

ZERO-DAY ATTACK DETECTION WITH DEEP LEARNING IN NETWORKS

MSc Research Project
Cyber Security

Baran Diloglu
Student ID: 20221142

School of Computing
National College of Ireland

Supervisor: Imran Khan


National College of Ireland
Project Submission Sheet
School of Computing



Student Name:	Baran Diloglu
Student ID:	20221142
Programme:	Cyber Security
Year:	2021
Module:	MSc Research Project
Supervisor:	Imran Khan
Submission Due Date:	15/08/2022
Project Title:	ZERO-DAY ATTACK DETECTION WITH DEEP LEARNING IN NETWORKS
Word Count:	7343
Page Count:	24

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	14th August 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission , to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project , both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

ZERO-DAY ATTACK DETECTION WITH DEEP LEARNING IN NETWORKS

Baran Diloglu
20221142

Abstract

Efficiency and accuracy plays a very important role in order to detect and prevent cyber-attacks before any damage done to system or user. Artificial intelligence is growing exponentially rather than type of cyber-attacks. This might allow us to improve network security using help of AI with investigating previous cyber-attack behaviors. All it needs to be done is experimenting with different deep learning models depending on the previous network records and testing these models by cross checking with different network datasets. Choosing deep learning for evaluation purposes can help us to achieve more efficient system rather than machine learning based intrusion detection systems. Also, used deep learning model's learning process can come with higher accuracy than previous projects in this area. Biggest advantage of using deep learning models in detection systems, AI model can feed itself to grow during the time with new features and specifications. This will eventually help intrusion detection systems to be ready for new type of attacks or same attacks with different features during the time. Proposed algorithms and analysis will show how accurate cyber-attacks can be detected simultaneously. While creation of neural networks to detect attacks, this research will be helpful to identify what kind of datasets need to be used and how datasets should be seperated individually. Answers will be gathered by creating neural networks with almost same algorithms using mixed and singular attack type based datasets.

1 Introduction

The growth of the Internet of Things concept over the last ten years has resulted in a huge amount of zero-day attacks on IoT devices and systems, many of which are capable of causing significant amount of damage. Machine learning and deep learning has become very popular and handfull techniques to make the tasks easy. This includes intrusion detection, which has a strong probability of detecting both zero-day and well-known signature attacks. As a result, the demand for powerful intrusion detection systems capable of detecting zero-day threats are increasing. Also, these systems need to be ready for the unknown attacks and attacks that have unknown signatures. Tools such as intrusion detection systems, firewalls, scanners, and antivirus software, among others, are used to prevent various types of attacks. Various attacks are frequently launched against network-connected devices. The internet allows networks to communicate with one another while also allowing hardware, intelligence, software, information, and data to be transmitted. As a result, computer networks are extremely vulnerable to viruses and other cyber-attacks. To make the networks more secure AI systems can be implemented

to the intrusion detection systems using previous network data flows and various cyber-attack effects on the network. Main problem is the efficiency of this implementation because of the necessity of parallel work with network data flow and intrusion detection system.

Research Question. How efficient and accurate is detecting zero-day attacks with using deep learning in networks?

Due different researches about intrusion detection systems that has been conducted in past decade as can be seen in Section 2 literature review about related works, only one question comes to minds. Is it really possible to actually detect zero day attacks in the network systems automatically with great deal of accuracy and efficiency? And can these systems work under the new circumstances like unknown intrusions and zero day attacks? In this research question main aim will be conducting a detailed research about creating a intrusion detection system with a high call capability.

Structure of the Report. The rest of the paper is structured as follows: Section 2 gives an overview of related works and researches about this topic in subtitles. Also, it gives an overview about previous works that have been conducted using datasets I chose to use in this research. These subtitles starts with underlining technical overview of intrusion detection systems and previous research about these systems. Section continues with main topic of this research which is zero-day attacks. It explains zero-day attack behaviours and how to catch those attacks using different techniques. Differences between machine learning and deep learning process generally explain as following zero-day attacks and this title gives an general overview about advantages of deep learning over machine learning idea. And lastly, related works with datasets are divided with previous works about each dataset and their results using different deep learning models and those models diagrams. Section 3 includes research process methodologies phase by phase and dataset informations show the main usage and differences between them. This section also includes research resources, analysis and design differences of the main research than previous ones. Following Section 4 gives a general overview about architecture of deep learning models and main software requirements for this projects. At the end of this section functionality is explained to understand accuracy and concept of compile process of proposed model. In Section 5, explains implementation process and decision of choosing datasets for this research. While doing the implementation of proposed model to these datasets, this section also includes differences between results of different datasets with same and different models in this research. In order to understand these process, Section 6 explains results of those steps and main purpose of decision those models depending on their results of accuracy if they satisfy our general expectation and if they can achieve a contribution to previous studies. At the end of this research paper, Section 7 finalize general thoughts about research and if research question has been answered successfully. Key findings and general discussion about efficiency due to it's limitations. Future work about this research and if there are possible commercialisation can be find in this section. If not what changes can be done about project or approach about this topic to make possible or more efficient regarding to answer the possible commercialisation.

