Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Babylon
College of Science for Women
Department of Computer Science

Improvement the Intrusion Detection System Based on Hybrid Deep Learning Model

A Thesis

Submitted to the Council of College of Science for Women, University of Babylon in a Partial Fulfilment of the Requirements for the Degree of Master in Science\ Computer Sciences

Submitted By

Duaa Wahab Raheem Al-Safaar

Supervised By

Asst. prof. Dr. Wathiq Laftah Abd-Ali Al-Yaseen

January 2023 A. D. **1444 A. H.**

بسم الله الرحمن الرحيم

يرفع الله الذين آمنوا منكم والذين أوتوا العلم
درجات والله بما تعملون خبير

صدق الله العلي العظيم

سورة المجادلة الاية 11

Supervisors Certification

We certify that this thesis entitled **"Improvement the Intrusion Detection System based on Hybrid Deep Learning Model"** is done by **(Duaa Wahab Raheem Al-Safaar)** under our supervision.

Signature:
Name: Asst. prof. Dr. Wathiq Laftah Abd-Ali Al-Yaseen
Date: / / 2023
Address: university of Al-Furat Al-Awsat Technical / Kerbala Technical Institute

Head of the Department Certification

In view of the available recommendations, I forward the thesis entitled **"Improvement the Intrusion Detection System based on Hybrid Deep Learning Model"** for debate by the examining committee.

Signature:

Name: Asst. prof. Dr. Saif Mahmoud Khalaf

Date: / / 2023

Address: University of Babylon/College of Science for Women

Dedication

To the one who sought and struggled to enjoy comfort and contentment, to the one who was stingy with nothing in order to push me on the path of happiness, to the one who taught me to ascend the ladder of life with wisdom and patience...**to the soul of the martyr, my dear father**

To the spring that does not bored with giving, to the one who woven my happiness with threads woven from her heart, and who completed the path after my father, to the one who waited for this moment impatiently...**my dear mother**

To whom their love flows in my veins, and their remembrance fills my heart, to the joy of my heart, the joy of my happiness, and the forgetfulness of my life... **my brothers and sisters**

To someone who was keen to elevate my ambitions, knowledge and culture, and be proud of my success and achievement...**my husband and my life partner**

To whom was the blessing of their presence in my life, and whose beautiful laughter filled my life... **my beloved children**

To those who walked together as we pave the way together toward success and creativity...**my friends and colleagues**

I dedicate this research.

Acknowledgments

Thank God, thank you very much for what you have honored me with the completion of this thesis...

I extend my sincere thanks, appreciation and gratitude to my supervisor, **Dr. Wathiq Laftah Abd Ali AL-Yaseen**, who kindly supervised me, His guidance, valuable advice, helpful discussions and depth, knowledge and keen attention to detail have been inspiring and kept my work on a track with this thesis.

Appreciation extends to **all my distinguished Drs in the Department of Computer Science.**

Thank you very much.

Abstract

Networking has a significant part in contemporary life. Cyber security is essential to Internet safety on the network. A cyber security system known as an intrusion detection system (IDS) keeps track of and finds unexpected threats to the hardware and software on a network. Despite the fact that there are numerous IDS already in place, there are still difficulties with increasing the accuracy of spotting security flaws, not enough ways to lower the level of alertness, and spotting intrusion attempts.

The effectiveness of the techniques and the approaches employed to increase classification accuracy and cut down on training and testing times are key components of successful of IDS.

The above issues have been addressed by numerous researchers who have concentrated on building IDSs through Deep Learning (DL) techniques. DL techniques can distinguish between normal and abnormal data and recognize data from prior experiences.

The proposed model enhances intrusion detection and introduces a new strategy to intrusion detection studies. In our approach, we provide a powerful DL approach that relies on hybrid Autoencoder and Multi-Layer Perceptron (MLP) methods. The proposed approach is developed utilizing the Autoencoder technique, it is a successful learning approach for the unsupervised reconstruction of new feature representation by reducing the number of features based on the number of features that we specify by using just the encoder phase. After the training phase, the new data are introduced into the MLP classifier for classification and prediction of the class. The

hybrid model will increase the accuracy of its classification and intrusion detection.

Additionally, the performance of the suggested algorithm was evaluated using the whole Network Security Laboratory-Knowledge Discovery and Data Mining (NSL-KDD) and UNSW-NB15 training and testing datasets for binary and multi-class classification, and it was compared to other approaches that also employed these datasets, like Convolutional Neural Network (CNN), MLP, Recurrent Neural Network (RNN), Support Vector Machine (SVM), and SAE-SVM. The findings indicate that our model sped up testing times and outperformed the majority of the earlier techniques in terms of performance measures for binary and multi-class classification. Obtaining an accuracy rate is 87.6%, and 81.06%, a testing time of 0.768 ms and 0.647 ms (in binary and multi-class classification) respectively, for the NSL-KDD dataset.

Obtaining 88.178%, and 77.47%, a testing time of 0.219 ms and 0.203 ms (in binary and multi-class classification) respectively, for the UNSW-NB15 dataset.

List of Contents

List of Tables

List of Figures

List of Algorithms

List of Abbreviations

Abbreviation	Meaning
AE	Autoencoder
Acc	Accuracy
ANN	Artificial Neural Network
BiLSTM	Bidirectional Long Short-Term Memory
CPU	Central Processing Unit
CNN	Convolutional Neural Network
DL	Deep Learning
DM	Data Mining
DR	Detection Rate
DoS	Denial of Service
FN	False Negatives
FP	False Positives
FAR	False Alarm Rate
HIDS	Host-based Intrusion Detection System
IDS	Intrusion Detection System
KDD	Knowledge Discovery and Data Mining
KNN	K-Nearest Neighbors
LSTM	Long Short-Term Memory
ML	Machine Learning
MLP	Multi-Layer Perception
NSL-KDD	Network Security Laboratory-Knowledge Discovery and Data Mining
NIDS	Network-based Intrusion Detection System
NLP	Natural Language Processing
NB	Naive Bayes
RNNs	Recurrent Neural Networks
ROC	Receiver Operation Characteristic
ReLU	Rectifier Linear Unit
SVM	Support Vector Machine
TN	True Negatives
TP	True Positives

1.1 Introduction

Various network users profit from the Internet's ongoing advancement and widespread use in many different ways. As well as networks are more used, network security becomes considerably more crucial. Network security is intimately connected to computers, networks, applications, different data, etc. [1].

However, the growing number of internet-connected systems become network attack targets, leading to considerable danger and harm. In essence, it's important to develop efficient methods for detecting assaults, fending them off, and maintaining network security.

Moreover, the various attacks need to be handled in various ways. Thus, the primary difficulty in the domain of network security is how to recognize various types of network threats, particularly ones that have never been observed before. Conventional defenses like firewalls and antivirus software are not enough to stop all threats. Therefore, in addition to the conventional approaches, it is necessary to provide more security.

Intrusion Detection Systems (IDS) plays an effective role in this respect. It attentively monitors network traffic data and determines if the content is authoritative or malicious [2].

In order to find malicious behavior, network traffic is monitored by an IDS. It can better identify attacks that get past the firewall. It continuously monitors the network detects threats, and alerts the management when an intrusion is detected. However, despite decades of research, current IDSs still challenge to detection of undiscovered threats, decrease the number of false alarms, and enhance detection accuracy [3].

There are common methods to build an IDS including Machine Learning (ML), statistic knowledge, Data Mining (DM), and expert rules [4].

Through the last several years, researchers have classified network threats using a variety of ML techniques without having any advanced knowledge of their specific properties. However, because of their constraints in computational complexity, conventional ML algorithms are unable to provide discrete feature explanations for the difficulties in threat detection [4].

Recently, there is substantial progress in machine learning through the use of neurons, which are known as DL techniques due to the deep tiers' overall structure to address challenging troubles. DL approaches have lately acquired prominence as strong techniques due to promising outcomes in the disciplines of image processing, Computer Vision (CV), Natural Language Processing (NLP), and other areas [5]. Numerous DL techniques have recently been used in intrusion detection. To achieve intrusion detection, DL techniques can categorize and extract the data automatically.

1.2 Problem Statement

In recent years, the discovery of large numbers of threats besides the fast development of the networks has the difficulty to improve the IDS by using various types of DM and ML algorithms.

The main problem of IDS until now is associated with its performance, especially when dealing with new attacks.

Various techniques have been introduced, but these techniques still need new suggestions for improvement, that is this improvement is still continuing on the issue of the accuracy of the performance of the IDS.

1.2 Thesis Objectives

This research can achieve the following objectives:

- Propose an efficient hybrid model utilizing DL approaches based Autoencoder and MLP methods to improve the effectiveness of detection accuracy and processing time.
- Evaluate the proposed model by using the entire 10% NSL-KDD and UNSW-NB15 datasets for binary classification and multi-classification.
- Employed a new dataset (UNSW-NB15) that contains the latest attacks, As compared to the NSL-KDD dataset.

1.3 Thesis Contributions

- The Hybrid AE-MLP model enhances IDS performance and introduces a new research approach for intrusion detection.
- Introduce an AE model that encodes the input features and the obtained features are compressed and different from the original data then this new data is sent to the MLP classifier.
- Two intrusion datasets, UNSW-NB15 and NSL-KDD, have been used to validate the effectiveness of the suggested method. The outcomes of the trial demonstrate that AE-MLP performs more effectively than conventional approaches.

1.4 Related Works

The most crucial component of the protection networking systems architecture for cyber security is intrusion detection. A variety of ML and DL techniques are applied in IDS to discover and differentiate between legitimate network activity and threats. We review the existing state-of-the-art in improved detection and classification using a DL based neural network here. Where divided the review papers based on the type of dataset that used. The summary of the related works for NSL-KDD and UNSW-NB15 datasets are illustrates in table (1.1) and (1.2) respectively.

I. The Related Works for NSL-KDD dataset

C. zhang et al., 2019 [6], proposed approach is based on DL for IDS to address the issue to some extent. Used Autoencoder and the suggested method developed the network and discovered threats more quickly by using the encoder of the deep autoencoder to reduce the less significant characteristics and extract crucial features without a decoder. The framework was evaluated by using NSL-KDD datasets for 5 – class, the accuracy rate is 79.74%.

S. Z. Lin1et al., 2018 [7], introduced CNN structure in char CNN-IDS which contains 11 layers, 7 of which are convolutional and 4 completely linked. The framework is evaluated by using NSL-KDD datasets for binary class and multi class, the degree of accuracy is 85.07% and 79.05 respectively.

K. WU et al., 2018 [8], suggested a CNN techniques to address a challenge with an imbalanced data. This model would Collect traffic attributes using raw data sets and using each class's numbers, calculate the weighted cost function factor.

The model increases the class's accuracy when working with limited values while simultaneously lowering the false alarm rate (FAR). They changed the original vector traffic data converted to image format to further lower calculating costs. Used the common NSL-KDD data set to evaluate how well the suggested approach performed and the accuracy of this model is 79.48%.

G. Kamdem et al., 2019 [9], proposed building a CNN that can identify breaches in a cyber-physical system by utilizing recent advancements in deep learning. A feature extraction technique with a cross-entropy loss function has been employed with CNN. There are 448 features in the output features based on the NSL-KDD training data. After that, A CNN classifier was used to categorize the 448 characteristics that were produced by the CNN features extraction using two separate class classifications.

The NSL- KDD dataset is employed to assess the suggested CNN-CNN intrusion detection scheme. The framework is tested through the use of NSL-KDD datasets for binary class and multi-class, the accuracy rate is 80.07% and 77.15% respectively.

B. A. TAMA et al., 2019 [10], proposed enhanced IDS using two-level classifier ensembles and hybrid feature selection. To minimize the size of the training data's attributes, a hybrid feature selection strategy which merges 3 techniques Particle Swarm Optimization, Ant Colony and genetic algorithm is used. NSL-KDD and UNSW-NB15 are discussed in this study. A reduced error pruning tree (REPT) classifier's performance is considered when determining attributes.

Then it is suggested to use a two-level classifier ensemble based on rotation forest and bagging, two meta-learners. On the NSL-KDD dataset the suggested algorithm got 85.8% accuracy and on the UNSW-NB15 the accuracy is 91. 27%.

M. AL-QATF et al., 2018 [11], suggested a successful DL strategy built on the IDS for self-taught learning (STL). The Sparse Autoencoder (SAE) and SVM are combined in the suggested method to learn features and reduce dimensionality.

It effectively increases the SVM attack prediction accuracy while reducing the training and testing times by a significant amount. The NSL-KDD dataset is used to validate the methodology for binary and multi-classification, KDDTrain+ 10-fold cross-validation must be employed to determine whether the model is effective for five-category categorization. The accuracy rate was 84.96% and 80.48% respectively.

M. Yousefi-Azar et al., 2017 [12], proposed to learn the latent representation of various feature sets using techniques that utilize Auto-encoders (AEs). They demonstrate how effectively the AE can learn a logical concept of semantic similarity between input data automatically. AE-Gaussian Naive Bayes. The model was evaluated by using the NSL-KDD dataset for binary classification, they achieved an accuracy rate of 83.34%.

C. YIN et al., 2017 [13], presented a DL technique to detecting intrusions based on Recurrent Neural Networks (RNN-IDS). Moreover, they studied the number of neurons and various learning rate that affects how well the suggested system will work. The model is evaluated by using the NSL-KDD dataset for binary classification and they achieved an accuracy rate of 83.28 %.

Y. Wu 1, 2 et al., 2020 [14], proposed Stacked Autoencoder (SAE), SVM, and the kernel approximation method are combined in a joint training model. It was able to utilize a SVM to a properly approximating linear SVM for the Gaussian kernel SVM approximate to the Gaussian kernel SVM thanks to the training model's usage of the SAE to achieve decrease of the attribute dimension, Random Fourier (RF) attributes to accomplish kernel approximation, and explicit application of RF translating to the sample's subset. The SAE then engages in cooperative training with the effective linear SVM. The NSL-KDD dataset was employed to verify the approach and the accuracy for binary classification was 85.8%.

Z. Li et al., 2019 [15], proposed RNN with various numbers of latent layers: Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), as well as analyze which proposed the Broad Learning System (BLS) and its additions. The NLS-KDD dataset is applied to build and validate the methodologies. The methods are constructed based on accuracy and F-Score. The accuracy for binary classification was 81.34 %.

Z. Li et al., 2017 [16], suggested CNN in IDS using the visual converting of the NSL-KDD data structures. Used (ResNet 50) and (Google Net) as CNN algorithms for comparative. Both algorithms utilized cross entropy as the cost function and gradient descent as the optimizer. The accuracy rate for multi and binary classification respectively proposed algorithm 1 (ResNet50) 79.14%, 81.57%, proposed algorithm 2 (Google Net) 77.04%, 81.84%.

II. The Related Works for UNSW-NB15 dataset

Sydney M. Kasongo & Yanxia Sun, 2020 [17], Presented a study of the intrusion detection dataset UNSW-NB15 was utilized for testing and training approach. Additionally, they use the XGBoost method to choose a strategy for feature reduction dependent on filters, then ML techniques utilizing the compressed attribute space to implement the SVM, K-Nearest-Neighbor (KNN), Logistic Regression (LR), Artificial Neural Network (ANN), and Decision Tree (DT). Both binary and multiclass classification sets were taken into account in the trials.

The outcomes showed that techniques like the DT can improve the accuracy rate by using the XGBoost-based feature selection mechanism. By lowering the dimension of the attribute parameter from 42 to 19, the majority of machine learning techniques employed in this study increased their accuracy rate for the multiclass classification method. An improvement above 75.62% (42 characteristics with 30 neurons in the hidden layer) to 77.51% (19 instances and 15 neurons in the hidden layer) was observed in the ANN case.

When employing the 19 features produced by the XGBoost technique instead of the 42 features used in the KNN technique, the accuracy for multiclass improved from 70.09% to 72.30% on the testing set. Utilizing 42 and 19 characteristics for the multiclass classification, the DT classifier's accuracy rate went up from 66.03 to 67.57%. The accuracy rate in the multiclass class increased from 61.09 to 61.53% when the SVM technique was used with the best input vector (19 characteristics). Additionally, the SVM method improved the binary classification accuracy from 70.63 to 75.51%.

S. Almogren, 2020 [18], proposed a method to fully utilize the potential of the IoT by promptly and correctly detecting intruding behaviors in the IoT network. Particularly, a Deep Belief Network (DBN) depending on a powerful intrusion detection method was proposed. Researched various detection models using various DBN architectures, additionally, after the feature subset is chosen, a DBN method uses it to do the estimates.

The UNSW-NB15 was utilized by the authors to evaluate the effectiveness of their suggested approach. The primary performance measure was the accuracy discovered using testing data. The topology of a network (layer1= 64 hidden items, layer2= 60 hidden items, and the serial value of the DBN structure is 28) obtained an accuracy of 85.73% for the two class. The authors carried out the experiments utilizing various architectures of DBN networks.

J. Gao et al., 2019 [19], suggested combined Extreme Learning Machine (IELM) incremental technique and the Advanced Principal

Component (APCA) method. The APCA's job is to produce an appropriate threat prediction by flexibly choosing the attributes that the IELM considers more important.

The UNSW-NB15 was employed by the researchers to measure the effectiveness of the suggested IDS approach. The most crucial factor taken into account to reach their result was the accuracy gained from the testing data. The researchers additionally thought about additional parameters like the FAR and the detection rate (DR). According to the experimental outcomes, the IELM-APCA achieved an accuracy of 70.51%, DR of 77.36%, and FAR of 35.09%.

Zong et al., 2018 [20], presented a Two-Stage (TS) architecture according to the RF method. The criteria for the binary classification required attributes were chosen using the Information Gain (IG) approach. The classified features of the UNSW-NB15 dataset were modified by the researchers using the one-hot encoding method.

The first part of the IG-TS involves the identification of minority classes, while the second part involves the identification of majority classes. The overall prediction is then produced by combining the outcomes of each step. The Accuracy and the FAR were used by the authors as evaluation criteria. The IG-TS obtained an Accuracy of 85.78% and a FAR of 15.78% after multiple testing.

KAIYUAN et al., 2020 [21], suggested an IDS that integrated deep network and hybrid sampling. They first eliminated the noise data in the majority class using One-Side Selection (OSS), then, they increased the minority instances using the Synthetic Minority Over-sampling Technique (SMOTE).

By doing this, a balanced data may be created, allowing the approach to properly understand the attributes of minority instances while also significantly reducing the amount of time it needs to train. Next, they create a deep network model using CNN to obtain spatial characteristics and a Bi-directional Long Short-Term Memory (BiLSTM) to discover temporal characteristics. Trials were used to validate the suggested intrusion detection technique on the UNSW-NB15 dataset, and the classification accuracy can be obtained at 77.16%.

Dishan Jing and Hai-Bao Chen, 2019 [22], Proposed SVM. The performance of the method is evaluated with the UNSW-NB15 dataset via accuracy, DR, and FBR.

The empirical findings demonstrate the suggested SVM method's advantage over alternative approaches. When it comes to binary class, the accuracy of the suggested approach is 85.99%, versus 78.47% for Expectation-Maximization (EM) clustering. The accuracy of the suggested SVM for multi-classification is 75.77%, 6.17% more than that of Nave Bayes (NB). In the binary class, the DR is 97% and the FPR is 3%, DR is 50.9% and FPR is 3.0% for multi classification.

The above works show that most researchers try to combine various techniques to achieve high accuracy for IDS; there is still room for improvement. Therefore we believe the proposed work in this research can improve the current research domain.

Table (1.1): The Summary of Related Works for NSL-KDD dataset.

Reference	**Techniques**	**Supervised or unsupervised**	**Datasets**	**Performance (accuracy)**
C. zhang et al., 2019 [6]	AE	Unsupervised	Full NSL-KDD	79.74% in multi classification.
S. Z. Lin et al., 2018 [7]	CNN	Supervised	Full NSL-KDD	85.07% in 2 class 79.05% in 5 class

K. WU et al., 2018 [8]	CNN	Supervised	Full NSL-KDD	79.48% in 5 class.
G. Kamdem et al., 2019 [9]	CNN-CNN	Supervised	Full NSL-KDD	80.07% in 2 class 77.15%. in 5 class
B. A. TAMA et al., 2019 [10]	(PSO + Ant Colony + GA).	Unsupervised	20% of NSL-KDD dataset, KDDTrain+, it includes 25,192 instance, with 13,499 attack and 11,743 normal instance. Two separated test sets, KDDTest+ (22,544 instance) and KDDTest21 (11,850 instance).	85.8% In binary classification.
M. AL-QATF et al., 2018 [11]	Self-taught learning (STL), (SAE + SVM).	Unsupervised + supervised	NSL-KDD (Used 10-cross validation).	In 2 class 84.96% In 5 class 80.48%.
M. Yousefi-Azar et al., 2017 [12]	AE-Gaussian Naive Bayes.	Unsupervised + supervised	Full NSL-KDD	83.34% for binary class.
C. YIN et al., 2017 [13]	RNN	Supervised	NSL-KDD (Used 10-cross validation).	83.28 % in binary class.
Yukun Wu 1, 2 et al., 2020 [14]	(JSAE-FSVM) SAE + SVM + the kernel approximation method.	Unsupervised + supervised	Full NSL-KDD	85.8% in 2 class.
Z. Li et al., 2019 [15]	RNN using a varying number of latent tiers: (LSTM) + Gated Recurrent Unit (GRU).	Supervised	Full NSL-KDD	81.34 in 2 class.
Z. Li et al., 2017 [16]	Google Net and ResNet 50 as CNN models.	Supervised	Full NSL-KDD	(ResNet50) 79.14%, 81.57%, (Google Net) 77.04%, 81.84%. For multi and binary

				classification respectively.

