A SEMI SUPERVISED CLASSIFICATION BASED APPROACH USING AUTOENCODERS FOR INTRUSION DETECTION SYSTEM

Hamdi AKTAR

Master of Science Thesis

Department of Computer Engineering

Supervisor: Assist. Prof. Cahit PERKGÖZ

Eskişehir

Eskişehir Technical University

Institute of Graduate Programs

January 2022

07/01/2022

SUPERVISOR APPROVAL

Master of Science student Hamdi AKTAR, whom I supervise, has completed this thesis titled A SEMI SUPERVISED CLASSIFICATION BASED APPROACH USING AUTOENCODERS FOR INTRUSION DETECTION SYSTEM. According to my inspections, the work is scientifically and ethically appropriate for the student to the thesis defense exam.

Supervisor

Assist. Prof. Cahit PERKGÖZ

ABSTRACT

A SEMI SUPERVISED CLASSIFICATION BASED APPROACH USING AUTOENCODERS FOR INTRUSION DETECTION SYSTEM

Hamdi AKTAR

Department of Computer Engineering

Eskişehir Technical University, Institute of Graduate Programs, January 2022

Supervisor: Assist. Prof. Cahit PERKGÖZ

With the major developments in Internet-of-Things (IoT), cloud-based solutions, and wide selection of data traffic, there is a rising demand for powerful anomaly detection methods for intrusion detection systems (IDS) that can overcome sophisticated network attacks. Rapid advancements in deep learning (DP) and machine learning (ML) have gained the attention of researchers as it has the opportunity to bring a powerful solution to complex attacks. In comparison to other conventional techniques, such as firewall, an Intrusion Detection System (IDS) is the first layer of protection against network-based cyberattacks.

Currently available intrusion detection approaches are often based on conventional machine learning models such as Random Forest and Support Vector Machine, and while they depend heavily on manual feature extraction, they have relatively a lower level of accuracy. To overcome the shortcomings of intrusion detection systems in terms of detection and feature extraction, a unique approach to incorporating both supervised learning as well as unsupervised learning (Semi supervised) into Intrusion Detection Systems (IDS) is proposed. The proposed method is utilizing Deep Autoencoder (DAE) being a dimensionality reduction technique. The trained DAE encoder cannot just automatically extract features, also the latent layer can be used as an input to supervised learning algorithms such as Naive Bayes and K-Nearest Neighbor to enhance the detection accuracy of IDS. The overall performance of the proposed model is evaluated using the UNSW-NB15 dataset. The results show that the model has better performance than a number of models where Deep Autoencoder (DAE) is not used.

Keywords: Autoencoder, Classification, Semi-Supervised, Intrusion Detection

ÖZET

AUTOENCODER KULLANAN YARI DENETİMLİ SINIFLANDIRMA TABANLI SALDIRI TESPİT SİSTEMİ

Hamdi AKTAR

Bilgisayar Mühendisliği Anabilim Dalı Anabilim Dalı

Eskişehir Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü, Ocak 2022

Danışman: Assist. Prof. Cahit PERKGÖZ

Nesnelerin İnterneti (IoT), bulut tabanlı çözümler ve geniş veri trafiği alanındaki gelişmeler, karmaşık ağ saldırılarının üstesinden gelebilecek, anormallik tespit yöntemlerine yönelik güçlü çözümler içeren yeni Saldırı Tespit Sistemlerinin (IDS) tasarlanabilmesinin yolunu açmıştır. Derin öğrenme (DP) ve makine öğrenimindeki (ML) hızlı gelişmeler, karmaşık saldırılara güçlü bir çözüm getirme fırsatına sahip olduğu için araştırmacıların dikkatini çekmiştir. Güvenlik duvarı gibi diğer geleneksel tekniklerle karşılaştırıldığında, Saldırı Tespit Sistemi (IDS), ağ tabanlı siber saldırılara karşı ilk koruma katmanıdır.

Önceden var olan saldırı tespit sistemleri, genellikle Rassal Orman ve Karar Ağacı gibi geleneksel makine öğrenimi modellerine dayanmaktadır, ancak bu modeller daha çok yapay özellik çıkarımına (feature extraction) dayanmaktadır ve nispeten daha düşük doğruluğa sahiptir. Bu çalışmada Saldırı Tespit Sistemlerinde düşük algılama ve özellik çıkarma sorunlarını çözmek için, hem Güdümlü hem de Güdümsüz Öğrenme (Yarı Güdümlü) yaklaşımlarını Saldırı Tespit Sistemlerine (IDS) dahil etmek için yeni bir yaklaşım önerilmiştir. Önerilen model, bir boyut azaltma tekniği olan Derin Otokodlayıcı içermektedir. Eğitimli DAE kodlayıcı, özellikleri yalnızca otomatik olarak çıkarmaz, aynı zamanda bu gizli katman, IDS'nin algılama doğruluğunu artırmak için Naive Bayes ve Rassal Orman gibi Güdümlü Öğrenme algoritmalarına girdi olarak kullanılabilir. UNSW-NB15 veri setinde önerilen modelin genel performansı değerlendirilmiş ve sonuçların, modelin Derin Otokodlayıcı (DAE) kullanılmayan diğer modellere göre daha iyi bir performansa sahip olduğunu gösterdiği görülmüştür.

Anahtar Sözcükler: Otokodlayıcı, Sınıflandırma, Yarı Denetimli, Saldırı Tespit Sistemi.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my research supervisor, Assist. Prof. Cahit PERKGÖZ. Without his assistance and dedicated involvement in every step throughout the process, this thesis would have never been accomplished. I would like to thank him very much for his support and understanding over these past two years.

Also, I would like to express my gratitude to my family especially my parents. They encouraged me to keep going even in the times when I was ready to quit. Without their tremendous understanding and encouragement in the past few years, it would be impossible for me to complete my study.

Hamdi AKTAR

07/01/2022

STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES

I hereby truthfully declare that this thesis is an original work prepared by me; that I have behaved in accordance with the scientific ethical principles and rules throughout the stages of preparation, data collection, analysis and presentation of my work; that I have cited the sources of all the data and information that could be obtained within the scope of this study, and included these sources in the references section; and that this study has been scanned for plagiarism with "scientific plagiarism detection program" used by Eskişehir Technical University, and that "it does not have any plagiarism" whatsoever. I also declare that, if a case contrary to my declaration is detected in my work at any time, I hereby express my consent to all the ethical and legal consequences that are involved.

Hamdi AKTAR

CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

GLOSSARY OF SYMBOLS AND ABBREVIATIONS

IoT	:	Internet of Things
IDS	:	Intrusion Detection Systems
DP	:	Deep Learning
ML	:	Machine Learning
RL	:	Reinforcement Learning
DAE	:	Deep Autoencoder
SVM	:	Support Vector Machine
RF	:	Random Forest
DT	:	Decision Tree
KNN	:	K-nearest Neighbor
NB	:	Naive Bayes
LR	:	Logistic Regression
AR	:	Association Rule
NN	:	Neural Networks
ANN	:	Artificial Neural Networks
CNN	:	Convolutional Neural Networks
GAN	:	Generative Adversarial Networks
RNN	:	Recurrent Neural Networks
LSTM	:	Long-Short Term Memory
RBM	:	Restricted Boltzmann Machines
FDN	:	Feed forward Deep Networks
DBN	:	Deep Belief Networks
PCA	:	Principal Component Analysis
IP	:	Internet Protocol
LAN	:	Local Area Network
WAN	:	Wide Area Network
RFID	:	Radio Frequency Identification
BLE	:	Bluetooth Low Energy
LTE	:	Long Term Evolution
NFC	:	Near-field communication
DDoS	:	Distributed Denial of Service Attack

1. INTRODUCTION

Cybersecurity threats in the Internet of Things (IoT) are becoming a more important difficulty as network intrusion methods and attack technologies continue to evolve and become more sophisticated. How to quickly detect attacks in the IoT has emerged as one of the most pressing issues in network security. It is vital that more effective detection technologies be developed.

An intrusion detection system (IDS) as an active defense technology is able to monitor hosts or networks and alert when an attack is discovered. Network security can be much better ensured via intrusion detection techniques where the network attack behavior could be learned through data analysis & modeling.

Intrusion detection systems can be classified into two types: anomaly-based IDSs and signature-based IDSs. A major technological challenge is that IDSs must identify never-before-seen threats. In a signature-based IDS, intrusions are detected using pre-installed rules. By comparing the network's traffic flow to the latest attack signature databases, the intrusion in the network traffic dataset can be discovered. By analyzing the irregularity in the network's data stream, anomaly-based IDSs identify if the network traffic is abnormal or normal. Network activity that deviates from the expected data pattern is referred to as abnormal behavior.

In this work anomaly-based IDSs are mostly studied due to their ability to detect abnormal or unknown attacks. Anomaly detection techniques are utilized in national security departments, communications, securities, finance, banking, and medical-related applications (Kumar, Kumar, & Sachdeva, 2010). In fact, the IDS mostly determines whether the network traffic is abnormal or normal, so that the IDS could be linked to a classification problem. By enhancing the overall performance of the classifier in successfully identifying abnormal traffic, the reliability and accuracy of intrusion detection is significantly improved. Hence improving the reliability of IDS is realized by enhancing the overall performance of the classifier in identifying abnormal traffic.

Recently, a growing number of intrusion detection methods are already applied to intrusion detection, which includes machine learning techniques, ensemble learning methods, and deep learning solutions. Machine learning techniques first select features, after that they use classifiers to identify intrusions, for example support vector machine

(SVM), random forest (RF), as well as decision tree (DT). Deep learning methods are able to automatically extract features then classify to recognize intrusion detection, for example long short term memory (LSTM), deep neural networks (DNN), as well as autoencoders. The ensemble learning method makes use of many hybrid and ensemble solutions for intrusion detection, which includes stacking, bagging, stacking, in addition to combined classifier methods.

Various Machine learning algorithms have actually been used in IDS to improve their performance, leading to breakthroughs (Kumar, Kumar, & Sachdeva, 2010). Nevertheless, traditional machine learning approaches work better when they are applied on small datasets. They're less reliable when dealing with huge intrusion data classification in a real-world network application scenario. When large-scale data has high-dimensional characteristics or non-linear feature spaces, conventional machine learning methods typically face difficulties with high time and space complexity. As a result, developing or implementing a more effective approach for reducing the dimensionality of a high-dimensional data is an essential step in the development of IDSs.

Hinton et al., (2015) introduced the deep learning (DL) concept for the first time in 2006. DL is now widely utilized in image recognition, machine translation as well as speech recognition. Deep learning has the benefit of being able to generate abstract meaningful representations of raw data using multilayer neural networks.

Deep learning differs from conventional machine learning techniques of how it can extract features from raw data rather than incorporating manually constructed features. Effective features can significantly enhance IDS intrusion detection performance. In DL, feature learning is completed automatically by a multi-layer neural network. Each layer of the neurons extracts features from the data input and passes them along to the next layer of neurons. Deep learning, as opposed to conventional machine learning, relies less on the previous data distribution. The number of non-linear transformations of the data grows as the network depth increases, as does the model representation ability for data, resulting in a more significant characterization of the data. DL also has advantages when dealing with large-scale, high-dimensional data.

Taking into account the aforementioned aspects, we propose a new intrusion detection model that essentially incorporates unsupervised learning using Deep

autoencoder (DAE) and supervised learning classification algorithms, getting the benefits of both. DAE is used to compress the input to the latent layer, the DAE encoder latent layer, which is the dimensionality reduction representation of the input, is used as an input to one of the supervised learning algorithms to classify intrusions. The hybrid model is implemented using several machine learning methods, then the performance of the new model is compared to other models that did not employ the DAE. A comprehensive evaluation of the UNSW-NB15 dataset is performed to validate the effectiveness of the proposed model.

The following are the paper's key contributions and findings:

I. A deep learning model is proposed. The model combines unsupervised learning by utilizing Deep autoencoder (DAE) and supervised learning classification algorithms, the proposed model improves the accuracy of IDS and provides a new research method for intrusion detection;
II. The impact of incorporating autoencoders into the model is studied and compared with the performance of other models where autoencoders are not utilized.
III. A real UNSW-NB15 dataset is used to evaluate our proposed model.

The remainder of the paper is as follows: Section 2 provides background for the related technologies utilized in this study. Section 3 summarizes other work relevant to the study. Section 4 provides a deeper insight on the proposed intrusion detection methodology. Section 5 contains the experimental results as well as a comparative analysis. Section 6 presents our findings and discusses potential future work.

2. BACKGROUND

2.1. What is Internet of Things (IoT)

The Internet of Things (IoT) is a paradigm that includes connecting a large number of separate end devices and using the data obtained to create a variety of digital services. In 1999, Kevin Ashton came up with the term "Internet of Things" to define a global network of devices connected to a supply chain via Radio Frequency Identification (RFID) (Mocnej, 2018).

New technologies have emerged since then, and the Internet of Things has spread into new domains. Despite this, the fundamental goal has remained the same: to provide computers the ability to gather data without the need for human intervention (Mocnej, 2018). The Internet of Things aims to provide everyday things with computer power and an Internet connection, allowing them to observe, calculate, communicate, and control their surroundings. These capabilities are required in many applications because of their potential influence, ranging from making our lives easier and safer, to improving and simplifying current processes, to giving entirely new possibilities and opportunities (Mocnej, 2018).

The Internet of Things offers a new viewpoint on what kind of data should be collected, how often, and from where, providing access to previously unavailable data. However, in many cases, this benefit necessitates the transmission of low-power devices through lossy networks, putting significant strain on the installed technology. As a result, we need to understand the requirements of IoT application use cases in order to create more efficient IoT solutions.

According to Cisco Systems, by 2020, the Internet will have absorbed over 50 billion linked "things", including cars, home appliances, TV’s, surveillance cameras, cellphones, intra-body sensors, thermometers, and practically anything else we can think of (Cisco Systems, 2017). As a result, annual revenues for IoT providers supplying hardware, software, and full services for the IoT are expected to exceed $470 billion (Columbus, 2016). From smart home and smart city to schooling, health-care, production, mining, utilities, retail, transportation, surveillance, infrastructure services, and supply chain and logistics, IoT goods and services will pervade every area and business.

