop some memories, while the input gate helps select and add new memories. The C_t is sent out without any further transformation. At each step, some memories are dropped, and some memories are added (Graves, 2018). In addition, the long-term information is copied and passed through the *tanh* function, and the result is filtered by the output gate, which produces the short-term state H_t. One advantage of LSTM is

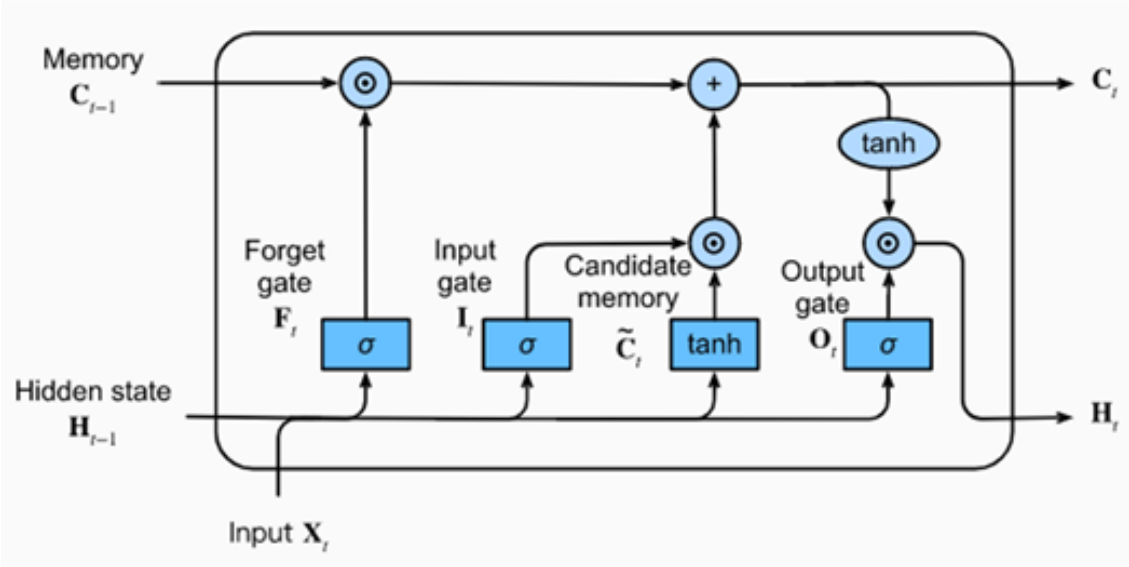
SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

that it can process raw data directly without the need for feature selection (Choudhary & Kesswani, 2020). The performance of LSTM has been proven to be better than the basic units of RNNs (Vinayakumar et al., 2017).

Figure 2
The Structure of LSTM



Note. This figure is reprinted from "Implementation of long short-term memory (LSTM). Medium." by Aaqilah, 2023, *Medium*.

Gated Recurrent Unit (GRU) cells are a simpler variant of the LSTM cell that was introduced by Kyunghyun Cho et al. in 2014 (Géron, 2019). Although it has a similar performance to LSTM, it is a bit less sophisticated. In GRU networks, both long-term state and short-term state merge into a single vector H_t, and there is only one gate controller Z_t controls both the forget gate and the input gate. In addition, there is no output gate. Instead, there is a new gate controller R_t that controls which part of the previous state will be shown to the main layer.

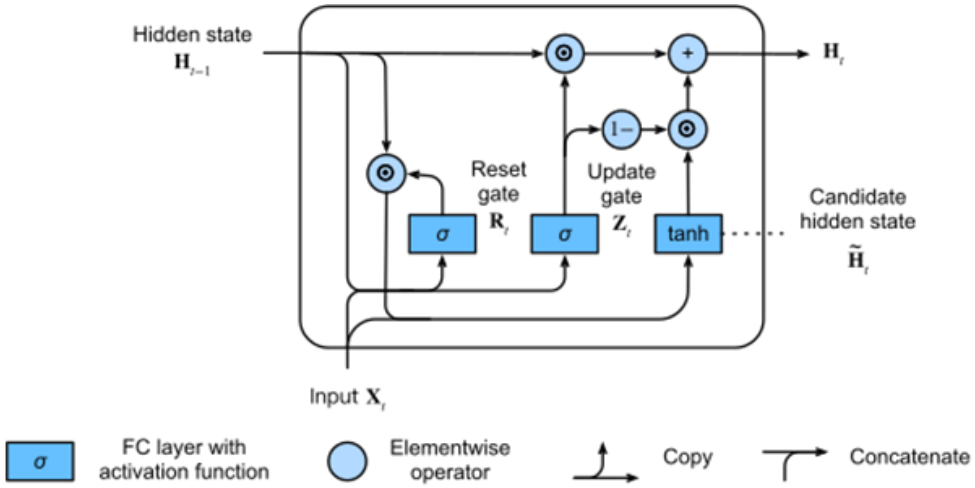
Generally, both LSTM and GRU cells can process a much longer sequence than basic RNNs.

Figure 3
The Structure of GRU

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025



Note. This figure is reprinted from "Gated recurrent units (GRU) — Dive into deep learning 1.0.3 documentation." by A. Zhang et al., 2014.

DATASETS

Datasets are critical to help examine different methods of intrusion detection. The commercial product network packet analysis dataset is not accessible because of privacy concerns. However, three publicly available benchmark datasets are widely used for N-IDS evaluation. They include KDD99, NSL-KDD, and UNSW-NB15.

