Ministry of Higher Education and Scientific Research

University of Blida 1
Faculty of Science
Department of Computer Science

Master Thesis

Field : Computer Science

Option : Security of Information Systems

Intrusion detection system for wireless sensor networks based on Deep learning

By :
Amine SI HADJ MOHAND

Proposed by:
Mrs Siham BACHA

In front of jury:

- Mrs Fatma Zohra ZAHRA, **President**
- Mrs Soraya CHERIGUENE, **Examiner**

2020/2021

Remerciement

Au cours de ce mémoire, j'ai reçu beaucoup de soutien et d'aide de nombreuses personnes. Avant tout, je remercie le bon dieu pour ses bénédictions et de m’avoir donné la volonté, le courage et la patience pour présenter ce travail. Je remercie énormément mes parents d'être toujours de mon côté.

Je tiens à exprimer toute ma reconnaissance à mon encadreur, Madame Siham Bacha. Je la remercie de m’avoir encadré, orienté, aidé et conseillé. Son expertise a été inestimable dans la formulation des questions et de la méthodologie de recherche. Vos commentaires perspicaces m'ont poussé à affiner ma réflexion et à soulever mon travail à un niveau supérieur.

J’adresse également, mes remerciements à Madame F. ZAHRA qui me fait l’honneur de présider le jury de ma soutenance. Mes remerciements s’adressent également à Madame S. CHERIGUENE pour avoir accepté de faire partie de ce jury.

Je tiens à remercier tous les professeurs de l’USDB et surtout du département d'informatique. Je tiens à vous remercier tous pour votre travail acharné et pour toutes les connaissances que j'ai acquises grâce à votre supervision et à vos conseils tout au long de mon cursus universitaire.

Aussi j’adresse mes sincères remerciements à toutes les personnes qui ont guidé mes réflexions et ont accepté de me rencontrer et de répondre à mes questions durant mon cursus et mes recherches.

Abstract

Human kind is an evolving creature that has always looked to make his life easier, that led him to create modern civilization. Technology is a non-stop arrow that keeps moving forward towards a bright future, brilliant innovations and new challenges. Wireless sensor networks (WSN) is one of the leading topics in our modern society, it’s the key to achieve the dream of global smart cities with IOT (Internet Of Things) devices and it’s the key for many exciting domains like remote medicine, smart agronomy, etc. However, WSNs are exposed to cyber threats every instant, whether they are intentionally by an adversary or because of poor system management. Thus, security in WSN is very critical and challenging at the same time due to the constraints of sensors like the concise abilities such as limited memory, energy consumption constraint. Therefore, any security measurements must take into consideration these factors in order to grant the network the fullest performance that it should get without any delay or packet loss or any sort of abnormal functioning.

In this thesis, an Intrusion detection system (IDS) engine is proposed based on machine learning and deep learning techniques combined with features engineering to obtain the highest results for good packet classification. KDDCUP99 and NSL-KDD are the datasets used to test the model, the data is preprocessed and then passed as input to a feature selection algorithm XGBoost that selects columns based on features scores .Since WSNs computation power is limited, binary classification is used, to classify normal from abnormal traffic using a Deep Neural network of four layers, input layer, two hidden layers and the output layer.

The results obtained during this study are accurate and precise compared to what researchers have accomplished in their published papers. For KDDCUP precision and accuracy scores are 99.86%, for NSL-KDD 83.21% precision score and 76.21% accuracy score.

ملخص

الإنسان مخلوق متطور يسعى دوما لجعل حياته أسهل، مما أدى به إلى إنشاء الحضارة الحديثة. من بين إسهاماته المتميزة التكنولوجيا التي هي عبارة عن سهم لا يتوقف، يستمر في المضي قدمًا نحو مستقبل مشرق وابتكارات مذهلة وتحديات جديدة. تعد شبكات الاستشعار اللاسلكية (WSN) أحد الموضوعات الرائدة في التكنولوجيا الحديثة، فهي المفتاح لتحقيق حلم المدن الذكية العالمية باستخدام أجهزة IOT (إنترنت الأشياء), كما هي المفتاح للعديد من المجالات المثيرة مثل ممارسة الطب عن بعد والهندسة الزراعية الذكية، إلخ. مع ذلك ، تتعرض شبكات WSN للتهديدات السيبرانية في كل لحظة، سواء كانت عن قصد من قبل خصم ما أو بسبب سوء إدارة للنظام. فبالتالي ، يعد الأمان في WSN أمرًا بالغ الأهمية وصعبًا في نفس الوقت بسبب القيود المحددة في أجهزة الاستشعار وشبكات WSN بشكل عام، نظرا للقدرات المحدودة و الضئيلة مثل الذاكرة الصغيرة نسبيا و وجوب الحفاظ على ادنى استهلاك ممكن للطاقة. لذلك، يجب أن تأخذ أي بروتوكولات أمنية في الاعتبار هذه العوامل من أجل منح الشبكة أقصى أداء يمكن أن تحصل عليه دون أي تأخير أو فقدان للحزم أو أي نوع من الأداء الغير الطبيعي.

حاولنا من خلال هذه الأطروحة اقتراح محرك نظام كشف التسلل (IDS) بناءً على التعلم الآلي وتقنيات التعلم العميق بالإضافة الى هندسة الميزات للحصول على نتائج عالية الدقة في تصنيف الحزم. KDDCUP99 و NSL-KDD هما مجموعتي البيانات المستخدمة لاختبار النموذج ، تم معالجة البيانات بتقنية Data Preprocessing ثم يتم تمريرها إلى خوارزمية اختيار الميزات XGBoost التي تختار الميزات بناءً على العلامات المتحصل عليها لكل ميزة من خلال الخوارزمية. نظرًا لأن قوة حساب WSN محدودة، تم استخدام التصنيف الثنائي لتصنيف حركة الحزم حسبما كانت عادية أو غير عادية أي مخترقة باستخدام شبكة عصبية عميقة من أربع طبقات: طبقة إدخال, طبقتين مخفيتين وطبقة الإخراج.

النتائج المتحصل عليها خلال هذه الدراسة دقيقة و جيدة مقارنة بما أنجزه الباحثون في أبحاثهم المنشورة. بالنسبة لKDDCUP ، تم الحصول على 99.86٪ لكل من accuracy و precision ، أما لمجموعة البيانات NSL-KDD تم الحصول على 83.21٪ precision و 76.21٪ accuracy.

Résumé

L’être humain est une créature en évolution qui a toujours cherché à se faciliter la vie, ce qui l'a amené à créer une civilisation moderne. La technologie est un de ces fondations qui constituent une flèche en mouvement qui continue d'avancer vers un avenir radieux, des innovations brillantes et vers de nouveaux défis. Les réseaux de capteurs sans fils (WSN) sont l'un des sujets intéressants dans l’actualité, c'est la clé pour réaliser le rêve des villes intelligentes dotées d’appareils IOT (Internet des objets) et la clé de nombreux domaines passionnants comme la médecine à distance, l'agronomie intelligente, etc. Cependant, les WSN sont exposés à des cyber menaces à chaque instant, qu'elles soient intentionnellement causées par un adversaire ou en raison d'une mauvaise gestion du système. Ainsi, la sécurité dans les WSN est très critique et difficile à la fois en raison des contraintes restrictives des capteurs du WSN comme la mémoire limitée, la contrainte de consommation d’énergie. Par conséquent, toutes les mesures de sécurité doivent prendre en compte ces facteurs afin d'accorder au réseau les performances les plus complètes qui devraient être obtenues sans aucun retard et aucune perte des paquets assurant aucun fonctionnement anormal.

Dans ce mémoire, un moteur de système de détection d'intrusion (IDS) est proposé basé sur des techniques d'apprentissage automatique et d'apprentissage profond combiné avec la sélection des caractéristiques visant à obtenir des bons résultats pour assurer une bonne classification des paquets. KDDCUP99 et NSL-KDD sont les ensembles de données utilisés pour tester le modèle, les données sont prétraitées puis injectées en entrée à l’algorithme de sélection de caractéristiques XGBoost qui sélectionne les colonnes en fonction des scores de caractéristiques. Étant donné que la puissance de calcul des WSN est limitée, une classification binaire est utilisée, pour classer le trafic normal du trafic anormal à l'aide d'un réseau de neurones profonds de quatre couches, une couche d'entrée, deux couches cachées et la couche de sortie.

Les résultats obtenus au cours de cette étude sont bons et précis par rapport à ce que les chercheurs ont accompli dans leurs articles publiés. Pour KDDCUP, les scores de précision et d’accuracy sont de 99,86%. Pour NSL-KDD, le score de précision est 83,21% et le score d’accuracy est 76,21%.

Table of content

Figures Table

Tables list

General Introduction

Technology is a critical need to our modern world, “*Once a new technology rolls over you, if you're not part of the steamroller, you're part of the road.”[1]* , which brings new challenges and obstacles to the specialists to enhance the quality of services offered, techniques used and systems security.

Information systems are always under threat of malicious entities “It takes 20 years to build a reputation and few minutes of cyber-incident to ruin it” by Stephane Nappo, it’s why every system has to acquire robust, accurate and neoteric security measures.

An intrusion-detection system (IDS) can be defined as the tools, methods, and resources to help identify, assess, and report unauthorized or unapproved network activity [2], it is a set of techniques and methods that are used to detect suspicious activity both at the network and host level. Intrusion detection systems fall into two basic categories: signature-based intrusion detection systems and anomaly detection systems [3].

A Wireless Sensor Network is considered to be the eyes and ears of IOT systems where they gather data from the external physical environment and transform it to understandable signals that could be treated for various WSNs application domains such as disaster prevention, agricultural management …etc.

The WSN field plays a huge role in future Internet of things systems by collecting a big amount of environmental data that could be used by intelligent machines to make the right decisions at the right timing, like Industry 4.0, IOT agriculture, Real-Time battlefield drones ...etc. Such complex and distributed systems are vulnerable and exposed to various types of attacks that need to be handled and eliminated if possible.

WSNs have unfortunately several constraints that could affect the well-functioning of the global system, since they are small devices with small capabilities, they should save energy as much as possible, also the fact that they contain small memory, the more data is in getting treated the more the quality of the network is infected there for it results a higher latency communication.

Since world technology is in progress, AI (artificial intelligence) takes the biggest share in papers and inventions. It is inevitable that AI is achieving huge impacts on major scientific fields like data science, astronomy …etc. Intrusion detection systems aren’t an exception when it comes to building a robust and a precise detection system. Scientists have used different techniques of machine learning and deep learning to train models for classification and separation of authentic traffic from the malicious one.

In this thesis, a detection engine of an Intrusion Detection System (IDS) is built for a Wireless Sensor Network (WSN) using deep neural network and machine learning methods. Combining deep neural networks with feature engineering techniques is the main approach used to achieve good results compared to what papers have discussed in earlier studies.

We have opted for binary classification in order to respect the WSNs constraints and in order to deploy our security protocol without infecting the quality of service of the network.

The thesis is structured of four chapters, chapter one contains general concepts of wireless sensor networks, definition of a sensor node, WSN topologies, communication protocols, field of applications, constraints such as energy and power consumption, security in WSN and finally types of attacks on them.

Chapter number two is about intrusion detection systems (IDS) and machine learning. It talks about IDS definition, it's architecture, methodology of detection, machine learning and deep and the state of the art on the application of an IDS into wireless sensor networks and recent attributions.

Chapter three discusses the proposed approach and methodology with a global scheme, explaining the three parts of the proposed detection model like preprocessing, feature selection and deep neural network structure and layers and some machine learning techniques and algorithms.

The last chapter contains the environment and libraries used in this work, the datasets used to train and test the model, and the results of the proposed approach. The results are displayed through confusion matrix, precision score, recall score, F-score and accuracy. Those results are compared in this chapter with the state of the art by following the same testing protocols as the papers have followed. It ends by displaying our proposed simulator with our proper simulation scenario.

Chapter 1: General concepts of WSNs

1. Introduction

Recent advances in wireless communications and electronics have enabled the development of low-cost, low-power, multifunctional sensor nodes that are small in size and communicate untethered in short distances. These tiny sensor nodes, which consist of sensing, data processing, and communicating components, leverage the idea of sensor networks. Sensor networks represent a significant improvement over traditional sensors [4]. So what is a sensor node? and what is it used for?

2. Sensor node

A Sensor node is a small and inexpensive device with limited resources of battery and computation power which are deployed in a region to monitor the environment, it is a device that possesses the capacity to gather sensor information from the environment, process the information and communicate with other nodes.[5] as it is shown on figure 1 a sensing region is a group of sensor nodes that are distributed in physical environment.

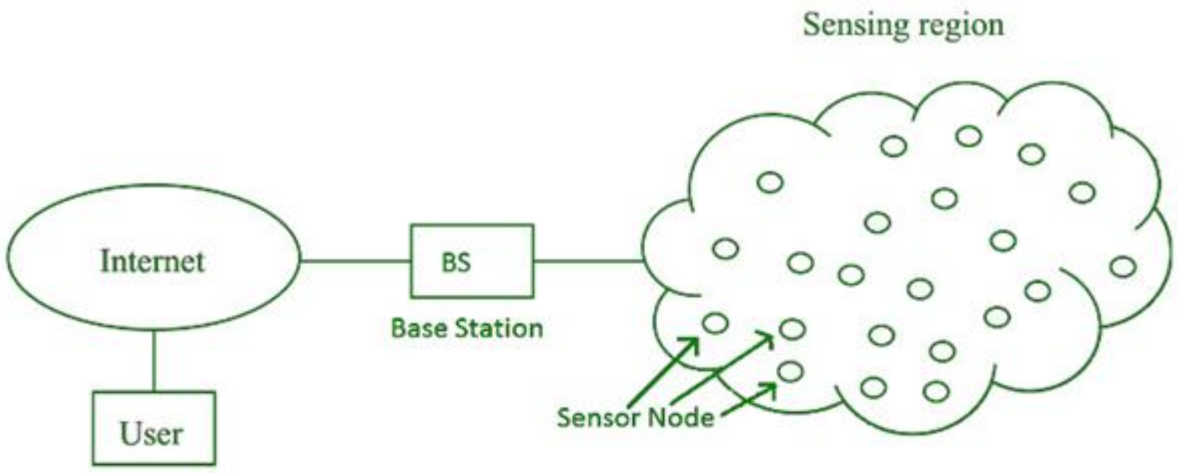


Figure 1 : Shows the general concept of sensor nodes [6]

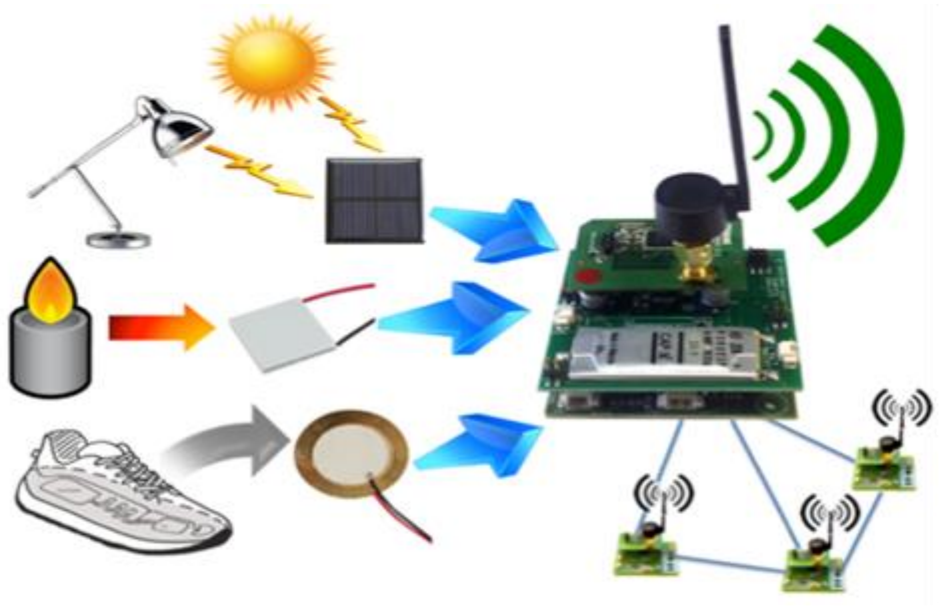


Figure 2: Shows how sensor nodes are related to real environment [7]

3. Wireless sensor network

A wireless sensor network is a collection of nodes organized into a cooperative network. Sensor networks spatially distributed autonomous sensors to monitor physical and environmental conditions at different locations, such as temperature, pressure, motion sound, vibration, etc. [8]

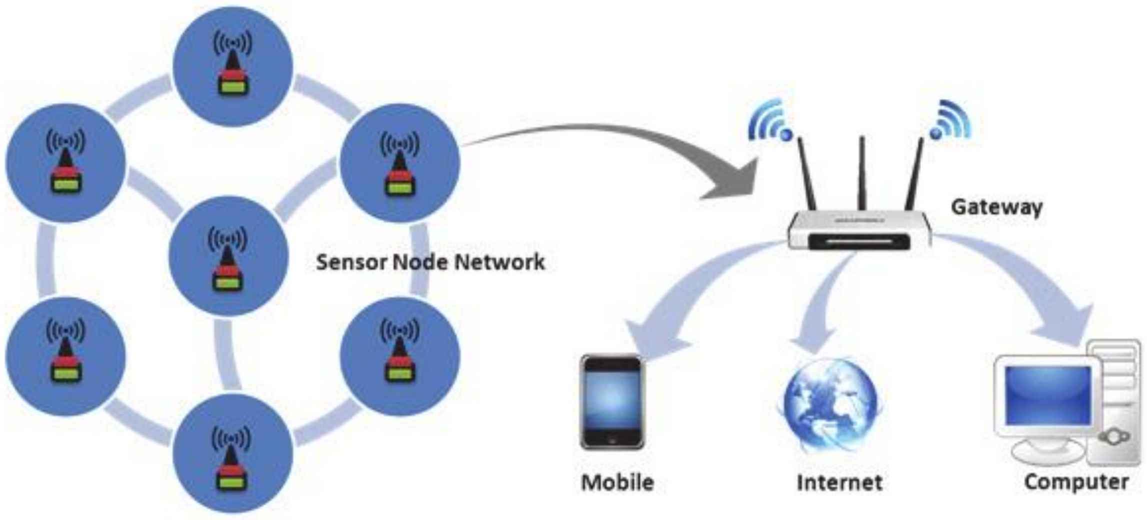


Figure 3: how a wireless sensor network could be linked to external networks [9]

4. WSN topologies

Various needs require various topologies, here are the common topologies among the WSN:

4.1. Point to point topology

Links two sensor nodes together directly.

4.2. Bus topology

It is a configuration in which each node is connected to a shared communication bus as shown in figure 2. A signal is transmitted in both directions along the bus until it reaches its intended destination. Bus networks must include a collision avoidance system to resolve issues when two nodes simultaneously send out data on the bus. Bus networks are simple and easy to install. However, there is a single point of failure: if the bus fails, the entire network fails. [10]

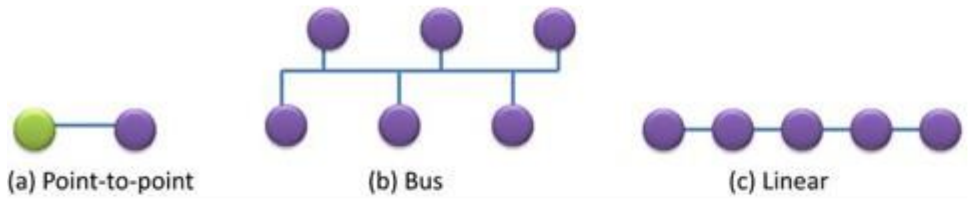


Figure 4: demonstration of Bus and point to point topology [10]

4.3. Tree topology

The network uses a central hub called a root node as the main communication router. In the hierarchy, the central hub is one level below from the root node. This lower level forms a star network. The tree network can be considered a hybrid of both the Star and Peer to Peer networking topologies as shown in figure below [8].

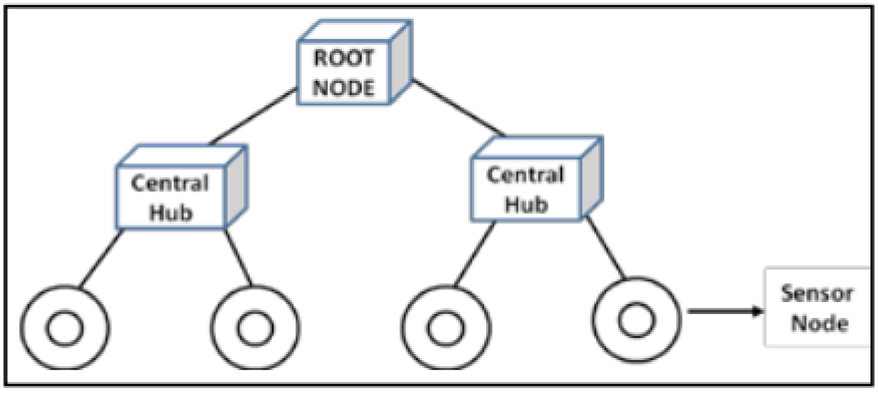


Figure 5: Tree Topology [8]

4.4. Star topology

Star networks are connected to a centralized communication hub (sink) and the nodes cannot communicate directly with each other. The entire communication must be routed through the centralized hub. Each node is then a "client" while the central hub is the "server or sink" as shown in the figure below. But there is a disadvantage of single path communication [8].

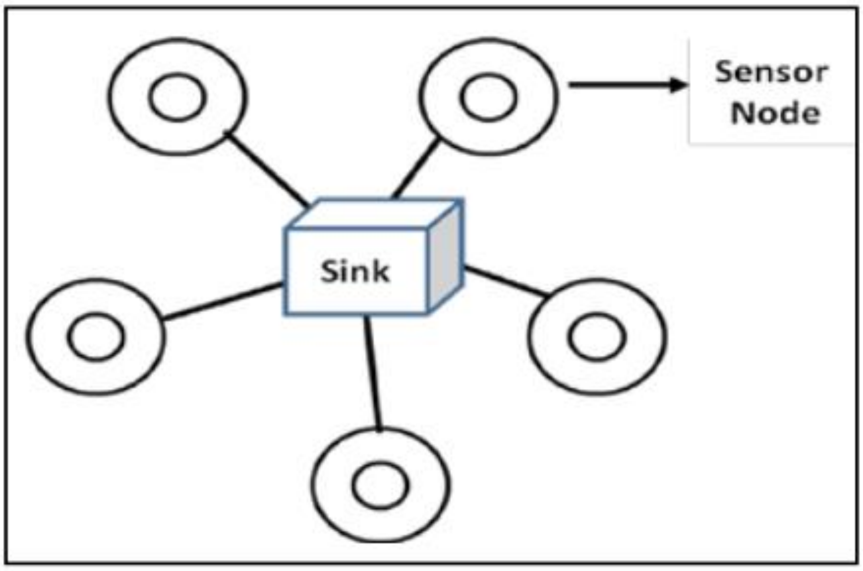


Figure 6: Star topology [8]

4.5. Ring topology

In a ring network, every node has exactly two neighbors for communication purposes. All messages travel through a ring in the same direction (either "clockwise" or "counterclockwise"). A failure in a node breaks the loop and can take down the entire network. but congestion of traffic and double path communication[8].