Table (1.2): The Summary of Related Works for UNSW-NB15 dataset.

Reference	Techniques	Supervised or unsupervised	Datasets	Performance (accuracy)
Sydney M. Kasongo & Yanxia Sun, 2020 [17]	ANN LR KNN SVM DT	Supervised	UNSW-NB15 (Division of the UNSW-NB15-TRAIN into two parts: UNSW-NB15-TRAIN-1 (75% of the whole training data) for training and UNSW-NB15-VAL (25% of the entire training set) for validation prior to testing.	In multi class 75.62, 65.53, 70.09, 61.09, 66.03 respectively. In binary class 86.71, 79.59, 83.18, 62.42, 88.13 respectively.
A. S. Almogren, 2020 [18]	Deep Belief Network (DBN) classifier	Unsupervised	Full UNSW-NB15	85.73% in binary classification.
J. Gao et al. , 2019 [19]	Extreme Learning Machine (IELM) progressive strategy using Developed Principal Component algorithm (APCA).	Unsupervised	Full UNSW-NB15	70.51% in multi class.
Zong et al. , 2018 [20]	Depending on the RF classifier, the two-stage (TS) classifier approach, Information Gain (IG) technique. (IG-TS).	Supervised	Full UNSW-NB15	85.78 % in binary classification.

K. JIANG et al., 2020 [21]	CNN +BiLSTM	Supervised	Full UNSW-NB15	77.16 in multi class
D. Jing and H. Chen, 2019 [22]	(SVM)	Supervised	Full UNSW-NB15	85.99% for binary classification 75.77% for multi classification

1.5 Thesis Organization

This research is organized into five chapters, the summary of each chapter is as follows:

- Chapter one includes introduction about this research.
- Chapter two includes the background on the intrusion detection systems and the deep learning techniques based on IDS, specific problems and challenges related with the intrusion detection system.
- Chapter three describes the framework of the proposed work and its level.
- Chapter four presents the obtaining results, analysis, and discussions.
- Finally, chapter five presents conclusion and future works.

1.6 Summary

- Reference studies, as well as this research, focused on the importance of intrusion detection systems in protecting information from attacks sophisticated and growing spread across the Internet and various types of networks.
- The reference studies included different algorithms and approaches that were applied and tested in the design of intrusion detection systems research machine learning methods in designing an intrusion detection system, then the results were compared with the results of the techniques in previous studies.

- The purpose of this research is to improve the detection rates of types of violence within the NSL-KDD and UNSW-NB15 datasets gained from previous research improved the false alarm rates to be as low as possible.

2.1 Introduction

Network security is a crucial area of study since the internet's expansion has a growing impact on contemporary life. Network information manipulation, counterfeiting, viruses, and hacking are the main types of network security breaches. Serious losses come from these security failures. To protect ourselves from these risks, we utilize cyber security such as anti-virus software, firewalls, an intrusion detection system (IDS) is used to deal with increasingly complex and serious cyber-attacks IDS still has trouble detecting unexpected violation, lowering the false alarm rate, and improving detection accuracy [23].

The goal of security violation is typically to obtain access to sensitive information systems without authorization and then change, reveal, or use that information [24]. Consequently, in light of their rising ferocity and variety, therefore, developing efficient and effective strategies to control cyber security threats seems to be crucial given their growing intensity and variety.

2.2 Intrusion Detection Systems (IDS)

The Internet is becoming more and more important to our work and daily lives as internet technology continues to improve and get better. However, a lot of data is created, processed, and transferred when people use and interact with the internet.

2.2.1 Overview

These data have been used for illegal purposes, posing a serious risk to the safety of networks. Fortunately, the intrusion detection system (IDS) can effectively address these issues. Jim Anderson initially put forth the concept of IDS in 1980. Since then, multiple IDS programs have advanced to meet the demands of system security [25].

IDS monitors networks or hosts as an active security technology and issues notifications when threats are found. Through the use of intrusion detection techniques, network attack behavior can be better protected by understanding it through data analysis and modeling. A crucial issue that cannot be avoided is how to recognize any form of illegal network activity, particularly unexpected harmful [26].

The researchers have investigated how to employ machine learning (ML) and deep learning (DL) approaches to meet the needs of an efficient IDS. The main goal of ML and DL is to gather useful information from large amounts of data. These techniques have gained a lot of traction in the field of cybersecurity as a result of the past ten-year increase in the performance of graphics processing units (GPUs).

Both ML and DL are effective ways to extract important information from connectivity, forecast both regular and unusual action based on the similarities discovered, and identify key features. For the ML-based IDS to get useful information from network activity, the concept of data preprocessing is crucial. While DL-based IDS do not depend on feature extraction and are efficient learning automatically complicated attributes of the original data because of their deep network [25].

2.2.2 Concept of IDS

The terms "intrusion detection system," or IDS, are a combination of the two terms. Unauthorized access to data in a computer or network system that violates its availability, confidentiality, or security is referred to as an intrusion [25]. Whereas a detection system is a security tool for spotting this sort of violent conduct. Therefore, Intrusion Detection systems (IDS) are hardware or software systems that monitor and control the operation display network or computer system occurrences and analyze them from a security point of view. As network threats rise, IDS is now a crucial complement to cybersecurity. [24].

In order to protect the security and dependability of systems and resources, intrusion detection gathers data from a number of crucial points in computer network systems, including audit data of the operating system, system logs, and network data packages. The collected data is then analyzed to identify behaviors that violate security rules and adversely affect system security. Availability and reliability intrusion detection system is an expansion of the conventional firewall and other protection mechanisms. It focuses on monitoring system and network communication data, and by taking specific corrective actions, it lowers the system's risk of victimization and fully secures the host and network [25].

Obtaining a good dataset is a substantial difficulty in and of itself since many academics suffer to get complete and reliable datasets to test and analyze their suggested methodologies. Reliable datasets that (i) contain both benign and many threats, (ii) interact with real requirements, and (iii) are easily accessible are required in order to verify the effectiveness of such techniques [26].

2.2.3 Classification of IDS

It is possible to categorize intrusion detection systems using several methodologies illustrated in figure (2.1) [27]. It can be classified based on attack type for Network Intrusion Detection System (NIDS) and Host Intrusion Detection System (HIDS). NIDS are positioned strategically throughout the network to watch traffic. While scanning all incoming and outgoing traffic would be ideal, doing so could slow down the network as a whole. In contrast, HIDS operates on different hosts or the network's devices and only monitors incoming and outgoing packets from the device, alerting the user or supervisor if a strange activity is discovered. Our proposed system is NIDS.

There are several IDS that look for specific signatures of known risks in order to find threats, the same as how antivirus programs normally identify threats and guard against malware, a process known as Signature-based Detection (SD). A pattern or string that is associated with a known threat or attack is called a signature. SD is sometimes referred to as knowledge-based detection or misuse detection because of the fact that it makes use of the information gathered by particular threats and system flaws [27].

There are IDS that check for anomalies by comparing traffic patterns to a baseline and detecting them. An illustration of this kind of IDS is anomaly-based IDS, which will track network activity and evaluate it in comparison to a predetermined standard. The baseline can observe often network threats by comparing the present anomalous behavior with what is thought to be usual. It will notify the supervisor or user when connectivity is observed that is anomalous, or completely different, from the baseline. The baseline will define an intrusion when the analyzed activities in computer systems display a huge variance from the usual case profile constructed on long-term regular activities [27]. Our proposed system is anomalies.

IDS are divided into active and passive IDS when they are categorized according to how they behaved following the attack. The benefit of the active IDS is that it can immediately take corrective action in the event of an attack. An IDS that just monitors and analyzes network traffic activity and notifies an operator of potential threats and flaws is known as a passive IDS. A passive IDS cannot, by itself, carry out any defensive or remedial actions [27]. Our proposed system is passive IDS.

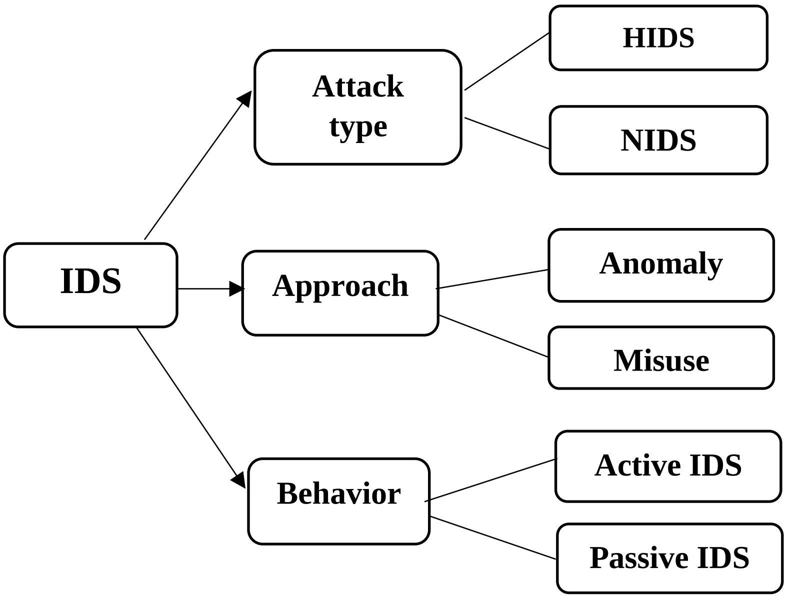


Figure (2.1): The IDS classification types [27].

2.3 Machine Learning Models (ML)

ML is used to teach machines how to handle data more efficiently. Sometimes after viewing the data, we cannot interpret the extracted information from the data. In that case, we apply machine learning. With the abundance of datasets available, the demand for ML is on the rise. Many industries apply machine learning to extract relevant data.

The purpose of machine learning is to learn from the data.

Many studies have been done on how to make machines learn by themselves without being explicitly programmed. Many mathematicians and programmers apply several approaches to find the solution to this problem which is having huge data sets [50].

There are two main types of ML: supervised and unsupervised learning. Supervised learning relies on useful information in labeled data. Classification is the most common task in supervised learning (and is also used most frequently in IDS); however, labeling data manually is expensive and time consuming. Consequently, the lack of sufficient labeled data forms the main bottleneck to supervised learning.

In contrast, unsupervised learning extracts valuable feature information from unlabeled data, making it much easier to obtain training data. However, the detection performance of unsupervised learning methods is usually inferior to those of supervised learning methods. The common ML algorithms used in IDSs are shown in Figure (2.2) and the advantages, disadvantages of various ML techniques in table (2.1) [3].

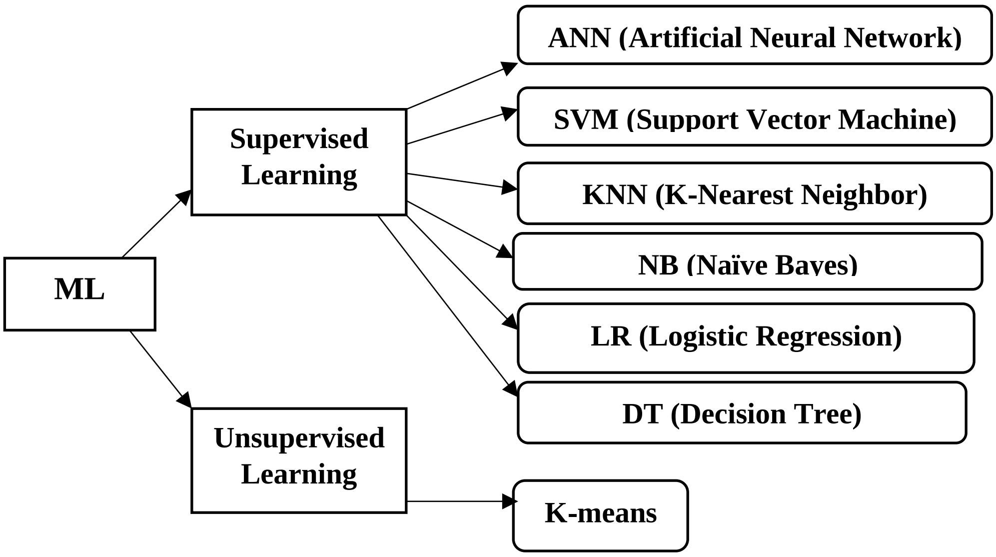


Figure (2.2): Taxonomy of ML techniques [3].

Table (2.1): The Advantages and Disadvantages of Various ML techniques.

Techniques	Advantages	Disadvantages
ANN	• Able to deal with nonlinear data; • Strong fitting ability	• Apt to overfitting; • Prone to become stuck in a local optimum; • Model training is time consuming
SVM	• Learn useful information from small train set; • Strong generation capability	• Do not perform well on big data or multiple classification tasks; • Sensitive to kernel function parameters
KNN	• Apply to massive data; • Suitable to nonlinear data; • Train quickly; • Robust to noise	• Low accuracy on the minority class; • Long test times; Sensitive to the parameter K

Naïve Bayes	• Robust to noise; • Able to learn incrementally	• Do not perform well on attribute-related data
LR	• Simple, can be trained rapidly; • Automatically scale features	• Do not perform well on nonlinear data; • Apt to overfitting
DT	• Automatically select features; • Strong interpretation	• Classification result trends to majority class; • Ignore the correlation of data
K-means	• Simple, can be trained rapidly; • Strong scalability; • Can fit to big data	• Do not perform well on nonconvex data; • Sensitive to initialization; • Sensitive to the parameter K

2.4 Deep Learning Models (DL)

In essence, it's important to develop efficient methods for detecting violent acts, fending them off, and maintaining network security. Furthermore, various violence types typically need to be handled in various ways. Thus, the primary problem in the field of network security is how to recognize several forms of malicious activities, particularly ones which haven't been seen previously [28].

Across recent years, researchers have categorized network violent acts using a variety of ML techniques without having any previous knowledge of their specific properties. Furthermore, owing to the constraints on model accuracy, existing machine learning techniques cannot offer distinguishing feature descriptors to describe the difficulty of detection accuracy. Lately, DL techniques named for their overall design of deep layers to address challenging troubles have achieved a significant advancement in ML by modeling the human brain with neural network structure [27].

One of the hot topics in current academic study is the use of DL for network intrusion detection. The advancement of hardware computer power and the quick increase in data size has encouraged the development of DL, which has substantially increased its applicability and popularity [28].

DL is a ML technology created to enable artificial intelligence to enhance computer systems through experience and data. To describe data learning, DL employs numerous nonlinear feature transformations or processing layers created by multilayer perceptual processes [29]. Application of DL in computer vision [30], speech recognition [31], natural language processing [32], biomedicine [33], and malicious code detection [34], along with numerous other domains.

Since 2015, DL research for network security has increasingly come into focus, garnering significant interest from academics. The IDS system is depicted in figure (2.3), and the researchers concentrated on DL techniques for its design. It is notable that only 20% of recommended solutions are based on ML models, while 60% of proposed ways are totally dependent on DL techniques, and 20% of solutions use a hybrid strategy that combines ML and DL-based techniques. DL enhances detection effectiveness and lowers false positives when compared with standard ML, which is currently employed mostly in the two core network security domains of malware detection and network intrusion detection [25].

Additionally, deep learning techniques eliminate the need for feature engineering and have the capacity to automatically recognize attack features, aiding in the detection of potential security flaws.

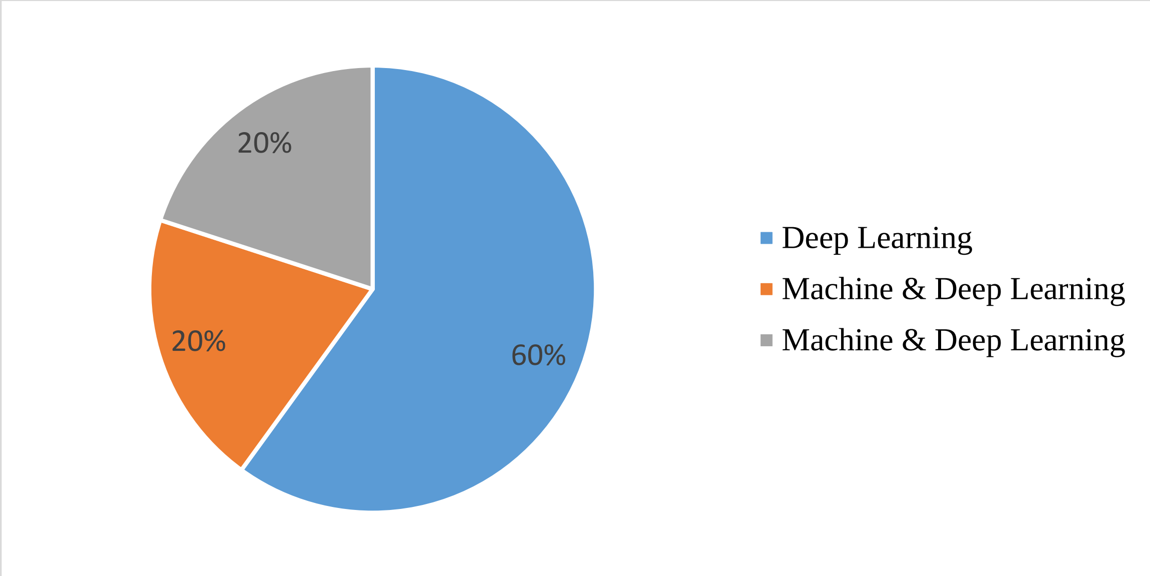


Figure (2.3): The Rate of Researchers Focused on DL Methods for Effective IDS [25].

Moreover, figure (2.4) displays how frequently ML or DL-based techniques are used by researchers to create effective IDS solutions. The four most often employed techniques, all of which are DL in type, are noticed to be AE, DNN, CNN, and RNN, in that order. Following that, ML-based methods like RF and SVM are included in the ranking and are primarily utilized in combination designs to support and enhance DL-based techniques. Additionally, less often used ML-based techniques include DT, KNN, and FLN [25].

The ability of deep architectures to learn features with different levels of abstraction at multiple processing stages without human involvement is the primary distinction between shallow ML and DL. Deep models therefore automatically identify intricate mappings and relationships between raw input and output [25].

Additionally, while shallow ML algorithms break a problem into multiple sections, solve each one separately, and then combine the findings, DL offers end-to-end problem-solving. Both dimensionality reduction and classification problems in anomaly detection can be performed using deep learning techniques. DL methods automatically learn complicated data.

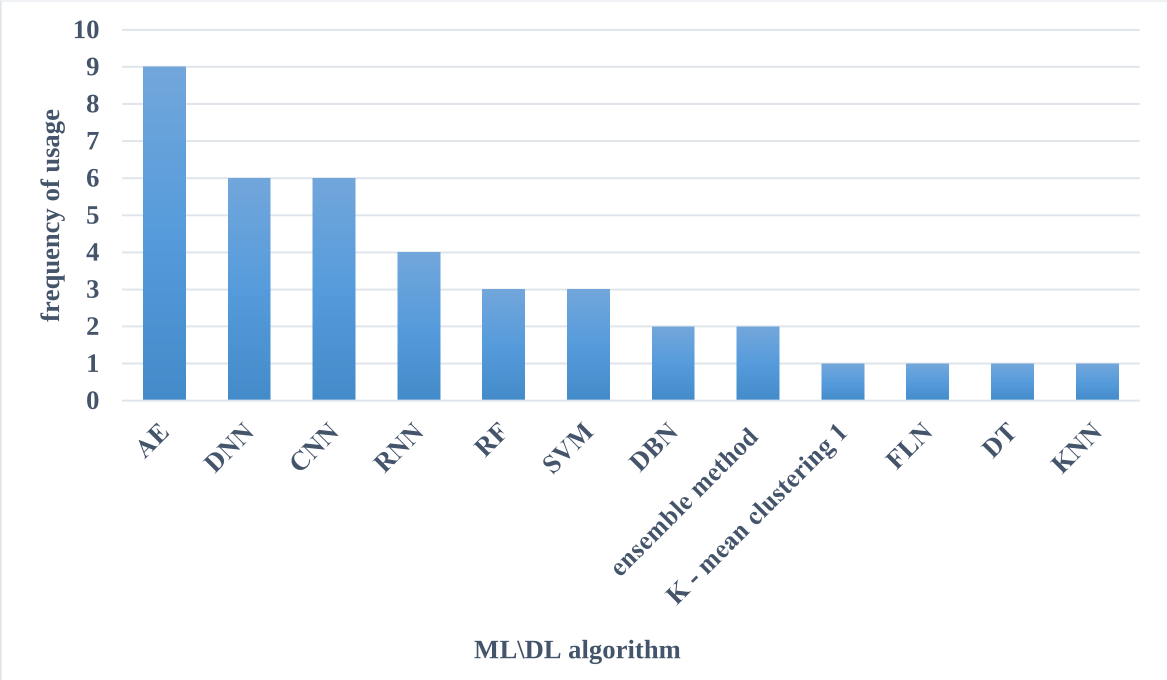


Figure (2.4): The Rate at Which Researchers Use ML or DL Methods to Develop Efficient IDS Solutions [25].