Benchmark Datasets

1. KDD99

The KDD99 dataset was created by DARPA in 1999 by using recorded network traffic from a 1998 dataset, and is the most famous and widely used benchmark dataset (Liu & Lang, 2019; Choudhary & Kesswani, 2020). It comprises 41 features that are categorized into four groups: basic features, content features, host-based features, and time-based features. However, the dataset has several limitations. First, it is imbalanced, which can lead to biased classification more toward the majority classes. Moreover, it contains many duplicate records; hence, there is a need for data cleaning before the analysis. Lastly, the KDD99 data is relatively old; it may not represent the current network environments and attack patterns.

2. NSL-KDD

The NSL-KDD dataset was developed to address the various biases attached to KDD99 (Seiba et al., 2021). The NSL-KDD records have been carefully selected from the KDD99 dataset, and their classes are much more balanced, therefore reducing classification bias. Though the NSL-KDD dataset fixes some of the flaws in

SCHOLARLY REVIEW

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

the KDD99, it still lacks new samples of minority classes and is becoming outdated (Liang et al., 2020).

3. UNSW-NB15

The UNSW-NB15 dataset was created by researchers from the University of New South Wales and was published in 2015 (Liu & Lang, 2019; Choudhary & Kesswani, 2020). It is based on network traffic captured from three virtual servers. Unlike KDD99 and NSL-KDD, UNSW-NB15 contains a broader range of attack types, including normal traffic and nine attack types. The features can be categorized into six groups, including basic features, flow features, content features, additional features, and labeled features. As one of the most recent datasets in intrusion detection, UNSW-NB15 is crucial for evaluating ML methods against new and evolving types of attacks.

Dependent Variable - Attack Types

1. KDD99 / NSL-KDD

In the KDD99 and NSL-KDD, there are four types of attack: Probe, User to Root (U2R), Remote to Local (R2L), and Denial of Service (DoS) (Kalita,2018).

(1) **Probe attacks** are attempts to gather information on target external network sources, which can include port sweeps or IP sweeps. Probe attacks allow attackers to spy, access, or gather network information (Abdullahi et al., 2022).

(2) **User to Root (U2R)** attacks intend to gain access to systems using any normal account (Abdullahi et al., 2022). U2R attacks may manipulate, spy on, or interrupt the normal behavior of systems.

(3) **Remote to Local (R2L)** attacks occur when users send packets to systems to which they do not have legal access, such as through xclock or guest passwords (Abdullahi et al., 2022).

(4) **Denial of Service (DoS)** attacks are the most common attacks due to their easy implementation (Abdullahi et al., 2022). They can be carried out by disturbing the targets' network traffic, which includes DDoS and UDP storms. DoS attacks make the system resources too busy to serve other genuine requests in the networks.

2. UNSW-NB15

UNSW-NB15 incorporates new and complex attack patterns. There are nine types of attacks, including Fuzzer, Analysis, Backdoor, DoS, Exploit, Generic, Reconnaissance, Shellcode, and Worm (UNSW Sydney, 2015).

(1) **Fuzzer attacks** are related to the transmission of ill-formed or random data to the target systems to identify any security vulnerabilities in them. The attackers automatically generate a large number of random inputs in order to recognize flaws in the systems.

(2) **Analysis attacks** use many different techniques to draw information from the networks or systems in hopes of locating possible vulnerabilities. These activities include port scanning, vulnerability scanning, and other reconnaissance activities.

(3) **Backdoor attacks** involve downloading hidden entrances into the systems so that attackers can gain unauthorized access at a later time. It is possible to install such backdoors through malware or by exploiting vulnerabilities in the systems.

(4) **DoS attacks** are identical to the DoS attacks in KDD99/NSL-KDD, all with a common purpose: making the network or system resources unavailable to their intended users. This is done by overwhelming the systems with a large amount of traffic or other forms of attack to make the services unavailable.

(5) **Exploit attacks** are usually based on exploiting flaws or vulnerabilities in software, hardware, or network protocols to execute unauthorized commands or gain unauthorized access to the systems. This may lead to system compromise, leakage of information, or elevation of privileges.

(6) **Generic attacks** are independent cryptographic attacks that include the exploitation of weaknesses in some encryption algorithms or breaking cryptographic protections brute-force to compromise the security of the systems.

(7) **Reconnaissance attacks** are executed to gather information about the networks or systems (Burgio, 2019). Probing or scanning is utilized in recognizing targets or vulnerabilities within a network or system. These attacks consist of IP sweeps, port sweeps, and other types of scanning.

(8) **Shellcode attacks** occur when attackers inject and execute shell code, a small piece of code used in exploitation as a payload, for the sole purpose of taking control over the targeted systems (Foster & Price, 2005). This may result in remote code execution or system compromise.

(9) **Worm attacks** are a form of self-replicating malware that propagates on its own through the networks, exploiting systems that have vulnerabilities, and does not require any user interaction. These attacks cause immense harm in terms of consuming network resources, erasing or corrupting files, and creating further infections.

Independent Variable - Data Features

In terms of comparison, the KDD99 and NSL-KDD datasets include 41 features, and UNSW-NB15 includes 49 features, while 42 features are used in the model. The features in KDD99 and NSL-KDD are categorized into four groups. The first group (basic features) includes necessary information, such as duration, protocol, and service. The second group (content features) includes content-related features, such as login activities, which are important to detect U2R and R2L attacks (Tavallaee et al., 2009). The third group (time-based features) includes time-based features, such as the number of connections that are related to the same host in the past two seconds. The fourth group (host-based features) includes host-based features, such as the count of connections with the same destination host (Choudhary & Kesswani, 2020).

