People's Democratic Republic of Algeria
الجمهورية الجزائرية الديمقراطية الشعبية

Ministry of Higher Education and Scientific Research
وزارة التعليم العالي والبحث العلمي

National Polytechnic School of Oran -Maurice Audin-
Department of Computer systems Engineering

End of Studies Project Thesis For Obtaining State Engineering Degree

Intrusion detection based on Machine Learning Techniques

Presented By :
Mr. MAHDJOUR Oussama
Mr. ZIANE BOUZIANE Mohammed Amin

Supervided By :
Mme. KABLI Fatima (ENPO)
Mme. BOUMDJOUT Amal (ENPO)

Presented on 25/06/2022 in Front Of The Jury Compose Of :

Mme. BENDIMRRAD NAWEL : ENPO - President

Mr. BELBACHIR REDOUANE : ENPO - Examiner

College year: 2023/2022

DEDICATION YOUCEF

With the expression of my gratitude, I dedicate this work :

To My loving Parents, Whose Unwavering support, encouragement, and sacrifices have been my constant source of inspiration throughout my academic journey.

To My Dear Sisters.,

To All My Friends

I would also like to dedicate this to Assil , Amani, Yousra and Adam . I hope this work inspires and motivates you to pursue your dreams for the future

To all those who we would have forgotten to mention but who exist at the bottom of my heart and my mind.

- Youcef

DEDICATION OUSSAMA

With the expression of my gratitude, I dedicate this work to :

To My Loving Parents, Whose Unwavering Support, Encouragement, and Sacrifices have been my constant source of inspiration throughout my academic journey.

To My Dear Sisters, who have always been there for me, offering their love, guidance, and encouragement.

To My Brother, whose unwavering belief in my abilities has been a driving force behind my successes.

To My Uncles and Aunts, whose guidance and wisdom have shaped my perspective on life and provided valuable advice.

To all my Friends, both old and new, who have walked with me on this path, offering their support, friendship, and laughter. Your presence has made this journey all the more memorable and enjoyable.

I am eternally grateful for your love, support, and encouragement. You have played a key part in my successes, and I am honored to have you in my life.

- Oussama

ACKNOWLEDGEMENT

it seems opportune to start this report with thank God the Merciful, who gave us the strength, we send all our gratitude to our mentor ***Mme. KABLI Fatima*** *and* ***Mme. BOUMDJET Amal*** *for their patience, their availability, and especially their precious advice . also wish to acknowledge our mentor* ***Maria Bouchair****, who provided unwavering support and valuable guidance during our internship.*

We would like to take this opportunity to express our sincere appreciation to the faculty and administrative staff of ***the National Polytechnic School of Oran - Maurice Audin*** *as well as all the teachers who have contributed to our education throughout our academic journey, from near or far. Our thanks also to all the members of the jury for agreeing to examine our work with patience. Our warm thanks go to our dear parents who have always been there for us and who have given us a magnificent model of hard work and perseverance. We hope they will find our gratitude, appreciation, and love in this work. We would like to express our gratitude to our friends and colleagues who brought us their moral and intellectual support throughout our process Finally, our most sincere thanks go to all those who have contributed, from near and far, to the improvement of this memory.*

Abstract

Abstract :

The escalating threat of Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks poses a significant challenge to network security. In this work, we propose an innovative intrusion detection methodology that leverages machine learning and deep learning techniques to overcome the limitations of previous studies. Our approach involves analyzing diverse network traffic patterns and enhancing attack identification precision. We implement a deep learning model, specifically an Auto-Encoder, for accurate anomaly detection by processing vast volumes of normal data. Additionally, we develop a machine learning model to classify hazardous cyber-attacks, improving the robustness of our intrusion detection system. Our aim is to generalize the classification model to cover a broader range of DDoS/DoS attacks. Our work presents fascinating results that surpass previous proposed work which achieves our objective, By achieving these objectives, Our work contributes to enhancing network stability and cybersecurity.

Keywords : Machine Learning, Deep Learning, Intrusion Detection Systems, Cybersecurity, Anomaly Detection, Classification, Denial of Service, Network.

Resume :

La menace croissante des attaques de Denial of Service (DoS) et de Distributed Denial of Service (DDoS) représente un défi majeur pour la sécurité des réseaux. Dans ce travail, nous proposons une méthodologie innovante de détection d'intrusion qui exploite les techniques d'apprentissage automatique et d'apprentissage profond pour surmonter les limites des études précédentes. Notre approche consiste à analyser des modèles de trafic réseau diversifiés et à améliorer la précision de l'identification des attaques. Nous mettons en œuvre un modèle d'apprentissage profond, plus précisément un Auto-Encoder, pour une détection précise des anomalies en traitant de vastes volumes de données normales. De plus, nous développons un modèle d'apprentissage automatique pour classifier les cyber-attaques dangereuses, améliorant ainsi la robustesse de notre système de détection d'intrusion. Notre objectif est de généraliser le modèle de classification pour couvrir un éventail plus large d'attaques DDoS/DoS. Notre travail présente des résultats fascinants qui surpassent les travaux proposés précédents qui atteignent notre objectif. En atteignant ces objectifs, notre travail contribue à améliorer la stabilité du réseau et la cybersécurité.

Mots-clés : Cyber-sécurité, Déni de Service, Déni de Service Distribué, Systèmes de Détection d'Intrusion,Apprentissage Automatique, Apprentissage Profond, Auto-Encoder, Classification, Attaques Cybernétiques, Réseaux.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

GENERAL INTRODUCTION AND OBJECTIVE

Introduction

It is practically impossible to imagine living without the Internet in recent years. The internet is critical in linking billions of devices in numerous applications such as healthcare, industry, and business. etc. However, this widespread connectivity also brings forth new challenges and one of the prominent concerns is the emergence of Denial of Service (DoS) attacks. These attacks involve malicious individuals, commonly known as hackers, intentionally disrupting network services by overwhelming the target server with a massive influx of packets. The primary objective of these attacks is to cause prolonged service disruption rather than directly stealing sensitive data or monetary gains from the targeted system.

The escalating threat of Denial of Service (DoS) attacks highlights the critical importance of robust security measures, specifically Intrusion Detection Systems (IDS). IDS have become indispensable elements of computer and system security, offering defense against various attack types, including DoS attacks. Although current classification methods possess their own advantages and limitations, there remains an ongoing need to develop methodologies that overcome the deficiencies of existing techniques. In the realm of intrusion detection systems (IDS), there has been a notable surge in proposed techniques and the exploration of machine learning and deep learning applications. This surge is primarily driven by the rapid advancement of machine learning and the growing sophistication of network hacker attacks. Researchers in the broader field of cybersecurity, with a specific focus on IDS, have devoted substantial efforts to enhancing performance. Consequently, as machine learning continues to progress, researchers are actively investigating its potential applications in the field of cybersecurity.

Machine Learning and Deep Learning stand out as highly promising and rapidly evolving fields in this context. These techniques offer the potential to extract valuable insights from data, leading to transformative advancements across various industries. However, implementing machine learning techniques in the realm of cybersecurity presents unique challenges and opportunities. Within this landscape, intrusion detection plays a crucial role in safeguarding information security. Accurate detection is vital for enabling prompt responses and implementing appropriate measures to prevent intrusions.

Scholars have made considerable efforts in recent research to apply machine learning methodologies in intrusion detection and classification. These investigations aim to develop machine learning-based IDS by utilizing specific algorithms on legacy datasets. The evolving nature of machine learning contributes to the ongoing academic interest in this field. Machine learning's adaptability and learning capabilities are well-suited to address dynamic cybersecurity threats. As technology advances, the complexity and volume of

network traffic data increase, highlighting the relevance of machine learning approaches in intrusion detection systems. Enhancing and optimizing machine learning methodologies in cybersecurity, particularly for intrusion detection, remains an active area of research. However, challenges such as the need for large, relevant, and current datasets, as well as computational resources and expertise, should be acknowledged. Continued research and development in this domain are crucial considering these considerations.

The performance of an Intrusion Detection System (IDS) is critical in ensuring the security of network systems. However, the effectiveness of an IDS heavily relies on its ability to accurately identify and differentiate between normal network behavior and potential threats. In our work, we specifically concentrate on implementing Amachine and deep learning techniques to create an intelligent intrusion detection system that addresses the limitations of outdated detection methods. These limitations include high rates of false positive alarms, resulting in unnecessary alerts and resource wastage. Additionally, false negatives pose a risk as they can lead to system slowdowns and potential vulnerabilities within the company. By implementing advanced machine and deep learning algorithms, utilizing anomaly detection, and refining data pre-processing techniques, our intelligent IDS aims to overcome these challenges.

Objective

In this research, our primary goal is to establish an innovative intrusion detection methodology leveraging the robust capabilities of machine learning and deep learning. We aim to deploy these advanced techniques on contemporary datasets to create a model that can potentially overcome the existing challenges of earlier studies, such as elevated rates of false positives and false negatives in detection.

The central tenet of our approach is the analysis of diverse network traffic patterns, enhancing the precision of attack identification. Consequently, we can enhance the reliability and efficacy of intrusion detection systems. In more detail, the objectives of our research are as follows :

- Improve IDS Performance : Enhance the accuracy of an Intrusion Detection System (IDS) in detecting network anomalies while significantly reducing the number of false-positive alarms. By minimizing these errors, we aim to develop a more reliable and efficient intrusion detection system.
- Implement Deep Learning Architecture : Implement a model grounded on a deep learning architecture, specifically an Auto-Encoder, for intrusion detection. This form of learning is crucial due to its capability to process and analyze vast volumes of normal data, enabling more accurate anomaly detection.
- Develop a Machine Learning Model : Develop a machine learning model to further enhance the effectiveness of our solution. This includes adding the capability to accurately classify the most hazardous types of cyber-attacks, contributing to the robustness of our intrusion detection system.
- Expand the Classification Range : Generalize our classification model to include a broader spectrum of DDoS/DoS attacks. By incorporating the capability to classify a wider array of these threats, our system will be more up-to-date and prepared to respond to newer attack methods.
- Utilize a Comprehensive Dataset : Our research emphasizes the significance of a large and diverse dataset for training and evaluating our intrusion detection system. This comprehensive dataset captures real-world network traffic complexities, enabling our system to adapt to various network anomalies, including DDoS/DoS

attacks. The substantial volume of data enhances the system's robustness and reliability in handling evolving cybersecurity threats.

By successfully achieving these objectives, we hope to make significant contributions to the field of intrusion detection and cybersecurity, making networks more resilient to the evolving threat landscape.

Contents of Chapters

The rest of the document is organized into five (5) chapters followed by a general conclusion, each representing a part of the work of this project, whether it is theoretical or practical :

- **Chapter 1 Machine and Deep : Learning** This chapter provides an in-depth exploration of machine learning, its principles, and its applications.
- **Chapter 2 Security Techniques :** This chapter is dedicated to the basic concepts of security and intrusion detection systems.
- **Chapter 3 State-of-the-Art :** In this chapter, we present a comparative study of the main existing approaches mentioned in the literature.
- **Chapter 4 Conception :** In this chapter, we present the conception of our research, detailing the methodologies and strategies employed.
- **Chapter 5 Implementation :** We introduce the development and the deployment environment, then illustrates a summary of the results obtained

CHAPITRE 1

MACHINE AND DEEP LEARNING

1.1 Introduction

In today's fast-paced technological landscape, we are encountering a rapid growth of data-driven applications, primarily due to the increasing amount of data available in various domains and the rapid evolution of machine and deep learning that resulted in the transformation in complex data analysis and decision-making. This chapter covers the basic concepts of the machine and deep learning fields and explains the data mining process and how knowledge extraction from data. Finally presents the most common algorithms and the domains of applications. Overall, this chapter aims to provide an overview of the machine and deep learning and their applications and lay a foundation for the subsequent chapters of this thesis.

1.2 KDD Process

The term KDD (Knowledge Discovery in Databases), has been used since 1989. It is the result of the convergence of research in automatic learning, recognition of forms, databases, statistics, artificial intelligence, and data visualization. There are so many definitions of the KDD process, Han and Kamer (Han and Kamer,2006) explain the KDD process as the process of analysis of the database, often having a large size, to discover unsuspected relationships and summarize the data in an understandable and useful way. According to Fayyad in 1996 KDD is the process of extracting knowledge based on identifying unknown patterns that can be exploited in the database, this process aims to transform a large data, in multiform, stored in different formats and gathered from different places into usable knowledge that can be used in decision making. KDD process model consists of a set of steps to follow ; He takes data assets and provides new knowledge as final outings. The stages of the KDD process, are illustrated in the following figure :

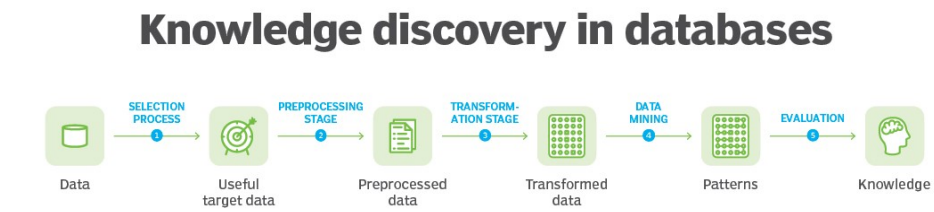


FIGURE 1.1 – KDD-Process
[1]

1.2.1 Data Selection and Extraction

This step consists of two steps :

1.2.1.1 Understanding the domain of application

This step includes understanding the domain of application, the discovery of relevant previous knowledge, and the definition of the objective of the KDD process.

1.2.1.2 The creation of the target data set

This step includes selecting a data set or concentration on a subset of variables or data samples. In addition, it includes data integration if the data is of different heterogeneous resources. The final output of this step is the set of data on which the discovery must be made.

1.2.2 Data Preprocessing and Cleaning

This step includes deleting noise or aberrant values, collecting the information necessary to model or process noise, use of previous knowledge to delete inconsistencies and duplicates from data, choosing or use of strategies to manage missing data fields, etc. Most of the data collected is incomplete and inconsistent, which leads to confusing the rest of the process and leads to unreliable and non-valid results. For this, this step must be established to improve the quality of the data and arrive at a good result as a final outing. Here are some of the basic steps in data pre-processing :

1.2.2.1 Handling missing values

This can cause a really big problem in data manipulation and model training because the NaN values occupy a space in memory that have no value which can't be processed by the computer, this problem also accrues while handling infinite values. To solve this problem, we have two approaches either by dropping the rows that contain missing values if the dataset is large enough or we can generate new data to fill the missing value using the statistical approach

1.2.2.2 Handling categorical data

There are two types of data : qualitative data which can represent categories and have groups or classes and quantitative data which represents numerical values Most machine learning algorithms can't process categorical data so in this step we need to encode categorical data into numerical.

1.2.2.3 Handling outliers

Outliers are the values that fall off the extreme of the value distribution that represents rare cases and that can cause errors in the algorithm to overfit on rare cases therefore to ensure the best results we need to handle those outliers

1.2.2.4 Normalization of the dataset

The value of the dataset can be large numbers and this can take a lot of processing power and time, to solve this problem we need to normalize the data this can be done by statistical tools

1.2.2.5 Handle Dataset imbalances

Often the dataset has an imbalance in the number of instances in target classes that can cause to model to have a bias toward the majority classes while neglecting minority classes to solve this problem we have two methods :

Under sampling : Using different algorithms, we can reduce the size of the majority classes to have a balanced ratio with the minority classes, this approach is optimal with large datasets but can reduce the results of the models when applied to small datasets.

Data Augmentation : Using different algorithms, we can generate new data in the minority classes to augment the ratio with the majority classes

1.2.2.6 Correlation analyses

The machine learning algorithm can struggle with large sets of features that can drain an unnecessary number of resources so it is optimal to analyze the relationship between variables and combine strongly related variables

1.2.3 Data Transformation and Reduction

This step includes the search for useful features to represent the data, depending on the objective of the process. More specifically, processing techniques are used in this step to make data more appropriate for exploitation such as aggregation and normalization ... and reduction techniques are applied to produce a reduced representation of data such as aggregation and dimension reduction.

1.2.4 Data Mining and Pattern Extraction

It is the application of a group of different methods on ready-to-use data. Due to the great diversity of the data used, the result is a large number of data excavation methods. The latter come from various fields such as statistics, data analysis, automatic learning, etc. In addition, some of these methods can be combined to reduce the disadvantages of one or the other. The choice of data excavation methods depends on the one hand, the needs expressed by the user and on the other hand, the used data.

1.2.5 Pattern Evaluation and Interpretation

This phase consists of the evaluation of the model, which measures the performance of the model on many different levels and then presents the results to the user using different visualization techniques.

1.3 Machine learning

Machine learning is a sub-field of artificial intelligence that focuses on extracting knowledge from data. It is a field based on the fusion of statistics and computer science to create systems and models that learn and improve based on the data the process

1.3.1 Definition

The term “Machine learning” was defined in 1959 by IBM employee “Arthur Samuel” [2] as the following : “Machine learning is the discipline giving computers the ability to learn without being explicitly programmed”. Machine learning algorithms create models by training the algorithms on large amounts of data to identify patterns and relationships and then use them to make predictions or decisions on new data. Those models enable computers to automatically improve their performance on a task by learning from experience.

1.3.2 Types of Machine Learning

Machine learning algorithms can be divided into three main types : supervised learning, unsupervised learning, and reinforcement learning

1.3.2.1 Supervised Learning :

Supervised learning is a type of machine learning where the algorithm trains on labeled data where the desired output or target is known for each input data point, and then it makes predictions on new, unseen data. the algorithms of supervised learning are the most widely used, it exists two main categories in supervised learning : regression and classification.[3]

Classification : In classification, the algorithm predicts a categorical label or class, such as "spam" or "not spam". The output variable is qualitative.

Regression : In regression, the algorithm predicts a continuous value, such as a price or a temperature. The output variable is quantitative.

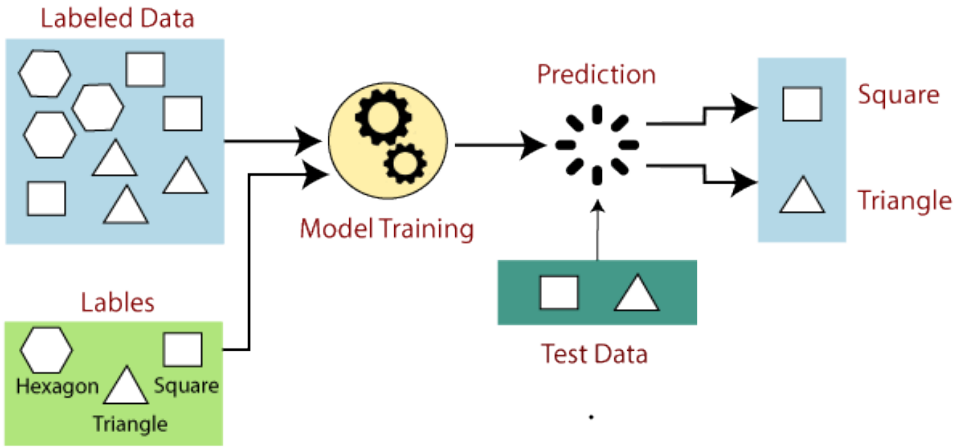


Figure 1.2 – illustration of supervised machine learning [4]

A. Linear Regression Linear regression is a regression-type of supervised learning algorithm capable of establishing a linear relationship between a dependent variable, and

one or more independent variables. The main goal is to fit a better line that comes as close as possible to a set of points, represented by a linear equation.[5]

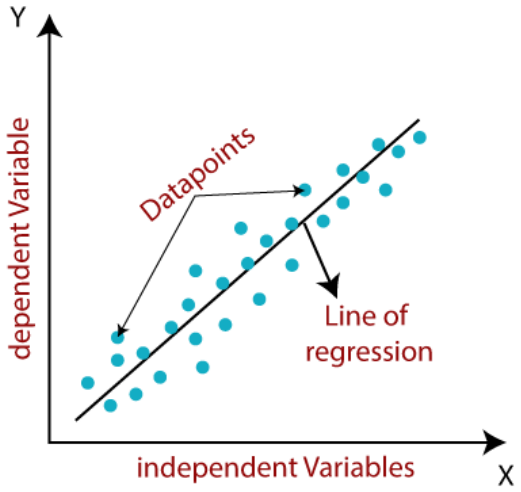


FIGURE 1.3 – illustration of Linear Regression Architecture
[6]

B. Polynomial Regression Polynomial Regression is a regression algorithm that models the relationship between a dependent(y) and independent variable(x) as an nth-degree polynomial, this considers an upgrade to the linear regression algorithm so it can have better results on complex data.

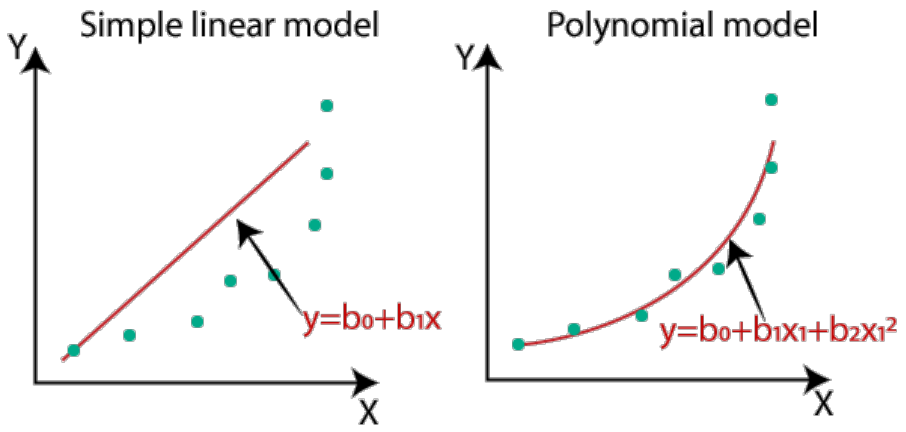


FIGURE 1.4 – illustration of Poly-nominal Regression Architecture
[7]

C. Logistic Regression : Logistic regression is one of the most known supervised learning methods ; this type of algorithm uses the logistic function on the independent variables to predict the dependent categorical variable. logistic regression is a form of regression that is used for classification where the output is the probability of the class.

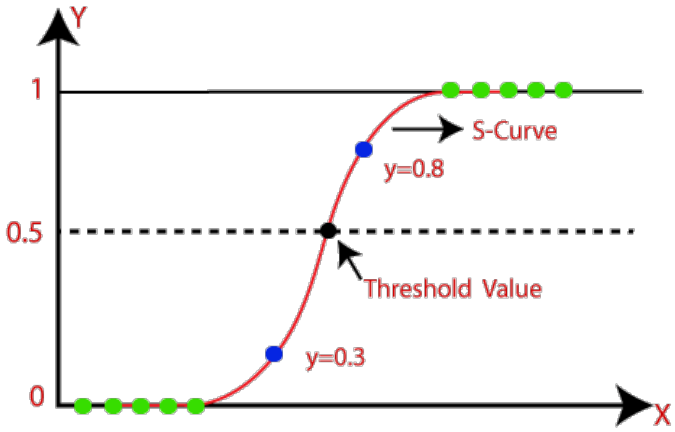


FIGURE 1.5 – illustration of logistic Regression Architecture
[6]

D. K-Nearest Neighbours (KNN) The KNN algorithm is one of the simplest supervised learning algorithms. It aims to classify each new data with the use of a distance metric such as Euclidean distance.

The simple founding idea is to start from a labeled database, we can estimate the class of new data by looking at what is the majority class of the k nearest neighboring data (hence the name of the 'algorithm). The only parameter to fix is k, the number of neighbors to consider.

This algorithm can be used for Regression as well as for Classification but mostly it is used for Classification problems.

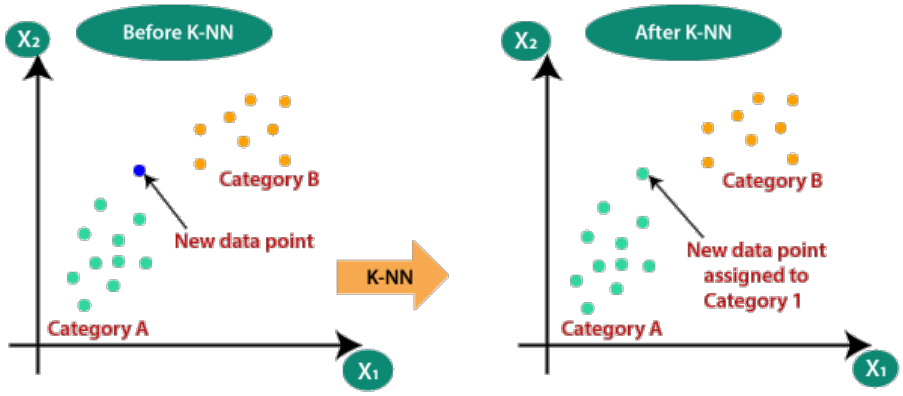


FIGURE 1.6 – illustration of KNN Architecture
[8]

Algorithm 1: K-Nearest Neighbor Algorithm

Input: i : number of instances in the training dataset ($i > 0$), C : Class of i sample, $k > 0$, D : Training Dataset, $Dist$: Function of calculation distance, X : example to classify

Output: Y : the label of the instance X

$KNN(X) \leftarrow \{\}$
for $(i, C) \in D$ **do**
    `// Calculate the distance`
    $KNN(X) \leftarrow Dist(i, X)$
end
$Z \leftarrow \{\}$
for $i \in KNN(X)$ **do**
    `// Calculate the number of occurrences of each class` C `in` Z
    $Z \leftarrow$ Occurrences of class C in i
end
$y \leftarrow \max(Z)$
Result: $Y \leftarrow y$

E. Naive Bayes A probabilistic model that calculates the probability of each class given the input features using Bayes' theorem, it is one of the simple and most effective Classification algorithms which helps in building fast machine learning models that can make quick predictions. This algorithm is a probabilistic classifier, which means it predicts on the basis of the probability of an object, as the name suggests this algorithm uses the Bayes rule which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability according to the following formula

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1.1}$$

P(A|B) represents the Probability of hypothesis A on the observed event B.
P(B|A) represents the Probability of the evidence given that the probability of a hypothesis is true.
P(A) represents the Probability of the hypothesis before observing the evidence.
P(B) represents the Probability of Evidence.

F. Support Vector Machine (SVM) Support vector machine is a popular algorithm of machine learning that was developed in the 1990s, it is capable of performing linear and non-linear classification and regression. the principle of this algorithm is to seek a hyperplane or border which best separates two classes of data by guaranteeing that the margin between the two classes is maximum. Multiple planes can separate the two classes, but a single plane can maximize the headroom or distance between classes.

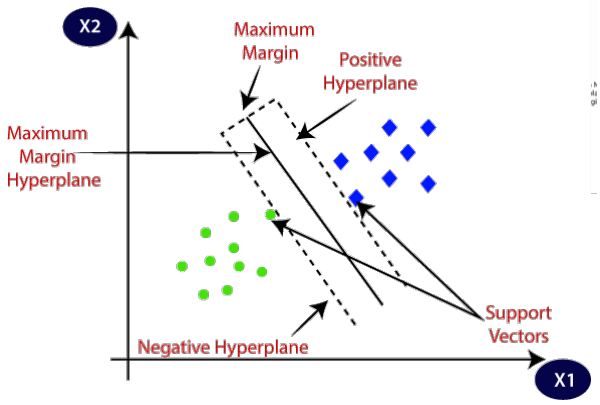


FIGURE 1.7 – illustration of SVM Architecture
[9]

G. Decision Tree Decision trees are supervised learning models, which can be used for classification as well as regression, they have several algorithms known as ID3 (Quinlan 1986) and C4.5 (Quinlan 1993), which generate a tree structure where each node internally refers to a test on an attribute. Each branch represents the test result and each leaf node contains a decision which is a unique value. Inputs and outputs can be discrete or continuous.

