COMPARISON OF INTRUSION DETECTION FOR THE INTERNET OF THINGS WITH MACHINE AND DEEP LEARNING METHODS

A THESIS SUBMITTED TO
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
KOCAELİ UNIVERSITY

BY

SIHAM AMAROUCHE

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

KOCAELİ 2021

COMPARISON OF INTRUSION DETECTION FOR THE INTERNET OF THINGS WITH MACHINE AND DEEP LEARNING METHODS

A THESIS SUBMITTED TO
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
KOCAELİ UNIVERSITY

BY

SIHAM AMAROUCHE

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

Prof.Dr. Kerem KÜÇÜK		
Supervisor,	**Kocaeli University**	**.....**
Prof.Dr. Adnan KAVAK		
Jury member,	**Kocaeli University**	**.....**
Assoc.Prof.Dr. Ali ÇALHAN		
Jury member,	**Düzce University**	**.....**

Thesis Defense Date: 25.06.2021

ACKNOWLEDGMENTS

I want to show appreciation to my research supervisor, Prof.Dr. Kerem Küçük for his support, guidance, understanding, and motivation during my master's studies. I am heartily thankful to my teacher who encouraged me to complete my thesis studies.

I extend my sincere thanks to my dear father Mohammed, my precious mother Khadija, my lovely sister Asmaa, as well as my brothers. I would like to express my sincere appreciation to them for their support, encouragement, prayers, and unconditional love for me during my education journey. I sincerely extend gratitude to my parents for their mental, financial, and spiritual support, thanks words cannot do it justice, may Allah reward them and bless them all with success, health, happiness, and strength.

May 2021 Siham AMAROUCHE

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

SYMBOLS AND ABBREVIATIONS

Abbreviations

AI	: Artificial Intelligence
AIDS	: Anomaly-based Intrusion Detection
ANN	: Artificial Neural Networks
CNN	: Convolutional Neural Networks
DA	: Deep Autoencoders
DBM	: Deep Boltzmann Machines
DBN	: Deep Belief Networks
DL	: Deep Learning
DNN	: Deep Neural Networks
DT	: Decision Tree
FAR	: False Alarm Rate
GAN	: Generative Adversarial Network
GRU	: Gated Recurrent Network
IDS	: Intrusion Detection System
IoT	: Internet of Things
IPS	: Intrusion Prevention System
kNN	: k-Nearest-Neighbor
LDA	: Linear Discriminant Analysis
LR	: Logistic Regression
LSTM	: Long Short-Term Memory
ML	: Machine Learning
MLP	: Multi-layer Perceptron
NB	: Naïve Bayes
NIDS	: Network Intrusion Detection
PCA	: Principal Component Analysis
RBM	: Restricted Boltzmann Machine
RF	: Random Forest
RNN	: Recurrent Neural Network
SDN	: Software Defined Network
SIDS	: Signature-based Intrusion Detection System
SIEM	: Security Information and Event Management
SVM	: Support Vector Machine
WSN	: Wireless Sensor Networks

MAKİNE VE DERİN ÖĞRENME YÖNTEMLERİ İLE NESNELERİN İNTERNETİ İÇİN SALDIRI TESPİTİNİN KARŞILAŞTIRILMASI

ÖZET

Saldırı tespiti ve siber güvenliği, güzümüzde nesnelerin interneti (IoT) alanında en önemli konularından biridir. Nesnelerin internetinde kablosuz ağlara bağlı nesnelerinin kullanımı yaygınlaşmasıyla birlikte ağ sistemleri üzerinden paylaştığımız veri miktarı hızla artmaktadır. Bu veriler saldırılara ve tehditlere karşı savunmasız olabilir ve sistemin gizliliğini, bütünlüğünü, kullanılabilirliğini ve güvenilirliğini artırmak için güvenliğini sağlamaları gerekir. Saldırılar daha karmaşık ve tespit edilmesi zor hale gelmektedir. İnsan kontrolüne veya manuel incelemeye ihtiyaç duymadan yapay zeka algoritmalarını kullanarak saldırıları otonom olarak tespit etme süreci, ağ saldırı tespit sistemlerinde (ASTS) trend konusu haline gelmiştir. Bu çalışmada, UNSW-NB15 açık veri kümesi üzerinde farklı klasik makine öğrenme (MÖ) ve derin öğrenme (DÖ) yöntemlerini uygulanmıştır. Derin öğrenme yöntemlerinde, özellik seçimi işlemine gerek kalmadan bu yöntemler doğrusal olmayan kombinasyonlar üretir, özelliklerin daha az etkisi olan, otomatik olarak daha az ağırlık alır, ancak DÖ yöntemleriyle aşırı öğrenme sorunu hala devam edebilir ve bunu çözmek için çapraz doğrulama, erken durdurma ve parametreleri ayarlama gibi farklı teknikler kullanılmıştır. IoT tabanlı anomali tespiti işlemine en iyi metodu bulmak için deneyler geliştirilmiş, farklı yapay zeka (YZ) modelleri arasında karşılaştırmalar yapılmış ve saldırı tespit sistemlerinde (STS) performansı iyileştirmek ve doğruluğu artırmak için yeni teknikler ve akıllı çözümler sunulmuştur.

Anahtar Kelimeler: Derin Öğrenme, IoT Güvenliği, Makine Öğrenmesi, Saldırı Tespiti, Siber Güvenlik.

COMPARISON OF INTRUSION DETECTION FOR THE INTERNET OF THINGS WITH MACHINE AND DEEP LEARNING METHODS

ABSTRACT

Intrusion detection and cyber security are important topics in the internet of things (IoT) domain nowadays. With the expansion of using objects that are connected to wireless networks in IoT, the amount of data that we share via network systems is growing rapidly. This data may be vulnerable to attacks and threats and need to secure it to increase the system's confidentiality, integrity, availability, and reliability. Attacks are becoming more complex and difficult to detect. The process of detecting attacks using artificial intelligence algorithms autonomously without the need for human control or manual examination has become a trend topic in network intrusion detection systems (NIDS). In this article, we decide to apply different classical machine learning (ML) and deep learning (DL) methods on UNSW-NB15 open dataset. In deep learning methods we exclude the need to feature selection these methods generate the non-linear combinations the features have less effect get lesser weights automatically, but the problem of overfitting with DL methods is still remaining and to solve it we used different techniques like cross-validation, early stopping, and parameters tuning techniques. We make experiments to find out the best way to identify the anomaly in IoT based environment, make comparisons between different artificial intelligence (AI) models and propose new techniques and smart solutions to improve performance and increase accuracy in intrusion detection systems (IDS).

Keywords: Deep Learning, IoT Security, Machine Learning, Intrusion Detection, Cyber Security.

INTRODUCTION

The Internet of Things (IoT) is a technology in which a network connects anything with the Internet, based on embedded systems, specific protocols, and sensors to conduct information exchange and communications in order to obtain smart recognitions, monitoring, localization, tracking, and control systems [1].

The sensitivity and importance of the information carried out by IoT devices and networks signifies the importance of its security. To overhead challenges and problems on the server end we used different models as a decision engine to decide about traffic data type, whether it is normal or malicious. Cyber threats have become more widespread and several new types of attacks have been generated targeting organizations, companies, and governments. Furthermore, since the IoT has emerged the number of devices and objects that are connected to wireless networks increased. The proposed research here is found on the intersection between intrusion detection and mitigation, and Artificial Intelligence (AI) technologies. To mitigate cyber-attack, cybersecurity analysts heavily depend on Intrusion Detection System (IDS). IDS can detect malicious activities by matching patterns of known attacks using the signature-based detection method, or observing anomaly activities using anomaly-based intrusion detection systems this method is introduced to detect unknown attacks [2].

Obviously, we can see that all governments and security intelligence try to protect their information and not allow spies to eavesdrop on it and its decisions. For the importance of cybersecurity topics, we research in this work the effectiveness of using ML and DL models in cybersecurity as well as current challenges that face security analysts and we aim to use different methodologies to prevent, mitigate attacks and drop the malicious packets and threats.

In our experimental process, over UNSW-NB15 dataset we applied the following supervised Machine Learning (ML) methods for IDS: Naïve Bayes (NB), k-Nearest-Neighbor (kNN), Logistic Regression (LR), Decision Tree (DT), and Random Forest (RF). Also Deep Learning (DL) methodologies such as Deep Neural Network (DNN),

Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), and CNN-LSTM model. We applied Random Forest algorithm over UNSW-NB15 dataset to calculate the feature importance measure for each feature, select more important features and generate reduced optimal feature vectors, this process may increase the accuracy of intrusion detection and increase the speed of models to get performance results. We considered two schemes binary and multiclass classification configurations. We implemented different hyperparameters such as epoch numbers, batch size values, and activation functions in hidden layers over the best-performing multiclass classification based deep learning model, to find the best hyperparameters that could be applied to improve our model's performance. We noticed the effect of parameters on the model's accuracy.

This thesis is organized as follows: Chapter 1 provides background, basic information about the internet of things and categorization of IoT attacks, IDS system, different types and techniques of intrusion detection system, comparison between IDS and IPS systems, big data with its analytic application, and cloud computing. Chapter 2 presents an overview of related work in intrusion detection systems. The chapter is divided into 4 subsections which are literature reviews on solutions for threat detection, machine learning-based applications in intrusion detection, deep learning-based applications in intrusion detection, and feature engineering and its effect on intrusion detection methods. Chapter 3 introduces technical details, gives a general review of IDS datasets provided in the literature, presents the UNSW-NB15 dataset which is used in our study, gives information about attack types on it, and also clarifies the technologies that are used. Chapter 4 elaborates on the traditional ML and DL methods used in this work. Chapter 5 discusses the evaluation and experimental results of classical ML and DL methods. Finally, the conclusion and future works are given in Chapter 6.

1. BACKGROUND

1.1. Internet of Things (IoT) and Categorization of IoT Threats

Internet of things (IoT) defines the network of things and objects that are embedded with technologies, programs, and sensors. IoT can be used in different fields such as healthcare, smart cities, autonomous vehicle technology, energy, industrial automation, building, supply chain, agriculture, etc. [3]. There were 8.4 billion IoT devices at the end of 2017, and this number is expected to rise every year [4]. In the case of IoT, data about a user's everyday life is gathered so that the "thing" in the user's environment can collaborate to provide better services that meet the user's personal needs. [5] As the data collected about a user moves across several hops in a network as a result of the diverse integration of devices, services, and networks, the information stored on a device is vulnerable to security breaches caused by nodes in an internet of things network being compromised [6].

From the aforementioned, we can see that intrusion detection and prevention play a critical role in cyber security for IoT applications. In Figure 1.1, different types of threats in the IoT environment are represented as below; IoT attacks that are shown based on three layers; physical layer, software layer, and network layer.

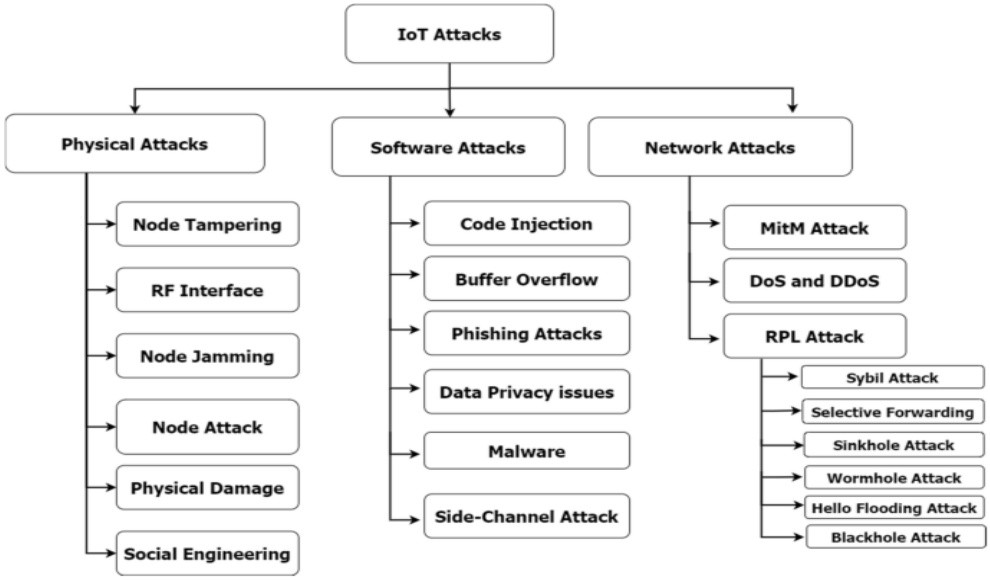


Figure 1. 1. Taxonomy of IoT attacks based on IoT system layers [7]

1.2. Intrusion Detection Systems (IDS)

1.2.1. Definition of intrusion detection system (IDS)

An intrusion detection system (IDS) is a network technology, device, or software application that monitors systems or network traffic for detecting intrusions, attacks, and malicious activity. Any intrusion activity or vulnerability infractions are reported to an administrator or collected centrally using a security information and event management (SIEM) system. A SIEM system collects data from many sources and employs alarm filtering techniques to discriminate between false alarms and intrusion activity[8]. CIA triad is an important component that is also known with its three fundamental concepts of information security which are; confidentiality, integrity, and availability. Malicious activity or intrusion is defined as any illegal activity that combines any of these cores of information security. These members of the classic infoSec triad are interchangeably referred to in the literature as security goals, information criteria, critical information characteristics, and security attributes. However, debate remains over whether or not this CIA trio is sufficient to satisfy quickly evolving business requirements and technology, with recommendations to expand on the intersections of availability and confidentiality, as well as the relationship between privacy and security [9].

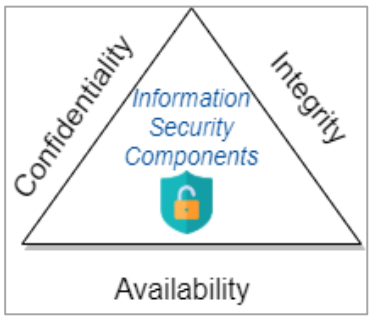


Figure 1. 2. CIA triad

- Definition of confidentiality by ISO/IEC 27000 standard as the “property that information is not made available or disclose to unauthorized individuals, entities or processes” [10]. Confidentiality involves a set of rules or promises that limit access or restrictions on a certain type of information that means only authorized users and processes can be modified or access data.

- Integrity is defined by ISO/IEC 27000 standard as the "property of accuracy and completeness" [8]. The data should be kept in a secure state and nobody should be able to modify it wrongly, either intentionally or accidentally. Integrity is maintained when the information remains unchanged during transmission and usage that does not involve data alteration. If we want to compare confidentiality with integrity, we can see that whereas confidentiality concerns the prevention of unauthorized reading, integrity concerns the detection of unauthorized writing. For example, assume you want to transfer payments electronically from one account to another. You may not want others to know about this transaction and keep it private, in this situation encryption successfully provides the needed confidentiality. Even you are concerned about confidentiality or not, you certainly want the transaction to be received accurately and completely; At this point, integrity comes into the picture [11].
- Availability is described by ISO/IEC 27000 standard as the "property of being accessible and usable on demand by an authorized entity" [10]. Availability means data that is provided for authorized users should be available as needed at any time. Denial of service attack (DoS) is one of the information security threats, as we know the primary goal of DoS attacks is rendering an information resource unavailable or in simpler terms main target is information availability; as we can see we can explain the concept of availability uniquely by defining denial of service as an attack example. Availability as a system property has been categorized into, basic availability and high availability. High availability can be achieved with a system that has redundancy in components (hardware and software) in order to achieve its functions. While basic availability is related to a standalone system that is developed with the necessary basic components and has a single point of failure, i.e. it will provide services as long as there is no DoS attack or any maintenance procedure, high availability has an alternate server that will take over the required tasks and functions in situations like this [12].

As we defined previously IDS, is a system that monitors important operating systems and attempts to determine some cases where violation and malicious exploitation occurring. Figure 1.3 shows the diagram for IDS where the system monitors network traffic and checks the incoming data, packets, and information of the system. Then it

applies signature-based detection where the system checks if the data matches with known patterns. If the answer is no, it also applies an anomaly-based detection method where it checks if attacks are found. If there is an attack, the system takes some actions, like reporting the packet as an intrusion, alerting the administration in case of malicious activity occurs or cutting down connections.

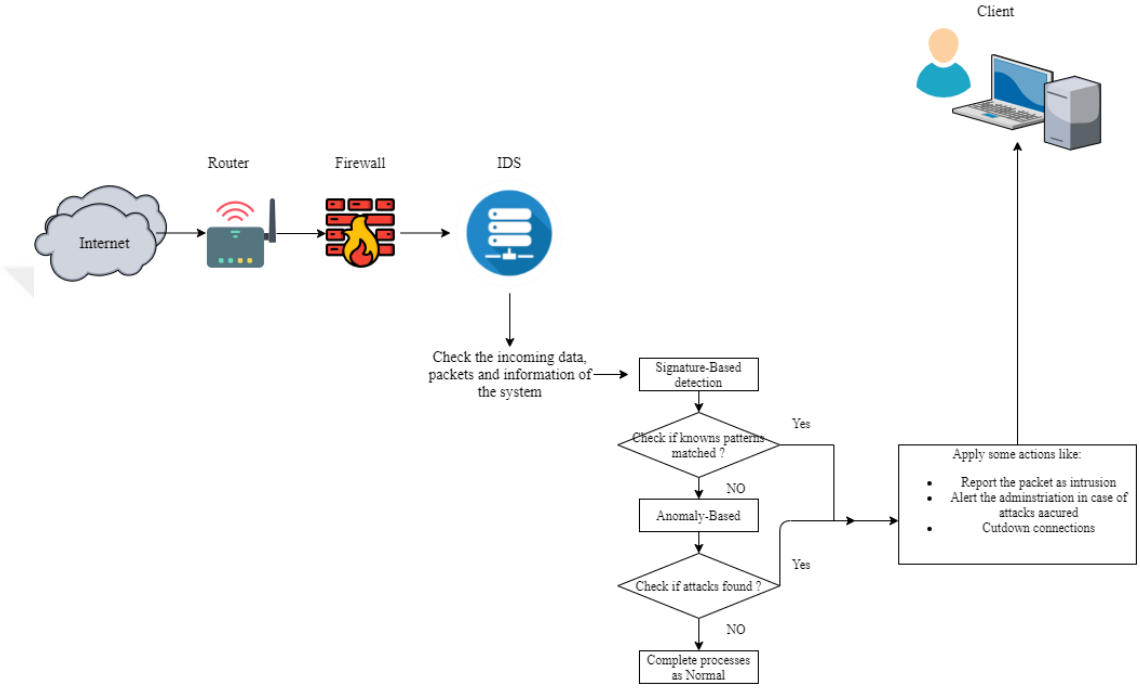


Figure 1. 3. IDS flow diagram

Briefly, we can see that tracking network traffic and systems to detect attacks and policy violations this process is called “intrusion detection”. Every software program or hardware device whose aim is to behavior attack detection is considered intrusion detection system (IDS). Figure 1.4 shows in a brief way how an IDS functions monitor activities and send alerts by using its database, statistics, or artificial intelligence techniques.

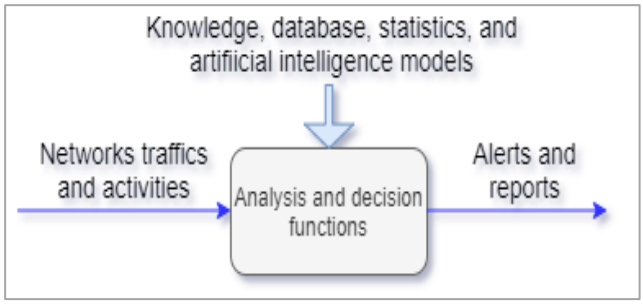


Figure 1. 4. Intrusion detection system function

1.2.2. Types and techniques of IDS

Intrusion is a set of actions and processes aimed at compromising the network security goals like integrity, confidentiality, and availability of a computing resource. A network intrusion refers to any unauthorized activity on a system. With the aim of proactive detection and responses versus network intrusions, here where is IDS's importance comes into the picture. There are different classification types and taxonomy for intrusion detection systems.