The possibilities afforded by the Internet of Things are limitless, and its maximum capabilities will soon be realized as more and more gadgets connect Online every day.

2.2. IoT Structure and technologies

Many objects are connected to the Web as part of the Internet of Things. The devices are able to respond to data taken from their surroundings, with the purpose of improving the intelligence of smart environments and apps in areas such as health, transportation, farming, power sources and other sectors. The internet of things' structure can be divided into three layers (Atzori, Iera, & Morabito, 2010), (Zhao & Ge, 2013).

The perception layer houses numerous sensors for sensing and acquiring environmental data, which is then sent to the network layer. The network layer consists mostly of a gate that connects the perception layer to the cloud. For transmitting information between objects, this interacting and cooperating necessitates integrating numerous communicating ways, such as Wireless Communication, 5G, Zigbee, and others. The data obtained out of the network layer is used by the application layer to produce the solutions or processes that users require. Although these layers have distinct goals, they are all subject to different types of attacks. Injection of viruses, eavesdropping, capturing nodes and interfering are all possible attacks against the perception layer.

2.2.1. The Perception Layer

Basically, the perception layer consists mostly of sensors and their data processors. The sensors can be defined as where an electronic signal is generated by a device in response to a physical state or event. An accelerometer for example. Other types of sensors have become compact, cheap, and reliable to produce data from anything, starting from fetal heartbeats transmitted via the mother's clothes through conductive material to the aircraft engines flying in the sky.

2.2.2. The Network Layer

A method of transmitting electrical signals. Wi-Fi networks, for example, can provide transmission capabilities ranging from 0.3 to 1 Gbps with wide-spread transmission.

Sensor-generated data rarely reaches its full potential at production. The signals will frequently be relayed to different sites for gathering and analytical processing. This

usually includes sending data through a network. Various networking ends, such as routers, hubs, gates and switching devices, are used to link sensors and other ends on the network. Based on their functionality, computers, mobile devices, and others, for example, are frequently accessing the web via a networking device, such as Wi-Fi.

The primary stage in transferring information from end to end within a network is to recognize every machine individually. Each "thing" on a network must have a separate identity in IoT. A group of protocols that determine how devices can contact one another is known as a network protocol. These protocols have two types in general; private or public.

Private communication protocols allow computers that have special equipment to be identified and authorized, so the customizing process can be more feasible, and to allow the producers to specialize their offers. Interoperability between heterogeneous devices is enabled through open protocols, which improves scalability.

IP or Internet Protocol is a standard protocol for allocating distinctive addresses to different Internet connected devices. IPv4 (Internet Protocol version 4) and IPv6 (Internet Protocol version 6) are the two types of IP that are currently used. IP was originally implemented to identify computers, before it became extended to address other objects as well. Around 4 billion IPv4 addresses have been allocated out of 6 billion addresses. IPv6 is more scalable than IPv4 with about 3.4x1038 unique addresses against IPv4's 6 billion. IPv6 adoption has been critical in facilitating the IoT, with the rapid growth number of linked devices.

2.2.3. The Application Layer

The application layer works as a connector between the IoT unit and the network it will communicate with. It handles data processing and presentation and acts as a link between what the IoT unit is doing and the produced data being transferred over the network.

Action restrictions or prescriptions that are widely recognized. Technical standards, for example, enable data processing and aggregated data set compatibility. We may see requirements from industry regulation contributors and/or standards regulatory organizations for the IoT standards in the near future. Analytical methods, i.e. *Augmented intelligence,* that help to better characterize, predict, and exploit correlations between

phenomena. For example: Even when provided with unorganized information, petabyte-size data can now be queried and studied.

Technologies and methods that increase adherence to specified actions, aka augmented behavior. Machine-to-machine connections, for example, are removing flawed human interaction from otherwise effective operations. Information on human-related cognitive biases is turning augmented intelligence-based action prescriptions more effective and reliable.

2.2.4. Network technologies for an IoT Environment

Wired and wireless networks are the two types of network technologies. Wireless networks offer ease via near-continuous connectivity, but wired connections are still necessary for much more robust, high-volume and secured network routes due to the constant mobility of users and devices. (Buczak & Guven, 2016)

Choosing the technology of the network is mostly decided by the geographical region that will be served. Devices may communicate over short ranges (e.g., inside a room) using wireless personal area network (PAN) innovations like Bluetooth and ZigBee, as well as cable connections leveraging technologies such as Universal Serial Bus (USB). Local area network (LAN) solutions enable devices to transfer data across a relatively broad area, as in an apartment. Fiber optics and Ethernet are two technologies used in wired LANs. Examples of wireless LAN networks include Wi-Fi and several other wireless LAN-based technologies.

When data must be carried over a larger region than buildings and cities, a WAN is established by linking a number of LAN networks via routers. A WAN is much like the Internet. When looking for a network solution for a certain situation, data transmission rates and energy demands are two important factors to take into account. Due to the high transmit capability, 4G /5G solutions are optimal for IoT systems. For energy-constrained devices, technologies like Low-Powered Bluetooth and Low-Powered WiFi are suitable.

Bluetooth-Based Solutions

Bluetooth is a wireless technology that was first introduced in 1999 and is notable for the capability to transport information within small areas. (History of the Bluetooth special interest group, 2015) Bluetooth (Low-Energy) or BLTE is a relatively new enhancement that uses around ½ of the consumption of the original Bluetooth solution.

(A look at the basics of Bluetooth technology, 2015) The lower searching duration required for BLTE units to detect other devices (0.6 to 1.2 milliseconds ms vs. 22.5 ms for Bluetooth Classic) accounts for BLTE's energy efficiency. Furthermore, when compared to Bluetooth Classic, the efficient transfer of data during the sending and receiving states allows BLTE to achieve higher energy efficiency.

Lower data speeds are given for greater energy efficiency: BLTE provides 0.260 Mbps, however the original Bluetooth can provide speeds up to 2.1 Mbps. Because of its widespread adoption and inexpensive device prices, BLTE is an ideal technology for IoT applications. Interoperability, on the other hand, is a persistent bottleneck: Only the newest Bluetooth devices with dual modes (they support both the classic Bluetooth and BLTE) are compatible with BLTE; legacy Bluetooth Classic devices are incompatible. (Tauchmann & Sikora, 2015)

Wi-Fi and Low Power Wi-Fi

Despite the fact that Ethernet came out around the 70s, Wi-Fi is a newer wireless solution, and also has a good popularity for the high data transmit rates in both the PAN and LAN networks. Wi-Fi devices often maintain a minimal latency in data transfer by being powered on during periods of inactivity. These types of Wi-Fi solutions are typically supplied with a dedicated power source or batteries that must be recharged every few hours of operation. When not transmitting data, more costly, low-power Wi-Fi devices "hibernate" and may "wake up" in less than 10 milliseconds when contacted. (Dobkin & Aboussouan, 2009) Low Power Wi-Fi with batteries may be utilized for remote sensing and control applications. For telemetry and control applications, low-power Wi-Fi or batteries are a viable choice.

The Long-Term Evolution Technology or LTE

Long Term Evolution is a wireless wide-area network technology developed by members of the Third Generation Partnership Project in 2008. This new technology is capable of transmitting data at speeds of up to 300 megabits per second. LTE-Advanced (LTE-A) is a relatively recent addition to the LTE family of technologies, enabling data speeds of up to 1 Gbps in comparison to the conventional 300 Mbps of LTE.

2.3. Unique Challenges in IoT Security

In cyberspace, securing IoT devices remains a serious concern. Millions of devices (e.g., cameras) with vital ports (e.g., like 143 for IMAP or 445 for MDS) have been discovered using tools like Shodan, an IoT device search engine. Default login credentials are still used by many of these devices. In reality, with the rapid growth of the Internet of Things, insecure protocols like Telnet are being reintroduced. Furthermore, many of these Internet-enabled devices use wireless communication methods (Zeadally & Tsikerdekis, 2020).

The attackers utilized the device to get access to the internal network and download data from the high-roller database, which contained confidential information about the casino's wealthiest guests. Infiltrated IoT devices have a far-reaching impact that extends beyond the side that hosts the devices on their networks. Compromised ends can be used to perform large-scale denial-of-service (DoS) assaults on some other systems as part of a botnet (MT, C, & D., 2017). The cost of such attacks on businesses and society as a whole has been assessed in a recent report. IoT hacks cost small businesses in the United States 13% of overall annual revenue (Help Net Security, 2017)

Strong, cost-effective IoT security is more difficult to implement than traditional information security for a variety of reasons, including (L & EC, 2018):

(a) the physical environment in which these IoT devices operate. Many IoT devices now use wireless connections that may be accessible from outside the internal perimeter;

(b) standard IT systems are often updated and vulnerable patched. In the case of consumer IoT security, however, automatic firmware/software upgrades and patching techniques are not properly established;

(c) the limited hardware, size, and computing power of many IoT devices make it difficult to apply the same security controls that are deployed on traditional networked computing systems; in many cases, the user is left to do the updates or patching, which is in most cases ineffective (Zeadally & Tsikerdekis, 2020).

2.4. Security Threats in IoT

The Internet of Things (IoT) systems, which include devices, networks and services, are susceptible to physical, application and network threats, as well as privacy breaches. (Xiao, Wan, Lu, Zhang, & Wu, 2018).

In general, IoT devices operate in a variety of environments to achieve a variety of objectives. Their functioning, On the other hand, must satisfy a complete security requirement in both the physical and cyber domains. The Internet of Things is a complex and diverse field of study. As a consequence, balancing security requirements with the IoT system's huge attack surface is challenging. The solution should take into account all aspects of security in order to meet the necessary level of protection. IoT devices, on the other hand, are typically used in an unattended setting. As a consequence, an attacker may be able to physically access these devices. IoT devices are generally linked through wireless networks, which exposes sensitive information if an intruder eavesdrops on the communication channel. Because of their finite compute and power resources, IoT devices are unable to handle complicated security systems. As a result, protecting the IoT system is a hard and comprehensive job. Because the major goal of the IoT system is for it to be accessible to anybody, everywhere, and at any time, cyberattacks and surfaces become exposed to attackers.

The below sections describe some of the attacks that the attackers have been using in the recent years.

Attackers may utilize distributed denial of service (DDoS) attacks to overwhelm the targeted database with unauthorized requests, thereby blocking IoT devices from accessing different services. Bots are malicious queries created by an Internet of Things (IoT) network of devices (Bertino & Islam, 2017). DDoS attacks can deplete all of the service provider's resources. It has the ability to block legitimate users and make network resources unavailable.

Threats on the physical layer of IoT environments are called RFID attacks. The device's integrity is compromised as a result of this attack. Attackers try to alter data while it is stored on the node or while it is being sent over the network. At the sensor node, popular vulnerabilities include attacks on availability, attacks on authenticity, attacks on

confidentiality, and brute-forcing cryptography keys (Zhang & Green, 2015). Password protection, data encryption, and limited access control are all countermeasures to assure the prevention of such attacks.

Internet-based attacks: IoT devices can connect to the Internet to access a variety of resources. Attackers employ spamming tactics to steal data from other systems or to ensure that their target website is viewed on a regular basis (Kim, Jeong, C. Kim, & So, 2011). Ad fraud is a popular strategy used for this. It creates phony clicks on a certain website for monetary gain. Cyber criminals are a group that practices these techniques.

NFC Threats: The majority of these threats are focused on financial fraud. Unencrypted transport, eavesdropping, and label alteration are all plausible assaults. Conditional privacy protection is the answer to this dilemma. As a result, the attacker is unable to generate the identical profile using the user's unique identifier. A trustworthy service manager generates random public keys for this model (Eun, Lee, & Oh, 2013).

2.5. Intrusion Detection System (IDS)

The IDS is a solution that observes and gives insights on network and system activities in order to detect intrusions or attacks. Internal or external incursions are possible. Internal incursions are carried out by authorized customers who want to increase their access privileges by misusing unauthorized privileges. External intrusions, on the other hand, are carried out by people from outside the network, who try to get unwanted access to the system (Huang, 2003). When an attack is discovered, the IDS monitors the host or network's event sequences and creates an alarm.

The intrusion detection systems can be divided into two types:

- Host-based IDS (HIDS) is a type of attack detection technology that connects to a host or device and monitors suspicious activities from a variety of sources, including file system records and process operations.
- Network-based IDS (NIDS) use sniffing tools like TCPDUMP to track network traffic data.

IDSs can be categorized into three groups (Sherasiya, Upadhyay, & Patel, 2016), (Zarpelão, Miani, & de Alvarenga, 2017) :

- IDS with signatures; identifies intrusions based on common threats. It's quite easy to use. It is simple to use, however the more the threats grow, more processing and space is required. The main disadvantage of such a system is that it only discovers incursions that have signatures (Liao, Richard Lin, Lin, & Tung, 2013). As a result, new attack signatures must be added to the database on a frequent basis.
- Anomaly Based IDS: creates regular usage patterns within the outset, analyzes system actions to a regular usage pattern, and notifies the admins when the difference exceeds a threshold (Liu, Xiao, & Chen, 2012), (Zarpelão, Miani, & de Alvarenga, 2017). This method is useful for finding unfamiliar threats, but it produces a lot of mistaken results. The reason for that is a divergence from usual activity does not always imply a threat.
- Due to the complementary nature, specification-based IDS utilize both of the preceding methodologies (Zarpelão, Miani, & de Alvarenga, 2017). On the one hand, it uses both methodologies to detect undiscovered threats while reducing false positives on the other. However, in terms of resource usage, this process is quite costly.

2.6. Machine Learning

Machine learning is an artificial intelligence technique that trains machines using a variety of algorithms and enables devices to learn from their experiences rather than being explicitly programmed (Jordan & Mitchell, 2015). ML does not require human intervention, complex mathematical formulae, and can operate in dynamic networks.

Due to the unique way in which learning algorithms solve problems, they have been rapidly used in a broad variety of real-world applications. These algorithms are used to design machines that learn continuously through experience. Learning methods have been widely utilized in practice in recent years. Development of new algorithms and

access to large datasets have fueled recent advancements in learning algorithms, as well as the introduction of low-cost algorithms (Jordan & Mitchell, 2015).