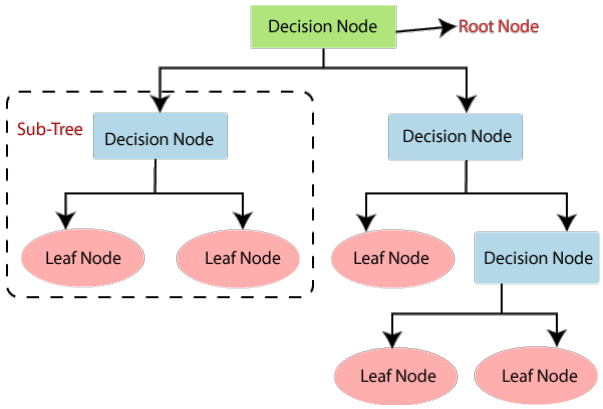


FIGURE 1.8 – illustration of Decision tree Architecture
[10]

H. Ensemble learning Ensemble learning is a group of techniques of machine learning called meta-algorithms that consists of combining a group of unique models to optimize the performance and have better generalization. ensemble learning methods are divided into two main categories based on the nature of the combination : bagging (parallel training) and Boosting (sequential training).

Bagging The Bagging ensemble technique is the acronym for "bootstrap aggregating" and is one of the earliest ensemble methods proposed. It consists of creating subsamples from a dataset called "bootstrap sampling". Then train several Machine Learning models on those subsets to create independent models called weak learners. During test time, the predictions from all weak learners are accounted for.

Boosting The boosting ensemble mechanism works in a way slightly different from the bagging mechanism. where, instead of parallel processing of data, sequential processing of the dataset occurs. The first weak learner is fed with the entire dataset, and the predictions are analyzed. The instances where the previous weak learner fails to produce correct predictions will be fed to the next weak learner. This is done so that the next weak learner can specifically focus on the problematic areas and correct the errors made by the previous weak learner. Similarly, further steps of the same idea are employed, and then the ensemble of all these previous classifiers is computed to make the final prediction on the test data.

I. Random Forest Random Forest is a popular ensemble-based supervised machine learning algorithm used for both Classification and Regression problems in machine learning. Random Forest is a classifier that contains several decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.[11]

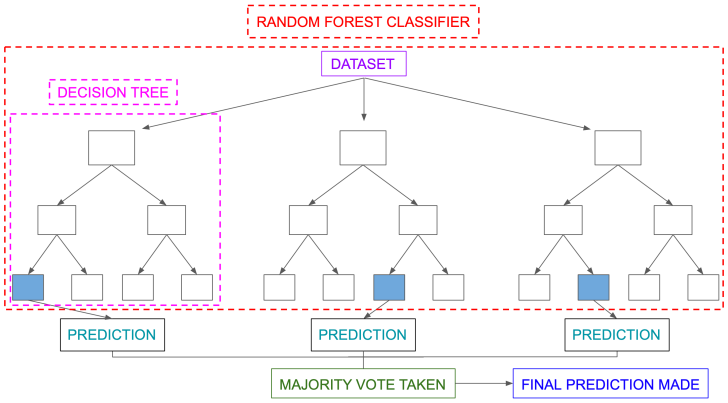


FIGURE 1.9 – illustration of Random Forest Architecture [12]

J. Adaptive boosting (Adaboost) AdaBoost (Adaptive boosting), proposed by "Freund and Schapire" in 1996, was the first boosting algorithm to combine various weak classifiers into a single strong classifier in the history of machine learning. Each time a basic estimator is trained, the previously misclassified instances are weighted with a higher weight, with the aim that during the next iterations, the new models correct the errors of the previous models, which should improve the overall performance.[13]

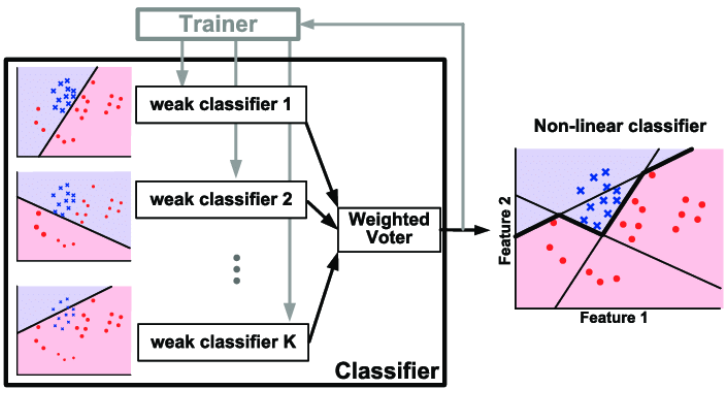


FIGURE 1.10 – illustration of ADA Boost Architecture [14]

K. gradient boosting machine (GBM) Gradient boosting machine (GBM) is one of the most popular sequential ensemble learning algorithms, GBM can be used for creating predictive models for classification or regression problems. The GBM algorithm is similar to the Adaboost algorithm with one key difference in how the model is being updated after incorrect prediction, It works on the principle that many weak learners can train a more accurate model, where each model fits the residual from the previous step to improve the model using the Gradient Descent algorithm, the residual is the gap between prediction and reality and can be calculated using different loss functions for different problems, for example, we can use mean absolute error for regression problems or we can use log loss function for classification problems

L. Extreme Gradient Boost (XGBoost) XGBoost is an evolutionary ensemble-based supervised learning system that is used to reinforce decision tree-based models such as random forest, it was proposed by Chen and Guestrin in 2016. Their principal is to combine the results of weak simple models to produce butter prediction, contrary to the random forest algorithm this model uses the boosting method for their ensemble learning. their principle of sequential auto-optimization includes the use of a large number of hyper-parameters to minimize the loss function and it consists of three elements :

- optimized loss function
- weak learner model
- An additive model to combine our weak learners to minimize the loss function

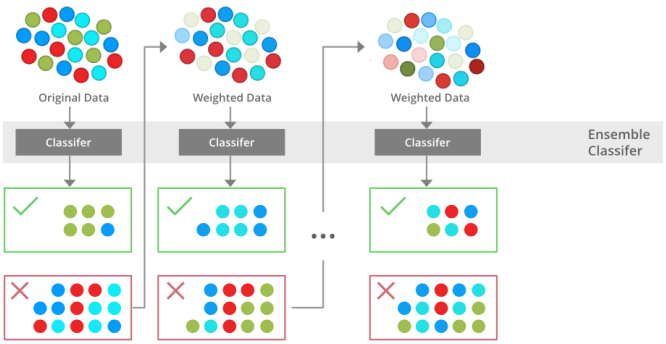


FIGURE 1.11 – illustration of XGBoost Architecture [15]

M. LigtGBM Algorithm Introduced by Microsoft in January 2017, the Light Gradient Boosting Machine (LightGBM) represents an enhanced version of the Gradient Boosting Machine (GBM). This model improves on the conventional decision tree algorithm by using a gradient-based one-side sampling (GOSS) approach and an exclusive feature bundling (EFB) technique. LightGBM was designed to be more efficient, consuming less memory and delivering improved performance, especially for large-scale data processing.[15]

Contrary to the traditional level-wise growth of decision trees, LightGBM implements a leaf-wise growth strategy. After the initial split, subsequent divisions are carried out only on the leaf node that incurs the highest loss. This unique approach aids in reducing loss more rapidly, contributing to the algorithm's efficiency.

N. CatBoost Algorithm Developed by Yandex in 2017, the CatBoost algorithm—derived from the term 'Categorical Boosting'—is particularly adept at handling categorical va-

riables in data. This model transforms categorical variables into numerical ones by applying various statistical techniques on feature combinations.

Many machine learning algorithms struggle to directly process data represented in strings or categories. Hence, the conversion of categorical variables into numerical values becomes an essential preprocessing step. The CatBoost algorithm excels in this conversion, thus eliminating the need for explicit preprocessing. [15]

The distinguishing feature of CatBoost is its capacity to handle categorical data directly, enabling it to maintain a high level of accuracy even when dealing with data that has numerous categorical features.

1.3.2.2 Unsupervised Learning

unsupervised learning consists of building a model from unlabeled input data. To do this, the system will cross-reference the information submitted to it, to be able to group in the same class the elements with certain similarities.

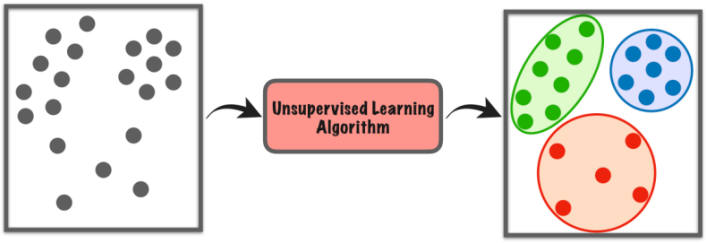


FIGURE 1.12 – illustration of unsupervised learning
[16]

there are three main types of unsupervised learning :

A. Clustering Clustering is the method of statistical analyses used to group the data points from unlabelled datasets into homogeneous Clusters with each cluster consisting of similar data points, with the object with similarities in the same group, most of the clustering algorithms work by finding similar patterns in the data points like shape or colors. the clustering algorithms can be divided into two main groups :
Hard Clustering : were each data point can only exist in a single cluster, and **Soft Clustering :** where each data point can exist in multiple clusters

B. Dimensionality reduction Dimensionality reduction is a technique used to reduce the complexity of data by reducing the number of features or variables while preserving the important information contained in the data. Its main goal is to simplify the data without sacrificing crucial information. This is usually achieved by grouping similar features together or by transforming the original features into a new set of features that represent the original data more efficiently.

Dimensionality reduction is widely used in various fields that handle high-dimensional data, such as speech recognition, signal processing, image processing, and machine learning. The reduction in dimensionality makes the data more manageable, easier to analyze, and often leads to improved performance of machine learning algorithms. Additionally, dimensionality reduction can help to eliminate noise, redundancy, and irrelevant information, which can improve the accuracy and efficiency of data analysis.

C. Association rules The algorithms used for association rule mining belong to the family of unsupervised learning methods. These methods enable the identification of patterns within a group of transaction data, which can then be used to generate rules representing possible associations between different items. The resulting association rules can help to identify correlations and co-occurrences between datasets and are particularly useful for explaining patterns in seemingly independent information repositories, such as relational and transactional databases. Association rule mining is a powerful tool for discovering hidden patterns and relationships within large datasets and can provide valuable insights for decision-making in a variety of fields.

1.3.2.3 Reinforcement learning

Reinforcement learning (RL) is a distinct type of machine learning that differs from other approaches in that it involves a multi-agent system. In RL, the learning process is based on successive experiences, where an agent interacts with an environment by taking actions in order to find an optimal solution based on a reward system, with the ultimate goal of finding the best strategy.

RL is particularly useful in complex, dynamic environments where it may be difficult to model all possible scenarios. By learning from experience, RL algorithms can adapt to changing environments and find optimal solutions. Examples of RL applications include robotics, game-playing, and autonomous vehicle control.

1.4 Deep learning

Deep learning is the newfound sub-field of machine learning, it is the technological key component of today's revolutionary AI application that makes it possible to perform tasks that were still unthinkable a few years back such as chat-bots that can generate human-like written text like ChatGPT or a generative AI that can generate an image starting with a descriptive prompt. The success of deep learning is based both on the technical evolution of computing capability, and the huge advancement in information modeling and representation in statistical learning methods.

1.4.1 Definition of deep learning

It is the newest field of research in machine learning, and it was created to surpass the limitations of machine learning with data pre-processing and feature extraction. deep learning algorithm inspired by the power of the human brain and how it can process large data in a short time, this resulted in deep learning algorithms having better performance on large data than classical machine learning algorithms Deep learning algorithms are based on one key competent called artificial neural networks which are curated to handle large data without any problems Deep learning models surpass machine learning models with their ability to extract important features from raw data without human intervention, this is due to the nature of the neural network and its deep layers that compose a group of linear or nonlinear equations those feature than the process through deep hidden layers to then finally generate the final output of the model [17][18][19][20]

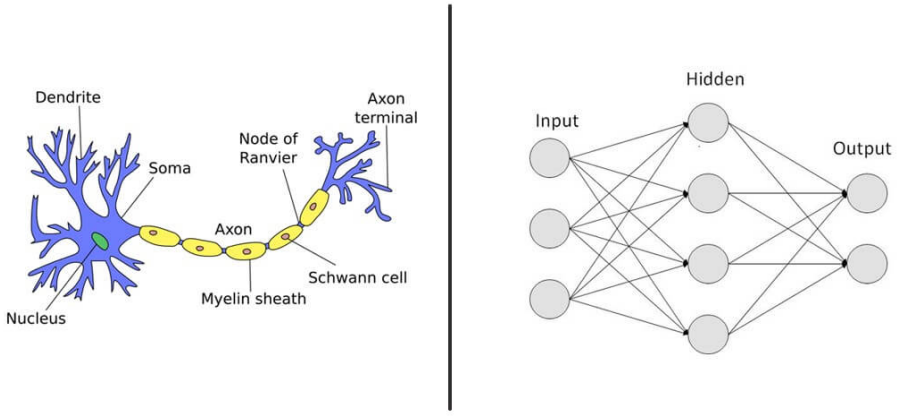


FIGURE 1.13 – illustration of Neural Network and Biological Neural [21]

1.4.2 Why Deep Learning ?

with the rise of data-driven applications and the technical advancement in electronics, more data have been stored and processed than enabled machine learning algorithms to solve imaginable problems but due to the nature of machine learning algorithms they need human intervention to help process the data and eliminate unnecessary features to have good performance, this intervention limited the advancement of AI application to a certain domain of application where the data can be pre-processed by humans efficiency. The deep learning algorithm, on the other hand, came to solve this problem and help surpass the machine learning limitation by mimicking the human brain structure with artificial neural networks this structure can extract important features from given information without any human intervention needed, and this opened up a lot more domain of application for AI where it was impossible to achieve with traditional machine learning algorithms like computer vision and AI chatbots. Due to the nature of deep learning algorithms, it needs a large set of data to extract feature and outperform machine learning, studies show that machine learning algorithms outperform deep learning algorithm in small data but when the data is increased the machine learning algorithm stops improving while deep learning algorithms outperform them, with amount data large enough machine learning algorithms can outperform humans in problems like image processing and pattern recognition [22]

1.4.3 Neural networks

Deep learning algorithms consist of artificial neural networks called ANN, which are models that treat information inspired by the structure of biological neural systems. It is similar to the way the human brain processes information. all neural networks consist of an ensemble of interconnected nodes called layers.

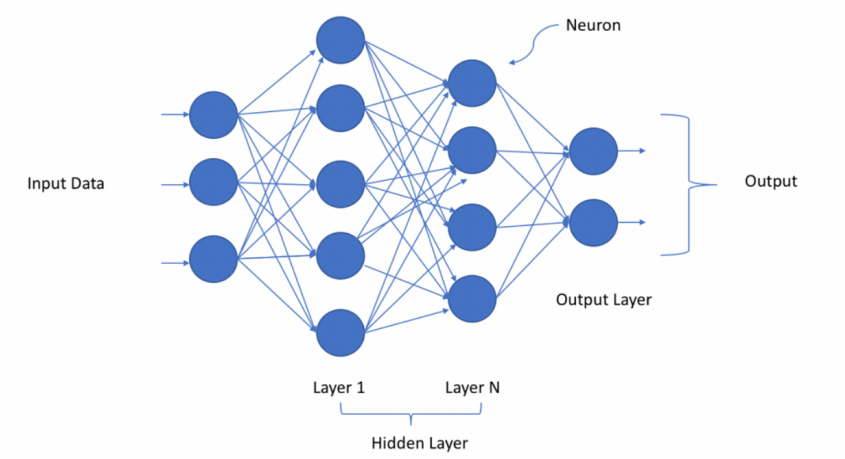


FIGURE 1.14 – illustration of Artificial Neural Network
[23]

1.4.3.1 Activation functions

They are key components of the neural network, they are mathematical functions that are applied in the calculation process of the value of each node in the layers, this value is then inputted into the next or previous layer as input depending on the complexity of the network, this function helps to optimize the learning process by assist in the weight updating process of each node. activation functions are usually nonlinear functions necessary for solving complex problems within the neural network. Therefore a lot of research has been done to create optimal activation functions here are some examples of widely used activation functions out there :[24] [25]

A. Sigmoid This function is one of the most commonly used activation functions, also known as the logistic function, or logistic sigmoid, this function's output is ranged between 0 and 1 where this output represents the stochastic probability of node activation. Due to the nature of this function, it is primarily used for binary classification problems.

B. Rectified Linear Function (ReLu) This is one of the fastest activation functions used in building deep learning models. This function is one of the most popular functions to be used in hidden layers because of its optimized performance and the speed of calculating the desired output, this function is a combination of two functions.

C. Softmax softmax function is a type of regression function that is a generalization of logistic regression primarily used for multi-classification problems, unlike other activation functions softmax function the output of node in this layer dependent on the output of all the other nodes exist in this function which assures that the sum of all the nodes outputs equals to 1.

D. Hyperbolic Tangent Functions (tanH) tanH function is a superior function compared to other widely used activation functions like the sigmoid function, However, it takes less account of the relationships and it is slower to converge.

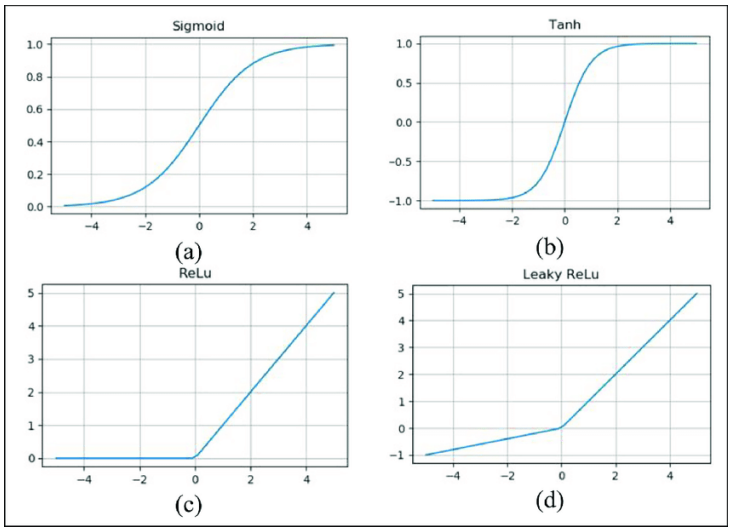


FIGURE 1.15 – illustration of Activation Functions
[26]

1.4.3.2 Back-propagation

Is the principle of updating the weight of each node to minimize the loss ; this process is the key component in the power of deep learning algorithms. This process is dependent largely on the choice of the loss function that is optimal for the problem, this process is controlled with a parameter called learning rate which measures the intensity of the changes in the node's weights.

1.4.3.3 Types of neural networks

The structure of neural networks depends on the problems we want to solve. Still, we can define three main classes of neural networks :

A. Single-layer perceptron (feed-forward) This form is the simplest example of a neural network, in this form, we have only two layers : an input layer and an output layer. where the propagation of the information is linear from the input where the information enters the network to pass by the activation function then it outputs via the output layer as a system response. This structure is simple as there is no returning connection or cross-connection in the output layer.[27]

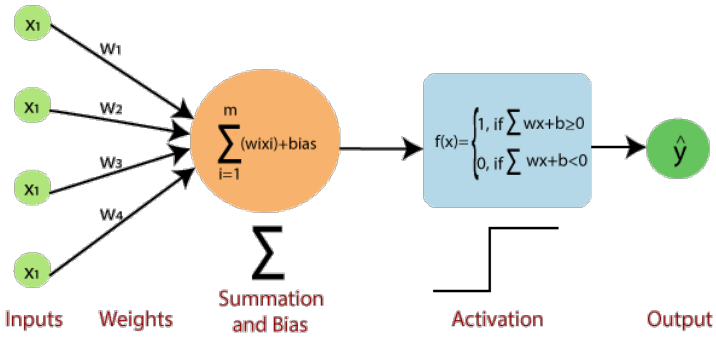


FIGURE 1.16 – illustration of Single-Layer perceptron
[28]

B. Multi-layer perceptrons (feed-forward) Multi-layer perceptrons, also called MLP (Multi-layer Perceptron), are more general neural networks than single-layer perceptrons. This type of network consists of one or more hidden layers, whose computational nodes are called hidden neurons or hidden units. The function of hidden neurons is to interact in a useful way between the external input and the network output and to extract higher-order statistics. The source nodes of the input layer of the network provide the input signal to the neurons of the second layer (1st hidden layer). The output signals from the second layer are used as inputs to the third layer and so on. The set of output signals from the neurons in the output layer of the network constitutes the global response of the network to the activation model provided by the source nodes of the first input layer.[29]

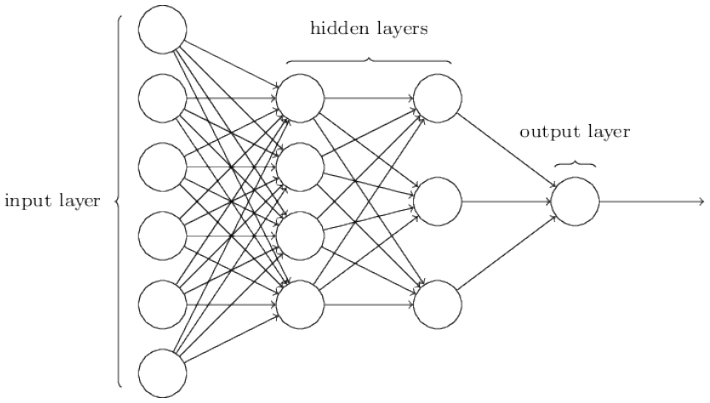


FIGURE 1.17 – illustration of Multi-Layer perceptrons
[30]

C. Recurrent deep neural network (feed-backward) A recurrent network differs from the previous ones in that it is cyclic : comprising one or more hidden layers with at least one feedback loop. The presence of cycles has a profound impact on the learning capacities of the network and on its performance, in particular making the system dynamic, which leads to a nonlinear dynamic behavior.[29][31]

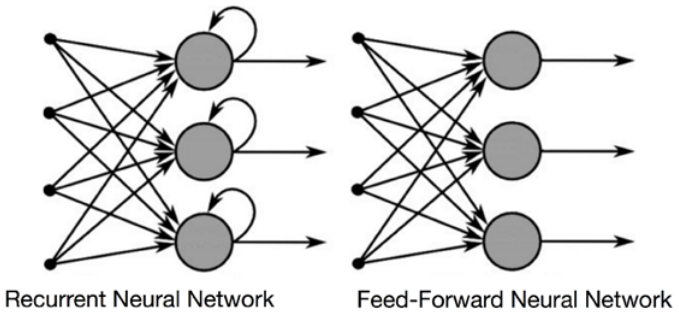


FIGURE 1.18 – illustration of Recurrent Neural Network
[32]

1.4.4 Architectures

Deep learning is based on multi-layered artificial neural networks. This technology has allowed scientists to solve complex problems and advance the field of data science and AI to a whole new level. These multi-layer neural networks can comprise millions of neurons, distributed in several tens of layers. They are used in deep learning to design supervised and unsupervised learning mechanisms. the modification in the different parameters of

the neural network such as the type of each layer and the way those layers are stacked up against each other results in different network architectures that can be used to solve various problems. here are some of the most widely used network architectures in the field of deep learning

1.4.4.1 Convolutional Neural Networks (CNN)

Convolutional neural networks are a type of specialized neural network for data processing having a grid-like topology. CNNs have had considerable success in practical applications. We can then define a CNN as a neural network that has one or more convolution layers in the network architecture. A nonlinear activation function is then applied to the output of these convolutions and the convolution process. The name "convolutional neural network" indicates that the network employs a mathematical operation called convolution. Convolution is a special linear operation. Convolutional networks are simply neural networks that use convolution instead of matrix multiplication in at least one of their layers. They have wide applications in image and video recognition, recommendation systems and natural language processing, etc.

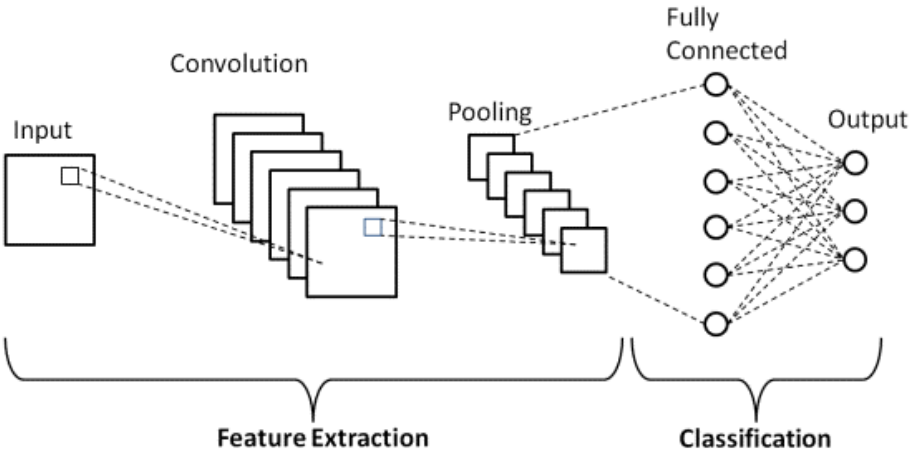


FIGURE 1.19 – illustration of Convolutional Neural Network Architecture [33]

1.4.4.2 Recurrent Neural Networks (RNN)

The recurrent neural network is based on the principle of saving the output of a layer and feeding it back to the input to help predict the outcome of the layer. The first layer is trained similarly to the feed-forward neural network with the product of the sum of weights and features. The recurrent neural network process starts once this is calculated, each neuron will remember some information that it had at the previous time step. This causes each neuron to act as a memory cell during calculations. In this process, we have to let the neural network work on forward propagation and remember the information it needs for later use. RNNs are called recurrent because they perform the same task for each element of a sequence, with the output dependent on previous computations.

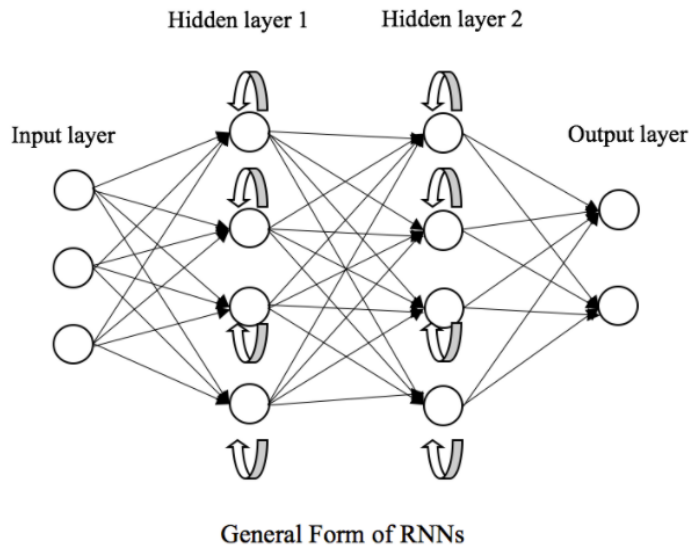


FIGURE 1.20 – illustration of Recurrent Neural Networks architecture [34]

1.4.4.3 Generative Adversarial Networks (GAN)

GANs for Generative Adversarial Networks. GANs are an example of a network that uses unsupervised learning to train two models in parallel. The first model is the generator. It generates a sample and tricks the discriminator when it generates data, on the other hand, the second network (adversary, discriminator), tries to detect if a sample is real or if it is the result of the generator. Its purpose is to learn to differentiate between these two types of data.