2.5 Deep Learning Techniques

Three main categories are used to classify deep networks: unsupervised (e.g., Autoencoder (AE), deep belief network (DBN), and generative adversarial network (GAN)), supervised (e.g., deep neural network (DNN), convolutional neural network (CNN), and recurrent neural network (RNN)), and other hybrid techniques; we display the classification information to Security and Communication Networks in figure (2.5) [35].

Various DL techniques could offer multiple benefits for attack detection techniques. Because of the large amount of data offered by manually labeled samples, unsupervised learning-based algorithms frequently produce excellent accuracy. Supervised learning-based techniques typically perform poorly without enough information from labeled data. However, manually labeling takes a while, particularly for complicated attacks. Due to the intrinsic complexity of actual network attacks, there are circumstances that cannot be adequately characterized by a single label. Therefore, approaches based on unsupervised learning could function effectively without being aware of attacks beforehand, which is a clear advantage.

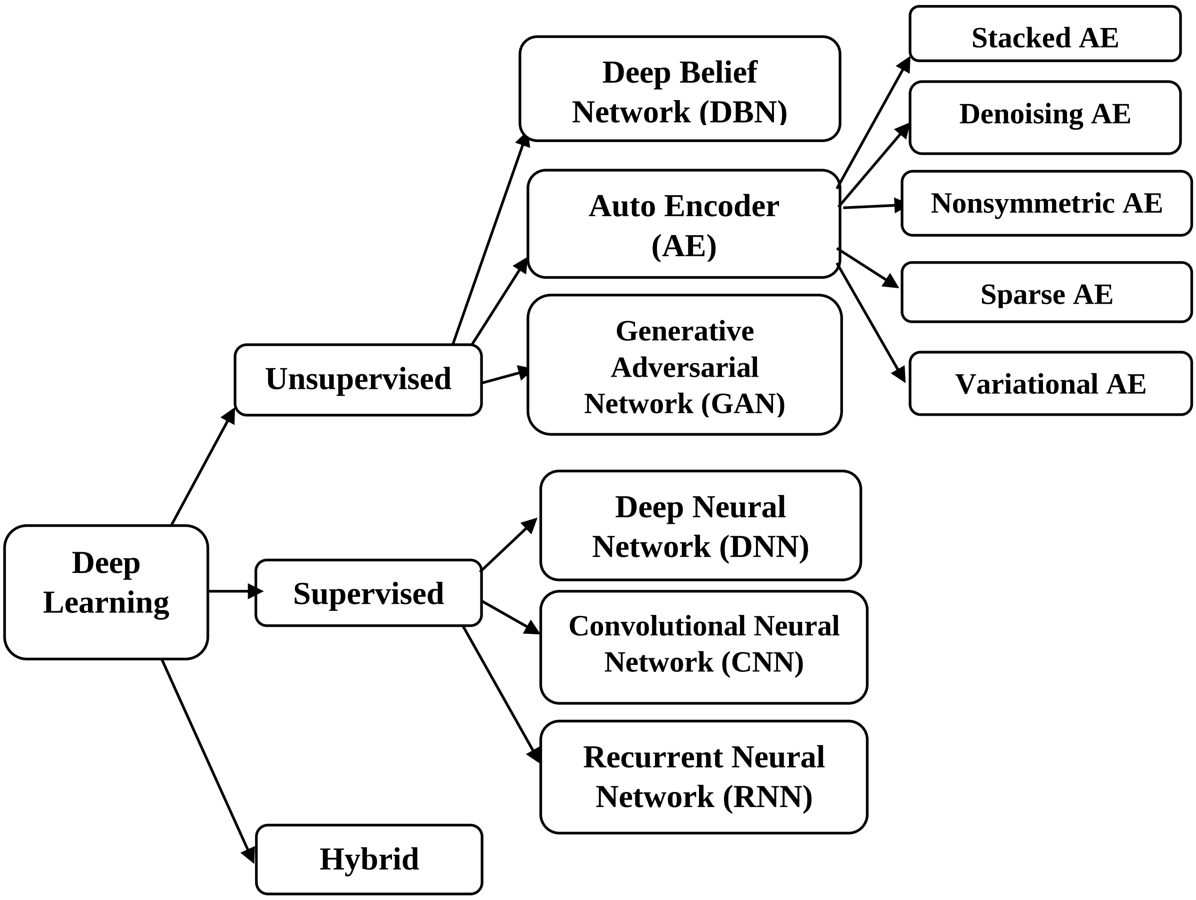


Figure (2.5): Classification of the Available DL Intrusion Detection Techniques [35].

Hybrid approaches use fewer training samples while maintaining excellent performance, making them suited for dealing with a variety of assault scenarios. However, its widespread use is limited by its typically intricate design and lengthy processing time [35]. Table (2.2) illustrates the comparison between the deep learning techniques widely used.

Table (2.2): Comparison between DL Techniques.

Techniques	Supervised Or Unsupervised	Advantages	Disadvantages
Auto Encoder	Unsupervised	• Reducing the dimensionality of the data we are using and the learning time. • The compactness and speed in coding using backpropagation.	• To maintain a robust Autoencoder, you need a large representative data set and that training a robust autoencoder will take time.
DBN	Unsupervised	• Special resilience in categorization size, and location, the same neural network technique can be used on multiple applications and data formats. • Efficient in the use of hidden layers greater performance increase by adding layers compared to Multilayer perceptron. • Even when there is a large quantity of data, DBN's characteristics are routinely obtained and properly tailored for predicted outcomes, avoiding time-consuming procedures.	• Machines are required. • It is challenging for those with less experience to utilize. • It requires classifiers to grasp the output. • It requires huge data to adopt efficient strategies. • Training is costly because it has complicated data structures,
GAN	Unsupervised	• It creates data that resembles the original data. • It can also produce multiple versions of the text, video, and audio, as well as dive into the minutiae of data and readily interpret it into different forms, making it	• It's more difficult to train since you have to constantly submit new sorts of data to see if it's accurate or not. • It's difficult to get outcomes from text or speech.

		useful for machine learning work. • It can measure the distance between different items.	
DNN	Supervised	• The identical neural network-based strategy may be used for more varied applications and data sources. • The DL approach is adaptable enough to handle new challenges as they arise.	• It requires a lot of data to outperform alternative methods. • Because of the intricate data structures, training is costly. • Since selecting the appropriate DL tools necessitates an understanding of structure, training techniques, and other features, there is no established theory to guide you.
CNN	Supervised	• It is developed to analyze multidimensional array data. • It can identify local feature correlations and improve feature extraction accuracy. • Combining CNN with Dropout, ADAM, and Softmax classifiers.	• If the system doesn't have a strong GPU, CNN training will take a lengthy time. • A CNN needs a vast dataset to process and train.
RNN	Supervised	• Remembers each and every piece of information through time. • The ability to identify prior inputs makes it valuable for time series prediction, this is called Long Short Term Memory. • Used to improve the effective pixel area with convolutional layers.	• It is exceedingly tough to train an RNN. • If tanh or ReLU are used as the activation function, it cannot analyze extremely large amounts of data.

Table (2.3) presents the comparison of the deep learning and machine learning techniques [51]. However, the important issue in both methods is the quality of the data and it determines the quality of the final result.

Table (2.3): The Comparison of Deep Learning and Machine Learning Techniques.

Deep Learning (DL)	**Machine Learning (ML)**
• Need a large dataset for training.	• Can be trained with small datasets.
• Can learn features from the raw data.	• External intervention is necessary to provide the right input.
• Do not require human intervention.	• Need to be retrained through human intervention.
• Regarding the overheads of the DL methods, they are more appropriate for dealing with largescale problems.	• Because ML needs labeled data, it is not appropriate for handling large-scale problems that need a large set of labeled data.
• Solving large-scale problems using high-end computer systems or special hardware such as Graphics Processing Unit (GPU) can increase the performance of the DL algorithms.	• ML algorithms can be handled using CPU on low-end computer systems.
• DL networks require more execution time.	• ML techniques need less time for execution.
• The problem solving using DL methods, based on multi-layered ANNs, are more intertwined and complex.	• The problem solving using ML such as decision tree and linear regression, benefit from a simple structure and is much easier to interpret.
• The output of the DL methods can be in various forms.	• The output of the ML techniques is in numerical form.

For this work, the typical methods used in experiments, each one of these methods will be briefly discussed in the following:

2.5.1 Autoencoder

It is an artificial neural network that teaches a reduced (encoded) version of the input data. With an initial tier that is smaller neurons in their hidden layers and the exact size as the previous layer, these networks have a unique hourglass shape. A basic Autoencoders objective is to rebuild the input data so that x_i' is as similar to x_i as possible. To put it in another way, an Autoencoder picks up a rough estimate of the identity function. The problem is that the network must identify patterns and constructions in the input data in order to encode it (in a hidden layer) and subsequently decode it (output layer) because there are only a finite number of hidden neurons [36].

There are two symmetrical elements in an auto encoder: an encoder and a decoder. The encoder takes the raw data and derives features, whereas the decoder uses those characteristics to reconstruct the data. Slowly reducing the discrepancy between the decoder's output and the encoder's input occurs throughout training. The encoder's factors reflect the data's core if the decoder is able to recreate the data using the extracted features.

The AE constructions includes an input layer, a hidden layer, and an output layer. As shown in figure (2.6) [26], the input layer and the output layer are equal in size, and the size of the hidden layer must be smaller than that of the input layer.

For a given training dataset $x = \{x_1, x_2, \dots, x_m\}$ with m instances, where xi is a d-dimensional attribute vector, the input vector is transformed by the encoder xi to a hidden representation vector h_i through a deterministic transforming f_Θ as given in (2.1)

$$h_i = f_{\Theta}(x_i) = s(w * x_i + b) \qquad (2.1)$$

Where w, $\acute{d} \times d$, $\acute{d}$ the number of hidden tiers, b is a bias vector, θ is the transforming variable set $\theta = \{w , b \}$.

The ReLU (rectified linear unit) function was used. The ReLU function can be expressed as $f(x)$ = max (0, x), and if x > 0 the output is a straight line with a slope of 1, and if x < 0 the output value is always zero.

The decoder return the hidden form that produced h_i to a rebuilt a vector in the d-dimension y_i in the input layer like in (2.2):

$$y_i = g_{\acute{\Theta}\,(x_i)} = s\left(w_{h_i} + \acute{b}\right) \qquad (2.2)$$

Where $\acute{w}$ is a $d \times \acute{d}$, $\acute{b}$ is a bias vector and $\acute{\Theta} = \left\{\{\acute{w}, \acute{b}\}\right\}$. The autoencoder must be trained to minimize the variance between input and output. Consequently, the below equation is used to calculate a loss function in (2.3):

$$\mathcal{L}(x, y) = {}^{1}\!/_{m} \sum m_i = 1 \; \|x_i - y_i\|\text{^}2 \qquad (2.3)$$

Where m the overall number of training set.

The important goal is to obtain the best parameters(Θ and $\acute{\Theta}$). This efficiently reduces the disparity among input and rebuilt output across the full training set like on (2.4):

$$\Theta = \{w, b\} = arg_{\Theta} minL(x, y) \qquad (2.4)$$

Adjusting the parameters of the encoder and decoder minimizes errors between the output data and the original data.

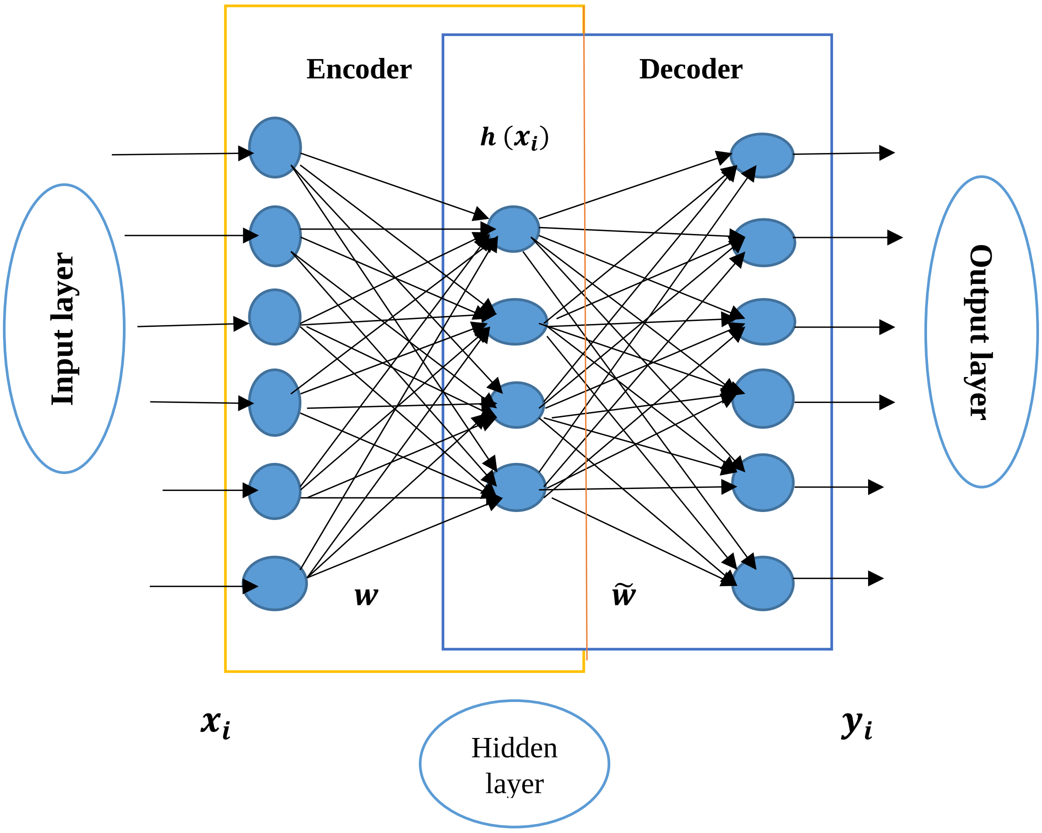


Figure (2.6): The Structure of AE [26].

2.5.2 Types of AE

There are many types of Autoencoder used for various purposes:

1. Stacked Autoencoder

Another construct with symmetry is the SAE. AEs are stacked one on top of the other to create SAE. The next AE is trained to utilize the hidden layer from the initial AE after the single-layer AE has been trained. By doing this step again.

To obtain greedy hierarchical learning, AEs are tired, with the lth hidden layer serving as input to the stack's l + 1st hidden layer. Rather than initializing the weights at random, the SAE outcomes are utilized to pre-train the fully linked DNN. This technique aids in improving the optimal effect and initializing model parameters around desirable local minimum values [36].

2. Sparse Autoencoder

It is utilized for data reduction and unsupervised rebuilding of additional data representation. Minimal training time is required by the sparse Autoencoder, which also improves prediction performance [37].

3. Denoising Autoencoder

A Denoising Autoencoder works by rebuilding the original input after part damaging the actual model input in order to train an Autoencoder algorithm to provide a resilient input feature representation. The procedure described here is how a Denoising autoencoder learns [38].

4. Variational Autoencoder

In contrast to conventional Autoencoders, a variational Autoencoder requires the hidden representation's probability to adhere to a specified probability. In principle, variational Autoencoders work similarly to stacked autoencoders in that the encoder creates a hidden representation, and the decoder reconstructs the input data using the hidden representation [39].

5. Nonsymmetric Autoencoder

It has many non-symmetric hidden tiers and an autoencoder. Essentially, this entails the suggested change from the symmetric encoder-decoder architecture to relying solely on the encoder phase (Nonsymmetric). This is justified by the fact that, given the right ability tests, it is possible to minimize the impact on precision and efficiency while reducing computational and time overheads.

NDAE functions effectively as a scalable hierarchical unsupervised feature extractor for high-dimensional inputs. It employs a training method similar to that of a normal Autoencoder to learn non-trivial features [40].

2.5.3 Multi-Layer Perceptron (MLP)

It is one very common varieties of ANN, is an artificial neural network that feeds information forward. It consists of an input layer, one or more hidden layers, and an output layer. One or more neurons make up each layer, and their job is to apply what is known as an "activation function" to the input. The neuron also adds a value known as "bias," and each link between nodes has a weight corresponding to its significance in the network [41].

Deep MLP is defined as having more than one hidden layer. Before being supplanted by other machine learning techniques like support vector machines in the 1990s, MLPs were widely employed for pattern identification and handwriting recognition the 1980s. MLPs are frequently employed for intrusion detection since they are among the simplest neural network architectures to develop. Figure (2.7) illustrates the structure of MLP model [42].

Numerous neurons make up each tier, and each neuron is completely coupled with the neuron in the tier below by nodes connected. As a result, the network can be expressed generally as given in (2.5):

$$Y = F\left[\sum_{j=1}^{m} W_{kj} \, . \, F\left[\sum_{i=1}^{n} W_{ij} \, X_i + B_j\right] + B_k\right] \tag{2.5}$$

where W_{ij} are the weights between hidden and output layer, W_{kj} are weights between input and hidden layers; and X_i are input parameters; m is the amount of cells in a hidden layer; n is the amount of cells in an input layer, B_j and B_k are the bias values of the cells in the hidden and output layers, respectively; F is the transfer function; and Y is the output [43].

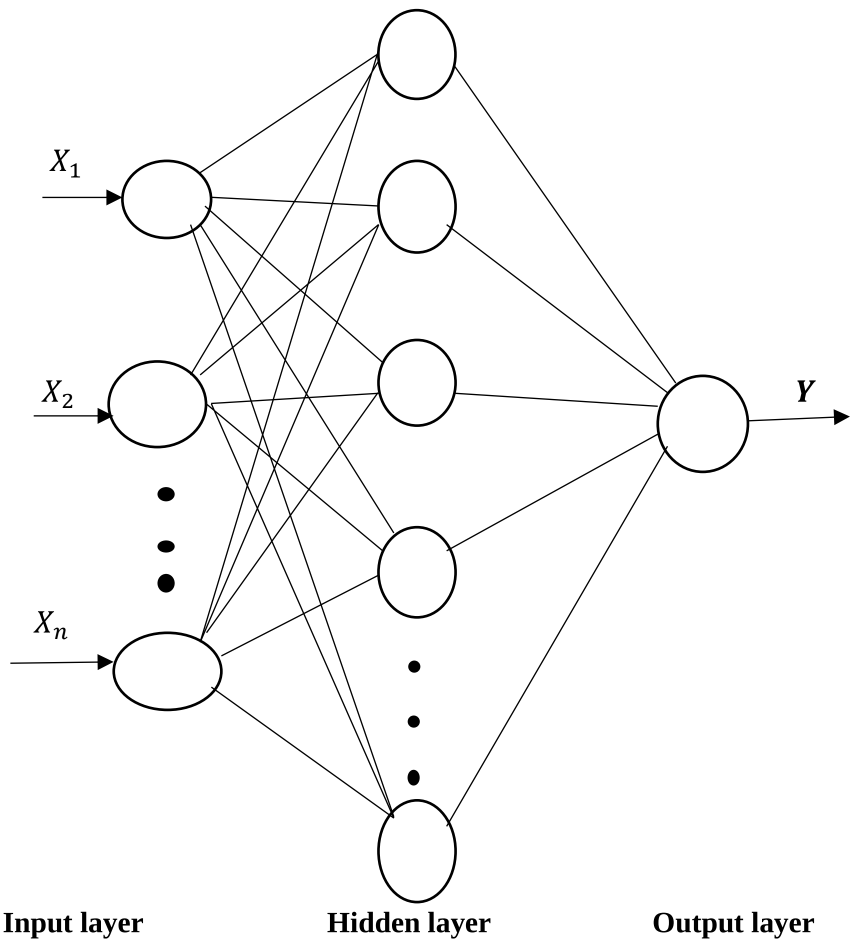


Figure (2.7): The structure of MLP [42].

2.6 Benchmark Datasets

Standard datasets are a crucial component used to evaluate the effectiveness of the suggested strategy. Figure (2.8) displays the analysis for using the available datasets [25]. The NSL-KDD and KDD Cup'99 were utilized for testing and verifying 60% in the literatures, as shown. Because of the availability of numerous outcomes in the literature, despite the fact that both of these datasets are fairly old, researchers continue to use them frequently and found from the UNSW-NB15 modern datasets were more used also. Therefore, we will use these datasets to evaluate the performance of proposed DL-IDS in all phases.