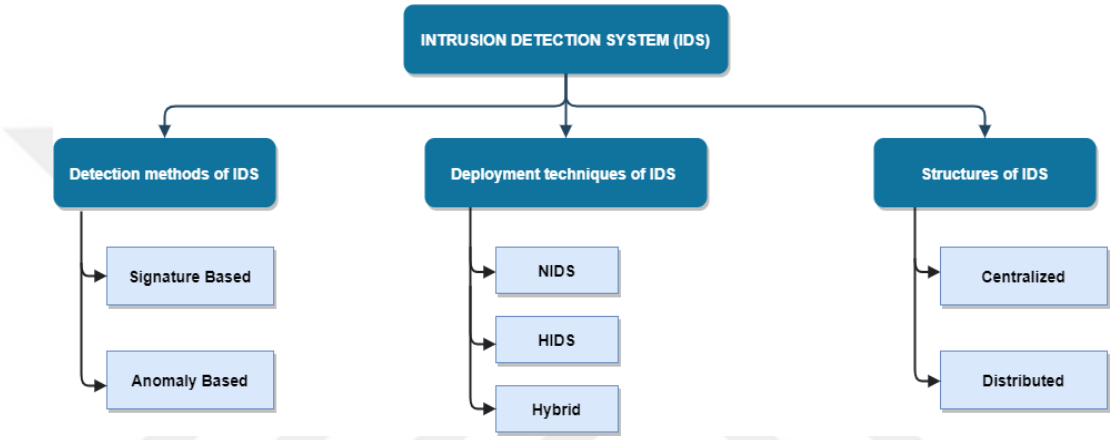


Figure 1. 5. Classification of Intrusion Detection System

The categorization of techniques and methodologies used in NIDS the majority of the literature supports the following categorizations:

- Misuse-based or signature-based intrusion detection systems (SIDS) are systems that use patterns or indicators extracted from known attacks. Every day new attacks appear and their maintenance, particularly in the light of the increased attack rate, is becoming a critical problem. This method detects malicious activities based on a dataset that saved attacks features and signatures.
- Anomaly-based intrusion detection systems (AIDS) are techniques that depend on the network behavior, anomaly-based has the advantage that it can detect zero-day attacks and it requires a training phase to construct and develop the database of general attacks. Anomaly detection is implemented in either the host or the network systems [13].
- Hybrid systems are the combination of the aforementioned approaches.

As detection method based IDS has two approaches: a signature-based approach that is a traditional and anomaly-based approach. In our study, we used anomaly-based approaches, which are based on machine learning techniques. The constant evolution of attacks, the rate of intrusions, and advancements in big data analytics and cloud computing make machine learning-based methods more appealing than they have ever been. The organization should be able to detect any type of attacks both old and new intrusions to make the system more reliable, secure, and consistent.

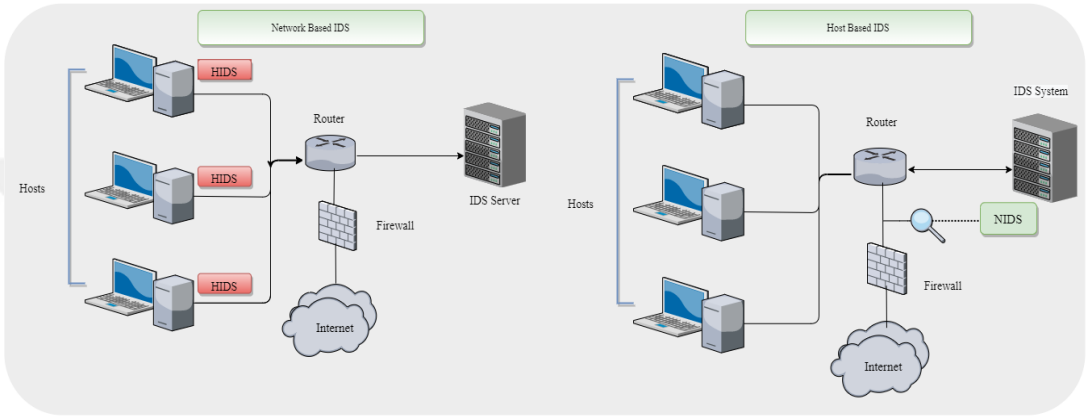


Figure 1. 6. NIDS vs. HIDS architectures

Based on deployment and data collection system, the intrusion detection system can be classified into three categories as below [14,15];

- A network-based intrusion detection system (NIDS) monitors, tracks, and analyzes network traffic in order to detect suspicious activity. It is a real-time detection method that evaluates all packet content and header information as it moves across the network.
- A host-based intrusion detection system monitors, tracks, and analyzes the application activity. HIDS examines historical information, it is based on individual users' systems and it is installed on any device, desktop, or server.
- A Hybrid detection technique is one of the best methods from the network point of view because it is a combination of the above methods, both host-based IDS and network-based IDS.

We can see from Figure 1.6 the difference between NIDS and HIDS in the structure.

1.2.3. Characteristics of IDS

We can introduce the characteristics of intrusion detection system as below [16,17]:

- The intrusion detection system is analyzing and monitors all the systems completely or sometimes the part of the system that the administrator needs.
- Intrusion detection might be advertised publicly or stealthily.
- Intrusion detection methods can depend on the behavior of attacks qualify it as a behavior-based method or on the information about attack which is called knowledge-based detection method.
- Behavior on detection, when the IDS detects an attack there is different responses can be obtained. Either passive behavior or active one we can choose the type of response that will be acquired.
- Usage frequency and analysis timing we can use continuous monitoring or periodic analysis.

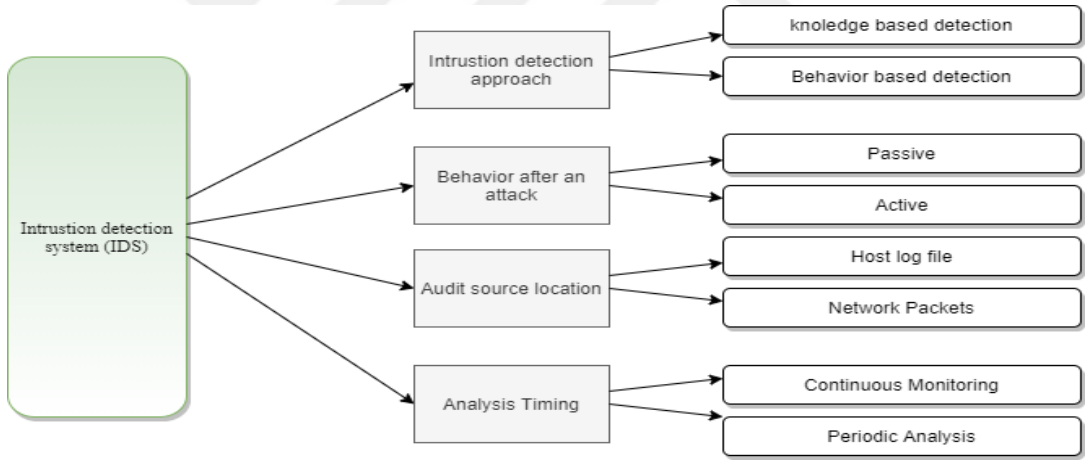


Figure 1. 7. Characteristics of Intrusion Detection System

We described the characteristics of intrusion detection systems and summarized them as shown in Figure 1.7 above.

1.2.4. IDS vs. IPS

An intrusion detection system (IDS) is software that automated system that analyzes, watches, and monitors the network traffic and generates a response to violation activity and it has become one of the most important countermeasures in the network security defenses. An intrusion prevention system (IPS) is software that has all the capabilities

of an intrusion detection system and can also attempt to stop suspicious efforts and prevent possible incidents. The main difference between them is that IDS is a monitoring system, while IPS is a control system, IDS does not alter network packets in any way whereas IPS prevents packets based on the delivered contents similar to how a firewall blocks traffic by IP address. In another way, we can say that the difference between deployment of these systems in the network IDS is out of the band in the system, which means it sits off-line network channel, but IPS is in-line, which means it can pass through devices and network paths. IDS generates only alert when suspected abnormal traffic is detected, alerts may be false negative or false positive, implying that IDS does not take action typically just triggers a response from the system, whereas IPS has both detection and prevention functionalities taking automatic or manual action against reported malignant intrusion such as dropping, blocking, or cut downing the connections [18].

1.3. Big Data and Analytic Applications

Every day the number of connected devices to the internet network is growing. As a result of the internet of things (IoT) technology, our data scalability is rapidly increasing. For that, we need storage space that is deployed and makes computation processes and analysis in an effective way. Today, big data has become one of the most widely used principles in computer science. Big data is characterized by the five V's which are Volume, Velocity, Variety, Value, and Veracity [19]. With the big data in large volume and velocity that is shared through networks, important data could be exploits. In this situation, the importance of the intrusion detection process comes into the picture. We need big data analytics applications to process, analyze and organize the data. We can process the data as batches or as streams. Batches consume time to create the batches from the given data. On the other hand, stream processing involves accessing the data directly in real-time as it arrives at the memory [20]. With the help of different frameworks and tools such as Hadoop, Strom, Spark, and Flink, we can managing data effectively. Recent works on big data for intrusion detection have proposed using big data processing methods to detect intrusion in cloud environments. The proposed solutions in this thesis combine classical machine learning models with deep learning to create a system that can detect intrusions. Various public IDS datasets were used to test the solutions recommended in this thesis.

1.4. Cloud Computing and Intrusion Detection in Cloud Networks

Cloud computing is the on-demand availability of computer system resources, particularly cloud storage and computing power, without the need for user management directly [21]. The term is generally used to refer to data centers that are accessible via the internet to a wide range of users [22].

These technological advancements have made cloud computing a popular choice to store many cloud services and operations. The advantages of cloud services have made many companies migrate and transform their systems to those platforms. Hence, with this progressively growing, the systems have become a target for cyberattacks and violations. Vulnerabilities of cloud get a need for security measures to detect and prevent those attacks in generally take action. It is an important step to protect and defend our systems from network breaches. Some detection methods that already exist can be capable of cloud services, but some new approaches have been proposed specifically to solve this problem for cloud systems. Systems support a massive amount of data collected from monitors, which must be processed quickly in order to detect attacks.

2. RELATED WORKS

In this chapter, we tried to analyze and investigate intrusion detection methods from both aspects, from general networks aspect also internet of things.

2.1. Literature Reviews on Solutions for Threats Detection

With the constant growth of the internet, cyberattacks are increasing in both diversity and quantity, Every day there are new threats generated which are known as zero-day exploits. Firewalls and antivirus software are no longer efficient to guarantee the protection of systems and establishment networks which constructed on multiple layers of security and had more complex structures. One of the more effective techniques to protect our systems against ransomware threats, Intrusion Detection System (IDS) with its feature of continuous monitoring and tracking.

IDS is divided into two main subcategories: Signature-based Intrusion Detection System (SIDS) and Anomaly-based Intrusion Detection (AIDS). Signature intrusion detection system (SIDS) is compared packets with previously known patterns to find a known attack, this system is also known as Knowledge-based Detection or Misuse Detection. The main concept for SIDS is to build a database of threat signatures and to compare the current activities against signatures that already exist and trigger an alarm if a match is found. describe in database. On the other hand, Anomaly-based intrusion detection system (AIDS) depends on the behavior of a network system is created using machine learning, statistical-based, knowledge-based methods [23].

Over the last few years, Artificial intelligence (AI) advancements such as machine learning and deep learning approaches have been applied to improve security and IDS for IoT. Several studies utilizing various machine learning and deep learning techniques on different datasets. However, it is unclear which dataset with machine learning or deep learning methodology is more useful for developing an efficient IoT IDS. Furthermore, the time cost incurred for developing IoT IDS is not considered when evaluating certain IDSs methodologies, despite it is a significant factor for

calculating the effectiveness of IDSs [24]. Figure 2.1 shows the relationship amongst AI, ML, and DL techniques. In this section, we will handle anomaly-based IDS techniques which is depend on heuristics, statistics or rules, rather than signatures or patterns.

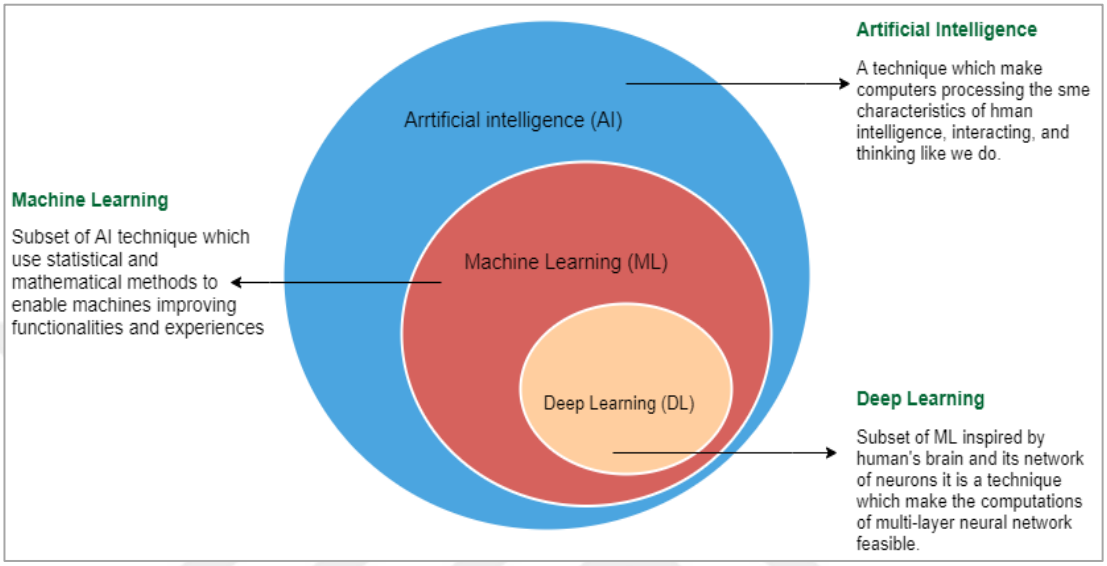


Figure 2. 1. Relationship amongst AI, ML, and DL techniques.

2.2. Machine Learning-Based Applications in Intrusion Detection

Machine learning is a subset of AI techniques that use statistical and mathematical methods to enable machines to improve functionalities and experiences. ML aims to give systems the ability to learn automatically through experiences without the need to be explicitly programmed. Depending on the type of "signal" or "feedback" available to the learning system, machine learning approaches had been classified into three categories: supervised learning, unsupervised learning, and reinforcement learning. Supervised learning algorithm, which is used labeled instances to make a decision, and prediction, where the algorithm learns a general rule that maps inputs with outputs. Types of supervised learning algorithms are including classification and regression. Unsupervised learning algorithm no labels are provided, leaving it to its own methods to discover hidden patterns in data, here algorithm attempts to find clusters of data points. Reinforcement learning involves software agents which interact with their environment, in order to increase some measure of cumulative reward [25]. ML has many applications which include data mining, image recognition, natural

language programming (NLP), recommendation system, bioinformatics, computer vision, search engine, genetic algorithms, statistical learning methods [26].

In the literature, there are many studies have been proposed to enhance the IDS performance. In this section, we focus on the works that have used machine learning methods. In [27] Rashid et al. explored an attack detection technique based on machine learning algorithms (LR, SVM, DT, RF, KNN, and ANN) to mitigate IoT threats in a smart city. They considered feature selection, cross-validation, and multiclass classification binary classification. The feature selection method that is implemented here is the information gain ratio based method and it selects the top 25 features which are highly relevant to the prediction for both datasets. The authors also introduced ensemble methods such as bagging, boosting, and stacking to increase the performance of the detection system. UNSW-NB15 and CICIDS2017 datasets were used to evaluate the methods. Its results indicated that the ensemble stacking model can give better detection in IoT-based smart cities. Ullah et al. [28] proposed a two-level anomalous activity detection method for the system in IoT. The proposed model is constructed on flow-based features of the IoT network which are extracted from the IoT Botnet dataset. The level-1 model categorized the network flow as normal flow or anomaly flow, while the level-2 model is investigated to classify the category or subcategory of the detected malicious behavior. In their research, they analyzed the network flow from each device and extracted features using Wireshark or TCPdump to intercept the network packets. The decision tree model produced the highest predictive results for level-1 as 99.99%, while at level-2 the random forest classifier yielded the highest predictive results with 99.99% rate. IoT devices are now connected without human intervention for a longer period this refers to intelligent network-based security solutions.

In this study [29] new features were extracted from the Bot-IoT and compared with existing studies from the literature. Alsamiri et al. extracted features using CICFlowMeter. In the evaluation phase, seven different machine learning (KNN, QDA, ID3, RF, AdaBoost, MLP, and NB) were used. They observed that Adaboost was the best performing algorithm, followed by KNN and ID3. In [30] the botnet attack detection is achieved by using j48, NB, and ANN machine learning models. The proposed solution has two phases: (1) “Model Builder”, and (2) “Attack Detector”. Its

structure depends on sequential attack architecture, where the "Attack Detector" will sequentially perform the intrusion detection. The N_BaIoT dataset was used to evaluate the suggested model also the hybrid classification (serial/parallel) was provided.

2.3. Deep Learning-Based Applications in Intrusion Detection

Deep learning is a subset of machine learning methods based on artificial neural networks with feature/representation learning. Artificial neural network (ANN) is inspired by information processing, human brain, and distributed communication nodes in biological systems. ANN is based on collections of connected units/artificial neurons. ANN, also known as connection systems, are computing systems that learn and improve their abilities over time. is often used to solve complex problems which are difficult to express with traditional algorithms that depend on rule-based programming [31]. Deng and Yu [32] stated that deep learning methods can be classified into two models: discriminative/supervised models and generative/unsupervised models. The deep discriminative approaches include recurrent neural network (RNN), deep neural network (DNN), and convolutional neural network (CNN). The generative/unsupervised models include approaches, namely deep autoencoders, restricted Boltzmann machine (RBM), and deep Boltzmann machines (DBM), deep belief networks (DBN), generative adversarial network (GAN). Depending on how the architectures are intended for use, for example, synthesis/generation or recognition/classification, the majority of the work in this area can be divided into three categories as follows; Category 1: deep networks for supervised learning are designed to provide discriminative power for pattern classification, and these models are used to map an input to an output using examples of input-output pairs, and also to analyze the training data and produce an inferred algorithm that can be applied to correctly determine the class labels for unseen instances, Category 2: deep networks for unsupervised or generative learning which are used for generative tasks with no information about target class labels, and Category 3: hybrid deep networks which are the combination of the above 1 and 2 categories.

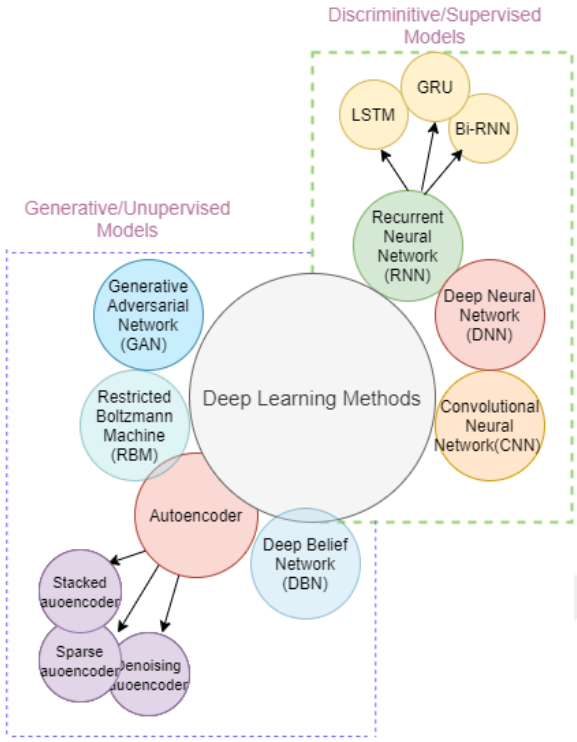


Figure 2. 2. Taxonomy of deep learning approaches

In the literature, in order to increase the performance and effectiveness in detecting intrusions, deep learning-based models are also proposed and used. As in the study [33], the authors suggested the detection of a threat based on a two-stage two-layer deep learning model, which is called TSDL. The TSDL model is based on a stacked auto-encoder with a softmax classifier. To evaluate the model's effectiveness, benchmark KDD99, and UNSW-NB15 datasets are used. The proposed model achieved a high classification rate, up to 99.996% for KDD99 and 89.134% for UNSW-NB15. In [34] the primary objective of this work is to design a model that combines an improved conditional variational AutoEncoder with a deep neural network, namely ICVAE-DNN. The NSL-KDD and UNSW-NB15 datasets are used to evaluate the effectiveness of the improved technique. The ICVAE-DNN model consists of three main phases: (1) training phase, where the training samples used to train the proposed model and reconstruction loss are calculated and stored, (2) generation, where method generated new samples and merged into original in order to balance the training data and increase the diversity of the training instances, and (3) detecting attack where DNN classifier is evaluated over the merged training data set. The authors aimed to minimize the loss of the encoder-decoder model. The results

indicate that the highest accuracy of 89.08% and DR of 95.68% on the UNSW-NB15 data set were achieved.

In [35] to provide zero-day threats and anomaly attack detection the deep belief network (DBN) is used. To evaluate the method, the authors used both real-network traces and simulation for demonstrating evidence of its scalability. The Cooja simulator Contiki, Keras library (open-source Python library), and Texas Instruments sensor tags CC2650 are employed to evaluate the performance. The Keras library is used for the development of a sequential deep-learning model and IDS technique tested on a low-powered Raspberry Pi. In their study, various attacks such as sinkhole attacks, DDoS, blockhole attacks, wormhole attacks, and service attacks are simulated to test the proposed detection model. The observed results show an average precision rate of 95% and a recall rate of 97% for different attack scenarios. Basumallik et al. [36] implemented a convolutional neural network (CNN) to increase the security of phasor measurement units (PMU) and detect packet-data anomalies. By using convolutional neural network filter-based with Adam gradient descent and categorical cross-entropy, event features (signatures) can be extracted from phasor measurement units. IEEE-30 bus and IEEE-118 bus systems are used for all simulations. The study resulted that CNN-2d achieved the highest classification accuracy of 98.67% with λ = 0.0001, time=540 second, dropout probability of 0.5, and fully connected layer with 512 neurons. According to the authors, the proposed convolutional neural network-based filters outperform other machine learning-based detection techniques such as RNN, LSTM, SVM, bagged, and boosted.