In general, learning algorithms seek to improve a task performance by training and learning from the experience. For example, in intrusion detection systems, the purpose is to identify behavior of the system as abnormal or normal. A performance improvement can be obtained by increasing the accuracy of the classification, and the experiences where the algorithms train and learn from a collection of normal system behavior.

Learning algorithms can be classified into four categories; supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning algorithms.

Supervised learning takes place when a specified set of inputs is used to achieve a specific set of goals. This form of learning begins with labeling the data, followed by training on the labeled data (providing inputs as well as intended outputs). It aims to automatically discover rules from provided datasets, create multiple classes, and eventually predict whether elements (individuals, objects and criteria) belong to a specific class.

Unsupervised learning occurs when the environment offers just inputs without a relation towards the intended output. Unsupervised learning does not need labeled data; instead, it can look for patterns in unlabeled data and use those patterns to classify the data into different categories.

While supervised and unsupervised learning approaches are mostly used for solving data analysis problems, reinforcement learning is recommended for solving comparison and decision-making challenges. This categorization and related choice of machine learning technique is determined by the nature of the available input data. Supervised learning is applied when the kind of the input data and the expected outputs (labels) are known. In this case, the system needs to be trained to just map inputs to desired outputs. Regression and classification are examples of supervised learning approaches, the difference between the two is that regression can work with continuous data and classification works with discrete data. Support Vector Regression, polynomial regression and linear regression are some of the most popular approaches used for

regression. By comparison, classification utilizes discrete data values (class-labeled data). Support Vector Machine, Logistic Regression and K-Nearest Neighbor are some of the most often used classification techniques. Some algorithms, such as neural networks, may be applied for both regression and classification. Unsupervised learning techniques are used to train systems when the outcomes are not specifically defined and the system must identify the pattern within the raw data. Clustering is a kind of unsupervised learning in which items are grouped according to specified similarity criteria, – for example K-means clustering. The accuracy of predictive analytics is determined by how effectively the machine learning technology utilizes previous data to construct models and how well it predicts future values. Naive Bayes, neural networks SVR are some of the predictive modeling algorithms.

Semi-supervised Learning: in supervised and unsupervised learning, either all observations in the dataset are labeled or all observations do not have labels. In between these two, semi-supervised learning takes place. In many practical cases, the cost of labeling the data is fairly significant, since it needs the expertise of trained human specialists. Thus, when most observations lack labels but present in some, semi-supervised techniques are the best options for model building.

In Reinforcement Learning (RL) agents do not have any predefined objectives and must adapt to their environments based on what they learn through their interactions. It does certain activities and makes decisions based on the reward received. An agent may be rewarded for doing good actions or punished for doing bad actions, and the feedback criteria can be used to maximize the long-term rewards for an agent's performance. It is significantly influenced by observation of human and animal behaviors. These characteristics make it an appealing option for highly dynamic robotics applications where the system can learn to perform complex tasks without explicit instructions (Ambedkar, 2017). It is also critical to choose an appropriate reward function, since the agent's success or failure is based on the cumulative total reward received (Wirth, Akrour, Neumann, & Frnkranz, 2017).

2.7. Deep Learning

Deep Learning (DL) is a machine learning approach derived from artificial neural networks (ANN). The neural network is made up of neurons (which can be seen as

variables) that are linked using weighted connections (can be considered as parameters). An unsupervised or supervised learning approach is used in conjunction with the network in order to reach the expected set of outputs. The learning process begins with unlabeled and labeled data obtained by unsupervised or supervised learning approaches, respectively, and is followed by iterative adjustments of the weights between all pairs of neurons. Because of this, while addressing deep learning, we refer to deep large neural networks that have a high number of layers where the term "deep" refers to how many layers there are in a network (al., 2014), (al., 2015).

DL is well known for its distributed computing capabilities, as well as for the analyzing and learning from large amounts of unlabeled, unsupervised, uncategorized data. It provides a hierarchy shaped model of learning as well as feature representation that is inspired by the human brain's layered learning process. By improving classification modeling and creating better data samples, deep learning models contribute to a variety of machine learning applications such as voice recognition, computer vision, and natural language processing. Additionally, these models enhance data compression and restoration in both the spatially and temporally domains due to their ability in extracting features and patterns from huge volumes of data.

There are various deep learning architectures available, including Convolutional Neural Networks (CNN), Generative Adversarial Networks (GAN), Recurrent Neural Networks (RNN), Long-Short Term Memory (LSTM), Networks Boltzmann Machines (BM), Feed forward Deep Networks (FDN), and Deep Belief Networks (DBN). The most commonly used DL architectures are CNN (used in spatially-distributed data) and RNN (for time series data).

2.8. Machine Learning Techniques Used in IoT Security

In the last several years, machine learning algorithms have made significant advancements in the IoT security field. As a result, machine learning techniques may be used to identify potential IoT threats early on by examining the behavior of the devices. Additionally, suitable solutions may be supplied for resource-constrained IoT devices employing a variety of machine learning methods. ML techniques such as supervised learning, unsupervised learning, and reinforcement learning could be used to detect intelligent attacks on IoT devices and construct a solid defensive strategy. Supervised machine learning techniques

Supervised machine learning techniques can be used for labeling the network for attack detection, models such as neural networks (NN), K-nearest neighbor (K-NN), support vector machines (SVM), random forest and naive Bayes are utilized. These models effectively identified DoS, DDoS, intrusion, and malware threats in IoT devices.

The subsections that follow will highlight various types of supervised techniques utilized in IoT security.

2.8.1.1. Support vector machine (SVM)

The SVM method is used to perform classification and regression analysis on data. Between two classes, SVM builds a plane called a hyperplane. The hyperplane aims to maximize the distance between each class that differentiates each class with the minimum possible errors at the maximum possible margin (Buczak & Guven, 2016). If the hyperplane has become nonlinear as a result of the analysis, SVM employs the kernel function to linearize it by adding new features. Sometimes, it is difficult to select the optimal kernel function in SVM. However, SVM has a high level of accuracy, which makes them useful for IoT security applications such as intrusion detection (Liu & Pi, 2017), malware detection (Ham, Kim, Kim, & Choi, 2014), and smart grid attacks (Karimipour & Dinavahi, 2017).

2.8.1.2. Bayesian theorem

The Bayesian theorem is based on the Bayesian probability theorem for learning distributions. Using Bayesian probability, this type of supervised learning approach generates new outcomes based on existing data. This is referred to as Naive Bayes (NB). As a result, NB has been a commonly utilized learning algorithm since it requires prior knowledge in order to incorporate Bayesian probability and predict possible outcomes. This is one of the difficulties that IoT can address effectively. NB is often used in the Internet of Things to detect intrusions at the network layer (Panda & Patra, 2007) and anomalies (Agrawal & Agrawal, 2015). NB offers a number of benefits, including being straightforward to understand, needing less data for classification, being simple to execute, and being relevant to multistage calcification. NB is dependent on features, their interactions, and prior knowledge, all of which may work against obtaining a correct result (Box & Tiao, 2011).

2.8.1.3. K-nearest neighbor (KNN)

KNN is a statistical nonparametric technique used in supervised learning that is often based on Euclidean distance. (Chen et al., 2015). In KNN, the Euclidean distance between two nodes determines the average value of the unknown node, which is the sum of its k closest neighbors (Deng, Zhu, Cheng, Zong, & Zhang, 2016). For example, if a node is lost, the next neighbor's average value may be used to predict the lost value. Although this number is imprecise, it helps in identifying the potential missing node. In the Internet of Things, the KNN approach is utilized for intrusion detection, malware detection, and anomaly detection. The KNN algorithm is straightforward, cost effective, and simple to implement (Adetunmbi, Falaki, Adewale, & Alese, 2008). In contrast, identifying the missing nodes is a time-consuming operation that is tricky in terms of precision and accuracy.

2.8.1.4. Random forest (RF)

RF is a unique machine learning technique that uses a couple of Decision trees (DT) to develop an algorithm for generating an accurate and robust estimating model for outcomes. These trees are generated randomly and trained towards a certain behavior that becomes the model's overall outcome. While RF makes use of DTs, the learning procedure is different since RF takes the average of the output into account and requires less inputs (Cutler & al., 2007). RF is often employed in network surface attacks to identify DDoS attacks (Doshi, Apthorpe, & Feamster, 2018), anomaly detection (Chang, Li, & Yang, 2017), and unauthorized IoT device identification (Meidan & al., 2017). Previous research indicates that RF outperforms SVM, ANN, and KNN in detecting DDoS attacks (Doshi, Apthorpe, & Feamster, 2018). Despite the fact that RF is not suitable for real-time applications, it requires a greater number of training datasets to create DTs capable of detecting intrusions.

2.8.1.5. Association rule (AR)

AR is also a kind of supervised machine learning technique that is used to find the unknown variable based on their mutual relationship in a given dataset (Tajbakhsh, Rahmati, & Mirzaei, 2009), the AR approach was successfully applied for intrusion detection, where fuzzy AR has been used to identify network intrusions. AR is straightforward and simple to implement; however, it is not widely employed in IoT because of its high temporal complexity and depend on assumptions which may not

provide an accurate result for a big and complicated model (Kotsiantis & Kanellopoulos, 2006).

2.8.1.6. Decision Tree (DT)

DT is a kind of natural supervised learning that resembles a tree with branches and leaves. Different branches serve as edges while leaves serve as nodes in DT. DTs usually are used to sort the sampled data according to the featured values. DT in machine learning is classified mainly as regression and classification (Kotsiantis S. , 2013). DT offers many benefits over other machine learning algorithms, including its simplicity of design, ease of implementation, ability to handle massive data samples, and transparency. By comparison, this technique has a number of drawbacks, including the fact that it requires a large amount of storage space due to its structure. This complicates the learning method if many DTs are considered to solve the problem (Kotsiantis, Zaharakis, & Pintelas, Supervised machine learning: a review of classification techniques, 2007), (Quinlan, 1986). DTs are often employed as classifiers in security applications such as DDoS detection and intrusion detection (Kim, Lee, & Kim, 2014).

2.8.1.7. Neural network (NN)

The NN technique is based on the structure of the human brain, which is composed of neurons. NN makes extensive use of machine learning methods that allow it to handle complicated and nonlinear problems (Gondhi & Gupta, 2017) , (Hush & Horne, 1993). The two primary network types in NN algorithms are hierarchical and interconnected networks, which are based on multiple functional layers of the neuron (generally: input, hidden, and output layers). NN approaches reduce the network's response time, which improves the IoT system's performance. NNs, on the other hand, are computationally demanding and difficult to deploy in distributed IoT systems.

2.8.2. Unsupervised machine learning techniques:

Unsupervised learning is characterized by the absence of output data for the provided input variables. The majority of the data is unlabeled, and the system attempts to discover patterns across the datasets. As a result, it splits them into different clusters. Several unsupervised learning techniques have been applied to the security of IoT devices in order to identify denial-of-service (DoS) attacks (through multivariate correlation analysis) and to protect users' privacy (by the infinite Gaussian mixture model (IGMM)) (Tan, Jamdagni, He, Nanda, & Liu, 2013). The next subsection will discuss unsupervised

learning techniques such as K-means Clustering and Principal Component Analysis (PCA).

2.8.2.1. Principal Component Analysis (PCA)

PCA, also known as feature reduction, reduces large datasets into smaller ones while retaining the same amount of useful information as the original. Thus, PCA reduces a system's complexity. This approach may be used to select features for detecting real-time intrusion threats in IoT systems (Wold, Esbensen, & Geladi, 1987). A combination of PCA and other machine learning techniques may be used to create a robust security protocol. A model provided by (Zhao, Li, Zia, & Zomaya, 2017) optimizes the system via the use of PCA and classifier techniques including KNN and softmax regression.

2.8.2.2. K-mean clustering

This unsupervised learning approach divides data samples into small groups in order to cluster them. This is also known as a clustering algorithm. There are a few simple rules to follow when implementing this method, including the following: 1) Divide the given dataset into multiple clusters, each with its own centroid (k-centroid), with the primary goal of determining the k-centroid of each cluster; 2) Then, choose a node of each cluster and relate it to the nearest centroid, repeating this process until all nodes are contacted. The technique is then recalculated using the average value of each node in each cluster; 3) Finally, the approach repeats its previous stages until they coincide to provide the K-mean value. K-mean learning algorithms are very advantageous for locating suitable living zones in smart cities. Due to its simplicity, K-mean approaches are also effective in IoT systems where labeled data isn't really necessary. However, this technique for unsupervised learning is less effective than supervised learning. The K-mean clustering approach is frequently used in anomaly and Sybil attack detection (Xie, Huang, Bai, & Hu, 2017).

2.8.3. Reinforcement machine learning techniques:

These models allow an IoT system to choose security protocols and key parameters via trial and error in order to defend against various threats.

Several IoT devices (sensors, electric glass, and air conditioners, for example) employ reinforcement learning to adapt to their environment. Additionally, RL methods like Dyna-Q, deep Q-network (DQN), Q-learning and post-decision state (PDS) have

been utilized to secure IoT devices, detecting different IoT attacks and providing appropriate security protocols for IoT devices. Q-learning has been utilized for authentication, jamming attacks, and malicious inputs in (Xiao, Li, Han, Liu, & Zhuang, 2016) and (Xiao, Li, Huang, & Du, 2017). while Dyna-Q has been used for malware detection and authentication. Additionally, PDS and DQN are both capable of providing protection against malware detection and jammer attacks, respectively (Han, Xiao, & Poor, 2017).

2.9. Deep learning Techniques Used in IoT Security

Recently, the application of deep learning to IoT systems has become a significant subject of research. The major benefit of deep learning over traditional machine learning is its higher performance on huge datasets. Many IoT devices generate significant amounts of data; hence, DL techniques are well suited for such systems. Additionally, DL can extract complicated representations from data automatically. Deep linking of IoT ecosystems is possible using DL methods. Deep linking is a uniform protocol that enables devices and applications connected to the Internet of Things to communicate automatically and without human intervention. For instance, IoT devices made for smart home applications may interact automatically to create a completely smart home (Li, Ota, & Dong, 2018).