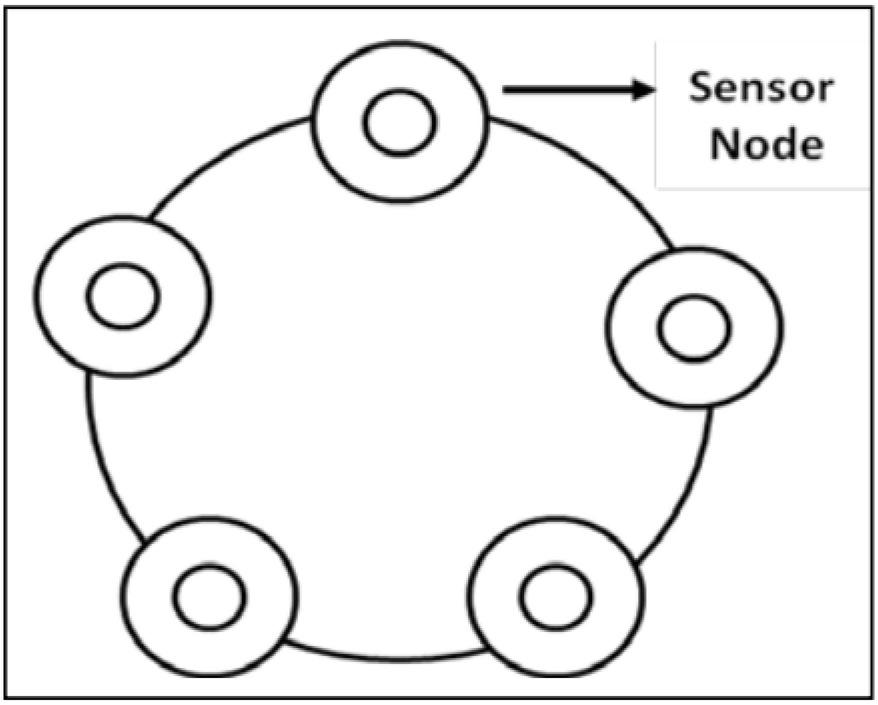


Figure 7: Ring topology [8]

4.6. Mesh topology

Nodes disseminate their own data and also act as relays to propagate the data from other nodes. There are two forms of mesh topology: a *partially connected mesh*, in which some nodes are connected to more than one other node, shown in Figure 8 (a); and a *fully connected mesh*, in which every node is connected to every other node in the mesh, shown in Figure 8 (b). Mesh networks are self-healing, as data can be routed along a different path if a node fails. Fully connected mesh networks are not suitable for large sensor networks as the number of connections required become unmanageable. Partially connected mesh networks provide the self-healing capability of a fully connected network without the connection overhead. Mesh topologies are most commonly found in wireless networking[10].

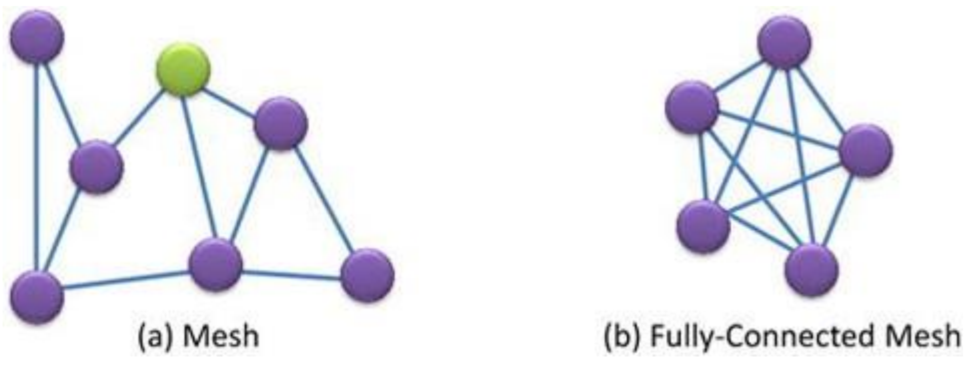


Figure 8: Mesh topology

4.7. Circular topology

In this topology, there is a circular sensing area and that the sensing area has a sink (at center). The sensor nodes sense the event of interest and transmit these data to the sink. The nodes are randomly deployed with uniform density all around the sink as shown in figure . Depending on the distance of a node from the sink and the transmission range of the nodes, data has to traverse single or multiple hops before being received by the sink[8].

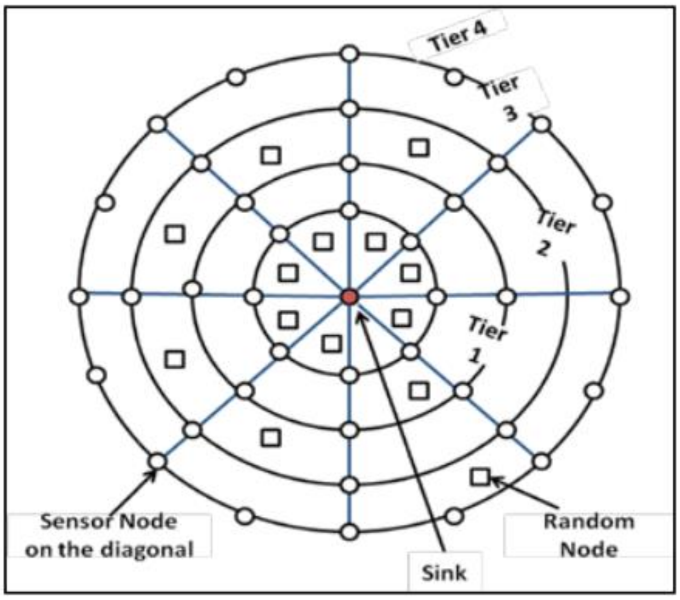


Figure 9: Circular topology [8]

4.8. Grid topology

The sensor network field divides into grids as shown in figure 8. The network area is partitioned into a non-overlapping square grid with the same size. There should be at least one and only one node in working state in each grid at any time. In order to extend the network lifetime, the nodes in a grid should work in turn. Inside each grid, one node is selected as a grid head which is responsible for forwarding routing information and transmitting data packets. Routing is performed in a grid-by- grid manner. Grid-based multi-path routing protocol intended to route packets fast, utilize and extend sensor nodes energy in addition to avoiding and handling network congestion when happens in the network [8].

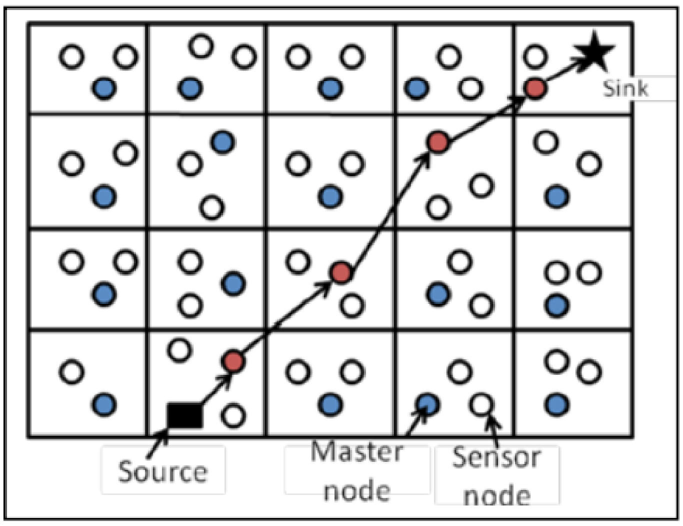


Figure 10: Grid topology [8]

5. Communication protocols

There are plenty of communication protocols in WSNs that are:

5.1. Direct transmission protocols

In this protocol, each sensor sends its data directly to the base station. If the base station is far away from the nodes, direct communication will require a large amount of transmit power from each node[11].

5.2. Minimum transfer energy protocols

Nodes get to act like routers in this protocol for other nodes without losing the capacity of sensing the environment. Some of these protocols only consider the energy of the transmitter and neglect the energy dissipation of the receivers in determining the routes[11].

5.3. Clustering protocol

In this protocol nodes are organized into clusters that communicate with a local base station, and these local base stations transmit the data to the global base station, where it is accessed by the end-user. This greatly reduces the distance nodes need to transmit their data, as typically the local base station is close to all the nodes in the cluster[11].

What is interesting about this type of protocol is that they are considered as energy efficient, since others tend to consume more energy.

5.4. LEACH (Low Energy Adaptive Clustering Hierarchy)

It is a self-organizing, adaptive clustering protocol that uses randomization to distribute the energy load evenly among the sensors in the network. In LEACH, the nodes organize themselves into local clusters, with one node acting as the local base station or cluster head [11], as it is shown in the figure 11:

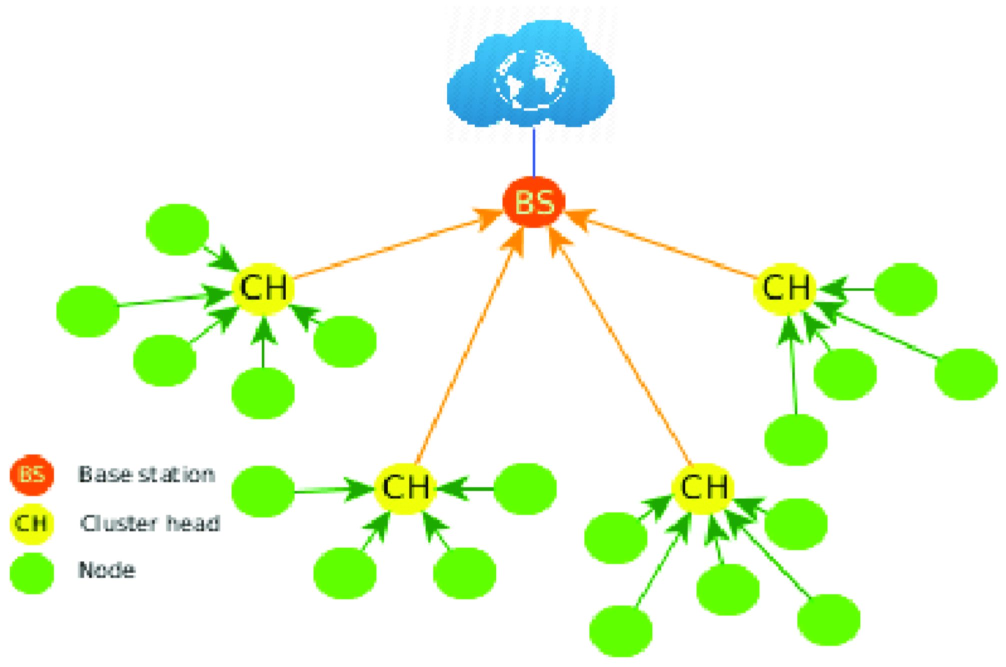


Figure 11: LEACH Protocol [12]

5.5. Stable Election Protocol

It is proposed based on weighted election probabilities of each node to become cluster-head (CHs) according to their respective energy. This approach ensures that the cluster head election is randomly selected and distributed based on the fraction of energy of each node assuring a uniform use of the node's energy. SEP also considered two types of nodes and two level hierarchies[11].

5.6. Hierarchical Cluster-Based Routing (HCR) Protocol

In HCR nodes self-organize themselves into clusters, each cluster is managed by a set of associates called head-set. Using round-robin technique, each associate acts as a cluster head (CH) [13].

The sensor nodes transmit data to their cluster heads, which transmit the aggregated data to the base station. Moreover, the energy-efficient clusters are retained for a longer period of time; the energy-efficient clusters are identified using heuristics-based approaches[11].

5.7. Genetic Algorithm

A genetic algorithm (GA) is used to create energy efficient clusters for data dissemination in wireless sensor networks. A GA is used at the base station, which provides energy efficient solutions to the optimizer. This provides the base station with the ability to determine the best cluster formation that will give minimum energy consumption during run time[11].

6. Fields of application of WSN

There are plenty of domains where WSNs could take place such as :

6.1. Environmental Observation and Forecasting

Environmental Observation and Forecasting have many of the application like Volcanic Studies and Eruption Warning System, Meteorological Observation, Fire Detection, Earthquake Studies and Warning System, Water Quality

Monitoring, Flood, Cyclone and Tsunami Warning System can be used to save the lives[14].

6.2. Disaster Prevention

In addition to the warning systems sensor networks can be used for hazardous workspaces like underground mining, steelworks, and refineries. Most of these places entail a high risk by nature which is amplified by poorly engineered constructions in developing countries[14].

6.3. Agricultural Management

The ability to retrieve soil moisture in real time enables efficient irrigation and agricultural planning which is especially important in semi-arid regions of developing countries[15].

6.4. Habitat Monitoring

The author in [14] says that the same holds for habitat monitoring, where wildlife can be studied without unnecessary human intrusion in remote areas. of special concern are the disturbance effects of even well-intended researchers frequently trampling into the animal’s habitat this can lead to distorted results by changing behavioral patterns or distributions or even reduce sensitive populations by increasing stress factors.

6.5. Structure Health Monitoring

As for structure health monitoring, safety, environmental, and commercial aspects come into play. Fast detection of a leaking oil pipeline can certainly reduce environmental damages and thus reduce negative impacts on the health of people and animals. At the same time, remote controlling of railroads and pipelines enhance effectiveness and help to reduce expenses[14].

6.6. Smart aeration

Aerating is the process of punching holes into land to allow water, oxygen, fertilizers, and other nutrients to penetrate the soil and better reach the roots of your grass. Wireless underground sensor networks (WUSN) can be used for it. They measure the physical phenomena and inform the node about the lack of water, oxygen and other values [16].

6.7. Intelligent lighting control

Intelligent light control system, which finds and manages the best light actuation, profiles using incident light measurements by light sensors and user requirements[14].

6.8. Public security

Wireless magnetic sensor networks offer a very attractive, low-cost technology for traffic measurement in freeways, urban street intersections and presence detection in parking lots[14]

6.9. Supply-chain monitoring in state-of-art production plants

RESCUEIT project by SAP Labs France is integration of Wireless Sensor Networks for secure Supply Chain Management Systems. The objective for deployment is a secure solution within WSNs in order to mitigate identified risk in the supply chain. On the background of these application domains, we foreseen the deployment of security mechanisms in WSNs based on identified risk in the Supply Chain. Solutions for secure tracking, optimization of the supply chain with preserving privacy of all parties involved in the supply chain is one of our major concerns[17].

6.10. WSN for healthcare

WSNs can effectively be used in healthcare for health monitoring, smart nursing homes, in-home assistance, telemedicine, and wireless body area networks[18].

7. WSN constraints

Since sensor networks rely on devices that don't acquire high capabilities in performance, energy, storage …etc. It is relevant that deploying security measures in the WSNs is difficult and it is recommended to optimize the solutions proposed for better security.

In [19], the author has mentioned the constraints of WSNs:

7.1. Energy constraint

Energy is an important factor in WSNs since they are small devices with low battery endurance. Energy consumption in WSNs are categorized to three types, energy for the sensor transducer, energy for communication among sensor nodes and energy for microprocessor computation.

This study [20] shows that each bit transmitted in WSNs consumes about as much power as executing 800 to 1000 instructions, which means that communication costs much more than computation.

7.2. Memory limitation

The sensor is a small device that possesses a small memory and a small storage space. Memory is a sensor node that usually includes flash memory and RAM. Flash memory is used for storing downloaded application code and RAM is used for storing application programs, sensor data, and intermediate results of computations.

7.3. Unreliable communication

WSNs communication are vulnerable due to their wireless nature, it is considered as a threat since Normally the packet-based routing of sensor networks is based on connectionless protocols and thus inherently unreliable, a wireless sensor network could be a victim of various attacks such as SinkHole, BlackHole, DDOS and many more. Packets might be damaged or dropped in congested nodes.

7.4. Higher latency in communication

The multi-hop routing, low computing capacity, network congestion may directly affect the latency in packet transmission badly. Security mechanisms that depend on synchronization would be at its worst from this constraint such as cryptographic key distribution, instant malicious events reporting …etc.

8. Security in WSNs

Wireless sensor networks share many commonalities with typical computer networks but it’s still a special type of network with unique characteristics. It is highly demanded that the resources of WSNs must be protected from attacks and prevent nodes from behaving maliciously without forgetting the importance of protecting the data that is communicated through the network .

Most important security constraints are listed below[19]:

8.1. Data confidentiality

Simply no unauthorized person that intercepts data would be able to understand the message transmitted, the security measures must ensure the confidentiality of every bit of the traffic in the network.

A sensor node should not allow its neighbors to access or read data unless it’s authorized to act like it. The key distribution mechanism should be vigorous and optimal. public information such as sensor identities, and public keys of the nodes should also be encrypted in certain cases to protect against traffic analysis attacks.

8.2. Data integrity

No message sent or received by any node in the network is altered or modified, an attack that aims to harm data integrity is mostly vital and dangerous. Security measures must ensure this constraint fulfillment.

8.3. Availability

Dos/DDOS attacks are a huge research field that keeps growing with time, since this kind of attack plays with this constraint and makes resources/data unreachable which directly affects the availability. Availability means that under any circumstance (internal or external attack) the services of WSNs should be always present.

8.4. Data freshness

Freshness of data means that the data is recent and which ensures no adversary can replay old messages. Shared key technique that WSN uses for message communication relies on data freshness since an attacker could perform a replay attack using old key as a new one is being refreshed and propagated to all the nodes in the WSN.

A nonce or time-specific counter could be added to each packet to verify the recentness of the packet.

8.5. Self-organization

"Each node in a WSN should be self-organizing and self-healing. This feature of a WSN also poses a great challenge to security. The dynamic nature of a WSN makes it sometimes impossible to deploy any pre-installed shared key mechanism among the nodes and the base station"[21].

It is recommended that the nodes of a WSN self-organize themselves for multi-hop routing and to obtain a good key management since the application of public key cryptographic technique requires an efficient and robust key-distribution system.

8.6. Secure localization

Having the exact location of a sensor within the WSN is an essential need, a potential attacker may falsify node locations by reporting false signal strength, replaying intercepted messages … etc.

In this article [22] the author presents how an unsafe localization system can be compromised in various ways to negotiate the entire functioning of a WSN, and thus lead to erroneous data aggregation and incorrect decision making.

8.7. Time synchronization

Sensor networks activities and applications generally require a high performing time synchronization. Thus, every security mechanism deployed on it must be also time synchronized such as event reporting or cryptographic key distribution.

8.8. Authentication

Ensuring authentication is a must in every information system, WSNs aren't different, the system must confirm that the communicating node is the exact one it claims to be. Man in the middle attack could be harmful in situations where authentication isn't ensured with highly secured protocol, it is essential for a receiver to possess a mechanism to verify that the packets received from node X are actually sent from node X and no other node.

9. Types of attacks

Like every information system, WSNs are exposed to various types of attacks that are mentioned below:

9.1. Attacks on secrecy and authentication

Secrecy and authentication are a vital points in WSNs as much as every information system, below are some common attacks on them:

9.1.1. Node replication attack

The attacker aims to add a node to a wireless senor network by copying the identifiers of an existing node which may lead to have a compromised network, it could lead to serious damages and severe disruption in message transmission whether by forwarding or corrupting the packets in wrong routes[19].

If the attacker attempts to put the malicious node in strategic network locations he could easily manipulate the flow in the network and could cause network partitioning, and if he gains the physical access he could copy the cryptographic keys used in communication and use them freely.

9.1.2. Attacks on privacy

Data privacy is without doubt an essential concept in security and a requirement in every information system, WSNs aren't any different, they collect data from the vast environment automatically which gives a bigger exploitable vulnerability depending on the size of the network.

If an opponent tries to collect data from the network by performing network sniffing technics and tries to aggregate the collected data properly he may arrive to break through the privacy measures used within the system, which leads us to comprehend that the privacy is a very challenging problem when it comes to WSNs and crowded ones especially.

The author here [19] explains some of common attacks on privacy:

- Eavesdropping and passive monitoring: In this kind of attacks, an attacker could easily obtain the content of transmitted data if they are not protected with cryptography measures.

- Traffic analysis: The authors here [23] have explained how easily an attacker can identify the base station in a wireless sensor network by only analyzing the traffic without understanding the content of messages delivered. Eavesdropping combined with traffic analysis could make an effective threat on privacy.

- Camouflage: The term camouflage in military means hiding within the nature and environment without letting the opponent sense the existence of danger, in case of WSN camouflage means hiding or compromising a node and make it seem as a normal authentic one. The malicious node at this case could deliver false routing informations or attract packets from another nodes.

9.2. Attacks on availability

Availability of services in WSNs is an important concept that the attackers aim to disturb it by various attacks and harmful acts, here are some common ones [19]:

9.2.1. Denial of service attacks (DoS)

DoS attacks attempt to reduce the networks capabilities as much as it is possible and prevent it to function properly and It consists of exhausting the network resources (memory and computation capabilities).

The researchers have proposed many approaches and techniques to detect and reduce the effect of such harmful attack but it’s still a domain of research since attackers are smart enough to create new DoS attack technics which means that it’s still and remains (until the proof of the contrary) a challenging attack to security experts.

9.2.2. Physical layer attacks

“The physical layer is responsible for frequency selection, carrier frequency generation, signal detection, modulation, and data encryption”[24].

There are two broad categories for physical layer attacks [19]:

- Jamming: It’s an attack where radio frequencies interfere in WSNs communications. A jamming attack could be very powerful that it shuts down the whole network.

- Tampering: Since WSNs are distributed within the opened environment physical attacks are highly possible, which means the attempt of attackers to tamper node/nodes captured is more likely to happen in a non-surveilled environment, and it could be for many reasons such as extracting the cryptographic keys or modify the program within the node or aim for a an attack on privacy like camouflage.

9.2.3. Link layer attacks

"The link layer is responsible for multiplexing of data-streams, data frame detection, medium access control, and error control"[24].

Collision and resource exhaustion are the main purpose of attacks on this layer, it occurs when two nodes attempt to transmit on the same frequency simultaneously.

9.2.4. Network layer attacks

SinkHole, Hello Flood, Sybil, Selective packet forwarding are very common attacks on network layer. The nature of the layer is vulnerable to this kind of attacks, it is important to counter them for an accurate network.

SinkHole is an attack that makes a malicious node attract packets from its neighbors by making them chose the compromised node as the next hop to route their data through.

Selective packet forwarding is by dropping or forwarding packets in multi-hop network, it is achieved by compromising a node, SinkHole is an advanced attack of selective forwarding .

9.2.5. Transport layer attacks

Flooding attack and de-synchronized attack are the ones that are launched more often in the transport layer, the consist of [19]:

- Flooding: memory exhaustion through flooding is a vulnerability that raises from communication protocols that maintains a certain state that leads to resources consumption in case of an attack.

- De-synchronization: It means the disruption of an existing connection.

9.3. Stealthy attack against integrity

The main goal of the attacker from performing such attack is to make the network accept a false data values. For example, an attacker compromises a sensor node and injects a false data value through that sensor node.

10. Conclusion

In this chapter , we have defined a sensor node, a wireless sensor network architecture. Also we have discussed about different existing WSN topologies such as Bus, tree, mesh, etc. There are various communication protocols that control the traffic flow within WSN like LEACH, HCR, etc. WSNs could be used in many fields and they could be deployed in various domains in our modern lives such as disaster prevention, agricultural management, habitat monitoring, etc.

Just like every technology, WSN has several constraints such as energy constraint, memory limitation and unreliable communication due to the wireless communication nature of WSNs. WSNs must ensure data confidentiality, integrity, availability and freshness. Also they must be self-organized and synchronized.