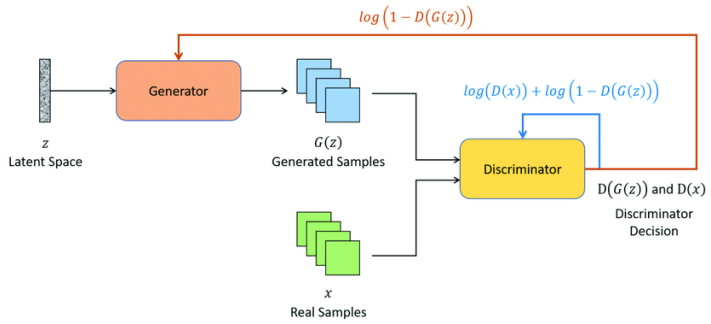


FIGURE 1.21 – illustration of GAN Architecture [35]

1.4.4.4 Automatic encoders (Auto Encoder) :

Auto-encoders (AEs) are a type of artificial neural network that can learn to compress data and then reconstruct it with minimal loss of information. The AE structure consists of two parts : the encoder and the decoder.

The encoder compresses the input data into a lower-dimensional representation, while the decoder reconstructs the original input from the compressed representation. The encoder and decoder are trained together to minimize the difference between the input and the output.

AEs have various use cases, including data compression, image denoising, image colorization, anomaly detection, and dimensionality reduction. For example, in data compression, the encoder compresses the data into a smaller representation, reducing the

storage requirements. In anomaly detection, the AE is trained on normal data, and when an anomaly is detected, the reconstruction error is higher, indicating that the input is not normal. In dimensionality reduction, the AE learns a low-dimensional representation of high-dimensional data, which can be useful for visualization or for reducing the complexity of a problem.

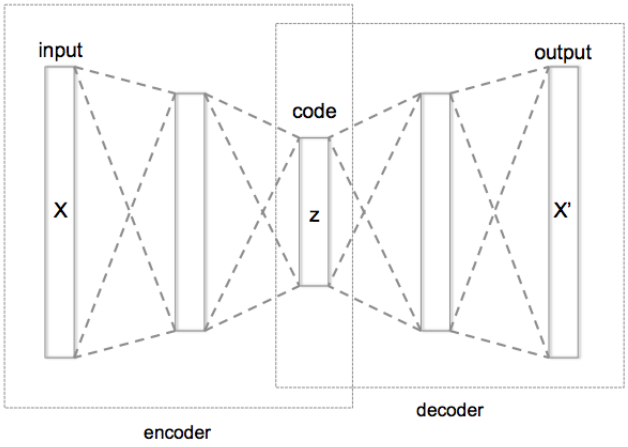


FIGURE 1.22 – illustration of AutoEncoder architecture
[36]

1.4.4.5 Transfer Learning

Transfer Learning refers to the set of methods that enable the transfer of knowledge gained from solving one problem to another. This has become particularly relevant in the era of Deep Learning, as models in this domain often require significant computational resources and time. By using pre-trained models as a starting point, Transfer Learning allows for the rapid development of high-performing models and the efficient resolution of complex problems in areas such as Computer Vision or Natural Language Processing (NLP).

Transfer Learning has become a powerful technique in machine learning, allowing the efficient use of resources and improving performance. It has been applied in various fields such as image classification, object detection, sentiment analysis, and speech recognition.

Transfer Learning draws heavily on the way humans learn, with the idea of reusing knowledge gained in other settings (source) to solve a particular problem (target). There are several approaches to Transfer Learning, depending on what is being transferred, when, and how the transfer is done. Generally, there are three types of Transfer Learning :

A. Inductive Transfer Learning In this configuration, the source and target domains are the same (same data), but the source and target tasks are different but closely related. The idea is to use existing models to advantageously reduce the scope of possible models (model bias), as illustrated in the figure below.

B. Unsupervised Transfer Learning Similar to inductive Transfer Learning, the source and target domains are similar, but the tasks are different. However, the data in both domains are not labeled. Obtaining large amounts of unlabeled data from databases and sources on the web, for example, is often easier than obtaining labeled data. Hence, the idea of using unsupervised learning in combination with Transfer Learning is gaining great interest.

C. Transductive Transfer Learning In this configuration, the source and target tasks are similar, but the corresponding domains differ either in terms of data or marginal probability distributions.

1.5 Machine and Deep learning applications

Artificial intelligence have various applications and that is thanks to the availability of data and the reliance of modern society on technology, this cause a lot of data scientist to work hard to research and apply ai to every aspect of the digital world and every economic sector. This made the application of AI dependent on the field expertise and the types of data handled in those fields, many data scientists like to differentiate between AI applications and the way those data translate in the field of application based on the type of data here are some of the major machine and deep learning applications :

1.5.1 Computer Vision

Image processing has a huge impact on a lot of critical economic sectors such as the medical, industrial, and security sectors this made the domain of computer vision a major field of application for Artificial intelligence, due to the nature of images and the advancement in processing power, a lot of research has been done to create optimal algorithms for image processing such as conventional neural networks that have revolutionized this field and enabled a wide range of application to existing such as facial recognition and object detection.

1.5.2 Natural language processing

Human communication is a huge challenge in the face of the creation of a global society and international affairs, languages are a key component of human communication but due to the nature of language and its difficulty the application of AI in this field is a must to optimize the human interaction and facilitate the globalization of the world. Natural language processing or NLP is a field of AI that consist of teaching computers how to interpret human language in text Due to its nature, researchers had faced a lot of problems applying machine and deep learning techniques to it, but thanks to openAI researcher teams that lay the foundation of natural language processing with their paper ‘attention is all you need where they introduce a new approach into text embedding called the ‘Attention mechanism’ and created a new architecture for deep learning called transformers, which was a revolution in the domain of AI and resulted in a lot of advancement in the field, such as the creation of the BERT model that is used for semantic analyses and GPT model that have taken the world by a storm with its capability to generate texts natural language processing has a lot of applications such as translation that enables access to all the information across all languages, semantic analyses like comment and review analyses that enable companies to understand and optimize their interactions with customers, and lastly chatbots like OpenAI’s chatGPT and Google’s Bard that can fully automate human interaction.

1.5.3 Speech and audio processing

The gap between humans and machines is based on the availability and simplicity of the interactions between them, and the advancement made in NLP that made it so that the machine can understand human languages can make those interactions easier but is still dependent on text. speech and audio processing in machine learning include a wide range of technologies that handle data in the form of audio such as automatic speech

recognition some of those technologies are often used to differentiate between sound and speaker others are used in the segmentation of clips into classes.

1.5.4 Time series analyses

Time series analysis consists of using Machine and deep learning methods to analyze specific sequences of data points collected throughout time, those data points are usually time-sensitive, which makes time series data represent the evolution of the variable through time best example of time series problems is the stock market where the change in time can directly affect the output.

1.5.5 Anomaly detection

consists of applying machine and deep learning algorithms to examine specific data points and detecting rare occurrences that seem suspicious because they are different from the design default patterns of behavior changes. the techniques used in anomaly detection problems are mainly focused on learning the default patterns and later any changes in the patterns as an anomaly

1.6 Machine and Deep Learning in Cyber-Security

The utilization of Machine Learning (ML) and Deep Learning (DL) methodologies has substantially expanded the field of cybersecurity, primarily due to their ability to discern patterns, extrapolate from historical data, and predict potential future threats. This section elucidates the diverse range of applications for ML and DL within the cybersecurity sector, accompanied by succinct examples to provide context.

1.6.1 Machine Learning Applications in Cybersecurity

Machine Learning, with its diverse analytical capabilities, has revolutionized the cybersecurity landscape, proving instrumental in several key applications :

1.6.1.1 Anomaly Detection

ML models can be trained to understand and identify typical system or network behavior. This allows them to recognize any deviations from this norm as potential threats. For instance, clustering algorithms such as K-means or DBSCAN can be employed to discern unusual patterns in network traffic indicative of a potential intrusion.

1.6.1.2 Malware Detection

Signature-based methodologies are often insufficient against the ever-evolving landscape of malware. Machine learning techniques such as decision trees and random forests can categorize files based on their properties, rendering them benign or malicious. This provides a more dynamic and adaptive approach to malware detection.

1.6.1.3 Spam Filtering

Machine Learning algorithms have proven to be effective tools in detecting and filtering spam emails. Naive Bayes classifiers, for example, can be trained on large datasets to identify common characteristics of spam messages, thereby augmenting the efficiency of spam filters.

1.6.2 Deep Learning Applications in Cybersecurity

Deep Learning, a specialized branch of machine learning that emulates the neural networks of the human brain, has demonstrated significant potential in addressing sophisticated cybersecurity challenges. The principal applications include :

1.6.2.1 Intrusion Detection

Deep learning models such as Convolutional Neural Networks (CNNs) can be utilized to enhance intrusion detection systems by learning to identify malicious activities from raw network data.

1.6.2.2 Malicious URL Detection

CNNs can also be deployed for the detection of malicious URLs. By training on raw input data, these networks can independently learn the characteristics of harmful URLs, thereby improving detection accuracy.

1.6.2.3 Anomaly Detection in Time-Series Data

Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory (LSTM) networks, are proficient in identifying anomalies in time-series data, such as network traffic or system logs. Their ability to 'remember' past inputs aids in recognizing irregularities that might indicate a cyber attack.

1.7 Conclusion

In this chapter we have presented an overview of artificial intelligence and its relation with machine and deep learning, we started this chapter with a definition of machine learning and its different existing architectures, and after that, we presented the knowledge discovery in the database process in detail and described each step of the process, next we presented a definition and basic concept of deep learning and most used architectures, in the end, we presented machine and deep learning domains of applications. in the next chapter, we will present a definition of cyber security and its core concepts in detail and we will give different types of cyber attacks, how intrusion detection systems work, and how vital network security is.

CHAPITRE 2

SECURITY TECHNIQUES

2.1 Introduction

Over the last few years, Cyber Attacks have been increasing, hackers keep adapting their strategies to exploit every possible vulnerability that might be found in a system or a network. Activities on the network could be proliferated effortlessly, leading to the emergence of intrusion detection systems. This chapter presents two essential sections,

the first section describes the general concept of Security information systems. And lists the various cyber-attacks. the second section describes the intrusion detection system, its types, and the method of attack detection.

2.2 The General Concept of Security

In order to understand the security concept, it is necessary to define some terms :

2.2.1 Vulnerabilities

is a weakness in a computer system that allows a threat actor, such as an attacker to undermine the integrity of this system. All systems have vulnerabilities. Even though the technologies are improving but the number of vulnerabilities are increasing. These vulnerabilities are the result of weaknesses in the design, implementation, or use of a hardware or software component of the system, but they are often software anomalies related to programming errors or bad practices. These software malfunctions are usually corrected as they are discovered but the user remains exposed to possible exploitation until the patch (temporary or final) is published and installed. And To exploit a vulnerability, an attacker must have at least one applicable tool or technique that can connect to a system weakness. In this frame, vulnerabilities are also known as the attack surface.

2.2.2 Threats

A threat in the context of computer security, Is a potential negative action or event that can take advantage of a vulnerability to breach security and negatively alter, or harm objects or objects of interest. the threat is anything that could compromise confidentiality, integrity, or availability of systems or data. It can lead to attacks on computer systems, networks, and more.

2.2.3 Attacks

attack is an attempt to break into the operating system of a remote computer. the attacker attempts network attacks to establish control over the operating system, cause an operating system denial of service, or access sensitive information without authorized access or permission. It happens to both individuals and organizations. There are two main types of network attacks : passive and active. In passive network attacks, malicious parties gain unauthorized access to networks, monitor, and steal private data without making any alterations. Active network attacks involve modifying, encrypting, or damaging data.

2.2.4 Malware

Malware , stand for "malicious software" is a file or code, typically delivered over a network, that infects, explores, steals or conducts virtually any behavior an attacker wants. And because malware comes in so many variants, there are numerous methods to infect computer systems. Though varied in type and capabilities, malware usually has one of the following objectives :

- Provide remote control for an attacker to use an infected machine.
- Send spam from the infected machine to unsuspecting targets.
- Investigate the infected user's local network.
- Steal sensitive data.

Most malware attacks result from zero-day exploits and unpatched applications or operating systems [37]

2.3 Security of Information Systems

The concept of information system security covers a range of methods, techniques, and tools to protect the resources of an information system to ensure the availability of services, confidentiality, and integrity of information. There are several different techniques that organizations can implement to protect their information systems, such as :

2.3.1 Firewalls

is a hardware or software-based network security device that monitors all incoming and outgoing traffic and decides whether to allow, block or drop specific traffic based on the rule set defined in its table. Once the rule is matched, associate action is applied to the network traffic. [38]
However, if the rule is not defined in the table, the firewalls will follow the default policy which is to drop any traffic.

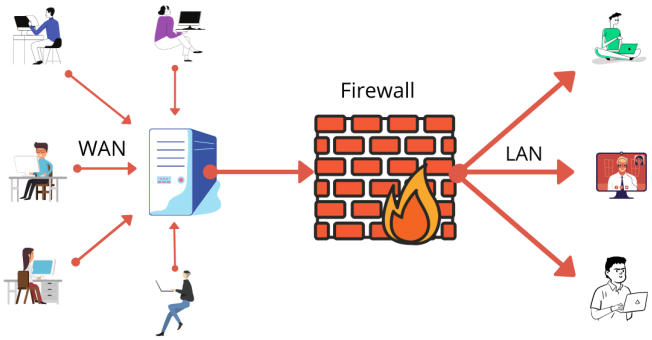


FIGURE 2.1 – illustration of Firewall
[39]

There are different types of firewalls, and each of them has its own set of purposes. Among these firewalls, we can mention

- **First Generation- Packet Filtering Firewall** It is a technique used to control network access by monitoring outgoing and incoming packets and allowing them to pass or drop based on the source and destination Internet Protocol (IP) addresses, protocols, and ports. This firewall is also known as a static firewall.
- **Second Generation- Stateful Inspection Firewall** : It is also a type of packet filtering that is used to control how data packets move through a firewall. It is also called dynamic packet filtering. This firewall can inspect if the packet belongs to a particular session or not. It only permits communication if and only if, the session is perfectly established between two endpoints else it will block the communication.
- **Third Generation- Application Layer Firewall** This firewall can examine application layer information such as HTTP requests. If the firewall finds some suspicious application that can be responsible for harming the network then the application gets blocked right away.
- **Next Generation Firewalls (NGFW)** : This firewall is called an intelligent firewall. It can perform all the tasks that are performed by the other types of firewalls. In addition, it includes additional features such as application awareness and control, integrated intrusion prevention, and cloud-delivered threat intelligence.
- **Circuit-Level Gateways** A circuit-level gateway is a firewall that provides User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) connection security and works between transport and application layers such as the session layer.
- **Software Firewall** The software firewall is a type of computer software that runs on the computers. It protects the system from any external attacks such as unauthorized access, malicious attacks, etc. This firewall notifies us about the danger that can occur if we open a particular mail or if we try to open a website that is not secure.
- **Hardware Firewall** A hardware firewall is a physical appliance that is deployed to enforce a network boundary. All network links crossing this boundary pass through this firewall, which enables it to perform an inspection of both inbound and outbound network traffic and enforce access controls and other security policies.

2.3.2 Encryption

Sometimes the information is transmitted over the internet so the risk of anyone accessing it increases and now the tools have to be strong to avoid it. In this scenario, the information can be easily accessed and modified by anyone. To avoid this, a new tool is put to work, Encryption. Using encryption, one can put confidential information into bits of unreadable characters that are difficult to decrypt and only the authorized receivers of the information can read it easily. There are various types of encryption, each with its own set of objectives such as :

2.3.2.1 Symmetric-key :

This is the simplest kind of encryption that involves only one secret key to cipher and decipher the information. Symmetric encryption is an old and best-known technique. It uses a secret key that can either be a number, a word, or a string of random letters. It is blended with the plain text of a message to change the content in a particular way. The sender and the recipient should know the secret key that is used to encrypt and decrypt all the messages. Blowfish, AES, RC4, DES, RC5, and RC6 are examples of symmetric encryption. The most widely used symmetric algorithm is AES-128, AES-192, and AES-256. [40]

The main disadvantage of symmetric key encryption is that all parties involved have to exchange the key used to encrypt the data before they can decrypt it.

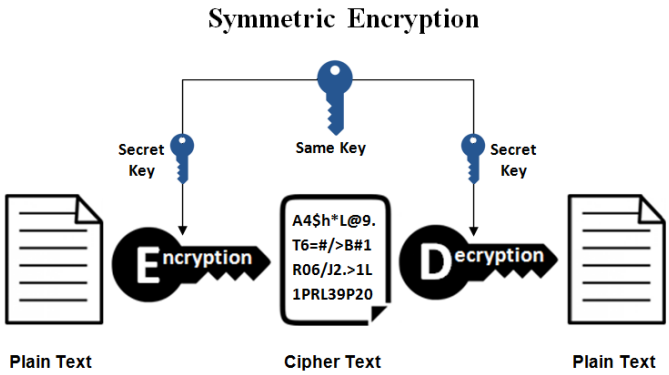


FIGURE 2.2 – illustration of Symmetric-key [41]

2.3.2.2 Asymmetric-key

Asymmetric encryption is also known as public key cryptography, which is a relatively new method, compared to symmetric encryption. Asymmetric encryption uses two keys to encrypt and decrypt a text. The first one named public key is used by anyone to decrypt the message and this is why asymmetric encryption uses two related keys to boost security. A public key is made freely available to anyone who might want to send you a message. The second private key is kept a secret. It permits to decrypt the encrypted message. The most widely used symmetric algorithm is RSA, Elliptic curve cryptography.

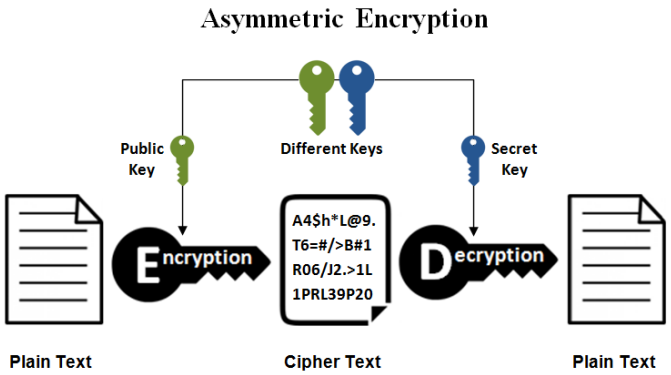


FIGURE 2.3 – illustration of Asymmetric Encryption
[41]

2.3.2.3 Shared Secret key

also known as a symmetric key or secret key, is a cryptographic key that is shared between two or more parties who want to communicate securely. It is called "symmetric" because the same key is used for both encryption and decryption.
The main characteristic of a shared secret key scheme is that all parties involved in the communication must possess the same secret key. This shared key serves as the basis for secure encryption and decryption operations.

2.3.2.4 Session-key

refers to a method of encrypting data using a temporary key that is generated for a specific session or communication. The session key is a symmetric key, meaning the same key is used for both encryption and decryption.

2.3.3 Security Policies

also called an information security policy or IT security policy. It is a definition of what it means to be secure for a system, organization, or other entity. A security policy is a document that spells out the rules, guidelines and procedures, and overall approach that an organization uses to maintain the confidentiality, integrity, and availability of the sensitive enterprise information system from different attacks. For an organization, it addresses the constraints on the behavior of its members as well as constraints imposed on adversaries by mechanisms such as doors, locks, keys, and walls. For systems, the security policy addresses constraints on functions and flow among them, constraints on access by external systems and adversaries including programs, and access to data by people. Security policies are living documents that are continuously updated and changing as technologies, vulnerabilities, and security requirements change.

2.3.4 VPN

VPN stands for "Virtual Private Network". It permits to establish a protected network connection when using public networks. VPN encrypts the internet traffic and adds security and anonymity to users when they connect to web-based services and sites. This protects the network from external access and makes it more difficult for third parties to track your activities online and steal data. The encryption takes place in **real time**. The entire network Traffic that is sent through the VPN tunnel will be encrypted and kept confidential from hackers on a network or the internet. Without an encryption mechanism, the traffic will be attacked and it would take millions of years for a computer to

decipher the code in the event of a brute-force attack. VPN is also the best and cheapest option to protect the data.

2.3.5 Demilitarized Zone

A demilitarized Zone or DMZ for short is a network barrier between the trusted and untrusted network. The main purpose of demilitarized zones is to add an extra layer of security to the internal network by restricting access to sensitive data and servers and A company can minimize the vulnerabilities of its Local Area Network, creating an environment safe from threats while also ensuring employees can communicate efficiently and share information directly via a safe connection. Organizations typically store external-facing services and resources, as well as servers for the Domain Name System (DNS), File Transfer Protocol (FTP), mail, proxy, Voice over Internet Protocol (VoIP), and web servers, in the DMZ.[42]

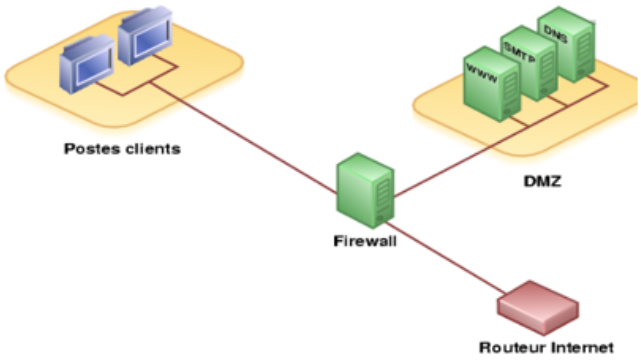


FIGURE 2.4 – illustration of Demilitarized Zone
[43]

2.4 Security Purpose

The goal of IT security is to ensure the protection of information in the organization by preserving aspects of :

2.4.1 Access Control

Is a fundamental component of security that can be used to regulate who or what can view or use company information and resources. Through authentication and authorization rules, access control policies make sure users are who they say they are and that they have appropriate access to company data there are two main types :

- **Access Control List (ACL)** : it is the list of individuals who are eligible to access the information
- **Role-Based Access Control List (RBAC)** : This list comprises the names of authorized personnel and the respective actions they are authorized to perform over the information.

2.4.2 Authentication

means confirming and verifying that a user is who they say they are. This ensures only those with authorized credentials gain access to secure systems. It is used by both the server and the client. The server uses authentication when someone wants to access the information, and the server needs to know who is accessing the information. The client uses it when he wants to know that it is the same server that it claims to be. When a user attempts to access information on a network, they must provide secret credentials to prove their identity such as :

- Biometrics : This method of authentication is based on the unique biological characteristics of each user such as fingerprints, voice or face recognition, signatures, and eyes.
- Fingerprints : Fingerprints are believed to be unique across the entire human population
- Signature : Every individual has a unique style of handwriting, and this feature is reflected in the signatures of a person.
- Voice : This method records the frequency pattern of the voice of an individual speaker.

Authentication allows us to grant access to the right user at the right time with confidence. But this doesn't occur in isolation.

2.4.3 Confidentiality

Confidentiality has been defined by the International Organization for Standardization (ISO) as "ensuring that information is only accessible to those who are authorized access".[44]

In information security, Sensitive information or data should be disclosed to authorized users only. thus, confidentiality is one of the five pillars of Information Assurance, it protects information from getting misused by any illegal access. And it makes sure that only authorized people can access private information. This property involves the implementation of encryption algorithms for example RC4 or DES.

2.4.4 Integrity

is the protection of data against unauthorized modification by the user and the stored data must remain unchanged in an information system, as well as during the transport of data. For this purpose, using the Hash Function (one-way function) such as MD5 or SHA1 which has a huge role in making a System Secure as it converts normal data given to it as an irregular value of fixed length. The Irregular value it outputs is known as "Hash Value». There are simply numbers but are often written in Hexadecimal.

2.4.5 Availability

Availability is Refers to the accessibility of information and systems and the networked services that house the information. The concept of availability states that a person or organization's data and information must be available to them when they need it. Since denying access to data and information is a typical security risk, cybersecurity practitioners must think about availability while developing cybersecurity strategies. There are mainly two threats to the availability of the system which are as follows :

- Denial of Service
- Loss of Data Processing Capabilities

2.5 The Various Cyber-Attacks

According to the statistics of Cisco, there had been more than twelve million effective attacks in the year 202. The same study estimates the growth of the former to arrive at fifteen million for the year 2023. Thus, all companies must learn about these attacks for their protection. We classify these attacks into three categories

2.5.1 Network Attacks

2.5.1.1 IP spoofing

IP spoofing [45]is the process of sending Internet Protocol (IP) packets with a fake source IP address in order to hide the identity of the sender, to impersonate another computer system, or both .IP spoofing, is. Cybercriminals can use IP spoofing to carry out harmful acts without being detected. It is a technique often used by bad actors to invoke DoS and DDoS attacks against a target device or the surrounding infrastructure. Sending and receiving IP packets is a primary way in which networked computers and other devices communicate, and constitutes the basis of the modern internet. All IP packets contain a header which precedes the body of the packet and contains important routing information, including the source address. In a normal packet, the source IP address is the address of the sender of the packet. If the packet has been spoofed, the source address will be forged. DoS attacks will often utilize spoofing with a goal of overwhelming a target with traffic while masking the identity of the malicious source.

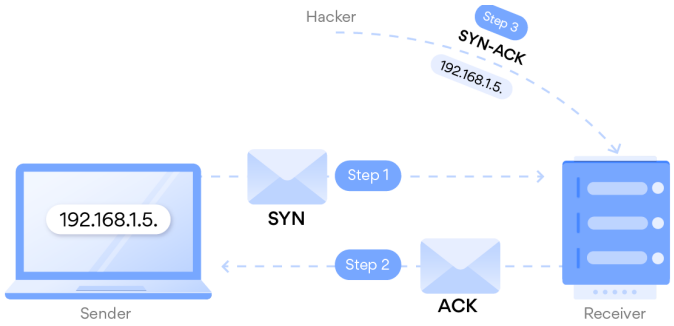


FIGURE 2.5 – illustration of IP spoofing attacks [45]

2.5.1.2 ARP Spoofing

ARP spoofing is a cyber-attack that allows hackers to intercept communications between network devices on a network. Hackers can also use ARP spoofing to alter or block all traffic between devices on the network.