2 Related Work

The technical and managerial measures taken to ensure the confidentiality, integrity, availability, controllability, and non-repudiation of electronic data are referred to as in-

formation security. In today's information age, computer networks are critical. Due to its openness, scalability, and variety of terminal distribution, computer networks are vulnerable to computer viruses, hackers, and malware in addition to technological defects and human irresponsibility Zeng et al. (2020). Information security must be emphasized in the face of multiple network security risks. There are two forms of network security: network security and information security. Network security include both system security (physical components, operating systems, and applications) and service security (continuity and efficiency). Data integrity involves data encryption, backup, programs, and so on. The longevity and frequency of zero-day attacks, which exploit vulnerabilities that have not been publicly published, are unknown. Cyber criminals who are aware of new vulnerabilities have full reign to attack any target they want while going undiscovered. Unfortunately, analyzing these major risks is challenging since data is generally not available until after an attack has been found. Furthermore, zero-day attacks are extremely unusual instances that are unlikely to be detected in honeypots or laboratory studies Bilge and Dumitrag (2012). Prior research Schneier (2000), on the other hand, has concentrated on the entire window of vulnerability exposure, which lasts until all vulnerable hosts have been fixed and includes attacks launched after the vulnerability has been published. For instance, a review of three exploit archives revealed that 15% of these attacks were written before the vulnerability was publicly disclosed Frei (2009). This study does not provide information about security vulnerabilities that can be used for exploitation in the networks or systems eventually. To catch these attacks in action there are some system architectures based using Intrusion detection systems in firewalls. Before understanding these detection systems, true definition of intrusion need to be known. Intrusion can be defined as a threat that threatens to users' confidentiality, integrity with using personal information without any aurtherization, copying or damaging system files which are storing these informations. Intrusion Detection Technology is a system that helps maintain computer security by detecting dangerous network activities Khraisat et al. (2019). Online shopping systems, businesses, banking infrastructures, cryptocurrency trading systems and even blockchain operations rely on computer network systems. Therefore, as the reliance on computer networks grows, so do the risks to networks. Because of the growing number of threats, network security has become a top priority. According to this report Society (2019) these type of cyber attacks made companies and governments to lose \$45 billion all over the world. To prevent these losses caused by cyber-attacks, several companies like Cisco and Fortinet, developed different hardwares run with unique softwares for enterprises, small businesses and even home networks. But these hardwares which are made as firewalls for networks are not enough for unknown or undisclosed type of attacks, this situation became a reason for creation of intrusion detection systems (IDS). IDSs can be implemented to the network as a hardware or software between local network and firewalls to monitor the entire network flow for attacks that can pass through the firewall. As seen in figure 1 diagram of IDS demonstration, it monitors traffic between the network and analysis every request in the internet flow by running forensics analysis with known signature database and if it identifies any malicious activity, it raises an alarm with alert administration Çakır (2019).

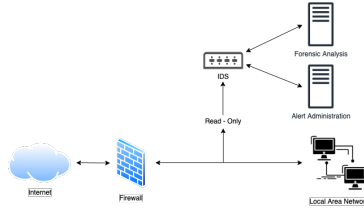


Figure 1: Intrusion Detection System

2.1 Intrusion Detection Systems

Security infrastructures are now relying on intrusion detection systems to identify threats before they have the chance to inflict widespread damage. These systems include hardware, software, policies, or some mix of these are the components of an IDS that are tasked with detecting potential breaches in a network’s collected data. IPS seeks to avoid intrusions in the network, whereas IDS works on the concept that no matter how secure a network may be, intrusions are bound to occur, and it tries to find out whether or not there were any in the first place Hamid et al. (2016). Intrusion detection systems can be characterized as either host-based or network-based Venter and Eloff (2003). In order to secure the local area network, a network-based IDS is put in place as a second line of protection after the firewall. It is designed to detect intrusions from many hosts. Whereas, the host-based solution requires installation on each machine, making it effective in detecting U2R and R2L attacks, but at the expense of significant operation and maintenance costs Chen et al. (2010).