Three types of attacks could be performed on WSN which are : attacks on secrecy and authentication, attacks on availability and attacks against integrity.

Chapter 2: Intrusion Detection Systems (IDS) and machine learning

1. Introduction

Every information technology infrastructure is exposed to both known and unknown threats every instant. Malicious entities could execute attacks on the information system using various methods aiming for different goals, whether to steal data, or destroy it, or maybe just make it unreachable.

Scientists have studied and tried to prevent those malicious entities from taking advantage of systems weaknesses or at least make it as hard as possible for them to exploit vulnerabilities. The ideal victim is the one that follows weak security protocols. Therefore, every system has to acquire updated, robust and accurate security measures in order to protect its data from any threats.

An Intrusion Detection System (IDS) is a famous example of what could be installed within a network to detect any sort of malicious activity. And what makes it more interesting is the fact that some IDSs have started to use artificial intelligence techniques in order to gain a good rate of intrusion detection.

2. Definition

An intrusion-detection system (IDS) can be defined as the tools, methods, and resources to help identify, assess, and report unauthorized or unapproved network activity [2], it is a set of techniques and methods that are used to detect suspicious activity both at the network and host level. Intrusion detection systems fall into two basic categories: signature-based intrusion detection systems and anomaly detection systems [3].

It is accurate that each information system has an intrusion detection system installed.

3. General architecture

From [25], There has been many proposed architectures in the literature [26, 27] [28] but they generally rely on the scheme in figure 12:

- Data gathering (sensor): Data collection unit from the monitored system
- Detector (Intrusion Detection analysis Engine): The processing unit of the incoming data from the sensor in order to spot any malicious activity
- Knowledge base: It is the guideline or the reference that the IDS relies on to identify the intrusions, it contains information collected from sensors in a preprocessed format.
- Configuration device: Current state of the IDS.
- Response component: It reacts when an intrusion is detected in an active (automated) way or in an inactive state that requires human interaction.
-

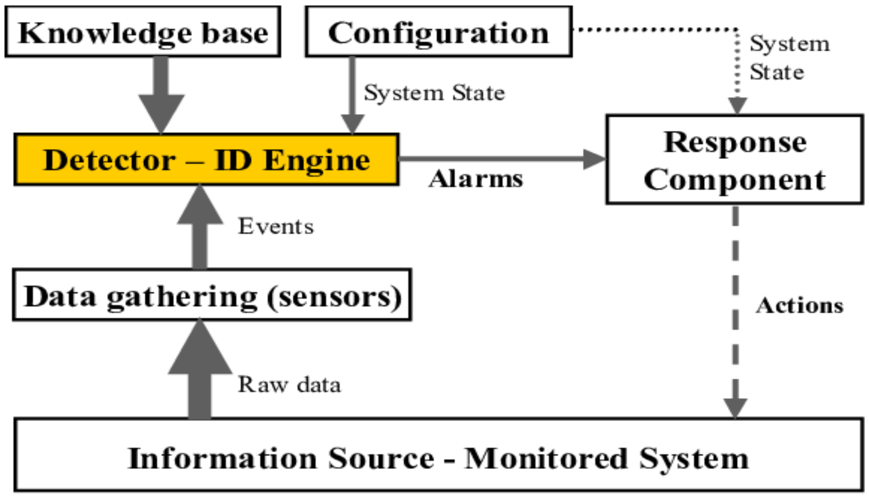


Figure 12: IDS General architecture [25]

4. Methodology of detection

In the literature [29] [30] many methodologies of intrusion detection was proposed during several years for different needs and for different systems:

4.1. Anomaly Detection

Anomaly IDS are based on identifying patterns defining normal and abnormal traffic. These IDS can be classified into subcategories based on the training method used. These categories are identified respectively as statistical, knowledge-based and machine learning based[30]. Statistical (4.2.2.1) includes univariate, multivariate and time series. Knowledge-based uses finite state machines and rules like case-based, n-based, expert systems and descriptor languages. Finally, machine learning includes artificial neural networks, clustering, genetic algorithms, deep learning, . . . Specification-based combines the strength of both signature and anomaly based to form a hybrid model.

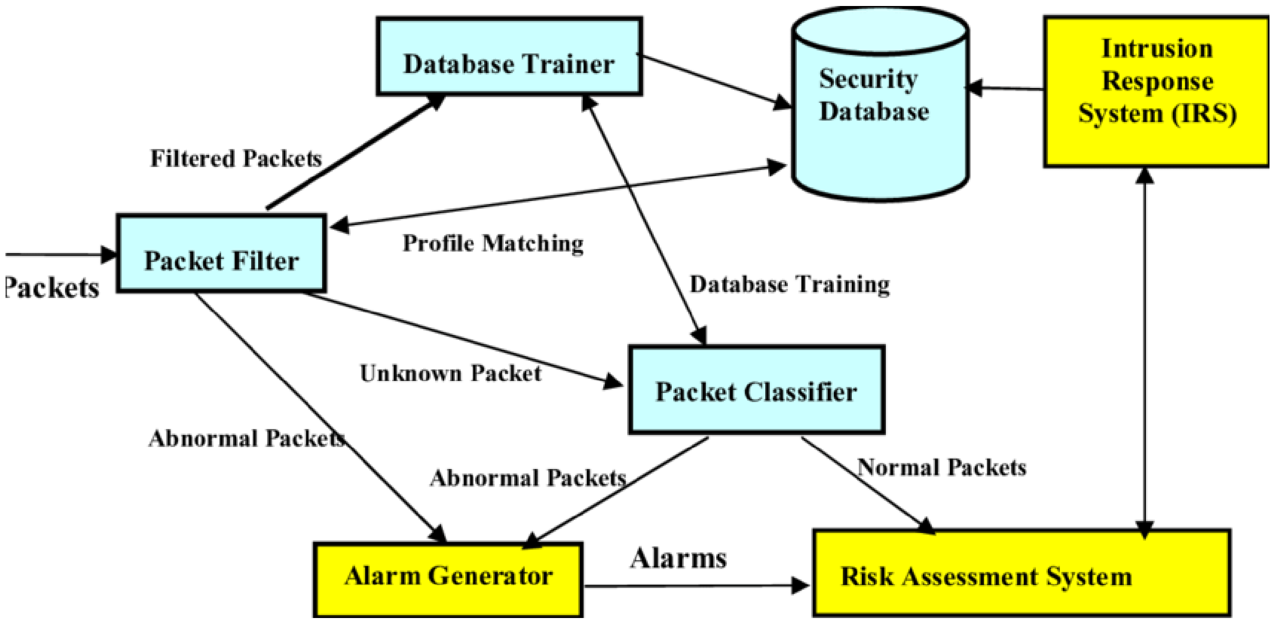


Figure 13: Anomaly Based IDS Architecture [31]

4.2. Signature Detection

Signature detection signifies the ability to detect a certain pattern within an attack like byte sequences in network data traffic or a certain sequence of a known malicious instructions, in [30], the author says that Signature based IDS are based on prior threat detection and the creation of accurate signatures. The main advantage of this method is the high accuracy for known attacks.

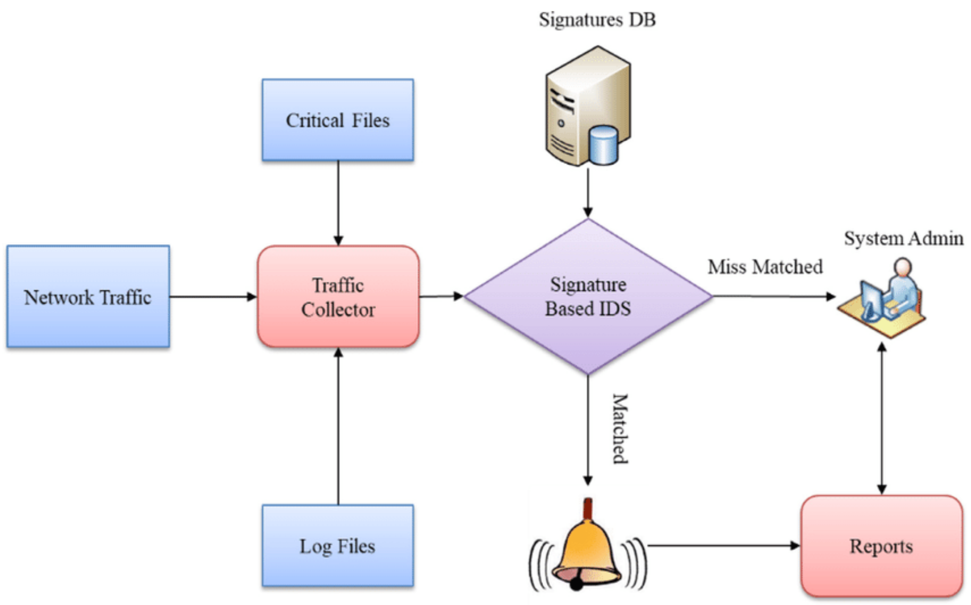


Figure 14: Signature based IDS Architecture[32]

4.3. Specification Detection

It combines the both signature and anomaly based to form a hybrid model.

5. Types of intrusion detection systems

5.1. Network based Intrusion Detection System (NIDS):

Network based IDS is the analysis tool of the network that spots malicious activity depending on the accuracy of the detection engine and the strategy of implantation within the network.

It is used in packet level analysis for all systems in the network segment by checking IP, transport-network and application protocol level activities and headers of packets to detect many IP-based DOS attacks like TCP SYN attack, fragment packet attack[33]. It can detect attacks as they occur. However, NIDS does not indicate if such attacks are successful or not since it doesn't analyze the log system[29]. The problem with NIDS is that it has restricted visibility inside the host machine, and there is no effective way to analyze encrypted network traffic to detect an attack[33].

5.2. Host based Intrusion Detection System (HIDS):

HIDS are installed on hosts or devices in a network, some security policies requires controlling the traffic in and out in the nodes of a network, It monitors and analyzes the internal computing system or system level activities of single host such as: system configuration, application activity, wireless network traffic (only for that host) or network interface, system logs or audit log, running user or application processes, file access and modification[29].

5.3. Hybrid Intrusion Detection System (HIDS):

Combines two types or more of IDS to achieve the advantages of IDS and complete an accurate detection [34] but it takes a long time in analyzing data.

A Hybrid IDS analyzes the traffic to and from the specific computer on which the intrusion detection software is installed. A host-based system also has the ability to monitor key system files and any attempt to overwrite these files[35].

6. Machine learning

Before talking about machine learning we need to comprehend what artificial intelligence (AI) is and what are its goals. AI is considered to be any cognitive process done by a machine for example image recognition, voice recognition which are fundamental abilities of a human brain.

The author [36] defines machine learning as : "*Machine learning (ML) refers to a system's ability to acquire, and integrate knowledge through large-scale observations, and to improve, and extend itself by learning new knowledge rather than by being programmed with that knowledge".* It is considered as one of the most human life changing aspects of all time, ML has opened limitless possibilities of advancement in various fields such as economy, biology, data science and many more.

Machine learning algorithms are able to process large amounts of data and extract useful information. In this way, they can improve upon their previous iterations by learning from the data they are provided.

There are mainly three basic types of machine learning algorithms[37] :

6.1. Supervised learning:

Supervised learning is defined by its use of labeled datasets to train algorithms to classify data or predict outcomes accurately. It uses a training set from the dataset in order to teach the model the right pattern in order to generate accurate outputs.

The training set contains inputs and correct outputs to make the model learn over time, the algorithm then finds relationships between the parameters given. The accuracy of the model if calculated using the loss function.

There are many application examples of supervised learning, it is used for predictive analysis, sentiment analysis, spam detection, image recognition and many other problems.

There are two categories of supervised learning :

6.1.1. Classification

The most known example for this category is the variables and concludes the label of the testing set as outputs. Decision trees, K-nearest neighbor, random forest and linear classifier are widely common among supervised classification machine learning algorithms classification of the IRIS flowers from sepal and petal dimensions where we have the dataset that contains as variables (or features) the sepal length and the width and also the petal length and width and since we are on the supervised learning the class is also assigned to each row of the dataset.

The classification uses an algorithm to classify the testing set according to the previous experience with the training set, it recognizes specific entities and patterns within the data.

6.1.2. Regression

Linear regression, polynomial regression and logistical regression are very popular regression algorithms that are used to understand the relationship between features whether they are dependent or independent. Regression problem occurs when the output variable is a real or a continuous value like “weight” or “height” for example attempting to predict the age of a person.

6.2. Unsupervised learning:

On the contrary of supervised learning, unsupervised learning doesn’t use labeled training sets, it is used in order to analyze and cluster unlabeled datasets.

Unsupervised learning algorithms aim to dig and extract hidden patterns or data groupings without human interference, which is a huge advantage that gives the ability to work with a larger scale of unlabeled datasets.

Supervised learning offers the capacity to comprehend the exact nature of relationship of any two data points, which is not available with unsupervised learning, the relationships are completely determined by the algorithms used in an abstract manner.

"Its ability to discover similarities and differences in information make it the ideal solution for exploratory data analysis, cross-selling strategies, customer segmentation, and image recognition."[38].

Mentioning unsupervised learning without the clustering technique is impossible, it's a technique that groups unlabeled data based on their similarities or differences. Clustering algorithms can be categorized as follows: specifically exclusive, overlapping, hierarchical, and probabilistic.

K-means is a famous example for exclusive clustering, where the data is affected to K group where K is the number of clusters based on the distance from each group's centroid.

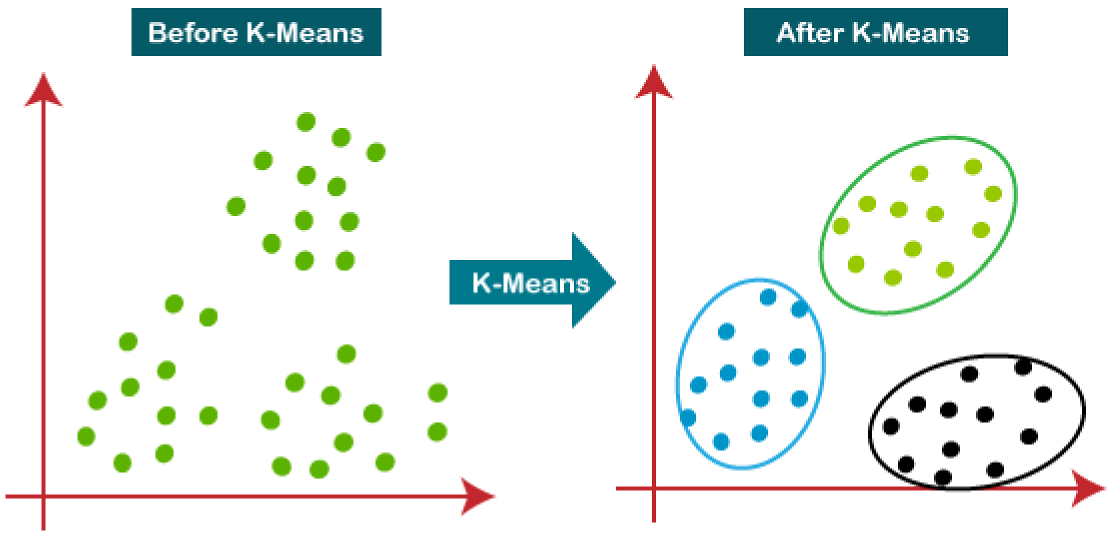


Figure 15: The effect of K-Means application [39]

Fuzzy clustering is under the overlapping category; it's similar to K-means, the difference is that a data point could belong to more than one cluster.

There are many application examples of unsupervised learning such as medical imaging, anomaly detection, computer vision, recommendation engines and many more.

6.3. Reinforcement learning

Reinforcement learning is a similar approach to supervised learning, they both use mapping between input and output, however the model isn't trained using sample data, it learns by using trial and error as it goes. The encountered problem gets solved by building a sequence of successful results that are reinforced to obtain the best recommendation or policy.

Unlike supervised learning, reinforcement learning uses reward and punishment as signals for positive and negative behavior[40] where it's not the case in supervised learning, in order to perform a task the model is guided by a correct set of actions during the training.

It has the ability to learn how to map a series of inputs to outputs with dependencies [41], Favorable outputs are encouraged or reinforced, and non-favorable ones are discouraged, it's a technique that enables an agent to learn in an interactive environment from its own actions and experiences.

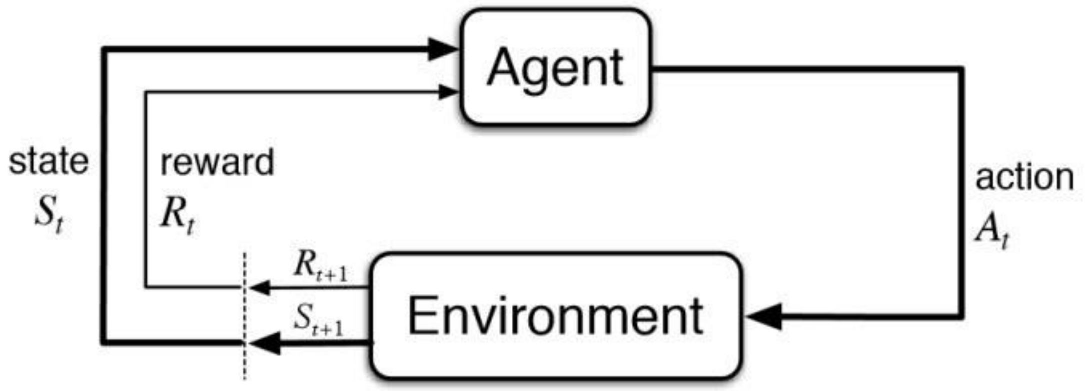


Figure 16: Reinforcement learning mechanism [40]

There are plenty of applications example for reinforcement learning, autonomous car driving is one huge domain that learning by reinforcement is the most accurate since the physical environment has always something new to learn (for example : a new object crossing the road in a new location where the car needs to comprehend to take the decision to stop, which is not the case when a moving car is in front of the self-driving vehicle).

Trading and finance has always been unpredictable patterns especially stocks which makes them also a perfect domain for reinforcement learning application.

Since we are talking about interactive new environments, gaming should be taken into consideration in reinforcement learning, as an example the AlphaGO game that was developed by Google DeepMind company that has learned how to play Go-game (It's a famous Japanese game) it even defeated a professional player.

6.4. Generative vs discriminative models

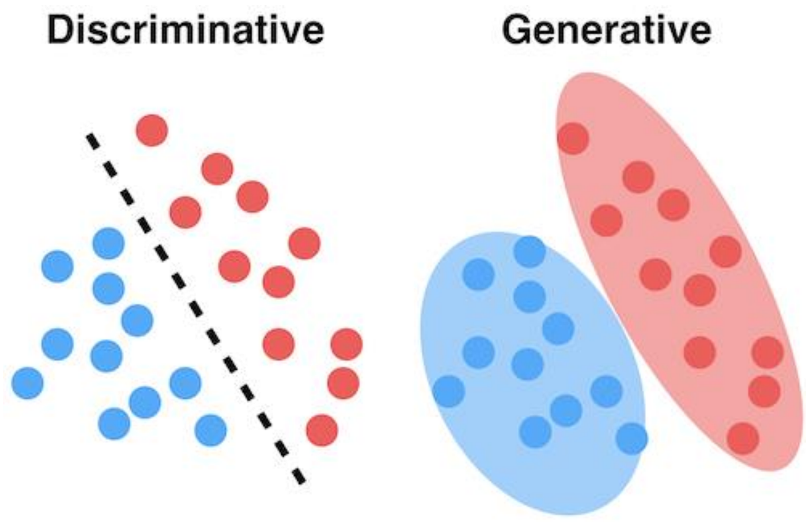


Figure 17: Description of generative and discriminative models effect on data [42]

A discriminative model makes predictions on the unseen data based on conditional probability and can be used either for classification or regression problem statements; they are known as conditional models because they try to learn the boundaries and limits between every class or labels in a dataset. Some examples of discriminative models: Boosting, decision trees, random forests, linear regression …etc.

On the contrary, a generative model focuses on the distribution of a dataset to return a probability for a given example; they are called generative because they can generate new data instances. They could be used for modeling data points or to describe a phenomena in data. Some examples of generative models: Generative adversarial network (GAN), Boltzmann machine, Deep belief networks … etc.

7. Deep learning

Deep learning is a machine learning method based on artificial neural networks, " *Deep learning describes a family of learning algorithms rather than a single method that can be used to learn complex prediction models, e.g., multi-layer neural networks with many hidden units"[43] [44].* Deep learning has been used on various domains such as image recognition, speech recognition, computational biology, natural language processing and has proven its efficiency in many other domains.

Artificial neural network is a mathematical function that simulates the functioning of a biological brain cell, it's drawn as interconnected circles, each circle represents an artificial neuron. Neurons compute the weighted average of its inputs, then do the sum of it and then pass it through a nonlinear function called activation function such as sigmoid. The output of one single neuron could be sent to another neuron for further computation as it shown in the figure below (wi represents weights and xi inputs):

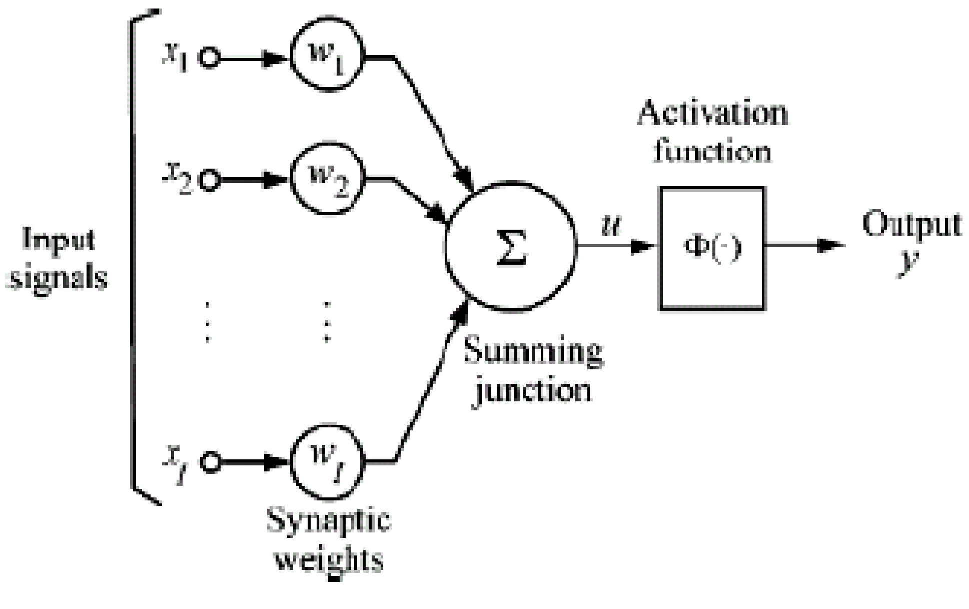


Figure 18: The architecture of a neural node [45]

An Artificial Neural Network (ANN) is a massively parallel distributed processor made up of simple processing units that has a natural propensity for storing experimental knowledge and making it available for use in analytical way [46], the basic architecture of neural network is: Composed of three layers of artificial neurons, input layer, hidden layers and output layers as it is shown in the figure below:

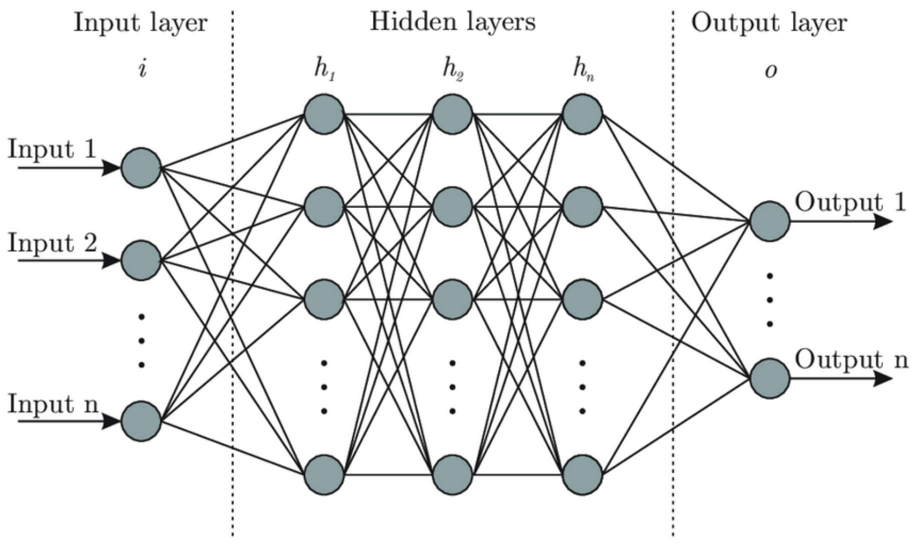


Figure 19: General architecture of deep learning neural network [47]

Each neuron of the input layer distributes its values to all of the neurons in the middle layers. Along each connection between input and middle neurons there is a connection weight so that the middle neuron receives the product of the value from the input neuron and the connection weight. Each neuron in the middle layer takes the sum of its weighted inputs and then applies a nonlinear function to the sum. The result of the function then becomes the output from that particular middle neuron. Each middle neuron of the final layer is connected to each output neuron. Along each connection between a middle neuron and the output neuron there is a connection weight. In the final step, the output neuron takes the weighted sum of its inputs and applies the non-linear function to the weighted sum. The result of this function becomes the output for the entire artificial neural network [48].