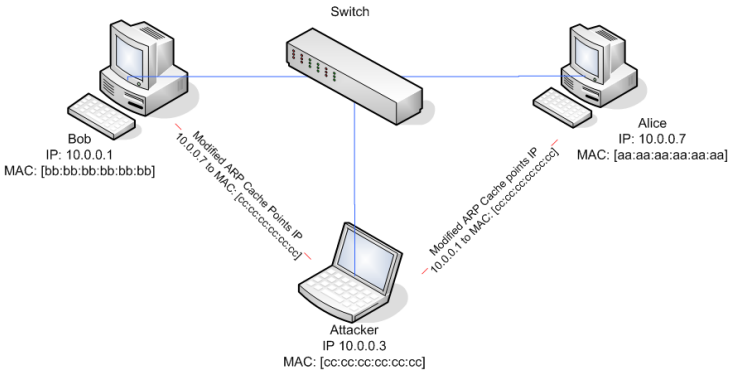


FIGURE 2.6 – illustration of ARP Spoofing attacks

The attack works as follows :

- The attacker must have access to the network. They scan the network to determine the IP addresses of at least two devices—let's say these are a workstation and a router.
- The attacker uses a spoofing tool, such as Arp spoof or Driftnet, to send out forged ARP responses.
- The forged responses advertise that the correct MAC address for both IP addresses, belonging to the router and workstation, is the attacker's MAC address. This fools both router and workstation to connect to the attacker's machine, instead of to each other
- The two devices update their ARP cache entries and from that point onwards, communicate with the attacker instead of directly with each other.
- The attacker is now secretly in the middle of all communications.

2.5.1.3 DNS Spoofing

is an attack in which DNS name resolution is tampered with specifically to the IP address of a domain name being faked. This means that a user is forced to navigate to a fake website disguised to look like a real one, with the intention of diverting traffic or stealing the credentials of the users. DNS spoofing is done by replacing the IP addresses stored in the DNS server with the ones under the control of the attacker. Once it is done, whenever users try to go to a particular website, they get directed to the false websites placed by the attacker in the spoofed DNS server. For example, when you go to Netflix .com on an unsecured network. The hacker redirects you to a fake Netflix page and he will able to steal your information such as your password.

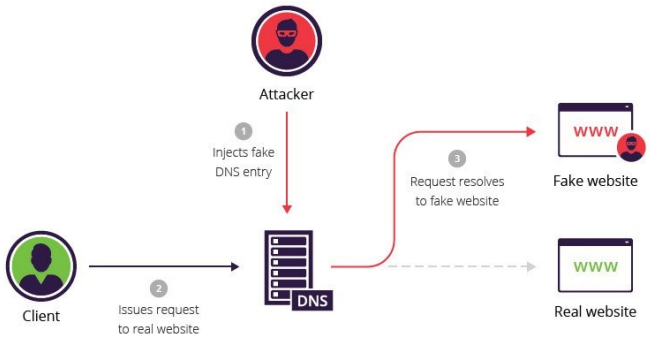


FIGURE 2.7 – illustration of DNS Spoofing attacks
[46]

There are mainly two methods by which DNS spoofing is carried out – DNS cache poisoning and DNS ID spoofing.

- In DNS cache poisoning, the local DNS server is replaced with a compromised DNS server containing customized entries of website names with the attacker's own IP addresses.
- In DNS ID spoofing, the packet ID and IP information generated for the resolve request sent by the client is duplicated with false information inside it. the client accepts the response containing information that is not expected.

2.5.1.4 Man-In-The-Middle (MitM)

MitM attacks are one of the oldest forms of cyberattack.it is a type of cyber attacks where a malicious attacker inserts a conversation between sender and receiver who believe that they are directly communicating with each other, both sender and receiver appear to communicate normally often to steal login credentials or personal information, spy on victims, sabotage communications, or corrupt data. The sender does not understand that the receiver is a malicious attacker and attacker trying to access or edit the message before re-transmitting it to the receiver.

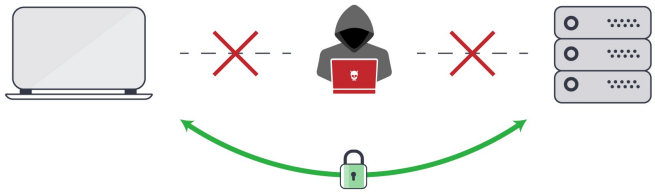


FIGURE 2.8 – illustration of Man-in-the-middle attacks [47]

2.5.2 Denial Of Service

A denial-of-service (DoS) attack is a type of cyber-attack in which a hacker or cybercriminal aims to render a computer or other device unavailable to its intended users by interrupting the device's normal functioning. DoS attacks typically function by overwhelming or flooding a targeted machine with requests until normal traffic is unable to be processed, resulting in denial-of-service to additional users. A DoS attack is characterized by using a single computer to launch the attack.

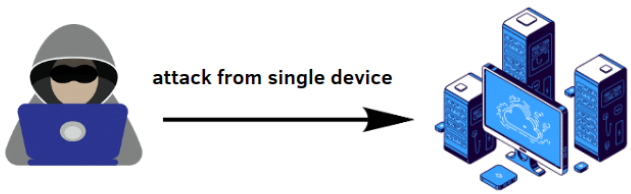


FIGURE 2.9 – illustration of Dos attacks

Here are a few known network attacks to make a service unavailable.

2.5.2.1 UDP Flood

UDP flood is a type of denial-of-service In which the attacker targets and overwhelms random ports on the host with IP packets containing User Datagram Protocol (UDP) packets to render a system, server, or machine unavailable for legitimate users. In this type of attack, the host looks for applications associated with these datagrams. When none are found, the host issues a “Destination Unreachable” packet back to the sender. UDP floods are highly effective and require few resources to execute. DoS or DDoS attacks are often part of highly complex threats that combine multiple attack vectors, to target an organization’s IT environment. Unlike TCP DDoS attacks, where threat actors leverage TCP SYN packets, UDP packets can be fragmented and cause as much harm as a normal UDP flood attack

2.5.2.2 Smurfing

it s a type of denial of service (DoS) attack that render computer networks inoperable in a short amount of time. The Smurf program accomplishes this by exploiting vulnerabilities of the Internet Protocol (IP) and Internet Control Message Protocols (ICMP) The steps in a Smurf attack are as follows :

- First, the malware creates a network packet attached to a false IP address
- Inside the packet is an ICMP ping message, asking network nodes that receive the packet to send back a reply
- These replies, or "echoes," are then sent back to network IP addresses again, setting up an infinite loop.
- When combined with IP broadcasting which sends the malicious packet to every IP address in a network ; the Smurf attack can quickly cause a complete denial of service.

2.5.2.3 Slowloris

Slowloris attack is a type of denial-of-service attack that targets layer 7 of the OSI model. This type of attack was designed to saturate a computer, a web server, a database, or an API by opening and maintaining multiple simultaneous TCP connections to a target FQDN and generating a low rate and/or volume of HTTP requests or HTTP connections per connected session. Slowloris tries to keep many connections to the target web server open and hold them open as long as possible It can be launched with very little bandwidth consumption, it is a script written in Python, and is extremely effective against many types of Web server software, but has proven highly-effective against Apache 1.x and 2.x. Slowloris attacks can be mitigated by :

- Limit the number of connections that a single IP address can open.
- Increase the minimum transfer speed allowed for any connection.
- Limit the length of time a customer is allowed to stay connected
- Increase the maximum number of clients allowed by the server.

2.5.2.4 GoldenEye

designed to overwhelm web servers’ resources by continuously requesting single or multiple URLs from many source-attacking machines. Goldeneye changes generated requests dynamically .it randomize user agents, referrers, and almost all of the various parameters

used. Goldeneye attempts to keep the connection alive and also adds a suffix to the end of URLs which will allow the request to bypass many CDN systems Also known as “No Cache”. When the servers’ limits of concurrent connections are reached, the server can no longer respond to legitimate requests from other users.

2.5.2.5 Ping Of Death

is a form of DoS attack in which an attacker sends a computer a series of incorrect pings and aims to disrupt a targeted machine by sending a packet larger than the maximum allowable size, The Data Link Layer, on the other hand, establishes a maximum frame size limit. A correctly formed ping packet is typically 56 bytes in size, or 64 bytes when the Internet Control Message Protocol (ICMP) header is considered, and 84 bytes including Internet Protocol (IP) version 4 header. However, any IPv4 packet including pings may be as large as 65,535 bytes. Some computer systems were never designed to properly handle a ping packet larger than the maximum packet size because it violates the Internet Protocol. Like other large but well-formed packets, a ping of death is fragmented into groups of 8 octets before transmission. However, when the target computer reassembles the malformed packet, a buffer overflow can occur, causing a system to freeze or crash.

2.5.2.6 DDos Attacks

DDos attacks is an amplification of the Dos attack. it partge the same goal which is to make a service unavailable to the user. To do this, the pirate will try to make himself master of a large number of machines. Thanks to faults (buffer overflows, RPC5 faults, ... etc.) .it will be able to take control of machines remotely and thus be able to control them. DDoS attacks can target a website, server, and other network resources.

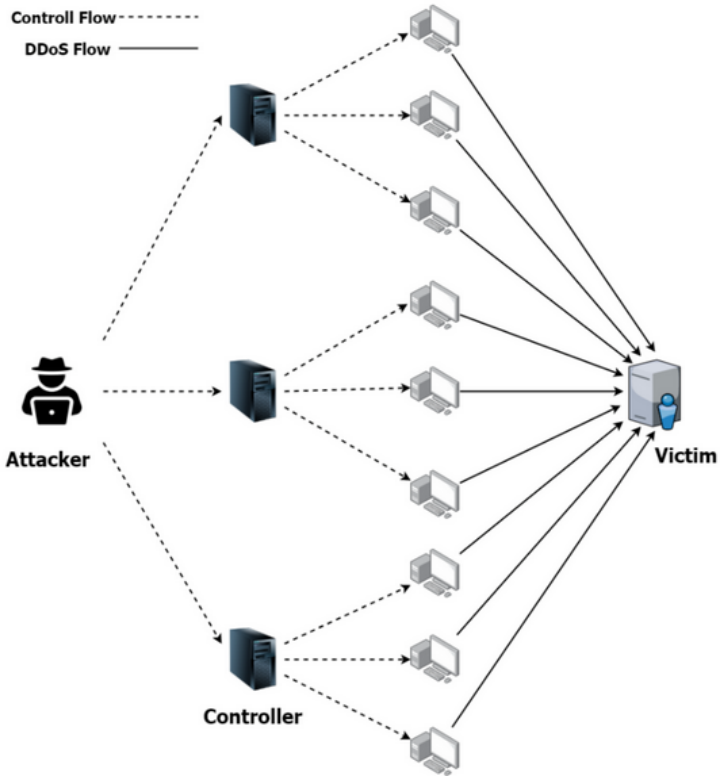


Figure 2.10 – illustration of DDos attacks

2.5.3 Data Attacks

It is crucial to ensure the security and integrity of data to protect sensitive information, maintain privacy, and prevent unauthorized access. However, numerous cyber attackers continuously seek to exploit vulnerabilities in systems and networks to compromise data.

2.5.3.1 Virus

A computer virus is a type of malware that attaches itself to other programs, self-replicates, and spreads between computers. When a virus infects a computer, it makes copies of itself and attaches to other files or documents. It then modifies those files and continues to spread. Viruses infect computers discreetly, and they're often designed to destroy personal files or gain control of devices. Making copies of themselves, some viruses are simply annoying while others cause damage to data and software that can destroy data, slow down the computer system resource

2.5.3.2 Worm

A worm virus refers to a malicious program that replicates itself without any human assistance and it does not need to attach itself to a software program in order to cause damage to data, the worm virus exploits vulnerabilities in your security software to steal sensitive information, install backdoors that can be used to access the system, corrupt files, and do other kinds of harm. It consumes large volumes of memory, bandwidth, and CPU.

2.5.3.3 Trojan Horse

Is a type of malicious code or software that is developed by hackers to disguise as legitimate software to gain access to victims' systems. A Trojan is designed to delete, modify, damage, block, or do some other harmful action on your data or network. Most trojans use the internet to pass information from infected machines to their criminal masters. They may also update themselves over the internet.

2.5.3.4 Logical Bomb

A logic bomb is part of a program or piece of code left lying in wait on a computer that will execute after a certain amount of time and take actions the owner of that computer would consider malicious. The actual code that does the dirty work sometimes referred to as slag code, might be a standalone application or hidden within a larger program. For example, a hacker may hide a piece of code that starts deleting files such as a salary database trigger, should they ever be terminated from the company.

2.6 Intrusion Detection System

An intrusion detection system (IDS) is an application that monitors network traffic and searches for known threats and suspicious or malicious activity. The IDS sends alerts to IT and security teams when it detects any security risks and threats. It is a software application that scans a network or a system for harmful activity or policy breaching. Most IDS solutions simply monitor and report suspicious activity and traffic when they detect an anomaly. However, some can go a step further by taking action when it detects anomalous activity, such as blocking malicious or suspicious traffic.

2.6.1 Classification Of Intrusion Detection System

The intrusion detection systems are classified into 5 types :

2.6.1.1 Network Intrusion Detection System (NIDS)

Network intrusion detection systems (NIDS) are set up at a planned point within the network to examine traffic from all devices on the network. It performs an observation of passing traffic on the entire subnet and matches the traffic that is passed on the subnets to the collection of known attacks. Once an attack is identified or abnormal behavior is observed, the alert can be sent to the administrator. An example of a NIDS is installing it on the subnet where firewalls are located in order to see if someone is trying to crack the firewall[48]

2.6.1.2 Host Intrusion Detection System (HIDS)

Host intrusion detection systems (HIDS) run on independent hosts or devices on the network. A HIDS monitors the incoming and outgoing packets from the device only and will alert the administrator if suspicious or malicious activity is detected. It takes a snapshot of existing system files and compares it with the previous snapshot. If the analytical system files were edited or deleted, an alert is sent to the administrator to investigate. An example of HIDS usage can be seen on mission-critical machines, which are not expected to change their layout.[48]

2.6.1.3 Protocol-based Intrusion Detection System (PIDS)

is an Intrusion Detection System that is typically installed on a web server, that used to monitor and analysis of the protocol between a user and the server. The system can protect your web server by monitoring inbound and outbound traffic. Because they focus on the protocol which is the way devices transmit information within a network. It is trying to secure the web server by regularly monitoring the HTTPS protocol stream and accepting the related HTTP protocol. As HTTPS is unencrypted and before instantly entering its web presentation layer then this system would need to reside in this interface, between to use the HTTPS[48]

2.6.1.4 Application Protocol-based Intrusion Detection System (APIDS)

An application Protocol-based Intrusion Detection System (APIDS) is a system or agent that generally resides within a group of servers. It identifies the intrusions by monitoring and interpreting the communication that occurs between applications and the server. It monitors the packets transmitted over application-specific protocols and identifies instructions, tracing them to individual users. A typical place for an APIDS would be between a web server and the database management system, monitoring the SQL protocol specific to the middleware or business logic as it interacts with the database[48]

2.6.1.5 Hybrid Intrusion Detection System

is an Intrusion Detection System that is made by the combination of two or more approaches of the intrusion detection system. In the hybrid intrusion detection system, the host agent or system data is combined with network information to develop a complete view of the network system. The hybrid intrusion detection system is more effective in comparison to the other intrusion detection system. [48]

2.6.2 Détection Method Of IDS

There are 2 main Intrusion Detection methods to identify malicious attacks or intrusion. However, both of these methods serve a different purpose.

2.6.2.1 Signature-Based Method

Signature-based IDSs use a database of known vulnerabilities or known attack patterns. It monitors the packets traversing the network, it compares these packets to the database of known attack signatures to flag any suspicious behavior.

For example, tools are available for an attacker to launch a SYN flood attack on a server by simply entering the IP address of the system to attack. The attack tool then floods the target system with synchronize (SYN) packets, but never completes the three-way Transmission Control Protocol (TCP) handshake with the final acknowledge (ACK) packet. If the attack isn't blocked, it can consume resources on a system and ultimately cause it to crash. However, this is a known attack with a specific pattern of successive SYN packets from one IP to another IP. The IDS can detect these patterns when the signature database includes the attack definitions. The process is very similar to what antivirus software uses to detect malware. One of the biggest limitations of signature-based IDS solutions is their inability to detect unknown attacks which require an update of signatures and patterns of new attacks.[48]

2.6.2.2 Anomaly-Based Method

also called heuristic-based or behavior-based .is an intrusion detection system for detecting both network and computer intrusions and misuse by monitoring system activity and classifying it as either normal or anomalous. Anomaly-based IDS was introduced to detect unknown malware attacks as new malware is developed rapidly. In anomaly-based IDS there is the use of machine learning to create a trustful activity model and anything coming is compared with that model and it is declared suspicious if it is not found in the model. The machine learning-based method has a better-generalized property in comparison to signature-based IDS as these models can be trained according to the applications and hardware configurations.[48]

2.6.3 Location Of The Intrusion Detection System In The Network

An intrusion detection is a very important asset in an information security architecture. IDS are available in Network and Host forms. Host intrusion detection is installed as an agent on a machine you wish to protect and monitor . network intrusion detection is placed behind a firewall and before the router on the edge of the network architecture. This location gives the highest visibility and maximizes effectiveness, as the firewall can handle different types of threats to an IDS, and both will want to be in front of the router so that malicious data does not reach the users to be on the network perimeter and can handle dropping a lot of the non-legitimate traffic.

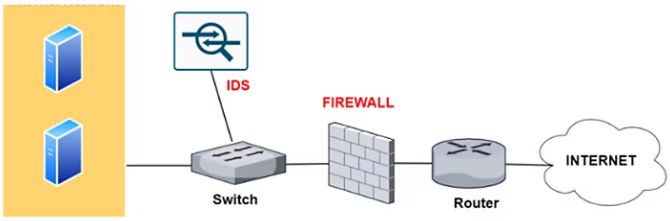


FIGURE 2.11 – illustration of location of the intrusion detection system in the network [49]

2.6.4 Comparison Of IDS With Firewalls

IDS and firewall both are related to network security devices but an IDS differs from a firewall as a firewall looks for threats externally by checking the traffic coming into the network by using IP addresses and port numbers to filter incoming and outgoing traffic based on predefined rules .it Placed at the perimeter of the network which is the first line of defense. Firewalls restrict access between networks to prevent intrusion and if an attack is from inside the network it doesn't signal. Intrusion detection systems offer a layer of security to the network. An IDS is either a hardware device or a software application that analyzes incoming network traffic for malicious activities or policy breaches and issues alerts when they are detected. It detects real-time traffic and searches for attack signatures or traffic patterns, then sends out alarms. So, using an IDS system and a firewall together can help to increase the network security than using just one of them.

2.7 Conclusion

In this chapter, we start by presenting the general context of security in the information system and the different cyberattacks, then we presented the subject which concerns the intrusion detection system by giving the definition, the classification and location of ids in the network.

In the next chapter, we will present a state-of-the-art work on what has recently been done by researchers on several approaches.

CHAPITRE 3

STATE-OF-THE-ART

3.1 Introduction

in our modern age, civilization has been more dependent on the digital world in everyday tasks due to the rise of data-driven applications and The rapid growth of the internet and its application, which resulted in the world being more dependent on personal data, this dependency made the digital world more valuable for criminal and hacker all over the world to make an effort and work on creating new attacks to sabotage those applications and steal valuable information. this rise of cyber-criminals resulted in the high demand for cyber-security experts by large companies to research the domain of intrusion detection and system protection. The balance between the cyber-criminal and cyber expert has been going back and forth with technological advancement but due to the nature of the cyber attacks it has become more difficult for experts to detect incoming attacks manually or with old detection systems. therefore cyber security experts have made a lot of research on how to optimize current intrusion detection systems and capitalizing on the rapid evolution of Artificial intelligence and the technical advancement in network research, experts have found new techniques to create smart intrusion detection systems that use machine and deep learning.

In this chapter, we will cover some of the research that has been made in using machine and deep learning algorithms in intrusion detection

3.2 Existence study

Many researchers suggest the application of machine learning and deep learning techniques to enhance intrusion detection systems. In this context, we present the state-of-the-art of recent works based on machine learning and deep learning.

TABLE 3.1 – literature review for the article "An Intelligent DDoS Attack Detection System Using Packet Analysis and Support Vector Machine"

an Intelligent DDoS Attack Detection System Using Packet Analysis and Support Vector Machine[50]				
author	purpose	data	Approach	Result
• Keisuke Kato. • Vitaly Klyuev.	• Suggest an approach using machine learning techniques and algorithms to improve DDoS attack detection system with high accuracy that can predict a forthcoming attack and detect the attack before the bandwidth is exhausted.	• The dataset contains approximately one hour of anonymized traffic traces that include the attack traffic on the victims and the response from the victims on August 4, 2007. The total compression dataset size is 5 GB and the original data set is 21 GB.	• The main purpose of this approach is to classify each IP address as an attacker or a normal. This binary classification Used (- 1) to represent "Normal IP" for the IP addresses of the victim pool and (+1) as "Attacker IP" for the IP addresses of the attacker pool, They Selected c-support vector classification (C-SVC) for training and testing data sets. • Created three types of training and test datasets including different patterns and different numbers of patterns. And test it through SVM algorithm.	• The first dataset has three features : the total number of packets, the time interval, and the total amount of bytes for each IP address. This dataset gives an Accuracy equal to 0.917. • The second dataset also has three other features : the number of packets per second, time interval, and the number of bytes per second. This dataset gives an Accuracy equal to 0.848. • Additionally, the third dataset has five features : the total number of packets, time interval, the total amount of bytes, the number of packets per second, and the mean packet size. This dataset gives an Accuracy equal to 0.986.

TABLE 3.2 – literature review for the article "A Novel PCA-Firefly Based XGBoost Classification Model for Intrusion Detection in Networks"

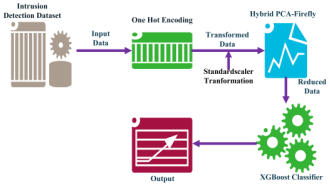
A Novel PCA-Firefly Based XGBoost Classification Model for Intrusion Detection in Networks Using GPU [51]				
author	purpose	data	Approach	Result
• Sweta Bhattacharya , • Siva Rama Krishnan S . • Praveen Kumar . • Reddy Maddikunta. • Rajesh Kaluri . • Saurabh Singh. • Thippa Reddy. • Gadekallu. • Mamoun Alazab. • Usman Tariq.	• Propose a hybrid principal component analysis (PCA) -Firefly-based XGBoost machine learning model for the classification of IDS datasets . • Make a comparison between the performance of the proposed model and the other models. .	• Using a dataset collected from the Kaggle website. This dataset contains 43 attributes holding categorical and numerical data, and 125,973 instances. Some of the important attributes of this dataset are duration, protocol type, service, source bytes, destination bytes, wrong fragments, urgent, hot, number of failed logins, logged-in, number compromised, error rate, etc.	• In this approach, they start with the One-Hot encoding algorithm for converting data from categorical data to a form, which is suitable for ML algorithms. The transformed data is then exposed to a hybrid PCA-firefly algorithm to reduce the number of redundant attributes. The XGBoost algorithm is applied to this reduced dataset for the classification of unanticipated cyber attacks.  • Study the performance of the KNN, naive Bayes, random forest, SVM, and XGBoost classifiers on intrusion detection datasets without dimensional reduction, in combination with PCA and hybrid PCA-firefly algorithms.	• The original dataset had 43 attributes. After the One-Hot encoding technique is applied, the number of attributes was enhanced to 3024 • After applying PCA, the number of attributes has been reduced to 2694. • The attributes are further reduced to 2386 with the application of the hybrid firefly algorithm. • The performance of XGBoost-PCA-firefly outperforms the other machine learning algorithms considering the accuracy (0.999), sensitivity (0.931) and specificity (0.999) metrics.

TABLE 3.3 – literature review for the article "boosting algorithms for network intrusion detection : A comparative evaluation of Real AdaBoost, Gentle AdaBoost and Modest AdaBoost"

boosting algorithms for network intrusion detection : A comparative evaluation of Real AdaBoost, Gentle AdaBoost and Modest AdaBoost [52]				
author	purpose	data	Approach	Result
• Amin Shahraki, • Mahmoud Abbasi. • Øystein Haugen .	• Study the feasibility of an intrusion detection system by using the most well-known version of AdaBoost algorithms. Comparative evaluation of Real AdaBoost, Gentle AdaBoost, and Modest AdaBoost in five publicly accessible data sets To show the effectiveness of AdaBoost algorithms.	• Using five publicly available intrusion datasets including **KDD Cup '99 dataset** that contains 41 features and more than five million records. they use 39 features and 50,000 records of the dataset to evaluate the AdaBoost algorithms. **UNSW-NB15** contains 49 features and two million and 540,044 records. select 36,628 records of the dataset and 39 features. **the TRAbID dataset** The simulation period for this dataset is 8 h,	• Using multiple learning models instead of a single model to obtain a better performance-boosting approach. • Comparing the error rate of Adaboost algorithms on different datasets and depths. • Comparing the running time of Adaboost algorithms on different datasets and depths in seconds.	The results show that Real AdaBoost and Gentle AdaBoost have the same performance as they are better than Modest AdaBoost in two aspects. First, they have approximately 70 better error rate than Modest Adaboost. Second, their error rate curves are stable after a few iterations as it is an important factor in IDSs. however Modest AdaBoost is about 7 faster than them and the boosting classifiers provide good results in comparison to other work.

author	purpose	data	Approach	Result
		with more than 615 million traffic packets. To evaluate the AdaBoost algorithms we select 36,628 records of the dataset and 36 features. • **NSL-KDD** is an optimized version of KDD CUP 99 that was created in 2009, this article selected 40,000 records of the dataset and 23 features. • **CICIDS2017** dataset This dataset contains more than 80 network traffic features they only consider 65 features for the simulations and 3100 records.		

TABLE 3.4 – literature review for the Article 'CNN-Based Network Intrusion Detection against Denial-of-Service Attacks'

CNN-Based Network Intrusion Detection against Denial-of-Service Attacks [53]				
author	purpose	data	Approach	Result
• Jiyeon Kim • Jiwon Kim • Hyunjung Kim • Minsun Shim • Eunjung Choi	Create a model based on a Convolutional Neural Network (CNN) and perform binary classification and multi-class classification attacks.	the data collected using : • **KDD dataset** : Each record has 41 network parameters and all data belong to one of the categories among 4 types of attacks (DoS, U2R, R2L, Probing) • **CSE-CIC-IDS 2018** : there are several attacks including Heartleech, LOIC UDP, TCP http and this is an advanced data set that includes all attacks in ISCXIDS 2012 and CICIDS 2017.	• A proposed CNN model against DoS attacks which considers hyperparameters such as the type of images (grayscale or RGB), the number of convolutions layers and the size of the kernel. By following these steps. • **Data Pre-processing** : The data pre-processing involves the following steps : 1. Protocol Types : There are three types of protocol types - ICMP, TCP, and UDP. These protocols are transformed into 3-dimensional vectors (1,0,0), (0,1,0), and (0,0,1) respectively using one-hot encoding. 2. Service Types : There are 67 types of services, including HTTP and FTP. They are transformed into a 67-dimensional vector.	• RGB images in binary and multi-class classifications are more accurate than grey-scale images • The type of scenarios : • both RGB and grayscale images performed best with three convolutional layers when the kernel sizes are 2 × 2 and 3 × 3 and when the kernel size is 4 × 4, the deployments of two convolutional layers had the highest accuracy.