Based on two layers of the neural network, Zeng et al. [37] developed a model for detecting malicious traffic, the first layer which is composed of the improved LetNet convolutional neural network method and the second layer uses long short-term memory. In detail, the LetNet CNN layer is proposed to extract the spatial features, while the LSTM layer is proposed to extract temporal features. To provide comparative studies with the DFR framework, for the classification efficiency the algorithms such as C4.5 decision tree (DT) and 1D convolutional neural network (1D-CNN) classification algorithms were used. On the other hand, for the intrusion detection efficiency, decision tree and KNN algorithms were used. The proposed system demonstrates superior accuracy, precision, recall, and F1-measure. Thus, the

presented approach is a light-weight framework utilizing deep learning for encrypted traffic classification and intrusion detection named deep-full-range (DFR). DFR is capable to learn raw traffic without requiring manual intervention, human process, or access to private information. The evaluation stage was conducted using two public datasets: ISCX VPN-nonVPN traffic dataset and ISCX 2012 IDS dataset [17] respectively. The experimental results demonstrate that DFR framework not only exceeds state-of-the-art methods by an average of 13.49% on the F1 score for encrypted traffic classification and 12.15% on the F1 score for intrusion detection but also requires significantly less storage resource.

Ferrag et al. [38] in their study tried to implement deep learning-based intrusion detection methods such as RNN, DNN, CNN which are classified as supervised/discriminative methods, also apply restricted Boltzmann machines (RBM), deep belief networks (DBN), deep Boltzmann machines (DBM), and deep autoencoders (DA) which are classified as unsupervised/generative models. The authors presented a comparative study between different DL models on two datasets CSE-CIC-IDS2018 and the Bot-IoT dataset. As a result, the best accuracy is achieved for Bot-IoT dataset with deep autoencoder as %98.394 while for CSE-CIC-IDS2018 dataset was achieved with CNN model as %97.376. Vinayakumar et al. [39] inferred from their work that experiments of families of RNN architecture achieved a low false positive rate in comparison to the traditional machine learning classifiers. The reason for that is that RNN architecture is able to memorize information over time. This work applied on publically available ID datasets, KDDCup '99' and UNSw-NB15. Idrissi et al. [40] in their research proposed a new solution called baptized BotIDS, which is based on deep neural convolutional neural networks. BotIDS is a solution that is planned to be placed in a fog node which gives its power of analyzing in real-time inbound/inside and outbound/outside traffic through the network system. Models are tested using well-known Botnet attacks and a Bot-IoT dataset. The accuracy of the solution was compared to that of other DL techniques such as RNN, LSTM, and GRU. The obtained results concluded that CNN is the best one for intrusion detection systems with its higher accuracy of 99.94% and lower loss rate of 0.58%.

Traditional intrusion detection systems have their limitations when applied to the IoT network due to resource constraints and complexity. Due to that, Liang et al. [41] in

his work proposed a more complicated model which uses a hybrid placement strategy based on a multi-agent system, blockchain, and deep learning algorithms. This suggested SAE + DNN algorithm is tested over NSL-KDD dataset. The simulation results demonstrated that the deep learning algorithm has a better performance than traditional methods on the same type of IoT network. The accuracy that is obtained from the provided method is 98.27%.

In [42] Ibitoye et al. conducted to discover suspicious behaviors using a self-normalizing neural network (SNN) and forward neural network (FNN) and compare their performance with each other. When tested the models for adversarial robustness, the SNN shows better performance against the adversarial samples from the IoT dataset. Also, the authors analyze the effects of feature normalization on the adversarial robustness of deep learning-based IDS in IoT. In [43] the authors suggested an FNN model for both binary and multi-class classifications to detect threats which are including Dos, DDos, reconnaissance, and information theft attacks against IoT devices. The results in binary classification illustrated a high accuracy, precision, recall, and F1 score by a score nearly to 0.99% for Dos/DDos and reconnaissance attacks classes. While in multiclass classification, the detection accuracy of above 0.99% for DDoS/DoS attacks was reported and for the normal traffic class, the accuracy of .98% was achieved.

2.4. Feature Engineering and its Effect on Intrusion Detection Methods

Feature engineering is the process of extracting features from raw data and converting them to machine learning model-compatible formats. Feature engineering techniques consist of feature transformation, dimensionality reduction, feature extraction, and feature selection methods [44].

Feature transformation is a method that includes processes like data scaling, standardization, and normalization. Dimensionality reduction is the method where the data transformed from a high-dimensional space to a low-dimensional representation, keeps some meaningful of the original data. The feature selection method eliminates irrelevant and non-useful features to reduce the complexity of the resulting model. Feature engineering allows improving the model's performance and enhancing the

accuracy rate through eliminating irrelevant features and applying feature transformation to increase the model's effectiveness. It also results in reducing the memory and time requirements of machine learning or deep learning workflows [44].

Here are some related works that are applied feature engineering methods like; features dimensionality reduction approach and feature selection technique. In [45] from a feature engineering perspective, Pajouh et. al applied the dimensionality reduction method. They proposed two-stage model dimension reduction and classification techniques perspectively to detect malicious activities in IoT networks, such as User to Root (U2R) and Remote to Local (R2L) attacks from the NSL-KDD dataset. In methods details, they implemented Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) to reduce features dimensions after that use the Naive Bayes and K-nearest Neighbor to identify suspicious behaviors and achieved 84.82% as a detection rate. Also in [7], the authors tried to analyze the performance of intrusion detection system using a feature selection method on the UNSW-NB15 dataset, then implemented the following machine learning approaches using the reduced feature space: Support Vector Machine, k-Nearest-Neighbour, Logistic Regression, Artificial Neural Network, and Decision Tree. The feature selection method that is applied was a filter-based feature reduction technique using the XGBoost algorithm. The results showed that the XGBoost-based feature selection method allows models such as the Decision tree model to enhance its test accuracy rate from 88.13% to 90.85% for the binary classification scheme.

Classical machine learning methods depend heavily on feature engineering, extracting features stage is often time-consuming and complex. As a result, it is impractical to detect attacks in real-time applications using traditional machine learning techniques [46]. For that in our work, we tried to apply besides traditional machine learning methods, also deep learning methods where there is no need for feature engineering, and feature representations have been learned automatically. Although all of those proposed methods are capable of detecting network intrusions efficiently, with the low-capacity nodes and complexity of IoT structure it is important to build an IDS that consumes minimal energy, achieves low computational costs, and requires less memory in the network nodes [47].

3. TECHNICAL DETAILS

In the literature, there are a lot of relevant intrusion detection datasets. As an example, KDD99, NSL-KDD, DS2OS, UNSW-NB15, CIC-IDS 2017, MQTT-IOT-IDS2020, and Bot-IoT Datasets. However, in this section, we describe the dataset we used in our work.

3.1. UNSW-NB15 IDS Dataset

In our study, we used the UNSW-NB15 dataset which is created by the Cyber Range Lab of the Australian Centre for Cyber Security [48]. We selected this dataset because it is a public dataset not private, diversity of attack types those included in this dataset, the ability to generate new features from PCA files using feature extraction tools like the CICFlowMeter tool, and regular updates that can be applied to this dataset.

The Bro-IDS, Argus tools are employed and twelve algorithms are developed to extract totally 49 features with the class label [48]. A part of this dataset is divided into training and testing sets, namely, UNSW_NB15_training-set.csv and UNSW_NB15_testing-set.csv respectively. The number of instances in the training set is 175,341 (68.05%) records and the testing set is 82,332 (31.95%) records from the different types of attack and normal. The UNSW-NB15 dataset includes nine types of attack classifications to describe malicious behaviors. Attack types included in the UNSW-NB15 dataset are Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. The distributions of attacks and normal samples in the training set of UNSW-NB15 are reflected in Figure 3.1 and Figure 3.2:

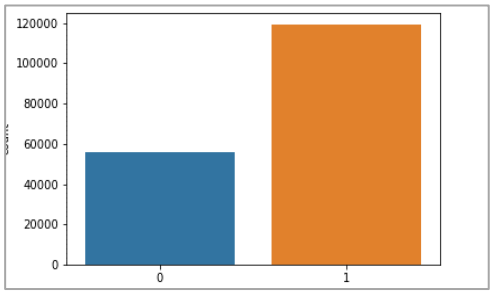


Figure 3. 1. Distribution of normal and attack samples in the training set

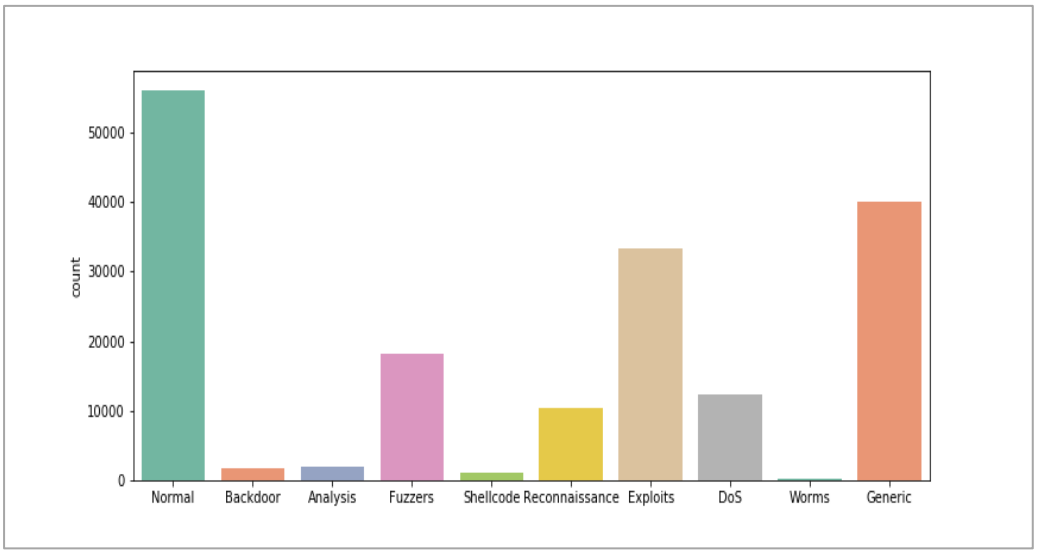


Figure 3. 2.Distribution of attack types in the training set of UNSW-NB15

The attributes of the UNSW-NB15 dataset are classified into five categories: Flow features, Basic features, Content features, Time features, and Additional generated features. The features contain Nominal, Integer, Float, Timestamp, and Binary types. The descriptions and information about the features in the dataset are shown in Table 3.1, also in Table 3.2, the definition of attacks is given.

Table 3. 1. Descriptions of the features in the dataset [48]

#	Name	Description	Type	Category
1	Srcip	Source IP address	Nominal	Flows features
2	Sport	Source port number	Integer	
3	Dstip	Destination IP address	Nominal	
4	Dsport	Destination port number	Integer	
5	Proto	Transaction protocol	Nominal	
6	State	The state and its dependent protocol, e.g. Acc, clo, else	Nominal	Basic features
7	Dur	Record total duration	Float	
8	Sbytes	Source to destination bytes	Integer	
9	Dbytes	Destination to source bytes	Integer	
10	Sstl	Source to destination time to live	Integer	
11	Dttl	Destination to source time to live	Integer	
12	Sloss	Source packets dropped or retransmitted	Integer	
13	Dloss	Destination packets dropped or retransmitted	Integer	
14	Service	Ssh, http, ftp, dns, smtp, else	Nominal	
15	Sload	Source bits loaded per second	Float	
16	dload	Destination bits loaded per second	Float	
17	spkts	Source to destination packet count	Integer	
18	Dpkts	Destination to source packet count	Integer	
19	Swin	Source TCP windows advertisement	Integer	Content features
20	Dwin	Destination TCP window advertisement	Integer	
21	Stcpb	Source TCP sequence number	Integer	
22	Dtcpb	Destination TCP sequence number	Integer	
23	Smeanz	Mean of the flow packet size transmitted by the src	Integer	

Table 3. 1. (Cont.) Descriptions of the features in the dataset [48]

#	Name	Description	Type	Category
24	Dmeanz	Mean of the flow packet size transmitted by the dst	Integer	Content features
25	Trans_depth	The depth into the connection of http response/request transaction	Integer	
26	Res_bdy_len	The content size of the data transferred from the server's http service	Integer	
27	Sjit	Source jitter (mSec)	Float	Time features
28	Djit	Destination jitter (mSec)	Float	
29	Stime	Record start time	Timestamp	
30	Ltime	Record last time	Timestamp	
31	Sinpkt	Source inter-packet arrival time (mSec)	Float	
32	Dintpkt	Destination inter-packet arrival time (mSec)	Float	
33	Tcprtt	The sum of 'synack' and 'ackdat' of the Tcp connection	Float	
34	Synack	The time interval between the syn and syn_ack packets of the Tcp connection	Float	
35	Ackdat	The time between the syn_ack and syn packets of the Tcp connection	Float	
36	Is_sm_ips_parts	If srcip (1) equals dstip (3) and sport (2) equals dsport (4), this variable has a value of 1; otherwise, it has a value of 0	Binary	Aditional generated features
37	Ct_state_ttl	No. for each state (6) according to a specific range of sttl (10) and dttl values (11)	Integer	
38	Ct_ftw_http_mthd	No. of flows that has methods such as Post and Get in http service	Integer	
39	Is_ftp_login	If the ftp session is accessed by user and password then 1 else 0	Binary	
40	Ct_ftp_cmd	Number of flows that has a command in ftp session	Integer	
41	Ct_srv_src	No. of records that contain the same service (14) and srcip (1) in 100 records according to the ltime (30)	Integer	
42	Ct_srv_dst	No. of records that contain the same service (14) and destip (3) in 100 records according to the ltime (30)	Integer	
43	Ct_dst_itm	No. of records of the same destination address (3) in 100 records according to the last time (30)	Integer	
44	Ct_src_itm	No. of records of the same source address (1) in 100 records according to the last time (30)	Integer	
45	Ct_src_dsport_itm	No. of connections of the same srcip (1) and the dsport (4) in 100 records according to the ltime (30)	Integer	
46	Ct_dst_sport_itm	No. of connections of the same destip (3) and the sport (2) in 100 records according to the ltime (30)	Integer	
47	Ct_dst_src_itm	No. of connections of the same srcip (1) and the dstip (3) in the 100 records according to the ltime (30)	Integer	

Table 3. 2. Descriptions of different types of attacks in the UNSW-NB15 dataset

No.	Traffic Type	Description
1	Normal	Normal transaction data without threat.
2	Analysis	An attack to infiltrating web applications using emails, port scans, or web scripts penetrations.
3	Fuzzers	Attempts to find security vulnerabilities in system, program, or network by feeding it with a lot of random data to interrupt the target system's services.
4	Backdoors	A technique to circumvent and bypass authentication process of a system in order to grant remote access to resources such as databases and files.
5	DoS	A malicious attempt to prevent authorized users from accessing a server or a network resource, typically by temporarily interrupting the services of an Internet-connected host or by flooding the server with unavailable authentication attempts forcing it to crash or bring down.

Table 3. 2. (Cont.) Descriptions of different types of attacks in the UNSW-NB15 dataset

No.	Traffic Type	Description
6	Exploits	The attacker is aware of a security problem in an operating system or software and leverages that information by exploiting the vulnerability.
7	Generic	A technique that works against all block-ciphers without regard for the block-cipher's structure by utilizing hash functions.
8	Reconnaissance	A probe that collects relevant information about the target system in order to circumvent the network security controls.
9	Shellcode	Set of instructions that inject and execute a command shell to exploit or take control of a compromised machine.
10	Worms	A type of malware that replicates itself and spreads copies of itself to additional computers via a computer network, utilizing security vulnerabilities on the target machine to get access to it.

As we mentioned before, we chose this data because it is a newly generated dataset versus the old other datasets, it is a public dataset, it contains a variety of attack types, the difficulty of evaluating and analyzing the UNSWNB15 on existing classification systems demonstrated that this data set contains complex patterns, the training and testing sets have a similar probability distribution, and the capability of receiving regular updates. The Venn diagram of different types of attacks between UNSW-NB15, NSL-KDD, and Bot-IoT datasets as shown in Figure 3.3, we can see that UNSW-NB15 dataset has nine types of attacks.

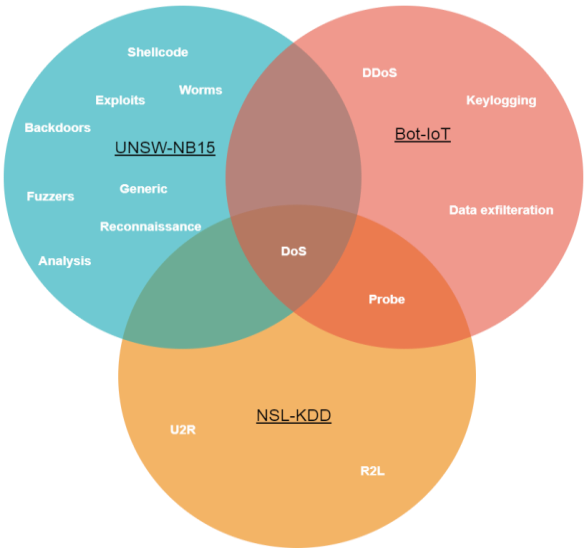


Figure 3. 3. The Venn diagram of different attacks between three IDS datasets

3.2. Hardware Characteristics

The results we obtained from the proposed models are implemented by using a Lenovo laptop with the Windows operating system version 8.1. The system contains an Intel (R) Core i7-4710HQ processor with a processing speed of 2.50 GHz. The installed RAM of the system was DRR3 16 GB. The graphics process unit of the machine was an NVidia GeForce GTX 860M.

3.3. Technologies Overview

In our work, we used Python programming language, Python is an interpreted, high-level, object-oriented. Today, machine learning has become more popular and attracted the attention of students, scientists, and researchers. Python with its useful packages and libraries enables to make complex computational tasks easily. Python has a lot of libraries such as Keras, Scikit-learn, Tensorflow, pandas, Numpy, and Matplotlib, etc. We used these libraries in our study to help us implementing feature engineering process methods over the dataset, classical machine learning, deep learning. Python is known for its features, consistency, and platform independence. The advantages that make Python the best for AI-based projects are its features, consistency, platform independence, great community, popularity, and also its extensive libraries and frameworks. Scikit-learn is designed for various classification, clustering, and regression, and dimensionality reduction algorithms. Scikit-learn Built on NumPy, SciPy, and Matplotlib. Numpy is used for scientific computing and data analysis with excellent performance. Pandas for general purpose data analysis [49]. Matplotlib is a library that provides a set of static, animated, and interactive visualizations and graphs [50]. Tensorflow is a comprehensive end-to-end open-source platform for machine learning, it helps researchers and developers to build and deploy ML-powered applications. [51]. Keras is an open-source software that executes an interface for artificial neural networks. Keras works as an interface for the Tensorflow library. It focuses on being fast, modular, and user-friendly [52]. All of these libraries have extensive documents containing code examples, methods lists, figures, definitions of functions, and parameters. The proposed models in our work are implemented in Jupyter Notebook by using the Python language and its libraries.

4. METHODS FOR INTRUSION DETECTION SYSTEM

4.1. Pre-processing Data

- Feature Transformation and Standardization:
  Before feeding the data to classical machine learning and deep learning models, feature transformation has been applied and this step is important because models accept only numerical data as input for that, all non-numerical values of the dataset are converted into numerical data. In this work, we used one hot encoding (ohe) technique to encode categorical features as a one-hot numeric array. The encoder derives the unique values for each feature and represents it as a one-hot array. On the other side, for the standardization process, we used the StandardScaler technique over the numerical data because the values of the dataset are in different ranges and we tried to standardize data in the same range. Standardization is a scaling technique where input values are centered around the mean with a unit standard deviation.

- Feature Selection and Dimensionality Reduction:
  In this step, we select the most important features and delete unnecessary features or reduce the dimensions of features in the dataset. For the task of feature extraction and dimension reduction processes, we can use different models like Principal component analysis (PCA), Linear discriminant analysis (LDA), Autoencoder, and t-SNE models. On the other hand, feature selection is used to reducing irrelevant and redundant variables and it measures the relevance of each feature with the output labels/classes based on feature importance metric. Feature selection technologies are divided into embedded, wrapper, and filter methods [53]. Our goal from applying the feature selection method and reducing the number of input variables is to improve the performance of the model in some cases and reduce the computing cost and time of the training model.

In this study, we applied only feature selection methods using Random Forest that depends on tree-based strategies and belongs to the category of embedded methods.