In the UNSW-NB15 dataset, the features are categorized into five groups. Besides basic features, content features, and time-based features as the other two datasets, UNSW-NB15 includes flow features, which

identify the protocols between the hosts, such as TCP or UDP. In addition, UNSW-NB15 includes additional features, such as the number of connections that contain the same service and destination address in 100 connections according to the last time (Choudhary & Kesswani, 2020).

Several features are common across the three datasets. They all include connection duration, protocol type, service used at the destination, and the number of transmitted data bytes between the source and destination (Petersen, 2015). Additionally, they all feature connection flags with similar characteristics. Notably, KDD99 and NSL-KDD datasets include more host-based features related to host connection information. Conversely, the UNSW-NB15 dataset consists of a higher proportion of time-based and size-based features.

PROCEDURE

Experimental Setup

The experiments were conducted on a server configured with an NVIDIA RTX 4060 GPU, an Intel Core i5-12600K CPU, and 16GB of RAM. The software environment included Python 3.12.4, with key libraries such as TensorFlow 2.x, Keras, Scikit-learn, and Pandas. These libraries were chosen for their capability to handle large datasets and support complex ML models, particularly deep learning architectures.

To ensure a fair comparison across all models, data preprocessing was applied uniformly. Extreme values of features were pruned to minimize distribution skewness (Kirstein, 2022). Specifically, if a feature's maximum value exceeded ten times its median, it was pruned to the 95th percentile. Additionally, the logarithmic transformation was applied to numeric features with more than 50 unique values to approximate a normal distribution.

In both the KDD99 and NSL-KDD datasets, the multiclass target variable includes 39 different types of attacks. Before model development, these attacks were grouped into four general categories under the "attack map" variable: DoS (Denial of Sērvice) attacks, Probe attacks, R2L (Remote to Local) attacks, and U2R (User to Root) attacks. For binary classification, a dummy target variable was created, with 1 representing "attack" and 0 representing "normal".

The table below illustrates the categorization of the 39 attack types into these four general attack categories for the KDD99 and NSL-KDD datasets (Choudhary & Kesswani, 2020).

Table 1
KDD99/NSL-KDD Attack Categories and Types

Attack Category	Attack Type
DoS	apache2, back, land, neptune, mailbomb, pod, processtable, smurf, teardrop, udpstorm, worm
Probe	ipsweep, mscan, nmap, portsweep, saint, satan

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

U2R	buffer overflow, loadmodule, perl, ps, rootkit, sqlattack, xterm
R2L	ftp_write, guess_passwd, http_tunnel, imap, multihop, named, phf, sendmail, snmpgetattack, snmpguess, spy, warezclient, warezmaster, xclock, xsnoop

In the UNSW-NB15 dataset, the "attack cat" variable is a multiclass target variable, including normal data and nine types of attacks: Reconnaissance, Backdoor, DoS, Exploits, Analysis, Fuzzers, Worms, Shellcode, and Generic (UNSW Sydney, 2015). The "label" variable is a dummy target variable, with 1 representing "attack" and 0 representing "normal."

To train and test the models, the datasets were divided into training and testing subsets using an 80/20 ratio: 80% of the data was used for training, and the remaining 20% was used for testing. Tables 2-4 illustrate the data distribution for each attack type within each dataset. Table 2 shows that the KDD99 dataset exhibits a significant imbalance, with DoS attacks comprising 79% of the data and normal data accounting for 20%. The remaining attack types, Probe, U2R, and R2L, collectively make up only 1% of the dataset, indicating a substantial class imbalance.

Table 3 shows that while the NSL-KDD dataset contains fewer observations than the KDD99 dataset, it is more balanced. In NSL-KDD, normal data accounts for 53% of the total, DoS attack makes up 36%, and the combined total of Probe, U2R, and R2L attacks accounts for 10%.

Table 2

KDD99 Data Frequency

	Attack Type				
	Normal	DoS	Probe	U2R	R2L
Train (80%)	77829	313166	3286	34	901
Test (20%)	19457	78292	821	9	225
Total	97286	391458	4107	43	1126

Table 3

NSL-KDD Data Frequency

	Attack Type				
	Normal	DoS	Probe	U2R	R2L
Train (80%)	53881	34741	9325	34	796
Test (20%)	13470	9186	2331	9	199
Total	67351	45927	11656	43	995

Table 4 shows that the UNSW-NB15 dataset includes nine types of attacks (UNSW Sydney, 2015). Compared to KDD99 and NSL-KDD, the data in UNSW-NB15 are more evenly distributed across these nine attack types.

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

Table 4
UNSW-NB15 Data Frequency

	Attack Type				
	Normal	Generic	Exploits	Fuzzers	DoS
Train (80%)	29600	15097	8905	4850	3271
Test (20%)	7400	3774	2227	1212	818
Total	37000	18871	11132	6062	4089
	Recon.	Analysis	Backdoor	Shellcode	Worms
Train (80%)	2797	542	466	302	35
Test (20%)	699	135	117	76	9
Total	3496	677	583	378	44

Model Training

The research focused on evaluating the performance of both binary and multiclass classification models across three commonly used intrusion detection datasets: KDD99, NSL-KDD, and UNSW-NB15. The ML algorithms employed in this study included kNN, Random Forest, GRU, and LSTM networks. Each model was trained and evaluated on both binary and multiclass targets, allowing for a comprehensive analysis of their effectiveness in different classification scenarios.