DL is a machine learning sub-field that makes use of many non-linear processing layers to abstract and manipulate discriminative or generative features for pattern analysis. Due to the fact that deep learning techniques are able to capture hierarchical representations in a deep architecture, they are sometimes referred to as hierarchical learning methods. DL's working concept is inspired by how the human brain and neurons process information. Deep networks are used for supervised learning (discriminative learning), unsupervised learning (generative learning), and a mixture of these two forms of learning, referred to as hybrid deep learning. Recurrent neural networks (RNN) and convolutional neural networks (CNN) are two types of discriminative deep learning (DL) algorithms. Generative adversarial networks (GAN), deep belief networks (DBN), deep autoencoders (AE) and restricted Boltzmann machines (RBM) are examples of hybrid DL techniques.

2.9.1. Convolutional neural networks (CNNs)

CNNs were developed to minimize the number of data parameters required by a conventional artificial neural network (ANN). Three techniques are used to limit the number of data parameters: sparse interaction, equivariant representation and parameter sharing, and (Goodfellow, Bengio, Courville, & Bengio, 2016). Reducing the number of connections between layers enhances the scalability and complexity of a CNN's training time. The primary advantage of CNNS is that it is widely applicable to DL training methodologies. Additionally, it allows high-performance automated learning of features and patterns from raw data. On the other hand, implementing CNNs on devices with constrained resources to enable onboard security features is challenging due to the high computational cost of CNNs.

CNNs have shown their robustness in a broad variety of use cases. For example; In IoT security, research (al. N. M., 2017) presented a CNN-based malicious detection approach for Android systems. CNN is used to automatically learn important features associated with malware detection from the data, completely removing the necessity for manual feature extraction. The key aspect about utilizing CNNs is that it is trained to learn relevant features and perform classification, thereby avoiding the extraction process needed in conventional machine learning and hence delivering an end-to-end model (al. N. M., 2017). However, attackers may use the CNNs' superior learning ability as a weapon. A prior work (Maghrebi, Portigliatti, & Prouff, 2016) showed that a CNN approach is capable of effectively breaking cryptographic implementations.

2.9.2. Recurrent neural networks (RNNs)

RNNs have been proposed as a method to deal with sequential data. Forecasting current output in various applications is based on analyzing relationships between multiple prior samples. Therefore, the neural network's output is dependent on its past and present inputs. A feed-forward NN is ineffective in such scenarios since the association between input and output layers is retained independent (Pascanu, Gulcehre, Cho, & Bengio, 2013). As a result, when the backpropagation algorithm was developed, its most notable use was for training RNNs. RNNs are suggested for applications that need sequential inputs (e.g., text, speech, and sensor data) (Hermans & Schrauwen, 2013).

RNNs can also be used to secure IoT devices. IoT devices produce high volumes of sequential data from a variety of sources, including network traffic flows, as these are

crucial for detecting a range of possible network attacks. For instance, recent research (Torres, Catania, Garcia, & Garino, 2016) examined the possibility of using an RNN to analyze network traffic behavior in order to identify possible attacks (malicious behavior) and proved the RNN's use in classifying network traffic for malicious behavior detection. Therefore, RNNs can provide feasible options in real-world situations. Studying RNNs and their variations is critical for enhancing IoT system security, particularly against time series-based attacks.

2.9.3. Deep autoencoders (AEs)

As an unsupervised learning approach, an autoencoder is a neural network method that incorporates encoding and decoding techniques. The autoencoder is trained to recreate the output as closely as possible to the original input. (More information can be found in section 2.10).

2.9.4. Restricted Boltzmann machines (RBMs)

Restricted Boltzmann machines are unsupervised generative models (Hinton, 2012). In an RBM, vertices in the same layer are not connected to one another since the model is completely undirected. RBMs are made up of two types of layers: one that is visible, and one that is hidden. The visible layer includes the actual inputs, whereas the hidden layer is made up of multiple layers that contain latent parameters. RBMs learn features from data in a hierarchical form, where the features obtained in the first layer are passed to the next layer as latent parameters.

RBMs can be used to develop network anomaly detection systems as the research in (Fiore, Palmieri, Castiglione, & Santis, 2013) proposes a model that is based on a discriminative RBM.

2.9.5. Deep belief networks (DBNs)

Deep belief networks are considered as generative algorithms (Hinton, Osindero, & Teh, A fast learning algorithm for deep belief nets, 2006). To achieve robust performance, a DBN is constructed of layered RBMs that undertake greedy layer-wise training in an unsupervised environment. In a DBN, training is carried out layer by layer, with each layer being implemented as an RBM trained layer on top of the previously trained layer.

DBNs have been effectively used to identify malicious attacks. A previous work (Chen, Y. Zhang, & Maharjan, 2017) offered a method for securing mobile edge computing using a deep learning-based technique for detecting malicious attacks. Automatic detection was performed using a DBN, and the suggested DBN-based model demonstrated a significant increase in the accuracy of malware detection when compared with conventional machine learning-based techniques (Chen, Y. Zhang, & Maharjan, 2017). This finding established the superiority of deep learning in general, and DBNs in particular, over manual feature engineering techniques in malware detection.

2.10. Auto Encoder

As an unsupervised learning approach, an autoencoder is a neural network structure that incorporates encoding and decoding techniques. The autoencoder is trained to recreate the output as closely as possible to the original input. As Figure 2.1 shows, the autoencoder is composed of three layers: input, output and hidden layers. One of the advantages of an autoencoder is that it is more powerful than Principal Component Analysis in identifying data structures via on-linear transformations (Wang, H. Ya, & Zhao, 2016). The procedure is implemented using the encoder-decoder methodology by applying backpropagation, the autoencoder adjusts the target output to match the inputs. During this procedure the input will be compressed (encoded) into a lower dimension and then extended to reconstruct the original data (decoder). The output of one layer is transmitted to the following layer once it has been trained. This method intends to reduce the dimensionality of the input data to achieve a better representation of the data. The intermediate encoded layer (latent layer) of the autoencoder is utilized to extract a feature for classification (Wang, H. Yao, & Zhao, 2016).

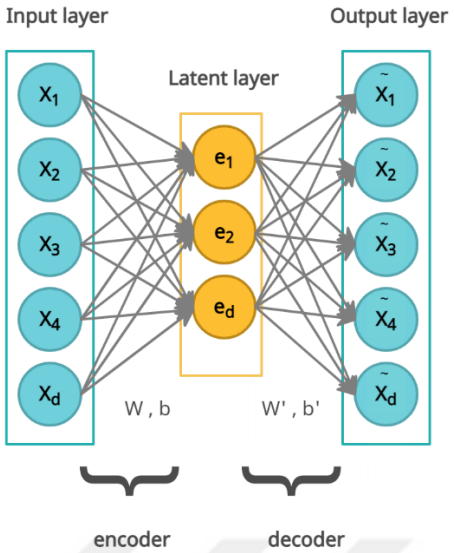

Figure 2.1. *The structure of AE.*

Assume that the output and input layers each have N nodes and the hidden layer has K nodes. For a training dataset X of m samples, the encoder applies the mapping function M to perform the mapping of the input vector to the hidden vector.

$$Z = M(wx + b)$$

where M is the activation function, w denotes to the matrix of weights, and b indicates the bias vector. The decoder converts z to x'.

$$x' = M'(w'Z + b')$$

Where M', w', and b' are the reconstruction parameters for Z. The objective of training AE is to minimize the reconstruction loss between the input and output. Therefore, the following equation is used to calculate the loss function.

$$L(x, y) = \frac{1}{m}\sum_{i=1}^{m}\|x_i - y_i\|^2$$

Stacked AE, Sparse AE, and Denoising AE are all variations of autoencoder. A Stacked Autoencoder (SAE) is a multi-layer auto-encoder in which each layer's output is linked to the inputs of the following layer to progressively compress the information and generate a new representation (Berman, Buczak, Chavis, & Corbett, 2019). Sparse AE constrains the sparsity of the hidden units where only a subset of the hidden units can be

active at any given instant. Denoising autoencoder rebuilds the original data from damaged input, helping in the discovery of robust representations and preventing the autoencoder from learning the less important features.

3. RELATED WORK

Studies have extensively constructed intrusion detection systems using a variety of different machine learning techniques, including k-nearest neighbor (KNN) (Li, Yi, Wu, Pan, & Li, 2014), Random Forest (RF) (Farnaaz & Jabbar, 2016), SVM (Reddy, Ramadevi, & Sunitha, 2016), Artificial Neural Networks (ANN) (Ingre & Yadav, 2015), Naive Bayes (NB) (Koc, Mazzuchi, & Sarkani, 2012) and others (Buczak & Guven, 2016), (Khan & Jain, 2016). While shallow learning technologies and classic machine learning methods have increased detection performance and lowered false alarm rates, the detection effectiveness of such methods is very feature-dependent.

Deep learning methods have been proven to be successful in speech recognition, image classification, and a variety of other areas, and have emerged as the preferred approaches for a wide variety of problems. These techniques have steadily been utilized in the intrusion detection fields achieving outstanding results. (Javaid, Niyaz, Sun, & Alam, 2016) present a technique based on deep learning for constructing an efficient NIDS. They perform network intrusion detection using Self-taught Learning (STL) on the NSL-KDD dataset. They achieved an accuracy of 88.39 % for binary classifications and 79.10 % for multiple classifications. (Yin, Zhu, Fei, & He, 2017) introduced a technique for detecting intrusions based on recurrent neural networks (RNN-IDS). using the NSL-KDD dataset, the achieved accuracy score is 81.29 %, which is better than that of previous machine learning approaches such as SVM and Random Forest.

As the number of features grows, the required training time of an algorithm grows as well, which might not reflect directly in the results. Some features may produce biases that are unrelated to the class. As a result, feature selection techniques are often utilized to increase supervised learning performance.

H.Zhang et al., (2018) introduced a DL based method for intrusion detection that utilizes denoising autoencoders to select features. This approach is primarily composed of two models that use deep learning to conduct feature selection as well as classification. The denoising autoencoder selects the features then the classification is carried out using a Multilayer perceptron model with the minimum number of features while maintaining a high degree of accuracy. The UNSW-NB15 dataset is used to evaluate the performance

of the proposed model. The results indicate that feature selection delivers a high accuracy performance while using less memory and computational resources.

Farahnakian and Heikkonen, (2018) utilized a denoising autoencoder model to extract relevant features from an unbalanced training dataset and then develop a model to identify normal and abnormal patterns. The model is trained using the denoising autoencoder then a layer that contains a softmax classifier is added at the end to realize the classification and generate the classification results. The KDD-CUP'99 dataset is used to evaluate the performance of the proposed model. The approach shows good results with accuracy of 94.71%.

Shone et al., (2018) describe a unique DL based approach for intrusion detection that employs an unsupervised feature extraction algorithm called the nonsymmetric deep autoencoder (NDAE). Additionally, this model employs Random Forest classifier incorporated with the multiple layers of NDAEs. The proposed model has shown good results when applied to the NSL-KDD and KDD-Cup'99 datasets. The findings indicate that this technique delivers a high degree of accuracy, precision, and recall while requiring less training time.

Lin et .al, (2018) proposed a convolutional neural network (CNN) based approach to implement instruction detection. The approach is divided into three stages: extraction of features, training of the model, and model verification. The proposed model is evaluated using KDD-Cup'99 dataset.

4. PROPOSED INTRUSION DETECTION SYSTEM

4.1. Introduction

Figure 4.1 illustrates the architecture of the proposed model. It consists mostly of three stages: (1) Data preprocessing: which includes scaling data using min-max normalization technique. The categorical features are then converted to values 0 and 1 using the one-hot encoding method. (2) A Deep Autoencoder (DAE) will be trained to feature extraction. (3) Detecting attacks: utilizing the DAE's latent layer to initialize the weights of the supervised learning stage, which applies several Machine learning classifiers to detect attacks.

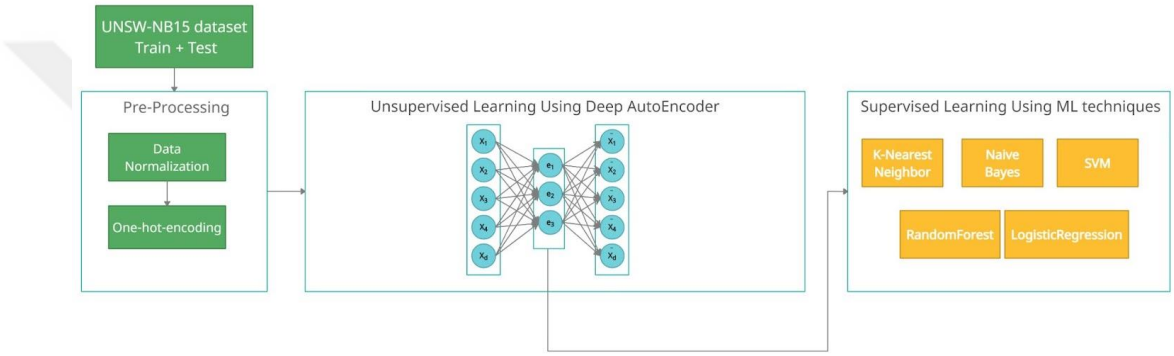

Figure 4.1. *The structure of the proposed intrusion detection model*

4.2. Data Preprocessing

IDS necessarily requires data preprocessing. It consists of two modules: data normalization and one-hot encoding.

4.2.1. Data normalization

Machine learning methods are used to extract patterns in the dataset by comparing data points from different dimensions. A challenging issue when attempting to employ machine learning is that there are dimensions with vastly different scales. The UNSW-NB15 dataset has 39 dimensions with widely varying values.

The min–max normalization approach is used in this study to decrease the various dimension scales. The original data is scaled to the interval [0, 1] using a linear transformation. The min-max algorithm transforms data using the following equation:

$$\tilde{x}_{ij} = \frac{x_{ij} - min(x_i)}{max(x_i) - min(x_i)}$$

where $min(x_i)$ and $max(x_i)$ are the minimum and maximum values of the i th numeric feature x_i, respectively, and $\tilde{x}_{ij}$ denotes that the normalized feature value is between [0,1].