Embedded methods combine the qualities of filter and wrapper methods, are more accurate, and generalize better. Features importance and its scores that calculated using Random Forest method as shown in Figure 4.1:

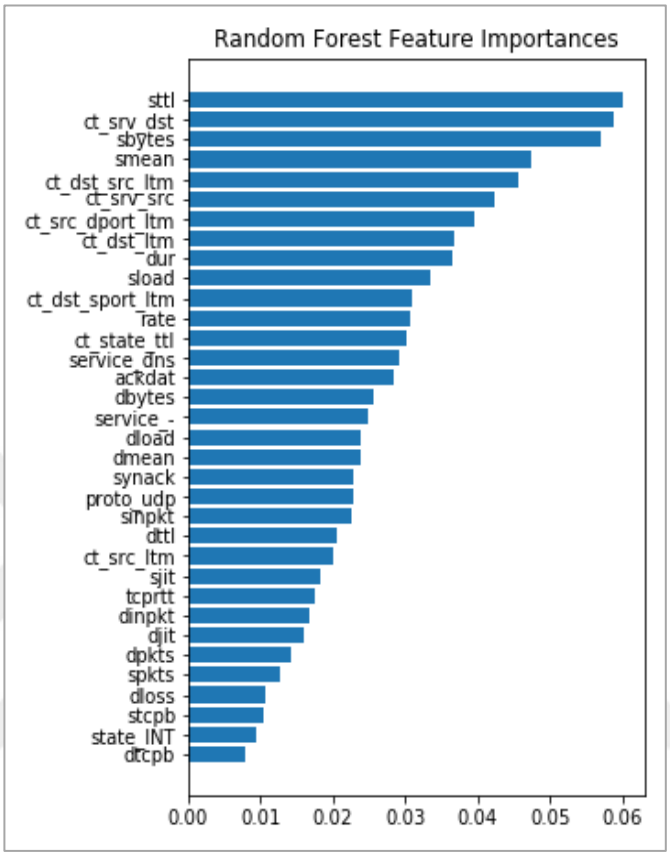


Figure 4. 1. Feature Selection using Random Forest

4.2. Classical Machine Learning Methods

4.2.1. Naïve bayes

Naive Bayes (NB) is a subset of Bayesian decision theory and it is a simple probabilistic machine learning model based on the Bayes theorem where assumptions between features are considered independent. This method is used for classification tasks. There are different types of Naive Bayes as Multi-nominal, Bernoulli, and Gaussian Naive Bayes algorithms [54]. In our work, we used Bernoulli Naive Bayes model. Bayes theorem mathematically can be described as follows:

$$P\,(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{4.1}$$

Where A and B are different events, P(A) and P(B) are the probabilities of observing A and B respectively without any given conditions, P(A|B) is the probability of event A occurring given that B is true and it is also called the posterior probability of A given B, and P(B|A) is a probability of event B occurring given that A is true.

4.2.2. K-nearest neighbors

K-nearest neighbors (KNN) is an ML algorithm that is capable of both supervised and unsupervised approaches and it is used for both classification and regression problems. In this research, we used it for supervised binary and multiclass classification tasks. KNN algorithm assumes that similar instances exist in the same area and proximity. Different distance metrics are used in this model. Distance metrics find the distance between two instances between a new data point and an existing point in the training dataset [55]. One of the commonly used distance metrics is Euclidean distance, the formula of it as follows:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{4.2}$$

This formula based on Pythagorean Theorem; and it can be used to calculate the distance between two data points x and y in Euclidean space.

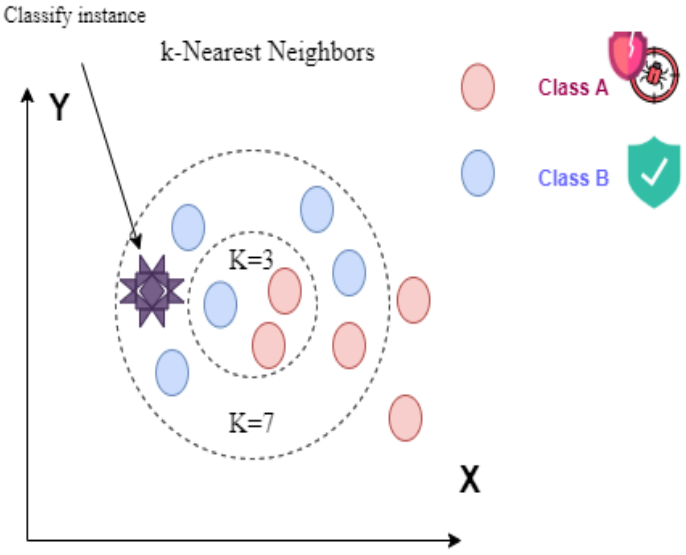


Figure 4. 2. k-NN classifier

4.2.3. Logistic regression

Logistic regression (LR) is a statistical method used for binary classification tasks. Although its name regression, it is a classification algorithm. Logistic regression tries to make a logarithmic line that distinguishes between classes and its estimation is done through maximum likelihood. LR model depends on the Sigmoid function where its logistic curve is limited between 0 and 1 values [56]. The expression of Sigmoid function as follows:

$$\sigma(x) = \frac{1}{1 + e^{(-x)}} \tag{4.3}$$

The mathematical definition of the logistic Sigmoid function shows that this function map any real value x into another value raged between 0 and 1. In machine learning, we used it to map predictions to probabilities.

4.2.4. Decision tree

Decision Tree (DT) is a decision support tool that uses a tree-like method. DT is a supervised model consisting of internal nodes that represent attributes, branches that represent the outcome of the tests, and leaf nodes that represent classes/labels and decisions after the computing process. The paths between root and leaf represent decision rules for classification tasks [57].

4.2.5. Random forest classification

Random Forest (RF) is an ensemble method for classification and regression. RF model is a combination of different decision trees. The ensemble method is a machine learning technique that combines several base models or decision trees to produce one optimal model and predict with better performance than utilizing a single model [58].

4.3. Deep Learning Methods

Deep learning (DL) is a subfield of machine learning inspired by the structure of the human brain and biological neural networks. DL is known with its high performance and efficiency across many types of data.

4.3.1. Deep neural network

Deep neural network (DNN) is an artificial neural network (ANN) model with high complexity, usually at least two hidden layers. This model has become a popular model for classification, regression, clustering, controlling models, and prediction in many applications [59]. Deep net process data by employing sophisticated math methods [60]. The feed-forward neural network was the first of neural networks (NN) that found and simplest type. In this network, the data move forwardly from the input layer through any hidden layer to the output without loops. The model can give different performance results depends on the number of hidden layers, the number of nodes in each layer, and the type of activation layer [59].

This model is applied in a supervised manner with the class labels and the input attributes. In this model, we have forwarding propagation which aims to predict results as an attack or normal by using a perceptron classifier. The main Equation of the perceptron in the artificial neural network is mentioned in Equation (4.4):

$$y = \sum_{i=1}^{n} X_i W_i + b \tag{4.4}$$

Where n denotes the number of nodes in the layer, X denotes the values of these nodes which are the samples values, W refers to weights (connection strength), and b to the biases of these nodes. These results will be inserted into different activation functions which return the probabilities for each class and then choose the largest value from the vector of probability values to give a more accurate value. Sigmoid, ReLU, Softmax, and Tanh are among the most frequently used activation functions; this model employed ReLU activation functions in the hidden layers stated in Equation (4.5).

$$R(z) = \begin{cases} z, & z > 0 \\ 0, & z \leq 0 \end{cases} \tag{4.5}$$

Also, we used Softmax activation function in the output layer stated in Equation (4.6).

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{4.6}$$

Here Softmax function transforms a vector of numbers into a vector of probabilities between 0 and 1, where $\vec{z}$ is the input vector, z_i represents the elements of the input vector which can take any real value, K is the number of classes and labels in the classifier, and $\sum_{j=1}^{K} e^{z_j}$ is the normalization term to ensure that all the function's output values sum to 1 and each is in the specified range between 0 and 1. After the forwarding propagation stage, the backpropagation step comes and is a technique to train deep neural networks by modifying the weights and biases. It includes loss function and optimizer [61]. In this step, the loss between predicted and true values will be calculated, then adjust the weights in the neural network according to the loss. Categorical cross entropy loss function have been applied in this work. The loss function needs to reach the optimal values of parameters (weight and bias) for that we used Adam optimizer to get the best parameter values. The optimizer is a way for tuning parameters [61].

In our work for DNN models, we used two different architectures DNN-2 and DNN-1. DNN-2 structure is more complicated than DNN-1. The structure of layers of models is described as below with Figure 4.3.a and Figure 4.3.b:

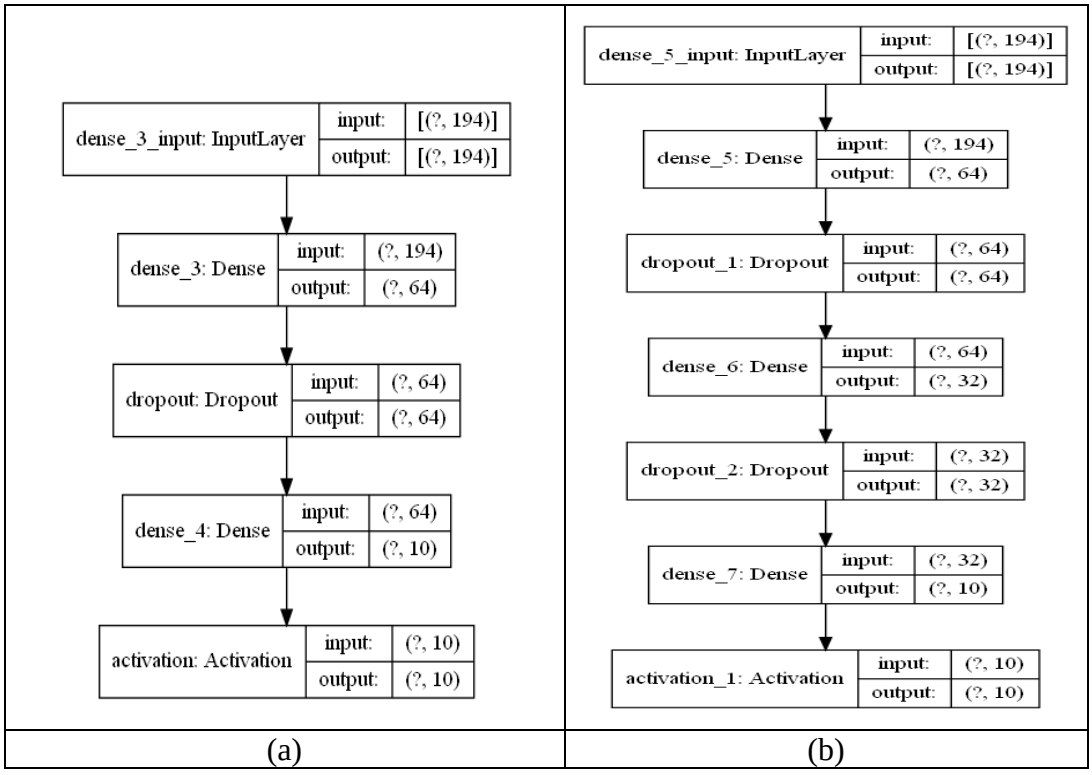


Figure 4. 3. Comparison between the structure of layers in DNN-1 and DNN-2 models: (a) DNN-1 model (b) DNN-2 model

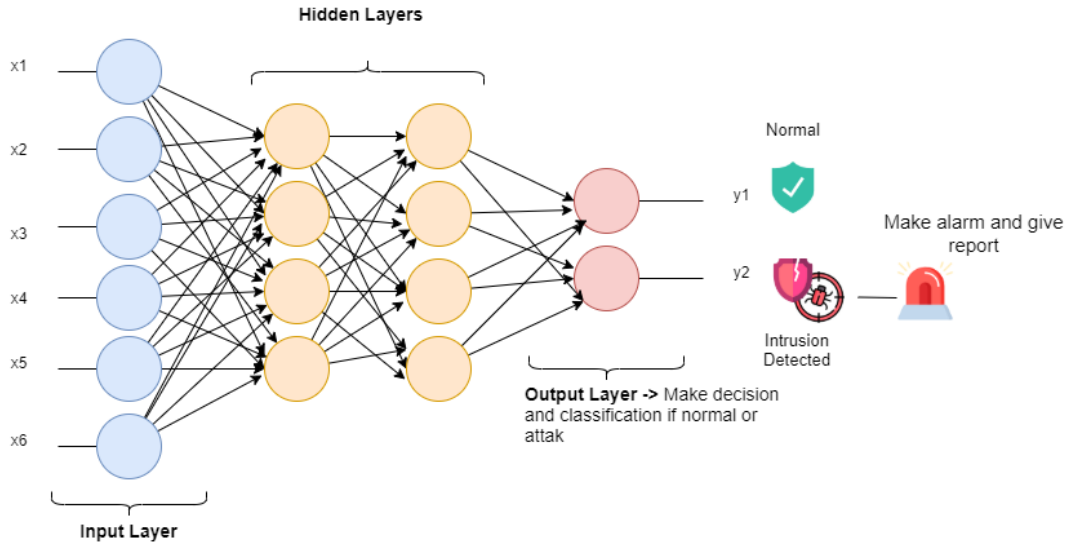


Figure 4. 4. Deep learning neural network model for intrusion detection

4.3.2. Convolutional neural network

Convolutional neural network (CNN) is one of the most powerful models in deep learning. CNN has excellent performance with different applications like image classification, video recognition, action recognition, and natural language processing (NLP). It handles input data as matrices for that we reshaped our input data before feeding it to be more convenient for the CNN module. CNN models have multiple layers, including convolutional layer, pooling layer, non-linearity layer, and fully connected layer [62].

In our work, we used 1D-CNN architecture for intrusion detection tasks while 2D-CNN architecture is mainly used for image processing tasks. In our proposed 1D-CNN architecture, we used a convolutional filter, followed by a max-pooling layer, and at the end of the neural network we added a fully-connected layer to perform the classification. We utilized Softmax function, which determines the probability for each class. Also, we used Dropout techniques that provide generalization before a fully connected layer. Similarly to DNN model, ReLU activation function was used for hidden layers, Softmax activation function was applied in the output layer, categorical cross entropy was used as a loss function, and Adam was used as an optimizer.

We designed two different CNN models and their architectures described consecutively as below in Figure 4.5:

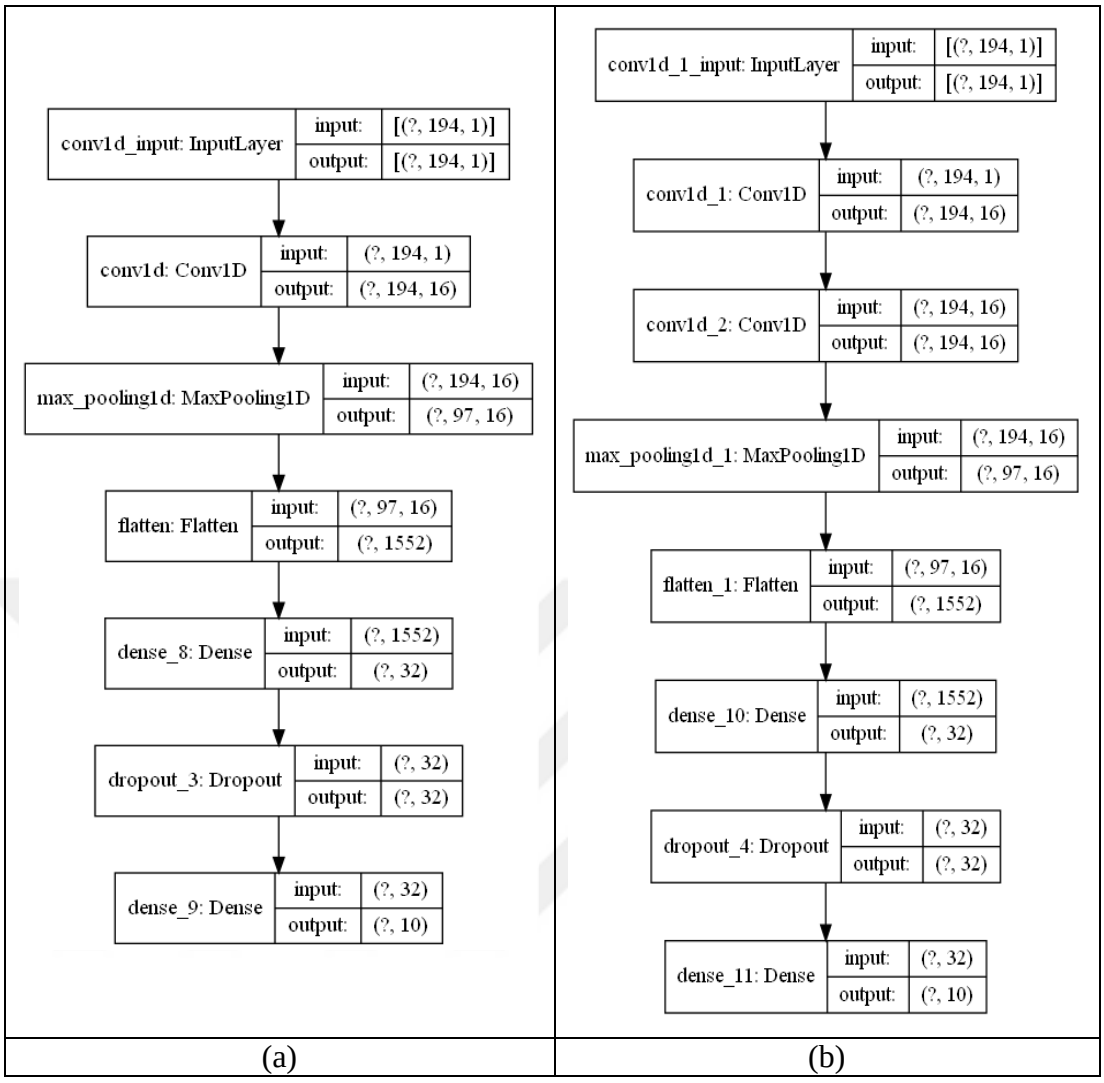


Figure 4. 5. Comparison between the structure of layers in CNN-1 and CNN-2 models: (a) CNN-1 model (b) CNN-2 model

As shown above, we can see that the structure of CNN-2 is more complicated than CNN-1.

4.3.3. Recurrent Neural Network

A recurrent neural network (RNN) is a type of artificial neural network in which nodes' connections form a directed graph through a temporal sequence. RNNs can process variable-length sequences of inputs by utilizing their internal state (memory) [63]. RNN model is used to memorize and remember previous computations. This model allows the use of previous outputs as inputs while maintaining hidden states [64] where for each timestep t, the activation function and the output are stated as follows;

$$a^{<t>} = g_1\left(W_{aa}a^{<t-1>} + W_{ax}x^{<t>}\right) \quad (4.7a)$$
$$\text{and} \quad y^{<t>} = g_2\left(W_{ya}a^{<t>} + b_y\right) \quad (4.7b) \qquad (4.7)$$

Where W_{ax}, W_{aa}, W_{ya}, b_a, b_y are temporally shared coefficients and g_1, g_2 activation functions.

RNN model in some cases can face the long-term dependency problem, the vanishing gradient, and exploding gradient problems. In order to solve these problems, the long short-term memory (LSTM) and gated recurrent unit (GRU) networks have been proposed. The structure of layers in the used RNN model is described in Figure 4.6.