No.	Scenario	Num. of Conv. Layer	Kernel Size	Num. of Kernels		
				Conv. Layer1	Conv. Layer2	Conv. Layer3
1	RGB-1	1	2×2	32	-	-
2	RGB-2	2	2×2	32	64	-
3	RGB-3	3	2×2	32	64	128
4	RGB-4	1	3×3	32	-	-
5	RGB-5	2	3×3	32	64	-
6	RGB-6	3	3×3	32	64	128
7	RGB-7	1	4×4	32	-	-
8	RGB-8	2	4×4	32	64	-
9	RGB-9	3	4×4	32	64	128
10	GS-1	1	2×2	32	-	-
11	GS-2	2	2×2	32	64	-
12	GS-3	3	2×2	32	64	128
13	GS-4	1	3×3	32	-	-
14	GS-5	2	3×3	32	64	-
15	GS-6	3	3×3	32	64	128
16	GS-7	1	4×4	32	-	-
17	GS-8	2	4×4	32	64	-
18	GS-9	3	4×4	32	64	128

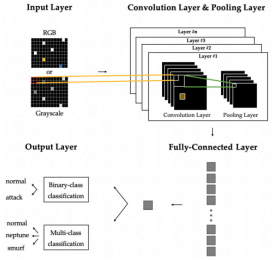
author	purpose	data	Approach	Result
			— Flag Types : The 9 features of the flag type are transformed into a 9-dimensional vector. • Conversion to Image : The 117-dimensional vector is converted into 13 9-pixel images. • **Model training with various scenario :** Create 18 types of scenarios take into consideration the hyperparameters such as the number of convolutional layer and the number of kernels to improve the performance of the model.  • **Model evaluation :** make a comparison with the RNN model to verify the performance of the proposed model.	• With KDD-datasets, the CNN model showed 0.99 or more results in binary and multiclass classifications, the RNN showed 0.99 accuracy in binary classification and 0.93 in multiclass classifications and with CSE-CIC-IDS 2018 datasets, the CNN model showed 0.915 of accuracy on average while the RNN model showed 0.65 of accuracy on average.

TABLE 3.5 – literature review for the article "Deep Learning for DDOS attack detection in Software Defined"

Deep Learning for DDOS attack detection in Software Defined Networking[54]				
author	purpose	data	Approach	Result
• Nisha Ahuja. • Gaurav Singal. • Debajyoti Mukhopadhyay	present a study on the application of deep learning techniques for detecting Distributed Denial-of-Service (DDoS) attacks in Software Defined Networking (SDN). The authors aim to classify network traffic into normal and malicious classes based on features given in a dataset.	The data used in the study is a Software Defined Network (SDN) based dataset provided by leadingindia.ai. The dataset consists of three different types of attacks and includes 22 features, mostly statistical. The dataset also includes information about the source and destination of the traffic, bytes, packets, duration of transfer, speed of transfer, etc.	The authors applied various deep learning techniques for classifying the traffic. These techniques include Convolution Neural Network (CNN), Recurrent Neural Network (RNN), Long-Short-Term-Memory (LSTM), CNN-LSTM, SVC-SOM, and Stacked Auto-Encoder-MLP (SAE-MLP). The authors also performed data pre-processing, applied machine learning classifiers, evaluated the models, and performed hyperparameter tuning.	The authors achieved the highest accuracy of 99.75%, with the Stacked Auto-Encoder-MLP (SAE-MLP) algorithm. Other models like CNN, LSTM, CNN-LSTM, and SVC-SOM also showed promising results, but SAE-MLP outperformed them in terms of accuracy.

TABLE 3.6 – literature review for the article "Machine-Learning-Based DDoS Attack Detection Using Mutual Information and Random Forest"

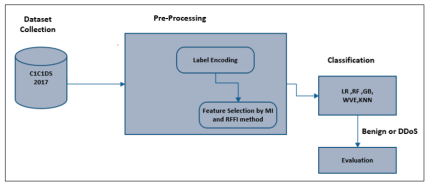
Machine-Learning-Based DDoS Attack Detection Using Mutual Information and Random Forest[55]				
author	purpose	data	Approach	Result
• Mona Alduailij • Qazi Waqas Khan • Muhammad Tahir • Muhammad Sardaraz • Mai Alduailij • and Fazila Malik	• proposes a method for detecting DDoS attacks in cloud computing using feature selection (MI, RFFI) and machine learning techniques to reduce miss-classification errors.	• in this article, they have used 2 different datasets acquired from the CIC institution • The CIC-IDs 2017 dataset consists of 3.1 million traffic flow records with 5 days of log files of traffic flow. The Friday evening log file is used for experimentation because it has 225,711 instances of Dos/DDoS attacks and 79 features. • The CIC-DDoS 2019 dataset contains 1,209,961 instances of different DDoS attacks and 84 input features, with a binary class label of benign and DDoS and divided by DDos type	• In this article, they used the Mutual Information (MI) and Random Forest Feature Importance (RFFI) methods for Feature selection from the dataset, which resulted in 3 different datasets with different sets of features (16, 19, 23, full features). • In the next step, They applied different machine learning algorithms like K-Nearest Neighbor (KNN), Logistic Regression (LR), Random Forest (RF), Gradient Boosting (GB), and Weighted Voting Ensemble (WVE) methods for attack detection. • Finally, for evaluation of The performance of the proposed method, they have used precision, recall, F measure, and accuracy. 	• The results show that RF has the best overall performance in detecting DDoS attacks, with a low miss classification rate compared to other methods, when using 16, 19, or 23 features. • The RFFI method is used to select the most relevant 19 features for DDoS attack detection and the prediction accuracy of WVE and RF is better than other methods. • The results show that using MI and RFFI algorithms to feature selection increases the performance of the models with the highest score on 23 features using the RF model. • The confusion matrix results show that the RF model has the lowest miss classification rate compared to other methods, using 23 features selected by MI and RFFI.

TABLE 3.7 – literature review for the article "Improving Ada Boost-based Intrusion Detection System (IDS) Performance on CIC IDS 2017 Dataset"

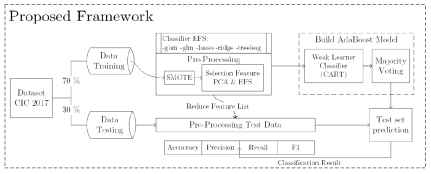
Improving Ada Boost-based Intrusion Detection System (IDS) Performance on CIC IDS 2017 Dataset[56]				
author	purpose	data	Approach	Result
• Arif Yulianto • Parman Sukarno • Novian Anggis Suwastika	• This article proposes a new approach for improving the performance of network intrusion detection. The author highlights the limitation of previous approaches and proposes an approach that overcomes these limitations by using AdaBoost with SMOTE and PCA for intrusion detection. The approach is evaluated using various datasets including NSL KDD, KDD Cup '99, and Canadian Institute for Cybersecurity (CIC) 2017.	• The CIC IDS 2017 dataset was developed by the Faculty of Computer Science, University of New Brunswick in 2017 based on the ISCX 2012 dataset. The dataset consists of 5 days of data collection with over 225,745 instances and more than 80 features, including normal and intrusion network activity. The attack simulation in the dataset is divided into 7 categories, including DDoS attacks.	• The proposed approach uses the Synthetic Minority Oversampling Technique (SMOTE) to handle the class imbalance in the dataset and improve the sensitivity of the arrangement for minority classes. • The feature selection is based on obtaining weighted values from the ensemble result by ranking them based on the information value obtained. • The model is created by training ADA boost classification algorithm on the training data. • The evaluation of the technique is done using accuracy, precision, recall and F1 metrics. 	• The proposed method outperforms the previous method with an accuracy of 81.83%, precision of 81.83%, recall of 100%, and F1 Score of 90.01%

TABLE 3.8 – literature review for the article "AdaBoost-Based Algorithm for Network Intrusion Detection"

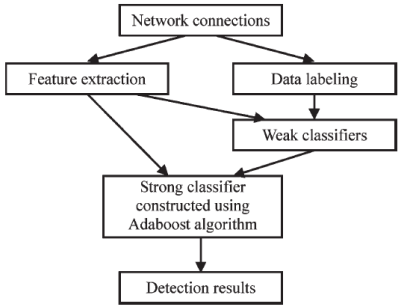
AdaBoost-Based Algorithm for Network Intrusion Detection[57]				
author	purpose	data	Approach	Result
• Weiming Hu • Wei Hu • Steve Maybank	This article proposes an intrusion detection approach based on the AdaBoost algorithm that achieves a high detection rate with a low false-alarm rate and low computational complexity. this approach uses decision stumps as weak classifiers and provides decision rules for both categorical and continuous features.	• The article uses the Knowledge Discovery and Data Mining CUP 1999 data set for testing the proposed intrusion detection algorithm. • The dataset contains four types of attacks : denial of service, user to root, remote to local, and PROBE. and includes 41 features •	• The approach consists of three modules : feature extraction, design of weak classifiers, and construction of the strong classifier. • Feature extraction involves extracting three groups of features for each network connection. • Design of weak classifiers involves creating a group of simple, easy-to-implement classifiers with low accuracy. • The strong classifier is obtained by combining the weak classifiers to increase classification accuracy. • The strong classifier is trained using sample data and then used to classify new network connections as either "normal" or "attack". 	• The approach was tested on the KDD CUP data set and showed competitive performance in terms of detection and false-alarm rates compared to existing solutions • The run speed of the approach was faster in the learning stage than the published run speeds of existing approaches

TABLE 3.9 – literature review for the article "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks"

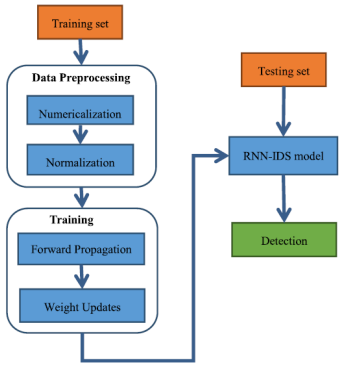
A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks[58]				
author	purpose	data	Approach	Result
• CHUANLONG YIN • YUEFEI ZHU • JINLONG FEI • XINZHENG HE	• This article proposes a deep learning approach for intrusion detection using recurrent neural networks (RNN) and studies its performance in comparison with other machine learning methods such as J48, ANN, random forest, SVM, etc. • The authors in this article highlight the impact of the number of neurons and different learning rates on the accuracy of the proposed model and compare it with the traditional machine learning methods on a benchmark dataset.	The study uses the KDD dataset that contains four different types of attacks (DOS, U2R, R2L, Probing) and has 41 feature	• A Recurrent Neural Network (RNN) model is used for supervised classification learning, the approach uses RNN because they have a loop that memorizes previous information and applies it to the current output, RNN same as all neural networks has an input layer, an output layer and hidden layer • The training of the RNN model consists of two parts Forward Propagation calculates the output values and Back Propagation updates the weights after a defined number of iterations called the learning rate 	• Two experiments were conducted to study the performance of the RNN model for binary (Normal vs anomaly) and five-category (Normal, DOS, R2L, U2R and Probe) classification. • The RNN model had an accuracy of 81.29% on the test set in the case of five-category classification and 83.28% in the case of binary classification, which was better than other classification algorithms including J48, naive Bayes, random forest, and multi-layer perceptron. • The best performance of the RNN model was achieved with 80 hidden nodes and a learning rate of 0.1 in binary classification and with 80 hidden nodes and a learning rate of 0.5 in five-category classification

TABLE 3.10 – literature review for the article "Autoencoder-based Unsupervised Intrusion Detection using Multi-Scale Convolutional Recurrent Networks"

Autoencoder-based Unsupervised Intrusion Detection using Multi-Scale Convolutional Recurrent Networks[59]				
author	purpose	data	Approach	Result
• Amardeep Singha • Julian Jang-Jaccarda	The purpose of the paper is to propose a new approach for intrusion detection using unsupervised machine learning. The authors aim to address the limitations of current intrusion detection solutions, which often require high computational cost, manual support in fine-tuning intrusion detection models, and labeling of data that limit real-time processing of network traffic.	The authors of this paper, utilized three distinct datasets for their research : NSL-KDD, UNSW-NB15, and CICDDoS2019. These datasets, which contain a variety of simulated intrusion scenarios, were used to evaluate the performance of different intrusion detection systems. The NSL-KDD dataset is designed to address some issues of the KDD'99 dataset, while the UNSW-NB15 and CICDDoS2019 datasets are designed for comprehensive intrusion and DDoS attack detection, respectively. Prior to being input into the model, these datasets underwent preprocessing, which included one-hot encoding for categorical features and min-max normalization for numerical features.	The authors propose a unified Autoencoder based on combining multi-scale convolutional neural network and long short-term memory (MSCNN-LSTM-AE) for anomaly detection in network traffic. The model first employs Multiscale Convolutional Neural Network Autoencoder (MSCNN-AE) to analyze the spatial features of the dataset, and then latent space features learned from MSCNN-AE employs Long Short-Term Memory (LSTM) based Autoencoder Network to process the temporal features. The model further employs two Isolation Forest algorithms as error correction mechanisms to detect false positives and false negatives to improve detection accuracy.	The proposed method significantly outperforms the conventional unsupervised methods and other existing studies on the dataset. The authors claim that their model achieves a higher recall and f-score compared to other state-of-the-art models.

3.3 Conclusion

Throughout this chapter, we have provided a summary of the work done and explore the advancements in developing intrusion detection systems using machine learning and deep learning techniques. These articles not only showcase the progress and findings achieved but also serve as a valuable resource for understanding the current state of research in this area.

In the next chapter, We will present in detail the architecture of our Proposed approach for intrusion detection systems

CHAPITRE 4

CONCEPTION

4.1 introduction

In this chapter, we will introduce our proposed approach for intrusion detection, Which utilizes Machine and Deep Learning techniques. We will provide a detailed overview of the design of our approach, which will enable a better understanding of the various functionalities of our system.

This approach consists of several phases which are data pre-processing, handling the normal traffic and classification of attack traffic. Moreover, we will mention the different techniques and tools that we will employ during each phase of our approach.The objective of our work is to provide a reliable and efficient intrusion detection system by leveraging the capabilities of both traditional Machine Learning techniques and modern Deep Learning methods. The approach is designed to address the limitations of traditional methods in detecting new and sophisticated attacks, as well as the challenges posed by the massive and complex network traffic data.

We aim to develop a robust and adaptive system that can learn from the data and adapt to new situations in real time. To achieve this goal, we have divided our approach into several phases, each of which plays a critical role in the overall success of the system. In this chapter, we will provide a detailed description of each phase and explain how they work together to achieve the desired results. We will also discuss the challenges and opportunities of our approach and present some preliminary results that demonstrate its effectiveness.

4.2 General Architecture

In computer and network security, Intrusion Detection Systems (IDS) play a crucial role. While existing classification methods have their own advantages and disadvantages, the issue with machine learning algorithms in intrusion detection lies in their potential to generate false positives and false negatives.
Our system has a mean target To balance the trade-off between detecting as many threats as possible and reducing false negatives while minimizing the number of false alerts (reducing false positives) which can have a significant impact on the effectiveness of intrusion detection systems.
It is composed of model stacking two different models with different objective, the first model is a deep neural network that uses the Autoencoder architecture to detect any anomaly in the network traffic, this result in the binary classification where this model

distinguish the benign traffic from the anomaly traffic then passes the anomaly traffic to the second model. the second model consists of an ensemble model to combine different machine learning classification models with the objective of multiclass classification of the anomaly in the network traffic .
In the next parts,we will Provide an elaborate discussion of our system and its phase as illustrated in the following figure :

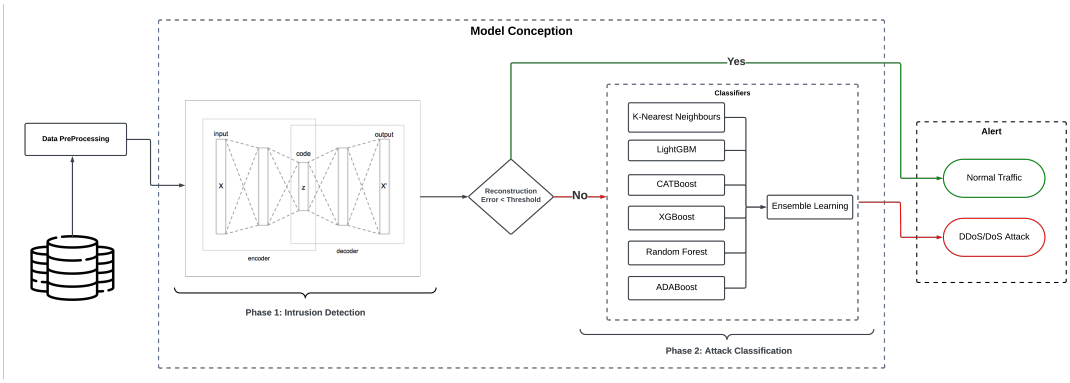


FIGURE 4.1 – illustration of the Approach Architecture

4.3 Data Collection and Preprocessing

Data collection and preprocessing are important steps in the machine-learning workflow. As part of our work, We performed specific steps to gather and preprocess relevant data, as illustrated in the following figure :

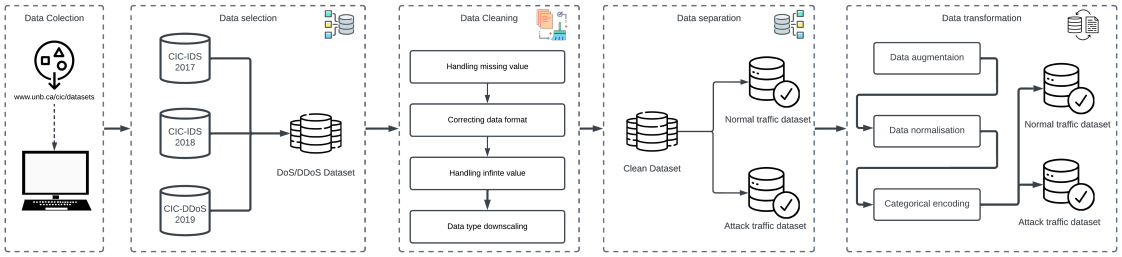


FIGURE 4.2 – illustration of the Preprocessing Architecture

4.3.1 Data Collection

We use three publicly available datasets to create a larger dataset for our model. It means we were able to improve the performance of our proposed model and achieve better results than previous research that only used one of these datasets. Details about each dataset will be presented in the following context :

1. **DDoS-2019[60]** : that contain benign and the most common DDoS attacks such as 'LDAP', 'MSSQL', 'NTP', 'NetBIOS', 'Syn', 'TFTP' and 'UDP'.it has been organized per day. Each day, they recorded the raw data including the network traffic and event logs. and which contain 88 feature extraction traffic. this dataset contains 88 attributes which are listed in the following figure 4.3 :

```
['ACK Flag Count' 'Active Max' 'Active Mean' 'Active Min' 'Active Std'
 'Average Packet Size' 'Avg Bwd Segment Size' 'Avg Fwd Segment Size'
 'Bwd Avg Bulk Rate' 'Bwd Avg Bytes/Bulk' 'Bwd Avg Packets/Bulk'
 'Bwd Header Length' 'Bwd IAT Max' 'Bwd IAT Mean' 'Bwd IAT Min'
 'Bwd IAT Std' 'Bwd IAT Total' 'Bwd PSH Flags' 'Bwd Packet Length Max'
 'Bwd Packet Length Mean' 'Bwd Packet Length Min' 'Bwd Packet Length Std'
 'Bwd Packets/s' 'Bwd URG Flags' 'CWE Flag Count' 'Destination IP'
 'Destination Port' 'Down/Up Ratio' 'ECE Flag Count' 'FIN Flag Count'
 'Flow Bytes/s' 'Flow Duration' 'Flow IAT Max' 'Flow IAT Mean'
 'Flow IAT Min' 'Flow IAT Std' 'Flow ID' 'Flow Packets/s'
 'Fwd Avg Bulk Rate' 'Fwd Avg Bytes/Bulk' 'Fwd Avg Packets/Bulk'
 'Fwd Header Length' 'Fwd Header Length.1' 'Fwd IAT Max' 'Fwd IAT Mean'
 'Fwd IAT Min' 'Fwd IAT Std' 'Fwd IAT Total' 'Fwd PSH Flags'
 'Fwd Packet Length Max' 'Fwd Packet Length Mean' 'Fwd Packet Length Min'
 'Fwd Packet Length Std' 'Fwd Packets/s' 'Fwd URG Flags' 'Idle Max'
 'Idle Mean' 'Idle Min' 'Idle Std' 'Inbound' 'Init_Win_bytes_backward'
 'Init_Win_bytes_forward' 'Label' 'Max Packet Length' 'Min Packet Length'
 'PSH Flag Count' 'Packet Length Mean' 'Packet Length Std'
 'Packet Length Variance' 'Protocol' 'RST Flag Count' 'SYN Flag Count'
 'SimillarHTTP' 'Source IP' 'Source Port' 'Subflow Bwd Bytes'
 'Subflow Bwd Packets' 'Subflow Fwd Bytes' 'Subflow Fwd Packets'
 'Timestamp' 'Total Backward Packets' 'Total Fwd Packets'
 'Total Length of Bwd Packets' 'Total Length of Fwd Packets'
 'URG Flag Count' 'Unnamed: 0' 'act_data_pkt_fwd' 'min_seg_size_forward']
```

FIGURE 4.3 – Features of Dataset CIC-DDoS-2019

2. **CIC-IDS2018**[61] : Which use the notion of profiles to generate datasets in a systematic manner, which will contain detailed descriptions of intrusions and abstract distribution models for applications, protocols, or lower-level network entities. It has normal traffic and some attacks such as DoS attacks (Hulk, GoldenEye, Slowloris, Slowhttptest), and DDoS attacks (DDoS attacks-LOIC-HTTP, DDoS-LOIC-UDP, DDoS-HOIC). This dataset contains 80 attributes which are listed in the following figure 4.4 :

```
['ACK Flag Cnt' 'Active Max' 'Active Mean' 'Active Min' 'Active Std'
 'Bwd Blk Rate Avg' 'Bwd Byts/b Avg' 'Bwd Header Len' 'Bwd IAT Max'
 'Bwd IAT Mean' 'Bwd IAT Min' 'Bwd IAT Std' 'Bwd IAT Tot' 'Bwd PSH Flags'
 'Bwd Pkt Len Max' 'Bwd Pkt Len Mean' 'Bwd Pkt Len Min' 'Bwd Pkt Len Std'
 'Bwd Pkts/b Avg' 'Bwd Pkts/s' 'Bwd Seg Size Avg' 'Bwd URG Flags'
 'CWE Flag Count' 'Down/Up Ratio' 'Dst Port' 'ECE Flag Cnt' 'FIN Flag Cnt'
 'Flow Byts/s' 'Flow Duration' 'Flow IAT Max' 'Flow IAT Mean'
 'Flow IAT Min' 'Flow IAT Std' 'Flow Pkts/s' 'Fwd Act Data Pkts'
 'Fwd Blk Rate Avg' 'Fwd Byts/b Avg' 'Fwd Header Len' 'Fwd IAT Max'
 'Fwd IAT Mean' 'Fwd IAT Min' 'Fwd IAT Std' 'Fwd IAT Tot' 'Fwd PSH Flags'
 'Fwd Pkt Len Max' 'Fwd Pkt Len Mean' 'Fwd Pkt Len Min' 'Fwd Pkt Len Std'
 'Fwd Pkts/b Avg' 'Fwd Pkts/s' 'Fwd Seg Size Avg' 'Fwd Seg Size Min'
 'Fwd URG Flags' 'Idle Max' 'Idle Mean' 'Idle Min' 'Idle Std'
 'Init Bwd Win Byts' 'Init Fwd Win Byts' 'Label' 'PSH Flag Cnt'
 'Pkt Len Max' 'Pkt Len Mean' 'Pkt Len Min' 'Pkt Len Std' 'Pkt Len Var'
 'Pkt Size Avg' 'Protocol' 'RST Flag Cnt' 'SYN Flag Cnt'
 'Subflow Bwd Byts' 'Subflow Bwd Pkts' 'Subflow Fwd Byts'
 'Subflow Fwd Pkts' 'Timestamp' 'Tot Bwd Pkts' 'Tot Fwd Pkts'
 'TotLen Bwd Pkts' 'TotLen Fwd Pkts' 'URG Flag Cnt']
```

FIGURE 4.4 – Features of dataset CIC-IDS-2018

3. **CIC-IDS2017**[62] : Dataset that contains benign and the Most up-to-date common attacks which resemble true real-world data (PCAPs). This dataset was Extracted into 85 network flow features from the generated network traffic using CIC-FlowMeter and delivered the network flow dataset as a CSV file. these attributes are listed in the following figure 4.5 :

```
['ACK Flag Count' 'Active Max' 'Active Mean' 'Active Min' 'Active Std'
 'Average Packet Size' 'Avg Bwd Segment Size' 'Avg Fwd Segment Size'
 'Bwd Avg Bulk Rate' 'Bwd Avg Bytes/Bulk' 'Bwd Avg Packets/Bulk'
 'Bwd Header Length' 'Bwd IAT Max' 'Bwd IAT Mean' 'Bwd IAT Min'
 'Bwd IAT Std' 'Bwd IAT Total' 'Bwd PSH Flags' 'Bwd Packet Length Max'
 'Bwd Packet Length Mean' 'Bwd Packet Length Min' 'Bwd Packet Length Std'
 'Bwd Packets/s' 'Bwd URG Flags' 'CWE Flag Count' 'Destination IP'
 'Destination Port' 'Down/Up Ratio' 'ECE Flag Count' 'FIN Flag Count'
 'Flow Bytes/s' 'Flow Duration' 'Flow IAT Max' 'Flow IAT Mean'
 'Flow IAT Min' 'Flow IAT Std' 'Flow ID' 'Flow Packets/s'
 'Fwd Avg Bulk Rate' 'Fwd Avg Bytes/Bulk' 'Fwd Avg Packets/Bulk'
 'Fwd Header Length' 'Fwd Header Length.1' 'Fwd IAT Max' 'Fwd IAT Mean'
 'Fwd IAT Min' 'Fwd IAT Std' 'Fwd IAT Total' 'Fwd PSH Flags'
 'Fwd Packet Length Max' 'Fwd Packet Length Mean' 'Fwd Packet Length Min'
 'Fwd Packet Length Std' 'Fwd Packets/s' 'Fwd URG Flags' 'Idle Max'
 'Idle Mean' 'Idle Min' 'Idle Std' 'Init_Win_bytes_backward'
 'Init_Win_bytes_forward' 'Label' 'Max Packet Length' 'Min Packet Length'
 'PSH Flag Count' 'Packet Length Mean' 'Packet Length Std'
 'Packet Length Variance' 'Protocol' 'RST Flag Count' 'SYN Flag Count'
 'Source IP' 'Source Port' 'Subflow Bwd Bytes' 'Subflow Bwd Packets'
 'Subflow Fwd Bytes' 'Subflow Fwd Packets' 'Timestamp'
 'Total Backward Packets' 'Total Fwd Packets'
 'Total Length of Bwd Packets' 'Total Length of Fwd Packets'
 'URG Flag Count' 'act_data_pkt_fwd' 'min_seg_size_forward']
```

FIGURE 4.5 – Features of dataset CIC-IDS-2017

4.3.1.1 Dataset

In our work, we have developed a new dataset for our approach by choosing the files that contain the records for DDoS attacks and benign traffic from CIC-IDS-2018 and CIC-IDS-2017. These records are later cleaned and added to the clean version of CIC-DDoS-2019 to create our Dataset that contains 80 attributes, the decision to choose 80 attributes was made after investigation and consultation with a field expert to ensure the feasibility of merging the datasets, and these attributes were obtained by dropped the following attributes form both CIC-IDS-2017 and CIC-DDoS-2019 :

- Destination IP, Flow ID, Source IP, Source Port : those attributes were deleted to evade over-fitting on the simulation configuration according to the domain expert.