Signature-based, anomaly-based, and specification-based approaches of intrusion detection are commonly separated in the literature Liao et al. (2013). Hybrid methods, which integrate two or more methodologies, are increasingly becoming more common Comar et al. (2013).

2.1.1 Signature-Based Intrusion Detection Systems

Signature-based IDS works as the name suggests, works with saved and publicly announced attack signatures which are already stored in internal database of hardware detection system Ioulianou, Vasilakis, Moscholios and Logothetis (2018). Today’s IDS solutions work with combination of hardware, software and cloud based systems. Reason of cloud usage is to create and use a worldwide signature based database for every detection system can use.

2.1.2 Anomaly-Based Intrusion Detection Systems

Anomaly-based IDS works with definition of network’s behavior. Acceptable network behavior need to learned by specification of network’s usage and it’s users. This definition is a base model of true network behavior for detection system. If the network behavior conforms to the base model behavior, it will be accepted or trigger the event on anomaly detection Ioulianou, Vassilakis and Moscholios (2018).

2.1.3 Specification-Based Intrusion Detection Systems

Specification-based intrusion detection has been offered as a viable alternative that combines the strengths of misuse detection and anomaly detection. This combination increases the power and success rate of catching unusual behavior with high accuracy detection of known attacks in database, and possibility to identify unknown threats Upuluri and Sekar (2001).

2.2 Zero-Day Attacks

Zero-day attacks are an attack that takes use of a vulnerability which has not been officially reported. When it comes to not published and fixed attacks, there is no defence against them. Due to unknown vulnerability, the compromised software cannot be fixed and because of this obscurity any anti-virus products using signature based scanning are unable to identify the threat. According to Chris Chapman Chapman (2016), "zero-day attacks are network traffic pattern of interest that does not have any matching patterns in malware or attack detection elements" Chapman (2016). And all new attacks can be categorized as zero-day attack characteristic at the beginning. There are numerous ways in which a zero-day vulnerability might be discovered; for example, it could be a lack of data encryption, a lack of authority to access the information or a flaw in the password management system. Relevant information regarding zero-day exploits is only made public after the exploit is discovered because of the nature of these vulnerabilities. Because zero-day attacks might produce unusual traffic or suspect scanning activity to originate from a client or service, we can apply and increase network anomaly detection technologies Bhuyan et al. (2014) such as deep learning.

2.3 Machine Learning and Deep Learning

Machine learning is a subset of artificial intelligence that is strongly related to computer science, which aims at making predictions with computers as well. It is closely associated with mathematical optimization, which provides the study with techniques, theories, and application areas Xin et al. (2018). Machine learning and data mining are sometimes combined Louridas and Ebert (2016), however the latter subject, described as unsupervised learning, tends to focus mostly on prior data analysis. Unsupervised machine learning technique can be used to train and obtain baseline behavioral profiles for a variety of entities, which can then be used to detect relevant abnormalities Jordan and Mitchell (2015). In 1959, Arthur Samuel made a definition about machine learning as "field of study that gives computers the ability to learn without being explicitly programmed". Classification and regression is a mainly focal point of machine learning that is based on earlier obtained attributes from the training data. The area of deep learning is a relatively new one in research of machine learning concept. Motivation of this new area is creation of a neural network that may be used to learn analytically by simulating the human brain. This means, it uses a similar system to imitate the human brain to analyse the collected data LeCun et al. (2015). In 2009, unsupervised layer by layer working deep belief network model of deep learning presented by Geoffrey E. Hinton Hinton (2009). Optimization was a big problem in deep architecture before this model. It offers the possibility for overcoming the optimization problem. After this proposal for optimization in 1986 solution for neural behavior in human brain was proposed by Hinton and his group Rumelhart et al. (1986) with combining Hebbian learning rules Hebb (1949) and

biological neural behavior with simplified neural model explained in 1982 by Erkki Oja (1982). Autoencoders are a key metaphor for unsupervised learning and for addressing the question of how neural synaptic alterations caused by local biological functions in human brain may be coordinated by itself to achieve global learning and intelligent behavior Baldi (2011). Furthermore, convolutional neural network was proposed Lecun et al. (1998) as a first genuine multi layer structure as a deep learning technique that leverages huge amount of space relative connection to minimize the number of data to learn and increase the training performance with this improvement.