7.1. Types of Deep Neural Networks

Scientists have developed over the years many types of deep neural networks here are some common ones:

7.1.1. Deep Feed-Forward neural networks

Called also Deep Neural Networks (DNN), it's a sophisticated version of the feed forward neural network but with more hidden layers in the middle *"A deep neural network (DNN) with multiple hidden layers can learn more higher-level, abstract representations of the input" [49]*. Therefore DNN is more accurate compared to feedforward networks.

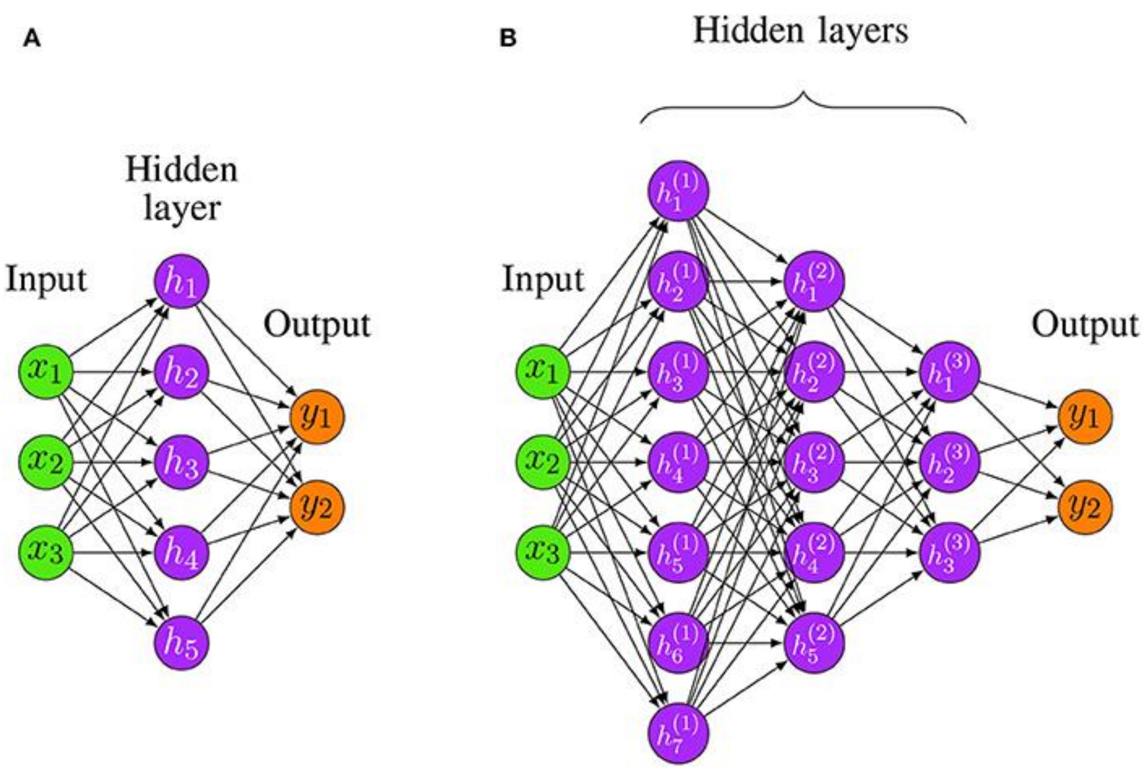


Figure 20: The figure A shows feedforward neural network and B deep feedforward neural network [43]

7.1.2. Convolutional neural networks

“A Convolutional Neural Network (CNN) is a special Feedforward Neural Network utilizing convolution, ReLU and pooling layers. Standard CNNs are normally composed of several Feedforward Neural Network layers including convolution, pooling, and fully-connected layers.”[43].

CNNs are composed of two major parts, feature learning and classification. The feature learning process is also composed of convolutional layers and pooling that iterates many times depending on the proposed architecture of the model for a certain problem.

After learning features, the classification process begins. It operates on the received data from the previous computations, flatten happens to the output and then it passes to a fully connected layer and it ends by SoftMax for classification [43].

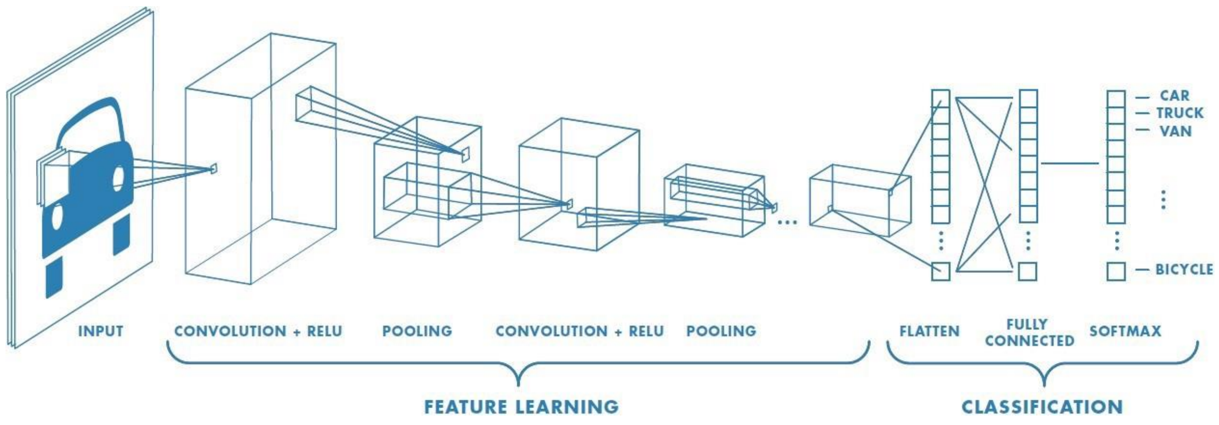


Figure 21:The architecture of a convolutional neural network [50]

7.1.3. Recurrent neural networks

“A recurrent neural network (RNN) is a type of artificial neural network which uses sequential data or time series data. These deep learning algorithms are commonly used for ordinal or temporal problems, such as language translation, natural language processing (NLP), speech recognition, and image captioning; they are incorporated into popular applications such as Siri, voice search, and Google Translate.”[51].

There are 5 types of RNNs, one to one which is a traditional neural network, one to many used for music generation, many to one used for sentiment classification , many to many for object recognition and bidirectional many to many used generally for machine translation [52]. RNNs allow previous outputs to be used as inputs while having hidden states The figure below shows the architecture of each type :

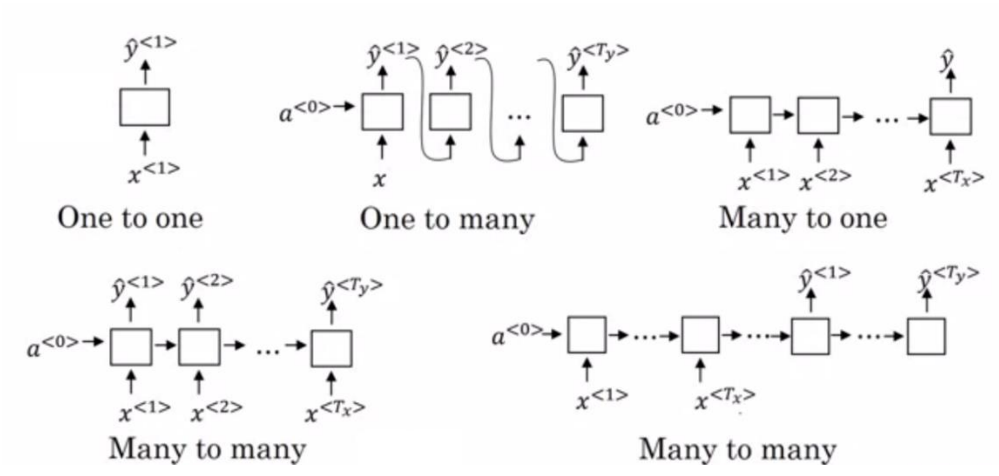


Figure 22: Types of RNNs [53]

8. Intrusion detection system in WSN using machine learning (State of the art):

As it was discussed in the previous chapter, WSNs have several constraints in term of power consumption and resources such as memory and computation ability. Integrating an IDS within a wireless sensor network is challenging and sometimes difficult to achieve, specialists have followed various approaches to solve this issue [54], by using various machine learning techniques for anomaly detection and by installing the WSN on the network in different approaches, there is a centralized IDS approach based on signature detection [55] where the detection process happens on the base station of the WSN for a better energy consumption and economy, distributed approach where the IDS is distributed across the nodes by taking advantage of the distributed communication protocols layers within the WSNs to detect any compromised node. And there is a hybrid approach where it combines both strategies [54].

Since the main subject of this thesis is deep learning and machine learning, it is evident that the fundamental discussed literature is around it. Hussein et al. [56] have discussed the feasibility of deep learning and machine learning in IDSs for WSNs. Restricted Boltzmann machine was used and called it RBC-IDS, which is an unsupervised learning algorithm of the generative artificial neural network type. The datasets used are KDDCUP99 and NSL-KDD. The method consists of the N clusters with C sensor nodes in each cluster. *"In each cluster, the Cluster Head (CH) is in charge of sending the sensor directed data to the IDS, which is installed in a central server"[56].*

[57] have used NSL-KDD for his NIDS that is built using a sparce autoencoder and SoftMax regression. His proposed NIDS have performed better than his predecessors of normal and abnormal traffic classification. He has used three major layers, the first one does feature learning from preprocessed data using sparse autoencoder, the second layer classifies the derived training data from the previous layer and passes it to SoftMax regression. The last layer does classification with Self-taught learning using sparse autoencoder and SoftMax regression.

An interesting model proposed by C. Yin et al. [58] who used Recurrent Neural Network (RNN) based IDS where the approach used NSL-KDD dataset. NSL-KDD is split into a training set and a testing set. The approach is divided into three steps, the first step is data processing where the input is the training set, it consists of numericalization (Changing non-numeric features to numeric ones) and normalization (using logarithmic scaling so that the features that have a very large scope could be mapped into a range of { 0,1 }). The second step consists of training using two phases: forward propagation and back propagation. "*Forward Propagation is responsible for calculating the output values, and Back*

Propagation is responsible for passing the residuals that were accumulated to update the weights"[58].

Convolutional neural networks (CNN or ConvNet) is a very common artificial neural network that was used for various problems. Altwaijry et al [59] have deployed CNN anomaly based intrusion detection system. The used datasets are NSL-KDD and UNSW-NB15. He studied both binary classification and multi-class classification and compared the results to the literature. The architecture of the CNN is structured as three parts, the first part is the input layer that receives 11 X 11 X 1 matrix as input for NSL-KDD dataset and 14 X 14 X 1 for UNSW-NB15 dataset. The second part is hidden layers which are five convolutional layers, two pooling layers and four fully connected layers. The last part is the output layer that is a five class SoftMax layer.

[60] have implemented a deep learning algorithm for detecting network intrusions for NSL-KDD dataset, it is constructed of a simple deep neural network with an input layer, three hidden layers and an output layer, The input dimension is six and the output dimension is two, The hidden layers contain twelve, six and three neurons respectively.

[61] proposed Chaotic Flower Pollination Algorithm (CFPA) to implement intrusion detection framework for both KDDCUP and NSL-KDD, *"The approach consists of two stages: In first stage, hybridization of CFPA and CFS is proposed to select optimal feature; the second stage uses CFPA-based kernel Adatron for the SVM classifier"[61].*

Approach	Training	Testing	Dataset	Precision
RBC-IDS [56]	70%	30%	KDDCUP99	99.12%
CFPA [61]	70%	30%	KDDCUP99	99.77%
FS + DNN	85%	15%	NSL-KDD	83.21%
STL[57]	85%	15%	NSL-KDD	85.44%
SMR [57]	85%	15%	NSL-KDD	96.56%
2-class RNN [58]	85%	15%	NSL-KDD	96.91%
BCNN [59]	80%	20%	NSL-KDD	95%
BCNN-DFS [59]	80%	20%	NSL-KDD	96%
DNN [60]	85%	15%	NSL-KDD	None
CFPA [61]	85%	15%	NSL-KDD	66.89%

Figure 23: Comparison of precision score between all the models in the state of art and their testing protocol

9. Conclusion

During this chapter, we have discussed the definition of an IDS, the general architecture, methodologies of detection (anomaly, signature and specification based IDS), also we have mentioned the types of IDSs (Network based, host based and hybrid).

After that, we have defined machine learning, its basic types (supervised, unsupervised and reinforcement learning) and deep learning.

We have showed the latest approaches found in the literature, where researchers have applied different machine learning techniques, where some of them have used feature engineering like [57] where he has used three major layers, the first one does feature learning from preprocessed data using sparse autoencoder and some others created their own algorithm like [61] where he have created a Chaotic Flower Pollination (CFPA).

Chapter 3: Proposed approach and Methodology

1. Introduction

There could be various types of approaches that could take place in wireless sensor networks intrusion detection engine systems. We have noticed that there was a lack in application of deep neural networks combined with feature selection in the literature, therefor we have proposed a model in this thesis where the general concept is displayed on the figure below:

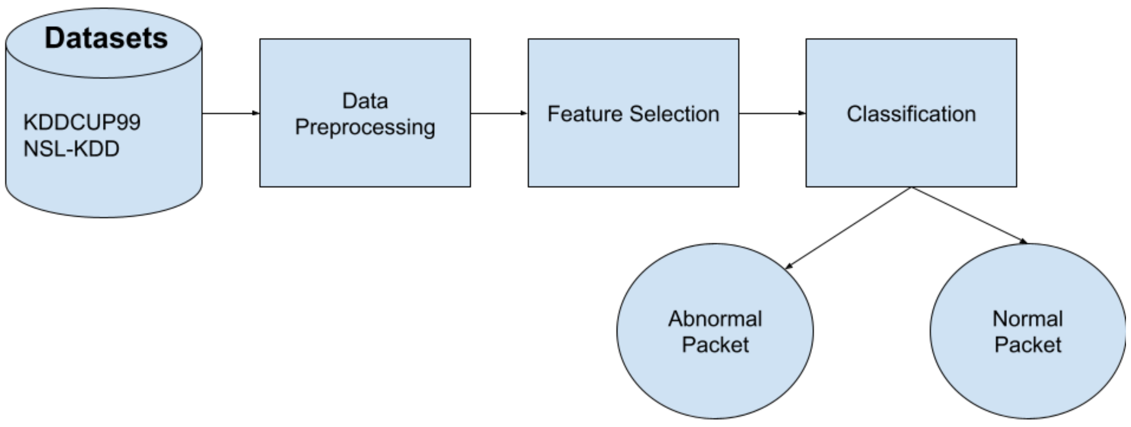


Figure 24: Model Overview of a detection engine for IDS

2. Data preprocessing

Datasets are a collection of data objects that are called events, vectors, samples or entities. They are described by a number of features that define the characteristics of the data object. Features could be categorical or numerical. But unfortunately machine learning algorithms can deal only with numerical data, that’s where the data preprocessing takes place [62]. Data preprocessing transforms unusable features into meaningful and usable variables. Label encoding is the method used for the data preprocessing step in the proposed approach.

Label Encoding

As mentioned in [63], in most datasets of machine learning or data science activities, they contain non-numerical values such as text or categorical values. As an example, let’s assume we have a dataset DT that has a feature called company name, it would have values like Company_A, Company_B, etc.

Label Encoding technique is a very simple and effective approach, it consists of converting every value (non-numerical) in a column to a number by assigning each category to a unique integer value as it is shown in the figure below from KDDCU99 dataset:

Table 1: The difference after applying LabelEncoding

Protocol_Type	Service	Flag
TCP	HTTP	SF

Protocol_Type	Service	Flag
1	24	9

3. Feature selection

Having too many data variables could result in a slow model, it could learn from irrelevant data which increases the inaccuracy rate. Feature selection is the process of reducing the input variables inserted into the model by using only relevant data and getting rid of noise in data, which helps avoid overfitting and it results a better accuracy rate, reduce training time. When more features get dropped the faster the model gets.

In the literature, there are two types of features selection, supervised and unsupervised. In this thesis the supervised method was applied. The supervised learning uses output label class for feature selection. There are three method categories: intrinsic method, wrapper method or filter method.

Filter method is based on the attempt of dropping certain features of the dataset based on the relation with the output or in other words how they are correlating with the output.

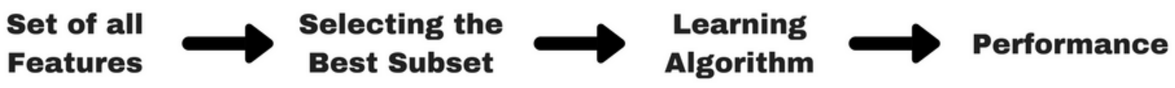


Figure 25: Architecture of filter method [64]

Wrapper method splits data into subsets and trains the model using them. Based on the output of the model the algorithm adds or subtracts features and retrains the model again.

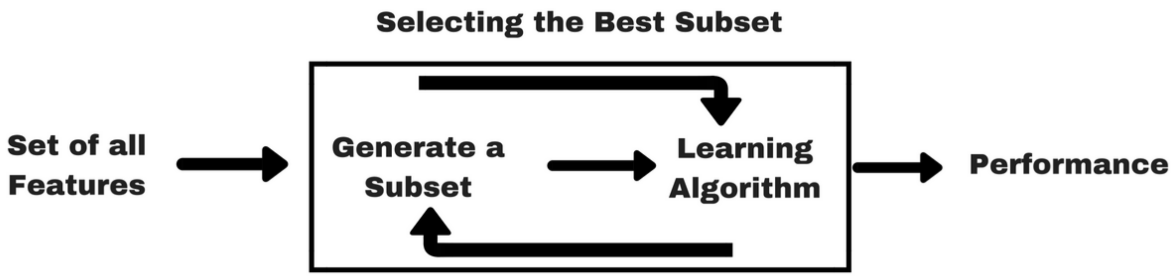


Figure 26: Wrapper method [64]

The intrinsic method combines the qualities of both methods filter and wrapper to create the best subset. It trains the model and checks the accuracy of the subsets and selects the most accurate one.

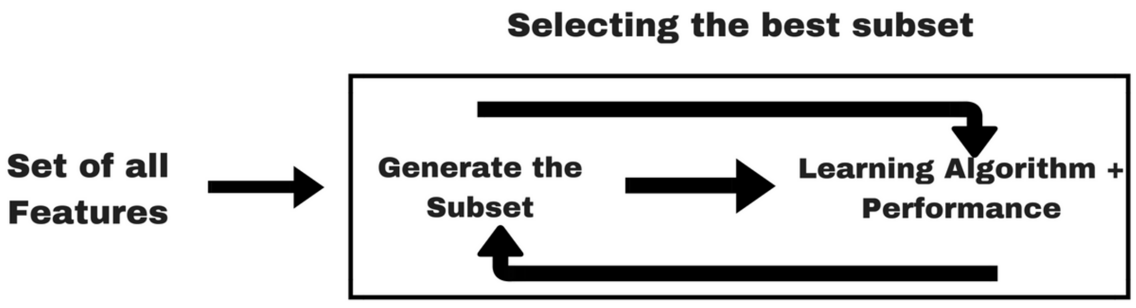


Figure 27: Architecture of intrinsic [64]

3.1. Feature Importance

It is a technique used to assign a certain score to features of the dataset to a predictive model in order to indicate the relative importance of every single feature while computing a prediction.

This technique helps for a better understanding of the data, obtain good results with the model and reduce the number of features injected to the model.

In this thesis XGBoost library was used in order to calculate feature importance of the dataset. It’s a very famous library that uses gradient boosting decision tree algorithm which is an intrinsic method.

To understand better what is Gradient Boosting we need to comprehend what’s bagging, boosting, gradient boosting.

3.2. Bagging (or Bootstrap aggregation)

It’s a machine learning technique that consists of assembling a great number of algorithms with weak performances individually to create a strong and efficient algorithm. Weak algorithms are called “Weak learners”, and the result obtained from bagging is “Strong learner”[65]

The weak learners could have various types and various performances but most importantly they should be separate and independent between each other.

Bagging technique aggregates the predictions to select the best prediction among them. It is usually applied to decision trees, where it significantly raises the stability of models in improving accuracy and reducing variance, which eliminates the challenge of overfitting.