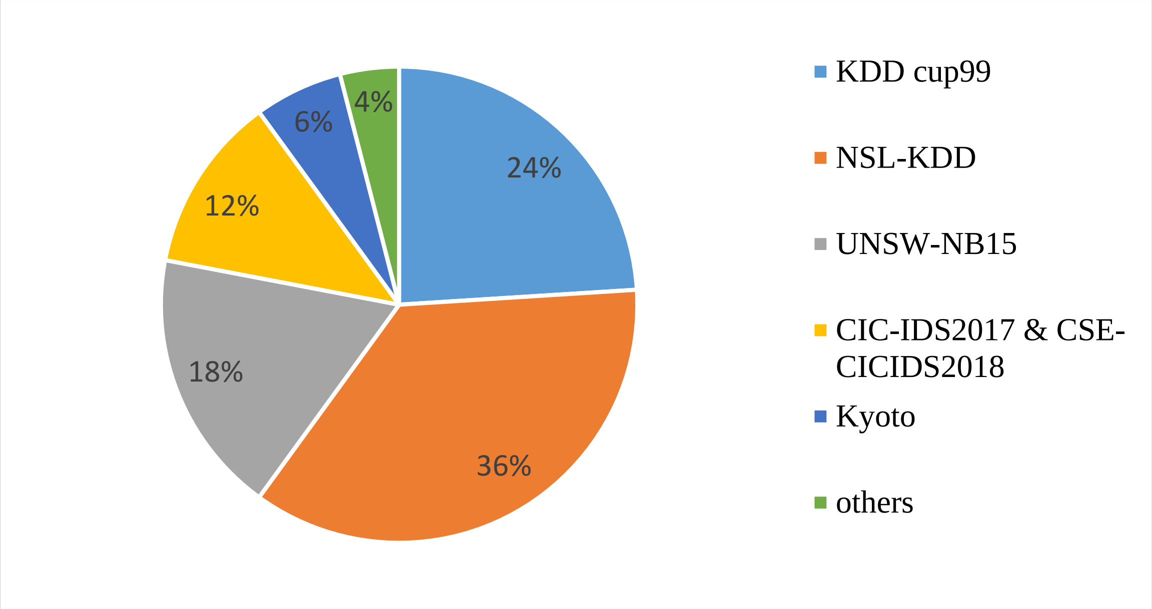


Figure (2.8): The Analysis for Using the Available Datasets [25]**.**

2.6.1 The NSL-KDD Dataset [53]

NSL-KDD dataset was enhanced version of KDDCup'99 intrusion detection benchmark dataset to solve some of the inherent problems. Table (2.4) illustrates the distribution of instances in NSL-KDD dataset for training and testing [44]. Each instance in the dataset displays 41 continuous and discrete attributes (38 numerical and 3 symbolic), to more details [45, 46]. Basic features, content features, traffic features, and Host-based features are the four categories of features in NSL-KDD.

As a result, NSL-KDD avoids the problem of categorization bias. Duplicate and superfluous records were also removed using the NSL-KDD, reducing the overall volume to a reasonable level. As a result, the tests may be run on the entire dataset, and the results from other studies are similar and consistent. To some extent, the NSL-KDD solves the concerns of information loss and bias.

Table (2.4): Distribution of Instances in NSL-KDD Dataset for Training and Testing

Types of Attack	Number of instances	
	Training Dataset	**Test Dataset**
Normal	67343 (53.46%)	9711 (43.08%)
Denial of service (DoS)	45927 (36.46%)	7456 (33.07%)
Probe	11656 (9.25%)	2421 (10.74%)
Remote to local (R2L)	995 (0.79%)	2756 (12.22%)
User to root (U2R)	52 (0.04%)	200 (0.89%)
Total	125973	22544

1. Normal: data is normal did not any attacks.
2. DoS (Denial of Service): by using up bandwidth or overtaxing compute power, an attacker tries to stop authorized users from using network resources.
3. Probe: an unauthorized user attempts to learn more about the target system prior to initiating an attack.
4. R2L Remote to User. Sending a packet over a network to a target host, an attacker who does not have an account, uses various vulnerabilities to acquire entry as a user on that host.
5. U2R User to Root: if an attacker gets into the system, they begin out with a regular account. Then, they use system flaws to get root access to the system.

2.6.2 The UNSW-NB15 Dataset [54]

Mustafa and Slay at the Australian Centre of Cyber Security (ACCS) developed UNSW-NB15, one of the newest current intrusion detection benchmark datasets, to evaluate IDS. In the dataset, which was collected in January and February of 2015, there are more than 100 GB of real network traffic.

The UNSW-NB15 has 39 of the 42 features are numeric, while 3 are non-numeric (categorical) features. The five sets that make up the dataset are: (1) content features; (2) basic features; (3) time features; (4) flow features; and (5) other originated attributes [47]. To more details about the features of UNSW-NB15 datasets.

Samples of the listed network threats may be found in the UNSW-NB15: Backdoor, Shellcode, Reconnaissance, Worms, Fuzzers, DoS, Generic, Analysis, and Exploits. Additionally, table (2.5) gives information on each violence class's specifics and values distribution throughout the data subsets [47].

Table (2.5): UNSW-NB15 Instances Repartition

Attack Type	UNSW-NB15-Train	UNSW-NB15-Test
Normal	56,000 (31.94%)	37,000 (44.94%)
Generic	40,000 (22.81%)	18,871 (22.92%)
Exploits	33,393 (19.04%)	11,132 (13.52%)
Fuzzers	18,184 (10.37%)	6062 (7.36%)
DoS	12,264 (6.99%)	4089 (4.97%)
Reconnaissance	10,491 (5.98%)	3496 (4.25%)
Analysis	2000 (1.14%)	677 (0.82%)
Backdoor	1746 (1.00%)	583 (0.71%)
Shellcode	1133 (0.65%)	378 (0.46%)
Worms	130 (0.07%)	44 (0.05%)
Total	175,341	82,332

2.7 The Basic Principles

In this part, we provide a quick explanation of the basic principle behind our suggested model.

2.7.1 Activation Function

The activation function provides the nonlinear modeling capability of the network. ReLU (Rectified Linear Unit) is the most popular function [48]; it can keep the gradient from attenuating, thus effectively solving the issue of gradient disappearance; the function expression is as follows: the

ReLU activation function produces 0 as an output when x < 0 and produces a linear with slope of 1 when x > 0:

$$\hat{y}' = \max(0, x). \qquad (2.6)$$

2.7.2 Cross-Entropy Loss Function

Cross-entropy loss measures the effectiveness of a classification model whose output is a probability ratio between 0 and 1. As the anticipated probability departs from the true label, it rises. The cross-entropy loss in binary classification, where M = 2 classes, can be determined as follows:

$$\text{Loss} = -\,(y\log(p)) - (1-y)\log(1-p). \qquad (2.7)$$

Where y is binary indicator (0 or 1) and $\mathcal{P}$ is expected probability.

If M > 2 (i.e., multiclass classification), we calculate a separate loss for each class label per observation and sum the results:

$$\text{Loss}' = -\sum_{c=1}^{M} y_{o,c} \log(\mathcal{P}_{o,c}). \qquad (2.8)$$

Where y is binary indicator (0 or 1) if class label c is the correct classification for observation o and $\mathcal{P}$ is expected probability of the observed o is of class c.

2.7.3 Optimizer

We use Adam (Adaptive moments) optimizer to learn the network weight parameters. Independent adaptive learning rates are created for various parameters by computing the first-order moment prediction and the second-order moment prediction of the gradient. Empirical results prove that Adam has larger benefits over other optimizers in practice [48]. Moving averages of gradient:

$$m_t = \beta_1 m_t - 1 + (1-\beta_1) g_t \qquad (2.9)$$

And squared gradient:

$$v_t = \beta_2 v_t - 1 + (1-\beta_2)\, g^2{}_t. \qquad (2.10)$$

2.7.4. One Hot Encoding

As a simple and effective encoding technique, one-hot encoding is the most popular approach to handling numeralization of categorical attributes [43]. It can convert each value of categorical attributes to a binary vector, whose element with value 1 is only one, the others are zeros. The element with value 1 indicates the presence of corresponding possible value of a categorical attribute.

In NSL-KDD dataset every record has 41 attributes, containing 3 symbolic attributes and 38 continuous attributes. To easy process this features, it is necessary to transform symbolic attributes into attribute values [49], as example to encoded the attributes “protocol_type”, ‘‘service’’, and ‘‘flag’’, that include 3, 70, and 11 characters, respectively. Figure (2.9) illustrates this method [43]. As example the attribute “protocol_type” contains three types of characters ‘‘tcp’’, ‘‘udp’’ and ‘‘icmp’’, and their numeric values are ‘‘100’’, ‘‘010’’, and ‘‘001’’. After one hot encoding method each record contains 121 features.

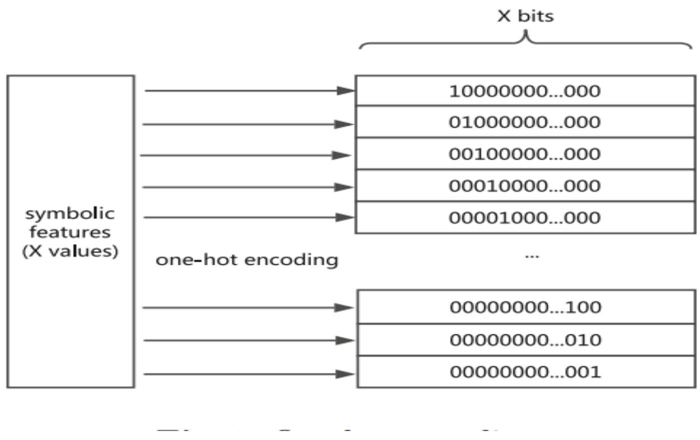


Figure (2.9): One Hot Encoding Method [43].

2.7.5. Normalization

A Normalize of both training and testing datasets is one of the important preprocessing steps. To minimize the effect the numeric scale of various characteristics for training sets. The value is scaled to [0, 1] via normalization, and the normalized value x' is

$$\acute{X} = (X - X_{min})/(X_{max} - X_{min}) \tag{2.11}$$

Where X is the starting value, X_{max} and X_{min} are the features heights and lowest values. The algorithm (2.1) illustrates the normalization of data.

<u>Algorithm (2.1): Normalization</u>

Input: dataset (DS)

Output: normalize DS

for i ← 1 to x do

  $\min_i$ ← find the minimum value of x ;

  $\max_i$ ← find the maximum value of x ;

end for;

for i ← 1 to size of DS do

  for j ← 1 to x do

    $x_{ij} \leftarrow \frac{x_{ij} - min_j}{max_j - min_j}$;

  end for;

end for;

2.8 Evaluation Metrics

Accuracy, or the percentage of correctly classified data, is a widely used performance evaluation measure. Table (2.6) provides an example of the confusion matrix using the terms true positive (TP), true negative (TN), false positive (FP), and false negative (FN).

The accuracy is as follows:

Accuracy = (TP + TN)/ (TP + TN + FP + FN) (2.12)

The following variables are also used to measure the model's performance because of the significant variation in record counts between categories in the NSL-KDD data set [30].

Recall or Detection Rate (DR) = TP/ (TP + FN) (2.13)

FPR or False Alarm Rate (FAR) = FP/ (FP + TN) (2.14)

$$\text{Precision} = TP / (TP + FP) \quad (2.15)$$

$$F - \text{score} = 2TP / (2TP + FN + FP) \quad (2.16)$$

Where the Recall is the percentage of accurately identified positives, also known as the true positive rate (TPR) or detection rate (DR); the percentage of negatives that are wrongly forecasted as positives is known as the false positive rate (FPR), often referred to as the False Alarm Rate (FAR); The F-score is the harmonic average of recall and precision, where precision is the percentage of expected to actual positives.

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters (TPR, FAR) [52].

Macro-averaged: all classes equally contribute to the final averaged metric. Weighted-averaged: each class contribution to the average is weighted by its size.

Table (2.6): Confusion Matrix.

Total population		**Predicted condition**	
		Normal	Anomaly
Actual condition	Normal	TN	FP
	Anomaly	FN	TP

2.9 Summary

This chapter has reviewed the overview that relates to intrusion detection systems, and the overview of the deep learning techniques based on intrusion detection systems. The important issues and challenges of existing proposed methods have been identified. Various studies in the fields were discussed in detail by addressing their methods, advantages, and disadvantages, as well as the summary. Then, a background and related work of Autoencoder and MLP methods were presented.

As well, the studies related have been discussed considering to achieve by these methods of high results. The final discussion shows several gaps in the field that provides the baseline for this research.

3.1 Introduction

The proposed work is discussed in this chapter. A hybrid model is proposed based Autoencoder and MLP methods that produce improvement of IDS performance.

This chapter is organized as follows: the details of the proposed model, the framework of the proposed model and the main phases, explain each phase in separate sections. Some figures, tables, and Algorithms will be used to illustrate the proposed model construction process.

3.2 Proposed Model

The framework of the proposed hybrid deep learning AE-MLP model is shown in figure (3.1). The figure illustrates the phases of the proposed model. We explain each phase in separate sections. But briefly, this structure is composed of The NSL-KDD and UNSW-NB15 datasets that be adopted in this thesis.

Preprocessing of data phases such as data cleaning, representation of data, and normalization of data to make the datasets more suitable for use by the proposed classifier. Propose a new model for IDS the hybrid AE-MLP model then implement and evaluate the experiment for this phase, evaluate and comparison of the proposed model with state of art methods then apply several performance criteria and validate the results with statistical tests.

Some figures, tables, and algorithms will be used to illustrate the system construction process.

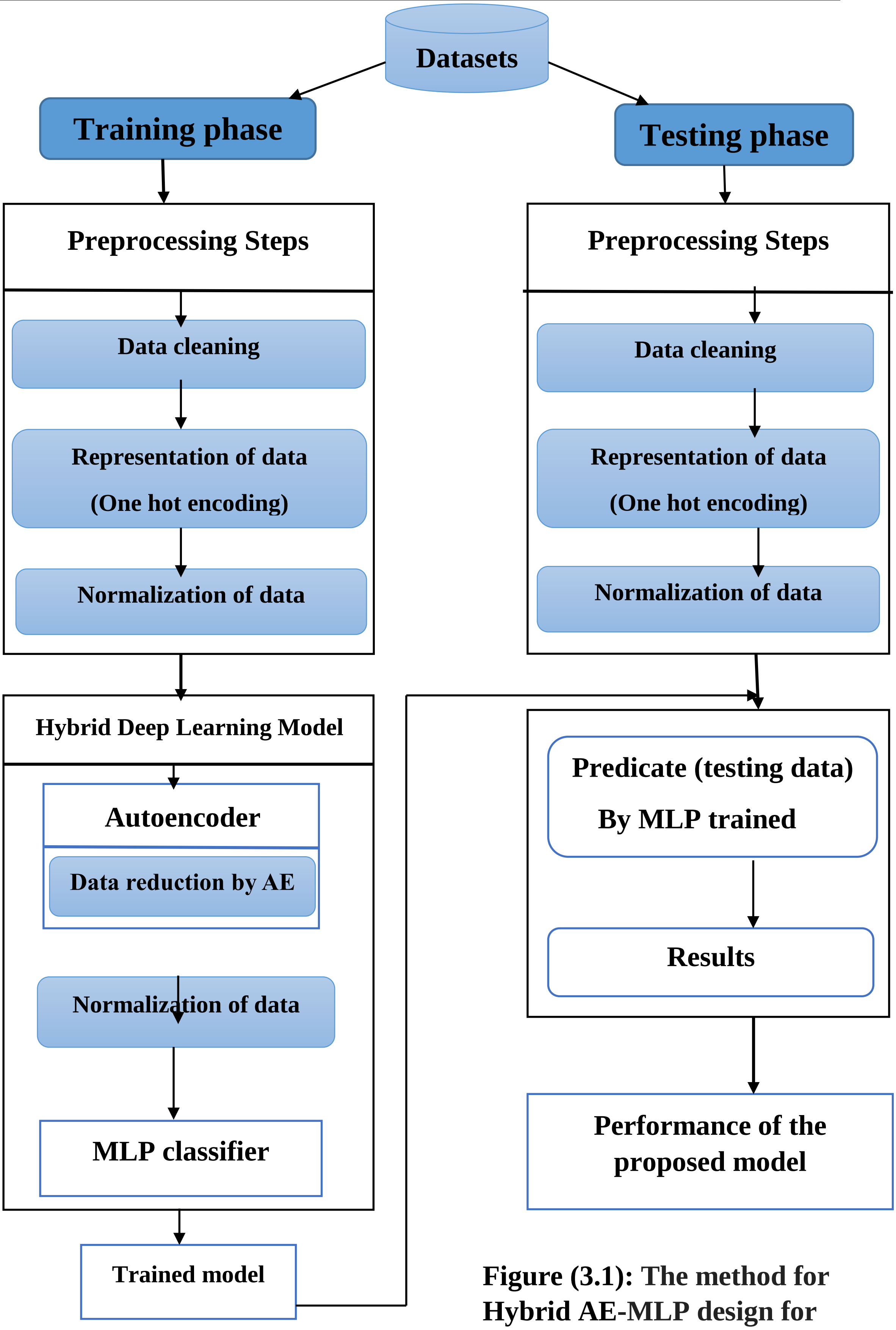


Figure (3.1): The method for Hybrid AE-MLP design for IDS.

3.2.1 Preprocessing of Data

The primary objective of this stage is to obtain appropriate data for the classification stage. The dataset pre-processing can be shown as including three stages: data cleaning, representation of data, and normalization of data discussed below.

I. Data Cleaning

To assure the quality of the data, data cleaning is a crucial data mining operation that is typically carried out prior to model training. Although the NSL-KDD data set has been updated, it has been discovered that "num outbound cmds," the twentieth attribute, often has a value of 0. As a result, this attribute is meaningless and should be eliminated. So, we eliminate this attribute from the dataset. Therefore each sample in the data collection has 40 attribute after data cleaning.

II. Representation of Data

In NSL-KDD dataset every record has 41 attributes, containing 3 symbolic attributes and 38 continuous attributes. To easy process this features, it is necessary to transform symbolic attributes into attribute values. We used the one hot encoding method to encoded the attributes “protocol_type”, ‘‘service’’, and ‘‘flag’’, that include 3, 70, and 11 characters, respectively. As discussed in chapter two.

After one hot encoding method each record contains 121 features. Table (3.1) below illustrates the NSL-KDD dataset after preprocessing phase and before the training phase, the number of features becomes 121 features in the training and testing data when applied the one hot encoding method and data cleaning.

Table (3.1): The NSL-KDD dataset after preprocessing of data.

Name	Type	Size	Value

C	str	13	Service_urh_i
Missing_cols	set	6	{‘servise_http_2784’, ‘service_harvest’, ‘service_http_8001’, ‘service…
Names	Index	**(121,)**	Index : 121 entries, duration to flag_SH
Scalar	Preprocessing._data.MinMaxScaler	1	MinMaxScaler object of sklearn.preprocessing._data module
Test	DataFrame	**(22544,121)**	Column names: duration, src_bytes, dst_bytes, land, wrong_fragment, ur…
Test_class	DataFrame	(22544,2)	Column names: 0,1
Train	DataFrame	**(125973,121**)	Column names: duration, src_bytes, dst_bytes, land, wrong_fragment, ur…
Train_class	DataFrame	(125973, 2)	Column names: 0,1

In UNSW-NB15 dataset every record has 41 attributes, containing 3 symbolic attributes and 38 continuous features. We also use one hot encoding method to encode the features ‘proto’, 'service', ‘state’ which contain 133, 13 and 9 characters respectively. After one hot encoding method each record contains 188 features.

Table (3.2) below illustrates the UNSW-NB15 dataset after preprocessing phase and before the training phase, the number of features in the training and testing data becomes 188 features when applied the one hot encoding method.

Table (3.2): The UNSW-NB15 dataset after preprocessing phase.

Name	Type	Size	Value
Missing_cols_in_test	Set	6	{‘state_URN’, ‘state_PAR’, ‘proto_icmp’, ‘state_ECO’, ‘pro …
Missing_cols_in_train	Set	2	{‘state_ACC’, ‘state_CLO’}
Names	Index	**(188,)**	Index : 188 entries, dur to state_RST
Scalar	Preprocessing._data.MinMaxScaler	1	MinMaxScaler object of sklearn.preprocessing._data module
Test	DataFrame	**(82332,188)**	Column names: dur, spkts, dpkts, sbytes, rate, sttl, dttl, slo …
Test_class	DataFrame	(82332,2)	Column names: 0,1
Train	DataFrame	**(175,341,188)**	Column names: dur, spkts, dpkts, sbytes, rate, sttl, dttl, slo …
Train_class	DataFrame	(175,341, 2)	Column names: 0,1

III. Normalization

Normalize training and testing datasets to minimize the effect of numeric of scale of various characteristics for training sets. In this study, the value is scaled to [0, 1] via normalization, the algorithm (2.1) illustrates the normalization of data and the details was discussed in chapter two.

We make two normalization in preprocessing of data phase, one after representation of data and the second after apply AE method.

3.2.2 Hybrid AE – MLP Proposed Method

In this thesis, a hybrid DL approach is proposed to effectively identify network IDS. This proposed model is develop a network IDS with a high accuracy, detection rate, low computational complexity and fewer false alarms rate.

Propose a new model Hybrid DL Model based on two techniques: Autoencoder (AE) and Multi-layer perceptron (MLP). Algorithm (3.4) illustrates the steps of the proposed model.

I. Autoencoder (AE)

The AE architecture has an input tier, a hidden tier, and an output tier. The input tier and the output tier are similar in size and the size of the hidden tier should be less than that of the input tier. A basic autoencoders objective is to rebuild the input data, which can reduce the number of the original features based on the number of features that we specify, as explained in chapter 2. The algorithm (3.1) illustrates the AE steps.

AE demonstrates better performance than other traditional machine learning classification techniques. The previous works [6,11,12,14] show that AE achieves good results versus other deep learning and machine learning techniques which are characterized by reducing the data are using and the learning time. AE is not computational costly, it can be applied to issues with binary classes and multiple classes. Moreover, AE can handle big training and testing datasets in few time [37]. So it is suitable for use as a robust classifier.