Deep learning is a machine learning approach based on characterizing data. Deep learning is a machine learning technique based on the analysis of data. Images, for example, can be described as vectors of each pixel's intensity value, or more abstractly in terms of a set of edges and a specific form. Learning tasks from examples is made simpler when specific representations are used. Deep learning approaches, like machine learning, offer supervised and unsupervised learning modes. Underlying frameworks have a significant impact on learning models. The use of unsupervised or semi-supervised feature learning and hierarchical feature extraction to change features manually is one of the advantages of deep learning Deng and Yu (2014). With increasing data sets, deep learning outperforms more standard machine learning methods. Data volumes are too limited for deep learning models to function properly, because deep learning algorithms need a lot of information to fully comprehend the data. Basic machine-learning algorithms will perform more effectively in this instance since they follow established rules LeCun et al. (2015). On the other hand, hardware specifications are very specific for deep learning models because of data volumes and depending on the huge amount data matrix operations will get bigger and complicated. As a result, a graphics card is required for the DL to function correctly. With the help of high-performance computers and GPUs, deep learning depends more heavily than classic machine learning methods.

2.4 Related Works with Datasets

Monitoring networks and systems for attack activity and policy breaches is the function of intrusion detection systems (IDS). Traffic anomalies should be easily detected by this type of technology, which should be able to identify them. Machine learning approaches to developing an anomaly-based IDS using several datasets, each with their own categorized characteristics and features based on published year and known attacks and their signatures, were discussed in different researches. IDS structures and results using complicated machine learning approaches, such as deep neural networks, gradient boost classifiers, or hidden Markov models (See Fig. 2), have appeared in a great number of studies since the dataset was provided. The inherent difficulties of training such complicated models are avoided by employing a basic closest neighbor classification strategy, as demonstrated in this research Andreucut (2022). Complex approaches like deep learning neural networks are used to create these solutions, whereas baseline solutions corresponding to simple but effective methods are not even considered. There are, however, some downsides to the methods mentioned, which are worth considering. Anomaly detection cannot be utilized for real-time data flow since intrusion detection systems, which are based on signature, are vulnerable to variation of attacks. This work Seo and Pak (2021) proposes a two-level classifier that simultaneously achieves excellent performance and real-time classification in order to address these difficulties. It uses classifiers at levels 1 and 2 internally. For incoming data flow, the level 1 classifier conducts real-time detec-

tion with middling accuracy at the beginning (See Fig 3). The classification regarding to the referred figure might be delayed because of data flow in the network. When data flow slows down or stops for some reason data classification can be achieved for high probability. In order to accomplish precise categorization, the level 2 classifier extracts statistical data from the traffic flow. The use of a two-level classification method, rather than existing methodologies, can yield greater performance in terms of both accuracy of detection and process time of detection the attacks. Several initiatives have attempted to apply the shallow and deep learning machine learning models and associated algorithms and structures. Additional information has been given in this paper Janiesch et al. (2021) about automated analytical model creation, including its four main components: data input, feature extraction, model construction and model evaluation. Last but not least, this study Janiesch et al. (2021) discusses four key issues for intelligent systems which are built on combination of machine learning and deep learning as part of the huge variety into real world electronic markets.

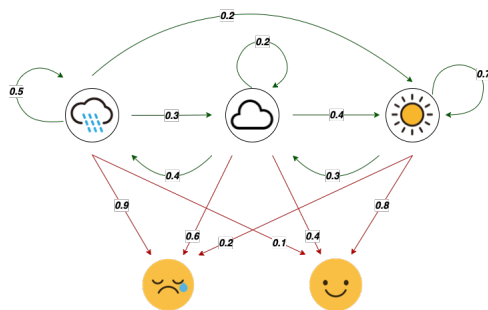


Figure 2: Hidden Markov Model Example

All numbers are hypothetical, they are used in illustration purposes only. They do not based on something.

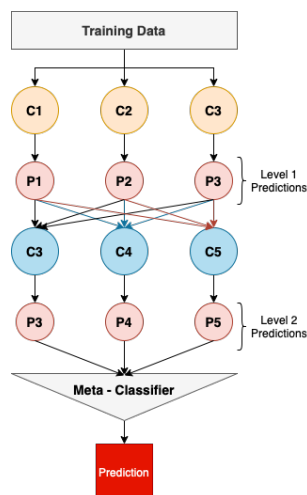


Figure 3: Classifier Stacking

2.4.1 KDD'99, NSL-KDD - UNSW-NB15

Analysis of these datasets have been conducted in few researches. Some of them are investigated individually and some of them are investigated in group of two. By using

rough-set theory (RST) this paper Al-Daweri et al. (2020) investigated differences between KDD'99 and UNSW-NB15 dataset. Other than RST, back-propagation neural network (BPNN), and lastly on of the cuttlefish algorithm discrete variant has been used. The goal of this experiment was to determine the relative importance of the features in the two datasets. For each malicious attempt in both datasets, the research found a few key traits that may be used to identify them. The results of this study's analysis are intended to assist cybersecurity academics in establishing an IDS model that to achieve high accuracy and as mentioned in section 3 this model should be lightweight in order to run simultaneously with network flow. There is one more research Moustafa and Slay (2016) for the evaluation of these systems based on analysis of these datasets to prove and support to grow all kind of attack detection systems. Gathered dataset analysis can help to improve making decision of the system that runs behind the curtain of these systems.