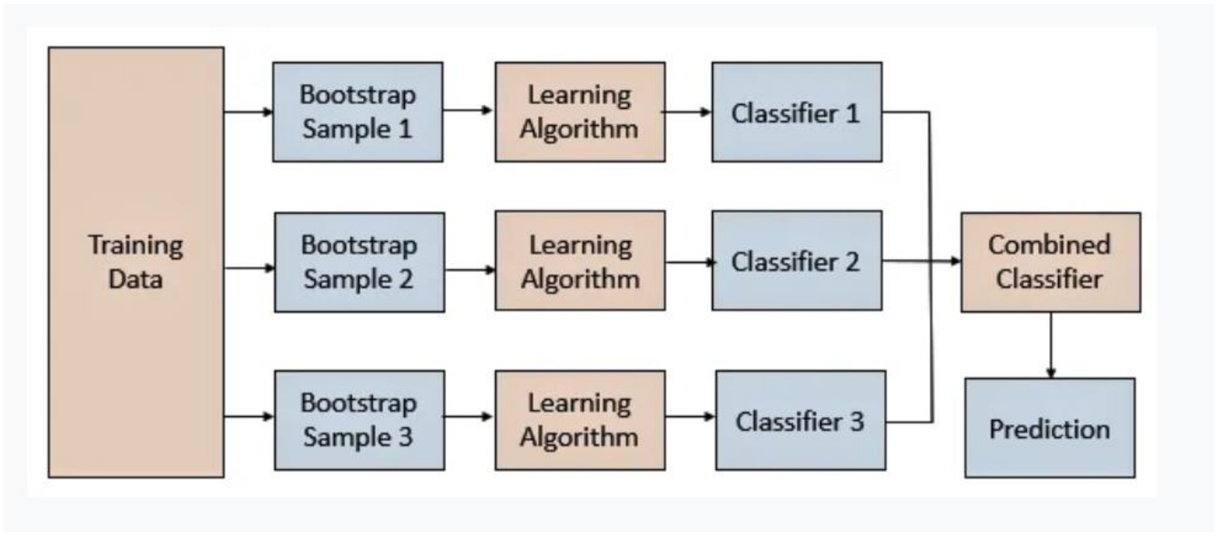


Figure 28: Bagging mechanism [66]

Bagging is composed of two parts (Bootstrapping and aggregation) :

- Boostrapping

“It is a sampling method, where a sample is chosen out of a set, using the replacement method. The learning algorithm is then run on the samples selected.”[67]

- Aggregation

The aggregation can be done based on the total number of outcomes or the probability of predictions derived from the bootstrapping of every model in the procedure.

3.3. Boosting

It is an ensemble meta-algorithm for primarily reducing bias, and also variance [68] in supervised learning, it also converts weak learners to strong learners. It shares the same principle with bagging which is the usage of multiple weak learners, and by combining the weak learners in series to achieve a strong learner from many sequentially connected weak learners.

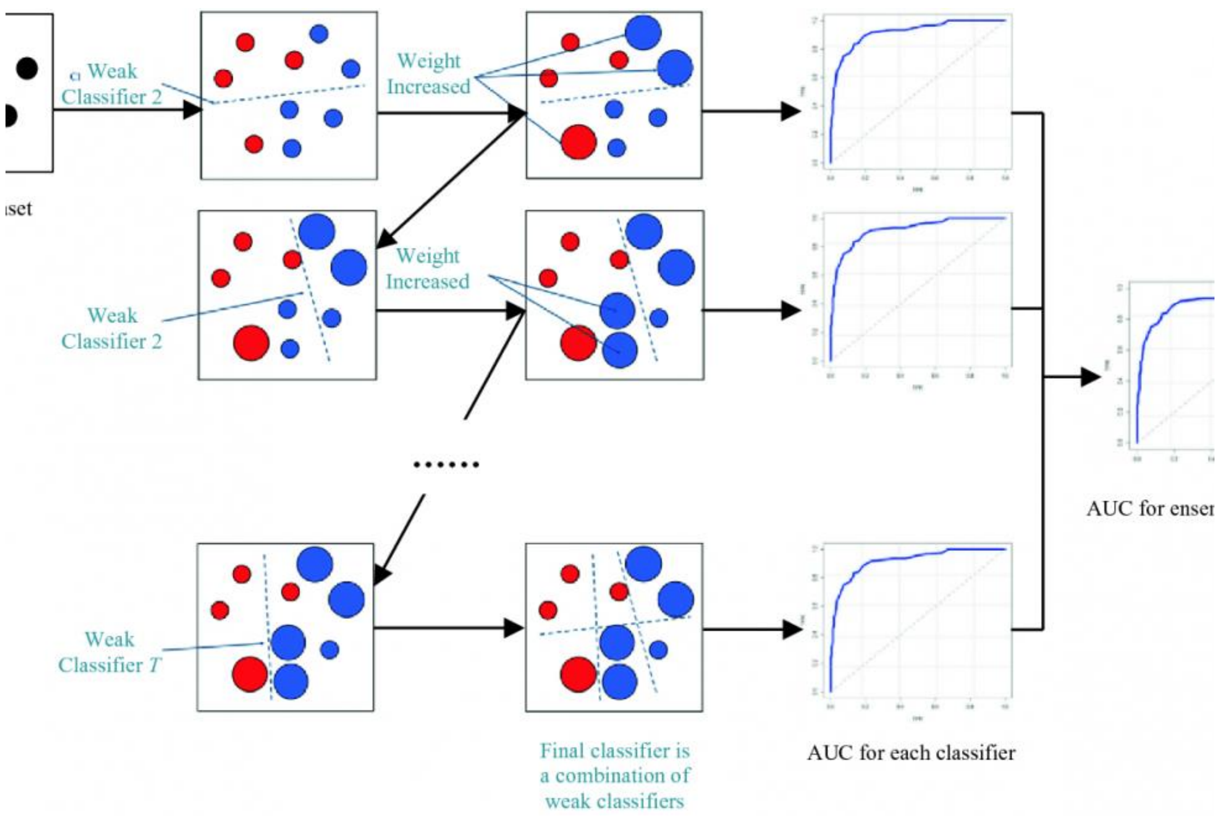


Figure 29: General concept of Boosting[69]

3.4. Gradient Boosting

It is one of the most powerful techniques used for predictive models. Gradient boosting is an ensemble learning technique, used for classification and regression problems [70], it has a lot in common with bagging which is creating a strong learner from multiple weak learners. however the difference is that the algorithms are dependent to each other, each weak learner is trained to correct the previous errors of the previous weak learners, by fitting to the weak learner the residuals from the previous step so that the model improves.

4. Classification

Classification is the most important part of all the proposed approaches, because the goal is to classify (detect) packets into normal and malicious, therefore, Binary classification is used.

4.1. Binary classification

It’s the attempt of classifying elements into two groups on the basis of classification rule, which is in the case of this thesis classifying “normal” traffic and “abnormal” traffic.

The question that could be asked in this part is that it could be much better for a wireless sensor network IDS to detect the nature of the attack attempted and it’s details , which could be performed by implementing a multi-classification model that could distinguish each attack on its own. It is correct to assume such a hypothesis, and it would be beneficial in terms of instant response to the intrusions detected.

Since we are in the field of wireless sensor networks, there are plenty of constraints that must respected which was discussed in the first chapter, such as power consumption constraint, low computing capacity and low memory performance, which means that the more complex and energy/memory/performance consuming IDS is installed within the network the more it affects the quality of the global system . It could result a high latency communication and non-accurate network in terms of quality of service.

Thus, binary classification is the ideal solution for such systems. Another IDS could be installed on an external server from the WSN that takes responsibility for distinguishing the nature of attacks performed on the network without affecting the global functioning of the entire network and respecting the WSN constraints.

Remark :

For each dataset the label column is set to “0” in case of normal traffic and “1” in case of any attack of any category.

4.2. DNN (Deep neural networks)

It is also called as Deep Feedforward Neural Networks (DFFNN), it’s a sophisticated version on the feed forward neural network but with more hidden layers in the middle *“A deep neural network (DNN) with multiple hidden layers can learn more higher-level, abstract representations of the input” [49].* Therefore DNN is more accurate compared to feedforward networks.

The neural network needs to learn all the time to solve tasks in a more qualified manner or even to use various methods to provide a better result. When it gets new information in the system, it learns how to act accordingly to a new situation.

It is used in this thesis as a classificatory model for classifying the normal and the abnormal traffic.

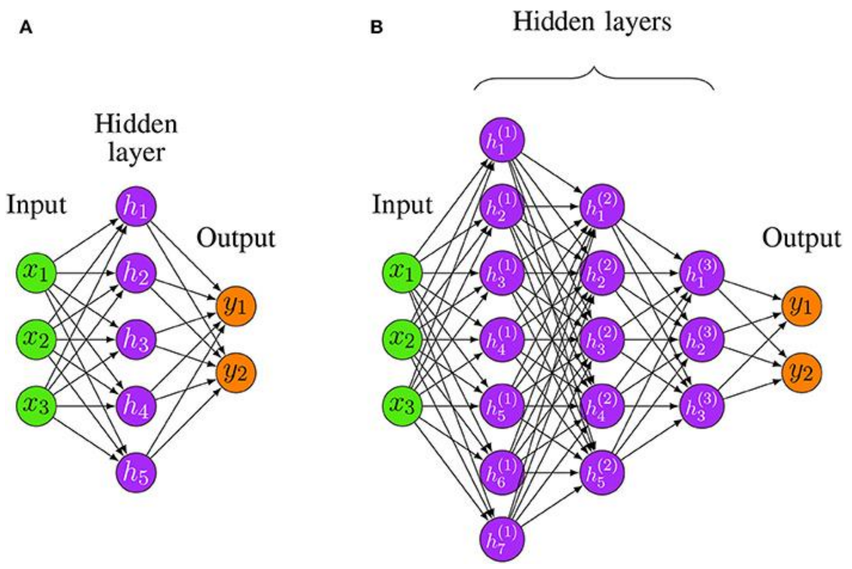

A
Input
Hidden layer
Output
x_1
x_2
x_3
h_1
h_2
h_3
h_4
h_5
y_1
y_2
B
Hidden layers
Input
Output
$h_1^{(1)}$ $h_2^{(1)}$ $h_3^{(1)}$ $h_4^{(1)}$ $h_5^{(1)}$ $h_6^{(1)}$ $h_7^{(1)}$
$h_1^{(2)}$ $h_2^{(2)}$ $h_3^{(2)}$ $h_4^{(2)}$ $h_5^{(2)}$
$h_1^{(3)}$ $h_2^{(3)}$ $h_3^{(3)}$

Figure 30: The figure A shows feedforward neural network and B deep feedforward neural network [43]

The architecture of the proposed DNN:

Algorithm 1 Architecture

```
1: $model = Sequential()$
2: $model.add(Dense(134, activation =' relu', input_{s}hape = (4,)))$
3: $model.add(Dropout(0.2))$
4: $model.add(Dense(60, activation =' relu'))$
5: $model.add(Dropout(0.2))$
6: $model.add(Dense(26, activation =' relu'))$
7: $model.add(Dropout(0.2))$
8: $model.add(Dense(13, activation =' softmax'))$
9: $model.compile(loss =' categorical_{c}rossentropy', optimizer ='$
   $adam', metrics = ['accuracy'])$
```

Figure 31: DNN Algorithm

It is composed of one input layer, two hidden layers and an output layer. Input layer contains 134 nodes with ReLU activation function, first hidden layer contains 60 nodes with ReLU activation function, second hidden layer contains 26 nodes with ReLU activation function and the output layer contains 13 nodes with SoftMax activation function. Each layer other than the output layer is followed by a dropout of 0.2 rate to help prevent overfitting.

Batch size is equal to 10, it refers to the number of training examples utilized in one iteration.

The number of epochs is equal to 30, it indicates the number of passes of the entire training dataset the machine learning algorithm has completed.

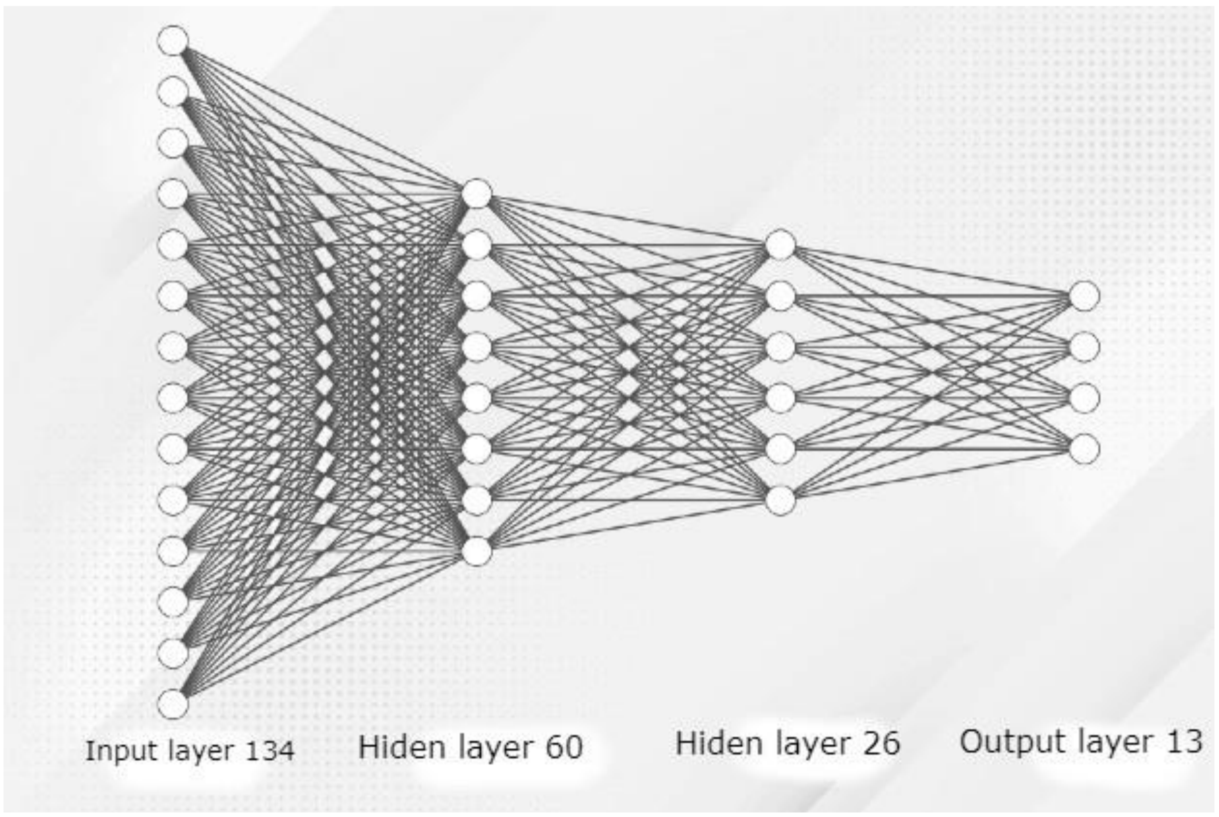


Figure 32: Graphical representation of the architecture of the DNN

4.2.1. Activation Function

An activation function of a node in a neural network is a function that defines the output of that node, given an input or set of inputs. The function returns 0 if it receives any negative input, but for any positive value x it returns that value back [71].

$$f(x) = \max(0, x)$$

There are many activation functions, ReLu and SoftMax are used in this model.

4.2.1.1. ReLu

It’s the most common activation function in the literature of hidden layers due its implementation simplicity. It’s less susceptible to vanishing gradient that prevent deep models from being trained. If the value of the input x is negative it returns 0.0 or else it returns the value of the input x.

$$f(x) = \max(0.0, x)$$

4.2.1.2. SoftMax

It’s an activation function for output layers to normalize the output of a network to a probability distribution over predicted output classes. It takes as input a vector z of K real numbers. After applying SoftMax all of the outputs are in the interval of {0 , 1 }.

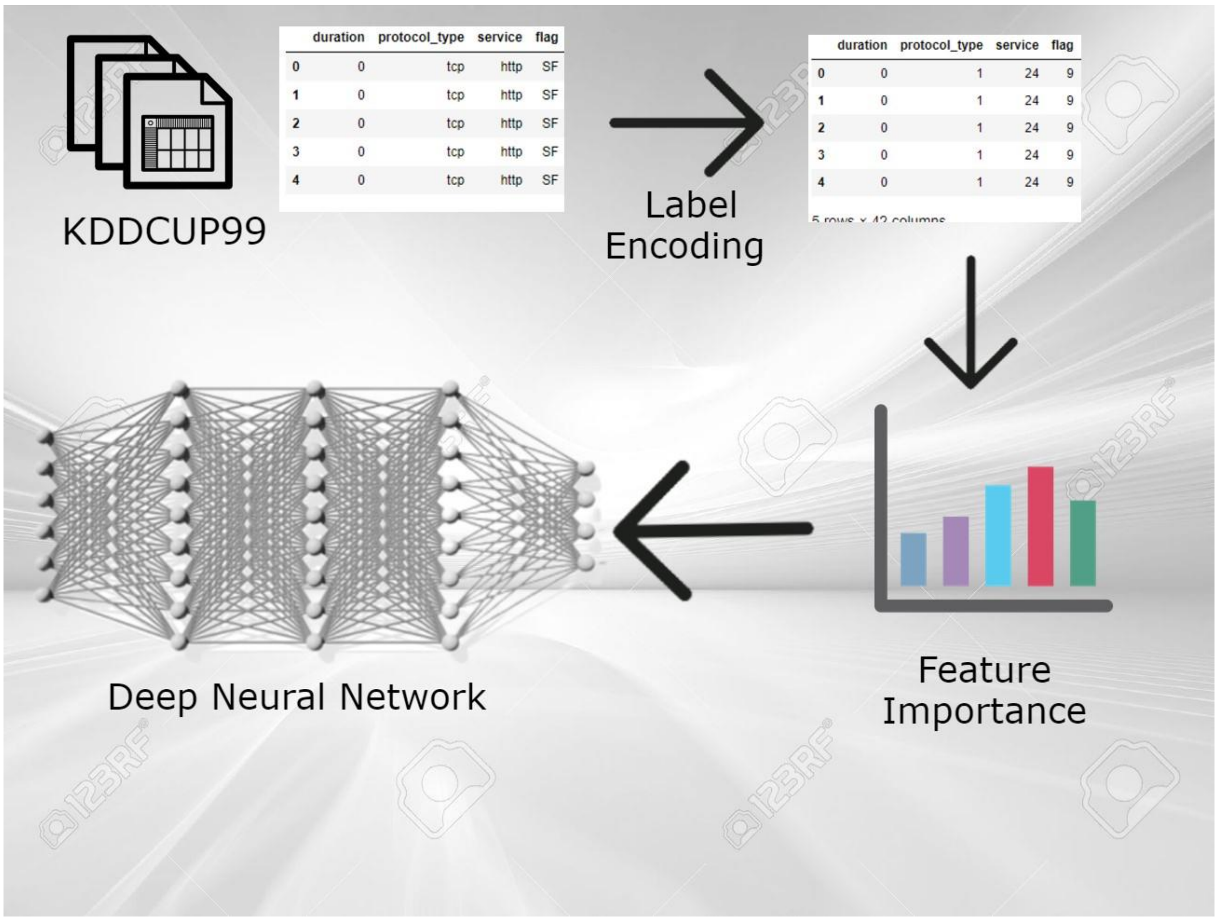
$$f(x) = \frac{e^{x}}{\sum_{i=1}^{K} e^{x_i}}$$

Initially, the datasets used are KDDCUP99 and NSL-KDD their characteristics are explained in detail in the next chapter.

5. Application on KDDCUP99

In this section we are going to discuss the data results of each phase of the mentioned approach for the KDDCUP99 dataset: Feature Selection using importance scores + DNN + optimization.

In this The figure below shows the architecture used for this approach:



	duration	protocol_type	service	flag
0	0	tcp	http	SF
1	0	tcp	http	SF
2	0	tcp	http	SF
3	0	tcp	http	SF
4	0	tcp	http	SF

	duration	protocol_type	service	flag
0	0	1	24	9
1	0	1	24	9
2	0	1	24	9
3	0	1	24	9
4	0	1	24	9

5 rows × 42 columns

Figure 33: Process of passing KDDCUP99 through the model

After the phase of **Feature Selection** the attributes selected using XGBoost gradient boosting decision tree algorithm are :

‘logged_in’, ‘protocol_type’, ‘dst_host_count’, ‘src_bytes’, ‘wrong_fragment’, ‘srv_serror_rate’, ‘dst_host_rerror_rate’, ‘dst_host_srv_diff_host_rate’, ‘dst_host_serror_rate’, ‘service’, ‘dst_host_srv_serror_rate’, ‘dst_host_diff_srv_rate’, ‘same_srv_rate’, ‘num_compromised’, ‘hot’, ‘dst_host_same_src_port_rate’, ‘dst_bytes’, ‘urgent’, ‘duration’, ‘dst_host_srv_rerror_rate’, ‘flag’, ‘rerror_rate’, ‘dst_host_srv_count’, ‘dst_host_same_srv_rate’, ‘count’, ‘srv_count’, ‘num_root’, ‘diff_srv_rate’, ‘root_shell’, ‘su_attempted’, ‘num_file_creations’, ‘serror_rate’, ‘srv_diff_host_rate’, ‘srv_rerror_rate’, ‘num_shells’, ‘is_guest_login’ .

Their count is 36 out of 41 columns which means there is feature reduction.

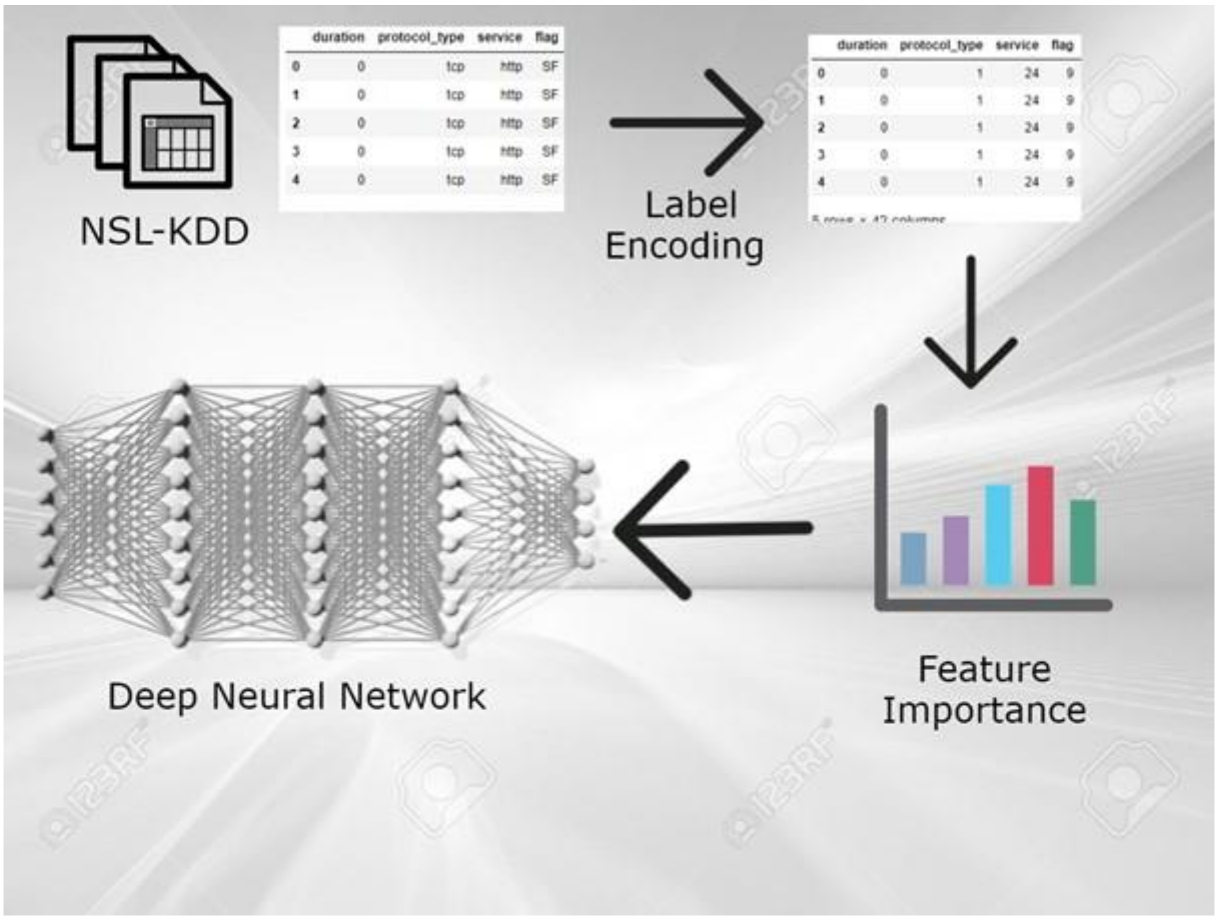
After the previous process, the only mentioned features above are kept in the dataset, the others have been dropped.