  Additionally, we dropped those four attributes in the CIC-DDoS-2019 dataset :

- Fwd Header Length.1 : this attribute was found duplicated with the attribute 'Fwd Header Length'.
- Inbound : dropped to evade over-fitting on the simulation network configuration.
- SimillarHTTP : this attribute contains a URL web registry that is irrelevant to our case study.
- Unnamed : 0 : it is the index column that raises by using the Dataframe function to CSV () from the pandas' library without defining index= False.

Finally, we divided this grouped dataset into two parts :

- **benign dataset** that contains normal traffic
- **attacks dataset** that contains records of various DDoS-Dos

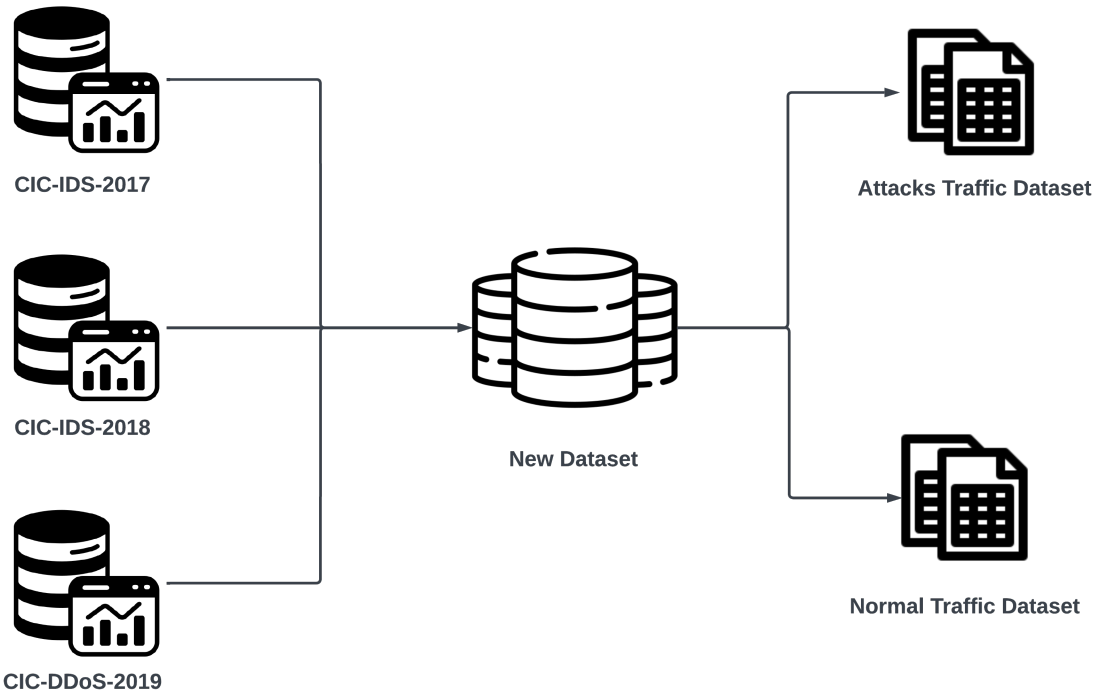


FIGURE 4.6 – illustration of the New Dataset

4.3.2 Data Pre-processing

The application of machine learning algorithms in our solution required certain conditions to optimize the global quality of the model training results and evaluation tests. our dataset is saved as a CSV file that contains records of network traffic in tabular which enables us to perform the following data cleaning process

4.3.2.1 Data Cleaning

Data cleaning is a crucial step in the data preprocessing phase, aimed at ensuring the accuracy, consistency, and completeness of the training data used in our intrusion detection system. Within our methodology, we have implemented a comprehensive data-cleaning process consisting of four steps.

A. Handling missing values The first step in our data cleaning process involves addressing the issue of missing data. Missing values can arise due to various reasons, such as Not-a-Number (NaN) values, null values, or infinite values. To tackle this problem, we have employed various techniques, including dropping rows containing missing values and replacing them using statistical functions.

Given the massive size of our dataset, we have made a decision to replace infinite and null values with NaN, a standard representation for missing values. Subsequently, we have chosen to drop all rows that contain NaN values. This approach allows us to maintain the integrity and size of the dataset while ensuring that the missing values do not compromise the overall quality of the data.

B. Handling duplicated data In the data preprocessing phase, specifically in the second step, we address the issue of duplicated data to ensure the high quality of our dataset. Duplicated data refers to instances where multiple rows in the dataset possess identical values across all columns. Dealing with duplicated data is crucial as it can lead to biased results and hinder the effectiveness of our intrusion detection system.

To resolve this issue, we have implemented a rigorous approach to identify and eliminate duplicated rows from our dataset. By thoroughly analyzing the dataset, we identify any instances of duplication and promptly remove them. This step guarantees that our dataset exclusively consists of unique and non-redundant information, enabling us to maintain the integrity and reliability of the data.

By eliminating duplicated data, we ensure that our dataset is streamlined and optimized for subsequent analysis and modeling. This meticulous process contributes to the overall accuracy and effectiveness of our intrusion detection system, allowing it to make informed decisions based on reliable and non-repetitive data.

C. Correcting Data Types One crucial aspect of data preprocessing is ensuring that the data is in the correct format for analysis. In our case, we encountered a situation where certain columns in the dataset were registered as object types, despite their content consisting of numerical values. This discrepancy between the data type and the actual nature of the data can introduce significant challenges during analysis and modeling processes.

To address this issue, we undertook the task of converting each column to its appropriate numerical value representation. By rectifying the data type inconsistency, we ensure that subsequent operations, such as statistical calculations and machine learning algorithms, can effectively interpret and process the data. Additionally, working with consistent numerical data types facilitates a more streamlined analysis workflow and reduces the risk of errors arising from incompatible data types.

D. Down-Scaling Data Type In addition to correcting the data type, we recognized the significance of optimizing the usage of RAM resources in our approach. Large datasets with high memory requirements can pose challenges in terms of computational efficiency, particularly when dealing with resource-constrained environments. Therefore, it becomes imperative to minimize the memory footprint of the dataset without compromising the integrity and information content of the data.

To achieve this objective, we employed a Down Scalar, a technique that transforms the data type of each column to the minimum possible value based on the range of values within that column. By downsizing the data type, such as using int8, int16, int32, or int64, we can significantly reduce the memory consumed by the dataset, allowing for more efficient processing and analysis.

The Down-scaling technique is advantageous as it adapts the data type to the specific characteristics of each column. By considering the minimum and maximum values within a column, we can accurately determine the most suitable data type that captures the range of values while minimizing the memory requirements. This approach ensures that the data representation remains faithful to the original values while optimizing resource utilization.

Algorithm 2: Data Cleaning Algorithm

Input: *dataset*
Output: *Clean_dataset*
1 $finite\_dataset \leftarrow$ ReplaceInfiniteValues($dataset$)
2 $dataset\_withoutNAN \leftarrow$ DropNA($finite\_dataset$)
3 $dataset\_withoutDuplicated \leftarrow$ DropDuplicates($dataset\_withoutNAN$)
4 $dataset\_correctDtype \leftarrow$ toNumeric($dataset\_withoutDuplicated$)
5 $Clean\_dataset \leftarrow$ Downsizing($dataset\_correctDtype$)
6 **return** $Clean\_dataset$

4.3.2.2 Data Transformation

A. Data Augmentation During the initial exploration of our dataset, we discovered a significant problem of class imbalance, whereby the number of instances of certain attack traffic was vastly outnumbered by other attack traffic. This imbalance could potentially lead to the machine learning algorithms being biased towards the majority classes and producing poor results in detecting minority classes. To overcome this issue, we decided to use a re-sampling method called Synthetic Minority Oversampling Technique (SMOTE) algorithm which is created in 2002 by Nitesh V. Chawla and al in [63]. This technique involves generating synthetic samples of the minority class to increase its representation in the dataset and balance it with the majority class. By using SMOTE, we can improve the effectiveness of our machine learning models in detecting attacks and reduce the risk of miss-classification.

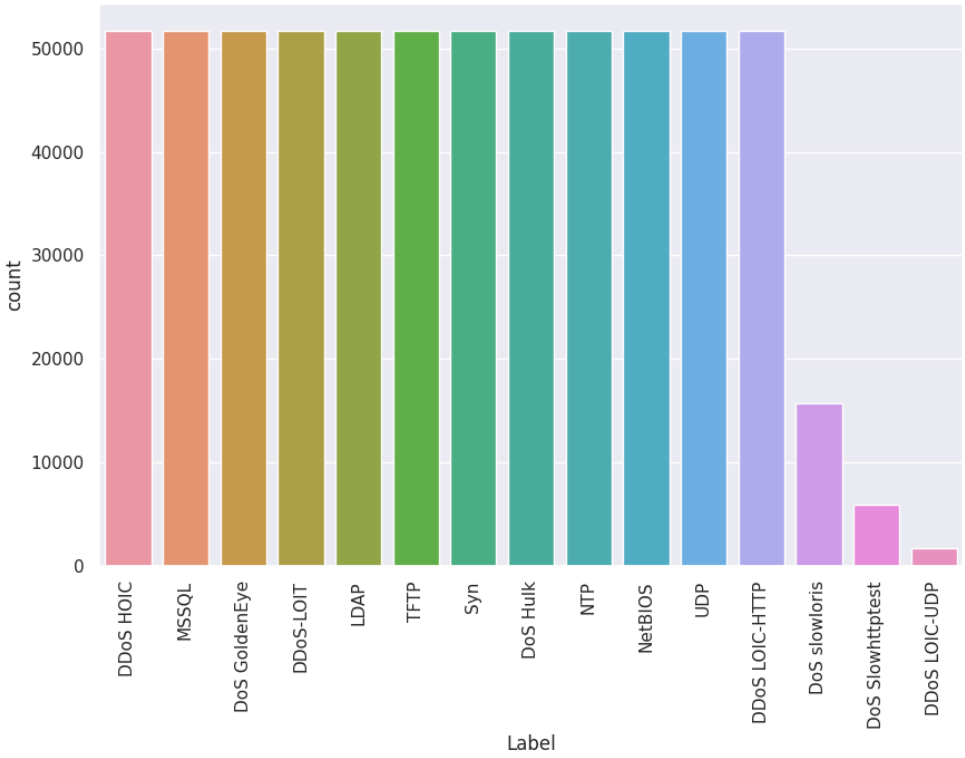


FIGURE 4.7 – Classes in the dataset before augmentation Process

the effect of our data augmentation process on the attack dataset is shown in the following figure :

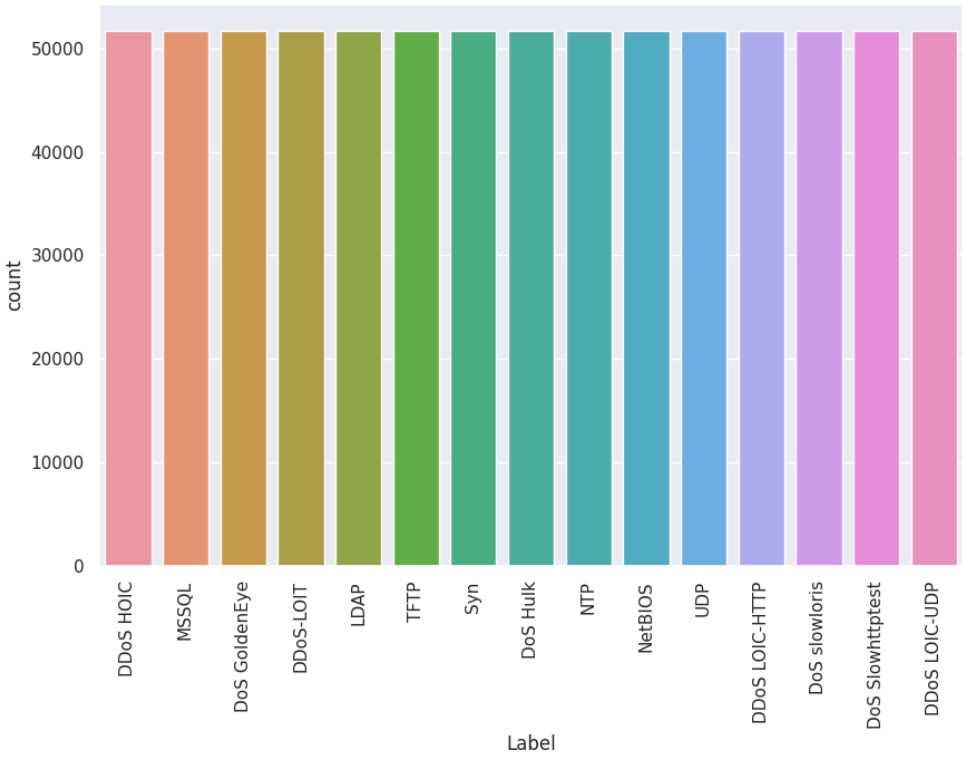


FIGURE 4.8 – Classes in the dataset after augmentation Process

B. Data Normalization Data normalization is a crucial step in our data preprocessing phase, aimed to reduce the processing power required for handling the dataset and optimizing the training process. To achieve this, we employ various statistical and mathematical methods. Specifically, we utilize the Z-score method, also known as Standard Scaler, to normalize the data. This technique standardizes the values by subtracting the mean and dividing by the standard deviation, resulting in a dataset with a mean of zero and a standard deviation of one. Normalizing the data using this method ensures that all features have a comparable scale, facilitating the training of machine learning models and improving their performance.

Algorithm 3: Data Transformation Algorithm

Input: *dataset*
Output: *normalized_dataset*
Function *NormalizeData(dataset)*
    `// Import StandardScaler from SKlearn`
    from SKlearn import StandardScaler
    `// Apply StandardScaler to the dataset`
    dataset ← `StandardScaler(X)`
    return *dataset*
Function *AugmentData(dataset)*
    `// Import SMOTE from SKlearn`
    from SKlearn import SMOTE
    `// Apply SMOTE to the dataset`
    `X, y` ← `SMOTE(`*dataset*`)`
    `// Concatenate X and y`
    augmented_dataset ← `Concat(X, y)`
    return *augmented_dataset*
Result: *augmented_dataset* ← AugmentData(*dataset*)
Result: *normalized_dataset* ← NormalizeData(*augmented_dataset*)
return *normalized_dataset*

4.4 Anomaly Detection Using Autoencoder-based Model

The constantly evolving nature of cyber attacks presents a significant challenge to intrusion detection systems that rely on machine learning. To address the issue of false positives, we propose a novel approach that separates the intrusion detection process from the classification process. In other words, we can isolate and identify anomalous traffic patterns that may indicate potential attacks. As part of the intrusion detection phase, we have opted to utilize the Autoencoder algorithm to detect any anomalies within the network. The Autoencoder is an unsupervised deep-learning algorithm that consists of an input layer, hidden layers, and an output layer. It is characterized by two main components encoder and decoder, the encoder and the decoder, which are connected in the middle by a hidden layer called the bottleneck layer.
as shown in the following figure :

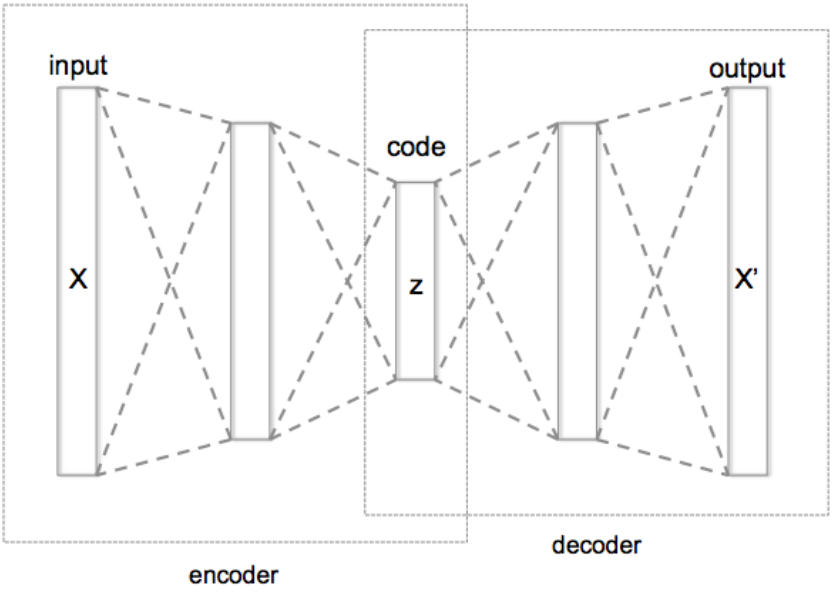


FIGURE 4.9 – illustration of Autoencoder algorithm

4.4.1 Encoder Phase

During the encoder phase, we apply our proposed approach by feeding the normal traffic dataset into the model's input layer, which then transforms it into a lower-dimensional latent representation, also known as a bottleneck layer. The Encoder applies a series of mathematical transformations from the input data to the last hidden layer to transform it in each layer into a more compact form. The encoder consists of several layers so, In our model, we have chosen to use 4 hidden layers in the encoding phase with each layer having a different number of nodes The exact number of nodes in each layer has been determined through experimentation and tuning to ensure optimal performance of the encoder. Here is the number of nodes in each layer :

- Takes the bottleneck layer as an input layer (8 nodes)
- 18 nodes in the first layer.
- 38 nodes in the second layer.
- 58 nodes in the third layer.
- Output layer that has the number of nodes the same as the number of attributes in our dataset, so 78 nodes.

Algorithm 4: Encoder Phase Algorithm

Input: *thenumberofattributesinourdataset*
Output: *bottelneck_data*
Function *Encoder(dataset)*
    data_58 ← hidden_layer1(bottleneck_data)
    data_38 ← hidden_layer2(data_18)
    data_18 ← hidden_layer3(data_38)
    bottelneck_data ← hidden_layer4(data_18)
    return *bottelneck_data*
return *bottelneck_data*

4.4.2 Decoder Phase

In the Autoencoder architecture, after the encoder phase. We move on to the decoder phase. , we pass the encoder output as the input layer and transform it back into the original input dimensions, The decoder output should be as close as possible to the original input data. It typically has a symmetrical structure to the encoder, with each layer corresponding to a layer in the encoder .thus, in our proposed decoder model, we have four hidden layer which are :

- Takes the bottleneck layer as an input layer (8 nodes)
- 18 nodes in the first layer.
- 38 nodes in the second layer.
- 58 nodes in the Third layer.
- 78 nodes in the Output layer.

Algorithm 5: Decoder Phase Algorthim

Input: *bottleneck_data*
Output: *dataset_copy*
Function *Decoder(bottleneck_data)* **:**
    data_18 ← hidden_layer1(*bottleneck_data*)
    data_38 ← hidden_layer2(*data_18*)
    data_58 ← hidden_layer3(*data_38*)
    dataset_copy ← hidden_layer4(*data_58*)
    return *dataset_copy*
return *dataset_copy*

4.5 Intrusion Detection Classification-based Model

One of the key challenges in intrusion detection is the classification of attacks, which involves identifying the type and severity of an intrusion in real-time. To address this challenge, we have used the concept of ensemble learning in our proposed approach for handling anomaly traffic. Ensemble learning is a machine learning technique that combines multiple algorithms to improve the overall performance of a model. We have implemented several widely used classification algorithms in our proposed approach, such as Random Forest, AdaBoost, k-nearest neighbors Light Gradient Boosting Machine, Catboost, and XGBoost. To select the most efficient ones for intrusion classification. By combining the strengths of these algorithms, we aim to create a more robust and accurate intrusion detection system. the following figure illustrates our approach for attack classification using six of the most powerful machine learning classifiers :

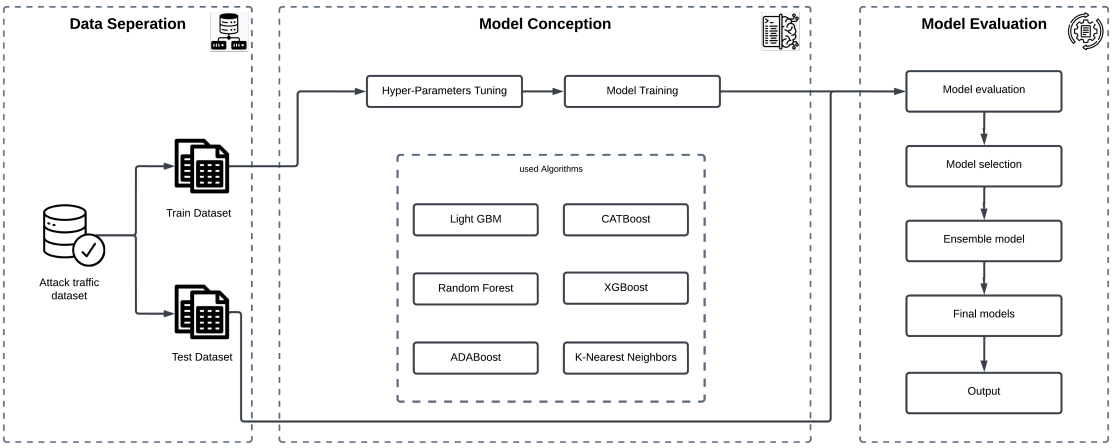


FIGURE 4.10 – illustration of the Classification Approach

4.5.1 Classifiers

In our approach, the objective was to perform the classification of different DDoS/DoS attacks. To achieve this, we carefully selected six classifiers based on their effectiveness in handling such attacks. These classifiers were chosen for their proven performance in similar scenarios and their ability to classify the types of attacks encountered in our dataset accurately. The following algorithms were utilized :

4.5.1.1 XGBoost

This classifier will process the characteristics in several compressed blocks. allowing them to be sorted and processed in parallel. We detailed this algorithm in the section 1.3.2.1 part (L) of the first chapter.

```
# import Classifier
from xgboost import XGBClassifier
#initiate the Classifier
Model=XGBClassifier(random_state=42, n_estimators=100, n_jobs=-1)
#training the model
Model.fit(X_train, y_train)
```

FIGURE 4.11 – XGboost Source Code

4.5.1.2 AdaBoost

Is one of the classification algorithms that turn data stumps into weak learners to create a robust powerful model. We detailed this algorithm in the section 1.3.2.1 part (J) of the first chapter.

```
# import Classifier
from sklearn.ensemble import AdaBoostClassifier
#initiate the Classifier
Model=AdaBoostClassifier(random_state=42, n_estimators=50, learning_rate=0.1)
#training the model
Model.fit(X_train, y_train)
```

FIGURE 4.12 – AdaBoost Source Code

4.5.1.3 Light Gradient Boosting Machine

Is an advanced version of GBM (Gradient Boosting Machine) that implements a vertical development approach based on leaf nodes, unlike traditional algorithms that use a horizontal level-wise approach. we detailed this algorithm in the section 1.3.2.1 part (M) of the first chapter.

```
# import Classifier
from lightgbm import LGBMClassifier
#initiate the Classifier
Model=LGBMClassifier(random_state=42, n_estimators=100, n_jobs=-1)
#training the model
Model.fit(X_train, y_train)
```

FIGURE 4.13 – Light Gradient Boosting Machine Source Code

4.5.1.4 Random Forest

One of the supervised classification algorithms, based on decision trees. We detailed this algorithm in the section 1.3.2.1 part (I) of the first chapter.

```
# import Classifier
from sklearn.ensemble import RandomForestClassifier
#initiate the Classifier
Model=RandomForestClassifier(random_state=42, n_estimators=100, n_jobs=-1)
#training the model
Model.fit(X_train, y_train)
```

FIGURE 4.14 – Random Forest Source Code

4.5.1.5 K-nearest neighbors

The KNN algorithm is a simple and effective approach for classification and regression tasks, especially when dealing with small datasets or non-linear relationships between the features and the target variable. We detailed this algorithm in the section 1.3.2.1 part (D) of the first chapter.

```
# import Classifier
from sklearn.neighbors import KNeighborsClassifier
#initiate the Classifier
Model=KNeighborsClassifier(n_neighbors=3, weights='uniform',
                           algorithm='ball_tree')
#training the model
Model.fit(X_train, y_train)
```

FIGURE 4.15 – k-nearest neighbors Source Code

4.5.1.6 CatBoost

This algorithm offers a distinct advantage by effectively handling categorical variables present in the dataset. More details regarding that are mentioned in the section 1.3.2.1 part (N) of the first chapter.

```
# import Classifier
from catboost import CatBoostClassifier
#initiate the Classifier
Model=CatBoostClassifier(random_state=42, iterations=100,
                         thread_count=multiprocessing.cpu_count())
#training the model
Model.fit(X_train, y_train)
```

FIGURE 4.16 – CatBoost source Code

4.5.2 Hyperparameters Tuning

Hyperparameter optimization plays a pivotal role in machine learning classification as it involves the hunt for the most effective combination of hyperparameter values that dictate the performance of a machine learning algorithm. These hyperparameters, set prior to training the model, can greatly influence various facets of the algorithm's behavior and consequently its efficiency in classifying data.

In the context of DoS/DDoS attack classification in our study, the tuning of hyperparameters for the chosen models is of paramount importance. The performance of these models in terms of precision, recall, and accuracy hinges largely on the optimal configuration of their hyperparameters. An inappropriate value can lead to poor results, such as overfitting, underfitting, or suboptimal classification outcomes.

To enhance the effectiveness of our machine learning models, we have employed the "Randomized Grid Search" methodology. This approach enables us to meticulously navigate and assess an assortment of parameter configurations in a more time-efficient manner than an exhaustive search, thereby facilitating the identification of the most advantageous hyperparameter combination.

The algorithm we implemented in our research for hyperparameter optimization is in the table below :

TABLE 4.1 – Algorithms parameters

Model	parameters
CatBoost	$random\_state = 42, iterations = 100, depth = 6, learning\_rate = 0.1, l2\_leaf\_reg = 1, thread\_count = multiprocessing.cpu\_count()$
Random Forest	$random\_state = 42, n\_estimators = 100, criterion = gini, max\_depth = None, min\_samples\_split = 2, min\_samples\_leaf = 1, max\_features = sqrt, n\_jobs = -1$
XGBoost	$random\_state = 42, n\_estimators = 100, max\_depth = 6, learning\_rate = 0.1, subsample = 0.8, colsample\_bytree = 0.8, n\_jobs = -1$
LGBM	$random\_state = 42, n\_estimators = 100, max\_depth = 6, learning\_rate = 0.1, subsample = 0.8, colsample\_bytree = 0.8, n\_jobs = -1$
K-Nearest-Neighbors	$n\_neighbors = 3, weights =' uniform', algorithm = ball\_tree$
ADABoost	$random\_state = 42, n\_estimators = 50, learning\_rate = 0.1, algorithm =' SAMME.R'$

4.5.3 Ensemble Learning

Ensemble learning is a powerful and flexible technique that can improve the accuracy, robustness, and generalization performance of the classifier in intrusion detection. By combining multiple models trained on different features or with different algorithms, the ensemble classifier can capture a wider range of attack patterns and provide a more reliable defense against various types of attacks Ensemble learning consists of several methods, including Bagging, Boosting, Majority Voting, Max Rule, and others.