2.4.2 CIC-IDS-2017

In order to deal with the problem of data dimensionality, researchers use feature selection. Lowers computational time and reduces dimensionality's curse effects by eliminating features. This can improve predictivity of performance by reducing the dimensionality of data Kurniabudi et al. (2020). Selecting an optimum feature subset that represents the dataset in its whole is one component of dimensional reduction Sheikhpour et al. (2017).

3 Methodology

Today's world's technological features and solution comes with different problems in different areas. One of the most important problem is security in networks. This doesn't mean it just includes network communication with World Wide Web (WWW), it can also mean a communication between devices in houses, coffee shops or in companies. Network can be created with a connectivity through more than one device. This communication can cause different security breaches most importantly effects personal life and privacy of users. One of the other way of these effects is for businesses. This can effect business continuity, customers' privacy, and communication in company between business partners or workers in teams. Being prepared for all kind of attacks is not easy because of exponentially rising technological features in devices and softwares depending on working areas. This means while technological features rising uncontrollable way because of companies are trying to catch up with different needs for different areas, gaps are rising with them in the same way. Thus, different type of attacks and security breaches are getting hard to identify and stop before they harm any system in any network. Antivirus companies have created a solution with building a common repository which includes different identifications for same type of viruses or attacks and same identifications for different type of viruses or attacks. This repository made things easier at some point before needs and features got out of control. Deep learning usage have become really important when it came to detecting different type of attacks or new type of attacks which are known as zero-day attacks before even they got into a system. Real question with this approach is how deep learning can provide this type of protection and how it works.

The area of machine learning (ML) has made a number of noteworthy improvements in complex learning algorithms and effective pre-processing methods during the past few decades. One of these developments was the development of artificial neural networks into progressively deep neural network topologies with better learning capacities, or deep

learning Goodfellow et al. (2016). Deep Learning has already proven itself to be superior to humans in a limited number of applications in restricted contexts. There are, however, a number of hurdles that must be surmounted before analytical models may be successfully implemented in real-world corporate environments Madani et al. (2018). Data bias and drift in addition to the mitigation of black-box qualities must be taken into account when selecting an appropriate implementation option from a wide range of possibilities available. To have a basic comprehension of the subject matter, it is required to distinguish numerous relevant terminology and concepts from each other. First, the fundamentals of artificial intelligence must be laid out, and then machine learning algorithms and artificial neural networks and deep neural networks can be differentiated. Figure 4 summarizes the hierarchical relationships between these terms.

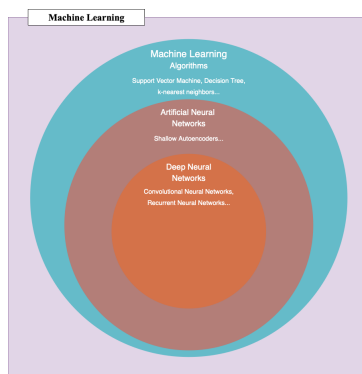


Figure 4: Machine Learning Concepts

Regression models, instance-based algorithms, decision trees, Bayesian approaches, and artificial neural networks are just few of the many kinds of machine learning algorithms available in the area, each of which comes in different specifications and variants based on the learning job at hand.

Particularly intriguing is the family of ANNs, which may be easily modified for use in any of the three ML applications. ANNs are computational models made up of interconnected "artificial neurons," which are inspired by the way information is processed in biological systems. Each neuronal connection, identical to a synapse in a biological brain, delivers signals, the strength of which can be amplified or dampened by a weight that is continuously modified throughout the learning process. The activation function determines the threshold above which signals are passed on to the next neuron for processing. Neurons are typically arranged in multilayered networks. The information is fed into the system at the input layer. Pieces of product photos are examples of the type of data we can submit. Further, the final product is generated by an output layer Goodfellow et al. (2016). In this specific scenario, the end outcome may be as simple as classifying the products into different groups. Hidden layers of code, numbering zero or more, are in charge of figuring out how to produce outputs that aren't linear in their inputs Bishop (2006). The learning algorithm is unable to discover the layer and neuron counts, along with other property selections like the learning rate and activation function Janiesch et al. (2021). The hyperparameters of a model are those variables that must be adjusted by hand or calculated via an optimization procedure. Many modern neural networks have several hidden layers and are structured in a hierarchical fashion. Additionally, in contrast to basic ANNs, they typically make use of complex neurons. Instead

of a straightforward activation function, they might employ more complex procedures or numerous activations within a single neuron. As a result of these features, deep neural networks can be given unprocessed data as input and will learn the appropriate representation for the task at hand on their own. This fundamental feature, known as deep learning, is at the heart of these networks (Fig. 4).