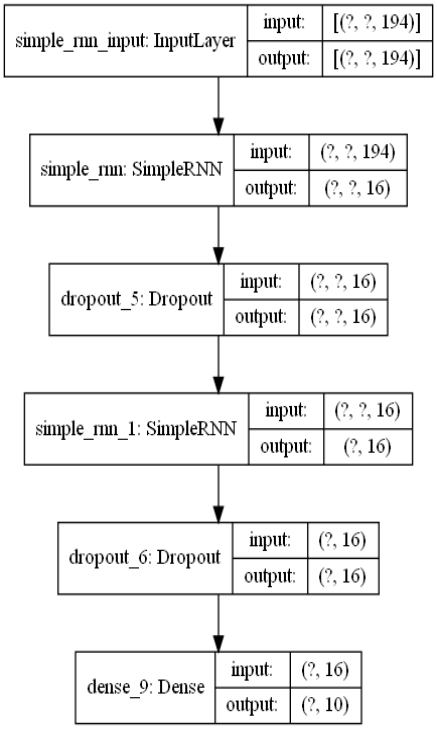
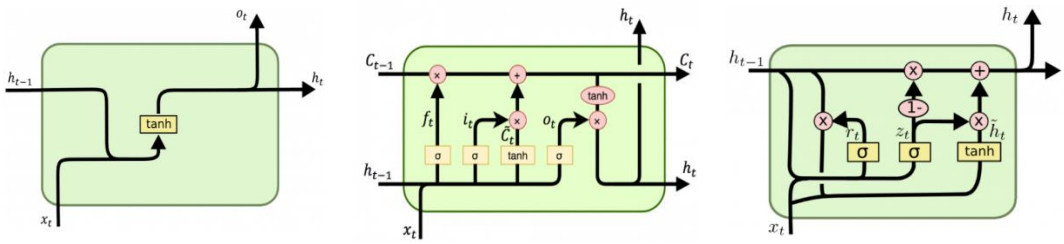


Figure 4. 6. The structure of layers in RNN model



(a) (b) (c)

Figure 4. 7. (a) RNN (b) LSTM (c) GRU models blocks[65]

4.3.4. Long Short-Term Memory

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture. In contrast to standard feedforward neural networks, LSTMs include feedback connections and process the data as sequences [66]. LSTM can be used to perform tasks such as connected handwriting recognition, speech recognition, and anomaly detection in network systems or intrusion detection systems (IDS) [67]. The main difference from a simple RNN is that memory blocks are used in place of nonlinear units in the hidden layers and this model offers three units called gates which are input gate, forget gate, and output gate. The following Equations (4.8) represents the gates in LSTM [68];

$$
\begin{aligned}
i_t &= \sigma\left(w_i[h_{t-1}, x_t] + b_i\right) \quad (4.8a)\\
f_t &= \sigma\left(w_f[h_{t-1}, x_t] + b_f\right) \quad (4.8b)\\
o_t &= \sigma\left(w_o[h_{t-1}, x_t] + b_o\right) \quad (4.8c)
\end{aligned}
\tag{4.8}
$$

Where "i" for the input gate, "f" for the forget gate, "o" for the output gate, "σ" is the Sigmoid function, "b_i" is the biases for the gate(x), "h_{t-1}" is the output of the previous LSTM block, and "x_t" is the current input. The structure of layers that is used in the LSTM model is shown in Figure 4.8.

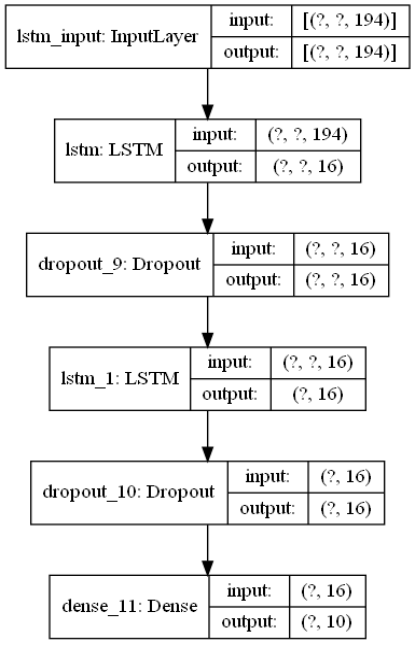


Figure 4. 8. The structure of layers in LSTM model

4.3.5. Gated Recurrent Unit

Gated recurrent units (GRU) are a gating mechanism in recurrent neural networks (RNN). The GRU is similar to a long short-term memory (LSTM) with a forget gate, but requires fewer parameters due to the absence of an output gate [69]. GRU is a simplified version of LSTM and it merges the forget and the input gates into a single "update gate", as well as merges cell and hidden state [70]. It is described by the following Equations (4.9);

$$r_t = \text{sigm}\,(W_{xr}x_t + W_{hr}h_{t-1} + b_r) \quad (4.9a) \qquad z_t = \text{sigm}\,(W_{xz}x_t + W_{hz}h_{t-1} + b_z) \quad (4.9b)$$
$$\tilde{h}_t = \tanh\,(W_{xh}x_t + W_{hh}(r_t * h_{t-1}) + b_h) \quad (4.9c) \qquad h_t = z_t * h_{t-1} + (1 - z_t) * \tilde{h}_t \quad (4.9d) \tag{4.9}$$

The structure of layers that is used in the GRU model is shown in Figure 4.9.

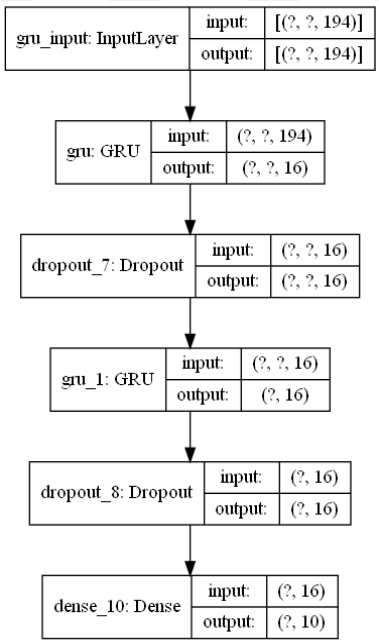


Figure 4. 9. The structure of layers in GRU model

4.3.6. CNN-LSTM Model

CNN-LSTM model is a hybrid model that combines convolutional neural networks (CNN) and long-short term memory (LSTM) networks. This architecture involves using CNN layers for feature extraction from input dataset, followed by an LSTM model to detect intrusions sequentially. In our work, we constructed our own CNN-LSTM model using Python's Keras library, the activation function that is used is ReLU

for CNN model, Tanh for LSTM model and Softmax for the output layer, loss function is sparse categorical cross entropy and the optimizer is Adam; shown as below in Figure 4.10:

```
def cnn_lstm_model():

    cnn_lstm = Sequential()
    cnn_lstm.add(Conv1D(16, 3, padding='same', activation="relu",input_shape=(nb_features, 1)))

    cnn_lstm.add(MaxPooling1D())

    cnn_lstm.add(LSTM(16, return_sequences=True))
    cnn_lstm.add(Dropout(0.01))
    cnn_lstm.add(LSTM(16, return_sequences=False))
    cnn_lstm.add(Dropout(0.01))

    cnn_lstm.add(Dense(n_classes, activation="softmax"))

    cnn_lstm.compile(loss="sparse_categorical_crossentropy", optimizer="adam",metrics=['accuracy'])
    return cnn_lstm
```

Figure 4. 10. Keras library usage for CNN-LSTM model

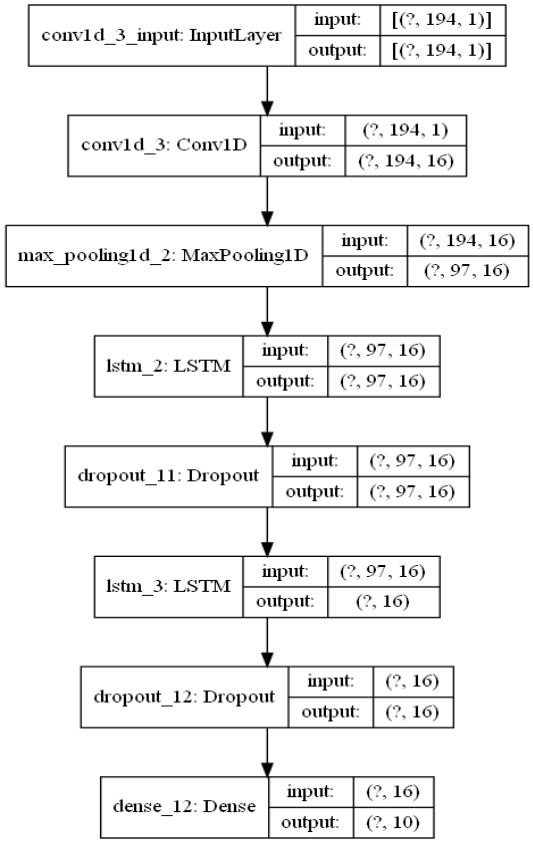


Figure 4. 11. The structure of layers in CNN-LSTM model

4.4. Evaluation Metrics

The applied models are evaluated by defining five performance parameters: accuracy, false alarm rate, precision, recall, and f1 score.

- Accuracy

$$\text{Accuracy} = \frac{\text{TN} + \text{TP}}{\text{FP} + \text{FN} + \text{TP} + \text{TN}} \tag{4.10}$$

- False alarm rate (FAR)

$$\text{FAR} = \frac{\text{FP} + \text{FN}}{\text{FP} + \text{FN} + \text{TP} + \text{TN}} \tag{4.11}$$

- Precision

$$\text{Precion} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.12}$$

- Recall

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.13}$$

- F1 score

$$\text{F1}_{\text{score}} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.14}$$

In addition to the accuracy, we used different evaluation metrics such as precision, recall, and F1 since our dataset is imbalanced. In this case, the accuracy may cause to mislead evaluation of performance in some situations. As we can see from Equation (4.14) F1 score is the harmonic mean value of recall and precision. TP, FP, TN, and FN denote true positive where the model correctly predicts the positive class, false positive where the model incorrectly predicts the positive class, true negative where the model correctly predicts the negative class, and false negative where the model incorrectly predicts the negative class, respectively.

4.5. Cross Validation and Early Stopping Methods

Cross-validation using k-folds splits data into k equally sized subsets, referred to as "fold". One of the k-folds will act as the test set, also known as the validation set or holdout set, while the remaining folds will be used to train the model. After each evaluation, a score is retained, and once all iterations are complete, the scores are averaged to determine the overall model's performance [71]. In our work, we implemented 5-fold stratified cross-validation for both ML models and DL models to give us real intrusion detection accuracy, to handle the imbalance of our dataset.

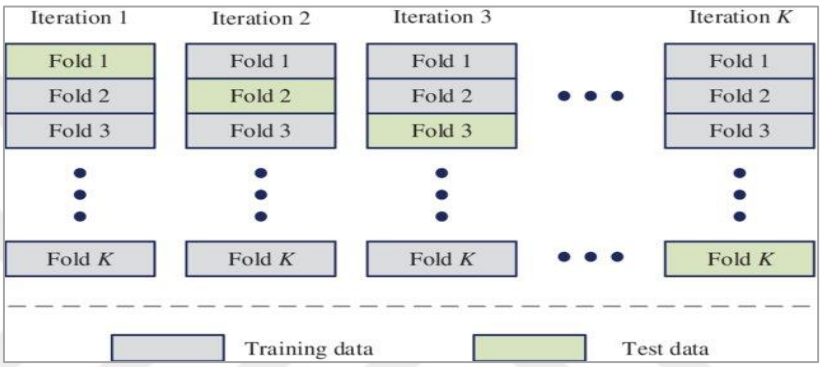


Figure 4. 12. K-fold cross-validation method representation [72]

In deep learning models, we used early stopping method that is a type of regularization used to avoid overfitting when training a learner with an iterative method such a gradient descent. Early Stopping monitors the performance of the model for every epoch on a test validation set during the training and terminates the training based on the validation performance [73], the model will stop the training when the validation error stops decreasing and start to go back up, which means there is no improvement.

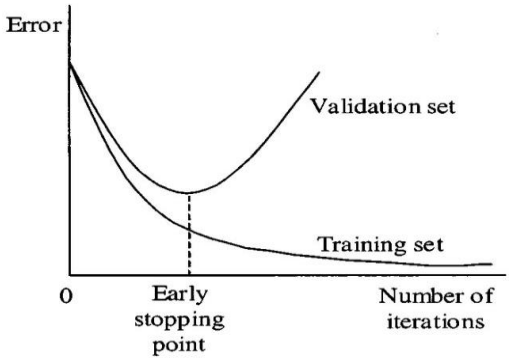


Figure 4. 13. Early stopping based on cross-validation [74]

```
41  max_epochs_value = 100
42  patience_value = 5
43
44  monitor = EarlyStopping(monitor='val_loss', patience=patience_value, verbose=1, mode='min')
45  checkpointer = ModelCheckpoint(filepath="best_weights.hdf5", verbose=0, save_best_only=True)
46
47  model.fit(x_train,y_train_cv,validation_data=(x_test,y_test_cv), callbacks=[monitor, checkpointer], verbose=0, epochs=max_e
48
49  model.load_weights('best_weights.hdf5') # load weights from best epoch
50  print("-"*20)
51  print("Number of epochs that run for the model: ", monitor.stopped_epoch)
52  print("Epochs with the best weights : ", (monitor.stopped_epoch)-patience_value)
53  print("-"*20)
54  epochs_list.append((monitor.stopped_epoch)-patience_value+1)
```

Figure 4. 14. Early stopping usage with Python

4.6. Flow Diagram and Architecture

Flow diagram of this study shown as in flow chart Figure 4.15, we applied the feature selection method only for ML models because in neural networks the important features are chosen automatically. Neural networks are an inherently black box and generate non-linear combinations where features having less discrimination effect will be associated with lesser weights.

We applied different Artificial Intelligence (AI) models: Deep Learning (DL) and Machine Learning (ML) models. Besides that, we handled our data with two different perspectives: binary and multiclass classification.

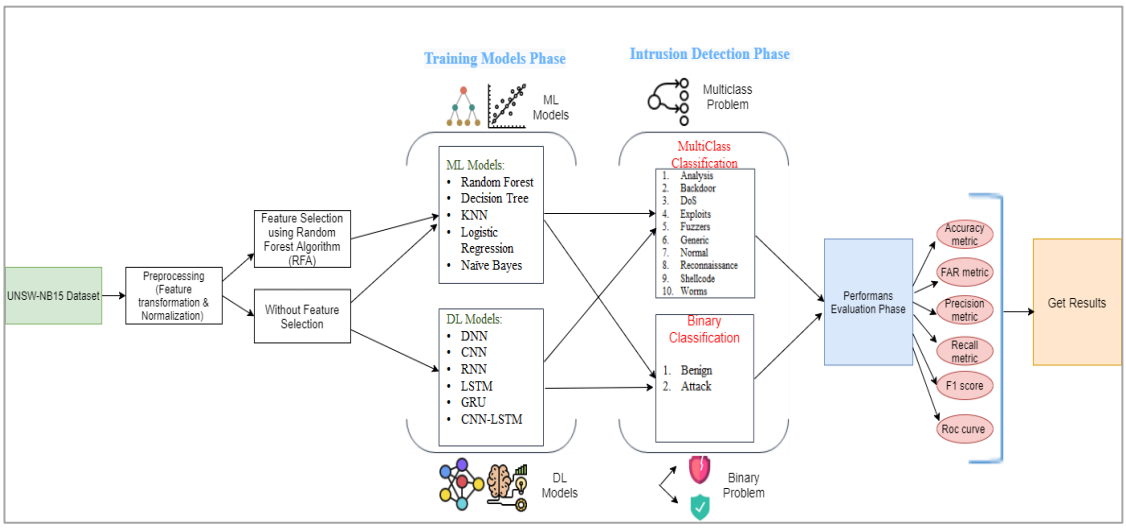


Figure 4. 15. Architecture and flow diagram of proposed work

5. ANALYSIS AND RESULTS

As we can see in Table 5.1 and 5.2 we employed all feature space of UNSW-NB15 dataset for multiclass classification and binary configuration. In Table 5.2 and 5.4 we used reduced feature vectors for multiclass classification and binary setting. In the experimental process, we applied different AI models that contain ML models and DL models. Here in the analysis section, we have 4 stages. In the first step, we tried to compare performance results for ML (binary and multiclass approaches) with full feature space that contains 42 attributes and with a reduced optimal vector (34 features) that generated using the Random Forest method. In the second stage, we tried to compare multiclass classification with binary configurations. In the third phase, we compare performance results between different DL models. In the fourth phase, we tried to compare DL performance results with ML results. In each table, Test Accuracy is the accuracy that obtained on testing data.

From Table 5.1 and 5.2, the performance results demonstrated that the feature selection methodology allows improving accuracies for such models as Random Forest accuracy increased from 75.38% to 75.90% and Decision Tree accuracy increased from 73.43% to 73.86%. These results represented for multiclass classification scheme.

For Table 5.3 machine learning (ML) models for binary classification were applied without feature selection by using all our 42 features and for Table 5.4 our ML models were applied with a random forest feature selection method and we selected the most important 34 features. As we can see from Table 5.3 and 5.4 there is no performance improvement for binary classification tasks with the feature selection technique.

If we tried to compare multiclass classification results with binary classification results either with using the feature selection technique or without it, obviously we can see that accuracies improved for models in binary classification approach. As we can see from Table 5.1 and 5.3 with all 42 features, Random Forest accuracy increased from 75.38% to 87.09%, Decision Tree accuracy increased from 73.43% to 86.36%, KNN

from 70.93% to 84.48%, Logistic Regression from 68.51% to 80.93%, and Naive Bayes from 53.45% to 74.78%. Also, from Table 5.2 and 5.4 with applying the feature selection method, Random Forest accuracy increased from 75.90% to 86.97%, Decision Tree accuracy increased from 73.86% to 86.18%, KNN from 70.59% to 82.10%, Logistic Regression from 67.92% to 80.33%, and Naive Bayes from 54.44% to 73.06%. This is a normal case because when classes number decreased, possible probabilities decrease, and the success rate will be increases and improves.

Table 5. 1. Results using ML models with 42 features – Multiclass classification

ML Model	**Cross Validation Results**				**Evaluation Results on Testing Data**				
	CV Accuracy mean	***CV Precision mean***	***CV Recall mean***	***CV F1 mean***	***Test Accuracy***	***Test FAR rate***	***Test Precision***	***Test Recall***	***Test F1***
Random Forest	82.43	82.30	82.43	81.54	75.38	24.62	83.80	75.38	77.51
Decision Tree	80.94	80.98	80.94	80.55	73.43	26.57	80.75	73.43	76.26
KNN	76.52	76.86	76.52	76.51	70.93	29.07	79.04	70.93	73.85
Logistic Regression	77.07	76.88	77.07	75.25	68.51	31.49	76.94	68.51	69.96
Naive Bayes	63.90	72.83	63.90	64.84	53.45	46.55	74.91	53.45	58.35

Table 5. 2. Results using ML models with 34 features (features selection) – Multiclass classification

ML Model	**Cross Validation Results**				**Evaluation Results on Testing Data**				
	CV Accuracy mean	***CV Precision mean***	***CV Recall mean***	***CV F1 mean***	***Test Accuracy***	***Test FAR rate***	***Test Precision***	***Test Recall***	***Test F1***
Random Forest	82.90	82.75	82.90	81.93	75.90	24.10	83.54	75.90	77.87
Decision Tree	81.08	80.93	81.08	80.53	73.86	26.14	80.52	73.86	76.35
KNN	76.31	76.80	76.31	76.37	70.59	29.41	78.72	70.59	73.53
Logistic Regression	76.34	75.88	76.34	74.39	67.92	32.08	76.04	67.92	69.52
Naive Bayes	65.13	73.69	65.13	65.61	54.44	45.56	72.95	54.44	58.22

Table 5. 3. Results using ML models with 42 features – Binary classification

ML Model	Cross Validation Results				Evaluation Results on Testing Data				
	CV Accuracy mean	*CV Precision mean*	*CV Recall mean*	*CV F1 mean*	*Test Accuracy*	*Test FAR rate*	*Test Precision*	*Test Recall*	*Test F1*
Random Forest	95.95	95.93	95.95	95.93	87.09	12.91	88.84	87.09	86.77
Decision Tree	94.87	94.88	94.87	94.87	86.36	13.64	87.29	86.36	86.13
KNN	93.79	93.76	93.79	93.76	84.48	15.52	86.17	84.48	84.07
Logistic Regression	93.52	93.75	93.52	93.37	80.93	19.07	84.03	80.93	80.07
Naive Bayes	74.63	82.26	74.63	75.47	74.78	25.22	76.68	74.78	74.74

Table 5. 4. Results using ML models with 34 features (features selection) – Binary classification

ML Model	Cross Validation Results				Evaluation Results on Testing Data				
	CV Accuracy mean	*CV Precision mean*	*CV Recall mean*	*CV F1 mean*	*Test Accuracy*	*Test FAR rate*	*Test Precision*	*Test Recall*	*Test F1*
Random Forest	95.87	95.86	95.87	95.85	86.97	13.03	88.82	86.97	86.64
Decision Tree	94.79	94.79	94.79	94.79	86.18	13.82	87.20	86.18	85.94
KNN	93.74	93.71	93.74	93.70	82.10	17.90	83.13	82.10	81.72
Logistic Regression	93.37	93.74	93.37	93.18	80.33	19.67	83.71	80.33	79.38
Naive Bayes	77.65	82.16	77.65	78.38	73.06	26.94	73.32	73.06	73.12

Table 5.5 and 5.6 obtained DL performance results by using full feature space with 42 features for multiclass classification and binary classification, respectively. In deep learning methods, we eliminate the need for features selection since these methods act as black boxes and generate non-linear combinations while the features that have less effect get lesser weight automatically. As introduced in Table 5.5, the best performing models for multiclass classification tasks are the more complicated models and more convenient models for the used dataset, such as DNN-2 with 77.36% accuracy, RNN with 76.69% accuracy, and CNN-LSTM with 76.50% accuracy. Besides that, in Table 5.6 for binary classification tasks, CNN-LSTM with 87.34% accuracy, LSTM with

86.64% accuracy, and RNN with 86.24% achieved better results for intrusion detection in our dataset. Also, as we explained, we know that DNN-2 is more complicated than DNN-1 and the structure of CNN-2 is more convoluted than CNN-1. In Table 5.5, DNN-2 (77.36%) model achieved better accuracy than DNN-1 (74.61%). The same for CNN models in Table 5.6 the results demonstrated that the accuracy increased from 86.98% for CNN-1 to 85.80% for CNN-2.