So, in the proposed work first, enter the original data on the AE and reduce the number of the original features based on the number of features that we specify. Then, the training phase was implemented on data and the new encoding data on training and testing data to predict the classification by using just the encoder stage (non-symmetric). This method called Nonsymmetric Autoencoder, which like AE structure but applying only the encoder stage (non-symmetric) after make the training dataset.

This entails the suggested change from the symmetric encoder-decoder architecture to relying solely on the encoder phase (Nonsymmetric). This is justified by the fact that, given the right ability tests, it is possible to minimize the impact on accuracy and efficiency while reducing computational and time overheads.

Algorithm (3.1): Autoencoder: training

Input: dataset $x = \{x_1, x_2, \ldots, x_m\}$ with m attributes
Output: trained AE
1: initialize parameters Ø, Θ, e, b
2: For i ← 1 to e do
3: For j ← 1 to b
4: // the input vector is transformed by the encoder xi to a hidden vector h_i through a deterministic transforming f_Θ
5: by equation (2.1)
6: // b is a bias vector, θ is the transforming variable set.
7: $\theta = \{w, b\}$
8: $f(x)$ = max (0, x),
9: If x > 0 the output is a straight line.
10: Else if x < 0 the output value is always zero.

```
            // the decoder return the hidden form that produced h_i to
                  a rebuilt a vector in the d-dimension y_i
              by equation (2.2)
              // calculate a loss function:
              by equation (2.3)
            // Adjusting the parameters of the encoder and decoder
              minimizes errors between the output data and the
              original data.
            by equation (2.4)
      end for;
end for;
return AE trained model.
```

II. Multi-Layer Perceptron (MLP).

A supplement of a feed-forward neural network. It consists of 3 types of tiers: the input tier, output tier, and hidden tier. The input tier receives the input signal to be processed, as explained in chapter 2. The required task such as prediction and classification is performed by the output tier. An arbitrary number of hidden tiers that are placed in between the input and output tiers are the true computational engine of the MLP.

Similar to a feed-forward network in an MLP the data flows in the forward direction from the input to the output tier. The neurons in the MLP are trained with the backpropagation learning algorithm. The algorithm (3.2) of MLP illustrates the steps.

The previous studies show that MLP also achieves high accuracy versus other deep learning and machine learning techniques. MLP is not computational costly, it can be applied to issues with binary classes and multiple classes. It is suitable for use as a robust classifier.

Used MLP Classifier, which is characterized by choosing the best epochs and the best batch size. Worked to train and test for the data, after training the autoencoder we took to the MLP classifier to classify the data into binary and multi class classification. The algorithm (3.2) illustrates the hybrid AE-MLP model steps. Figure (3.2) illustrates the hybrid of AE-MLP model.

Algorithm (3.2): MLP classifier

Input: the AE trained model

Output: the classification result

1: initialize parameters x, M, N, c

2: Choose a learning rate(x, c);

3: h_0 =x;

4: for M = 1 to N do

5: $g_M = n_M(hM-1) = W_M \times h_{M-1} + b_M$;

6: $h_M = \propto_M(g_M)$

7: end for;

8: return classification result.

Table (3.3) and table (3.4) shows part of the NSL-KDD data after applying the autoencoders model to rebuild the input data, it reduced the number of features in the training and testing data from 121 to 64 which can minimize the number of the original instances based on the number of features that we specify.

Table (3.3): The Number of Instances in NSL-KDD dataset after applied the AE method.

Name	Type	Size	Value
Missing_cols	Set	6	{'servise_http_2784', 'service_harvest', 'service_http_8001', 'service…
Names	Index	**(64,)**	Index : 64 entries, feature_0 to feature_63
Scalar	Preprocessing._data.MinMaxScaler	1	MinMaxScaler object of sklearn.preprocessing._data module

Reduced data	DataFrame	**(22544,64)**	Column names: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 …
Test	DataFrame	**(22544,64)**	Column names: feature_0, feature_1, feature_1, feature_2 …
Test_class	DataFrame	(22544,2)	Column names: 0,1
Train	DataFrame	**(125973,64**)	Column names: feature_0, feature_1, feature_1, feature_2 …
Train_class	DataFrame	(125973, 2)	Column names: 0,1

Table (3.4): The Part of Data in NSL-KDD dataset after applied AE model.

Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**
0	0.070595935	0.0	0.509116	0.65189064	0.8176483	0.6447177	**1.225056**
1	0.070595935	0.0	0.509116	0.65189064	0.8176483	0.6447177	**1.225056**
2	0.0	0.20918939	0.1363985	0.027176663	0.100837	0.06064854	0.0
3	0.0	0.059667535	0.23878935	0.24741846	0.29424486	0.48052326	0.0
4	0.0	0.31185228	0.0743429	0.0	0.0	0.0	0.0
5	0.0	0.48047942	0.06276462	0.0	0.0	0.018925441	0.0539698
6	0.0	0.19722195	0.049206033	0.15145475	0.0	0.19410554	0.20326233
7	0.0	0.3181411	0.17730913	0.16160911	0.0	0.12920398	0.03963999
8	0.0	0.5385625	0.0	0.0	0.0	0.0	0.009656033

Tables (3.5), (3,6) shows the number of features and the part of data in UNSW-NB15 dataset after reduced the original data by using Autoencoders model, which reduced the number of features in the training and testing data from 188 to 64 features and rebuild the new data.

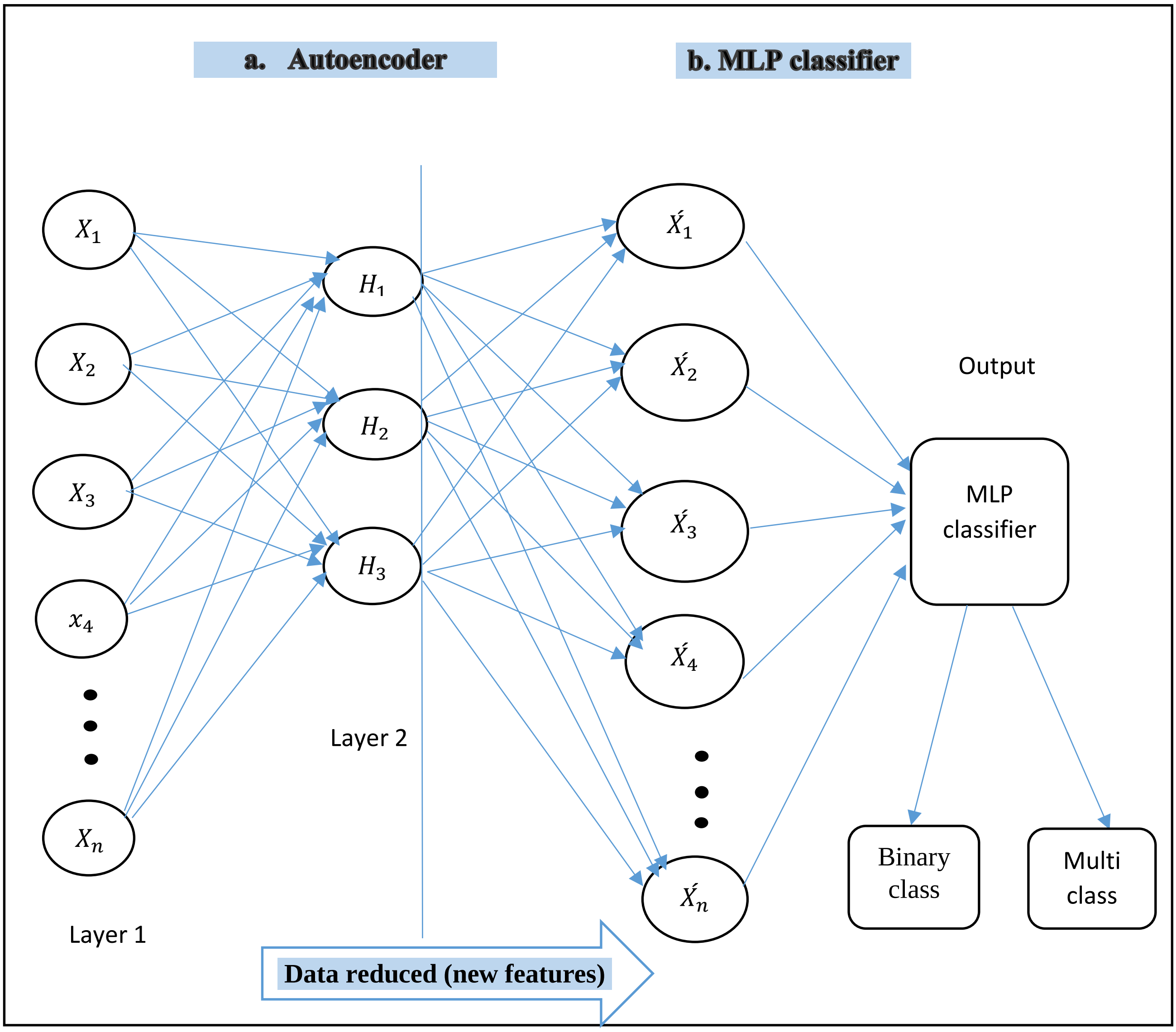


Figure 3.2: Hybrid DL (AE – MLP) model steps.

Table (3.5): The Number of Instances in UNSW-NB15 dataset after AE model.

Name	Type	Size	Value
Missing_cols _in_test	set	6	{'state_URN', 'state_PAR', 'proto_icmp', 'state_ECO', 'pro …
Names	Index	**(64,)**	Index : 64 entries, feature_0 to feature_63
Scalar	Preprocessin g._data.Min MaxScaler	1	MinMaxScaler object of sklearn.preprocessing._data module
Reduced data	DataFrame	**(82332,64)**	Column names: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13…
Test	DataFrame	**(82332,64)**	Column names: feature_0, feature_1, feature_2, feature_3 …

Test_class	DataFrame	(82332,2)	Column names: 0,1
Train	DataFrame	**(175,341,64)**	Column names: feature_0, feature_1, feature_2, feature_3 …
Train_class	DataFrame	(175,341, 2)	Column names: 0,1

Table (3.6): The Number of Instances in UNSW-NB15 dataset after reduced the data.

Index	0	1	2	3	4
13989	0.575487	0.057166062	0.884618	0.0	0.0
13990	0.74212414	0.0	0.7720365	0.0	0.0
13991	0.7733305	0.0	0.84058374	0.0	0.0
13992	0.8042454	0.2136108	**1.2882439**	0.0	0.0
13994	0.74212414	0.0	0.7720365	0.0	0.0
13995	0.62373847	0.16808325	**1.1967576**	0.0	0.0
13996	0.64883417	0.15879638	**1.167895**	0.0	0.0
13997	0.7733305	0.0	0.84058374	0.0	0.0

When reducing the number of features by using the AE model it produces new data we found that it needs to make normalization of data again to minimize the effect of the numerical range of various features and make it suitable for model training. Tables (3.7), (3.8) illustrates part of the data in NSL-KDD and UNSW-NB15 datasets after normalization.

Table (3.7): Part of NSL-KDD dataset after Normalization.

Index	Feature_0	Feature_1	Feature_2	Feature_3	Feature_4	Feature_5
0	0.14468609	0.0	0.6198586	0.6068301	0.42393652	0.8117917
1	0.06666248	0.0	0.6128985	0.51661474	0.5631415	0.42104
2	0.0	0.23888838	0.16420312	0.021537147	0.06944979	0.039607197
3	0.0	0.068138644	0.2874662	0.19607587	0.20265621	0.313811
4	0.0	0.35612652	0.08949759	0.0	0.0	0.0
5	0.0	0.54869395	0.075559095	0.0	0.0	0.012359468

Tables (3.8): Part of Data in UNSW-NB15 dataset after Normalization.

Index	0	1	2	3	4	5
0	0.19101687	0.078634545	0.041556083	0.09843846	0.0	0.0
1	0.21884051	0.104150735	0.07889659	0.108951256	0.0	0.0
2	0.175954	0.13608894	0.052961916	0.12283314	0.0	0.0
3	0.17626485	0.1325804	0.03885682	0.11511019	0.0	0.0
4	0.20780142	0.12931028	0.08895467	0.10845765	0.0	0.0

<u>Algorithm (3.3): The hybrid deep learning model based AE and MLP.</u>

Input: Training dataset NSL-KDDTrain+, Testing dataset NSL-KDDTest+, Training dataset UNSWTrain+, Testing dataset UNSWTest+;

Output: Classification results: Accuracy, Precision, DR, and F1-score;

```
Step 1: Data preprocessing
    data cleaning
    z2′, z3′, z4′=one−hot (z2, z3, z4);
    zf′=normalization (zf);
EndStep
Step 2: data reconstruction
    procedure code of AE train;
    Implemented the new encoding data on training and testing data to predict the classification by using just the encoder;
    zf′=normalization (zf); after training the AE;
EndStep
Step 3: Classification
    The weights of the latent tier of trained AE is used to initialize the weight of the MLP;
    Train the MLP classifier;
    Testing dataset NSL-KDDTest+ or UNSWTest+ are input into the trained AE-MLP classifier to detect attacks;
EndStep
Return the classification result.
```

3.2.3 Performance Evaluation

The popular benchmarks NSL-KDD and UNSW-NB15 datasets are employed to examine the suggested methods in all phases.

There are defined metrics for performance evaluation, such as accuracy, detection rate, false alarm rate, and the processing time of detection. IDS' primary objective is to reliably and promptly identify intrusions while using the fewest amounts of resources possible. Therefore, researchers are interested in the decrease in false-positive rates and the time needed to identify an incursion detection with increased detection rates.

Additionally, it takes time to evaluate and process the entire testing dataset, starting with preprocessing the data and ending with obtaining the final outcome of the detection of the typical attacks.

This study utilized two types of classification for IDS: binary and multi-classification. The binary is either normal or attack, but the multi is either normal or one of four categories of threat (DoS, Probe, R2L, U2R)in NSL-KDD and 9 categories of attack (Generic, Exploits, Fuzzers, DoS, Reconnaissance, Analysis, Backdoor, Shellcode, Worms) in UNSW-NB15 datasets.

In addition to determining the time metric, the interval time in milliseconds was obtained using the (time. time ()) python function. The test time is computed rather than the training time since the training phase only needs to be done once to develop the system, so its significance is not as great as the test time, which must be used each time the intrusion detection system is used. The time required for the test can be broken down into two parts: the time required for preprocessing, and the time required for classification.

3.3 Summary

This chapter presented the methodology of the research that acts as the significant point of the research. Four phases of research methodology that include a collection of datasets and pre-processing, propose hybrid AE and MLP classifier-based IDS, and performance evaluation was described in detail.

This chapter gives the datasets which are used to evaluate the performance of MAS-IDS. Then, the phase of proposing a model to improve the performance of IDS is subsequently discussed. This was followed by presenting the criteria that were used in measuring the performance of the proposed methods in this research.

4.1 Introduction

This chapter presents the experimental results concerning detecting and learning the unknown attacks and the proposed method's performance based on criteria like accuracy, precision, f-value and detection rate, false alarm rates, and testing time. Moreover, to confirm the model's performance, we compare it with other associated working techniques. The findings demonstrate that, in comparison to previous approaches, the suggested approach is effective at enhancing performance.

4.2 Experimental Setting

We implement examination to determine the effectiveness of the proposed hybrid AE-MLP. Our proposed was implemented in the python / ANACONDA / Spyder environment version 3.9.12, with processor 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 GHz and 8.00 GB RAM, 64-bit operating system, x64-based processor and Windows 10 Pro.

4.3 Experimental Results

A number of datasets have been produced to help in the evaluation and comparison process of intrusion detection systems. The NSL-KDD datasets are among these datasets that are used widely for evaluating the IDS's effectiveness and the UNSW-NB15 datasets which are one of the newest current intrusion detection benchmark datasets to evaluate IDS. Therefore, we implement experiments on the NSL-KDD and UNSW-NB15 datasets to evaluate the efficiency of the proposed model and examining the outcomes of the proposed hybrid AE-MLP model with commonly DL methods.

The data utilized in the evaluations are:

I. NSL-KDD dataset

In the field of intrusion detection, NSL-KDD are well-known dataset & Revathi study demonstrated that NSL-KDD data sets are appropriate for comparing various intrusion detection techniques [35]. Within this data set, every instance in the NSL-KDD dataset displays 41 continuous and discrete instances (38 numerical and 3 symbolic). The label mostly includes four categories of threat data and normal data (Dos, Probe, U2R, R2L).

During the experiments of this thesis, the training set composed of (125973) records and test set (22544) records in the NSL-KDD datasets is utilized as the model's training set and test set, in both.

The table (4.1) below show the number of attributes in the training and testing dataset in NSL-KDD before preprocessing of the data.

Table (4.1): Number of Features in the Training and Testing in NSL-KDD Dataset.

Name	Type	Size	Value
Test	Data Frame	(22544, 42)	Column names: duration, protocol_type, service, flag, src_bytes …
Train	Data Frame	(125973, 42)	Column names: duration, protocol_type, service, flag, src_bytes …

II. **UNSW-NB15 dataset**

The UNSW-NB15 dataset was produced by the Australian Centre for Cyber Security (ACCS) cyber security research group. The KDDCup99 and NSL-KDD datasets were determined to have problems, which were fixed by the UNSW-NB15. A partitioning of fully connected records is made up of (82337) test connection records and (175343) train connection records governed by 10 threats.

The partitioning dataset has 42 features with their parallel class labels which are normal and 9 different threats. The UNSW-NB15 has 39 of the 42 features are numeric, while 3 are non-numeric (categorical) features.

The table (4.2) below show the number of attributes in the training and testing dataset in UNSW-NB15 before preprocessing of the data.

Table (4.2): Number of Features in the Training and Testing in UNSW-NB15 Dataset.

Name	Type	Size	Value
Test	Data Frame	(82332, 43)	Column names: dur, proto, service, state, spkts, dpkts, sbytes, dbytes …
Train	Data Frame	(175341, 43)	Column names: dur, proto, service, state, spkts, dpkts, sbytes, dbytes …

4.3.1 Results by Using NSL-KDD Datasets

We implement tests on the NSL-KDD dataset to determine the efficiency of the proposed approach and examine the outcomes of the proposed hybrid AE-MLP with certain associated work techniques.

Each algorithm has a set of parameters whose values influence the classification performance. Optimizing these parameters is therefore an important step in improving detection accuracy. Tables (4.3), (4.4) illustrates the parameters in binary and multi-classification in this model.

In binary classification, the parameters of AE are: in the encoder, Dense 1 is 121 and Dense 2 is 64. In the decoder, Dense 1 is 64 and Dense 2 is 121 with epochs 10 and batch size 200. The optimizer of AE is Adam, and the activities per tier is ReLU. Additionally, softmax acts as the output tier's activation function. The parameters of MLP classifier are: [random_state = 1, learning_rate_init = 0.7, alpha = 0.008, epsilon = 1e-8].

In multi-classification, the parameters of AE are: in the encoder, Dense 1 is 121 and Dense 2 is 64. In the decoder, Dense 1 is 64 and Dense 2 is 121, with epochs 10 and batch size 500. The optimizer of AE is Adam, and the activities per tier is ReLU. Additionally, softmax acts as the output tier's activation function. The parameters of MLP classifier are: [random_state = 1, learning_rate_init = 0.08, alpha = 0.004, epsilon = 1e-4].

The optimal parameters to utilize in the AE and MLP algorithms are determined for every one of these parameters during the training phase using the trial-and-error technique.

Table (4.3): Training Parameters for 2 Classes, on the NSL-KDD Dataset.

Algorithms	**parameters**	**Values**
AE	Encoder \ Dense 1 Dense 2 Decoder \ Dense1 Dense 2	 121 64 64 121
	Hyper parameters	**Values**
	Epochs Batch size The optimizer Activation function of each layer Activation function of output layer	10 500 Adam RELU Softmax
MLP	Random state Learning rate Alpha Epsilon	1 0.7 0.008 Le-8

Table (4.4): Training parameters for 5 classes on the NSL-KDD dataset.

Algorithms	**parameters**	**Values**

AE	Encoder \ Dense 1 Dense 2 Decoder \ Dense1 Dense 2	 121 64 64 121
	Hyper parameters	**Values**
	Epochs Batch size The optimizer Activation function of each layer Activation function of output layer	10 500 Adam RELU Softmax
MLP	Random state Learning rate Alpha Epsilon	1 0.08 0.004 Le-4

4.3.1.1 Performance of this Strategy in Binary and Multiclass Classification

Depends on the NSL-KDD dataset, two sets of experiments have been created to determine the efficiency of the hybrid AE-MLP model in binary class and multi-class, respectively. Based on criteria like accuracy, precision, f-value and detection rate, we evaluate the model's performance. Lastly, based on the most recent study, we compare the efficiency of the hybrid AE-MLP model with some other methods.

The confusion matrix that was produced via the hybrid AE-MLP structure on the NSL-KDD dataset is displayed in Figures (4.1), (4.2). These shapes show the outcomes of the experiments of the hybrid AE-MLP approach for the 2-class and 5-class classifications, respectively. To better explain the confusion matrix's findings, consider that 9347 refers to the actual normal class, which is categorized as normal, 69 to the actual normal class, which is expected to be DoS, 6141 to the actual DoS class, which is classified or expected to be DoS class, and etc.