With all these information and deep learning usage in the research, followed steps can be found in the Fig 5. These steps may include different subsections inside them depending on preparation step and outcomes at the end. After dataset search, these subsection will be on preparation step and they can be divided by exploratory data analysis, dataset feature distributions, and experimental analysis to understand their differences. These datasets are; KDD'99 Hettich and Bay (1999), NSL-KDD of New Brunswick (2010), UNSW-NB15 Moustafa and Slay (2015) , Moustafa and Slay (2016) , Moustafa et al. (2019), Moustafa et al. (2017), Sarhan et al. (2021), CIC-IDS-2017 Sharafaldin. et al. (2018).

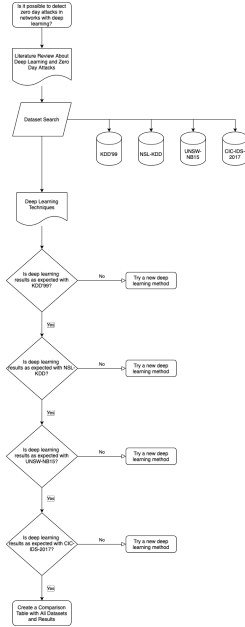


Figure 5: Followed Research Steps and Activities

3.1 Datasets & Feature Distribution

Main reason of choosing different and more than one dataset is basically try to create a test environment for different deep learning models and different test cases. These test cases may vary depending of datasets unique features and different attack types from each other. As can be known, features can vary to different attacks, this will be the main part of this study. It can be seen in Subsection 2.4.1, and 2.4.2 feature selections are different from each other. These selection will be discussed in Section 4 and it is going to be enlighten the main conclusion about the research in Section 6 step by step with graphs and numerical explanations.

4 Design Specification

Design specification are not solid for this research because of testing different deep learning models and to show differences between machine learning and deep learning in intrusion detection systems. At first architectural design started with creating a supervised deep learning models for all datasets in the research. During the literature review process and investigating related works with those datasets, some of the datasets do not have any work related under unsupervised learning models and some of them do not have any work under machine learning process. Decision of using different machine learning for related datasets (UNSW-NB15 CIC-IDS-2017) to explain general feature selection part became the most important thing in this project rather than creating a solid deep learning model at the end.

4.1 Architecture of Deep Learning Models

4.1.1 Unsupervised Deep Learning Models

Although there are more than one unsupervised deep learning models, choosing right model is depending on the data is being used in the project or data is being investigated. In our case autoencoders will be the best because of number of features are higher than normal datasets. Using classifier stacking (Figure 3) will improve learning process without knowing the outcome and it can increase accuracy of the model. As an addition to this model, batch normalization will be used to investigate the contribution of this model's accuracy and learning stability.

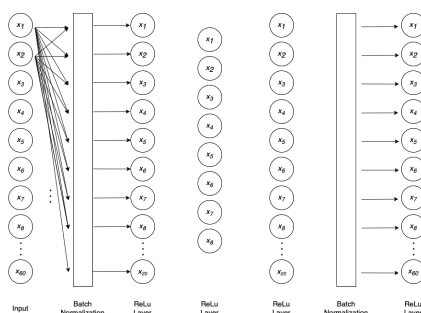


Figure 6: Autoencoders with Batch Normalization

When training extremely deep neural networks, batch normalization is used to uniformly average the inputs to a layer at the mini-batch level. When training deep networks, this stabilizes the learning process and significantly shortens the training time. For the decoder to be able to accurately recreate the input, the encoder must first extract its distinguishing features. When an encoder's output space is less than its input space, it memorizes the input rather than learning from it. This creates an autoencoder in the middle of learning process named undercomplete autoencoder.

4.1.2 Supervised Deep Learning Models

Supervised deep learning models are basically a subset of machine learning process as it mentioned in Section 3 and graphically showed the part of it in systems in Figure 4 as a

venn diagram. They contain three main layers in their models; input layer, hidden layer and output layer.