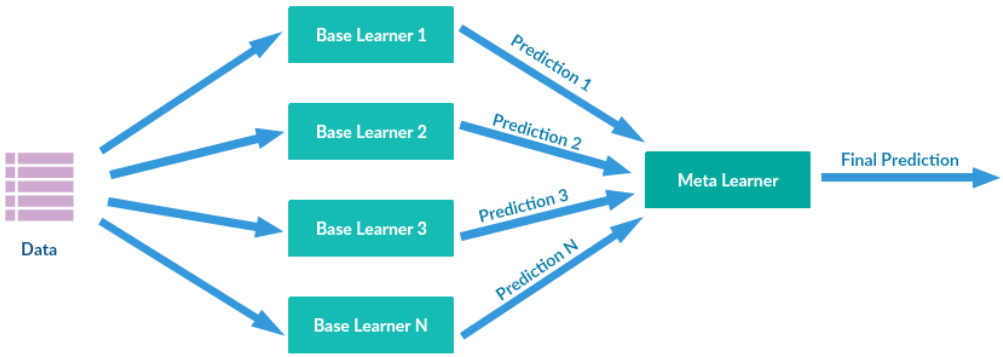


FIGURE 4.17 – illustration of the Ensemble learning [64]

For our proposed approach, we opted for the Majority Voting method, which is a popular technique in ensemble learning. It is one of the earliest and easiest ensemble schemes in the literature. The Majority Voting method combines predictions from multiple classifiers to make a final decision. It involves an odd number of classifiers that are trained on different subsets of the data. The predictions of each model are then combined to determine the most frequent prediction, which becomes the final decision of the

ensemble. Essentially, this method selects the class that receives the most votes from the pool of classifiers.

4.6 Conclusion

In this chapter, we have introduced our proposed approach, which uses deep learning and machine learning techniques to handle network traffic data. This approach involves different phases of data pre-processing, normal traffic handling, and attack traffic classification, each of which is carried out using various techniques and tools.
In the next chapter, we will dive into the implementation details of our proposed approach and present the results of various evaluation metrics. These tests will help us determine the effectiveness and viability of our approach and its ability to accurately detect intrusions in network traffic data.

CHAPITRE 5
IMPLEMENTATION

5.1 Introduction

In this final chapter, we present the implementation steps of our proposed approach in the field of intrusion detection. The chapter begins with a description of the development environment, including the tools and programming languages used. We then detail the testing procedures employed to evaluate the performance of the model. Finally, we present and analyze the results obtained from the tests to determine the effectiveness of the proposed model.

5.2 Environment and Tools

5.2.1 Libraries used

5.2.1.1 Python

Python is a popular high-level programming language that is widely used for general-purpose programming. It was created in the late 1980s by Guido van Rossum, and its design philosophy emphasizes code readability and ease of use. Python is known for its simplicity, clarity, and elegance, and it has a large and active community of developers who contribute to its growth and development. It is used for a wide range of applications, including web development, scientific computing, data analysis, artificial intelligence, and automation, among others. One of the main advantages of Python is its ease of learning and use, which makes it an ideal choice for beginners and professionals alike. Its syntax is straightforward and readable, and it offers a vast array of libraries and frameworks that simplify complex tasks, making it one of the most popular programming languages in the world.

FIGURE 5.1 – python
[65]

5.2.1.2 TensorFlow

TensorFlow is an open-source software library for machine learning and artificial intelligence developed by Google. It provides a platform for building and training machine learning models, especially deep neural networks, by allowing developers to define, manipulate and optimize mathematical expressions using multi-dimensional arrays called tensors. TensorFlow has gained popularity due to its ease of use, flexibility, and scalability, and has been widely used in a variety of industries, such as healthcare, finance, and retail. It is compatible with multiple programming languages, including Python, C++, and Java. It provides a wide range of tools and resources for developers, such as pre-built models, tutorials, and community support

FIGURE 5.2 – tensorFlow
[66]

5.2.1.3 Scikit-Learn

Scikit-Learn is a free and open-source machine learning library for Python programming language. It provides a set of tools for machine learning tasks, such as classification, regression, clustering, and dimensionality reduction, among others. Scikit-Learn is built on top of other popular Python libraries, such as NumPy, SciPy, and matplotlib, and is designed to be simple and easy to use. It provides a consistent interface for working with different machine-learning algorithms and also includes utilities for data preprocessing, model selection, and evaluation. Scikit-Learn is widely used in academia and industry and has become one of the most popular machine-learning libraries in the Python ecosystem. Its popularity is due to its simplicity, versatility, and performance, as well as its extensive documentation and active community support.

FIGURE 5.3 – Scikit-Learn
[67]

5.2.1.4 Keras

Keras is a high-level open-source neural network library written in Python. It provides a user-friendly API for building and training deep learning models with minimal code, making it accessible to both beginners and experts in machine learning. Keras is built on top of other popular machine learning libraries, such as TensorFlow and Theano, and allows for easy prototyping and experimentation with various neural network architectures. It supports a wide range of neural network models, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer models, among others

FIGURE 5.4 – Keras
[68]

5.2.1.5 Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it provides data structures and operations for manipulating numerical arrays and time series.

FIGURE 5.5 – pandas
[69]

5.2.1.6 Numpy

Numpy is a library for the Python programming language that adds support for large multidimensional arrays and matrices and a large collection of high-level mathematical functions to operate on these arrays.

FIGURE 5.6 – numpy
[70]

5.2.1.7 Matplotlib

Matplotlib is a plotting library for Python. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. It also provides a procedural interface for non-interactive plotting purposes. Matplotlib allows the creation of a wide variety of static, animated, and interactive visualizations in Python. It can be used to create publication-quality charts, graphs, and other types of visualizations for data analysis and scientific research.

FIGURE 5.7 – matplotlib
[71]

5.2.1.8 Seaborn

Seaborn is a Python data visualization library based on Matplotlib. It provides a high-level interface for creating informative and attractive statistical graphics. Seaborn is particularly useful for visualizing complex data sets such as those commonly encountered in statistical analyses. It provides a variety of built-in plotting functions that make it easy to create many types of visualizations, such as scatter plots, line plots, bar plots, heat maps, and more. Seaborn also includes several advanced features such as built-in color palettes, statistical functions, and support for visualizing categorical data

FIGURE 5.8 – seaborn
[72]

5.2.2 Tools

5.2.2.1 Jupyter Notebook

is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and narrative text. Jupyter Notebook is widely used in the field of AI and machine learning for exploratory data analysis, prototyping, and sharing research results.

FIGURE 5.9 – Jupyter Notebook
[73]

5.2.2.2 VS Code

Visual Studio Code is an open-source code editor developed by Microsoft that is both lightweight and customizable. It can be run on desktop computers and is compatible with Windows, macOS, and Linux. It has built-in support for JavaScript, TypeScript, and Node.js, as well as an extensive range of extensions for other languages and runtimes, including C++, C#, Java, Python, PHP, Go, and .NET. Despite its compact size, Visual Studio Code is a powerful editor with a wealth of features.

FIGURE 5.10 – VS Code
[74]

5.2.3 Hardware

The hardware configuration used in our implementation is as follows :

Table 5.1 – Table of Hardware Specs

Specs	Asus Vivo book	HP-EliteBook G3 820
Processor (CPU)	Intel(R) Core (TM) i7-8500U	Intel(R) Core (TM) i5-6300U
Graphic-Card(GPU)	Intel® UHD Graphics 620	Intel® UHD Graphics 520
Memory (RAM)	12 GB	16.0 GB
Operation System	Windows 10 Pro 64-bit	Windows 10 Pro 64-bit
Storage	1 TB HDD	1TB HDD + 250GB SSD

5.3 Performance Analyse

In order to evaluate the performance of deep learning models in a classification system, different parameters known as "evaluation metrics" will be utilized. These metrics will measure the accuracy of classifications, by categorizing elements into two groups : positive or negative.
There are four possible outcomes of these classifications :

1. True Positives (TP) : This term refers to instances where a positive element is correctly classified as positive.
2. False Positives (FP) : A false positive occurs when an element is mistakenly classified as positive when it is not.
3. True Negatives (TN) : This term refers to cases where a negative element is correctly classified as negative.
4. False Negatives (FN) : A false negative occurs when an element is incorrectly classified as negative when it is positive.

Next, we will introduce the commonly utilized evaluation metrics.

- **Accuracy** The basic performance metric is accuracy, which is the proportion of correctly predicted observations to all observations. Accuracy is a useful evaluation measure, only when the datasets are uniform, and the false positive and false negative values are almost comparable. Accuracy tells how correctly the classifier is predicting the data points, as shown in Equation 5.1.

$$Accuracy = \frac{TP}{TP + TN + FP + FN} \tag{5.1}$$

- **Precision** is defined as the proportion of accurately predicted positive observations to all expected positive observations. High precision is associated with a low false-positive rate. Precision gives a probability of how correctly the classifier is predicting the positive class. Precision is calculated with Equation 5.2.

$$Precision = \frac{TP}{TP + FP} \tag{5.2}$$

- **The recall** is defined as the ratio of accurately predicted positive observations to all observations in the actual class. Precision gives a probability of how correctly the classifier is predicting the actual positive class, as shown in Equation 5.3.

$$Recall = \frac{TP}{TP + FN} \tag{5.3}$$

- **F1 Score** is a normalized average of precision and recall. As a result, this score includes both false positives and false negatives. Although the F1 score is simpler than accuracy, it is more useful, especially if the class distribution is irregular. F1 score is a harmonic mean of precision and recall, as shown in Equation 5.4.

$$F1 - Score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{5.4}$$

- **Mean square error function** The MSE calculates the average squared difference between the predicted values and the actual values. It is often used to assess the accuracy and goodness of fit of regression models or prediction algorithms. The equation for the mean square error (MSE) is typically represented as follows 5.5.

$$MSE = \frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2 \tag{5.5}$$

- **Mean Absolute Error (MAE) function :**
  The MAE calculates the average absolute difference between the predicted values and the actual values. It provides a measure of the average magnitude of the errors without considering their direction. The equation for the mean absolute error (MAE) is typically represented as follows :

$$MAE = \frac{1}{n}\sum_{i=0}^{n}|y_i - \hat{y}_i| \tag{5.6}$$

- **Logcosh Function :**
  The logcosh function is a smooth approximation of the logarithm of the hyperbolic cosine. It is often used as a loss function in regression tasks. The equation for the logcosh loss function is typically represented as follows :

$$\text{logcosh} = \frac{1}{n}\sum_{i=0}^{n}\left(\log\left(\cosh(y_i - \hat{y}_i)\right)\right) \tag{5.7}$$

- **Mean Squared Logarithmic Error (MSLE) function :**
  The MSLE calculates the average logarithmic squared difference between the predicted values and the actual values. It is commonly used when the predicted values and the actual values span multiple orders of magnitude. The equation for the mean squared logarithmic error (MSLE) is typically represented as follows :

$$MSLE = \frac{1}{n}\sum_{i=0}^{n}\left(\log(1 + y_i) - \log(1 + \hat{y}_i)\right)^2 \tag{5.8}$$

5.4 Results and Discussion

In order to detail and explain the obtained results for the implemented approaches, we illustrate the evaluation metrics to analyze the impact of the adopted approach .

5.4.1 Result of Data-Cleaning

The cornerstone of machine learning is undoubtedly the dataset because its quality directly influences the performance of the model. In order to be based on a consistent dataset model for the detection of DoS/DDoS attacks, we carried out intensive research work. This research allowed us to detect datasets grouping up-to-date attacks. We have

studied and merged three distinct types of datasets in order to obtain a complete and voluminous dataset representing a good starting point for the model we have adopted. The detail of the new dataset is represented in the next table :

Table 5.2 – Table of Data Selection Results

Dataset	Total Data	Selected Data	Total attributes	Selected attribute
CIC-IDS-2017	3119345	2652833	85	79
CIC-IDS-2018	16233002	14480672	80	79
Ddos-2019	76540368	4420646	88	79
New-Dataset	21554151	8989375	79	79

5.4.2 Result And Discussion of Auto-Encoder

5.4.2.1 Result of Auto-Encoder

In our study for intrusion detection, we have employed an Autoencoder-based approach, as discussed in section 1.4.4.4, with the primary aim to harness the power of unsupervised learning for intrusion detection. By training the model on the reconstruction of benign normal traffic, we enable the model to discern key features indicative of normal network behavior.

This model facilitates the distinction between normal and abnormal network behavior by leveraging reconstruction loss error. In our research, we adopt the Mean Squared Error (MSE) as the loss function, allowing us to assess the autoencoder's proficiency in learning normal traffic patterns.

In the course of our investigation, we divided the training dataset into two subsets - one for training and another for validation. This strategy was adopted to monitor the progress of the model during the training phase and to guard against over-fitting the training data. Upon completion of the training process, we tested the model on a separate test dataset. This allowed us to evaluate the model's ability to generalize to unseen data. Finally, we subjected our model to a sample from the attack dataset to gauge its performance in detecting anomalous traffic patterns. During the training phase of our methodology, we meticulously documented the changes in the loss function throughout the process. This evolution of the loss function was then visualized, offering us a more intuitive method to analyze the data. The figure below illustrates the trajectory of the loss function throughout the training phase.

Table 5.3 – Table of Autoencoder Results

Loss Function	One Layer Normal Traffic	Three Layer Normal Traffic	One Layer Attack Traffic	Three Layer Attack Traffic
Mean Squared Error(MSE)	0.014188	0.009050	0.45328498	0.44900575
Mean Absolute Error(MAE)	0.030910	0.027519	0.2993047	0.32261175
Logcosh	0.003284	0.001837	0.13651925	0.13506278
Mean Squared Logarithmic Error (MSLE)	0.000900	0.008773	0.055928152	0.049676776

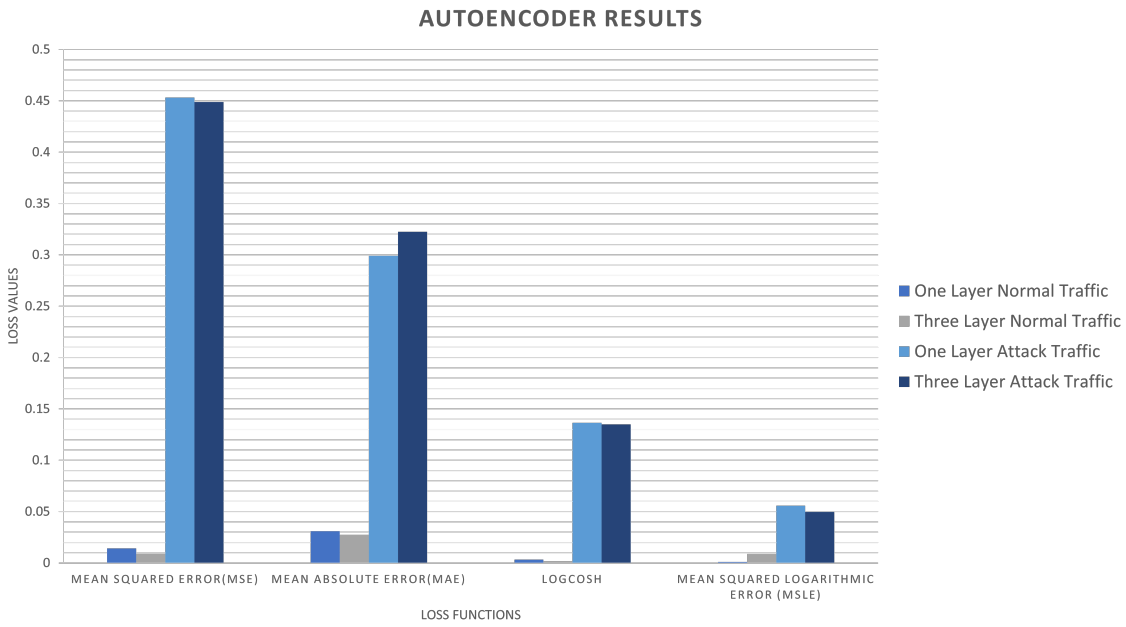


FIGURE 5.11 – models performance on test data

5.4.2.2 Discussion of Autoencoder Model

In our work, we experimented with two different architectures of the Autoencoder model, each implementing four different loss functions : logcosh, mean absolute error, mean squared error, and mean squared logarithmic error. This experimentation served to evaluate the optimal model architecture and loss function for efficient intrusion detection.

Upon examining the results, we can observe the variability in performance for different configurations. For both one-layer and three-layer architectures, the mean squared logarithmic error loss function exhibited the lowest loss on the test dataset, implying the highest efficiency in reconstructing the normal network traffic. This indicates the suitability of this loss function for this specific task, as it excels in handling the differences between the logarithms of predictions and actual values, which can be particularly useful when dealing with datasets with a wide value range, as is often the case in network traffic data.

When comparing the one-layer and three-layer architectures, we observe that both architectures perform comparably. We noticed that the three-layer architecture is performing marginally better for the mean squared error and mean squared logarithmic error loss functions. This ensures that increasing the complexity of the model, in terms of the number of layers, may slightly enhance the model's learning capacity, capturing more subtle patterns within the normal network traffic data.

Moving to the results on the attack traffic, we note a significant increase in the loss across all models. This is expected since the Autoencoder models are trained to reconstruct normal traffic. Therefore they should exhibit higher loss (lower performance) when dealing with attack traffic, which they are not trained on. This reinforces the effectiveness of our models in differentiating between normal and attack traffic. It's worth noting that the three-layer architecture with logcosh loss metric performed marginally better on attack traffic than the corresponding one-layer architecture. This could be interpreted as the three-layer model's ability to capture more complex patterns, enabling it to deal slightly better with unfamiliar attack traffic.

In conclusion, our analysis demonstrates that a three-layer Autoencoder model with a mean squared error loss metric may offer the most effective performance in detecting intrusion. These results underscore the value of comprehensive experimentation and evaluation in the development of an efficient intrusion detection system.

5.4.3 Discussion and Result of Classifiers

Our proposed methodology entails employing six distinct classifiers to identify the specific types of DoS/DDoS attacks. After this classification step, we advocate the use of ensemble learning techniques to further improve and enhance the performance of the system.

5.4.3.1 Classifiers Results

The following table illustrates the results yielded on the test dataset for each of our employed models :

TABLE 5.4 – Table of Classification Results

Model	Accuracy	F1 Score	Precision	Recall
CAT	0.98740143	0.98744583	0.98771098	0.98740143
RF	0.98913583	0.98914655	0.98928205	0.98913583
XGB	0.9900256	0.99005557	0.99031122	0.9900256
LGBM	0.99008363	0.99011102	0.99035626	0.99008363
KNN	0.98185007	0.98187526	0.98205492	0.98185007
ADA	0.3979832	0.34195691	0.41541662	0.3979832

These results are further represented in the form of a Bar graph to gain better information insights and clarify the comparison between the models to select the best-performing models

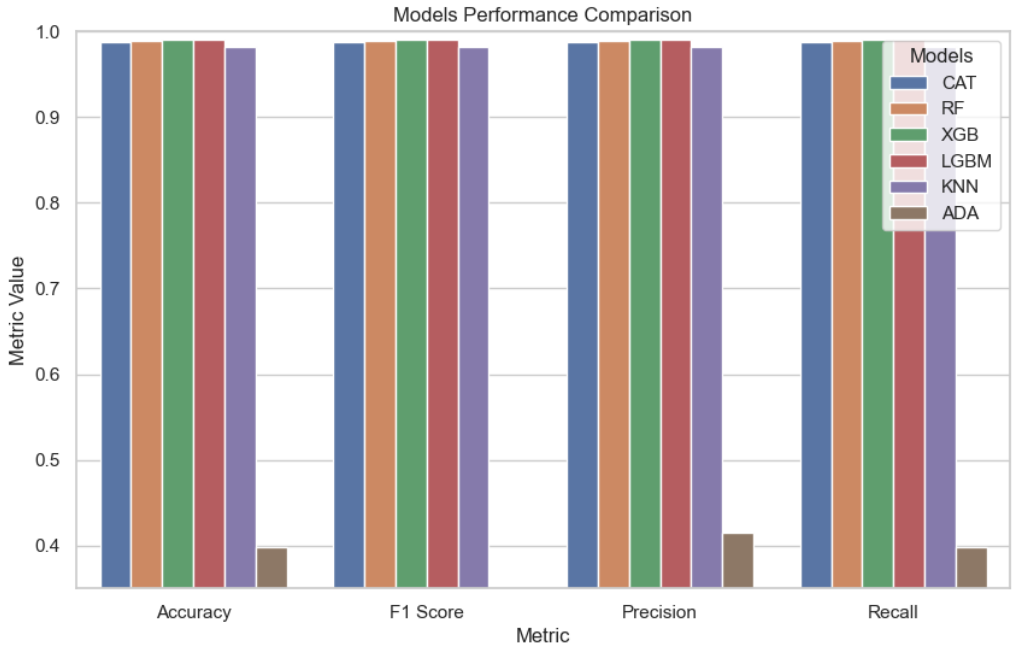


FIGURE 5.12 – Classifiers Evaluation Results

In addition to the evaluation metrics, we graphically depict the network traffic patterns of the actual and predicted test data for each classifier :