According to the experimental outcomes, the confusion matrix's diagonal is where the majority of the samples are located, demonstrating a very high level of overall classification performance. Therefore, the confusion matrix's main diagonal's numbers indicate that the actual classes are properly identified as identical actual classes.

Additionally, the percentile represents the ratio of the number of predictions to the total number of instances for a class such 96.52% is the predicted actual class, that is equivalent to 9347 on the overall number of normal instances in the testing collection, etc. Tables (4.5) and (4.6) shows the classification report in NSL-KDD dataset.

Table (4.7) shows the evaluation results of the hybrid AE-MLP model analyze the outcomes of the binary classification and multi-classification using the accuracy, precision, recall, and F1-score assessment indices.

As we can see, the suggested model's accuracy for binary classifications is 87.6%, its false alarm rate is 0.172, its testing time is 0.768 ms, and its precision, recall rate, and F1-score are each 90.80, 0.94, and 0.87. The accuracy of the suggested model is 81.06%, the false alarm rate is 0.1156, the testing duration is 0.647 ms, and the precision, recall rate, and F1-score are each 0.75, 0.96, and 0.84 in the multi-classification outcomes.

The effectiveness of the hybrid AE-MLP approach is enhanced and reaches 81.06% in five classes and 87.6% in two classes. This demonstrates that the suggested approach will work efficiently. Figure (4.3) displays the suggested approach's ROC curve for a binary class which ensures the superior performance of our suggested method.

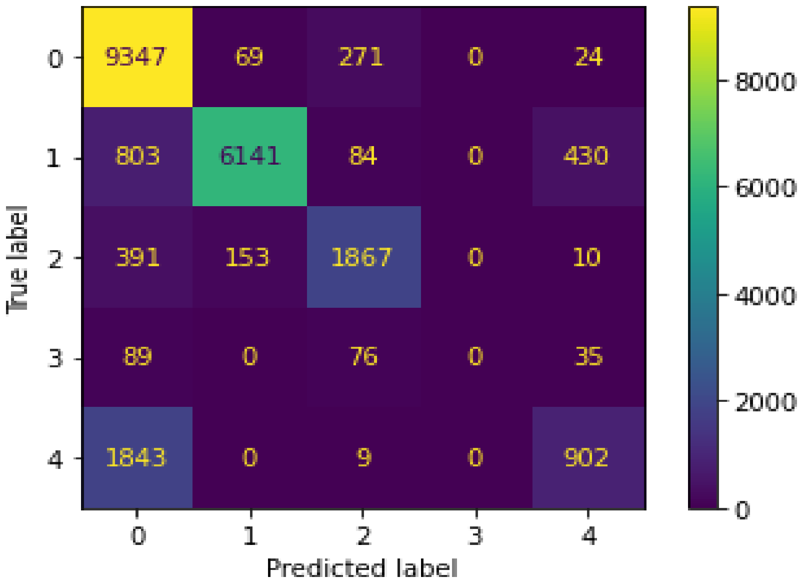


Figure (4.1): Multi-Class Confusion Matrix in the NSL-KDD dataset.

Table (4.5): Categorization Report the NSL-KDD dataset's Multi-Class.

	Precision	Recall or DR	f1-score	support
0	0.75	0.96	0.84	9711
1	0.97	0.82	0.89	7458
2	0.81	0.77	0.79	2421
3	0.00	0.00	0.00	200
4	0.64	0.33	0.43	2754
Accuracy		**0.81**		22544
Macro avg	0.63	0.58	0.59	22544
Weighted avg	0.81	0.81	0.79	22544

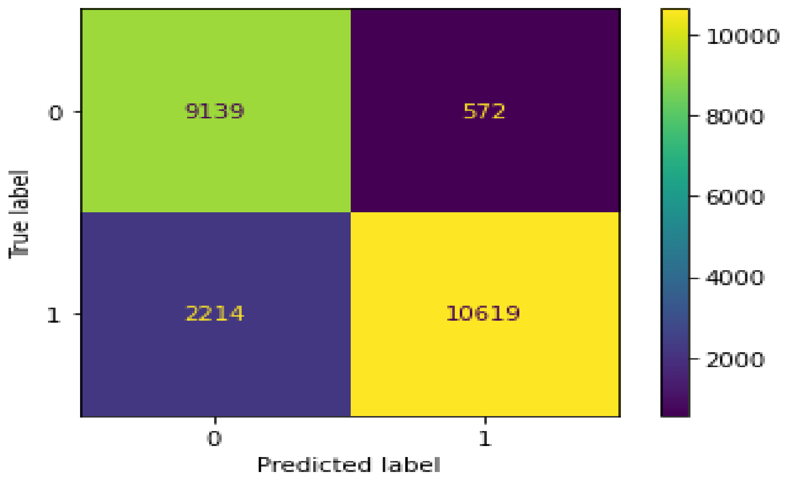


Figure (4.2): Binary Class Confusion Matrix in the NSL-KDD dataset.

Table (4.6): categorization report the NSL-KDD dataset's binary class.

	Precision	Recall or DR	f1-score	support
0	0.80	0.94	0.87	9711
1	0.95	0.83	0.88	12833
Accuracy	**0.88**			22544
Macro avg	0.88	0.88	0.88	22544
Weighted avg	0.89	0.88	0.88	22544

Table (4.7): metrics in 2 and 5-class NSL-KDD datasets.

Classification	Category	Accuracy	Precision (%)	Detection rate (%)	F-value (%)	FAR	Testing time
Multi-classification	Normal	96.52%	0.75	0.96	0.84		
	DoS	82.34%	0.97	0.82	0.89		
	U2R	77.11%	0.81	0.77	0.79		
	R2L	0.00	0.00	0.00	0.00		
	Probe	32.75%	0.64	0.33	0.43		
	Overall	**81.06%**	0.75	0.96	0.84	**0.115**	**0.647**
Binary classification	normal	94.10%	0.80	0.94	0.87		
	Attack	82.74%	0.95	0.83	0.88		
	Overall	**87.6 %**	0.80	0.94	0.87	**0.172**	**0.768**

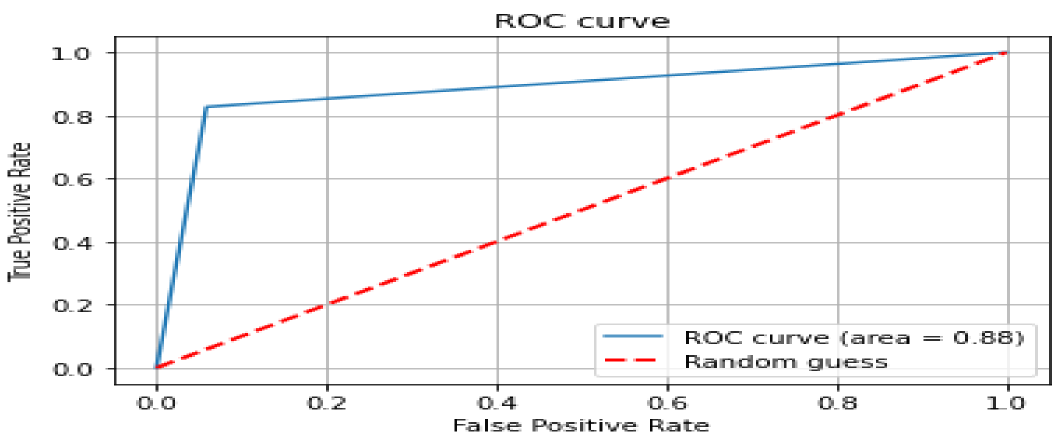


Figure (4.3): The ROC curve for binary class in NSL-KDD dataset.

4.3.1.2 Comparison with Other Methods

To demonstrate efficacy of the proposed approach, outcomes are compared with alternative techniques. It was evaluated against several research publications It utilized all NSL-KDD testing datasets and whose findings were based on them. The tests showed that the suggested methodology produced improved outcomes in classifications of five and binary classes.

Table (4.8) and figure (4.4) displays the contrast between the suggested method the hybrid AE-MLP and previously published techniques on NSL-KDDTest sets, and evaluates Outcomes from different assessment indices of accuracy, precision, recall, and F1-score include binary classification and multi-classification. We can observe that the accuracy of the hybrid AE-MLP model is higher than the other methods respectively. Out of these distinctions, we can observe that the proposed model can significantly increase the efficacy of intrusion detection.

We first applied each of the AE and MLP methods separately for each of the NSL-KDD datasets and achieved good results to show that the proposed method, which is a hybrid between these two methods, has been improved and is more efficient.

The accuracy rate in the Autoencoder method has reached 80.7% in multi-classification and 82% in binary classification, and the accuracy rate in the MLP classifier method achieved 80.4% in multi-classification and 84.56% for the NSL-KDD dataset. While the combination of these methods enhances the results and achieves high accuracy, the proposed model AE-MLP reaches 81.06% in multi-class and 87.6% with binary class.

So combining AE and MLP classifiers improves the results and achieves high accuracy.

The proposed model is a comparison with AE [6], this method obtained an accuracy rate of 79.47% in multi-class, were used the encoder of the deep Autoencoder without a decoder and used full NSL-KDD dataset. While the proposed model achieved better performance when applying the AE we obtained 80.7% and when applying the hybrid AE-MLP obtaining 81.06%.

The suggested strategy is compared with SAE+SVM [11], this technique resulted the accuracy rate and testing time was 84.96%, 4.648 (ms) and 80.48%, 5.395 (ms) in binary and multi-class respectively. Where it applied 10-fold cross-validation KDDTrain+ to evaluate the performance of the model. Although the suggested scheme improved performance by getting 87.6%, 0.647 (ms) and 81.06%, 0.768 (ms) of accuracy in binary and multi-class with a full NSL-KDD dataset.

CNN [7], which contains 11 tiers, 7 of which are convolutional and 4 completely linked and used full NSL-KDD dataset, obtained accuracy 79.05% in multi class and 85.07 in 2 class, it indicates a low accuracy in comparison to our developed approach.

In CNN-CNN [9] where CNN was used as a features extraction technique and then output from CNN was passed to a CNN classifier to be classified and used full NSL-KDD dataset, obtained accuracy 80.07% and 77.15 in binary and multi class classification. While the proposed model AE-MLP achieved better accuracy 81.06% and 87.6% in binary and multi classification.

The proposed model is comparison with (SAE) + SVM + the kernel approximation technique[14] which utilize a linear SVM to uniformly approximate to the Gaussian kernel SVM thanks to the training model's usage of SAE to achieve a lower in the attribute dimension, random Fourier attributes to accomplish kernel approximation and used full NSL-KDD dataset. This method obtained the accuracy rate 85.8% in 2 class which is lower accuracy when comparison with the proposed model achieved 87.6% in 2 class.

In [15] RNN with a variable number of hidden layers: (LSTM) + Gated Recurrent Unit (GRU) and used full NSL-KDD dataset. The accuracy rate and testing time was obtained 81.34%, 32.99 in 2 class which is lower accuracy when comparison with the proposed model achieved 87.6% in 2 class.

The outcomes show that the suggested methodology performs better than other approaches, especially in terms of accuracy.

Table (4.8): The comparison of the proposed approach with other methods in the NSL-KDD dataset.

Papers	Techniques	No. of instance	Testing time (ms)	Accuracy	Precision (%)	Detection rate (%)	F-value (%)
C. zhang et al., 2019, [6]	AE	Full	-	79.74% in multi classification	0.82	0.79	0.81
M. AL-QATF et al., 2018, [11]	SAE + SVM	10 cross validation	4.648 5.395	In 2 class 84.96% 80.48% in 5 class	96.23	76.56	85.28
S. Z. Lin et al., 2018, [7]	CNN	Full	-	79.05 in 5 class 85.07% in 2 class	91.68	81.12	86.08
G. Kamdem et al., 2019, [9]	CNN-CNN	Full	-	In 2 class 80.07% In 5 class 77.15%	80.07 75.15	85.00 74.14	-
Y. Wu et al., 2020, [14]	(JSAE-FSVM) Autoencoder (SAE) + SVM + the kernel approximation technique.	Full	-	85.8% in 2 class	0.919	0.785	0.847

Z. Li et al., 2019, [15]	RNN with a variable number of hidden layers: (LSTM) + Gated Recurrent Unit (GRU).	Full	32.99	81.34 in 2 class	-	-	81.99
Our proposed	**AE**	Full	1.044 0.745	80.7% in 5 class 82% in 2 class	-	-	-
	MLP	Full	0.525 0.215	80.4% in 5 class 84.56% in 2 class	-	-	-
	AE + MLP classifier	**Full**	**0.647** **0.768**	**81.06% in 5 class** **87.6 % in 2 class**	**0.80** **0.95**	**0.94** **0.83**	**0.87** **0.88**

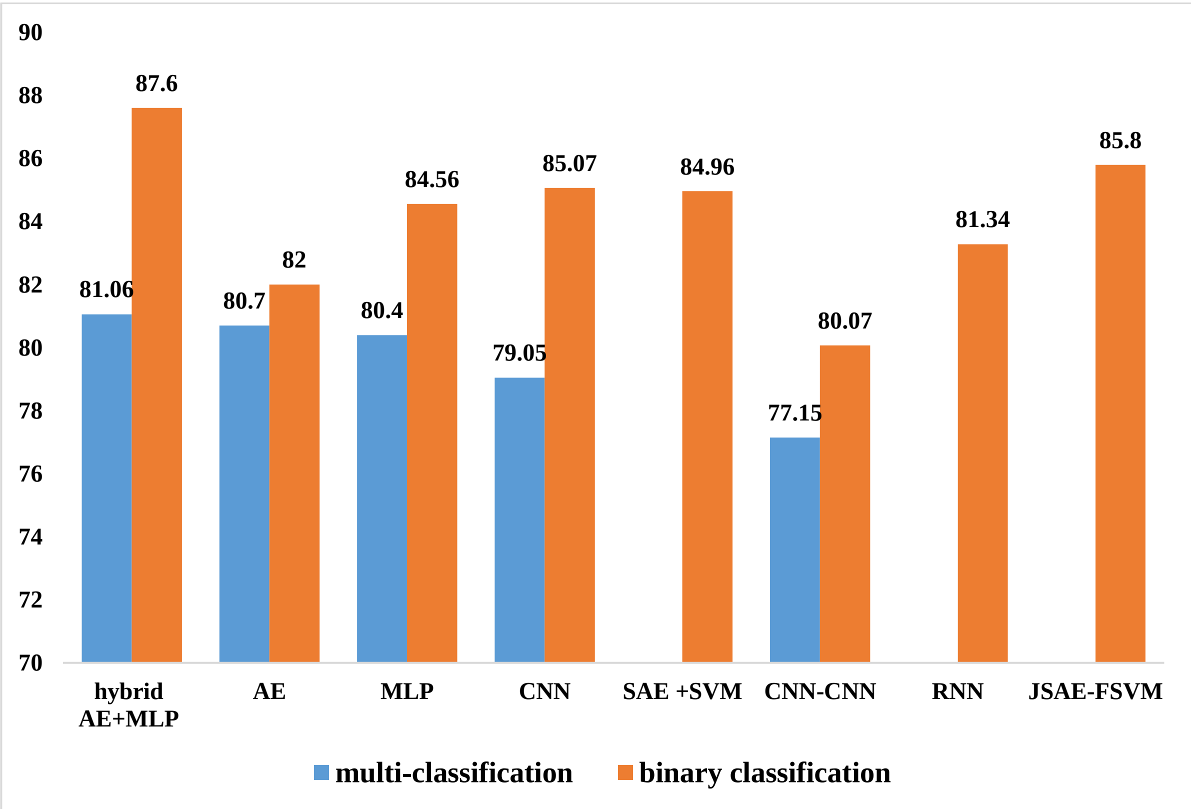


Figure (4.4): Comparisons of performance with other methods in the NSL-KDD dataset.

4.3.2 Results by Using UNSW-NB15 Datasets

We implement tests on the UNSW-NB15 dataset to determine the efficiency of the proposed approach and examine the outcomes of the proposed hybrid AE-MLP with some related work techniques. Tables (4.7) and (4.8) illustrate the parameters in binary and multi-classification in this model.

In multi-classification, the parameters of AE are: in the encoder the Dense 1 is 188 and dense 2 is 64, in decoder dense 1 is 64 and dense 2 is 188, with epochs 10 and batch size 100. The optimizer of AE is Adam, and the activities per tier is ReLU. Additionally, softmax acts as the output tier's activation function. The parameters of MLP classifier are: [random_state = 1, learning_rate_init = 0.09, alpha = 0.0009, epsilon = 1e-9].

In binary classification, the parameters of AE are: in the encoder the Dense 1 is 188 and dense 2 is 64, in decoder dense 1 is 64 and dense 2 is 188 with epochs 10 and batch size 500. The optimizer of AE is Adam, and the activities per tier is ReLU. Additionally, softmax acts as the output tier's activation function. The parameters of MLP classifier are: [random_state = 1, learning_rate_init = 0.09, alpha = 0.0009, epsilon = 1e-9].

The optimal parameters to utilize in the AE and MLP algorithms are determined for every one of these parameters during the training phase using the trial-and-error technique.

Table (4.7): Training Parameters for 2 Classes, on UNSW-NB15 Dataset.

Algorithms	**Parameters**	**Values**
AE	Encoder \ Dense 1 Dense 2 Decoder \ Dense1 Dense 2	 188 64 64 188

	Hyper parameters	Values
	Epochs Batch size The optimizer Activation function of each layer Activation function of output layer	10 500 Adam RELU Softmax
MLP	Random state Learning rate Alpha Epsilon	1 0.09 0.0009 Le-9

Table (4.8): Training Parameters in Multi Classification on UNSW-NB15 Dataset.

Algorithms	Parameters	Values
AE	Encoder \ Dense 1 Dense 2 Decoder \ Dense1 Dense 2	 188 64 64 188
	Hyper parameters	**Values**
	Epochs Batch size The optimizer Activation function of each layer Activation function of output layer	10 100 Adam RELU Softmax
MLP	Random state Learning rate Alpha Epsilon	1 0.09 0.0009 Le-9

4.3.2.1 Performance of this Strategy in Binary and Multiclass classification

Depends on the UNSW-NB15 dataset, two sets of experiments have been created to determine the efficiency of the hybrid AE-MLP model in binary class and multi-class, respectively. Based on criteria like accuracy, precision, f-value and detection rate, we evaluate the model's performance.

Lastly, based on the most recent study, we compare the efficiency of hybrid AE-MLP model with some other methods.

The confusion matrix that was produced via the hybrid AE-MLP structure on the UNSW-NB15 dataset is displayed in figures (4.5) and (4.6). These shapes show the outcomes of the experiments of the hybrid AE-MLP approach for the 2-class and multi-class classifications, respectively. To better explain the confusion matrix's findings, consider that 31380 refers to the actual normal class, which is categorized as normal, 21 to the actual normal class, which is expected to be Generic, and 17710 is the actual Generic class, which is classified or predicted as Generic class, and etc.

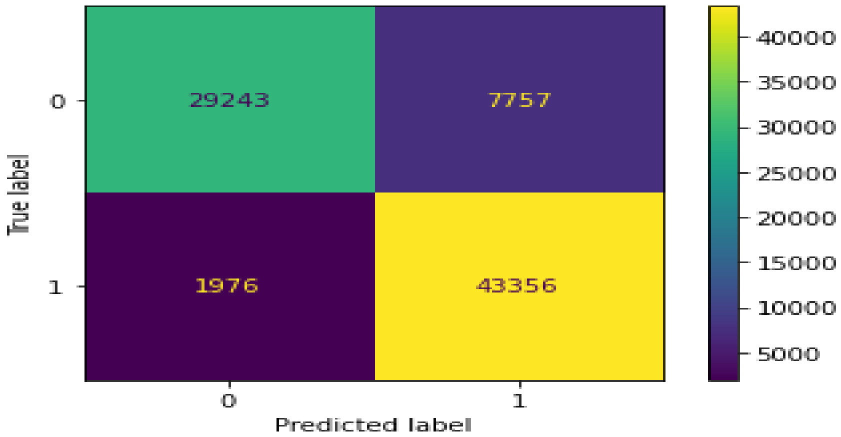


Figure (4.5): Binary Class Confusion Matrix in the UNSW-NB15 dataset.

Table (4.11): Categorization Report the UNSW-NB15 dataset's Binary Class.

	Precision	**Recall or DR**	**f1-score**	**Support**
0	0.94	0.79	0.86	37000
1	0.85	0.96	0.90	45332
Accuracy		**0.88**		**82332**
Macro avg	0.89	0.87	0.88	82332
Weighted avg	0.89	0.88	0.88	82332

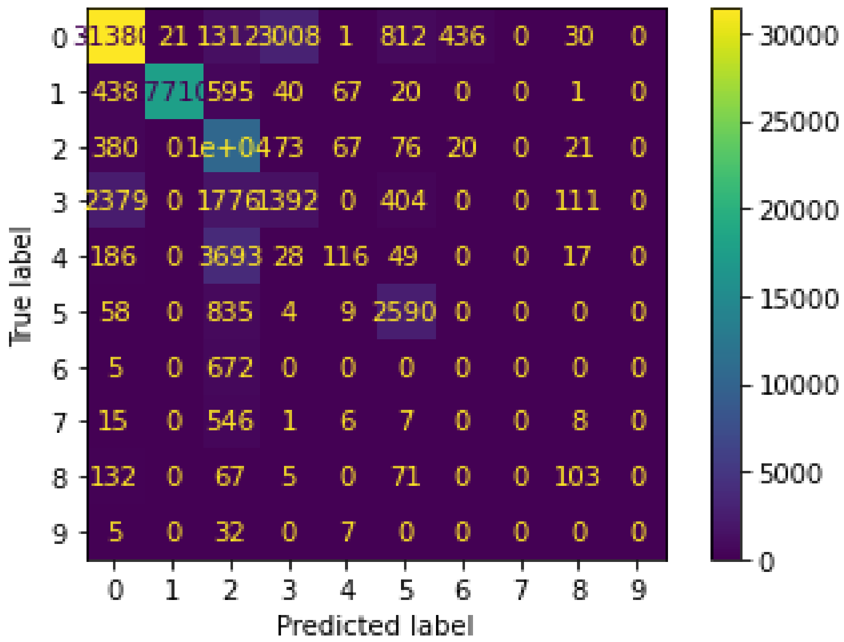


Figure (4.5): Multi-Class Confusion Matrix in the UNSW-NB15 dataset.