For the multiclass classification problem, Table 5.5 shows that DNN-1 has 74.61% accuracy and GRU has 72.96% accuracy these models get lower performance results. While for the binary classification task, CNN-2 with 84.91% accuracy and CNN-1 with 84.50% accuracy perform lower accuracies than other DL models and we think this happened with convolutional neural networks because our dataset in the binary classification problem has smaller dimensions, which leads to overfitting problem. In Table 5.5, we can observe that the best performing models run a lower number of epochs with early stopping based on 5-fold cross-validation and this method prevents overfitting problems in the dataset.

To comparison between ML and DL, we can say that DL models achieved higher accuracies than ML models. From Table 5.1 and 5.5, we observed that while accuracies in DL models for multiclass classification between 77.36% and 72.96%; the accuracy results in ML models decreased until 53.45%, and in general accuracies values are between 75.38% and 53.45%.

From Table 5.3 and 5.6, we can see that while accuracies in DL models for binary classification between 87.34% and 84.50%; the accuracy results in ML models decreased until 74.78%, and in general accuracies values are between 87.09% and 74.78%. We observed that random forest and decision tree perform well this is because it is ensemble methods. Ensemble methods are techniques that use multiple classifiers, and then combine them to provide enhanced performance. The predictions of ensemble methods are collected to determine the most often occurring outcome and usually produce more accurate results than a single model. Also, we observed that families of RNN architecture achieved a high accuracy rate in comparison to the traditional machine learning classifiers and the reason for that is the RNN architectures are capable of retraining information over time and maintaining connection sequence data.

As a result, from the experiments, we notice that are more complex models get higher accuracies in some cases. We applied 5-fold cross-validation for both the DL and ML models to get more accurate results about the performance and compared intrusion detection accuracies between different models.

Table 5. 5. Results using DL models with 42 features – Multiclass classification

DL Model	**Cross Validation Results**					**Evaluation Results on Testing Data**				
	Epochs for early stopping based on 5-fold CV	***CV Accuracy mean***	***CV Precision mean***	***CV Recall mean***	***CV F1 mean***	***Test Accuracy***	***Test FAR rate***	***Test Precision***	***Test Recall***	***Test F1***
DNN-2 Model	23, 2, 2, 7, 1	81.55	81.53	81.55	79.36	77.36	22.64	81.26	77.36	77.07
RNN Model	24, 22, 1, 5, 1	80.89	80.84	80.89	78.11	76.69	23.31	79.64	76.69	76.22
CNN-LSTM Model	31, 6, 1, 7, 9	81.41	81.26	81.41	78.66	76.50	23.50	81.25	76.50	76.76
CNN-1 Model	20, 9, 1, 11, 3	79.84	80.36	79.84	76.78	75.93	24.07	79.21	75.93	75.44
LSTM Model	44, 5, 1, 3, 1	81.76	81.49	81.76	79.64	75.83	24.17	79.94	75.83	76.19
CNN-2 Model	24, 2, 3, 3, 5	80.54	79.94	80.54	77.52	74.80	25.20	82.29	74.80	74.92
DNN-1 Model	24, 5, 1, 2, 1	81.10	80.92	81.10	79.37	74.61	25.39	81.41	74.61	75.08
GRU Model	45, 2, 1, 2, 3	81.65	81.66	81.65	79.61	72.96	27.04	80.57	72.96	74.51

Table 5. 6. Results using DL models with 42 features – Binary classification

DL Model	**Cross Validation Results**					**Evaluation Results on Testing Data**				
	Epochs for early stopping based on 5-fold CV	***CV Accuracy mean***	***CV Precision mean***	***CV Recall mean***	***CV F1 mean***	***Test Accuracy***	***Test FAR rate***	***Test Precision***	***Test Recall***	***Test F1***
CNN-LSTM Model	41, 3, 6, 2, 4	94.96	94.95	94.96	94.93	87.34	12.66	89.01	87.34	87.03
LSTM Model	18, 12, 2, 6, 4	94.89	94.87	94.89	94.85	86.64	13.36	88.14	86.64	86.33
RNN Model	28, 5, 5, 12, 2	94.82	94.81	94.82	94.78	86.24	13.76	88.04	86.24	85.88
DNN-1 Model	21, 1, 2, 3, 7	94.71	94.71	94.71	94.66	86.13	13.87	88.13	86.13	85.74
GRU Model	24, 21, 1, 4, 2	94.87	94.85	94.87	94.84	86.05	13.95	87.90	86.05	85.67
DNN-2 Model	25, 1, 2, 1, 3	94.92	94.91	94.92	94.88	86.03	13.97	87.89	86.03	85.66
CNN-2 Model	17, 5, 1, 5, 1	94.70	94.73	94.70	94.63	84.91	15.09	87.73	84.91	84.35
CNN-1 Model	23, 1, 6, 2, 1	94.49	94.56	94.49	94.41	84.50	15.50	87.42	84.50	83.91

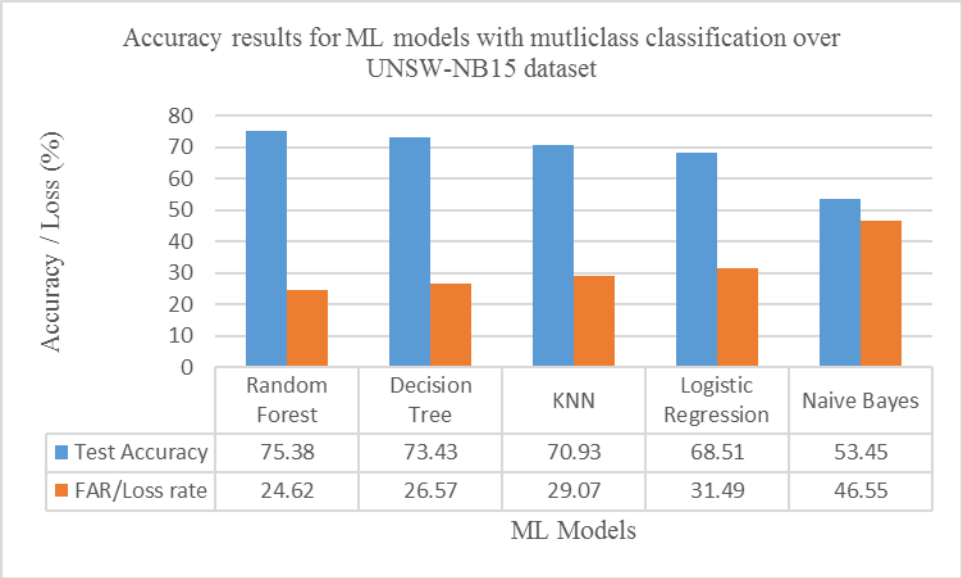


Figure 5. 1. ML models performance over multiclass classification with 42 features

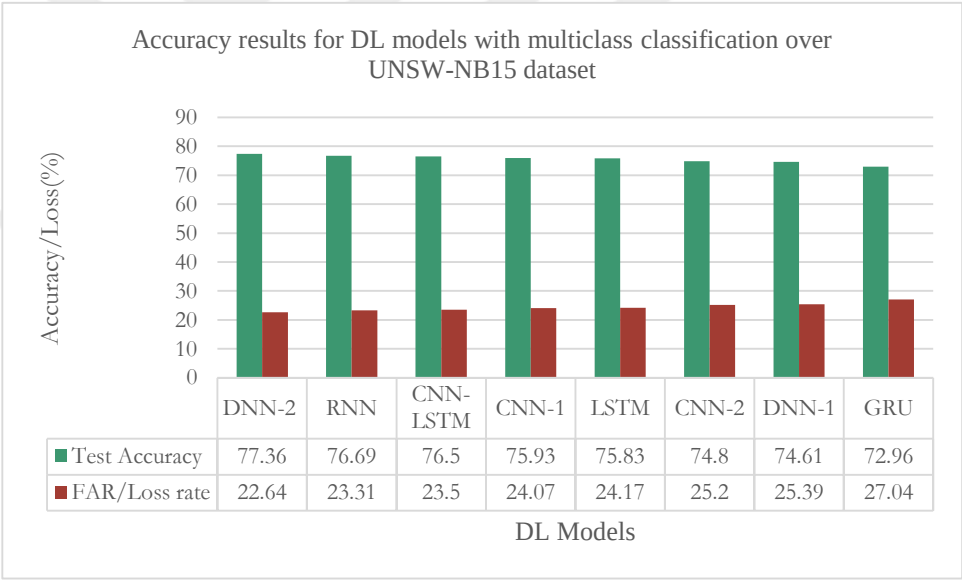


Figure 5. 2. DL models performance over multiclass classification

As we can observe from above Figure 5.1 and Figure 5.2, the traditional machine learning model's performance decreased to 53.45% against the case in the deep learning model the accuracy results decreased to 72.96%. We notice that decision tree and random forest ensemble methods produce improved results similar to deep learning methods and it is a normal case because these techniques aggregate multiple models and predictors to provide results that are more accurate. In addition, we inferred from the experiments that families of RNN architecture achieved a high

accuracy rate in comparison to the traditional machine learning classifiers. The reason for that is that RNN architectures are able to memorize information over time and have connection sequences of information that store previous blocks values. On the other hand, for binary classification, as illustrated in Figure 5.3 and Figure 5.4, the performance of classical machine learning models decreased to 74.78% while the accuracy of deep learning models decreased to 84.50%, which is a higher value than the machine learning model's value.

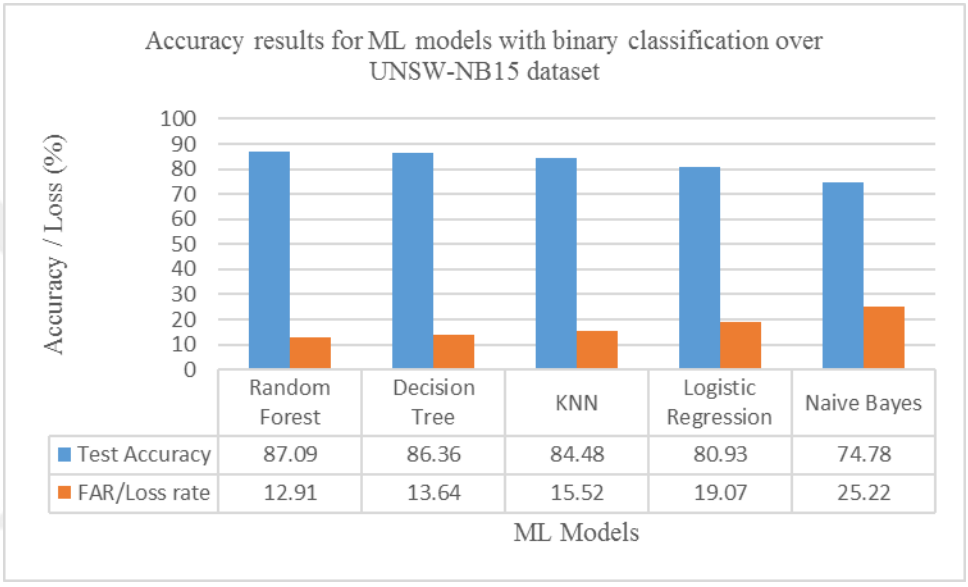


Figure 5. 3. ML models performance over binary classification with 42 features

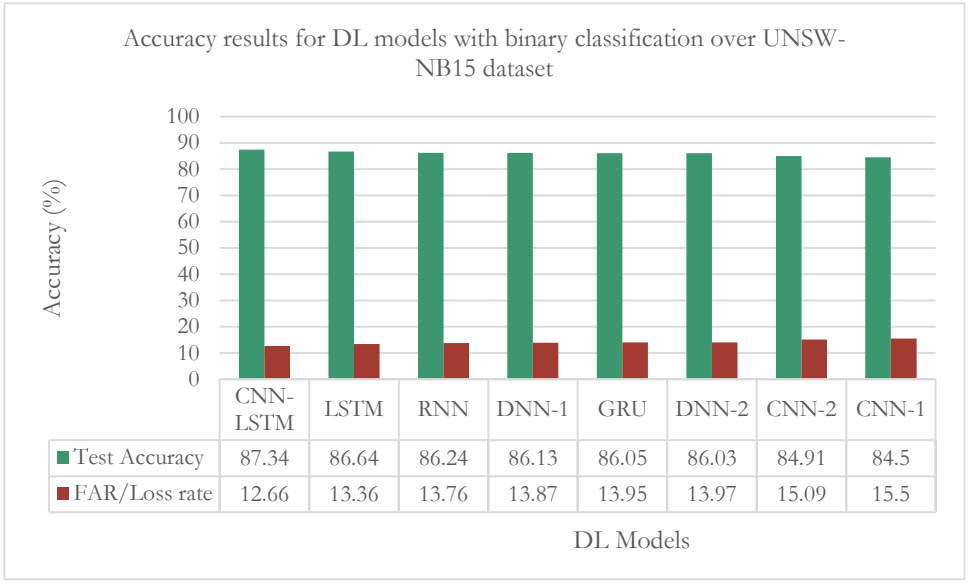


Figure 5. 4. DL models performance over binary classification

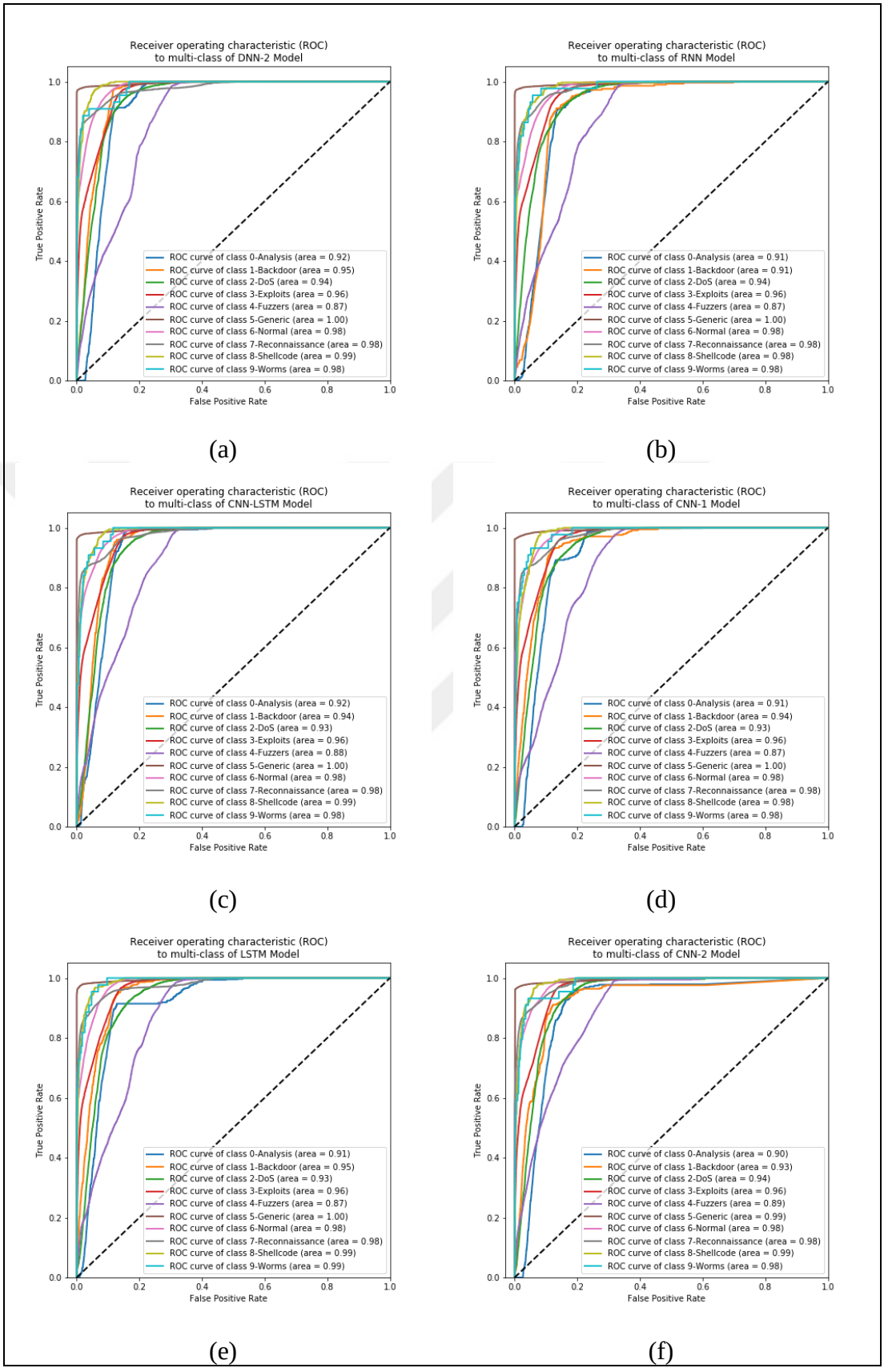
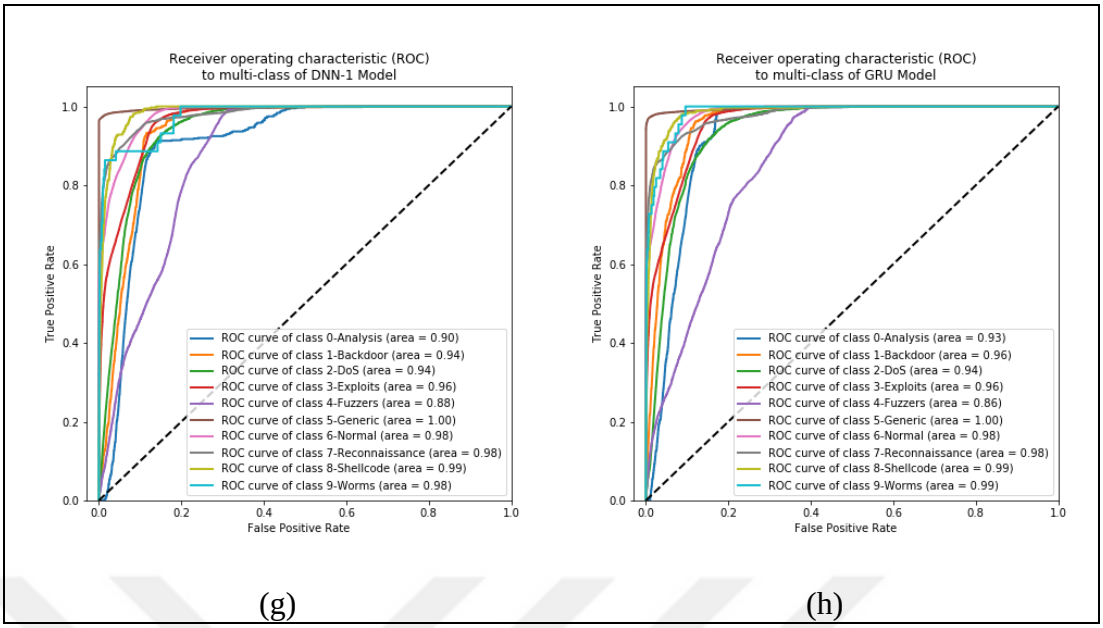
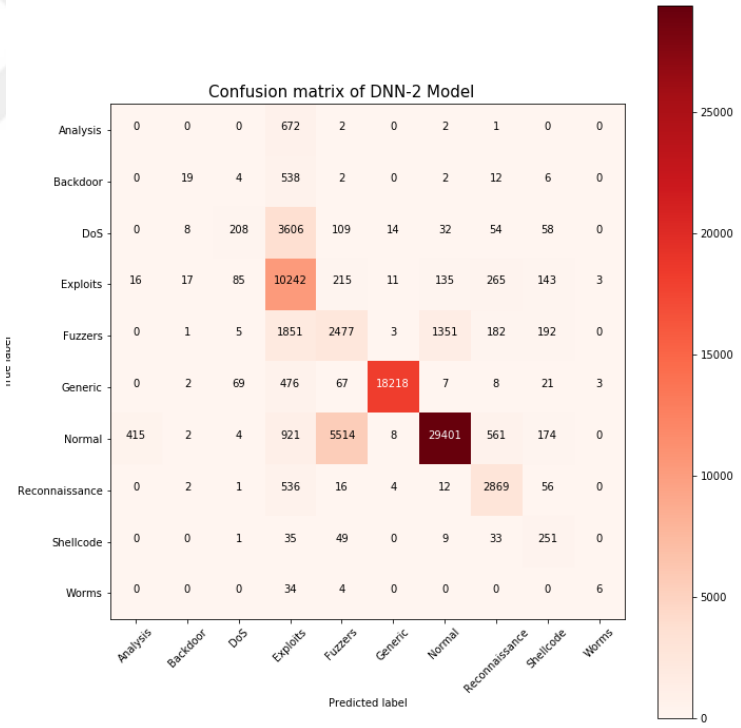


Figure 5. 5. ROC Curves comparison between different DL models for multiclass



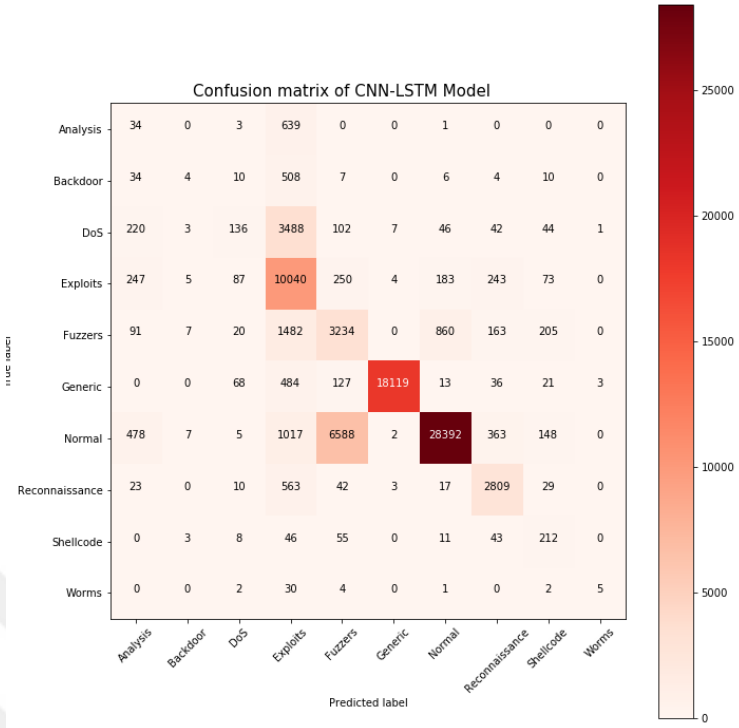
(g) (h)

Figure 5. 5. (Cont.) ROC Curves comparison between different DL models for multiclass



(a)

Figure 5. 6. Confusion matrix for the best-performing DL models: (a) DNN-2 model for multiclass classification (b) CNN-LSTM model for multiclass classification



(b)

Figure 5. 6. (Cont.) Confusion matrix for the best-performing DL models: (a) DNN-2 model for multiclass classification (b) CNN-LSTM model for multiclass classification

As we can observe from Figure 5.7 and Figure 5.8, with the difference that has been found between values of training vs. validation accuracy and loss in CNN-2 model against the little difference between them in DNN-2 and CNN-LSTM models, we can say that DNN-2 and CNN-LSTM models exceed CNN-2 with their performance in the attack detection problem. The difference between values of training vs. validation accuracy and loss can happen because of the lack of data samples we can perform for that data augmentation process in the preprocessing step. There are many ways to prevent overfitting problems, such as dropout layers, random data augmentation, adjusting the numbers of neurons, layers, and units, regularization functions, and other methods. In deep learning methods the problem of overfitting is remaining and to solve it in our study we applied cross-validation and early stopping techniques. Besides the techniques that we used in our work, there are many different methods to avoid overfitting. However, it happens when the model trains for too long on sample data or

the model is too complex, it can start to learn the noise or irrelevant information about the dataset and become unable to generalize well to new data.