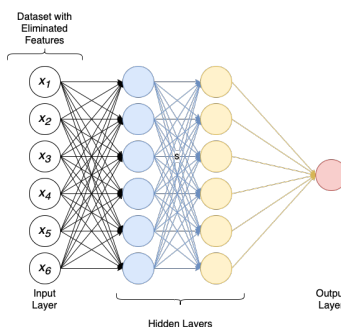


Figure 7: Artificial Neural Network Diagram

As it can be understandable from their names, input layer will be the dataset that is given by us to start learning process set up by our specifications. These specifications starts with feature elimination and selection and ends up with classified value in the output layer after learning process. Hidden layers contain number of neurons we want to add to the learning model. Number of these layers will increase the duration of learning process but they also increase accuracy of learned values. Although, number of neurons have hidden threshold depending on dataset that is being fed to system. Threshold stops the learning process and starts to give average results of accuracy or loss values as an output. In order to stop these process, accuracy or loss values need to be monitored and set up as a threshold. By doing this, model can compare new results with old ones and when learning stops it can create a base line that indicates best result of accuracy according used dataset with used specifications for used model.

4.2 Software Requirements

As discussed in architectures of deep learnings part of the report, each architecture and design needs different software requirements. These requirements change in library wise. Main softwares need to conduct this experimental process is basically Jupyter Lab and Python 3. After installing these softwares to the system, some libraries for Python 3 has to be installed. These required libraries can be found on Table 1.

Library Names	Library Names	Library Names
pandas	pandas	pandas
numpy	numpy	numpy
tensorflow	seaborn, matplotlib	seaborn, matplotlib
sklearn	keras	sklearn
	sklearn	keras
	xgboost, catboost, lightgbm, hdbscan	tensorflow
	pyod	

(a) KDD'99 & NSL-KDD Software Requirements

(b) UNSW-NB15 Software Requirements

(c) CIC-IDS-2017 Software Requirements

Table 1: Library Requirements for Each Dataset

4.3 Functionality of Deep Learning Models

When it comes to functionality, deep learning models are more functional than machine learning concept if case includes huge about of data with labels. Labelled datasets are important in functionality in terms of creating an accurate system. If dataset is not labelled, this system has to turn into an unsupervised model as it mentioned in Section 4.1.1 and as an example of system design Figure 6. Even when data is not labelled, unsupervised deep learning models can improve themselves in time because of their ability of continuity. Continuity means, these systems can be improved in time because of the dataset getting bigger and bigger. New entries in the dataset can create a big difference of the learning process. Thus, even these deep learning models were not achieved a high accuracy, they can improve in time rather than machine learning models. If process duration is enough to increase accuracy, these systems can even find new things what they programmed to find even users don't even know they are exist.

5 Implementation

In the beginning of the implementation process, decision of using four different datasets was challenging because of different features needs to reduced with different type of classifications. This classifications started with investigating datasets with importing them into laboratory environment which is Jupyter Lab in this research. As it can be seen in Figure 17 as an example of one of the datasets used in the research, due to amount of data in all datasets these investigation for feature selection process cannot be done by hand. But, for investigation purposes correlation heatmap and feature importance graphs are being used to visualize feature selection process for Dataset 6.2 and Dataset 6.3.

6 Evaluation

6.1 Experiment / Case Study 1 (KDD'99 & NSL-KDD)

6.1.1 Unsupervised Deep Learning Model

Implementation of unsupervised deep learning model to these datasets was not in experimental process plan at first. But during the related work and literature review, I couldn't find any related works with unsupervised deep learning including KDD'99 and NSL-KDD datasets. After getting unbelievably high accuracy (Figure 12 with using artificial neural networks in supervised deep learning section of these datasets, I wanted to see in these datasets can give high accuracy without having any label for the attacks. As it mentioned in Section 4.1.2, since dataset won't have any label in it, results can not classify attack types. This means if model detects any attack with investigating feature values it is going give a result of "malicious", it won't be able say attack type in this case.

After deleting 'outcome' label from the dataset, an ANN was crated based on Figure 6 without batch normalization. This network gave an output with 0.1191 loss of accuracy. Different than standard unsupervised network this loss of accuracy is really higher than usual.

```

prediction = model.predict(x_normal_test)
predScore_1 = np.sqrt(metrics.mean_squared_error(prediction,x_normal_test))

prediction = model.predict(x_normal)
predScore_2 = np.sqrt(metrics.mean_squared_error(prediction,x_normal))

prediction = model.predict(x_attack)
predScore_3 = np.sqrt(metrics.mean_squared_error(prediction,x_attack))

print("Out of Sample Normal Score (RMSE): {}".format(predScore_1))
print("All Sample Normal Score (RMSE): {}".format(predScore_2))
print("Attack Underway Score (RMSE): {}".format(predScore_3))

760/760 [=====] - 1s 643us/step
3040/3040 [=====] - 2s 792us/step
12399/12399 [=====] - 8s 643us/step
Out of Sample Normal Score (RMSE): 0.42966980763720186
All Sample Normal Score (RMSE): 0.415528409012631
Attack Underway Score (RMSE): 0.5196021720180707