Table (4.9): The Classification Report for multi-class in UNSW-NB15 dataset.

	Precision	**Recall or DR**	**f1-score**	**Support**
0	0.90	0.85	0.87	37000
1	1.00	0.94	0.97	18871
2	0.52	0.94	0.67	11132
3	0.31	0.23	0.26	6062
4	0.42	0.03	0.05	4089
5	0.64	0.74	0.69	3496
6	0.00	0.00	0.00	677
7	0.00	0.00	0.00	583
8	0.35	0.27	0.31	378
9	0.00	0.00	0.00	44
Accuracy		**0.77**		**82332**
Macro avg	0.41	0.40	0.38	82332
Weighted avg	0.78	0.77	0.76	82332

According to the experimental outcomes, the confusion matrix's diagonal is where the majority of the samples are located, demonstrating a very high level of overall classification performance. Therefore, the confusion matrix's main diagonal's numbers indicate that the actual classes are properly identified as identical actual classes.

Table (4.10): Multi-Class Confusion Matrix in the UNSW-NB15 dataset.

	Predict										
Actual		Normal	Gener ic	Exploit -s	Fuzzer s	DoS	Reconna issance	Anal ysis	Back -door	Shell code	Wor ms
	Normal	31380	21	1312	3008	1	812	436	0	3 0	0
	Generic	438	17710	595	40	67	20	0	0	1	0
	Exploits	380	0	10495	73	67	76	20	0	21	0
	Fuzzers	2379	0	1776	1392	0	404	0	0	111	0
	DoS	186	0	3693	28	116	49	0	0	17	0
	Reconna issance	58	0	835	4	9	2590	0	0	0	0
	Analysis	5	0	672	0	0	0	0	0	0	0
	Backdo or	15	0	546	1	6	7	0	0	8	0
	Shellcod e	132	0	67	5	0	71	0	0	103	0
	Worms	5	0	32	0	7	0	0	0	0	0

Additionally, the percentile represents the ratio of the number of predictions to the total number of instances for a class such 90 % is the predicted normal class, which is equal to 31380 on the overall number of normal instances in the testing collection, etc.

Table (4.12) shows the evaluation results of the hybrid AE-MLP model evaluate the binary classification and multi-classification results from four evaluation indexes of accuracy, precision, recall, and F1-score.

We can observe that in the binary classifications, the accuracy of the proposed approach is 88.178 %, the False Alarm Rate is 0.0435 and testing time is 0.219 ms, and the precision, recall rate, and F1-score are 94%, 79%, and 86%, respectively. In the multi-classification results, we can also observe the outcomes, the accuracy of the proposed technique is 77.47%, the False Alarm Rate is 0.024 and testing time is 0.203 ms, the precision, recall rate, and F1-score are 0.90, 0.85, and 0.87 respectively.

The effectiveness of the hybrid AE-MLP approach is enhanced and reaches 77.47% in multi-classes and 88.178% in binary classes. This shows that the suggested approach can be efficient. Figure (4.7) demonstrates the ROC curve of the suggested technique for a binary class that guarantees the superior performance of our proposed technique.

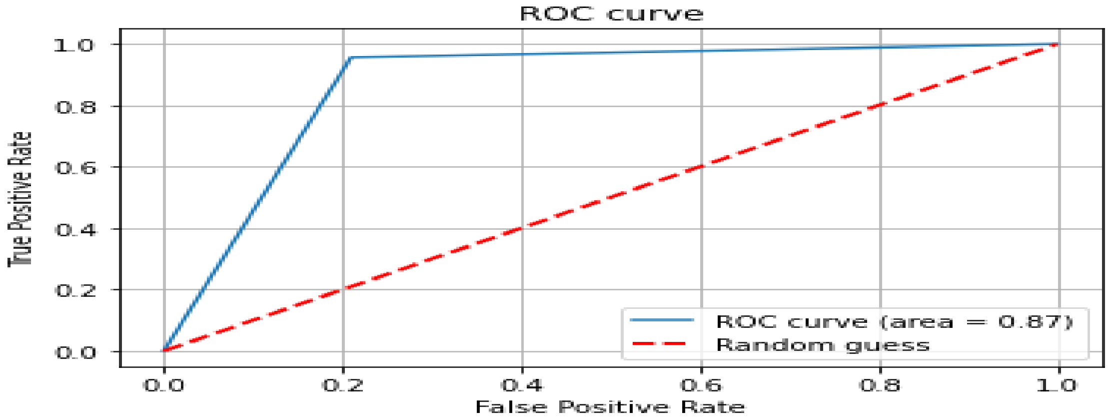


Figure (4.7): The ROC curve for binary class in UNSW-NB15 dataset.

Table (4.12): Metrics in binary and multi-class UNSW-NB15 datasets.

Classification	**Category**	**Accuracy (%)**	**Precision (%)**	**Detection rate (%)**	**F-value (%)**	**FAR**	**Testing time (ms)**
Multi-classification	Normal	84.81	0.90	0.85	0.87		
	Generic	93.84	1.00	0.94	0.97		
	Exploits	94.27	0.52	0.94	0.67		
	Fuzzers	22.96	0.31	0.23	0.26		
	DoS	02.83	0.42	0.03	0.05		
	Reconnaissance	74.08	0.64	0.74	0.69		
	Analysis	0	0.00	0.00	0.00		
	Backdoor	0	0.00	0.00	0.00		
	Shellcode	27.24	0.35	0.27	0.31		
	Worms	0	0.00	0.00	0.00		
	Overall	**77.47 %**				0.241	0.203
Binary classification	Normal	79.03	0.94	0.79	0.86		
	Attack	95.64	0.85	0.96	0.90		
	Overall	**88.178 %**				0.435	0.219

4.3.2.2 Comparison with Other Methods

To demonstrate efficacy of the proposed approach, outcomes are compared with alternative techniques. It was evaluated against several research publications It utilized all UNSW-NB15 testing datasets and whose findings were based on them. The tests showed that the suggested methodology produced improved outcomes in classifications of multi and binary classes.

Table (4.13) and figure (4.8) displays the contrast between the suggested method the hybrid AE-MLP and previously published techniques on UNSW-NB15 test sets, and evaluates outcomes from different assessment indices of accuracy, precision, recall, and F1-score include binary classification and multi-classification. We can observe that the accuracy of the hybrid AE-MLP model is higher than the other methods respectively. Out of these distinctions, we can observe that the proposed model can significantly increase the efficacy of intrusion detection.

We first applied each of the AE and MLP methods separately for the UNSW-NB15 datasets and achieved good results to show that the proposed method, which is a hybrid between these two methods, has been improved and is more efficient.

The accuracy rate in the AE method has reached 63% in multi-classification and 73.13% in binary classification, and the accuracy rate in the MLP classifier method achieved 75% in multi-classification and 82.35% in binary class for the UNSW-NB15 dataset. While the combination of these methods enhances the results and achieves high accuracy, the proposed model AE-MLP reaches 77.47% in multi-class and 88.178% with binary class.

So combining AE and MLP classifiers improves the results and achieves high accuracy.

The proposed model is a comparison with CNN +BiLSTM [21], this method obtained an accuracy rate of 77.16% in multi-class, , they create a deep network model using a convolution neural network (CNN) to obtain spatial characteristics and a bi-directional long short-term memory (BiLSTM) to extract temporal features and used full UNSW-NB15 dataset. While the proposed model achieved better performance when applying the hybrid AE-MLP obtaining 77.47%.

The suggested strategy is compared with Deep Belief Network (DBN) classifier [18], this method obtained the accuracy rate and testing time was 85.73% in binary class. Used full UNSW-NB15 dataset to evaluate the performance of the model. Although the suggested scheme improved performance by getting 88.178% of accuracy in binary classification with a full UNSW-NB15 dataset.

In [20] Depending on the RF classifier, the two-stage (TS) classifier architecture. Information Gain (IG) is a technique (IG-TS). Used full UNSW-NB15 dataset to evaluate the performance of the model, this method obtained an accuracy rate of 85.78 % in binary classification and the FAR is 15.64%. Although the suggested scheme improved performance by getting 88.178% of accuracy in binary classification with full UNSW-NB15 dataset and the FAR is 0.241.

In [19] the Advanced Principal Component (APCA) algorithm and the Extreme Learning Machine's (IELM) incremental approach. Used the full UNSW-NB15 and obtained the accuracy rate of 70.51% in multi class. The FAR is 0.3509 and testing time is 476.1880. While the proposed system achieved a better performance by obtaining 77.47% of accuracy in multi class classification with full UNSW-NB15 dataset and the FAR is 0.241and the testing time is 0.203.

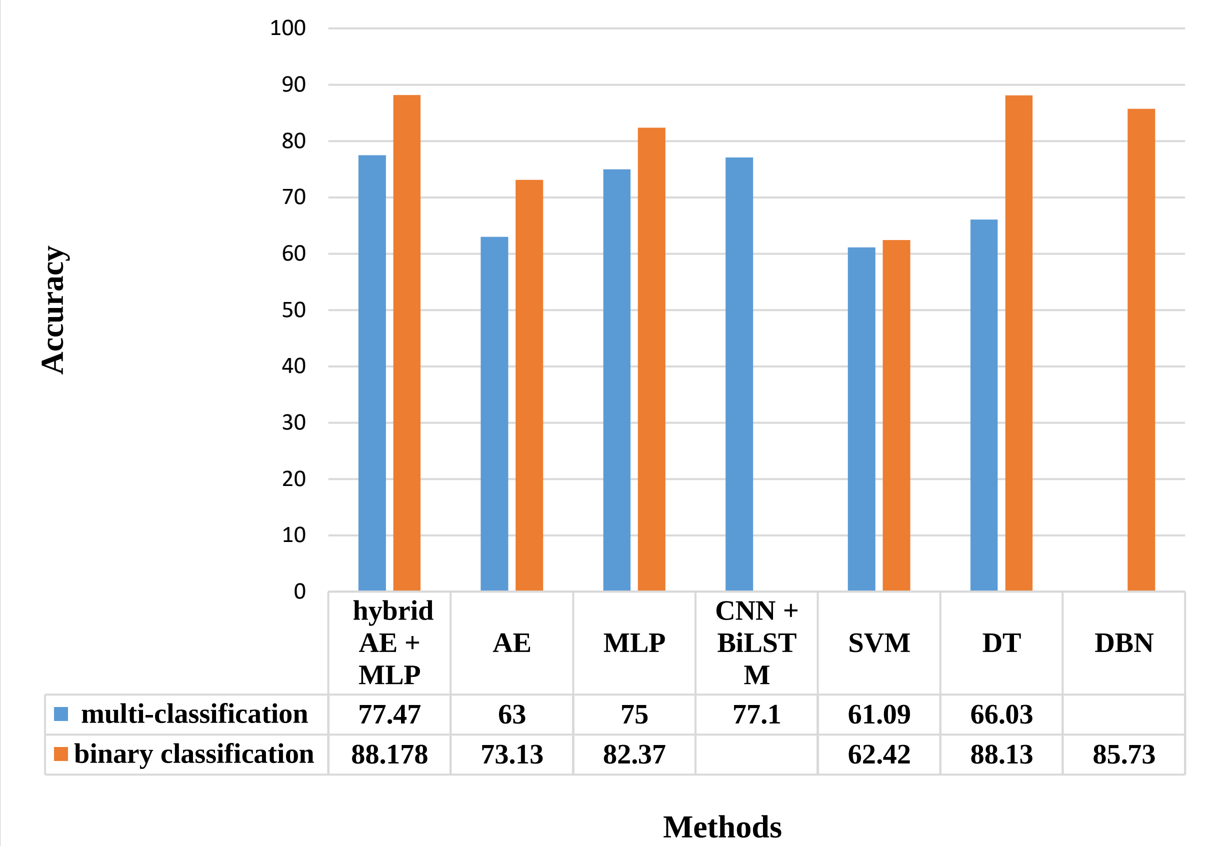
The proposed model is a comparison with [17] used ANN, LR, KNN, SVM, and DT methods and obtained the accuracy rate of 75.62, 65.53, 70.09, 61.09, 66.03 respectively in multi class, and in binary class 86.71, 79.59, 83.18, 62.42, 88.13 respectively. Used UNSW-NB15 dataset (divide the UNSW-NB15-TRAIN into the two sections below: UNSW-NB15-TRAIN-1 (representing 75% of the entire training set) for training additionally, the UNSW-NB15-VAL (25% of the total training set) was used for validation prior to testing. Although the suggested system showed improvement by getting 77.47% and 88.178% of accuracy in multi class and binary class classification with full UNSW-NB15 dataset.

The outcomes show that the suggested model performs better than other approaches, especially in terms of accuracy.

Table (4.13): The Comparison of the Proposed Model with other methods in the UNSW-NB15 dataset.

Papers	**Techniques**	**Accuracy**	**Precision (%)**	**Recall (%)**	**F-value (%)**	**FAR**	**Testing time (ms)**
K. JIANG et al., 2020 [21]	CNN +BiLSTM	77.16 in multi class	82.63	79.91	81.25	-	-
Sydney M. Kasongo & Yanxia Sun , 2020 [17]	ANN LR KNN SVM DT	In multi class 75.62 65.53 70.09 61.09 66.03 In binary class 86.71 79.59 83.18 62.42 88.13	79.92 76.91 75.79 47.47 79.82 78.56 73.18 80.31 58.89 80.33	75.61 65.54 70.21 62.00 66.04 98.53 93.74 95.09 95.88 98.38	76.58 66.62 72.03 53.77 51.12 87.42 82.20 87.08 72.97 88.45	-	-
Ahmad S. Almogren, 2020, [18]	DBN classifier	85.73% in binary classification.	-	-	-	-	-
Gao et al. , 2019, [19]	Incremental Extreme Learning Machine (IELM) +	70.51% in multi class.	-	-	0.7736	0.3509	476.1880

	an Advanced Principal Component (APCA)						
Zong et al. 2018, [20]	Two-Stage (TS) classifier model based on the RF classifier.	85.78 % in binary classification.	-	-	-	15.64% .	-
Our proposed	**AE**	63% in multi-class 73.13% in binary class	**-**	**-**	**-**	**-**	2.492 2.331
	MLP	75% in multi-class 82.35% in binary class	**-**	**-**	**-**	**-**	0.498 0.503
	AE + MLP classifier	**77.47% in multi class 88.178 % in binary class**	**0.90** **0.94**	**0.85** **0.79**	**0.87** **0.86**	**0.241** **0.435**	**0.203** **0.219**



	hybrid AE + MLP	AE	MLP	CNN + BiLSTM	SVM	DT	DBN
multi-classification	77.47	63	75	77.1	61.09	66.03	
binary classification	88.178	73.13	82.37		62.42	88.13	85.73

Figure (4.8): Comparisons of performance with other methods in the UNSW-NB15 dataset.

5.1 Introduction

This chapter summarizes the conclusions that result from the proposed work in the DL approach to IDS, the future works that can improve the existing research domain.

5.2 The Conclusions

Several important conclusions have been obtained as a result of building Intrusion Detection System.

- The AE was designated to process the training dataset and reduce the number of the original features based on the number of features we specify which can improve the overall performance of hybrid AE and MLP, and reduced processing time. Experiment results demonstrated that the new features are effective and achieved more accurate results in classification.
- A general framework is proposed that utilizes an MLP classifier after used AE that is used to perform the final classification. The final results showed the superiority of this approach in terms of efficiency and accuracy compared to individual classifiers.
- Preprocessing training and testing data is an essential step in the proposed work to assure the quality of the data. Where data cleaning is a crucial data mining operation that is typically carried out prior to model training. To easily process these features, it is necessary to transform symbolic attributes into attribute values. Finally, use normalized to minimize the effect of the numeric scale of various characteristics for training sets.
- The performance of the suggested algorithm was evaluated using the whole NSL-KDD and UNSW-NB15 training and testing datasets for

binary and multi-class classification, and it was compared to other approaches that also employed these datasets, like CNN, AE, MLP, RNN, SVM, and SAE-SVM. The findings indicate that our methodology sped up testing times and outperformed the majority of the earlier techniques in terms of performance measures for binary and multi-class classification.

5.3 Future Works

We believe that the suggested approach in this research can advance the existing research domain because there is still room for development to obtain better accuracy for IDS.

- We aim to research more practical techniques like CNN or LSTM to raise IDS performance in subsequent works.
- We hope to research more AE types. Such as a variational or sparse Autoencoder to improve the accuracy of detection.
- We hope to implement the proposed model in real time.

References

[1] Y. Wu, D. Wei, and J. Feng, “Network Attacks Detection Methods Based on Deep Learning Techniques: A Survey”, Security and Communication Networks Volume 2020, Article ID 8872923, 2020.

[2] G. Kocher1 and G. Kumar2, “PERFORMANCE ANALYSIS OF MACHINE LEARNING CLASSIFIERS FOR INTRUSION DETECTION USING UNSW-NB15 DATASET”, pp. 31-40, 2020.

[3] H. Liu * and B. Lang, “Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey”, Appl. Sci. 2019, 9, 4396; doi: 10.3390/app9204396, 2019.

[4] Y. Yan, L. Qi, J. Wang, Y. Lin, L. Chen, “A Network Intrusion Detection Method Based on Stacked Autoencoder and LSTM”, IEEE, Dublin, Ireland, 07-11 June 2020.

[5] O. Avci, O. Abdeljaber, S. Kiranyaz, M. Hussein, M. Gabbouj, D.J. Inman, “A review of vibration-based damage detection in civil structures: From traditional methods to Machine Learning and Deep Learning applications”. Mech. Syst. Signal Process. 2021, 147, 107077. [CrossRef].

[6] C. Zhang, F. Ruan, L. Yin, X. Chen, L. Zhai, F. Liu, “A Deep Learning Approach For Network Intrusion Detection based on NSL-KDD Dataset”, IEEE 13th International Conference on Anti-counterfeiting, Security, and Identification (ASID), Xiamen, China, 25-27 October 2019.

[7] S. Z. Lin, Y. Shi and Z. Xue, “Character-level Intrusion Detection Based on Convolutional Neural Networks”, International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 08-13 July 2018.

[8] K. WU, Z. CHEN, AND W. LI, “A Novel Intrusion Detection Model for a Massive Network Using Convolutional Neural Networks”, IEEE Access, 2169-3536, 2018.

[9] G. Kamdem, D. Teyou, J. Ziazet, “Convolutional Neural Network for Intrusion Detection System In Cyber Physical Systems”, arXiv: 1905.03168, 2019.

[10] B. A. TAMA, M. COMUZZI, AND K. RHEE, “TSE-IDS: A Two-Stage Classifier Ensemble for Intelligent Anomaly-Based Intrusion Detection System”, IEEE Access, 2169-3536, 2019.

[11] M. AL-QATF, Y. LASHENG, M. AL-HABIB, AND K. AL-SABAHI, “Deep Learning Approach Combining Sparse Autoencoder With SVM for Network Intrusion Detection”, IEEE Access, VOL. 6, 2169-3536, 2018.

[12] M. Yousefi-Azar, V. Varadharajan, L. Hamey and U. Tupakula, “Autoencoder-based Feature Learning for Cyber Security Applications” Faculty of Science and Engineering Macquarie University, Sydney, NSW, Australia, 2017.

[13] C. YIN, Y. ZHU, J. FEI, AND X. HE, “A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks”, IEEE Access, 2169-3536, 2017.

[14] Y. Wu, W. W. Lee, X. Gong and H. Wang, “A Hybrid Intrusion Detection Model Combining SAE with Kernel Approximation in Internet of Things”, MDPI Sensors, 20, 5710, doi:10.3390/s20195710, 2020.

[15] Z. Li, A. Rios, G. Xu, and L. Trajkovic´, “Machine Learning Techniques for Classifying Network Anomalies and Intrusions”, Canada under grant R31-611284, 2019.

[16] Z. Li, Z. Qin, K. Huang, X. Yang & S. Ye, “Intrusion Detection Using Convolutional Neural Networks for Representation Learning”, pp. 858–866, 2017.

[17] S. M. Kasongo & Y. Sun, “Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset”, 2020.

[18] A. S. Almogren, “Intrusion detection in Edge-of-Things computing”, Journal of Parallel and Distributed Computing, 2020.

[19] J. Gao, S. Chai, B. Zhang and Y. Xia, “Research on Network Intrusion Detection Based on Incremental Extreme Learning Machine and Adaptive Principal Component Analysis”, 2019.