The kNN and Random Forest models represent traditional machine learning approaches, while GRU and LSTM are deep learning models, particularly suited for sequential data like network traffic. Training these models involved tuning hyperparameters such as the number of neighbors for kNN, the number of trees for Random Forest, and the number of units and layers for GRU and LSTM models. The deep learning models were trained using the Adam optimizer.

Experimentation

The experiments aimed to evaluate both binary and multiclass intrusion classifications. For each dataset, four algorithms were implemented: two shallow models (kNN and Random Forest) and two deep learning models (GRU and LSTM).

The hyperparameters for each model were as follows:

- k-Nearest Neighbors (kNN): k=5, using the 5 nearest neighbors for model predictions.
- Random Forest: n estimators=100, employing 100 decision trees in the ensemble.
- LSTM/GRU: The input layer consisted of 20 neurons. All experiments were conducted for 200 epochs with a batch size of 2000. These hyperparameters were selected through systematic tuning to optimize performance. The ADAM optimizer was used, with sparse categorical crossentropy for

SCHOLARLY REVIEW

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

the binary classification model and categorical crossentropy for the multiclass classification model. The softmax function was applied in the dense layers. Additionally, training time was measured in seconds for each model.

IDS Performance Metrics

The performance of each model was assessed using the following key metrics:

Accuracy refers to how accurately the IDS is in detecting normal activity and attacks (Saylor Academy, 2024). It is the percentage of correctly classified instances against total instances (Gyei-Kark et al., 2023).

$$A = \frac{TP+TN}{TP+TN+FP+FN}$$

Precision (P) is the proportion of correctly identified positive instances against all those instances predicted as positive (FasterCapital, 2024). This tells the reliability of attack detection.

$$P = \frac{TP}{TP+FP}$$

Recall (R) refers to the proportion of true positive instances against all actual positive cases (Jena et al., 2024). It reveals a model's effectiveness in recognizing attacks.

$$R = \frac{TP}{TP+FN}$$

F-measure (F) is the harmonic average of precision and recall (Cavallaro et al., 2023). This gives a balanced measure in cases where both precision and recall are relevant.

$$F = \frac{2*P*R}{P+R}$$

False negative rate (FNR) is the proportion of false negatives against all actual positive cases (Google for Developers, 2024). In other words, it is the rate at which an attack will go undetected.

$$FNR = \frac{FN}{TP+FN}$$

False positive rate (FPR) is the proportion of false positives out of total predicted positives, which represents the tendency of the model to misclassify normal instances as attacks (Saripuddin et al., 2021).

$$FPR = \frac{FP}{TP+FP}$$

In these formulas, TP refers to True Positives, FP refers to False Positives, TN refers to True Negatives, and FN refers to False Negatives (Wen et al., 2022). The positive instance is an attack, and the negative instance is normal.

Training time is the amount of time taken to train the model (Panigrahi et al., 2021). This mainly

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

serves as an indicator of how computationally intensive the algorithm is.

Prediction time is defined as the time taken to make predictions on the test data, important for assessing the model's applicability in real-time systems (Hassanien et al., 2014).

Total time is defined as the sum of training and prediction times, providing an overall measure of the model's efficiency (Lu et al., 2024).

RESULTS

From Tables 5-10, we can evaluate the performance of each model on both binary and multiclass classification tasks across different datasets.

For binary classification on the KDD99 dataset, Table 5 shows that the Random Forest model is the most optimal, achieving 99.98% accuracy with a 0.03% false positive rate (FPR) and a 0.02% false negative rate (FNR). Its total running time is only 9.1 seconds. Both GRU and LSTM models also achieve high accuracy rates of 99.96%, but their total running times are significantly longer: 5923.3 seconds for GRU and 375.6 seconds for LSTM. The extended running times for deep learning models are due to their complexity and the impact of hyperparameters, such as the number of epochs. In this analysis, the number of epochs was set to 200, leading to longer training times as the models processed the dataset through multiple iterations.

Table 5

KDD99 Binary Classification

	Accuracy	Recall	Precision	F1-Score	FRR	FNR	Time to Train(s)	Time to Predict(s)	Total Time(s)
kNN	99.94%	99.94%	99.94%	99.94%	0.15%	0.04%	0.0	119.3	119.3
Random Forest	99.98%	99.98%	99.98%	99.98%	0.03%	0.02%	8.9	0.2	9.1
GRU	99.96%	99.96%	99.96%	99.96%	0.08%	0.03%	358.3	5564.9	5923.3
LSTM	99.96%	99.96%	99.96%	99.96%	0.09%	0.03%	296.0	79.5	375.6

For multiclass classification on the KDD99 dataset, Table 6 shows that Random Forest is the most optimal model, achieving 99.98% accuracy with a 0.02% FPR and a 0.02% FNR. Its total running time is just 10 seconds. Both GRU and LSTM models also achieve a high accuracy of 99.98%, but their running times are considerably longer, at 500.4 seconds for GRU and 666.7 seconds for LSTM.