The DNN receives the dataset of 36 columns in the input layer and then runs it through the neural network nodes for classification.

6. Application on NSL-KDD

In this section we are going to discuss the data results of each phase of the mentioned approach for the KDDCUP99 dataset: Feature Selection using importance scores + DNN + optimization.

The figure below demonstrates the architecture of this approach:



	duration	protocol_type	service	flag
0	0	tcp	http	SF
1	0	tcp	http	SF
2	0	tcp	http	SF
3	0	tcp	http	SF
4	0	tcp	http	SF

	duration	protocol_type	service	flag
0	0	1	24	9
1	0	1	24	9
2	0	1	24	9
3	0	1	24	9
4	0	1	24	9

5 rows × 42 columns

Figure 34: Process of passing NSL-KDD through the model

After the phase of **Feature Selection** the attributes selected using XGBoost gradient boosting decision tree algorithm are :

‘flag’, ‘dst_host_count’, ‘src_bytes’, ‘num_failed_logins’, ‘dst_host_srv_count’, ‘dst_host_rerror_rate’, ‘dst_host_srv_rerror_rate’, ‘dst_host_same_srv_rate’, ‘service’, ‘dst_host_srv_serror_rate’, ‘is_guest_login’, ‘srv_serror_rate’, ‘duration’, ‘protocol_type’, ‘srv_diff_host_rate’, ‘dst_host_serror_rate’, ‘srv_count’, ‘urgent’, ‘logged_in’, ‘dst_host_diff_srv_rate’, ‘su_attempted’, ‘dst_host_srv_diff_host_rate’, ‘dst_host_same_src_port_rate’, ‘serror_rate’, ‘dst_bytes’, ‘rerror_rate’, ‘count’, ‘same_srv_rate’, ‘land’, ‘num_root’, ‘srv_rerror_rate’, ‘is_host_login’

The count of selected features is 32 out of 41 columns which means there is feature reduction. Each attribute has its own importance score.

Thus the DNN input is set to 32 The DNN runs it through the neural network nodes for classification.

7. Conclusion

In this chapter, we have introduced our methodology that consists of three major phases, Data Preprocessing, Feature Selection using XGBoost, Classification using a four layer DNN (input, two hidden layers and output).

Chapter 4: Experiments and results

1. Introduction

In this chapter, we are going to discuss the environment used, libraries and APIs, datasets details, evaluation metrics, comparison of the results with the state of the art and a simulation scenario.

2. Environment

2.1. Anaconda

"Anaconda is a package manager, an environment manager, a Python/R science distribution and a collection of over +7500 open source packages"[72]. It is used for scientific computing like Data science, Machine learning applications, predictive analytics and many other fields.

This package manager makes installing and using libraries such (Scikit-Learn, Numpy, …etc) easy and simply, which grants a stable environment with less time consumption while developing a project.

It could be installed for Windows, Linux and MacOS, in our case It was installed in windows operating system following these steps[73]:

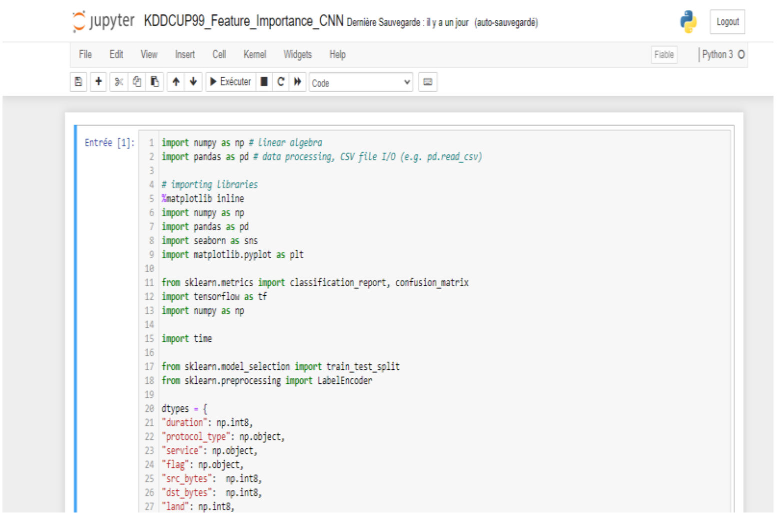
1- Downloading the Anaconda installer : (https://www.anaconda.com/download/#windows)

2- Verify data integrity with SHA-256

3- Following the steps on the Anaconda installer window

For installing any package needed a simple command `conda install Package_name`

2.2. Jupyter Notebook


jupyter KDDCUP99_Feature_Importance_CNN Dernière Sauvegarde : il y a un jour (auto-sauvegardé)
Logout
File Edit View Insert Cell Kernel Widgets Help
Fiable
Python 3
Exécuter
Code
Entrée [1]:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# importing libraries
%matplotlib inline
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.metrics import classification_report, confusion_matrix
import tensorflow as tf
import numpy as np