4.2.2. One-Hot encoding

One-hot encoding is the most widely used approach to dealing with the numeralization of categorical features since it is simple and effective. It can transform each value of a category attribute to a binary vector, with just one element with the value 1, while the rest are zeros. The presence of a matching possible value of a category property is indicated by the element with value 1. Suppose that the categorical feature color has just three potential values—say, blue, red, and green—with one-hot encoding, the blue may be encoded to (1, 0, 0), the red to (0, 1, 0), and the green to (0, 0, 1).

The UNSW-NB15 dataset has three category features: state, proto, and service (x2, x3, x4, respectively). Using the one-hot-encoding approach, all three characteristics may be turned into vectors. After mapping these three category characteristics, the input feature vector will have 172 columns.

4.3. Proposed Deep Autoencoder (DAE)

DAE is composed of three types of layers (Figure 4.1): input, hidden, and output. The input layer retrieves data from the training set. In the model, the input layer contains 172 features from the UNSW-NB15 dataset. The first hidden layer of the model takes features from the input data. The output of a hidden layer x is fed into a hidden layer x + 1 as an input. The last hidden layer (Latent layer) is the compressed representation of the input. After training the model the latent layer will become an input to the final stage of the model where the classification of the attacks will occur. There are several activation functions to choose from in the hidden layers, including linear, sigmoid, softmax, rectified linear functions and tanh. When the DAE model uses a non-linear activation function, it can extract more useful features. In this study, as a result of preliminary experiments, Tanh and Sigmoid activation functions have been selected to be used in the hidden layers.

4.4. Supervised Classification

After training the DAE using the input form the UNSW-NB15 dataset. As shown in Figure 4.2, the most compressed representation of the input can be obtained from the latent layer. After that, this vector is fed into the last stage of the model where the

supervised classification will take place. The output generated of the stage is the final result of the classification where it will predict for any given package whether it is an attack or normal traffic.

In this stage, several supervised Machine Learning algorithms have been tested to find which algorithm performs the best. the model is trained with each one of the algorithms separately. Later the DAE stage was disconnected and the model was re-trained again. After that the achieved results were checked against the results where the DAE stage was connected to check how much the performance of the model is improved over utilizing DAE in the model.

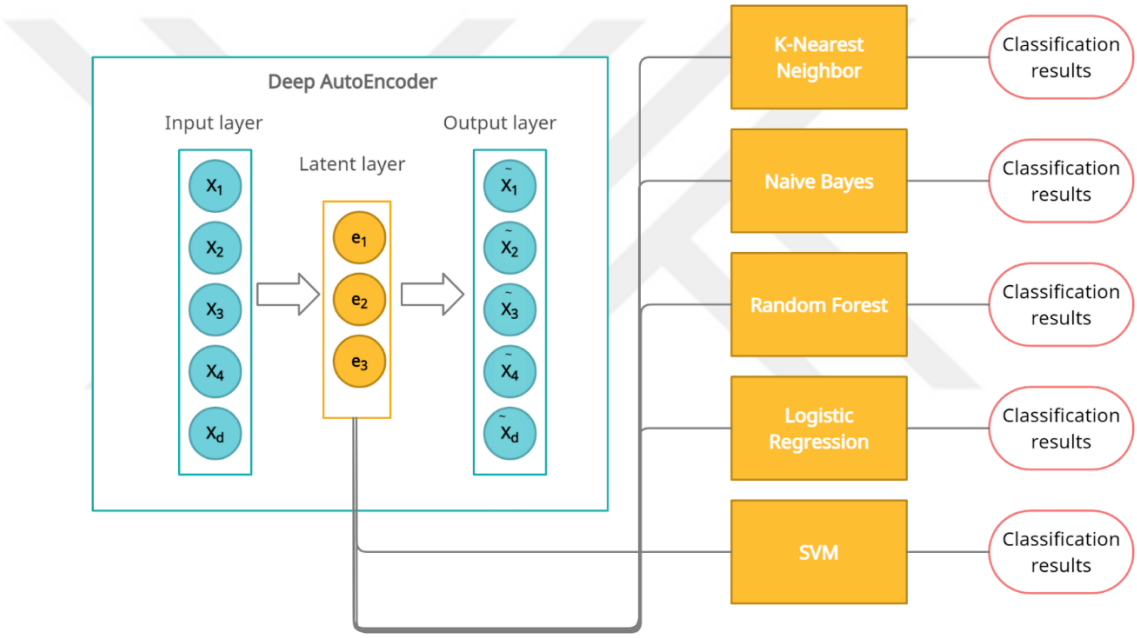

Figure 4.2. *Using the latent layer as an input for the supervised classification algorithms*

Scikit-learn has been used to implement all the different ML algorithms. Below each algorithm with its configurations will be mentioned:

4.4.1. N-Nearest Neighbors

KNeighborsClassifier is used to implement learning based on the k closest neighbors of all query points, where k is an integer number. The value of k is selected as k=12 based on preliminary studies.

4.4.2. Naive Bayes

Gaussian Naive Bayes *GaussianNB* is used to implement the Naive Bayes classification. The probabilities of features are expected to be Gaussian:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

4.4.3. Random Forest

The Random Forest method is implemented using the RandomForestClassifier. As the name suggests, a random forest is an estimator that incorporates a number of different decision tree classifiers into its model to boost its predictive power and control over-fitting. n_estimators=100 was used as the initial value for the classifier, where n_estimators represent the forest's tress count.

4.4.4. Logistic Regression

Logistic regression is a robust supervised machine learning method that is often used to solve binary classification challenges. Logistic regression is a technique that involves modeling binary outcomes using a logistic function. The main difference between logistic and linear regression is that the range of logistic regression is restricted to values between 0 and 1. Additionally, logistic regression method does not need a linear connection between the input and output variables, as linear regression requires. This is because of the nonlinear log transforms used.

4.4.5. Support vector Machine

The SVM algorithm's purpose is to determine the optimal line or decision borders that separates n-dimensional space into groups, allowing us to easily classify newer data points. This optimal decision border is referred to as a hyperplane. SVM selects the hyperplane's extreme vectors or limits which will be used during the creation of the hyperplane. Support vectors term is used to refer to these extreme points, and therefore the name support vector machine is used to refer to the algorithm.

Since the algorithm’s goal is to classify normal and attack network terrific, theoretically the dataset can be classified by a single hyperplane. As a result, the Linear SVM is selected to be used for this model. To implement linear SVM, SVC model is used and initialized with kernel = ‘linear’

Table 4.1. *Algorithm 1: Semi Supervised learning model using AutoEncoders for Intrusion Detection.*

Input: Training dataset UNSW_NB15_Train, Testing dataset UNSW_NB15_Test;

Output: Evaluation metrics of classification: accuracy, precision, recall, and F1-score;

1: **Step1**: Data preprocessing

2: xi= normalization(xi);

3: x2, x3, x4= one-hot (x2, x3, x4);

4: **EndStep**

5: **Step2**: Feature extraction

6: The structure of layers of the Autoencoder is: 172-150-125-100-125-150-172;

7: Train the DAE model using the UNSW_NB15_Train dataset and obtain the most compressed representation of the input

8: **EndStep**

9: **Step3**: Supervised Classification

10: The latent representation of the data generated from the trained DAE modal is used as an input for the supervised classification algorithms;

11: Train the model using N-Nearest Neighbors, Naive Bayes, SVM, Logistic Regression and Random Forest classifiers;

13: Testing dataset UNSW_NB15_Test is used as an input for the trained semi-supervised classifiers to detect attacks;

14: **EndStep**

15: Obtain the results of classification

5. EVALUATIONS AND RESULTS

5.1. Dataset

The UNSW-NB15 dataset has a mixture of real-world modern normal and modern synthetic network traffic malicious activities. To create the features of the UNSWNB15 dataset, both existing and novel approaches are used in conjunction with one another.

The UNSW-NB15 dataset has been used in order to test our experimental approaches. With a clean formatting, the UNSW-NB15 dataset has 42 features, as seen in Table 1. A total of 39 of the 42 features are numerical in nature, while 3 are categorical (non-numeric).

Table 5.1. *UNSW-NB15 dataset feature's list*

No.	Feature	Type
1	dur	Float
2	proto	Categorical
3	servic	Categorical
4	state	Categorical
5	spkts	Integer
6	dpkts	Integer
7	sbytes	Integer
8	dbytes	Integer
9	rate	Float
10	sttl	Integer
11	dttl	Integer
12	sload	Float
13	dload	Float
14	sloss	Integer
15	dloss	Integer
16	sinpkt	Float
17	dinpkt	Float
18	sjit	Float
19	djit	Float
20	swin	Integer
21	stcpb	Integer
22	dtcpb	Integer
23	dwin	Integer
24	tcprtt	Float
25	synack	Float

26	ackdat	Float
27	smean	Integer
28	dmean	Integer
29	trans_depth	Integer
30	response_body_len	Integer
31	ct_srv_src	Integer
32	ct_state_ttl	Integer
33	ct_dst_ltm	Integer
34	ct_src_dport_ltm	Integer
35	ct_dst_sport_ltm	Integer
36	ct_dst_src_ltm	Integer
37	is_ftp_login	Binary
38	ct_ftp_cmd	Integer
39	ct_flw_http_mthd	Integer
40	ct_src_ltm	Integer
41	ct_srv_dst	Integer
42	is_sm_ips_ports	Binary

The UNSW-NB15 consists of two primary datasets: the UNSW-NB15-Train dataset, which is used for training models, and the UNSW-NB15-Test dataset, which is used for testing models after the training is completed. Each dataset contains normal and abnormal traffic. There are 2 labels that classify each record:

- Label : 0 for normal and 1 for attack records.
- Attack_cat: Name of the attack's category. The dataset contains 9 categories (e.g.,Worms, DoS, Backdoors,and Fuzzers)

The description of each category of the attack’s types can be found below:

(1) Fuzzers : a Kind of attack where the attacker attempts to exploit security flaws in Operating Systems, software, or networks in order to suspend or even crash these resources.

(2) Analysis: a sort of attack that infects web applications using malicious web scripting, port scanning, and spam email distribution, among other methods.

(3) Backdoor: a method which the attacker would bypass regular authentication and gain remote access to systems without authorization.

(4) Denial of Service (DoS) attack: an intrusion where the attacker attempts to disrupt computational resources by keeping them overly busy in order to restrict permitted access to the resources.

(5) Exploit: incursions that take advantage of software flaws, errors, or glitches in operating systems or programs.

(6) Generic: these attacks target a cryptographic system and attempt to crack the security system's key.

(7) Reconnaissance: also, may be called probes; Attacks that acquire information about a target's computer network to pass its security controls.

(8) Shellcode: malware attack where the attacker gains control of the compromised computer by injecting a small piece of code in the shell.

(9) Worm: malwares or viruses that replicate themselves and spread to other computers over the network, relying on the security vulnerabilities on the target machine.

The Table below provides details and values distribution of each attack category within the test and train datasets.

Table 5.2. *UNSW-NB15 Train and Test sets record distribution*

Category		Training Set	Testing set
Normal	Normal	56,000	37,000
Attacks	Analysis	2,000	677
	Backdoor	1,746	583
	DOS	12,264	4,089
	Exploits	33,393	11,132
	Fuzzing	18,184	6,062
	Generic	40,000	18,871
	Reconnaissance	10,491	3,496
	Shellcode	1,133	378
	Worms	130	44
Total Records		175,341	82,332

5.2. Evaluation Metrics

Machine learning provides a few measures for evaluating the effectiveness of a classifier. It is critical to select the appropriate performance measures based on the

particular application's certain requirements. Four performance metrics are employed to efficiently evaluate the previously proposed intrusion detection system: precision, F1-score, recall, and accuracy. As shown in the Table below, the performance metrics are obtained using the network attack classification's confusion matrix.

Accuracy: one of the most popular classifier's performance metrics is accuracy. From an IDS perspective, it measures how effectively a classifier can detect the intrusion or attacks. In other words, it provides the ratio of correctly classified intrusion attempts to the total number of inputs.

$$\text{Accuracy} = \frac{correctly\ classified\ attacks}{Number\ of\ inputs} \times 100$$

Precision: While accuracy is a solid indicator of how effectively a model has been trained, it's not really the only metric used to draw conclusions. While a model may provide a high accuracy score while the input dataset is balanced, when the input changes, the model suffers to correctly perceive the data. Precision is the percentage of harmful packages accurately identified. A greater accuracy score indicates that the model is recognizing the attack packages properly. It can be calculated as follows:

$$\text{Precision} = \frac{\text{TruePositive}}{\text{TruePositive+ FalsePositive}}$$

Recall: The recall is concerned with sensitivity and the level to which we are confident that all positive occurrences were predicted positively.

$$\text{Recall} = \frac{\text{TruePositive}}{\text{TruePositive+ FalseNegtive}}$$

F1-Score incorporates precision and recall to determine the model's overall accuracy and performance. A high F1-score indicates that a model recognized attack traffic successfully while reducing false positives and negatives. It can be calculated as follow:

$$\text{F1-score} = \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \times 2$$

Table 5.3. *Confusion Matrix.*

	Predicted Attack	**Predicted Normal**
Actual attack	TruePositive	FalseNegative
Actual normal	FalsePositive	TrueNegative

5.3. Results and Discussion

5.3.1. Classification results

Figure 5.1 and Table 5.4 show the evaluation results of the proposed model on UNSW_NB15_Test dataset, and evaluate the binary-classification results using four evaluation matrices: accuracy, recall, precision and F1-score. From the results we can see that Random Forest performs the best among the rest with 85.70% accuracy, not too far KNN with 84.57% accuracy then Logistic Regression come next with 83.83% accuracy after that comes SVM with 81.59% accuracy and finally Naive Bayes comes at last with 74.59% accuracy.

In terms of Precision, Naive Bayes performs the best among the rest with 83.53%, KNN comes next with 81.17%, then Random Forest comes with 80.08% then Logistic Regression and SVM with 78.18% and 75.07% respectively.

In terms of Recall, SVM performs the best among the rest with 99.66%, Random Forest comes next with 98.53%, then Logistic Regression comes with 97.97% then KNN and Naive Bayes with 93.72% and 68.79% respectively.