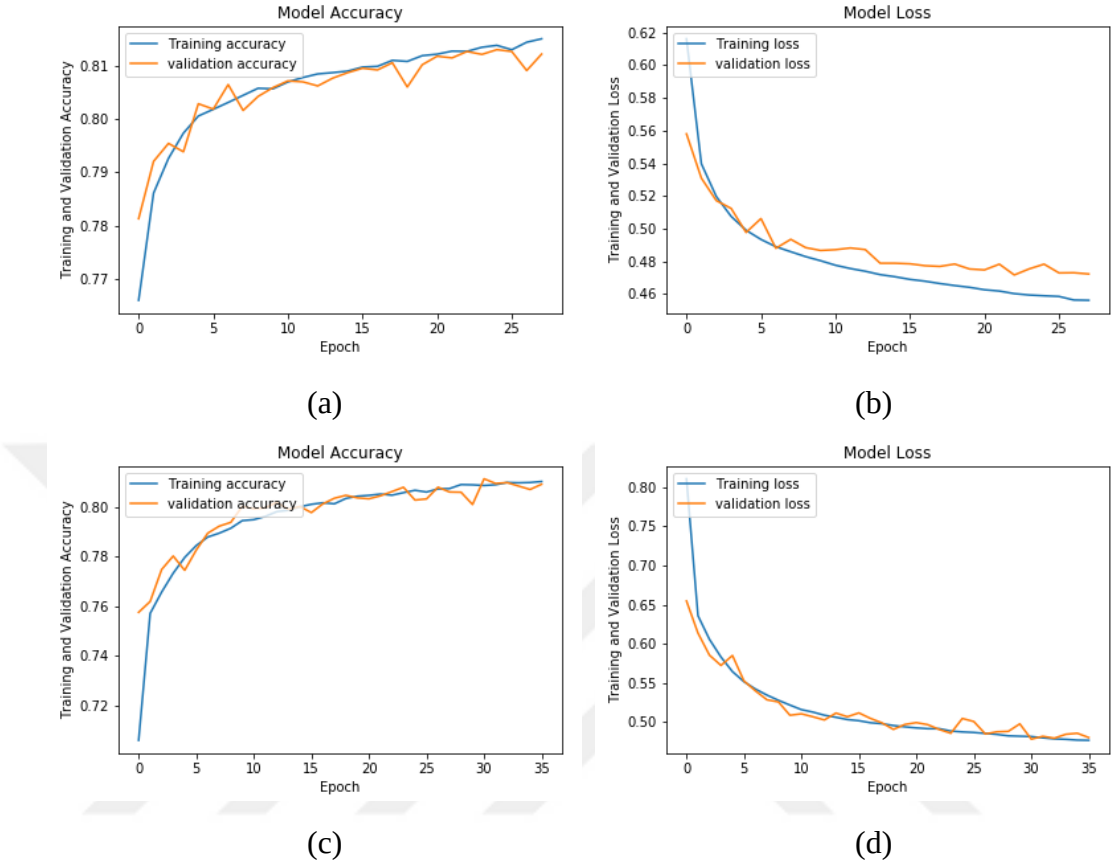


Figure 5. 7. Training vs. validation set accuracy and loss over an increasing number of epochs for the best-performing multiclass classification based DL models: (a) Accuracy in DNN-2 model (b) Loss in DNN-2 model (c) Accuracy in CNN-LSTM model (d) Loss in CNN-LSTM model

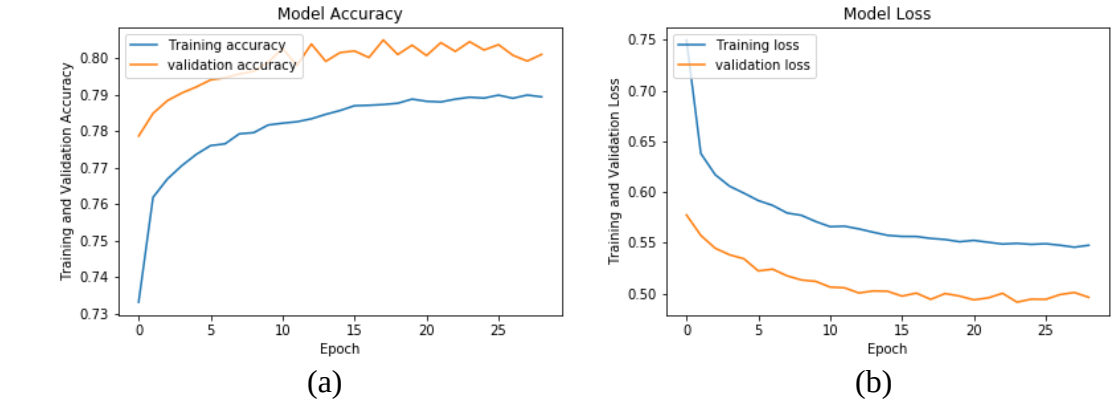
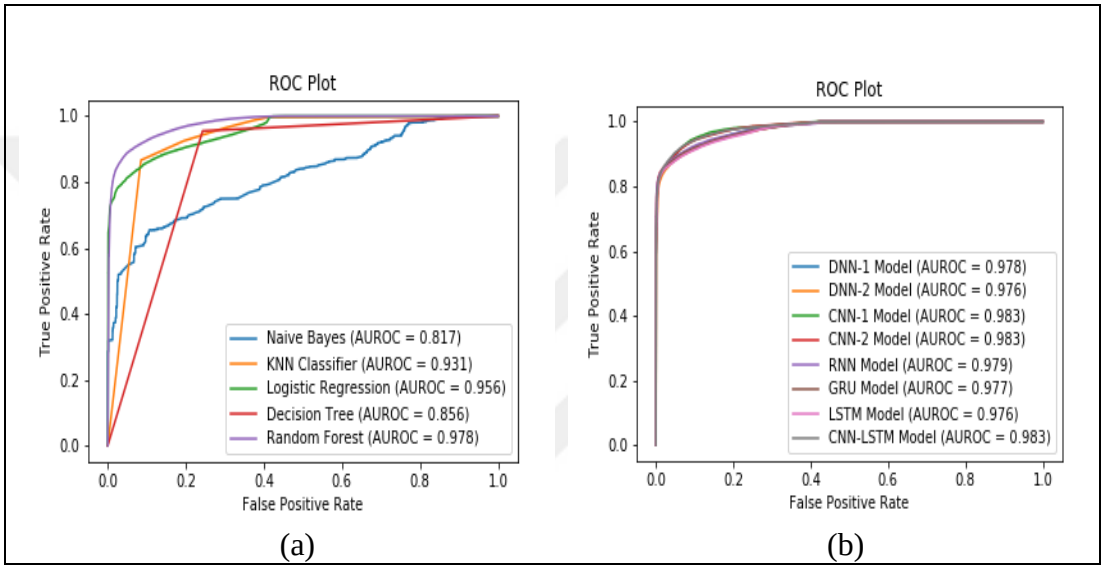


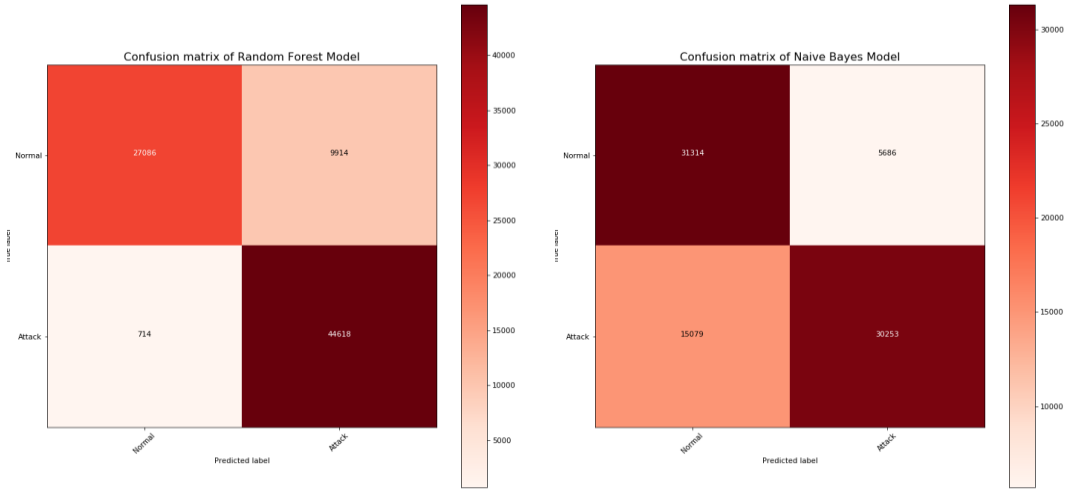
Figure 5. 8. Training vs. validation set accuracy and loss over an increasing iteration for the multiclass classification based DL model that has lower accuracy: (a) Accuracy in CNN-2 model (b) Loss in CNN-2 model

The receiver operating characteristic (ROC) is a graphical representation of a binary classifier system's diagnostic ability as its discrimination threshold is varied. This curve plots two parameters: False Positive Rate (FPR) and True Positive Rate (TPR). As shown in Figure 5.9, we can see the variance in performances between ML and DL models for binary classification problems, the ROC curve of DL models is better than the ROC curve of ML models. DL models are extremely close to each other in AUC-ROC values. On the other hand, there is a difference between the AUC-ROC values of ML models.



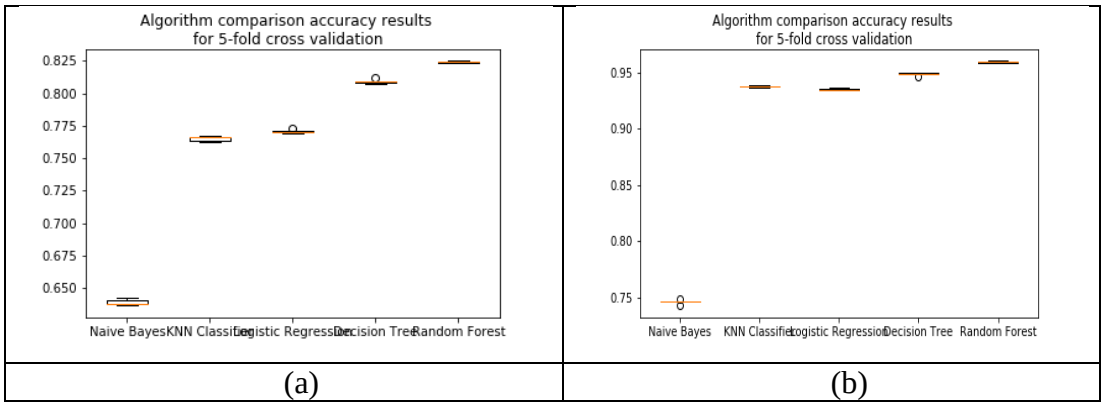
(a) (b)

Figure 5. 9. Comparison between ROC curves for classical ML models and DL models: (a) ROC curve for ML models in binary classification (b) ROC curve for DL models in the binary classification problem



(a) (b)

Figure 5. 7. Confusion matrix of Random Forest and Naïve Bayes models for binary classification problem



(a) (b)

Figure 5. 8. Classical ML models comparison for 5-fold cross validation in multi-class and binary classification problems without feature selection: (a) Binary classification (b) Multiclass classification

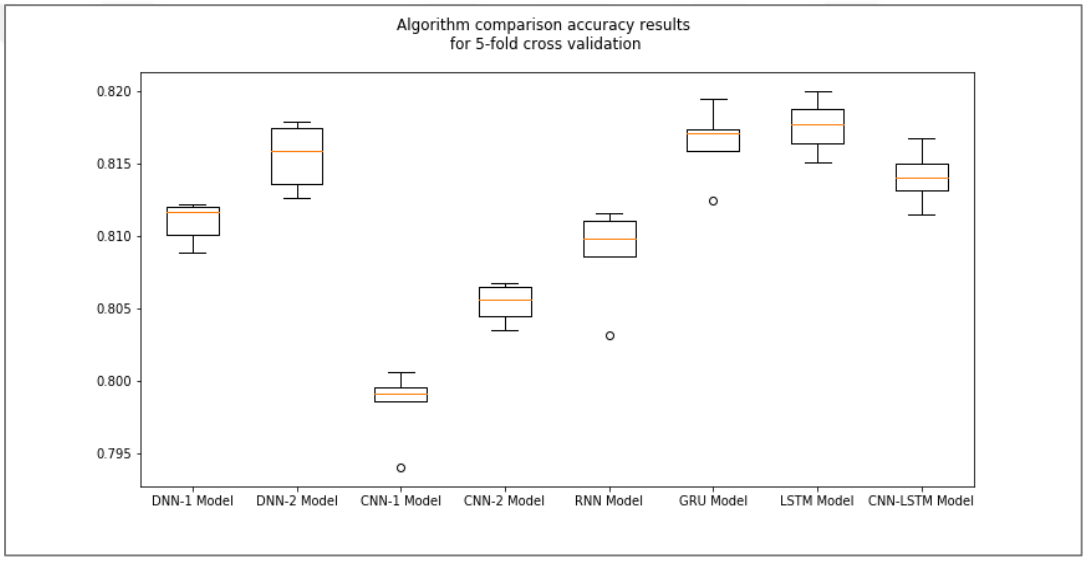


Figure 5. 9. DL models comparison for 5-fold cross validation in multi-class classification problem without feature selection

Implementation of deep learning models depends on different hyperparameters, which are a critical component of any deep network since they enable us to optimize the network's quality. Changes in parameters help to return an optimal model that reduces the predefined loss function and hence improves the accuracy of the results. To identify the performance of the intrusion detection and analysis module, the performance in different situations with different hyperparameters has been compared. In Table 5. 7, 5.8 and 5.9; we implemented different epoch numbers, batch size values, and activation functions in hidden layers over the best-performing multiclass classification based deep learning model which is DNN-2 model, to find the best hyperparameters that could be applied to improve our model's performance.

Table 5. 7. Intrusion detection results using DNN-2 DL model over multiclass classification with different number of epochs

DNN-2 Model	Epochs	Cross Validation Results				Evaluation Results on Testing Data				
		CV Accuracy mean	*CV Precision mean*	*CV Recall mean*	*CV F1 mean*	*Test Accuracy*	*Test FAR rate*	*Test Precision*	*Test Recall*	*Test F1*
* Activation function in hidden layers: ReLU & in output layer: Softmax * Optimizer: adam * Batch size: 32	40	82.09	82.16	82.09	80.09	77.17	22.83	81.57	77.17	78.14
	50	82.06	81.68	82.06	80.32	75.58	24.42	80.29	75.58	76.59
	100	82.25	81.90	82.25	80.47	75.39	24.61	81.81	75.39	76.45
	25	81.95	81.99	81.95	79.98	75.15	24.85	80.82	75.15	76.29
	10	81.86	81.50	81.86	79.98	75.02	24.98	80.97	75.02	76.25

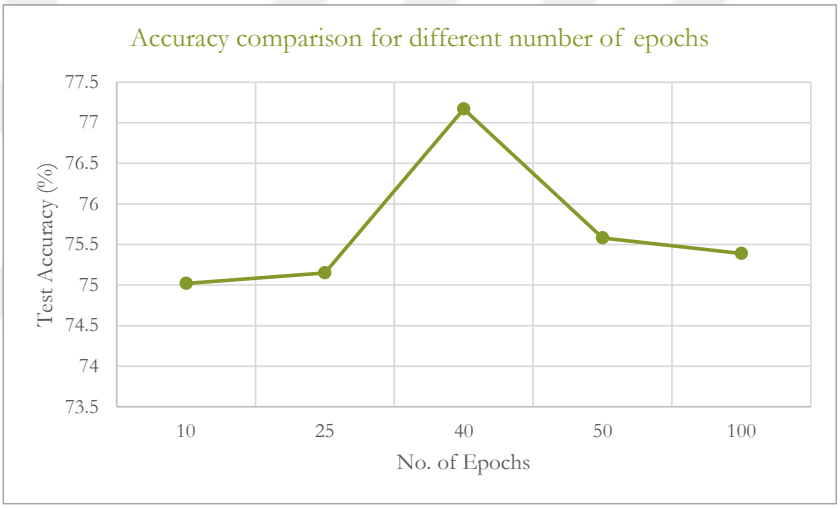


Figure 5. 10. DNN-2 model accuracy vs. number of epochs

Table 5. 8. Intrusion detection results using DNN-2 DL model over multiclass classification with different batch size

DNN-2 Model	Batch Size	Cross Validation Results				Evaluation Results on Testing Data				
		CV Accuracy mean	*CV Precision mean*	*CV Recall mean*	*CV F1 mean*	*Test Accuracy*	*Test FAR rate*	*Test Precision*	*Test Recall*	*Test F1*
* Activation function in hidden layers: ReLU & in output layer: Softmax * Optimizer: adam * Epochs: 40	16	81.78	81.63	81.78	79.84	76.80	23.20	81.42	76.80	77.36
	32	82.05	81.67	82.05	80.06	76.49	23.51	80.77	76.49	77
	64	81.86	81.59	81.86	80.02	75.82	24.18	80.97	75.82	76.92
	48	81.83	81.58	81.83	79.89	75.74	24.26	80.62	75.74	76.31
	128	81.96	81.73	81.96	79.83	75.62	24.38	81.15	75.62	76.73

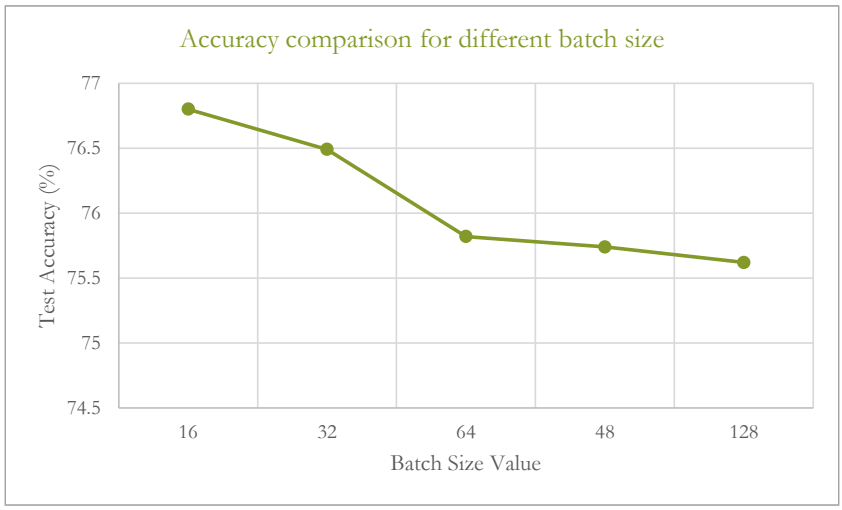


Figure 5. 11. DNN-2 model accuracy vs. batch size

Table 5. 9. Intrusion detection results using DNN-2 DL model over multiclass classification with different activation functions