import time

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

dtypes = {
"duration": np.int8,
"protocol_type": np.object,
"service": np.object,
"flag": np.object,
"src_bytes":  np.int8,
"dst_bytes":  np.int8,
"land": np.int8,
```

"The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more."[74]

It is used as a testing environment of the written code of the project.

3. Libraries and APIs

3.1. Keras

"Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages. It also has extensive documentation and developer guides."[75]

Keras has proven over the years that it's the most used framework, it is among the top 5 winning teams on Kaggle, it's easy to learn, flexible and customizable.

The most fascinating fact about Keras is that it is used by some of the most famous scientific organizations such as CERN, NASA, NIH and many other organizations.

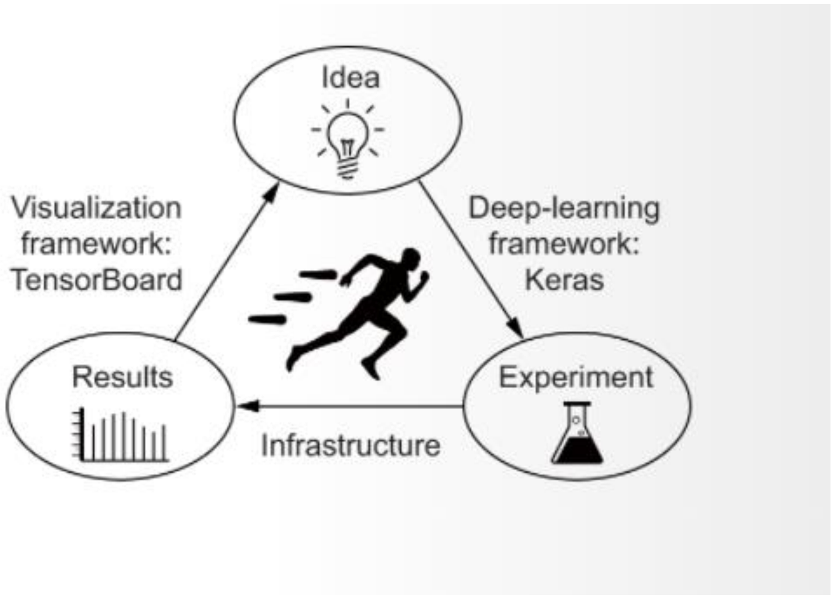

Idea
Visualization framework: TensorBoard
Deep-learning framework: Keras
Results
Experiment
Infrastructure

Figure 35: Keras Framework work process

3.2. Scikit-Learn

"Scikit-learn is an open source machine learning library that supports supervised and unsupervised learning. It also provides various tools for model fitting, data preprocessing, model selection and evaluation, and many other utilities."[76]

This library provides a large number of built-in machine leaning algorithms and models such as LabelEncoder for label encoding (explained later on this article), confusion_matrix to calculate the true/false negatives and true/false positives, train_test_split to split the data frame obtained from the dataset to a training set and to a testing set, and many other useful models.

3.3. Numpy

"NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more."[77]

It's an open source library that is used widely in Python programming and almost in all fields of science and engineering. It contains multidimensional array and matrix data structures, and can perform various mathematical operations on them.

3.4. Matplotlib

"Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python." [78], It's a visualization library that offers the ability to display data, results in various shapes and diagrams.

3.5. Pandas

“pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.”[79]

It provides a high performance, easily used data structures and data analysis tools, It’s the library that is used to manipulate datasets data frames

3.6. XGBoost Library

“XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way.” [80]

3.7. TensorFlow 2.3

“TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.”[81]

TensorFlow is a framework that provides an excellent architecture support which allows easy deployment of computations across a variety of platforms ranging from desktops to clusters of servers, mobiles, and edge devices. It was developed by engineers and researchers from Google AI Organization.

4. Datasets

A dataset is a collection of related, discrete items of related data represented by rows and columns. Columns are also called features which represent variables that could be numerical, categorical …etc. Rows are the instance of the variables (columns).

In this thesis two datasets were used, KDDCUP99 [82] and NSL-KDD [83].

4.1. KDDCUP99

KDDCUP99 is a very commonly used dataset among security experts and specialists, most papers on the field of intrusion detection systems have tested their models with this particular dataset due to the efficiency of it. It was proposed by Stolfo et al. [84] it was built based on the data captured in DARPA’98 IDS evaluation program [85].

“*DARPA’98 is about 4 gigabytes of compressed raw (binary) tcpdump data of 7 weeks of network traffic, which can be processed into about 5 million connection records, each with about 100 bytes. The two weeks of test data have around 2 million connection records. KDD training dataset consists of approximately 4,900,000 single connection vectors each of which contains 41 features and is labeled as either normal or an attack, with exactly one specific attack type*”[86].

KDDCUP99 contains 4898431 rows and 42 columns (features), features could be classified as :

- **Basic features:**

  They are all the features or attributes extracted from TCP/IP connection

- **Traffic features:**

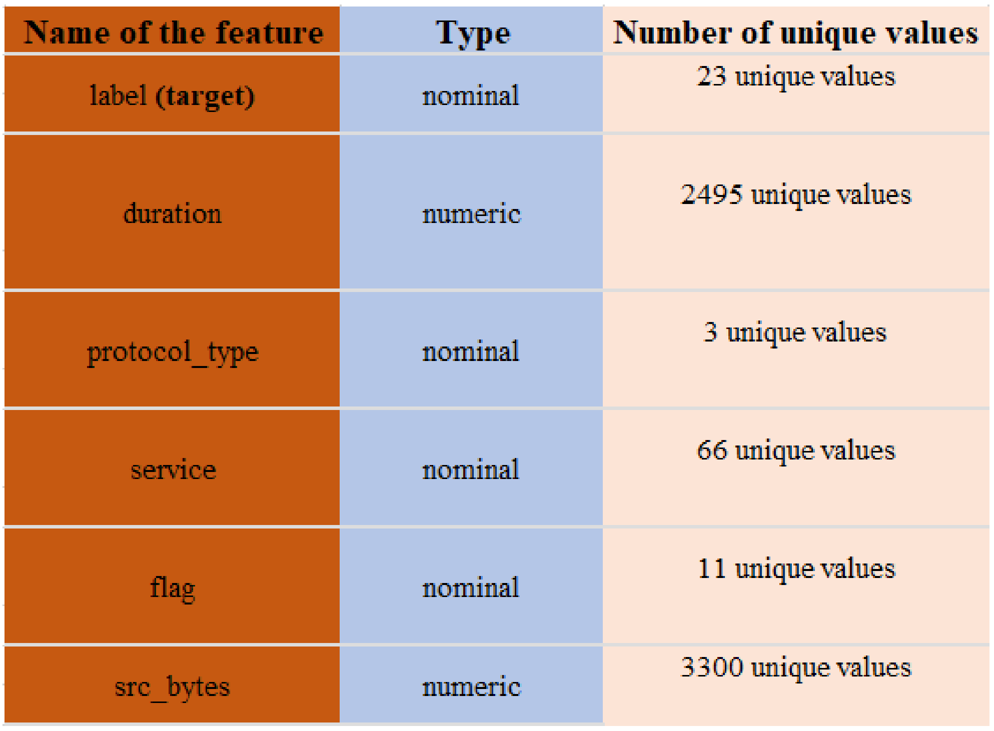
  It includes the attributes computed within a time frame, they are divided into two groups:
  - **Same host** features: *“examine only the connections in the past 2 seconds that have the same destination host as the current connection, and calculate statistics related to protocol behavior, service, etc.”[86]*

- **Same service** features: *“examine only the connections in the past 2 seconds that have the same service as the current connection”[86]*

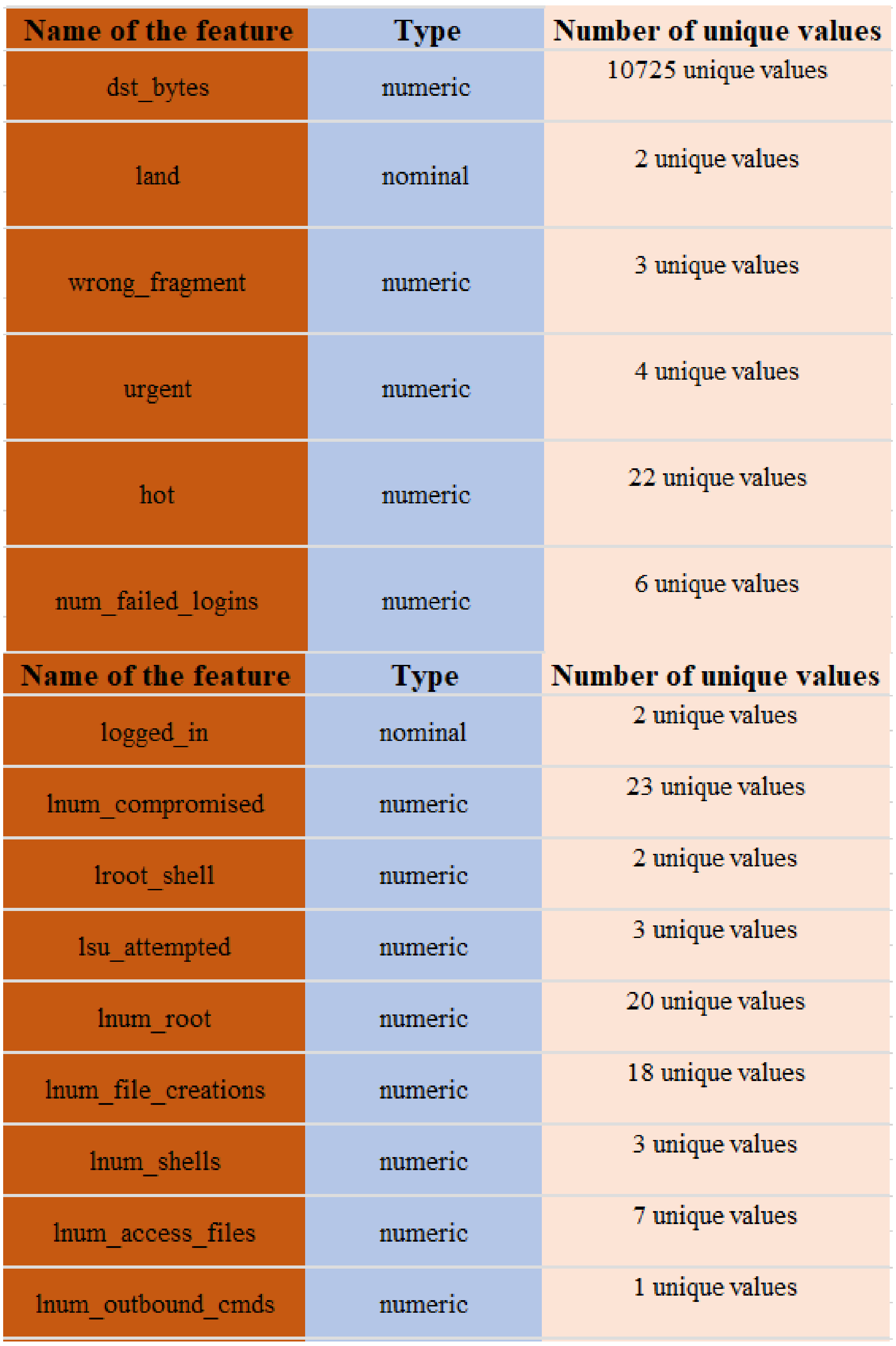
The above feature types mentioned are called **Time-based features,** the inconvenient with this type is that the time frame is only 2 seconds, some attacks may take a larger interval of time, thus intrusion patterns won’t be recognized. The solution of this vulnerability is by calculating not the time frame (x seconds) but based on recalculating the connection window of 100 connections.

- **Content features:**

Some kinds of attacks like user-to-root or remote-to-local don’t have patterns that could be identified by intrusion detection system, thus some features are needed to determine the suspicious behavior in the traffic such as num_failed_logins that counts the number of failed login attempts.
The figures below show the features, type and number of unique values of KDDCUP99:

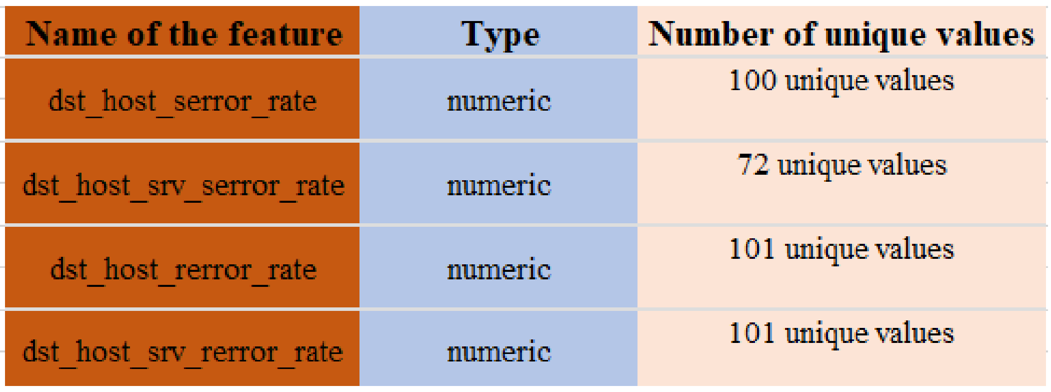


Name of the feature	Type	Number of unique values
label **(target)**	nominal	23 unique values
duration	numeric	2495 unique values
protocol_type	nominal	3 unique values
service	nominal	66 unique values
flag	nominal	11 unique values
src_bytes	numeric	3300 unique values



Name of the feature	Type	Number of unique values
dst_bytes	numeric	10725 unique values
land	nominal	2 unique values
wrong_fragment	numeric	3 unique values
urgent	numeric	4 unique values
hot	numeric	22 unique values
num_failed_logins	numeric	6 unique values
logged_in	nominal	2 unique values
lnum_compromised	numeric	23 unique values
lroot_shell	numeric	2 unique values
lsu_attempted	numeric	3 unique values
lnum_root	numeric	20 unique values
lnum_file_creations	numeric	18 unique values
lnum_shells	numeric	3 unique values
lnum_access_files	numeric	7 unique values
lnum_outbound_cmds	numeric	1 unique values

Name of the feature	Type	Number of unique values
is_host_login	nominal	1 unique values
is_guest_login	nominal	2 unique values
count	numeric	490 unique values
srv_count	numeric	470 unique values
serror_rate	numeric	92 unique values
srv_serror_rate	numeric	51 unique values
rerror_rate	numeric	77 unique values
srv_rerror_rate	numeric	51 unique values
same_srv_rate	numeric	99 unique values
diff_srv_rate	numeric	78 unique values
srv_diff_host_rate	numeric	64 unique values
dst_host_count	numeric	256 unique values
dst_host_srv_count	numeric	256 unique values
dst_host_same_srv_rate	numeric	101 unique values
dst_host_diff_srv_rate	numeric	101 unique values
dst_host_same_src_port_rate	numeric	101 unique values
dst_host_srv_diff_host_rate	numeric	65 unique values



Name of the feature	Type	Number of unique values
dst_host_serror_rate	numeric	100 unique values
dst_host_srv_serror_rate	numeric	72 unique values
dst_host_rerror_rate	numeric	101 unique values
dst_host_srv_rerror_rate	numeric	101 unique values

Figure 36: KDDCUP99 columns description

In the **"label"** feature we find classification of what packet is "normal" and what is considered as an attack "X", There are mainly 4 attack categories:

- **Denial of Service Attack (DoS):**

  The attempt of allocating the available resources in order to make them unavailable and unreachable to the legitimate users, in this dataset attacks of this category are:
  Back – Land- Neptune – Pod- Smurf – Teardrop.

- **User to Root Attack (U2R):**

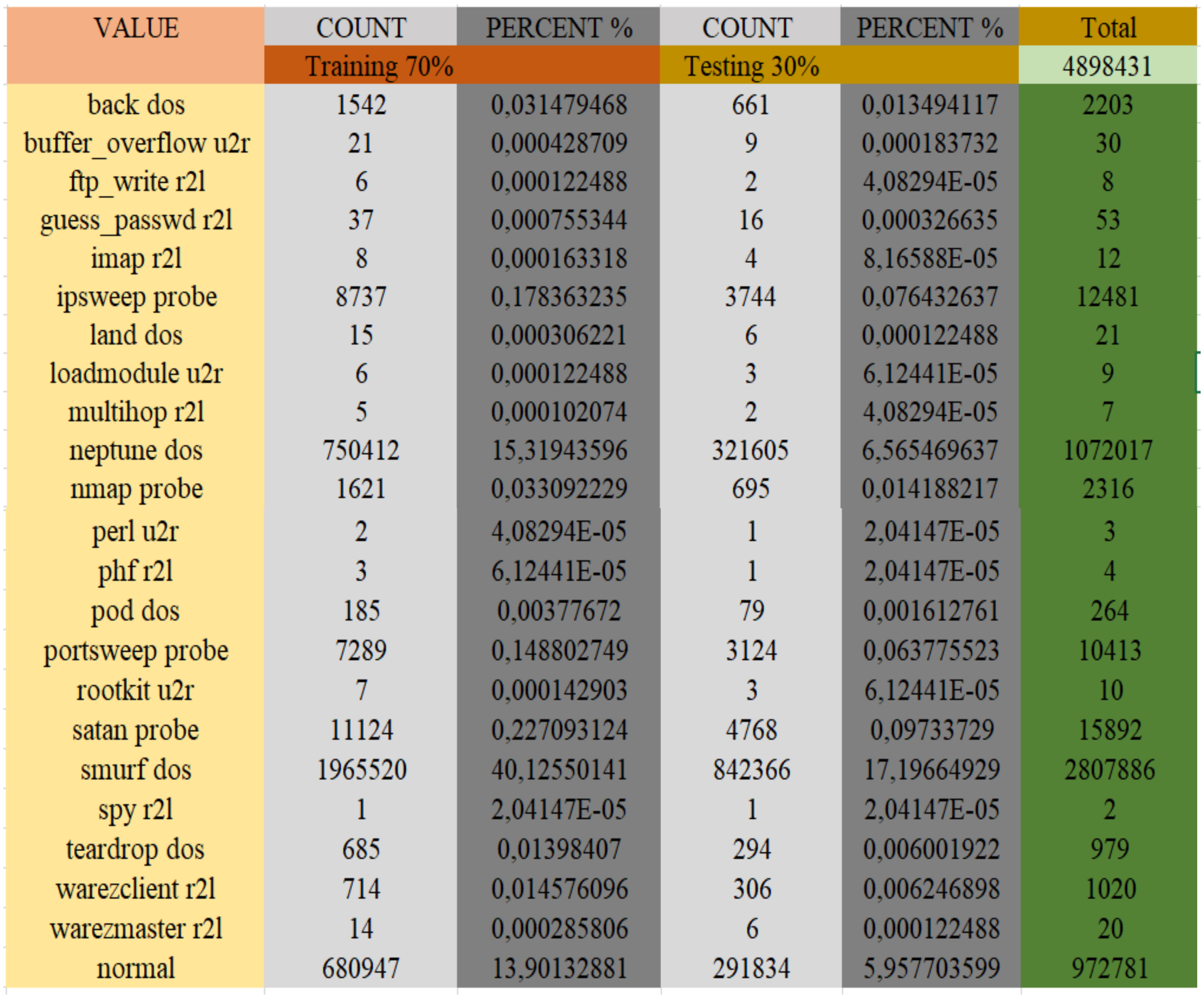
  From its name meaning, the attacker attempts to gain root access by exploiting the vulnerabilities within the network, starting from the user's ordinary system access to full root access. In this dataset attacks of this category are:
  Buffer overflow – LoadModule – Perl – Rootkit.

- **Remote to Local Attack (R2L):**

  In case the attacker has the ability to communicate with a node in the network, he tries to exploit certain vulnerabilities to grant himself the privilege of accessing the network as a local user and not as remote one.
  Here are some of the attacks in the dataset from this category:
  FTP write – Guess Password – Imap – Multihop – Phf – Spy – WarezClient – WarezMaster.

- **Probing Attack:**

It’s the attacker’s first step to comprehend the network, which is gathering information about the topology and the infrastructure of the network and the security controls installed.
In this dataset we have:
IPSweep – Nmap – PortSweep – Satan.



VALUE	COUNT	PERCENT %	COUNT	PERCENT %	Total
	Training 70%		Testing 30%		4898431
back dos	1542	0,031479468	661	0,013494117	2203
buffer_overflow u2r	21	0,000428709	9	0,000183732	30
ftp_write r2l	6	0,000122488	2	4,08294E-05	8
guess_passwd r2l	37	0,000755344	16	0,000326635	53
imap r2l	8	0,000163318	4	8,16588E-05	12
ipsweep probe	8737	0,178363235	3744	0,076432637	12481
land dos	15	0,000306221	6	0,000122488	21
loadmodule u2r	6	0,000122488	3	6,12441E-05	9
multihop r2l	5	0,000102074	2	4,08294E-05	7
neptune dos	750412	15,31943596	321605	6,565469637	1072017
nmap probe	1621	0,033092229	695	0,014188217	2316
perl u2r	2	4,08294E-05	1	2,04147E-05	3
phf r2l	3	6,12441E-05	1	2,04147E-05	4
pod dos	185	0,00377672	79	0,001612761	264
portsweep probe	7289	0,148802749	3124	0,063775523	10413
rootkit u2r	7	0,000142903	3	6,12441E-05	10
satan probe	11124	0,227093124	4768	0,09733729	15892
smurf dos	1965520	40,12550141	842366	17,19664929	2807886
spy r2l	1	2,04147E-05	1	2,04147E-05	2
teardrop dos	685	0,01398407	294	0,006001922	979
warezclient r2l	714	0,014576096	306	0,006246898	1020
warezmaster r2l	14	0,000285806	6	0,000122488	20
normal	680947	13,90132881	291834	5,957703599	972781

Figure 37: Test and Train split results of KDDCUP99

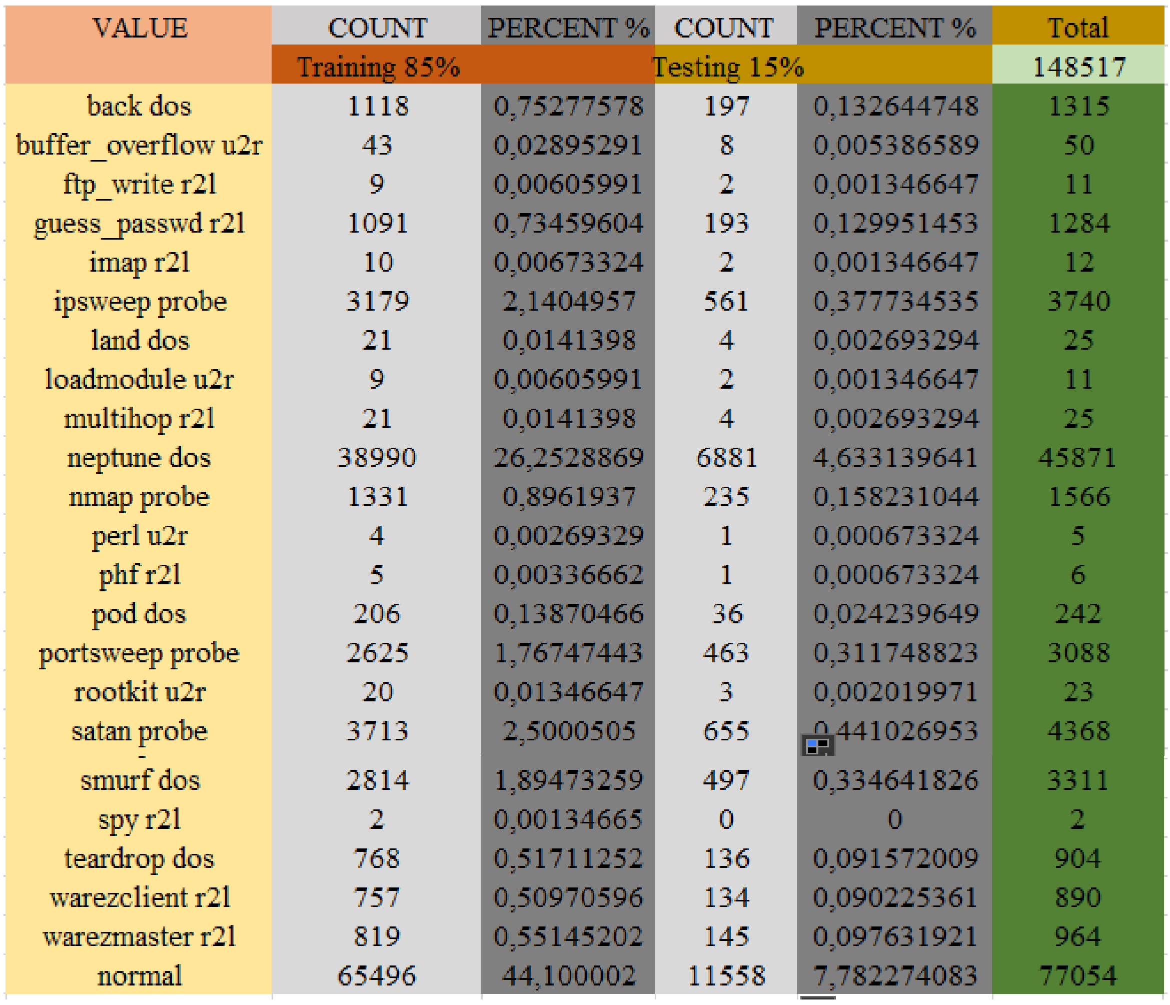
Figure 37 shows details about training and testing split percentage which is 70% 30% for KDDCUP99 and shows attacks categories, their count, their percentage within the dataset and the total of each attack.

4.2. NSL-KDD

It's a newer version of KDDCUP99 proposed by [86] in 2009, McHugh [87] has discussed the problems with KDDCUP99 (that are still in the NSL-KDD) since there is a lack of public datasets thus it's not considered as a perfect representative of a real existing network.

It contains the same number of features (42 columns) with the same names and type of data, 148517 rows.

The split of the dataset is based from the original files from the datasets website which are "KDDTrain+.txt" that is used for training and "KDDTest+.txt" that is used for testing.



VALUE	COUNT	PERCENT %	COUNT	PERCENT %	Total
	Training 85%		Testing 15%		148517
back dos	1118	0,75277578	197	0,132644748	1315
buffer_overflow u2r	43	0,02895291	8	0,005386589	50
ftp_write r2l	9	0,00605991	2	0,001346647	11
guess_passwd r2l	1091	0,73459604	193	0,129951453	1284
imap r2l	10	0,00673324	2	0,001346647	12
ipsweep probe	3179	2,1404957	561	0,377734535	3740
land dos	21	0,0141398	4	0,002693294	25
loadmodule u2r	9	0,00605991	2	0,001346647	11
multihop r2l	21	0,0141398	4	0,002693294	25
neptune dos	38990	26,2528869	6881	4,633139641	45871
nmap probe	1331	0,8961937	235	0,158231044	1566
perl u2r	4	0,00269329	1	0,000673324	5
phf r2l	5	0,00336662	1	0,000673324	6
pod dos	206	0,13870466	36	0,024239649	242
portsweep probe	2625	1,76747443	463	0,311748823	3088
rootkit u2r	20	0,01346647	3	0,002019971	23
satan probe	3713	2,5000505	655	0,441026953	4368
smurf dos	2814	1,89473259	497	0,334641826	3311
spy r2l	2	0,00134665	0	0	2
teardrop dos	768	0,51711252	136	0,091572009	904
warezclient r2l	757	0,50970596	134	0,090225361	890
warezmaster r2l	819	0,55145202	145	0,097631921	964
normal	65496	44,100002	11558	7,782274083	77054

Figure 38: Test and Train split results of NSL-KDD

Figure 38 shows details about training and testing split percentage which is 85% 15% for NSL-KDD and shows attacks categories, their count, their percentage within the dataset and the total of each attack.

Remark: Choosing to use KDDCUP99 and NSL-KDD for testing the model of an IDS engine for wireless sensor network is due to the lack of recent real world datasets. Major companies or institutions that use WSNs keep network traffic secret there for the chosen datasets are for study purposes, and they are used for comparison with the approaches in the literature.

5. Results & implementation

In this section we are going to discuss the results obtained from each phase of the proposed approach on each dataset:

5.1. KDDCUP99

Discussing the results obtained from KDDCUP99:

5.1.1. Feature selection

The count of the selected features is 36 out of 41 columns. Each attribute has its own importance score as it's shown in the figure below.

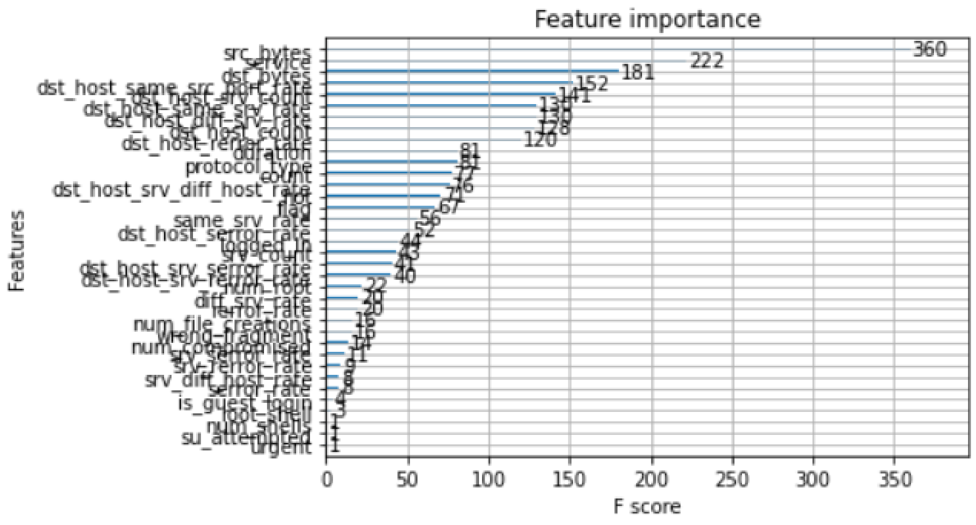
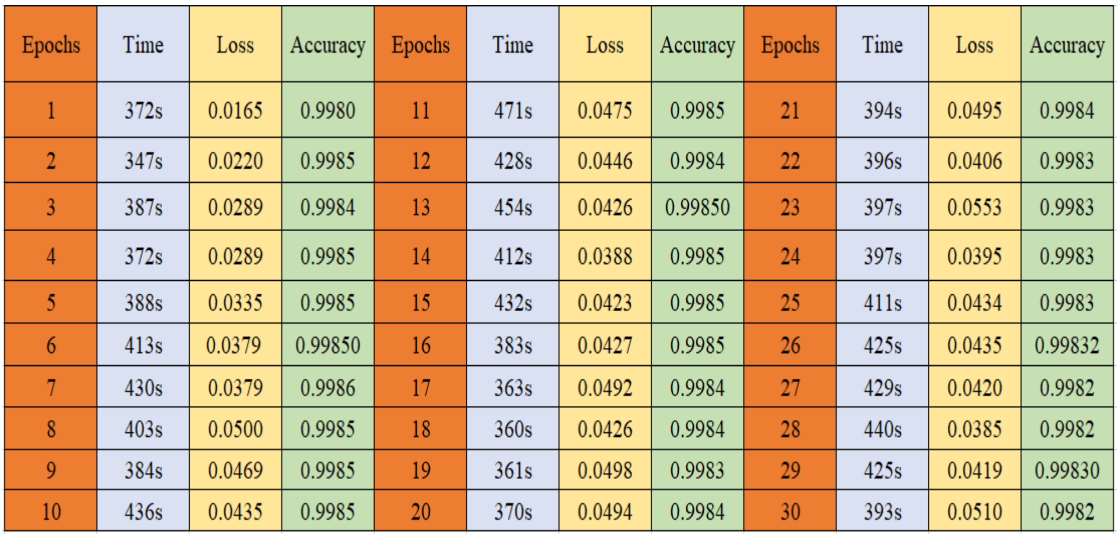


Figure 39: Features scores of training set of KDDCUP99

5.1.2. DNN Classification

The figure below shows the results while running the DNN of the 30 epochs with 10 batch size, their time, their loss and accuracy.



Epochs	Time	Loss	Accuracy	Epochs	Time	Loss	Accuracy	Epochs	Time	Loss	Accuracy
1	372s	0.0165	0.9980	11	471s	0.0475	0.9985	21	394s	0.0495	0.9984
2	347s	0.0220	0.9985	12	428s	0.0446	0.9984	22	396s	0.0406	0.9983
3	387s	0.0289	0.9984	13	454s	0.0426	0.99850	23	397s	0.0553	0.9983
4	372s	0.0289	0.9985	14	412s	0.0388	0.9985	24	397s	0.0395	0.9983
5	388s	0.0335	0.9985	15	432s	0.0423	0.9985	25	411s	0.0434	0.9983
6	413s	0.0379	0.99850	16	383s	0.0427	0.9985	26	425s	0.0435	0.99832
7	430s	0.0379	0.9986	17	363s	0.0492	0.9984	27	429s	0.0420	0.9982
8	403s	0.0500	0.9985	18	360s	0.0426	0.9984	28	440s	0.0385	0.9982
9	384s	0.0469	0.9985	19	361s	0.0498	0.9983	29	425s	0.0419	0.99830
10	436s	0.0435	0.9985	20	370s	0.0494	0.9984	30	393s	0.0510	0.9982

Figure 40: Details about epochs of the DNN (Loss, accuracy, time) on KDDCUP99 dataset

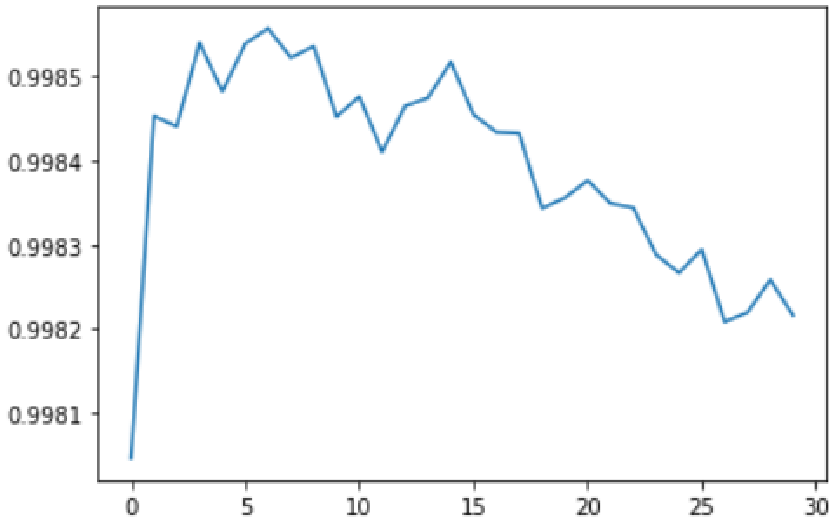


Figure 41: Graphical representation of DNN epochs and their accuracy on KDDCUP99 dataset

The figure 40 shows loss, accuracy and time for each and every 30 epochs that was used in the DNN for KDDCUP dataset. The highest accuracy is in the epoch number 7 of 0.9986 and the lowest is in the epoch number 1 of 0.998. The highest lost rate occurred in the epoch number 23 of 0.0553 and the lowest is in the first epoch of 0.0165.

5.2. NSL-KDD

Discussing the results obtained from NSL-KDD.

5.2.1. *Feature selection*

The count of the selected features is 32 out of 41 columns which means there is feature reduction. Each attributes has its own importance score as it's shown in the figure below:

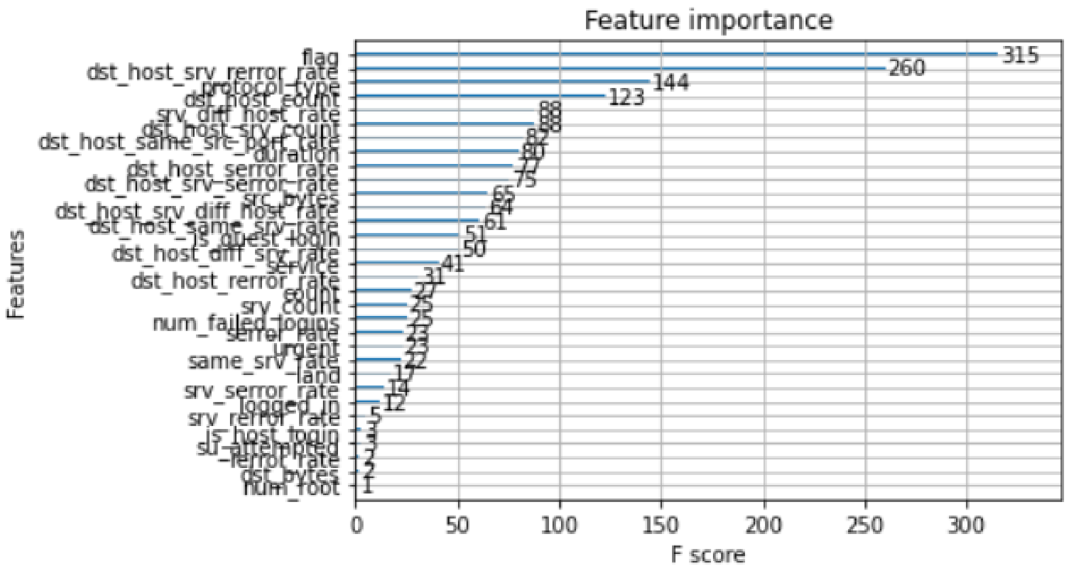
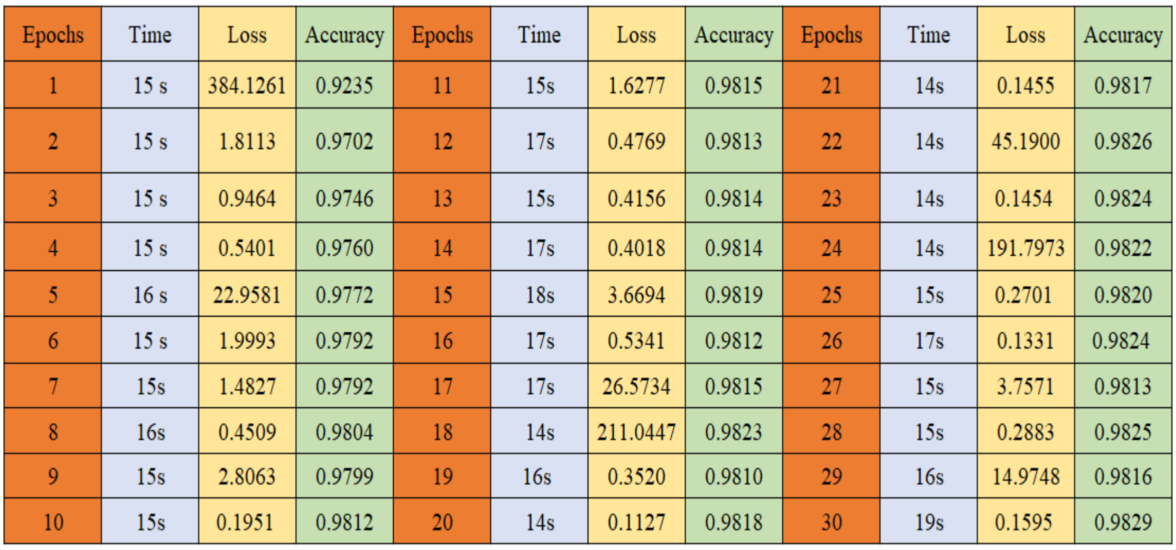


Figure 42: Features scores of training set of NSL-KDD

5.2.2. *DNN Classification*

The figure below shows the results while running the DNN of the 30 epochs with 10 batch size, their time, their loss and accuracy.



Epochs	Time	Loss	Accuracy	Epochs	Time	Loss	Accuracy	Epochs	Time	Loss	Accuracy
1	15 s	384.1261	0.9235	11	15s	1.6277	0.9815	21	14s	0.1455	0.9817
2	15 s	1.8113	0.9702	12	17s	0.4769	0.9813	22	14s	45.1900	0.9826
3	15 s	0.9464	0.9746	13	15s	0.4156	0.9814	23	14s	0.1454	0.9824
4	15 s	0.5401	0.9760	14	17s	0.4018	0.9814	24	14s	191.7973	0.9822
5	16 s	22.9581	0.9772	15	18s	3.6694	0.9819	25	15s	0.2701	0.9820
6	15 s	1.9993	0.9792	16	17s	0.5341	0.9812	26	17s	0.1331	0.9824
7	15s	1.4827	0.9792	17	17s	26.5734	0.9815	27	15s	3.7571	0.9813
8	16s	0.4509	0.9804	18	14s	211.0447	0.9823	28	15s	0.2883	0.9825
9	15s	2.8063	0.9799	19	16s	0.3520	0.9810	29	16s	14.9748	0.9816
10	15s	0.1951	0.9812	20	14s	0.1127	0.9818	30	19s	0.1595	0.9829

Figure 43: Details about epochs of the DNN (Loss, accuracy, time) on NSL-KDD dataset

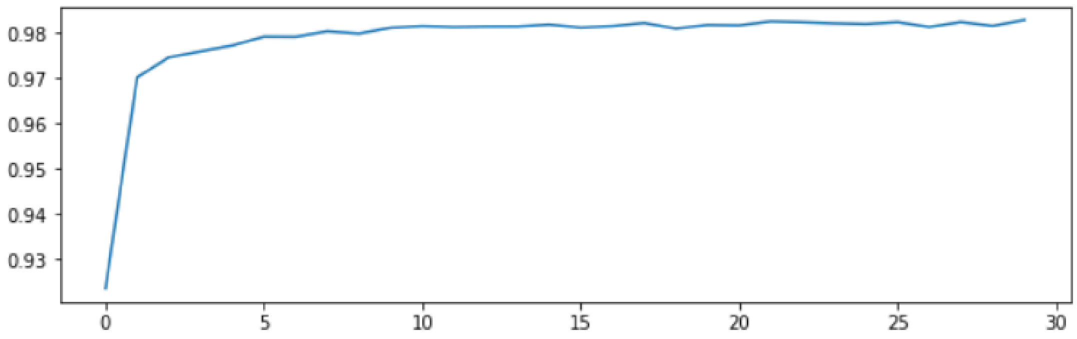


Figure 44: Graphical representation of DNN epochs and their accuracy on NSL-KDD dataset

The figure 43 shows loss, accuracy and time for each and every 30 epochs that was used in the DNN for NSL-KDD dataset. The highest accuracy is in the epoch number 30 of 0.9929 and the lowest is in the epoch number 1 of 0.99235. The highest lost rate occurred in the epoch number 1 of 384.1262 and the lowest is in epoch number 20 of 0.1127.

6. Metrics

The evaluation of performance of the proposed methods based on the following metrics:

- Accuracy (AC): the percentage of the number of records classified correctly versus total the records

$$AC = \frac{\mathrm{TP} + \mathrm{TN}}{TP + TN + FP + FN}$$

- Precision (P): Defined as the % ratio of the number of true positives (TP) records divided by the sum of true positives (TP) and false positives (FP) classified records.

$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \times 100\%$$

- Recall (R): Defined as the % ratio of number of true positives records divided by the sum of true positives and false negatives (FN) classified records.

$$R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \times 100\%$$

- F-Measure (F): Defined as the harmonic mean of precision and recall and represents a balance between them.

$$\mathrm{F} = \frac{2.\mathrm{P}.\mathrm{R}}{\mathrm{P} + \mathrm{R}}$$

The figure below show the results obtained from the proposed approach from KDDCUP99 dataset:

Table 2: Confusion matrix (TN, FP, FN, TP), Accuracy, Recall, Precision, F1 score for KDDCUP99

Precision	Recall	Accuracy	F1 Score