Finally in terms of F1 Score, Random Forest performs the best among the rest with 88.36%, Logistic Regression comes next with 86.99%, then KNN comes with 86.96% then SVM and Naive Bayes with 85.63% and 74.76% respectively.

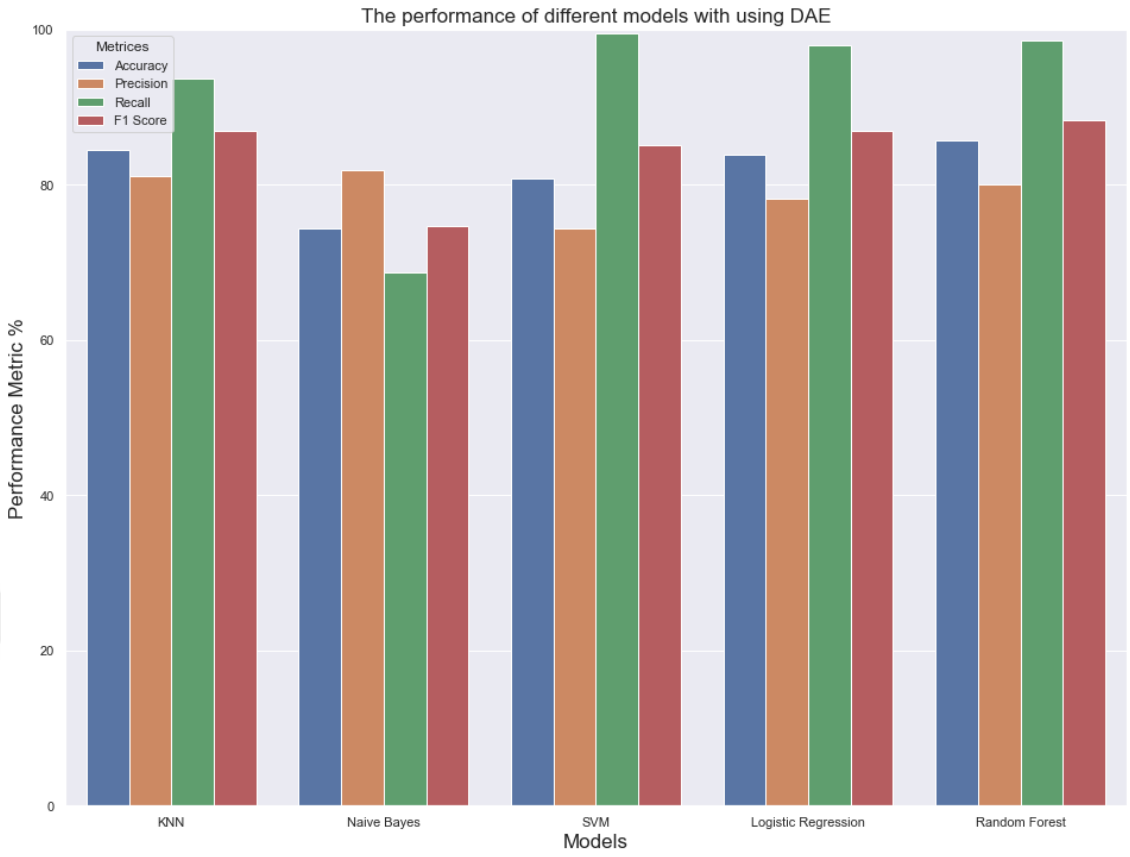

Figure 5.1. *Performance comparison using Autoencoder*

Table 5.4. *Performance of different models using DAE*

Method	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
KNN	84.57	81.17	93.72	86.99
Naive Bayes	74.42	81.86	68.79	74.76
SVM	81.59	75.07	99.66	85.63
LogisticRegression	83.83	78.18	97.97	86.96
Random Forest	85.70	80.08	98.53	88.36

Figure 5.2 and Table 5.5 show the evaluation results of the model on UNSW_NB15_Test dataset without including the Deep autoencoder in the model.

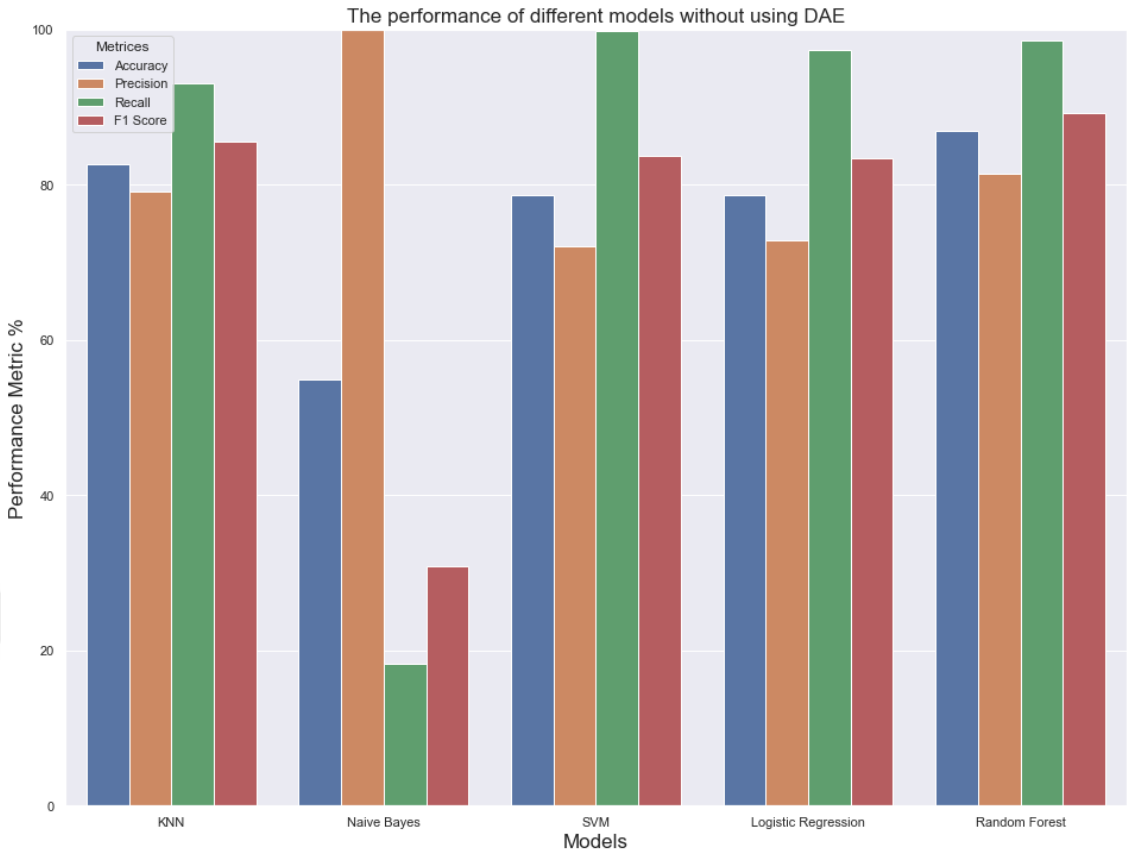

Figure 5.2. *Performance comparison without using Autoencoder*

Table 5.5. *Performance of different models without using DAE*

Method	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
KNN	82.74	79.20	93.10	85.59
Naive Bayes	54.98	100.0	18.24	30.86
SVM	78.68	72.13	99.86	83.76
LogisticRegression	78.66	72.91	97.43	83.41
Random Forest	86.88	81.44	98.66	89.23

Figure 5.3 shows the performance comparison of KNN algorithm when DAE is used and when it is not. By incorporating DAE, we can see the accuracy is increased by 1.83% and precision increased by 1.97% where Recall decreased by 0.62% and F1 Score increased by 1.40%

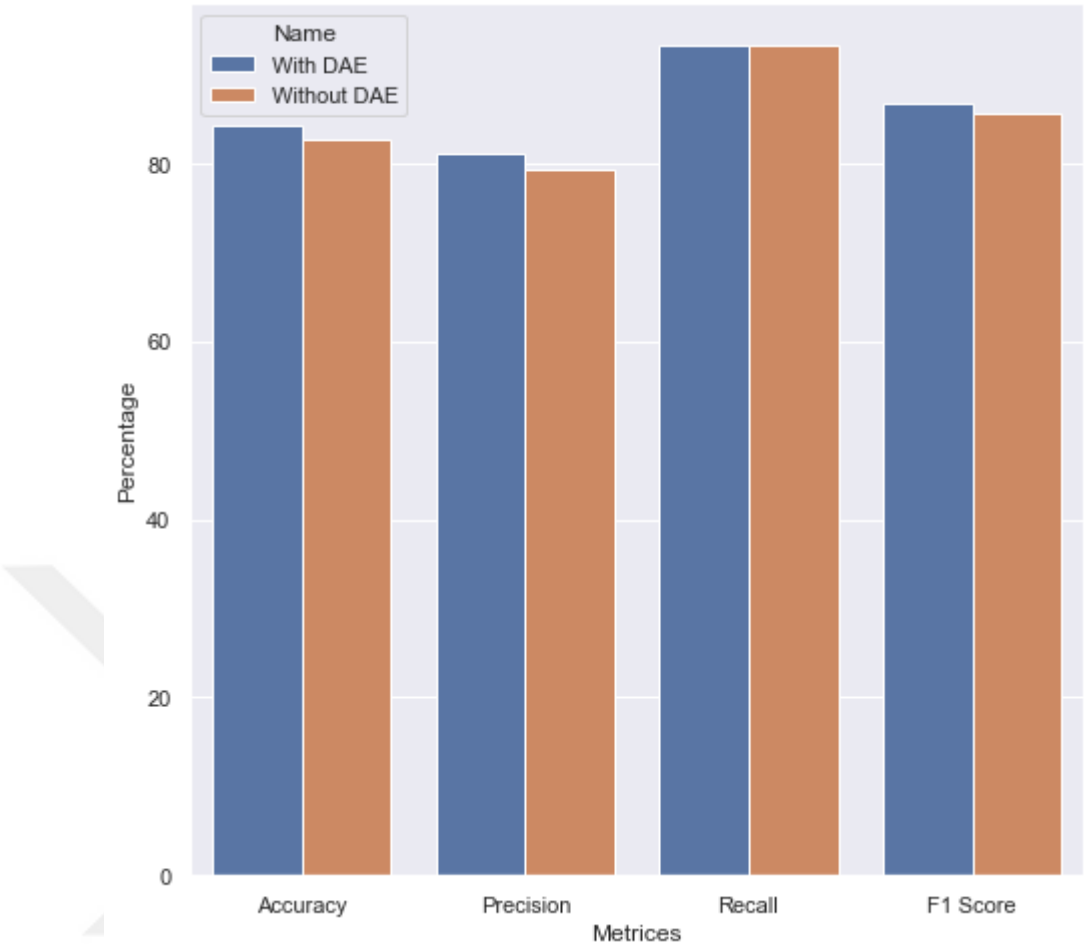

Figure 5.3. *Performance comparison of KNN*

Figure 5.4 shows the performance comparison of Naive Bayes algorithm when DAE is used and when it is not. By using DAE, NB has showed the biggest improvement in the accuracy among other algorithms, the accuracy is increased by 19.44% and precision decreased by 18.44% where Recall increased by 50.55% and F1 Score increased by 43.90%

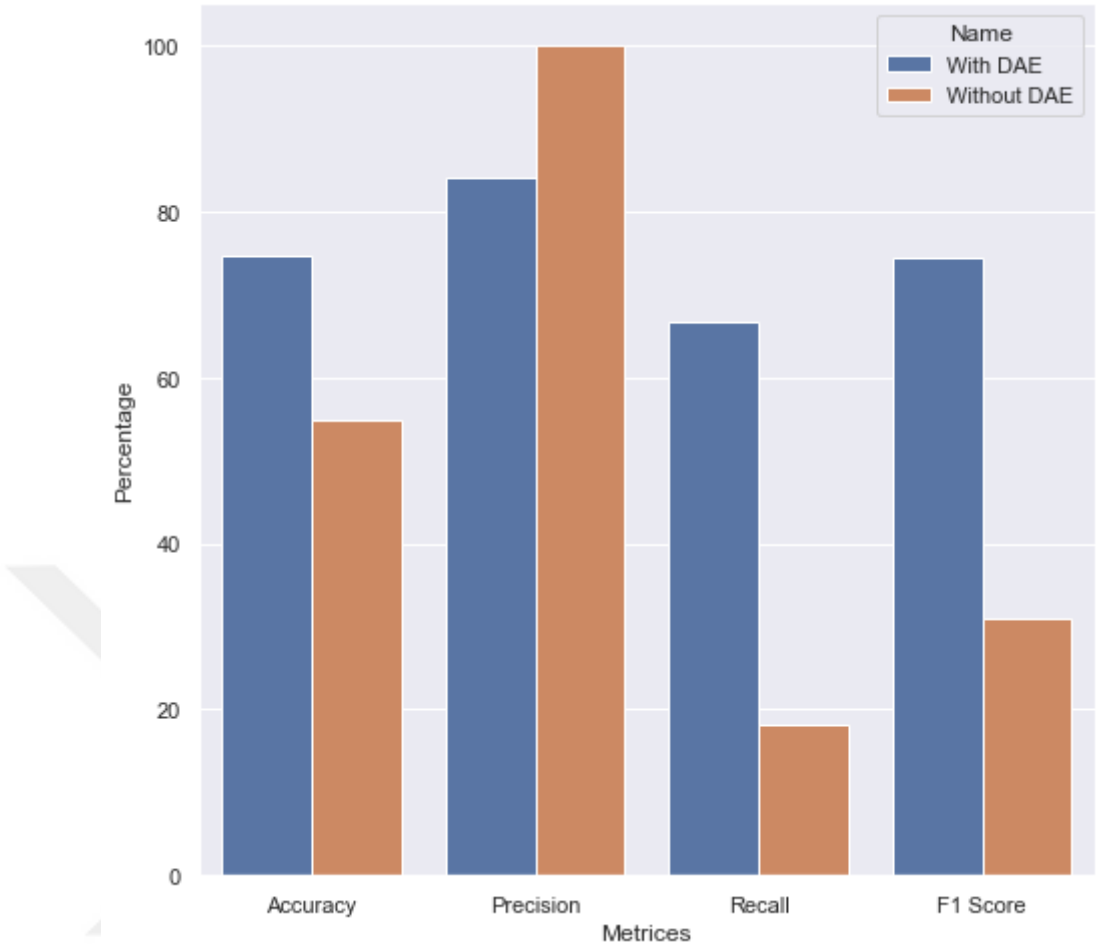

Figure 5.4. *Performance comparison of NB*

Figure 5.5 shows the performance comparison of SVM algorithm when DAE is used and when it is not. By incorporating DAE, we can see the accuracy is increased by 2.91% and precision increased by 2.94 % where Recall decreased by 0.2 % and F1 Score increased by 1.87%