1. Random Forest

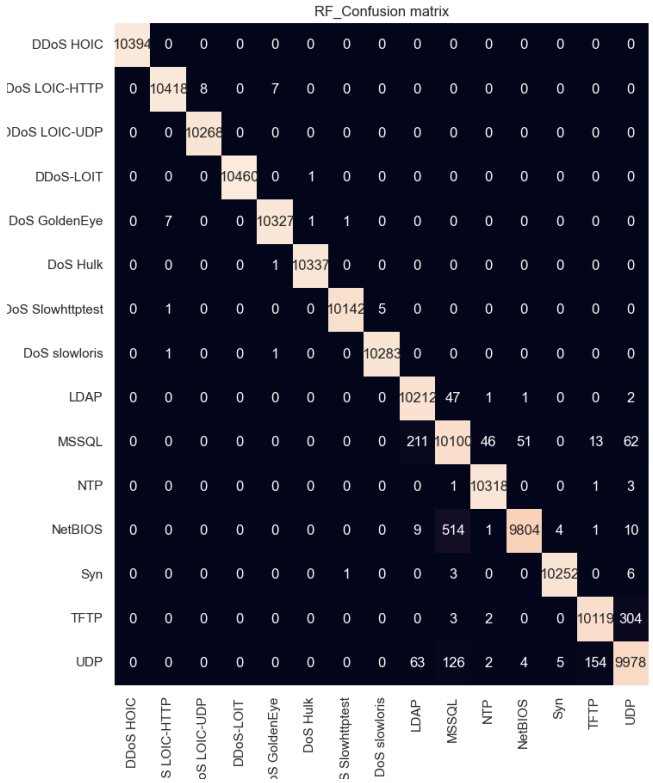


FIGURE 5.13 – Random Forest Confusion Matrix

2. LightGBM

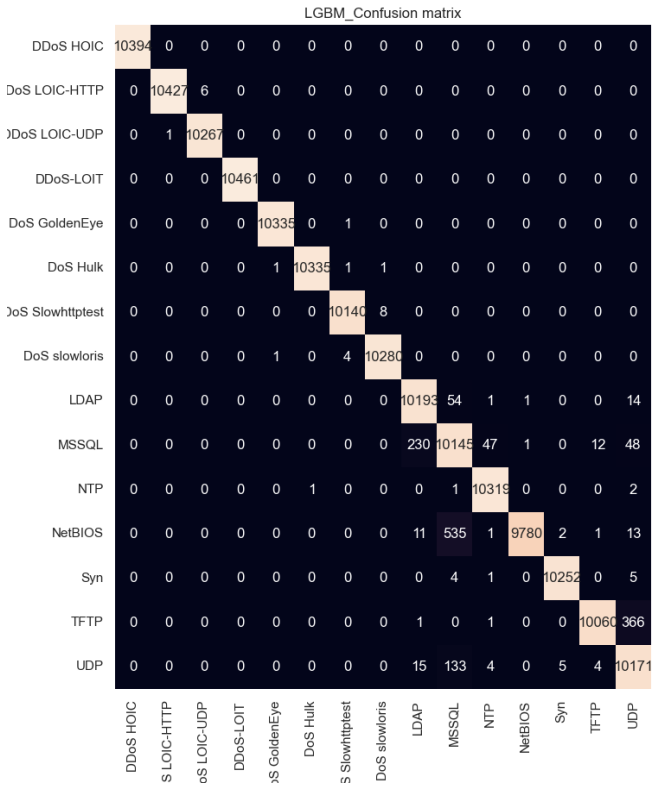


FIGURE 5.14 – Light GBM Confusion Matrix

3. XGBoost

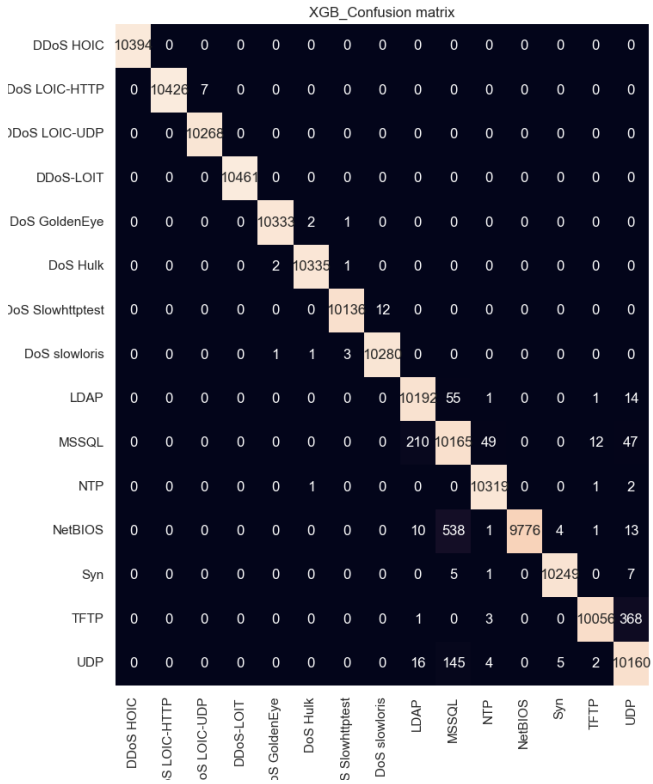


FIGURE 5.15 – XGBoost Confusion Matrix

4. Categorical Boost

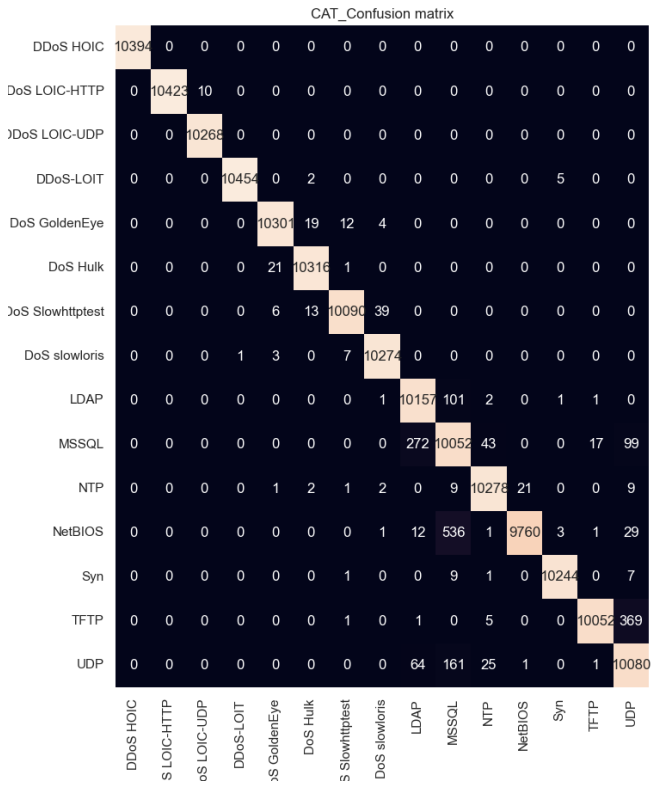


FIGURE 5.16 – Categorical Boost Confusion Matrix

5. Adaptive Boost

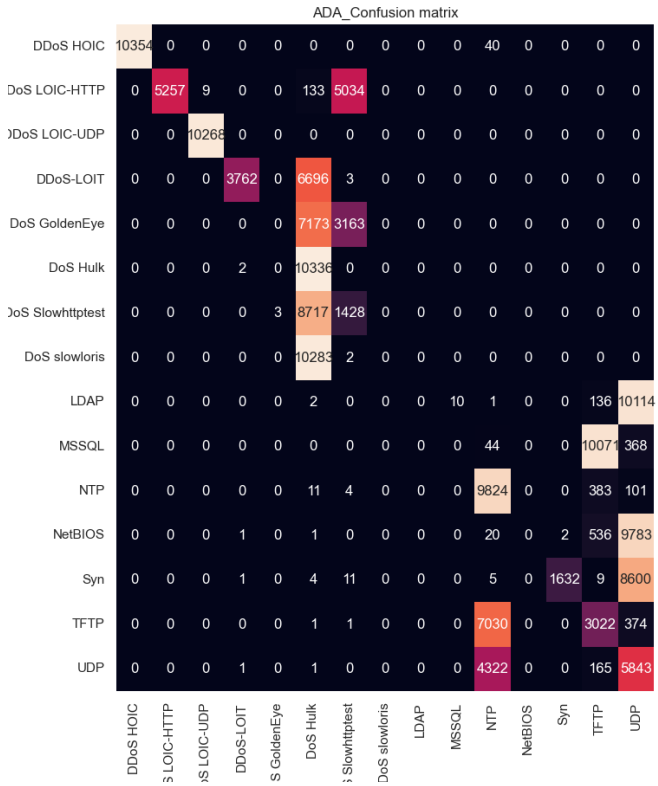


FIGURE 5.17 – Adaptive Boost Confusion Matrix

6. K-nearest Neighbor

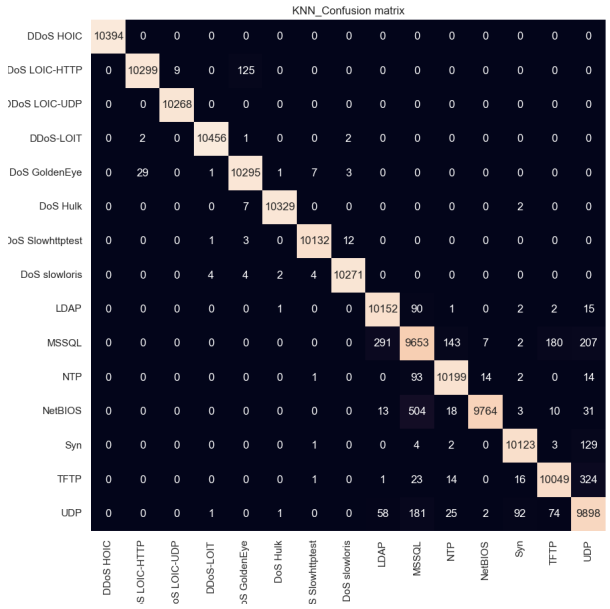


FIGURE 5.18 – K-nearest neighbor Confusion Matrix

5.4.3.2 Discussion Of Classifiers Results

After thoroughly examining the results presented in the table and analyzing the corresponding figures, it became evident that the employed classifiers achieved outstanding performance in accurately classifying the different types of attacks, with one exception observed in the case of the AdaBoost classifier. Upon further investigation, it was discovered that the AdaBoost model encountered challenges in distinguishing between similar attack classes. This issue can be attributed to the classifier's sensitivity to outliers and its reliance on weak classifiers. Consequently, the AdaBoost model exhibited relatively lower accuracy and less robust performance compared to the other classifiers. In contrast, the XGBoost and LightGBM classifiers, both of which are gradient boost-based models, demonstrated remarkable performance with an accuracy sur- passing 0.99. These models exhibited exceptional capability in effectively identifying various types of DDoS/DoS attacks, exhibiting minimal error rates. Their superior accuracy highlights the potential of gradient boost-based algorithms in intrusion detection tasks. Furthermore, the K-nearest neighbor, Random Forest, and CatBoost models also showcased commendable performance, achieving accuracy levels exceeding 0.98. These models displayed a robust ability to classify different types of attacks accurately, making them reliable choices for intrusion detection applications. The high accuracy achieved by the previously mentioned classifiers underscores the effectiveness of the employed machine learning techniques in detecting and categorizing intrusions accurately. These results validate the suitability of the selected classifiers for the intrusion detection task at hand. It is important to note that the superior performance of certain classifiers does not diminish the overall significance of the study. Instead, it highlights the potential for constructing an ensemble classifier that combines the strengths of these models to further improve detection accuracy and enhance the system's resilience against adversarial attacks.

5.4.4 Discussion and Result Of Ensemble Learning

5.4.4.1 Result Of Ensemble Learning

Our detailed evaluation of individual classifiers and ensemble models for DoS/DDoS attack classification offers significant insights into the performance and efficiency of the applied models. In our approach to ensemble learning, we used majority voting techniques with soft and hard voting methods the results of these methods are measured using accuracy, recall, precision, and f1 score as demonstrated in both the table and the figure below :

Table 5.5 – Table of Ensemble learning Results

Model	Accuracy	F1 Score	Precision	Recall
Hard Voting	0.98969677	0.98972163	0.98995049	0.98969677
Soft Voting	0.98974835	0.98977889	0.9900307	0.98974835

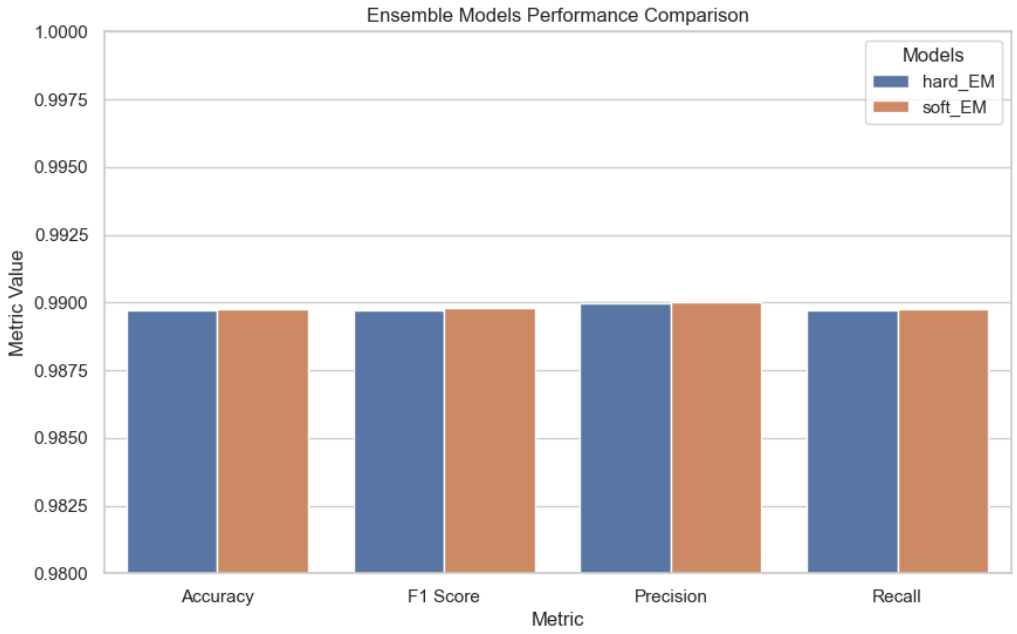


FIGURE 5.19 – Voting Ensemble Evaluation Results

To gain further insight into the performance of each model we visualized the prediction confusion matrix as shown in the following figures :

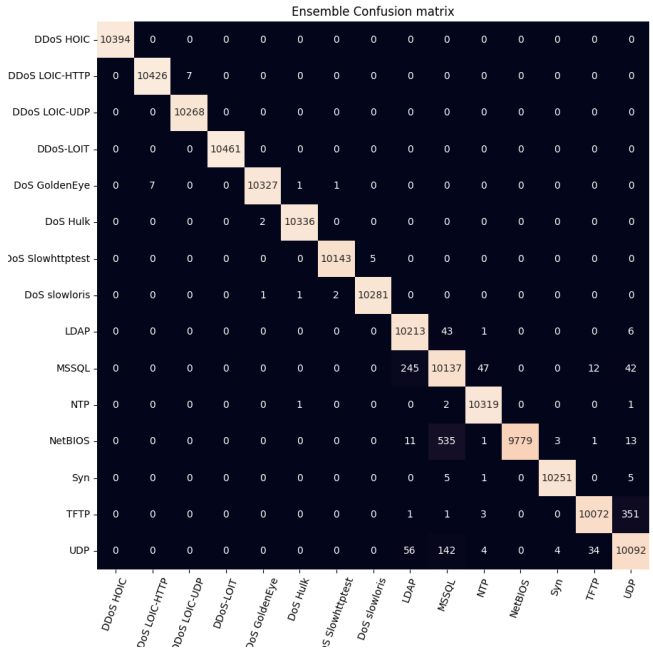


FIGURE 5.20 – Hard Voting Ensemble Confusion Matrix

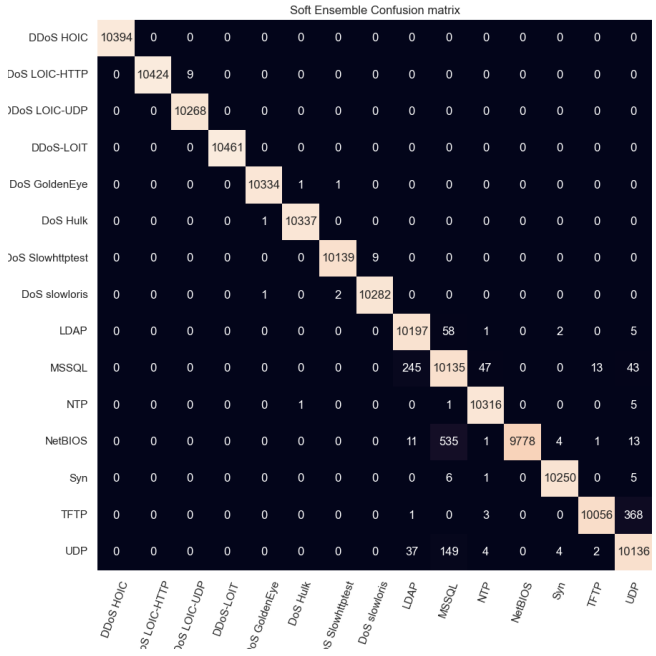


FIGURE 5.21 – Soft Voting Ensemble Confusion Matrix

To evaluate the efficiency of our ensemble methods we compared the results of the ensemble models with the individual classifiers the result is mentioned in the following figure :

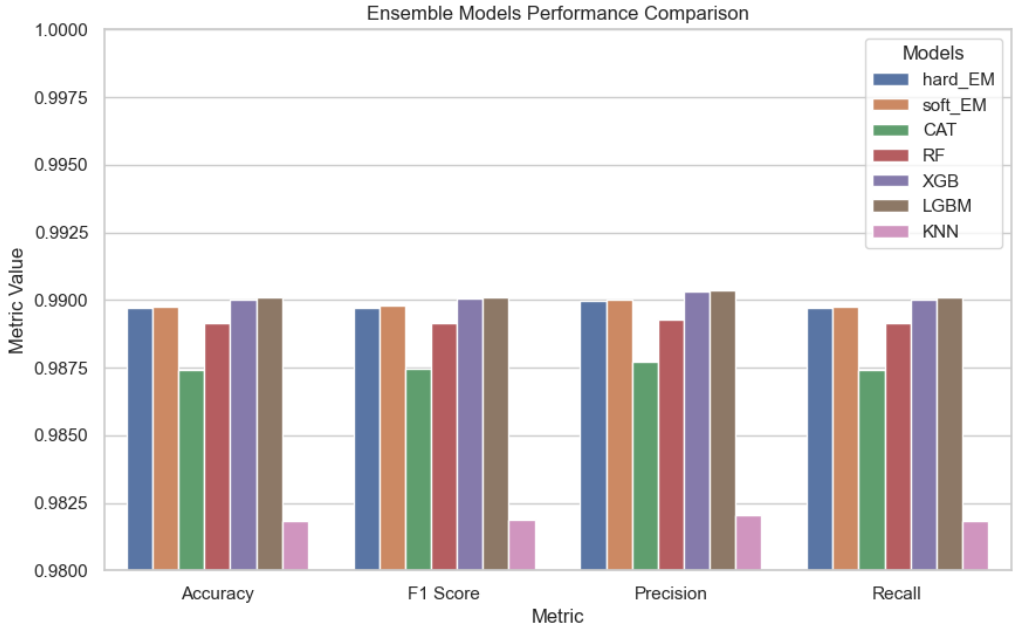


FIGURE 5.22 – Model Performance Comparison

5.4.4.2 Discussion Of Ensemble Learning

Our detailed evaluation of individual classifiers and ensemble models for DoS/DDoS attack classification offers significant insights into the performance and efficiency of the applied models. For the ensemble learning approach, we implemented both hard and soft voting classifiers. Hard voting classifiers predict the final class label as the class that gets the most votes from individual classifiers. On the other hand, soft-voting classifiers predict the final class label based on the sum of predicted probabilities for individual classifiers. The performance of the soft voting classifier appears superior to the hard voting one.

The Accuracy, F1 Score, Precision, and Recall of the soft voting classifier are all higher than those of the hard voting classifier. Specifically, the soft voting classifier achieves an Accuracy and F1 Score close to 99, indicating that it can classify DoS/DDoS attacks with a high degree of precision and recall. The hard voting classifier, while still performing relatively well, exhibits slightly lower performance metrics. These results highlight the importance of model selection and a set of strategies in the task of DoS/DDoS attack classification. The individual classifiers, particularly the XGB and LGBM models, deliver promising results, indicating their potential applicability in this task. Moreover, the ensemble approach, specifically the soft voting classifier, also demonstrates strong performance, ensuring its viability in improving classification accuracy and robustness in this context. Further studies and experiments may explore and optimize these models for even better performance and adaptability in real-world problems.

5.5 Conclusion

In this final chapter, we have presented in detail the evaluation and tests conducted on our approach. Initially, we introduced the development environment of our approach, and the results of the anomaly detection phase and the classification phase were thoroughly analyzed using performance measures such precision as Recall, Precision, and F1-score.

GENERAL CONCLUSION AND FUTURE WORK

Conclusion

To conclude, the ever-expanding internet and the increasing of cyber attacks, particularly Denial of Service (DoS) attacks, pose significant challenges to the security of network systems. Traditional intrusion detection methods have limitations in identifying and differentiating between normal network behavior and potential threats, leading to high rates of false positives and false negatives.

To overcome these limitations, We conducted bibliographic research to examine intrusion detection systems and Artificial Intelligence in order to understand their operating principles, their most prominent characteristics, etc. Subsequently, we developed two models, one for intrusion detection and the other for classifying types of attacks. furthermore, this research aims to develop a machine learning model that can accurately classify various types of cyber attacks, contributing to the robustness of the intrusion detection system. The objective is to expand the classification range to include a broader spectrum of DDoS/DoS attacks, ensuring the system is up-to-date and capable of responding to newer attack methods.

Overall, the successful achievement of these objectives will significantly enhance the reliability, efficacy, and resilience of intrusion detection systems in the face of evolving cyber threats. By leveraging machine learning and deep learning techniques, this research contributes to the field of intrusion detection and cyber security, making networks more secure and better prepared to counter the challenges posed by malicious actors on the internet.

In the end , The work on our project has allowed us to acquire new knowledge in the field of machine learning. It has also enabled us to learn the functioning of several machine learning algorithms.

Perspective

Although the initial goals have been achieved, there are still perspectives and possible improvements that can be made in the future. As part of our future plans, we aim to :

- Improve this work by incorporating other methods of deep learning. We will conduct a comparative study among these methods to evaluate their effectiveness in terms of results and performance.
- Develop and Integrate a dashboard into our system to facilitate alarm management. This dashboard will provide a centralized platform for users to monitor and manage alarms generated by the system.
- Enhance our system's security by incorporating prevention attack features. We will explore and implement techniques that can proactively identify and mitigate potential security threats. By strengthening the system's defenses against attacks, we aim to improve its resilience and protect against malicious activities.

  These perspectives and improvements reflect our commitment to continuous advancement

BIBLIOGRAPHIE

[1] Kdd in data mining assists data prep for machine learning | techtarget. www.techtarget.com/searchenterpriseai/feature/KDD-in-data-mining-assists-data-prep-for-machine-learning.

[2] Arthur L Samuel. Machine learning. *The Technology Review*, 62(1) :42–45, 1959.

[3] Hodhaifa Abdelghani BARAKA. *Étude sur les Méthodes d'Optimisation Utilisée dans l'Apprentissage Automatique*. PhD thesis, Directeur : Mr. RIMOUCHE Ali/Co-directeur : Mme. HANDOUZI Wahida, 2020.

[4] How does machine learning work - javatpoint. www.javatpoint.com/how-does-machine-learning-work.

[5] Julien Ah-Pine and Anne-Françoise Yao. Une approche par noyaux multiples pour l'apprentissage non-supervisé de représentation de données fonctionnelles dans des espaces de sobolev.

[6] Logistic regression in machine learning - javatpoint. www.javatpoint.com/logistic-regression-in-machine-learning.

[7] Machine learning polynomial regression - javatpoint. www.javatpoint.com/machine-learning-polynomial-regression.

[8] K-nearest neighbor(knn) algorithm for machine learning - javatpoint. www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning.

[9] Support vector machine (svm) algorithm - javatpoint. www.javatpoint.com/machine-learning-support-vector-machine-algorithm.

[10] Nilkanth Deshpande, Shilpa Gite, and Rajanikanth Aluvalu. A review of microscopic analysis of blood cells for disease detection with ai perspective. *PeerJ Computer Science*, 7 :e460, 04 2021.

[11] Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Citeseer, 1996.

[12] Introduction to random forest in machine learning | engineering education (enged) program | section. www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/.

[13] Tianqi Chen and Carlos Guestrin. Xgboost : A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[14] Zhuo Wang, Jintao Zhang, and Naveen Verma. Realizing low-energy classification systems by implementing matrix multiplication directly within an adc. *IEEE Transactions on Biomedical Circuits and Systems*, 9 :1–1, 12 2015.

[15] Xgboost - geeksforgeeks, NaN. www.geeksforgeeks.org/xgboost/.

[16] Alan Jeffares. Supervised vs unsupervised learning in 2 minutes, Sep 2020. towardsdatascience.com/supervised-vs-unsupervised-learning-in-2-minutes-72dad148f242.

[17] Li Deng, Dong Yu, et al. Deep learning : methods and applications. *Foundations and trends® in signal processing*, 7(3–4) :197–387, 2014.

[18] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1) :1–127, 2009.

[19] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning : A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8) :1798–1828, 2013.

[20] Jürgen Schmidhuber. Deep learning in neural networks : An overview. *Neural networks*, 61 :85–117, 2015.

[21] Vidyaesampally1998 Vidya. Artificial neural network v/s biological neural network, Feb 2021. vidyaesampally1998.medium.com/artificial-neural-network-v-s-biological-neural-network-a0862d12e9a8, journal=Medium.

[22] Andy Betts. Going deep with deep learning : Martech insights, action amp ; impact, Oct 2021. martech.org/going-deep-deep-learning-martech-insights-action-impact/.

[23] Timothée Lesort. Continual learning : Tackling catastrophic forgetting in deep neural networks with replay processes. 07 2020.

[24] Nikhil Buduma. *Fundamentals of Deep Learning : Designing next-generation Artificial Intelligence Algorithms*. O'Reilly, 2017.

[25] Valentino Zocca, Gianmario Spacagna, Daniel Slater, and Peter Roelants. *Python deep learning : Next generation techniques to revolutionize computer vision, AI, Speech and Data Analysis*. Packt Publishing, 2017.

[26] Ping-Huan Kuo, Ssu-Ting Lin, and Jun Hu. Dnae-gan : Noise-free acoustic signal generator by integrating autoencoder and generative adversarial network. *International Journal of Distributed Sensor Networks*, 16 :155014772092352, 05 2020.

[27] Pat Nakamoto. *Neural Networks amp ; Deep Learning : Explained to your granny*. Createspace Independent Publishing, 2017.

[28] Single layer perceptron in tensorflow - javatpoint. www.javatpoint.com/single-layer-perceptron-in-tensorflow.

[29] Marc Parizeau. Réseaux de neurones. *GIF-21140 et GIF-64326*, 124, 2004.

[30] Deep learning via multilayer perceptron classifier - dzone. dzone.com/articles/deep-learning-via-multilayer-perceptron-classifier.

[31] Valentino Zocca, Gianmario Spacagna, Daniel Slater, and Peter Roelants. *Python deep learning*. Packt Publishing Ltd, 2017.

[32] Andrea Volpini. How to build a keyword suggestion tool using tensorflow, Mar 2021. wordlift.io/blog/en/keyword-suggestion-tool-tensorflow/.

[33] basic-cnn-architecture, Oct 2022. www.upgrad.com/blog/basic-cnn-architecture/.

[34] ODSC Community. Understanding the mechanism and types of recurrent neural networks, Mar 2021. opendatascience.com/understanding-the-mechanism-and-types-of-recurring-neural-networks/.

[35] David Vint, Matthew Anderson, Yuhao Yang, Christos Ilioudis, Gaetano Di Caterina, and Carmine Clemente. Automatic target recognition for low resolution foliage penetrating sar images using cnns and gans. *Remote Sensing*, 13 :596, 02 2021.

[36] Autoencoder structure. www.upload.wikimedia.org/wikipedia/commons/2/28/Autoencoder_structure.png.

[37] Malware | what is malware how to stay protected from malware attacks. www.paloaltonetworks.com/cyberpedia/what-is-malware.

[38] Types of network firewall. www.geeksforgeeks.org/types-of-network-firewall/.

[39] Malware | what is malware how to stay protected from malware attacks. www.geekflare.com/fr/firewall-introduction/.

[40] What is encryption ? www.cloudflare.com/learning/ssl/what-is-encryption/.

[41] Symmetric vs. asymmetric encryption – what are differences ? www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences.

[42] What is demiltarized zone ? www.geeksforgeeks.org/what-is-demiltarized-zone/.

[43] Zone démilitarisée (informatique). https://fr.wikipedia.org/wiki/Zone_dÃĭmilitarisÃĭe_(informatique).

[44] Iso/iec 27001 - information security -. https://www.rigcert.org/iso_iec_27001-12.htm.

[45] What is ip spoofing, and how can you protect yourself. www.nordvpn.com/fr/blog/ip-spoofing/.

[46] Dns spoofing. www.imperva.com/learn/application-security/dns-spoofing/.

[47] What are man-in-the-middle attacks (mitm) and how to avoid them. www.invicti.com/blog/web-security/man-in-the-middle-attack-how-avoid.

[48] Intrusion detection system (ids) - geeksforgeeks. www.geeksforgeeks.org/intrusion-detection-system-ids/.

[49] What is an intrusion detection system ? - palo alto networks. www.paloaltonetworks.com/cyberpedia/what-is-an-intrusion-detection-system-ids.

[50] Keisuke Kato and Vitaly Klyuev. An intelligent ddos attack detection system using packet analysis and support vector machine. *IJICR*, 14(5) :3, 2014.

[51] Sweta Bhattacharya, Praveen Kumar Reddy Maddikunta, Rajesh Kaluri, Saurabh Singh, Thippa Reddy Gadekallu, Mamoun Alazab, and Usman Tariq. A novel pca-firefly based xgboost classification model for intrusion detection in networks using gpu. *Electronics*, 9(2) :219, 2020.

[52] Amin Shahraki, Mahmoud Abbasi, and Øystein Haugen. Boosting algorithms for network intrusion detection : A comparative evaluation of real adaboost, gentle adaboost and modest adaboost. *Engineering Applications of Artificial Intelligence*, 94 :103770, 2020.

[53] Jiyeon Kim, Jiwon Kim, Hyunjung Kim, Minsun Shim, and Eunjung Choi. Cnn-based network intrusion detection against denial-of-service attacks. *Electronics*, 9(6) :916, 2020.

[54] Nisha Ahuja, Gaurav Singal, and Debajyoti Mukhopadhyay. Dlsdn : Deep learning for ddos attack detection in software defined networking. In *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pages 683–688. IEEE, 2021.

[55] Mona Alduailij, Qazi Waqas Khan, Muhammad Tahir, Muhammad Sardaraz, Mai Alduailij, and Fazila Malik. Machine-learning-based ddos attack detection using mutual information and random forest feature importance method. *Symmetry*, 14(6) :1095, 2022.

[56] Arif Yulianto, Parman Sukarno, and Novian Anggis Suwastika. Improving adaboost-based intrusion detection system (ids) performance on cic ids 2017 dataset. In *Journal of Physics : Conference Series*, volume 1192, page 012018. IOP Publishing, 2019.

[57] Weiming Hu, Wei Hu, and Steve Maybank. Adaboost-based algorithm for network intrusion detection. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(2) :577–583, 2008.

[58] Chuanlong Yin, Yuefei Zhu, Jinlong Fei, and Xinzheng He. A deep learning approach for intrusion detection using recurrent neural networks. *Ieee Access*, 5 :21954–21961, 2017.

[59] Amardeep Singh and Julian Jang-Jaccard. Autoencoder-based unsupervised intrusion detection using multi-scale convolutional recurrent networks. *arXiv preprint arXiv :2204.03779*, 2022.

[60] Ddos 2019 | datasets | research | canadian institute for cybersecurity | unb. `www.unb.ca/cic/datasets/ddos-2019.html`.

[61] Ids 2018 | datasets | research | canadian institute for cybersecurity | unb. `www.unb.ca/cic/datasets/ids-2018.html`.

[62] Ids 2017 | datasets | research | canadian institute for cybersecurity | unb. `www.unb.ca/cic/datasets/ids-2017.html`.

[63] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote : synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16 :321–357, 2002.

[64] Simi Sanya. Ensemble deep learning. `www.analyticsvidhya.com/blog/2021/06/ensemble-deep-learning-an-ensemble-of-deep-learning-models/`.

[65] Python. `www.python.org/`.

[66] Tensorflow federated. `www.tensorflow.org/federated`.

[67] scikit-learn : machine learning in python. `www.scikit-learn.org`.

[68] Keras Team. Keras : Deep learning for humans. `www.keras.io`.

[69] pandas - python data analysis library. `WWW.pandas.pydata.org/`.

[70] Numpy. `www.numpy.org/`.

[71] Matplotlib — visualization with python. `www.matplotlib.org/`.

[72] seaborn : statistical data visualization. `www.seaborn.pydata.org/`.

[73] Project jupyter. `www.jupyter.org/`.

[74] Visual studio code - code editing. redefined. `www.code.visualstudio.com/`.