Table 6

KDD99 Multiclass Classification

	Accuracy	Recall	Precision	F1-Score	FRR	FNR	Time to Train(s)	Time to Predict(s)	Total Time(s)

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

kNN	99.96%	99.96%	99.96%	99.96%	0.04%	0.04%	0.4	110.8	111.2
Random Forest	99.98%	99.98%	99.98%	99.98%	0.02%	0.02%	9.7	0.3	10.0
GRU	99.98%	99.98%	99.98%	99.98%	0.02%	0.02%	331.9	168.6	500.4
LSTM	99.98%	99.98%	99.98%	99.98%	0.02%	0.02%	320.4	346.3	666.7

For binary classification on the NSL-KDD dataset, Table 7 shows that Random Forest is the most optimal model, achieving 99.97% accuracy with a 0.02% FPR and a 0.03% FNR. Its total running time is only 2.3 seconds. Both GRU and LSTM models also achieve relatively high accuracy, but with significantly longer running times of 140.1 seconds for GRU and 115.5 seconds for LSTM. Among the four models tested, kNN achieved the lowest accuracy at 99.87%.

Table 7

NSL-KDD Binary Classification

	Accuracy	Recall	Precision	F1-Score	FRR	FNR	Time to Train(s)	Time to Predict(s)	Total Time(s)
kNN	99.87%	99.87%	99.87%	99.87%	0.20%	0.06%	0.0	7.5	7.5
Random Forest	99.97%	99.97%	99.97%	99.97%	0.02%	0.03%	2.2	0.1	2.3
GRU	99.93%	99.93%	99.93%	99.93%	0.07%	0.06%	92.6	47.6	140.1
LSTM	99.91%	99.91%	99.91%	99.91%	0.12%	0.05%	92.6	22.9	115.5

For multiclass classification on the NSL-KDD dataset, Table 8 shows that Random Forest is the most optimal model, achieving 99.92% accuracy with a 0.08% FPR and a 0.08% FNR. Its total running time is only 2.4 seconds. Both GRU and LSTM models also achieve high accuracy, at 99.91%, but have much longer running times of 154.3 seconds for GRU and 102.1 seconds for LSTM. Among the four models tested, kNN achieved the lowest accuracy at 99.86%.

For binary classification on the UNSW-NB15 dataset, Table 9 shows that Random Forest is the most optimal model, achieving 97.68% accuracy with a 1.95% FPR and a 2.62% FNR. Its total running time is only 3 seconds. LSTM achieves the second highest accuracy at 96.59%, with a longer running time of 144.9 seconds. GRU follows with a third highest accuracy of 96.37% and a running time of 69.6 seconds. Among the four models, kNN achieved the lowest accuracy at 95.11%.

Table 8

NSL-KDD Multiclass Classification

	Accuracy	Recall	Precision	F1-Score	FRR	FNR	Time to Train(s)	Time to Predict(s)	Total Time(s)
kNN	99.86%	99.86%	99.87%	99.86%	0.14%	0.14%	0.0	7.4	7.4
Random Forest	99.92%	99.92%	99.93%	99.92%	0.08%	0.08%	2.3	0.1	2.4
GRU	99.91%	99.91%	99.91%	99.91%	0.09%	0.09%	94.5	59.8	154.3

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

LSTM	99.91%	99.91%	99.91%	99.91%	0.09%	0.09%	92.8	9.3	102.1

Table 9

UNSW-NB15 Binary Classification

	Accuracy	Recall	Precision	F1-Score	FRR	FNR	Time to Train(s)	Time to Predict(s)	Total Time(s)
kNN	95.11%	95.11%	95.19%	95.12%	3.43%	6.08%	0.0	3.3	3.3
Random Forest	97.68%	97.68%	97.69%	97.68%	1.95%	2.62%	2.9	0.1	3.0
GRU	96.37%	96.37%	96.37%	96.37%	3.16%	4.01%	60.5	9.1	69.6
LSTM	96.59%	96.59%	96.59%	96.59%	2.78%	3.93%	64.8	80.1	144.9

For multiclass classification on the UNSW-NB15 dataset, Table 10 shows that Random Forest is the most optimal model, achieving 87.19% accuracy with a 12.81% FPR and FNR. Its total running time is only 3.8 seconds. LSTM achieves the second highest accuracy at 86.48%, with a running time of 66.2 seconds. GRU follows with an accuracy of 86.47% and a running time of 90.6 seconds. Among the four models, kNN achieved the lowest accuracy at 84.36%.

Table 10

UNSW-NB15 Multiclass Classification

	Accuracy	Recall	Precision	F1-Score	FRR	FNR	Time to Train(s)	Time to Predict(s)	Total Time(s)
kNN	84.36%	84.36%	83.72%	83.94%	15.64%	15.64%	0.1	3.5	3.6
Random Forest	87.19%	87.19%	86.76%	86.62%	12.81%	12.81%	3.7	0.1	3.8
GRU	86.47%	86.47%	86.47%	86.47%	13.53%	13.53%	57.0	33.6	90.6
LSTM	86.48%	86.48%	86.48%	86.48%	13.52%	13.52%	54.1	12.1	66.2

DISCUSSION

Key Findings

In this research, we compared four machine learning algorithms—two shallow models and two deep learning models—for detecting cyberattacks. The models were developed for both binary and multiclass classification tasks.

For the KDD99 dataset, all four algorithms exhibited slightly better performance with the multiclass models compared to the binary models in terms of accuracy. Conversely, for the NSL-KDD and UNSW-NB15 datasets, all four algorithms showed slightly better performance with the binary models than with the multiclass models regarding accuracy.