DNN-2 Model	Activation Function	Cross Validation Results				Evaluation Results on Testing Data				
		CV Accuracy mean	*CV Precision mean*	*CV Recall mean*	*CV F1 mean*	*Test Accuracy*	*Test FAR rate*	*Test Precision*	*Test Recall*	*Test F1*
* Activation function in output layer: Softmax * Optimizer: adam * Epochs: 40 * Batch size: 16	Sigmoid	82.10	82.37	82.10	79.68	76.69	23.31	81.43	76.69	76.88
	Tanh	81.58	81.66	81.58	79.52	76.12	23.88	80.14	76.12	76.73
	ReLU	81.71	82.40	81.71	80.04	73.74	26.26	81.70	73.74	76.59

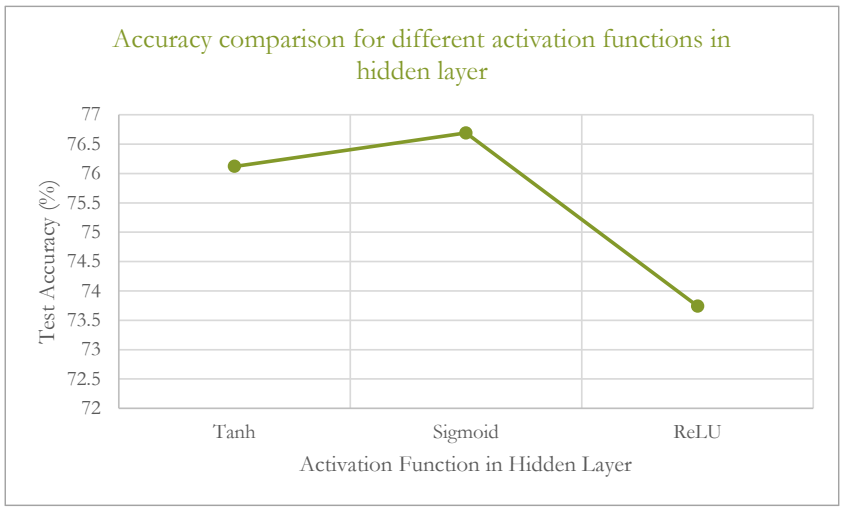


Figure 5. 12. DNN-2 model accuracy vs. activation function in hidden layers

From Table 5.7, we can observe that epochs number 40 will be a better choice for the DNN-2 model with 77.17% test accuracy and 78.14% test F1. Because of that, we chose it and then implemented it with different batch size values (16, 32, 48, 64, 128). As shown in Table 5.8, it indicates that the batch size value of 16 achieves better results with 76.80% test accuracy and 77.36% test F1. With epochs number 40 and batch size 16 we also applied different activation functions in hidden layers such as Sigmoid, Tanh, and ReLU. From Table 5.9 we notice that the Sigmoid activation function performs better with 76.69% test accuracy and 76.88% test F1.

Besides comparison between accuracy and different epoch numbers, batch size values, and activation functions in hidden layers over deep learning methods, we could also use different optimization algorithms, learning rate, batch number, number of neurons, and number of hidden layers.

6. CONCLUSION AND RECOMMENDATIONS

In this research, we applied the following supervised Machine Learning (ML) methods for IDS: Naïve Bayes (NB), k-Nearest-Neighbor (kNN), Logistic Regression (LR), Decision Tree (DT) and Random Forest (RF). Also, Deep Learning (DL) methodologies such as: Deep Neural Network (DNN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), and CNN-LSTM model. Traditional machine learning methods depend heavily on feature engineering, which is often time-consuming and complex and with the complexity of the IoT structure, it is critical to develop an IDS that achieves low computational costs and reduces the amount of energy consumed. As a result, it is impractical to detect anomalies in real-time using classical machine learning models. For that in our work, we also applied deep learning methods that are used to generate non-linear combinations, where the features that have a lesser effect are automatically given a lower weight. Since our data are labeled we employed supervised deep learning methods. We applied the Random Forest algorithm over UNSW-NB15 dataset to calculate the feature importance measure for each feature, generate reduced optimal feature vectors. We considered two schemes binary and multiclass classification configurations. We compared different models using their performance results and accuracies. We can conclude that deep learning methods exceed traditional methods with their performance in the attack detection tasks, also feature selection methods can enhance performance in some classical machine learning models.

In the future work, we can compare the experimental results of UNSW-NB15 dataset with other datasets like Bot-IoT, CIC-IDS2018, and N-BaIoT datasets or we can create our own dataset using simulation tools. We aim to be more creative in intrusion detection methods and increase accuracies using hybrid models by combining blockchain with deep learning algorithms. We think to add an intrusion prevention system (IPS) to IDS, this technology used to prevent and mitigate attacks and drop the malicious packets and threats.

REFERENCES

[1] Patel K. K., Patel S. M., Internet of things-IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application & Future Challenges, *International Journal of Engineering Science and Ccomputing*, 2016, **6**(5), 6122-6131.

[2] Khraisat A., Gondal I., Vamplew P., Kamruzzaman J., Survey of Intrusion Detection Systems: Techniques, Datasets and Challenges, *Cybersecurity*, DOI:10.1186/s42400-019-0038-7.

[3] Internet of Things, Wikipedia, https://en.wikipedia.org/wiki/Internet_of_ things, (Accessed: 2 June 2021).

[4] 8.4 Billion Connected Things Will be in Use 2017 | Gartner, Gartner newsroom, https://www.gartner.com/en/newsroom/press-releases/2017-02-07-gartner-says-8-billion-connected-things-will-be-in-use-in-2017-up-31-percent-from-2016, (Accessed: 2 June 2021).

[5] Zhang Z.-K., Cho M. C. Y., Wang C.-W., Hsu C.-W., Chen C.-K., Shieh S., IoT Security: Ongoing Challenges and Research Opportunities, *IEEE 7th International Conference on Service-Oriented Computing and Applications*, Japan, 17-19 November 2014.

[6] Khan M. A., Salah K., IoT Security: Review, Blockchain Solutions, and Open Challenges, *Future Generation Computer Systems*, 2018, **82**, 395–411.

[7] Kasongo S. M., Sun Y., Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset, *Journal of Big Data*, DOI:10.1186/s40537-020-00379-6.

[8] Gaudio P., *Cyber and Chemical, Biological, Radiological, Nuclear, Explosives Challenges*, Springer International Publishing, Cham, 2017.

[9] Samonas S., Coss D., The CIA Strikes Back: Redefining Confidentiality, Integrity and Availability in Security, *Journal of Information System Security*, 2014, **10**(3), 21-45.

[10] ISO/IEC 27000, Information Technology-Security Techniques-Information Security Management Systems-Overview and Vocabulary, ISO/IEC International Standards Organization, Switzerland Geneva, 2018.

[11] Stamp M., *Information Security: Principles and Practice*, Wiley Publishing, New Jersey, 5005.

[12] Qadir S., Quadri S. M. K., Information Availability: An Insight into the Most Important Attribute of Information Security, *Journal of Information Security,* 2016, **7**(3), 185–194.

[13] Jyothsna V., Prasad V. V. R., Prasad K. M., A Review of Anomaly Based Intrusion Detection Systems, *International Journal of Computer Applications,* 2011, **28**(7) 26–35.

[14] Singh A. P., Singh M. D., Analysis of Host-Based and Network-Based Intrusion Detection System, *International Journal of Computer Network & Information Security,* 2014, **6**(8), 41-47.

[15] Jose S., Malathi D., Reddy B., Jayaseeli D., A Survey on Anomaly Based Host Intrusion Detection System, *National Conference on Mathematical Techniques and its Applications,* India, 5-6 January 2018.

[16] Gondaliya T., Joshi H. D., Joshi H. J., Different Toolsand Types of Intrusion Detection System with Network Attacks " T&T-IDSys " -A Review, *2nd International Conference on Multidisciplinary Research & Practice*, India, 24 December 2015.

[17] Nyamugudza T., Rajasekar V., Sen P., Nirmala M., Viswanatham V. M., Network Traffic Intelligence Using a Low Interaction Honeypot, *14th IOP Conference Ser. Material Science and Engineering,* Tamil Nadu, India, 2-3 May 2017.

[18] Ashoor A. S., Gore S., Difference between Intrusion Detection System (IDS) and Intrusion Prevention System (IPS), 4th *International Conference on Network Security and Applications,* Chennai, India, 15-17 July 2011.

[19] Hadi H. J., Shnain A. H., Hadishaheed S., Ahmad A. A., Big data and Five V's Characteristics, *International Journal of Advances in Electronics and Computer Science,* 2015, **2**(1), 16–23.

[20] Shahrivari S., Beyond Batch Processing: Towards Real-Time and Streaming Big Data, *Computers,* 2014, **3**(4), 117–129.

[21] Cloud Computing, Wikipedia, https://en.wikipedia.org/wiki/Cloud_computing (Accessed: 2 June 2021).

[22] Montazerolghaem A., Yaghmaee M. H., Leon-Garcia A., Green Cloud Multimedia Networking: NFV/SDN Based Energy-Efficient Resource Allocation, *IEEE Transactions on Green Communications and Networking,* DOI: 10.1109/TGCN.2020.2982821.

[23] Khraisat A., Alazab A., A Critical Review of Intrusion Detection Systems in the Internet of Things: Techniques, Deployment Strategy, Validation Strategy, Attacks, Public Datasets and Challenges, *Cybersecurity,* DOI: 10.1186/s42400-021-00077-7.

[24] Khraisat A., Gondal I., Vamplew P., Kamruzzaman J., Survey of Intrusion Detection Systems: Techniques, Datasets and Challenges, *Cybersecurity,* DOI: 10.1186/s42400-019-0038-7.

[25] Machine Learning,Wikipedia, https://en.wikipedia.org/wiki/Machine_learning #Approaches, (Accessed: 30 May 2021).

[26] Outline of Machine Learning, Wikipedia, https://en.wikipedia.org/wiki/Outline of_machine_learning#Applications_of_machine_learning, (Accessed: 30 May 2021).

[27] Rashid M. M., Kamruzzaman J., Hassan M. M., Imam T., Gordon S., Cyberattacks Detection in IoT-Based Smart City Applications Using Machine Learning Techniques, *International Journal of Environmental Research and Public Health,* DOI: 10.3390/ijerph17249347.

[28] Ullah I., Mahmoud Q. H., A Two-Level Flow-Based Anomalous Activity Detection System for IoT Networks, *Electronics,* DOI: 10.3390/electronics9030530.

[29] Alsamiri J., Alsubhi K., Internet of Things Cyber Attacks Detection Using Machine Learning, *International Journal of Advanced Computer Science and Applications,* DOI: 10.14569/ijacsa.2019.0101280.

[30] Soe Y. N., Feng Y., Santosa P. I., Hartanto R., Sakurai K., Machine Learning-Based IoT-Botnet Attack Detection with Sequential Architecture, *Sensors,* DOI: 10.3390/s20164372.

[31] Deep learning, Wikipedia, https://en.wikipedia.org/wiki/Deep_learning, (Accessed: 30 May 2021).

[32] Deng L., Yu D., Deep learning: Methods and Applications, *Foundations and Trends in Signal Processing*, DOI: 10.1561/2000000039.

[33] Khan F. A., Gumaei A., Derhab A., Hussain A., A Novel Two-Stage Deep Learning Model for Efficient Network Intrusion Detection, *IEEE Access*, DOI: 10.1109/ACCESS.2019.2899721.

[34] Yang Y., Zheng K., Wu C., Yang Y., Improving the Classification Effectiveness of Intrusion Detection by Using Improved Conditional Variational Autoencoder and Deep Neural Network, *Sensors,* DOI: 10.3390/s19112528.

[35] Thamilarasu G., Chawla S., Towards Deep-Learning-Driven Intrusion Detection for the Internet of Things, *Sensors,* DOI: 10.3390/s19091977.

[36] Basumallik S., Ma R., Eftekharnejad S., Packet-data Anomaly Detection in PMU-based State Estimator Using Convolutional Neural Network, *International Journal of Electronic Power Energy & Energy Systems*, DOI: 10.1016/j.ijepes.2018.11.013.

[37] Zeng Y., Gu H., Wei W., Guo Y., Deep-Full-Range: A Deep Learning Based Network Encrypted Traffic Classification and Intrusion Detection Framework, *IEEE Access,* DOI: 10.1109/ACCESS.2019.2908225.

[38] Ferrag M. A., Maglaras L., Moschoyiannis S., Janicke H., Deep Learning for Cyber Security Intrusion Detection: Approaches, Datasets, and Comparative Study, *Journal of Information Security and Applications*, DOI: 10.1016/j.jisa.2019.102419.

[39] Vinayakumar R., Soman K.P., Poornachandran P., Evaluation of Recurrent Neural Network and its Variants for Intrusion Detection System (IDS), *International Journal of Information System Modeling and Design,* DOI: 10.4018/IJISMD.2017070103.

[40] Popoola S. I., Adebisi B., Ande R., Hammoudeh M., Anoh K., Atayero A. A., SMOTE-DRNN: A Deep Learning Algorithm for Botnet Detection in the Internet-of-Things Networks, *Sensors,* DOI: 10.3390/s21092985.

[41] Liang C., Shanmugam B., Azam S., Karim A., Islam A., Zamani M., Kavianpour S., Idris N.B., Intrusion Detection System for the Internet of Things Based on Blockchain and Multi-Agent Systems, *Electronics,* DOI: 10.3390/electronics9071120.

[42] Ibitoye O., Shafiq O., Matrawy A., Analyzing Adversarial Attacks against Deep Learning for Intrusion Detection in IoT Networks, *IEEE Global Communications Conference,* HI, USA, 9-13 December 2019.

[43] Ge M., Fu X., Syed N., Baig Z., Teo G., Robles-Kelly A., Deep Learning-Based Intrusion Detection for IoT Networks, IEEE 24th Pacific Rim International Symposium on Dependable Computing, Kyoto, Japan, 1-3 December 2019.

[44] Zheng A., Casari A., *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists,* O'Reilly Media, Inc., 2018.

[45] Pajouh H. H., Javidan R., Khayami R., Dehghantanha A., Choo K.-K. R., A Two-Layer Dimension Reduction and Two-Tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks, *IEEE Transactions on Emerging Topics in Computing*, DOI: 10.1109/TETC.2016.2633228.

[46] Liu H., Lang B., Liu M., Yan H., CNN and RNN Based Payload Classification Methods for Attack Detection, *Knowledge-Based Systems*, DOI: 10.1016/j.knosys.2018.08.036.

[47] Zarpelão B. B., Miani R. S., Kawakani C. T., de Alvarenga S. C., A Survey of Intrusion Detection in Internet of Things, *Journal of Network and Computer Applications, DOI: 10.1016/j.jnca.2017.02.009*.

[48] Moustafa N., Slay J., UNSW-NB15 : A Comprehensive Data Set for Network Intrusion Detection Systems, *Military Communications and Information Systems Conference,* Canberra, ACT, Australia, 10-12 November 2015.

[49] Beklemysheva A., Why Use Python for AI and Machine Learning, Steel Kiwi, https://steelkiwi.com/blog/python-for-ai-and-machine-learning/, (Accessed: 2 June 2021).

[50] Matplotlib: Visualization with Python, Matplotlib 3.4.2 Documentation, https://matplotlib.org/, (Accessed: 2 June 2021).

[51] TensorFlow, https://www.tensorflow.org/, (Accessed: 2 June 2021).

[52] Keras, Wikipedia, https://en.wikipedia.org/wiki/Keras, (Accessed: 2 June 2021).

[53] Chandrashekar G., Sahin F., A Survey on Feature Selection Methods, *Computers & Electrical Engineering*, DOI: 10.1016/j.compeleceng. 2013.11.024.

[54] Kaviani P., Dhotre S., Short Survey on Naive Bayes Algorithm, *International Journal of Advance Engineering and Research Development*, 2017, **4**(11), 607-611.

[55] Chomboon K., Chujai P., Teerarassammee P., Kerdprasop K., Kerdprasop N., An Empirical Study of Distance Metrics for k-Nearest Neighbor Algorithm, *3rd International Conference on Industrial Application Engineering,* Kitakyushu, Japan, 28-31 March 2015.

[56] Boateng E. Y., Abaye D. A., A Review of the Logistic Regression Model with Emphasis on Medical Research, *Journal of Data Analysis and Information Processing*, DOI: 10.4236/jdaip.2019.74012.

[57] Safavian S. R., Landgrebe D., A Survey of Decision Tree Classifier Methodology, *IEEE Transactions on Systems, Man, and Cybernetics,* DOI: 10.1109/21.97458.

[58] Breiman L. E. O., Random Forests, *Machine Learning,* 2001, **45**(1), 5–32.

[59] Isaac O., Jantan A., Esther A., State-of-the-art in Artificial Neural Network Applications : A survey, *Heliyon,* DOI: 10.1016/j.heliyon.2018.e00938.

[60] Jonathan J., What's a Deep Neural Network? Deep Nets Explained, BMC Software Blogs, https://www.bmc.com/blogs/deep-neural-network/, (Accessed: 10 March 2021).

[61] Manaswi N. K., *Deep learning with Applications Using Python: Chatbots and Face, Object, and Speech Recognition with Tensorflow and Keras,* Apress, Berkeley, CA, 2018.

[62] Albawi S., Mohammed T. A., Al-Zawi S., Understanding of a Convolutional Neural Network, *International Conference on Engineering and Technology*, Antalya, Turkey, 21-23 August 2017.

[63] Recurrent Neural Network, Wikipedia, https://en.wikipedia.org/wiki/Recurrent_neural_network, (Accessed: 23 June 2021).

[64] Amidi S., Amidi A., Recurrent Neural Networks Cheatsheet, Stanford Education, https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks#overview, (Accessed: 19 August 2021).

[65] Recuurent Neural Network (RNN)-Long Short Term Memory (LSTM) & Gated Recurrent Unit (GRU), http://dprogrammer.org/rnn-lstm-gru, (Accessed: 23 June 2021).

[66] Hochreiter S., Schmidhuber J., Long Short-Term Memory, Neural Computation, DOI: 10.1162/neco.1997.9.8.1735.

[67] Long Short-Term Memory, Wikipedia, https://en.wikipedia.org/wiki/Long_short-term_memory#cite_note-lstm1997-1, (Accessed: 23 June 2021).

[68] Thakur D., LSTM and its Equations, Medium, https://medium.com/@divyanshu132/lstm-and-its-equations-5ee9246d04af, (Accessed: 19 August 2021).

[69] Gated Recurrent Unit - Wikipedia, https://en.wikipedia.org/wiki/Gated_recurrent_unit, (Accessed: 23 June 2021).

[70] Jozefowicz R., Zaremba W., Sutskever I., An Empirical Exploration of Recurrent Network Architectures, *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France, 6-11 July 2015.

[71] Kudale P., K-Fold Cross Validation, Analytics Vidhya Medium, https://medium.com/analytics-vidhya/k-fold-cross-validation-d4b7e8777038, (Accessed: 24 June 2021).

[72] What is Overfitting, IBM Cloud Education, https://www.ibm.com/cloud/learn/overfitting, (Accessed: 24 June 2021).

[73] Early Stopping, Wikipedia, https://en.wikipedia.org/wiki/Early_stopping, (Accesse: 24 June 2021).

[74] Gençay R., Qi M., Pricing and Hedging Derivative Securities with Neural Networks: Bayesian Regularization, Early Stopping, and Bagging, *IEEE Transactions on Neural Networks*, DOI: 10.1109/72.935086.

PERSONAL PUBLICATIONS

Amarouche S., Küçük K., Comparison of Classical ML with DL Methods for Network Intrusion Detection in IoT, *2nd International Symposium on Applied Sciences and Engineering (ISASE)*, Erzurum, Turkey, 7-9 April 2021.

AUTOBIOGRAPHY

Siham Amarouche completed her high school education in Saudi Arabia at Jizan in 2012. She earned her Bsc. degree in Computer Engineering from Kocaeli University in the year 2018. During the education journey for a master's degree, she published a paper with the topic "Comparison of Classical ML with DL Methods for Network Intrusion Detection in IoT" also she is awarded to work at the Tübitak project for a one-year 2019-2020. She is currently a Master's student at the computer engineering department of Kocaeli University in Turkey. Her current research areas include artificial intelligence, machine learning application, internet of things, intrusion detection, natural language processing, and data mining.