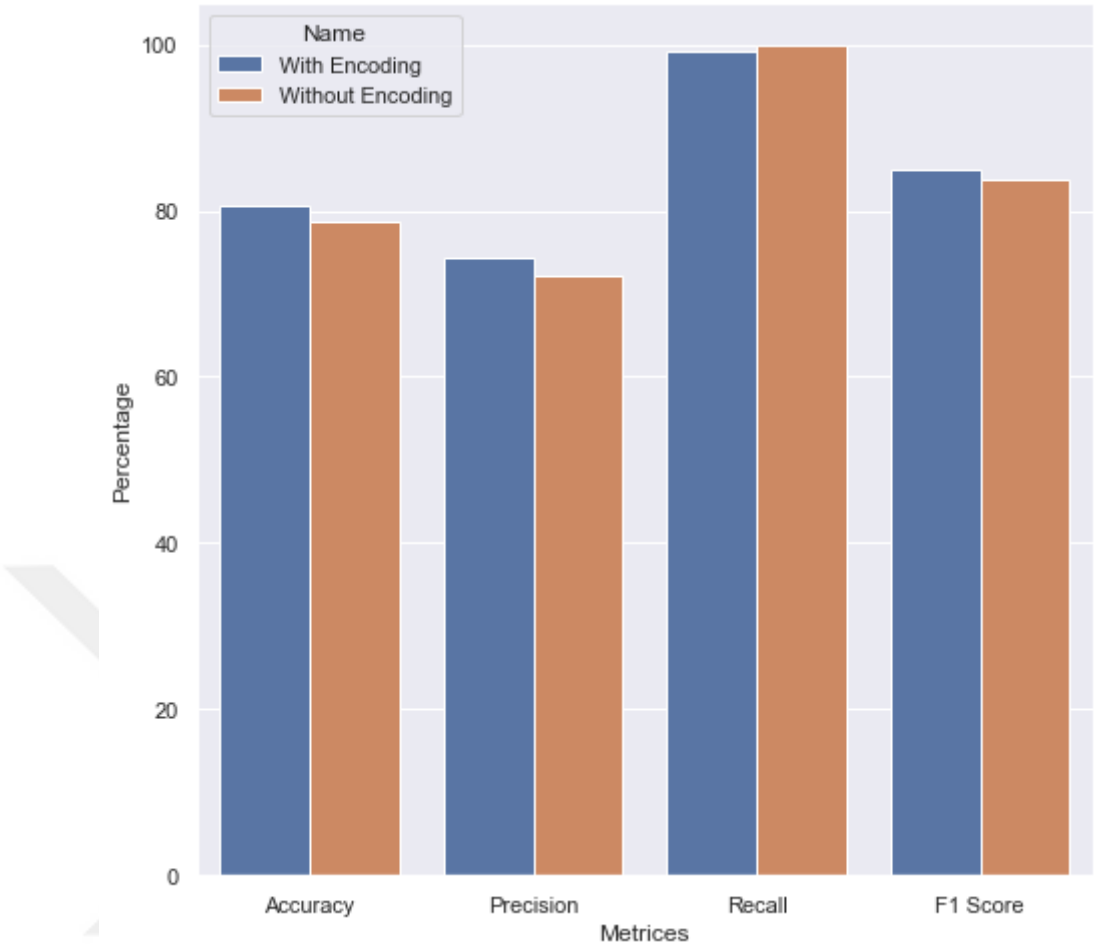

Figure 5.5. *Performance comparison of SVM*

Figure 5.6 shows the performance comparison of the Logistic Regression algorithm when DAE is used and when it is not. We can see the accuracy is increased by 5.17% and precision increased by 5.27% where Recall increased by 0.54% and F1 Score increased by 3.55%

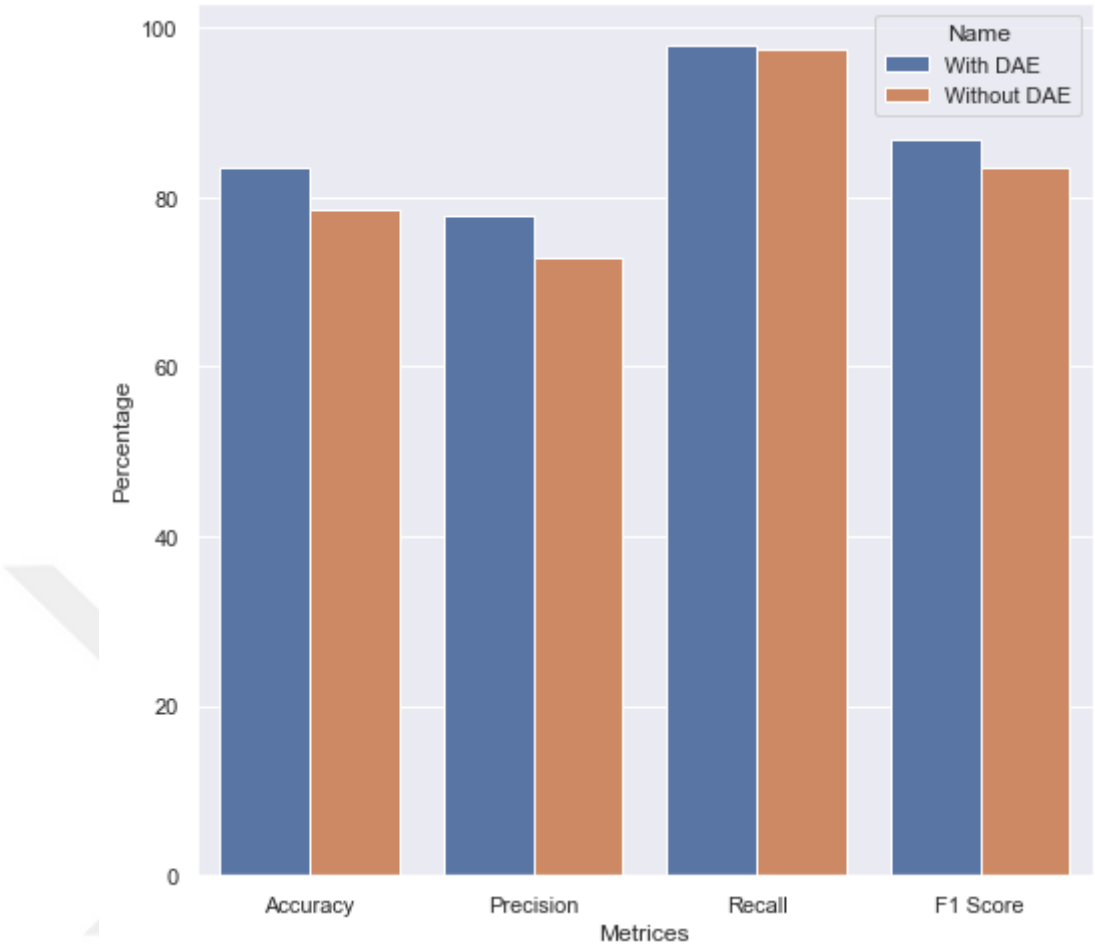

Figure 5.6. *Performance comparison of Logistic Regression*

Figure 5.7 shows the performance comparison of the Random Forest algorithm when DAE is used and when it is not. Unlike the others, RF was the only algorithm that did not show any improvement in the performance, instead of that we can see the accuracy decreased by 1.24% and precision decreased by 1.42% where Recall decreased by 0.15% and F1 Score decreased by 0.91%.

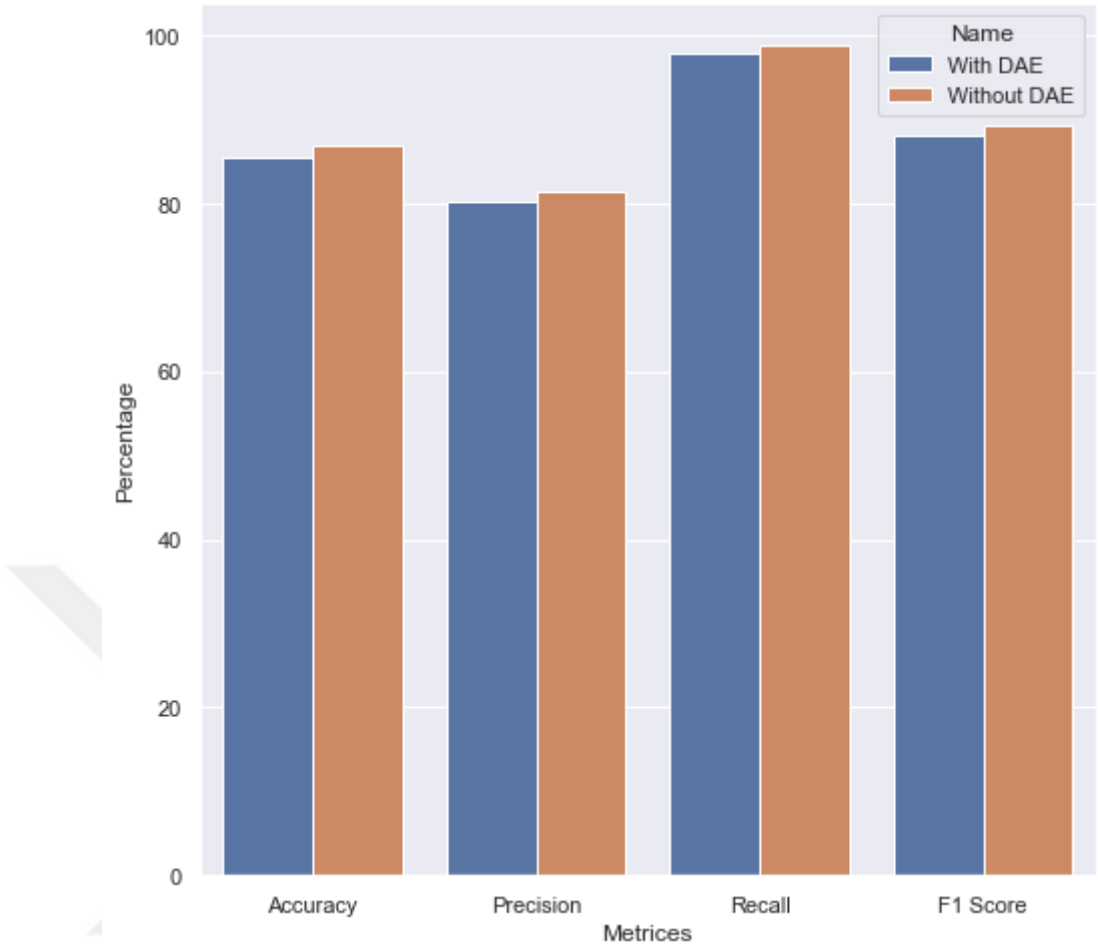

Figure 5.7. *Performance comparison of Random Forest*

5.3.2. Training and testing times

The amount of time and computing power required to create accurate predictions is among the most critical aspects of a successful model. The time taken for the proposed model to train and test on the UNSW-NB15 dataset was measured using the Python time package. Table 5.6 summarizes the training and testing times of each of the algorithms discussed in this study measured in seconds.

Table 5.6. *Training and Testing times*

Method		Training Time (seconds)		Testing Time (seconds)	Total Time (seconds)
		Autoencoder Training	Classification Training		
Using Autoencoder	KNN	367.43	0.03	345.36	812.82
	Naive Bayes	367.43	0.18	0.22	367.83
	SVM	367.43	1519.05	159.45	2045.93

	Logistic Regression	367.43	312.4	0.11	679.94
	Random Forest	367.43	128.62	1.29	497.34
Without using Autoencoder	KNN	-	0.05	311.39	311.44
	Naive Bayes	-	0.45	0.34	0.79
	SVM	-	1673.9	270.01	1943.91
	Logistic Regression	-	20.51	0.11	20.62
	Random Forest	-	35.12	1.6	36.72

From the results in Table 5.6 we can observe that the training time (classification time) in some algorithms such as KNN and Naive Bayes and SVM has been improved using Autoencoder. Meanwhile the training time (classification time) in other algorithms such Logistic Regression and Random Forest has been increased. Because of the Autoencoder training time, we can observe that the total training time has increased, therefore the total time taken for the proposed model to train and test is more efficient when Autoencoder is not used.

6. CONCLUSION

Developing effective Intrusion Detection Systems (IDSs) has acquired a lot of interest as a result of the massive growth in various types of network attacks.

In this paper, an intrusion detection model is proposed, which incorporates unsupervised learning using Deep autoencoder (DAE) and supervised learning classification algorithms, getting the benefits of both. For high-dimensional data, the proposed model can mine potential key features of data. At the same time, the trained latent layer of DAE, which is the dimensionality reduction representation of the input, is used as an input to some of the supervised learning algorithms to classify intrusions. The intrusion detection performance of the proposed model is evaluated on the UNSW-NB15 dataset. The impact of incorporating autoencoders into the model has been studied and the performance of the model is compared with the case when autoencoders are not used. Experimental results show that 4 out 5 studied supervised learning algorithms such as KNN, NB, SVM and Logistic Regression have shown an improvement in the intrusion detection and provide a higher accuracy compared when the autoencoder is not used. The performance of the model presented an improvement between 1.83% and 19.44%. In the light of these findings, we can say that Autoencoder can be used to enhance the performance of models when it is used as a dimensionality reduction method for feature extraction.