SCHOLARLY REVIEW

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

In all the experiments, Random Forest demonstrated the best performance in terms of accuracy and running time. It also achieved lower FPR and FNR compared to the other algorithms. For the KDD99 dataset, the Random Forest model achieved 99.98% accuracy for both binary and multiclass classifications, surpassing the 99.65% accuracy reported by Saranya et al. (2020) for the same dataset using Random Forest (Saranya et al., 2020).

While the deep learning models LSTM and GRU also achieved very high accuracy, their training and prediction times were significantly longer compared to Random Forest.

For the KDD99 dataset, both GRU and LSTM achieved an accuracy of 99.96% for the binary classification model and 99.98% for the multiclass model. Laghrissi et al. (2021) also utilized LSTM for attack detection on the KDD99 dataset, incorporating Principal Component Analysis (PCA) and Mutual Information (MI) for dimensionality reduction and feature selection (Laghrissi et al., 2021). Their research found that LSTM with PCA, using 2 components, achieved the best model performance with testing accuracies of 99.44% for binary classification and 99.36% for multi-class classification. In comparison, our models demonstrated slightly better performance on the KDD99 dataset.

For the NSL-KDD dataset, both GRU and LSTM models achieved accuracy above 99.9% for both binary and multiclass classifications. For the UNSW-NB15 dataset, GRU and LSTM models achieved accuracies above 96% for the binary classification and above 86% for the multiclass classification. Kasongo (2023) explored various RNN models, including Simple RNN, LSTM, and GRU, for network intrusion detection using the NSL-KDD and UNSW-NB15 datasets, and also employed XGBoost for feature selection (Kasongo, 2023). Their results showed that for binary classification, XGBoost-LSTM achieved the highest accuracy with 88.13% on NSL-KDD, while XGBoost-Simple-RNN achieved 87.07% on UNSW-NB15. For multiclass classification, XGBoost-LSTM achieved 86.93% on NSL-KDD, and XGBoost-GRU achieved 78.4%. In comparison, our GRU and LSTM models significantly outperformed Kasongo's results on both the NSL-KDD and UNSW-NB15 datasets for both binary and multiclass classifications.

Deep learning models are able to capture highly complex patterns, but do not significantly outperform Random Forest in terms of accuracy, which raises questions about the trade-offs between model complexity and real-world performance. In addition, the deep learning models require longer running times because of the model complexity.

Compared with the KDD99 and NSL-KDD datasets, the model performance on the UNSW-NB15 dataset was notably poorer, characterized by reduced accuracy and higher FPR and FNR across all algorithms, particularly for the multiclass classification. While kNN, Random Forest, LSTM, and GRU all achieved accuracy above 99% for both binary and multiclass classifications on KDD99 and NSL-KDD, their accuracy on UNSW-NB15 dropped below 90% for the multiclass experiment. The results are consistent with Vinayakumar et al.'s (2017) work, which also found that accuracy with the UNSW-NB15 dataset decreased by more than 10%

compared to the KDD99 dataset.

The diminished performance on the UNSW-NB15 dataset may be attributed to the presence of more modern and complex intrusion patterns compared to KDD99 and NSL-KDD (Tian et al., 2020). The UNSW-NB15 dataset includes a broader range of contemporary attacks that are more sophisticated and constantly evolving, making them harder to detect. This complexity challenges the models in distinguishing between various types of intrusions, particularly affecting simpler algorithms like kNN. Although deep learning models such as GRU and LSTM outperformed kNN, they still exhibited high rates of false positives and false negatives, suggesting that even advanced models struggle with the UNSW-NB15 dataset. These results indicate the need for future research to focus on developing and refining models that are better tailored to the complexities of modern intrusion patterns.

Future Work

The current study explored both shallow and deep learning models to assess the performance of various machine learning algorithms for IDS using three different benchmark datasets. However, there are several directions for future research that could enhance the efficiency and applicability of these models.

First, one interesting point of future work is to investigate the impact of feature selection. Since model performance would be driven by the most relevant features, feature selection would decrease the dimensionality of datasets and probably would improve both model accuracy and efficiency. It also minimizes the effects of irrelevant or redundant data, which could lead to more robust and interpretable models.

Second, a machine learning model performs based on its choice of hyperparameters. Future work could test different hyperparameters and check their impact on the model performance, including both accuracy and running time.

Third, additional machine learning models can be tested and compared. This study focused on kNN, Random Forest, GRU, and LSTM. However, there exist many other machine learning algorithms that could be explored for IDS. Future research could involve evaluating a broader range of algorithms to assess their performance across different classification tasks.

Lastly, future research should consider testing additional datasets. This study evaluated three benchmark datasets: KDD99, NSL-KDD, and UNSW-NB15. It would be beneficial to improve IDS by developing models using datasets with real-time traffic, and those with newly identified intrusions and attack patterns. Such efforts could further enhance the accuracy and efficiency of cybersecurity detection.

CONCLUSION

With the advancement of distributed computing systems, cyberattacks are continuously evolving.

Additionally, users can readily exploit advanced attack toolkits available on the internet (Tian et al., 2020). Consequently, cybersecurity research is crucial for safeguarding organizations and individuals from network breaches. An IDS is a key cybersecurity tool designed to monitor software and hardware activities, aiming to prevent unauthorized or illegal access to networks.