99.86%	99.86%	99.86%	99.86%
True negative	False Positive	False negative	True positive
291793	41	1973	1175723

The figure below show the results obtained from the proposed approach from NSL-KDD dataset:

Table 3: Confusion matrix (TN, FP, FN, TP), Accuracy, Recall, Precision, F1 score for NSL-KDD

Precision	Recall	Accuracy	F1 Score
83.21%	76.21%	76.21%	75.80%
True negative	False Positive	False negative	True positive
9480	231	5132	7701

7. Comparison of results with the literature

Table 4 displays a comparison between our model (Called FS + DNN where FS means feature selection, DNN refers to deep neural network) and between other models that were studied in chapter four in state of art section. It shows the protocol followed by each author of the article, where all the models performed on KDDCUP99 dataset have followed 70% training and 30% testing which is the same testing protocol as our model. For NSL-KDD five models have used 85% training and 15% testing same as our model, two other models BCNN and BCNN-DFS have used 80% training and 20% testing. Table 4 compares Accuracy, F measure, Recall and precision for each model.

Table 4: Comparison of results with the literature

Approach	Dataset					
	KDDCUP					
	Training	Testing	Accuracy	F measure	Recall	Precision
FS + DNN	70%	30%	99.86%	99.86%	99.86%	99.86%
RBC-IDS [56]	70%	30%	NONE	NONE	NONE	99.12%
CFPA [61]	70%	30%	87.63%	99.27%	99.99%	99.77%
	NSL-KDD					
	Training	Testing	Accuracy	F measure	Recall	Precision
FS + DNN	85%	15%	76.21%	75.80%	76.21%	83.21%
STL[57]	85%	15%	88.39%	90.40%	95.95%	85.44%
SMR [57]	85%	15%	78.06%	76.80%	63.73%	96.56%
2-class RNN [58]	85%	15%	83.28%	83.24%	72.95%	96.91%
BCNN [59]	80%	20%	94.42%	94%	94%	95%
BCNN-DFS [59]	80%	20%	95.71%	96%	96%	96%
DNN [60]	85%	15%	75.75%	None	None	None
CFPA [61]	85%	15%	60.56%	75.20%	100%	66.89%

8. Simulation

For illustrative purposes, a platform that simulates a small WSN is developed. It is structured of sensor nodes connected to a one node called cluster head (CH) which forms clusters of sensors. Each cluster head is connected to the base station, which is an architecture adopted on Low Energy Adaptive Clustering Hierarchy (LEACH) protocol.

The proposed approach of intrusion detection system engine is installed in the base station, it’s responsible for classifying normal and malicious packets.

The simulation starts when the user clicks on the RUN button and stops when he clicks on STOP. When the RUN button is clicked sensor nodes start sending packets to the base station each 1 second. Their state (normal or malicious node) could be switched by clicking on the node.

Details about number of packets sent, malicious nodes, normal nodes, malicious packets sent and number of malicious packets detected are displayed.

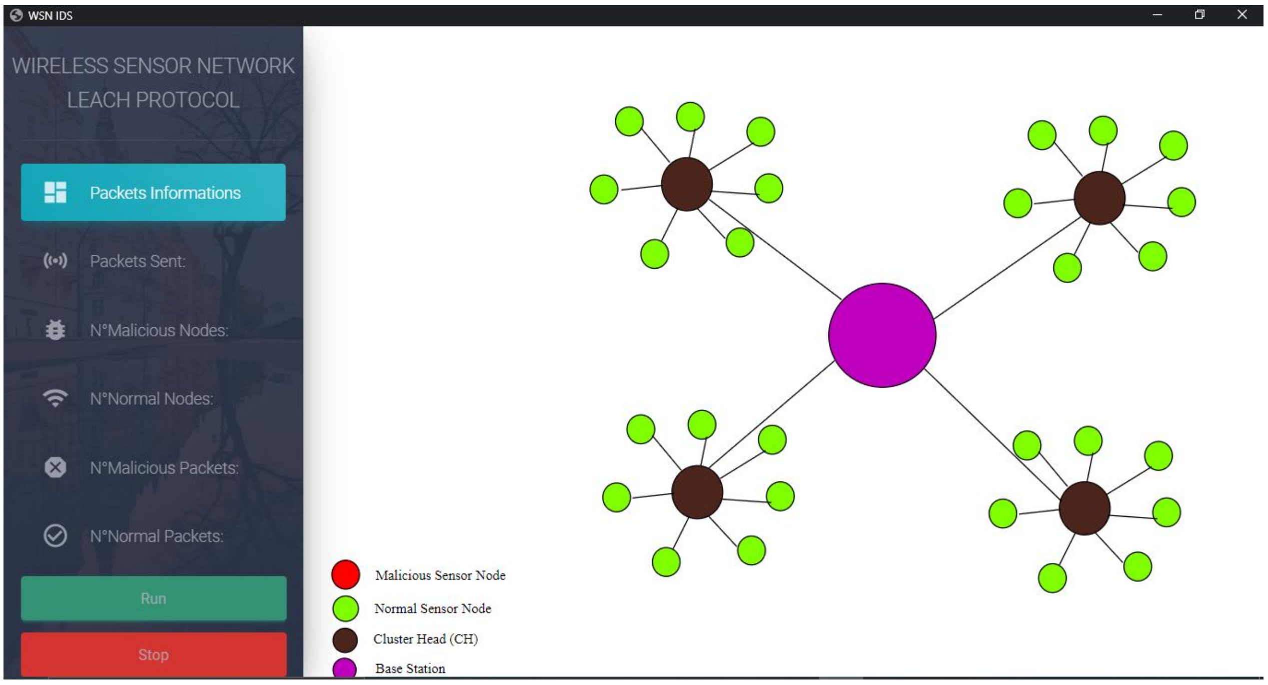


Figure 45: Screenshot of the interface of the implemented simulator

9. Conclusion

We have cited and defined the environment used during this study, libraries and APIs that helped the realization of the project, training and testing protocol for each dataset. We have displayed the results obtained for each phase on the model for both datasets KDDCUP99 and NSL-KDD.

The results obtained during this study are accurate and precise compared to what researchers have accomplished in their published papers. For KDDCUP precision, recall, F1_score and accuracy scores are 99.86%, for NSL-KDD 83.21% precision, 76.21% accuracy, F1_score 75.80% and recall 76.21%.

General conclusion

This thesis aimed to build a more accurate, fast and a precise intrusion detection system engine for wireless sensor networks based on recent artificial intelligence techniques such as Feature Engineering (Feature Selection using importance scores) and deep neural networks.

The intrusion detection technique used for classifying malicious traffic is by binary classification, which is a faster, less power consuming and less memory consumption approach comparing to multi-classification in the field of wireless sensor networks, since WSNs have several constraints that makes building an IDS for them very challenging and difficult, such as communication time latency and low computation capability.

The proposed approach is constituted of three parts: preprocessing datasets (NSL-KDD and KDDCUP99), feature selection based on features importance scores and a DNN classifier that is composed of four layers (input layer, two hidden layers and output layer).

Features selection phase have selected 32 features on NSL-KDD and 36 features on KDDCUP99.

The DNN classifier has resulted good accuracy metrics in both datasets compared with the literature.

For future works, an integration of the proposed model could be done by trying to install the IDS in different approaches such as central, distributed and hybrid approach in a real environment with real traffic data.

Various techniques could be applied in the feature engineering phase such as trying unsupervised feature selection or other supervised methods like wrapping and filtering.

A different classification model other than a typical DNN could be applied like Convolutional Neural Networks or Deep Belief Networks but with taking into consideration the WSNs constraints.

References

[1] S. Brand, *The media lab : inventing the future at MIT*. New York; London: Penguin, 1998.

[2] J. Kozio, G. Schultz, C. F. Endorf, and J. Mellander, "Intrusion Detection and Prevention," (in English), 2003.

[3] R. U. Rehman, *Intrusion detection systems with Snort advanced IDS techniques using Snort, Apache, MySQL, PHP and ACID*. Upper Saddle River, N.J: Prentice Hall, 2003.

[4] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *Communications Magazine, IEEE,* vol. 40, pp. 102-114, 09/01 2002.

[5] "Copyright Page," in *Building Intelligent Interactive Tutors*, B. P. Woolf, Ed. San Francisco: Morgan Kaufmann, 2009, p. iii.

[6] *Wireless Sensor network*. Available: https://www.geeksforgeeks.org/wireless-sensor-network-wsn/

[7] *Wireless Sensor Network Architecture and Its Applications*. Available: https://www.elprocus.com/architecture-of-wireless-sensor-network-and-applications/

[8] D. Sharma, S. Verma, and K. Sharma, "Network topologies in wireless sensor networks: A review," *Int. J. Electron. Commun. Technol.,* vol. 4, pp. 93-97, 01/01 2013.

[9] F. Hamad, H. Fakhouri, and O. Rababah, "Comprehensive Overview of Security and Privacy of Data Transfer in Wireless ad Hock Network," *Modern Applied Science,* vol. 12, p. 185, 11/26 2018.

[10] M. J. McGrath and C. N. Scanaill, "Sensor Network Topologies and Design Considerations," in *Sensor Technologies: Healthcare, Wellness, and Environmental Applications*, M. J. McGrath and C. N. Scanaill, Eds. Berkeley, CA: Apress, pp. 79-95, 2013.

[11] M. Ahlawat and A. Mittal, "Different Communication Protocols for Wireless Sensor Networks: A Review," *IJARCCE,* pp. 213-216, 03/30 2015.

[12] J. Anzola, G. Tarazona, J. Espada, and R. Gonzalez Crespo, "A Clustering WSN Routing Protocol Based on k-d Tree Algorithm," *Sensors,* vol. 18, pp. 1-25, 09/01 2018.

[13] G. Anastasi, M. Conti, M. Francesco, and A. Passarella, "Energy Conservation in Wireless Sensor Networks: A Survey," *Ad Hoc Networks,* vol. 7, pp. 537-568, 05/01 2009.

[14] T. S, K. Jain, and G. N. Purohit, "Application Domain of Wireless Sensor Network: - A Paradigm in Developed and Developing Countries," *International Journal of Computer Science Issues*, vol. 8, 07/01 2011.

[15] J. Panchard, S. Rao, T. V. Prabhakar, H. S. Jamadagni, and J. Hubaux, "COMMON-Sense Net: Improved Water Management for Resource-Poor Farmers via Sensor Networks," in *2006 International Conference on*

Information and Communication Technologies and Development, pp. 22-33, 2006.

[16] G. S. Abawi and T. L. Widmer, "Impact of soil health management practices on soilborne pathogens, nematodes and root diseases of vegetable crops," *Applied Soil Ecology,* vol. 15, pp. 37-47, 08/01 2000.

[17] S. Community. Available: http://weblogs.sdn.sap.com/cs/blank/view/wlg/1913

[18] M. Sadiku, K. Eze, and S. Musa, "Wireless Sensor Networks for Healthcare," 08/21 2018.

[19] J. Sen, "Security in Wireless Sensor Networks", pp. 407-460, 2012.

[20] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," presented at the Proceedings of the ninth international conference on Architectural support for programming languages and operating systems, Cambridge, Massachusetts, USA, 2000. Available: https://doi.org/10.1145/378993.379006

[21] Y. Zhou, Z. Zhao, and J. Yu, "A Key Management Scheme of WSN Based on Trust Mechanism," *Sensors and Transducers,* vol. 174, pp. 103-108, 07/01 2014.

[22] G. M. Edake, G. R. Pathak, and S. H. Patil, "Secure Localization and Location Verification of Wireless Sensor Network," in *2014 Fourth International Conference on Communication Systems and Network Technologies*, pp. 673-676, 2014.

[23] J. Deng, R. Han, and S. Mishra, *Countermeasures Against Traffic Analysis Attacks in Wireless Sensor Networks*. pp. 113-126, 2005.

[24] I. F. Akyildiz, S. Weilian, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine,* vol. 40, no. 8, pp. 102-114, 2002.

[25] A. Lazarevic, V. Kumar, and J. Srivastava, "Intrusion Detection: A Survey," vol. 5, pp. 19-78, 2005.

[26] T. Holz, M. Meier, and H. König, *High-Efficient Intrusion Detection Infrastructure*., pp. 217-232, 2003.

[27] S. Solutions. Available: https://www.security-audit.com/siem-solutions-log-management/

[28] D. E. Denning, "An Intrusion-Detection Model," *IEEE Transactions on Software Engineering,* vol. SE-13, no. 2, pp. 222-232, 1987.

[29] S. M. Othman, N. T. Alsohybe, F. M. Ba-Alwi, and A. T. Zahary, "Survey on intrusion detection system types," *International Journal of Cyber-Security and Digital Forensics,* vol. 7, no. 4, pp. 444-463, 2018.

[30] H. Hindy *et al.*, *A Taxonomy and Survey of Intrusion Detection System Design Techniques, Network Threats and Datasets*. 2018.

[31] K. Hwang, P. Dave, and S. Tanachaiwiwat, "NetShield: Protocol Anomaly Detection with Datamining Against DDoS Attacks," 01/01 2003.

[32] K. Khan, A. Mehmood, S. Khan, M. Altaf, Z. Iqbal, and W. Mashwani, "A survey on Intrusion Detection and Prevention in Wireless Ad-hoc

Networks," *Journal of Systems Architecture,* vol. 105, p. 101701, 12/01 2019.
[33] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in Cloud," *Journal of Network and Computer Applications,* vol. 36, no. 1, pp. 42-57, 2013/01/01/ 2013.
[34] H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *Journal of Network and Computer Applications,* vol. 36, no. 1, pp. 16-24, 2013/01/01/ 2013.
[35] Techopedia. Available: https://www.techopedia.com/definition/12826/host-based-intrusion-detection-system-hids
[36] B. P. Woolf, "Chapter 7 - Machine Learning," in *Building Intelligent Interactive Tutors*, B. P. Woolf, Ed. San Francisco: Morgan Kaufmann, pp. 221-297, 2009.
[37] IBM. *Supervised learning*. Available: https://www.ibm.com/cloud/learn/supervised-learning
[38] IBM. *Unsupervised learning*. Available: https://www.ibm.com/cloud/learn/unsupervised-learning
[39] Available: https://ichi.pro/fr/pour-commencer-avec-le-clustering-k-means-31262593745935
[40] *5 things you need to know about reinforcement learning*. Available: https://www.kdnuggets.com/2018/03/5-things-reinforcement-learning.html
[41] IBM. *Reinforcement learning*. Available: https://developer.ibm.com/articles/cc-models-machine-learning/#reinforcement-learning
[42] *Deep Understanding of Discriminative and Generative Models in Machine Learning*. Available: https://www.analyticsvidhya.com/blog/2021/07/deep-understanding-of-discriminative-and-generative-models-in-machine-learning/
[43] F. Emmert-Streib, Z. Yang, H. Feng, S. Tripathi, and M. Dehmer, "An Introductory Review of Deep Learning for Prediction Models With Big Data," (in English), Review vol. 3, no. 4, 2020-February-28 2020.
[44] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature,* vol. 521, no. 7553, pp. 436-444, 2015/05/01 2015.
[45] H. Boukadida, N. Hassen, Z. Gafsi, and K. Besbes, "A HIGHLY TIME-EFFICIENT DIGITAL MULTIPLIER BASED ON THE A2 BINARY REPRESENTATION," *International Journal of Engineering Science and Technology,* vol. 3, 05/01 2011.
[46] S. S. Haykin, *Neural Networks and Learning Machines*. Pearson, 2009.
[47] F. Bre, J. Gimenez, and V. Fachinotti, "Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks," *Energy and Buildings,* vol. 158, 11/01 2017.
[48] N. Tangri, D. Ansell, and D. Naimark, "Predicting technique survival in peritoneal dialysis patients: Comparing artificial neural networks and logistic regression," *Nephrology, dialysis, transplantation : official*

publication of the European Dialysis and Transplant Association - European Renal Association, vol. 23, pp. 2972-81, 05/01 2008.

[49] E. Arisoy, T. N. Sainath, B. Kingsbury, and B. Ramabhadran, "Deep neural network language models," presented at the Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT, Montreal, Canada, 2012.

[50] TowardsDataScience, "A Comprehensive Guide to Convolutional Neural Networks."

[51] IBM. (Recurrent neural network). Available: https://www.ibm.com/cloud/learn/recurrent-neural-networks?mhsrc=ibmsearch_a&mhq=RNN

[52] *Recurrent neural network*. Available: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks

[53] *Types of RNN*. Available: https://iq.opengenus.org/types-of-rnn/

[54] K. Kaur and B. J. I. J. o. C. A. Singh, "Wireless Sensor Network based: Design Principles & measuring performance of IDS," vol. 1, pp. 94-99, 2010.

[55] F. Hidoussi, H. Toral-Cruz, D. E. Boubiche, K. Lakhtaria, A. Mihovska, and M. Voznak, "Centralized IDS Based on Misuse Detection for Cluster-Based Wireless Sensors Networks," *Wireless Personal Communications,* vol. 85, no. 1, pp. 207-224, 2015/11/01 2015.

[56] S. Otoum, B. Kantarci, and H. T. Mouftah, "On the Feasibility of Deep Learning in Sensor Network Intrusion Detection," *IEEE Networking Letters,* vol. 1, no. 2, pp. 68-71, 2019.

[57] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, *A Deep Learning Approach for Network Intrusion Detection System*. 2015.

[58] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks," *IEEE Access,* vol. 5, pp. 21954-21961, 2017.

[59] "A Convolutional Neural Network for Improved Anomaly-Based Network Intrusion Detection," vol. 9, no. 3, pp. 233-252, 2021.

[60] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for Network Intrusion Detection in Software Defined Networking," in *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, pp. 258-263, 2016.

[61] A. Singh, A. Kaur, and S. Pal, "A novel Chaotic Flower Pollination-based intrusion detection framework," *Soft Computing,* vol. 24, 11/01 2020.

[62] *Data Preprocessing: Concepts*. Available: https://towardsdatascience.com/data-preprocessing-concepts-fa946d11c825

[63] TowardsDataScience. *Categorical encoding using Label-Encoding and One-Hot-Encoder*. Available: https://towardsdatascience.com/categorical-encoding-using-label-encoding-and-one-hot-encoder-911ef77fb5bd

[64] *Introduction to Feature Selection methods with an example (or how to select the right variables?)*. Available: https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/

[65] Datascientest. *Algorithmes de Boosting – AdaBoost, Gradient Boosting, XGBoost*. Available: https://datascientest.com/algorithmes-de-boosting-adaboost-gradient-boosting-xgboost

[66] *Bagging (Bootstrap aggregation)*. Available: https://corporatefinanceinstitute.com/resources/knowledge/other/bagging-bootstrap-aggregation/

[67] CFI. *Bagging (Bootstrap Aggregation)*. Available: https://corporatefinanceinstitute.com/resources/knowledge/other/bagging-bootstrap-aggregation/

[68] L. Breiman, "Bias, Variance , And Arcing Classifiers," *Technical Report 460, Statistics Department, University of California,* 11/28 2000.

[69] *Gradient Boosting – Ce que vous devez savoir*. Available: https://datascience.eu/fr/apprentissage-automatique/gradient-boosting-ce-que-vous-devez-savoir/

[70] C. Zhang, Y. Zhang, X. Shi, G. Almpanidis, G. Fan, and X. Shen, "On Incremental Learning for Gradient Boosting Decision Trees," *Neural Processing Letters,* vol. 50, no. 1, pp. 957-987, 2019.

[71] *How to Choose an Activation Function for Deep Learning*. Available: https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/

[72] Anaconda.com, "Anaconda."

[73] Anaconda.com, "Anaconda Installation guide."

[74] jupyter.org, "Jupyter."

[75] Keras. *Keras*. Available: https://keras.io/

[76] Scikitlearn. *Scikitlearn*. Available: https://scikit-learn.org/

[77] Numpy.org. *What is numpy*. Available: https://numpy.org/

[78] https://matplotlib.org/. *Matplotlib: visualisation with python*. Available: https://matplotlib.org/

[79] Pandas. *Pandas*. Available: https://pandas.pydata.org/

[80] XGBoost. *XGBoost*. Available: https://xgboost.readthedocs.io/

[81] TensorFlow. *TensorFlow*. Available: https://www.tensorflow.org/

[82] uci. *KDD Cup 1999 Data*. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[83] Unb. *NSL-KDD dataset*. Available: https://www.unb.ca/cic/datasets/nsl.html

[84] S. J. Stolfo, F. Wei, L. Wenke, A. Prodromidis, and P. K. Chan, "Cost-based modeling for fraud and intrusion detection: results from the JAM project," in *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00,* vol. 2, pp. 130-144 vol.2, 2000.

[85] R. P. Lippmann *et al.*, "Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation," in *Proceedings DARPA*

Information Survivability Conference and Exposition. DISCEX'00, vol. 2, pp. 12-26 vol.2, 2000.

[86] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 1-6, 2009.

[87] J. McHugh, "Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory," vol. 3, no. 4 %J ACM Trans. Inf. Syst. Secur., pp. 262–294, 2000.