In the future, we want to investigate more efficient methods for improving IDS performance. We want to explore more AE types and their latent layer in order to improve the extraction of core features, like variational autoencoders. For certain attacks with limited data samples, an adversarial learning technique is suggested. This method may synthesize similar attacks and enhance the variety of training data, hence improving the detection accuracy.

REFERENCES

L. Columbus. (2016). Roundup Of Internet Of Things Forecasts And Market Estimates.

A look at the basics of Bluetooth technology. (2015, January 20). Bluetooth: http://www.bluetooth.com/Pages/ Basics.aspx adresinden alındı

Adetunmbi, A., Falaki, S., Adewale, O., & Alese, B. (2008). Network intrusion detection based on rough set and knearest neighbor. *Int. J. Comput. Intell. ICT Res. 2 (1)*.

Agrawal, S., & Agrawal, J. (2015). Survey on anomaly detection using data mining techniques. *Procedia Comput. Sci. 60*, 708–713.

al., L. e. (2014). Tagoram: Real-time tracking of mobile RFID tags to high precision using COTS devices. *ACM international conference on Mobile computing and networking, vol. 1,*, (s. 237–248).

al., L. e. (2015). DeepEar: robust smartphone audio sensing in unconstrained acoustic environments using deep learning. *ACM International Con-ference on Pervasive and Ubiquitous Computing, vol. 1*, (s. 283–294).

al., N. M. (2017). Deep android malware detection. *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy* (s. 301-308). ACM.

Ambedkar, D. (2017). Reinforcement Learning Algorithms: Survey and Clas-sification,. *Indian Journal of Science and Technology, vol. 10*, 1–8.

Atzori, L., Iera, A., & Morabito, G. (2010). The internet of things: A survey. Computer Network.

Berman, D. S., Buczak, A. L., Chavis, J. S., & Corbett, C. L. (2019). A survey of deep learning methods for cybersecurity. *Information, vol.10, no.4*, 1-35.

Bertino, E., & Islam, N. (2017). Botnets and internet of things security,. *Computer,no 2*.

Box, G., & Tiao, G. (2011). Bayesian Inference in Statistical Analysis. *John Wiley &Sons*.

Buczak, A., & Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Commun. Surveys Tuts.*, 1153–1176.

Chang, Y., Li, W., & Yang, Z. (2017). Network intrusion detection based on random forest and support vector machines. *Computational Science and Engineering (CSE) and Embedded and Ubiquitous Computing (EUC)* (s. 635–638). IEEE International Conference on, vol. 1. IEEE.

Chen, F., Deng, P., Wan, J., Zhang, D., Vasilakos, A., & Rong, X. (2015). Data mining for the internet of things: literature review and challenges. *Int. J. Distributed Sens. Netw. 11, 431047*.

Chen, Y., Y. Zhang, & Maharjan, S. (2017). Deep Learning for Secure Mobile Edge Computing. *arXiv preprint arXiv:1709.08025*.

Cisco Systems. (2017). Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016-2021. *White Paper.*

Columbus, L. (2016). Roundup Of Internet Of Things Forecasts And Market Estimates. Forbes.

Cutler, D., & al., e. (2007). Random forests for classification in ecology. *Ecology 88 (11)*, 2783–2792.

Deng, Z., Zhu, X., Cheng, D., Zong, M., & Zhang, S. (2016). Efficient kNN classification algorithm for big data. *Neurocomputing 195*, 143–148.

Dobkin, D. M., & Aboussouan, B. (2009). Low power Wi-Fi (IEEE 802.11) for IP smart objects. *GainSpan Corporation.*

Doshi, R., Apthorpe, N., & Feamster, N. (2018). Machine Learning DDoS Detection for Consumer Internet of Things Devices. *arXiv:1804.04159*.

Eun, H., Lee, H., & Oh, H. (2013). Conditional privacy preserving security protocol for nfc applications. IEEE Transactions on Consumer Electronics.

Farahnakian, F., & Heikkonen, J. (2018). A deep auto-encoder based approach for intrusion detection system. *Int. Conf. on Advanced Communication Technology (ICACT)*, (s. 178–183).

Farnaaz, N., & Jabbar, M. (2016). Random forest modeling for network intrusion detection system. *Procedia Comput.Sci.*, 213–217.

Fiore, U., Palmieri, F., Castiglione, A., & Santis, A. D. (2013). Network anomaly detection with the restricted Boltzmann machine. *Neurocomputing, vol. 122*, 13-23.

Gondhi, N., & Gupta, A. (2017). Survey on Machine Learning Based Scheduling in Cloud Computing. 57–61.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep learning. *MIT press Cambridge*.

H.Zhang, Wu, Q., S.Gao, Z.Wang, Y.Xu, & Y.Liu. (2018). An Effective Deep Learning Based Scheme for Network Intrusion Detection. *International Conference on Pattern Recognition(ICPR).* Bejing,China.

Ham, H.-S., Kim, H.-H., Kim, M.-S., & Choi, M.-J. (2014). Linear SVMbased android malware detection for reliable IoT services. *J. Appl. Math*.

Han, G., Xiao, L., & Poor, H. (2017). Two dimensional anti jamming communication based on deep reinforcement learning. *Proc. IEEE Int. Conf. Acoustics Speech and Signal Processing*, (s. 2087–2091). New Orleans, LA.

Help Net Security. (2017). The cost of IoT hacks: up to 13% of revenue for smaller firms.

Hermans, M., & Schrauwen, B. (2013). Training and analyzing deep recurrent neural networks. *Advances in neural information processing systems*, (s. 190-198).

Hinton, G. E. (2012). A practical guide to training restricted Boltzmann machines. *Neural networks: Tricks of the trade* (s. 599-619). Springer.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation, vol. 18, no. 7*, 1527-1554.

History of the Bluetooth special interest group. (2015, January 19). Bluetooth: http://www.bluetooth. com/Pages/History-of-Bluetooth. aspx adresinden alındı

Huang, S. (2003). Dimensionality reduction in automatic knowledge acquisition: A simple greedy search approach.

Hush, D., & Horne, B. (1993). Progress in supervised neural networks. *IEEE Signal Process. Mag. 10 (1)*, 8–39.

Ingre, B., & Yadav, A. (2015). Performance analysis of NSL-KDD dataset using ANN. *International Conference on Signal Processing and Communication Engineering Systems*, (s. 92–96). Guntur, India.

Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016). A Deep Learning Approach for Network Intrusion Detection System. *EAI Endorsed Trans. Secur. Saf.*, 21–26.

Jordan, M., & Mitchell, T. (2015). Machine learning: trends, perspectives, and prospects. *Science 349 (6245)*, 255–260.

Karimipour, H., & Dinavahi, V. (2017). On false data injection attack against dynamic state estimation on smart power grids. *IEEE Int. Conf. on Smart Energy Grid Engineering*, (s. 1–7).

Khan, J., & Jain, N. (2016). A survey on intrusion detection systems and classification techniques. *Int. J. Sci. Res. Sci.Eng. Technol.*, 202–208.

Kim, G., Lee, S., & Kim, S. (2014). A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. *Expert Syst. Appl. 41 (4)*, 1690–1700.

Kim, W., Jeong, O.-R., C. Kim, & So, J. (2011). The dark side of the internet: Attacks, costs and responses. *Information systems, vol. 36, no. 3*.

Koc, L., Mazzuchi, T., & Sarkani, S. (2012). A network intrusion detection system based on a hidden Naïve Bayes multiclass classifier. *Expert Syst. Appl.*, 13492–13500.

Kotsiantis, S. (2013). Decision trees: a recent overview. *Artif. Intell. Rev. 39 (4)*, 261–283.

Kotsiantis, S., & Kanellopoulos, D. (2006). Association rules mining: a recent overview. *GESTS Int. Trans. Comput. Sci. Eng. 32 (1)*, 71–82.

Kotsiantis, S., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: a review of classification techniques. *Emerg. Artif. Intell. Appl. Comput. Eng. 160*, 3–24.

Kumar, G., Kumar, K., & Sachdeva, M. (2010). The use of artificial intelligence based techniques for intrusion detection:A review. *Artif. Intell*, (s. 34, 369–387).

L, M.-G., & EC, L. (2018). The secret of machine learning. *ITNOW*.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, (s. 521, 436–444).

Li, H., Ota, K., & Dong, M. (2018). Learning IoT in Edge: Deep Learning for the Internet of Things with Edge Computing. *IEEE Network, vol. 32, no. 1*, 96-101.

Li, W., Yi, P., Wu, Y., Pan, L., & Li, J. (2014). A new intrusion detection system based on KNN classification algorithm in wireless sensor network. *J. Elect. Comput. Eng.*

Liao, H., Richard Lin, C., Lin, Y., & Tung, K. (2013). Review intrusion detection system: A comprehensive review.

Lin, W., Lin, H., Wang, P., Wu, B., & Tsai, J. (2018). Using convolutional neural networks to network intrusion detection for cyber threats. *IEEE Int.Conf. on Applied Syst. Invention (ICASI)*, (s. 1107–1110).

Liu, J., Xiao, Y., & Chen, C. (2012). Authentication and access control in the internet of things.

Liu, Y., & Pi, D. (2017). A novel kernel SVM algorithm with game theory for network intrusion detection. *KSII Trans. Internet Inf. Syst. 11 (8)*.

Maghrebi, H., Portigliatti, T., & Prouff, E. (2016). Breaking cryptographic implementations using deep learning techniques. *International Conference on Security, Privacy, and Applied Cryptography Engineering* (s. 3-26). Springer.

Mclay, F. (2018). Privacy law: How to save face: data and privacy safeguards. *Gov Dir*.

Meidan, Y., & al., e. (2017). Detection of Unauthorized IoT Devices Using Machine Learning Techniques. *arXiv:1709.04647*.

Mocnej, J. P. (2018). Network traffic characteristics of the IoT application use cases. *School of Engineering and Computer Science.* Victoria University of Wellington.

MT, G., C, B., & D., M. (2017). Using SEIRS Epidemic Models for IoT Botnets Attacks. *Proceedings of the International Conference of Design of Reliable Communication Networks (DRCN '17).* Munich, Germany.

Panda, M., & Patra, M. (2007). Network intrusion detection using naive bayes. *Int. J. Comput. Sci. Netw. Secur. 7 (12)*, 258–263.

Pascanu, R., Gulcehre, C., Cho, K., & Bengio, Y. (2013). How to construct deep recurrent neural networks. *arXiv preprint arXiv:1312.6026*.

Quinlan, J. (1986). Induction of decision trees. *Mach. Learn. 1 (1)*, 81–106.

Reddy, R., Ramadevi, Y., & Sunitha, K. (2016). Effective discriminant function for intrusion detection using SVM. *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, (s. 1148–1153). Jaipur, India.

Sherasiya, T., Upadhyay, H., & Patel, H. (2016). A survey: Intrusion detection system for internet of things.

Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). A Deep Learning Approach to Network Intrusion Detection. *IEEE Transactions on Emerging topics in Computational Intelligence, vol. 2, no. 1*.

Tajbakhsh, A., Rahmati, M., & Mirzaei, A. (2009). Intrusion detection using fuzzy association rules. *Appl. Soft Comput. 9 (2)*, 462–469.

Tan, Z., Jamdagni, A., He, X., Nanda, P., & Liu, R. (2013). A system for Denial Of Service attack detection based on multivariate correlation analysis. *IEEE Trans. Parallel Distr. Syst. 25 (2)*, 447–456.

Tauchmann, D., & Sikora, A. (2015). Experiences and measurements with bluetooth low energy (ble) enabled and smartphone controlled embedded applications. *International Journal of Electronics and Electrical Engineering.*

Torres, P., Catania, C., Garcia, S., & Garino, C. G. (2016). An analysis of Recurrent Neural Networks for Botnet detection behavior. *Biennial Congress of Argentina* (s. 1-6). ARGENCON: IEEE.

Wang, Y., H. Yao, & Zhao, S. (2016). Auto-encoder based dimensionality reduction. *Neurocomputing, vol. 184*, 232–242.

Wirth, C., Akrour, R., Neumann, G., & Frnkranz, J. (2017). A Survey of Preference-Based Reinforcement Learning Methods. *Journal of Machine Learning Research, vol. 18*, 1–46.

Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. *Chemometr. Intell. Lab. Syst. 2 (13)*, 37–52.

Xiao, L., Li, Y., Han, G., Liu, G., & Zhuang, W. (2016). PHYlayer spoofing detection with reinforcement learning in wireless networks. *IEEE Trans. Veh. Technol. 65 (12)*, 10037–10047.

Xiao, L., Li, Y., Huang, X., & Du, X. (2017). Cloud based malware detection game for mobile devices with offloading. *IEEE Trans. Mobile Comput. 16 (10)*, 2742–2750.

Xiao, L., Wan, X., Lu, X., Zhang, Y., & Wu, D. (2018). IoT Security Techniques Based on Machine Learning: How Do IoT Devices Use AI to Enhance Security? *IEEE Signal Processing Magazine*.

Xie, M., Huang, M., Bai, Y., & Hu, Z. (2017). The anonymization protection algorithm based on fuzzy clustering for the ego of data in the Internet of Things. *J. Electr. Comput. Eng. 2017*.

Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. *IEEE Access*, 21954–21961.

Zarpelão, B., Miani, R., & de Alvarenga, S. (2017). A survey of intrusion detection in the internet of things.

Zeadally, S., & Tsikerdekis, M. (2020). Securing Internet of Things (IoT) with machine learning. *International Journal of Communication Systems*.

Zhang, C., & Green, R. (2015). Communication security in internet of thing: preventive measure and avoid ddos attack over iot network,. *Proceedings of the 18th Symposium on Communications & Networking.* Society for Computer Simulation International.

Zhao, K., & Ge, L. (2013). A survey on the Internet of Things security. 2013 Ninth International Conference on Computational Intelligence and Security. *Leshan*, (s. 663-667). China.

Zhao, S., Li, W., Zia, T., & Zomaya, A. (2017). A dimension reduction model and classifier for anomaly based intrusion detection in the internet of things.

Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence & Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (s. 836–843). IEEE 15th Intl.