Although many studies have examined and compared different machine learning models for IDS using one or two benchmark datasets, this paper offers a comprehensive comparison of selected shallow models and deep learning models across three benchmark datasets: KDD99, NSL-KDD, and UNSW-NB15. Additionally, the paper evaluates the models for both binary and multiclass classification tasks.

In this paper, we evaluated and compared two shallow learning algorithms and two deep learning algorithms for IDS using three publicly available datasets: KDD99, NSL-KDD, and UNSW-NB15. We developed both binary and multiclass models with four machine learning algorithms: kNN, Random Forest, GRU, and LSTM. The performance of these models was assessed using various metrics, including accuracy, precision, recall, F-measure, false positive rate, false negative rate, training times, and prediction times.

The experiments demonstrated that both shallow learning methods (kNN and Random Forest) and deep learning methods (LSTM and GRU) exhibit strong performance in intrusion detection. Among these, Random Forest consistently outperformed the other models across all datasets, offering high accuracy along with shorter training and prediction times. Deep learning models GRU and LSTM also delivered excellent performance, particularly in handling the sequential nature of network traffic data. These models effectively captured long-term dependencies, which are crucial for detecting sophisticated and evolving cyber threats.

However, GRU and LSTM models were more complex and required higher computational resources, resulting in increased training and prediction times. This highlights a key trade-off between model accuracy and computational efficiency, which is crucial when deploying IDS in real-world scenarios where timely detection and response are essential.

One notable observation was the variation in model performance across the three datasets. While high accuracy was achieved with the KDD99 and NSL-KDD datasets, the same was not true for the UNSW-NB15 dataset. The results indicated that achieving strong performance on UNSW-NB15, particularly for multiclass models, is challenging. This underscores the need for further improvements in IDS models to address these complex cyber threats. This finding aligns with Tian et al. (2020), who observed that evolving attack patterns are better represented in more recent datasets like UNSW-NB15 (Tian et al., 2020).

While machine learning offers valuable tools for enhancing IDS, selecting the appropriate model and dataset is crucial for optimal detection performance. Future research should concentrate on developing advanced models capable of addressing the complexity and diversity of modern attacks. Emphasis should be placed on utilizing real-time data and exploring more robust deep learning architectures to ensure effective intrusion detection in the ever-evolving cybersecurity landscape.

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

ACKNOWLEDGEMENTS

I would like to thank Dr. Suleyman Uludag for his guidance on cybersecurity concepts and for inspiring me to explore the use of machine learning in addressing security challenges. I am deeply grateful for his valuable feedback and generous support throughout this project.

REFERENCES

Aaqilah. (2023). *Implementation of long short-term memory (LSTM)*. Medium. https://medium.com/@pennQuin/implementation-of-long-short-term-memory-lstm-81e35fa5ca54

Abdullahi, M., Baashar, Y., Alhussian, H., Alwadain, A., Ariz, N., Capretz, L., & Abdulkadir, S. (2022). Detecting cybersecurity attacks in Internet of Things using artificial intelligence methods: A systematic literature review. *Electronics, 11*(2), 198. https://doi.org/10.3390/electronics11020198

Burgio, D. A. (2019). Reduction of false positives in intrusion detection based on extreme learning machine with situation awareness. https://nsuworks.nova.edu/gscis_etd/1093/

Cavallaro, C., Cutello, V., Pavone, M., & Zito, F. (2023). Discovering anomalies in big data: A review focused on the application of metaheuristics and machine learning techniques. *Frontiers in Big Data, 6*. https://doi.org/10.3389/fdata.2023.1158971

Chen, S. (2018). K-Nearest Neighbor Algorithm Optimization in Text Categorization. *IOP Conference Series: Earth and Environmental Science*, *108*, 052074. https://doi.org/10.1088/1755-1315/108/5/052074

Choudhary, S., & Kesswani, K. (2020). Analysis of KDD-CUP'99, NSL-KDD and UNSW-NB15 datasets using deep learning in IoT. *Procedia Computer Science, 167*, 1561–1573. https://doi.org/10.1016/j.procs.2020.03.386

Choudhry, A. (2023). *Getting started with AI: Building an RNN from scratch and practicing resilience*. Medium. https://medium.com/@adachoudhry26/getting-started-with-ai-building-an-rnn-from-scratch-and-practicing-resilience-ba3c10be6a22

FasterCapital. (2024). F1 score. https://fastercapital.com/keyword/4-f1-score.html

Fleck, A. (2024, February 22). *Infographic: Cybercrime Expected To Skyrocket in Coming Years*. Statista Daily Data; Statista. https://www.statista.com/chart/28878/

Foster, J. C., & Price, M. (2005). *Sockets, shellcode, porting, & coding: Reverse engineering exploits and tool coding for security professionals*. Syngress.

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. "O'Reilly Media, Inc."

Google for Developers. (2024). Classification: Accuracy, recall, precision, and related metrics. https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall

Graves, A. (2018). *Supervised sequence labelling with recurrent neural networks*. SpringerLink.

Gyei-Kark, P., Jana, D. K., Panja, P., & Abd Wahab, M. H. (2023). *Engineering mathematics and computing*. Springer.

Hassanien, A. E., Kim, T., Kacprzyk, J., & Awad, A. I. (Eds.). (2014). *Bio-inspiring cyber security and cloud services: Trends and innovations*. Springer.