[20] W. Zong, Y. W. Chow, W. Susilo, “A two-stage classifier approach for network intrusion detection”, 11125 329-340. Tokyo, Japan Information Security Practice and Experience: 14th International Conference, ISPEC, 2018.

[21] K. JIANG, W. WANG, A. WANG, AND H. WU, "Network Intrusion Detection Combined Hybrid Sampling With Deep Hierarchical Network", 2020.

[22] D. Jing, H. B. Chen, "SVM Based Network Intrusion Detection for the UNSW-NB15 Dataset", Computer Science, IEEE 13th International Conference on ASIC (ASICON), 2019.

[23] J. Liu, K. Xiao, L. Luo, Y. Li and L. Chen, "An intrusion detection system integrating network level intrusion detection and host-level intrusion detection", IEEE 20th International Conference on Software Quality, Reliability and Security (QRS), 2020.

[24] E. G. AbdAllah, M. Zulkernine, H. S. Hassanein, "Preventing unauthorized access in information centric networking", Security and Privacy 1, e33, 2018.

[25] Z. Ahmad, A. S. Khan, C. W. Shiang, J. Abdullah and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches", 2020.

[26] C. Tang, N. Luktarhan and Y. Zhao, "SAAE-DNN: Deep Learning Method on Intrusion Detection", *Symmetry*, *12*(10), 1695, 2020.

[27] H. Tun, S. Lupin, H. Htun Linn, K. Nay Zaw Lin, "Selection the Perimeter Protection Equipment in Security Systems", IEEE, 2018.

[28] A. Saxena, C. Mueller, "Intelligent Intrusion Detection in Computer Networks using Swarm Intelligence", 2018.

[29] G. Liu and J. Zhang, "CNID: Research of Network Intrusion Detection Based on Convolutional Neural Network", 2020.

[30] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[31] Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, Cambridge, MA, USA, 2016.

[32] Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Communications of the ACM, vol. 60, no. 2, pp. 2012–2025, 2012.

[33] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," IEEE Transactions on Audio, Speech, and Language Processing, vol. 20, no. 1, pp. 30–42, 2012.

[34] X. F. Xi and G. D. Ghou, “A survey on deep learning for natural language processing,” Acta Automatica Sinica, vol. 42, no. 10, pp. 1445–1465, 2016.
[35] Yirui Wu, Dabao Wei, and Jun Feng, 2020, Network Attacks Detection Methods Based on Deep Learning Techniques: A Survey.
[36] F. Farahnakian, J. Heikkonen, “A Deep Auto-Encoder based Approach for Intrusion Detection System”, 2018.
[37] Sang-Woong Lee a , Haval Mohammed sidqi b , Mokhtar Mohammadi c , Shima Rashidi d , Amir Masoud Rahmani, Mohammad Masdari g , Mehdi Hosseinzadeh a, “Towards secure intrusion detection systems using deep learning techniques: Comprehensive analysis and review”, 2021.
[38] Hyunseung Choi1, Mintae Kim1, Gyubok Lee1, Wooju Kim1, “Unsupervised learning approach for network intrusion detection system using autoencoders” , 2019.
[39] Hyunseung Choi1, Mintae Kim1, Gyubok Lee1, Wooju Kim1, “Unsupervised learning approach for network intrusion detection system using autoencoders”, 2019.
[40] Nathan Shone, Tran Nguyen Ngoc, Vu Dinh Phai, Qi Shi, “A Deep Learning Approach to Network Intrusion Detection”, 2017.
[41] Maxime Labonne, “Anomaly-based network intrusion detection using machine learning”, Cryptography and Security [cs.CR], Institut Polytechnique de Paris, 2020.
[42] Mehdi Moukhafi* and Khalid El Yassini, “Intelligent intrusion detection system using multilayer perceptron optimized by genetic algorithm”, 2020.
[43] Mohamed Amine Ferrag, Leandros Maglaras, Helge Janicke, Richard Smith, “Deep Learning Techniques for Cyber Security Intrusion Detection: A Detailed Analysis”, 2019.
[44] Vasudeva Pai , Devidas, Adesh N. D, “Comparative analysis of Machine Learning algorithms for Intrusion Detection”, 2021.
[45] Lin, S. w., Ying, K. c., Lee, C. y. & Lee, Z. j.,”An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection”, Applied Soft Computing Journal 12(10): 3285–3290, 2012.
[46] Somwang, P. & Lilakiatsakun, W., “Anomaly Traffic Detection Based on PCA and SFAM”, The International Arab Journal of Information Technology 12(3): 253–260, 2015.

[47] Sydney M. Kasongo & Yanxia Sun, "In Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset", 2020.
[48] Pengpeng Ding, Jinguo Li, Liangliang Wang, Mi Wen, and Yuyao Guan, "HYBRID-CNN: An Efficient Scheme for Abnormal Flow Detection in the SDN-Based Smart Grid", 2020.
[49] HAIXIA HOU1 , YINGYING XU 2 , MENGHAN CHEN2 , ZHI LIU (Member, IEEE), WEI GUO 3 , MINGCHENG GAO1 , YANG XIN 1 , AND LIZHEN CUI (Member, IEEE), "Hierarchical Long Short-Term Memory Network for Cyberattack Detection", 2020.
[50] B. Mahesh, Machine Learning Algorithms - A Review, International Journal of Science and Research (IJSR) ISSN: 2319-7064 Research Gate Impact Factor (2018): 0.28 | SJIF (2018): 7.426.
[51] Sarker, I.H., Kayes, A., Badsha, S., Alqahtani, H., Watters, P., Ng, A., "Cybersecurity data science: an overview from machine learning perspective", Journal of Big Data 7, 1–29, 2020.
[52] Classification: ROC Curve and AUC | Machine Learning, https://developers.google.com/machine-learning/crash course/classification/roc-and-auc.
[53] NSL-KDD99 Dataset. 2009. Available online: https://www. unb.ca/cic/datasets/nsl.html (accessed: 16.10.2019).
[54] UNSW-NB15 Dataset. 2015. Available online: https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15

Appendix A

Attack Types and Description for NSL-KDD Dataset.

Categories	Description	Examples
Normal	include data with no attack.	normal
Probe	include attacks in which attackers try to scan the computer networks with the purpose of collecting information and finding out vulnerabilities.	ipsweep, nmap, portsweep, satan, saint, mscan
DoS	include attacks in which attackers try to prevent legitimate users of a service from using the service	back, land, neptune, pod, smurf, teardrop, apache2, mailbomb, udpstorm, processtable
U2R	include attacks in which attackers have local access to the victim target machine and try to get super user privileges.	perl, rootkit, loadmodule, buffer_overflow, httptunnel, ps, sqlattack, xterm
R2L	include attacks in which attackers do not have a local account and try to gain access by sending packets to the target host over the internet.	ftp_write, guess_passwd, multihop, phf, imap, spy, warezclient, warezmaster, named, xsnoop, xlock, sendmail, worm, snmpgetattack, snmpguess

Attack Types and Description for UNSW-NB15 Dataset.

Sr. No.	Type	Description
1	Normal	Natural transaction data.
2	Analysis	An attack targets web applications through emails, ports, or web scripts.
3	Backdoor	Using backdoor to secure remote access.
4	DoS	Attacks computer memory
5	Exploits	An instruction that takes advantage of bugs/errors caused by unintentional behavior on the network.
6	Fuzzers	An attack to crash the system by inputting a lot of random data.
7	Generic	A technique to clash the block-cipher configuration by using hash functions.
8	Reconnaissance	A probe to evade network security controls by collecting relevant information.
9	Shellcode	Code is used to exploit software vulnerabilities.

Appendix B

Part of Data in NSL-KDD Dataset.

1	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_logins	logged_in	num_compromised	root_shell	su_attempted	num_root
2	0	tcp	ftp_data	SF	491	0	0	0	0	0	0	0	0	0	0	0
3	0	udp	other	SF	146	0	0	0	0	0	0	0	0	0	0	0
4	0	tcp	private	S0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	tcp	http	SF	232	8153	0	0	0	0	0	1	0	0	0	0
6	0	tcp	http	SF	199	420	0	0	0	0	0	1	0	0	0	0
7	0	tcp	private	REJ	0	0	0	0	0	0	0	0	0	0	0	0
8	0	tcp	private	S0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	tcp	private	S0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	tcp	remote_job	S0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	tcp	private	S0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	tcp	private	REJ	0	0	0	0	0	0	0	0	0	0	0	0
13	0	tcp	private	S0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	tcp	http	SF	287	2251	0	0	0	0	0	1	0	0	0	0
15	0	tcp	ftp_data	SF	334	0	0	0	0	0	0	1	0	0	0	0
16	0	tcp	name	S0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	tcp	netbios_ns	S0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	tcp	http	SF	300	13788	0	0	0	0	0	1	0	0	0	0
19	0	icmp	eco_i	SF	18	0	0	0	0	0	0	0	0	0	0	0
20	0	tcp	http	SF	233	616	0	0	0	0	0	1	0	0	0	0
21	0	tcp	http	SF	343	1178	0	0	0	0	0	1	0	0	0	0
22	0	tcp	mtp	S0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	tcp	private	S0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	tcp	http	SF	253	11905	0	0	0	0	0	1	0	0	0	0
25	5607	udp	other	SF	147	105	0	0	0	0	0	0	0	0	0	0
26	0	tcp	mtp	S0	0	0	0	0	0	0	0	0	0	0	0	0
27	507	tcp	telnet	SF	437	14421	0	0	0	0	0	1	3	0	0	0
28	0	tcp	private	S0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	tcp	http	SF	227	6588	0	0	0	0	0	1	0	0	0	0

Part of Data in UNSW-NB15 Dataset.

1	dur	proto	service	state	spkts	dpkts	sbytes	dbytes	rate	sttl	dttl	sload	dload	sloss	dloss	sinpkt	dinpkt	sjit	djit	swin	stcpb
2	0.121478	tcp	-	FIN	6	4	258	172	74.08749	252	254	14158.94238	8495.365234	0	0	24.2956	8.375	30.177547	11.830604	255	621772692
3	0.649902	tcp	-	FIN	14	38	734	42014	78.473372	62	252	8395.112305	503571.3125	2	17	49.915	15.432865	61.426934	1387.77833	255	1417884146
4	1.623129	tcp	-	FIN	8	16	364	13186	14.170161	62	252	1572.271851	60929.23047	1	6	231.875571	102.737203	17179.58686	11420.92623	255	2116150707
5	1.681642	tcp	ftp	FIN	12	12	628	770	13.677108	62	252	2740.178955	3358.62207	1	3	152.876547	90.235726	259.080172	4991.784669	255	1107119177
6	0.449454	tcp	-	FIN	10	6	534	268	33.373826	254	252	8561.499023	3987.059814	2	1	47.750333	75.659602	2415.837634	115.807	255	2436137549
7	0.380537	tcp	-	FIN	10	6	534	268	39.41798	254	252	10112.02539	4709.134766	2	1	39.928778	52.241	2223.730342	82.5505	255	3984155503
8	0.637109	tcp	-	FIN	10	8	534	354	26.683033	254	252	6039.783203	3892.58374	2	1	68.267778	81.137711	4286.82857	119.422719	255	1787309226
9	0.521584	tcp	-	FIN	10	8	534	354	32.593026	254	252	7377.527344	4754.74707	2	1	55.794	66.054141	3770.580726	118.962633	255	205985702
10	0.542905	tcp	-	FIN	10	8	534	354	31.313031	254	252	7087.796387	4568.018555	2	1	60.210889	68.109	4060.625597	106.611547	255	884094874
11	0.258687	tcp	-	FIN	10	6	534	268	57.985135	254	252	14875.12012	6927.291016	2	1	27.505111	39.106801	1413.686415	57.200395	255	3368447996
12	0.304853	tcp	-	FIN	12	6	4142	268	55.764583	254	252	99641.46875	5878.243164	3	1	25.948818	53.668801	1471.649189	80.404844	255	137150292
13	2.093085	tcp	smtp	FIN	62	28	56329	2212	42.520967	62	252	211825.125	8152.559082	28	8	34.312868	75.092445	3253.278833	106.113453	255	1824722662
14	0.416952	tcp	-	FIN	10	6	534	268	35.975363	254	252	9228.879883	4297.856445	2	1	45.088778	64.481199	2610.908343	99.860875	255	88408021
15	0.996221	tcp	-	FIN	10	8	564	354	17.064487	254	252	4079.416016	2489.407471	2	1	110.691222	131.48	6542.815197	202.433047	255	2321780530
16	0.576755	tcp	-	FIN	10	8	534	354	29.475254	254	252	6671.810547	4299.919434	2	1	64.083889	72.27157	4194.544962	116.493234	255	3772251972
17	0.000002	udp	snmp	INT	2	0	138	0	500000.0013	254	0	276000000	0	0	0	0.002	0	0	0	0	0
18	0.728252	tcp	-	FIN	10	6	534	268	20.597266	254	252	5283.885254	2460.686768	2	1	80.916889	122.462203	5333.802207	231.801797	255	1735938977
19	0.393556	tcp	http	FIN	10	8	860	1096	43.195886	62	252	15733.46582	19494.04883	2	2	43.728444	47.669145	2124.837873	71.805867	255	3882971404
20	0.387852	tcp	-	FIN	10	6	534	268	38.674545	254	252	9921.30957	4620.318848	2	1	38.119667	68.279203	2226.182762	105.440859	255	527350728
21	0.53784	tcp	-	FIN	10	8	534	354	31.607913	254	252	7154.543945	4611.036621	2	1	59.669444	68.351859	3893.395914	104.835672	255	1082242371
22	0.23372	tcp	-	FIN	10	6	534	268	64.179359	254	252	16464.14453	7667.293945	2	1	25.968889	40.812	1300.719343	65.724688	255	3154063438
23	0.338017	tcp	http	FIN	10	6	998	268	44.376468	254	252	21277.03711	5301.508789	2	1	35.218111	56.579801	1928.55071	82.022258	255	2665974075
24	0.964656	tcp	ftp	CON	14	12	690	950	25.915974	62	252	5315.884766	7223.300293	5	6	74.204308	59.700727	3869.615403	65.75518	255	996917599
25	0.449703	tcp	-	FIN	10	6	738	268	33.355347	254	252	11830.03027	3984.852051	2	1	42.611222	67.666	2352.173583	90.684992	255	3332877477
26	0.921025	tcp	-	FIN	10	6	534	268	16.286203	254	252	4177.954102	1945.658447	2	1	98.922556	170.070203	6572.489441	298.921438	255	3206068676
27	0.600982	tcp	-	FIN	10	8	534	354	28.287036	254	252	6402.854004	4126.57959	2	1	65.344556	78.634141	4216.405159	125.804047	255	27784615
28	0.568945	tcp	-	FIN	10	6	534	268	26.364588	254	252	6763.395508	3149.689453	2	1	59.910556	92.755203	2977.49563	131.706531	255	2091538399
29	0.497636	tcp	-	FIN	10	6	2074	268	30.142514	254	252	30013.90625	3601.025635	2	1	49.544111	83.069602	2540.823927	112.737664	255	3752199860

Publications

1. Duaa Al-Safaar, Wathiq Laftah Al-Yaseen, “Hybrid AE-MLP: Hybrid Deep Learning Model Based on Autoencoder and Multilayer Perceptron Model for Intrusion Detection System”, Received: October 26, 2022. Revised: December 5, 2022, International Journal of Intelligent Engineering and Systems, Vol.16, No.2, 2023.
2. Duaa Wahab Al-Safaar, Wathiq Laftah Al-Yaseen, “A survey of network intrusion detection systems based on deep learning approaches”. Scientific and Technical Journal of Information Technologies, Mechanics and Optics, 2023, vol. 23, no. 2, pp.

المستخلص

تلعب الشبكات دورًا مهمًا في الحياة المعاصرة. الأمن السيبراني ضروري لسلامة الإنترنت على الشبكة. يتتبع نظام الأمن السيبراني المعروف باسم نظام كشف التسلل (IDS) التهديدات غير المتوقعة للأجهزة والبرامج الموجودة على الشبكة ويكشف عنها. على الرغم من وجود العديد من أنظمة كشف التسلل ، إلا أنه لا تزال هناك صعوبات في زيادة دقة اكتشاف العيوب الأمنية ، ولا توجد طرق كافية لخفض مستوى اليقظة ، واكتشاف محاولات التسلل.

تعد فعالية التقنيات والأساليب المستخدمة لزيادة دقة التصنيف وتقليل أوقات التدريب والاختبار من المكونات الرئيسية لنجاح IDS.

تمت معالجة القضايا المذكورة أعلاه من قبل العديد من الباحثين الذين ركزوا على بناء IDS من خلال تقنيات التعلم العميق. يمكن لتقنيات التعلم العميق التمييز بين البيانات العادية وغير الطبيعية والتعرف على البيانات من التجارب السابقة. في طريقتنا المقترحة ، نقدم طريقة قوية للتعلم العميق تعتمد على طرق الترميزالتلقائي الهجين و MLP. تم تطوير الطريقة المقترحة باستخدام تقنية Autoencoder ، وهي طريقة ناجحة لإعادة البناء بدون إشراف لتمثيل الميزات الجديدة. بعد مرحلة ما قبل التدريب ، يتم إدخال البيانات الجديدة في مصنف MLP لزيادة دقة تصنيفها وكشف التسلل.

بالإضافة إلى ذلك ، تم تقييم أداء الطريقة المقترحة باستخدام مجموعات بيانات التدريب والاختبار NSL-KDD و UNSW-NB15 الكاملة للتصنيف الثنائي ومتعدد الفئات ، وتمت مقارنتها بالطرق الأخرى التي استخدمت أيضًا مجموعات البيانات هذه ، مثل AE، CNN، MLP و RNN و SVM و SAE-SVM. تشير النتائج إلى أن طريقتنا المقترحة سرّعت أوقات الاختبار وتفوقت على غالبية التقنيات السابقة من حيث مقاييس الأداء للتصنيف الثنائي والمتعدد الفئات. تم الحصول على معدل دقة هو 87.6٪ ، و 81.06٪ (في الفئات الثنائية والمتعددة) ، ووقت اختبار قدره 0.768 مللي ثانية و 0.647 ملي ثانية (في التصنيف الثنائي والمتعدد) لمجموعة بيانات NSL-KDD ، والحصول على 88.178٪ ، و 77.47٪ (في التصنيف الثنائي ومتعدد الفئات) ، وقت اختبار قدره 0.219 ملي ثانية و 0.203 ملي ثانية (في التصنيف الثنائي والمتعدد) لمجموعة بيانات UNSW-NB15. تعمل طريقة AE-MLP الهجينة المقترحة على تحسين اكتشاف التسلل وتقدم إستراتيجية جديدة لدراسات الكشف عن التسلل.

شكر وتقدير

احمد الله حمدا كثيرا طيبا مباركا على ماأكرمني به من اتمام هذه الرسالة ...

واتقدم بالشكر الجزيل والتقدير والامتنان لمشرفي الاستاذ الدكتور واثق لفته عبد علي الياسين الذي تفضل باشرافه لي

لتوجيهاته ونصائحه القيمة ومناقشاته المفيدة وعمقه، معرفته واهتمامه الشديد بالتفاصيل مصدر إلهام وأبقى عملي على المسار الصحيح في كل خطوة من هذه الرسالة.

يمتد التقدير لجميع اساتذتي الافاضل في قسم علوم الحاسبات شكرًا جزيلًا لكم، فأنتم من يحمل شعلة النجاح والتطور، فشكراً لكم ونتمنى لكم التوفيق والسداد دائمًا...

الاهداء

الى من سعى وشقى لأنعم بالراحة والهناء الى الذي لم يبخل بشيء من اجل دفعي في طريق السعادة

الى من علمني أن أرتقي سلم الحياة بحكمة وصبر ... الى روح الشهيد والدي العزيز

الى الينبوع الذي لايمل العطاء الى من حاكت سعادتي بخيوط منسوجة من قلبها والتي أكملت مسيرة الدرب بعد والدي الى من انتظرت هذه اللحظة بفارغ الصبر... والدتي العزيزة

الى من حبهم يجري في عروقي ويلهج بذكراهم قلبي الى مهجة فؤادي وبهجة سعادتي وانس حياتي ... اخوتي واخواتي

الى من حرص أن أرتقي بطموحاتي وعلمي وثقافتي وافتخر بنجاحي وانجازي ... زوجي و شريك حياتي

الى من حلّت بركة وجودهم في حياتي، ومن ملأت ضحكاتهم الجميلة عمري ... اطفالي الأحبة

الى من سرنا سويا ونحن نشق الطريق معا نحو النجاح والابداع ... صديقاتي وزميلاتي

أهدي هذا البحث.

وزارة التعليم العالي والبحث العلمي
جامعة بابل
كلية العلوم للبنات
قسم علوم الحاسوب

تحسين نظام كشف التسلل استنادًا إلى نموذج التعلم العميق الهجين

رسالة مقدمة إلى مجلس كلية العلوم للبنات - جامعة بابل وهي جزء من متطلبات نيل شهادة الماجستير في العلوم/علوم الحاسوب

مقدمة من قبل
دعاء وهاب رحيم الصفار

بإشراف
أ.م.د. واثق لفته عبد علي الياسين

2023 م   1444 هـ