Jena, S. K., Kumar, B., Mohanty, B., Singhal, A., & Barik, R. C. (2024). An advanced blockchain-based hyperledger fabric solution for tracing fraudulent claims in the healthcare industry. *Decision Analytics Journal, 10*, 100411. https://doi.org/10.1016/j.dajour.2023.100411

Kalita, P. (2018). Developing network intrusion detection systems using data cube and association rule. http://mzuir.inflibnet.ac.in:8080/jspui/handle/123456789/777

Kasongo, S. (2023). A deep learning technique for intrusion detection system using a recurrent neural networks based framework. *Computer Communications, 199*, 113–125. https://doi.org/10.1016/j.comcom.2022.12.010

Khamis, R. (2020). *Evaluating Adversarial Learning on Different Types of Deep Learning-based Intrusion Detection Systems using min-max optimization*. https://doi.org/10.22215/etd/2020-14257

Kirstein, C. (2022). UNSW_NB15 modelling - 97.7%. https://www.kaggle.com/code/carlkirstein/unsw-nb15-modelling-97-7/notebook

Kumar, S., Sivakumar, P., Chaturvedi, A., Ismail Bin Musirin, & Dulhare, U. N. (2024, August 21). *Advancements in Metaverse Security: Phishing Website Detection Through Optimal Feature Selection and Random Forest Classifier*. https://doi.org/10.4018/979-8-3693-3824-7.ch014

Laghrissi, F., Douzi, D., Douzi, K., & Hssina, B. (2021). Intrusion detection systems using long short-term memory (LSTM). *Journal of Big Data, 8*(65). https://doi.org/10.1186/s40537-021-00455-3

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

Liang, C., Shanmugam, B., Azam, S., & Idris, N. B. (2020). Intrusion detection system for the Internet of Things based on blockchain and multi-agent systems. *Electronics, 9*(7). https://doi.org/10.3390/electronics9071140

Liu, H., & Lang, B. (2019). Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey. *Applied Sciences*, *9*(20), 4396. https://doi.org/10.3390/app9204396

Lu, X., Guo, R., Huang, H., & Wang, S. (2024). Potential-based dynamic parking navigation for autonomous vehicles: Near-priority vs. distant-priority. *Transport Policy*. https://doi.org/10.1016/j.tranpol.2024.01.001

Panigrahi, G. R., Barpanda, N. K., Biswal, R., & Samantaray, D. (2021). A machine automated big data modular framework for finding network security vulnerabilities. In *Proceedings of the 2021 IEEE 11th International Conference on Computational Intelligence and Communication Networks (CICN)*. IEEE.

Petersen, R. (2015). Data mining for network intrusion detection: A comparison of data mining algorithms and an analysis of relevant features for detecting cyber-attacks. https://www.diva-portal.org/smash/get/diva2:939697/FULLTEXT01.pdf

Saranya, T., Sridevi, S., Deisy, C., Chung, T., & Khan, M. (2020). Performance analysis of machine learning algorithms in intrusion detection system: A review. *Procedia Computer Science, 171*, 1251–1260. https://doi.org/10.1016/j.procs.2020.04.134

Saripuddin, M., Suliman, A., Syarmila, S., & Jørgensen, B. N. (2021). Random undersampling on imbalance time series data for anomaly detection. In *Proceedings of the 2021 4th International Conference on Machine Learning and Machine Intelligence*.

Saylor Academy. (2024). *Intrusion detection systems*. https://learn.saylor.org/mod/book/tool/print/index.php?chapterid=&id=29755

Seiba, A., Abdul-Salaam, G., Missah, Y., & Anisi, M. H. (2021). Hybrid network intrusion detection systems: A systematic review. *Scientific and Practical Cyber Security Journal (SPCSJ), 7*(4), 1–35.

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. (2009). A detailed analysis of the KDD Cup 99 data set. In *Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*. IEEE.

Tian, Q., Han, D., Li, K.-C., & Castiglione, A. (2020). An intrusion detection approach based on improved deep belief network. *Applied Intelligence, 50*(3), 3162–3178. https://doi.org/10.1007/s10489-020-01737-3
UNSW Sydney. (2015). *The UNSW-NB15 dataset*. https://research.unsw.edu.au/projects/unsw-nb15-dataset

SCHOLARLY REVIEW JOURNAL

E-ISSN: 2996-8380
Published by Leadership & Innovation Lab

FALL 2025

Vinayakumar, R., Soman, K., & Poornachandran, P. (2017). Evaluation of recurrent neural network and its variants for intrusion detection system (IDS). *International Journal of Information System Modeling and Design, 8*(3). https://doi.org/10.4018/IJISMD.2017070101

Wang, Z. (2021). Deep Learning in Medical Ultrasound Image Segmentation: a Review. *ArXiv:2002.07703 [Cs, Eess, Stat]*. https://arxiv.org/abs/2002.07703

Wen, C., Lu, M., Bi, Y., Zhang, S., Xue, B., Zhang, M., Zhou, Q., & Wu, W. (2022). An object-based genetic programming approach for cropland field extraction. *Remote Sensing, 14*(5), 1275. https://doi.org/10.3390/rs14051275

What Is a Cyberattack? | Microsoft Security. www.microsoft.com. https://www.microsoft.com/en-gb/security/business/security-101/what-is-a-cyberattack

Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2014). 10.2. Gated recurrent units (GRU) — Dive into deep learning 1.0.3 documentation. https://d2l.ai/chapter_recurrent-modern/gru.html