```

Figure 8: KDD'99 & NSL-KDD Unsupervised Results without Batch Normalization

Since this dataset created in 1999, I was expecting really high accuracy in supervised network side of this evaluation but in unsupervised part's results were bigger than I have ever expected. This doesn't mean we can't lower this loss result. So, to make this decrease batch normalization had been used to create more accurate unsupervised network.

```

prediction = model.predict(x_normal_test)
predScore_1 = np.sqrt(metrics.mean_squared_error(prediction,x_normal_test))

prediction = model.predict(x_normal)
predScore_2 = np.sqrt(metrics.mean_squared_error(prediction,x_normal))

prediction = model.predict(x_attack)
predScore_3 = np.sqrt(metrics.mean_squared_error(prediction,x_attack))

print("Out of Sample Normal Score (RMSE): {}".format(predScore_1))
print("All Sample Normal Score (RMSE): {}".format(predScore_2))
print("Attack Underway Score (RMSE): {}".format(predScore_3))

760/760 [=====] - 1s 625us/step
3040/3040 [=====] - 2s 620us/step
12399/12399 [=====] - 8s 631us/step
Out of Sample Normal Score (RMSE): 5.117971347773509
All Sample Normal Score (RMSE): 3.701375080271995
Attack Underway Score (RMSE): 0.6156522954989787

```

Figure 9: KDD'99 & NSL-KDD Unsupervised with Batch Normalization

While creating this network with batch normalization, I decided to add 3 layers in hidden layer part of the neural network. This layers are 20 inputs of ReLu layer, 3 inputs of ReLu layer and 20 inputs of ReLu layer connected each other respectively (Figure 10). In the middle of the hidden layer part's number or inputs was selected randomly in this case. This training of network ended up with 0.61 deviation on identifying attacks in the dataset.

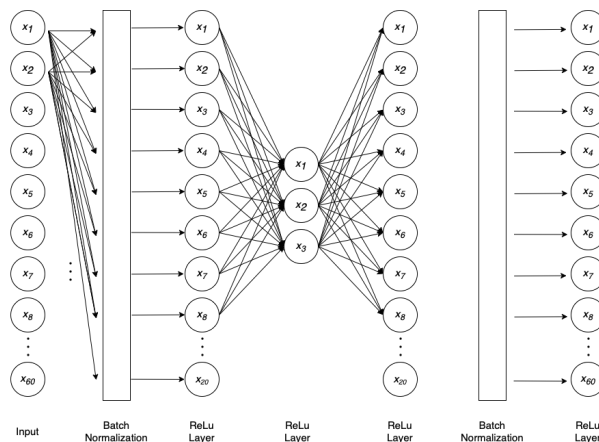


Figure 10: Autoencoder Design

While testing this neural network, hidden layer's middle input number was increased to 8 from 3 to see the results of deviation in the training. This increase of hidden layer inputs helped us to understand, the hidden layer's learning threshold is 3. This means every increase on hidden layer after 3, it is going decrease the accuracy on the network because learning process stops and due to number of inputs, network starts to getting confused since there are no labels indicate whether it's an attack or not.

```

prediction = model.predict(x_normal_test)
predScore_1 = np.sqrt(metrics.mean_squared_error(prediction,x_normal_test))

prediction = model.predict(x_normal)
predScore_2 = np.sqrt(metrics.mean_squared_error(prediction,x_normal))

prediction = model.predict(x_attack)
predScore_3 = np.sqrt(metrics.mean_squared_error(prediction,x_attack))

print("Out of Sample Normal Score (RMSE): {}".format(predScore_1))
print("All Sample Normal Score (RMSE): {}".format(predScore_2))
print("Attack Underway Score (RMSE): {}".format(predScore_3))

760/760 [=====] - 1s 823us/step
3840/3840 [=====] - 3s 921us/step
12399/12399 [=====] - 10s 794us/step
Out of Sample Normal Score (RMSE): 3.1451839089486535
All Sample Normal Score (RMSE): 6.64333453974772
Attack Underway Score (RMSE): 9.93526082792704

```

Figure 11: KDD'99 & NSL-KDD Unsupervised with Batch Normalization Over Threshold

6.1.2 Supervised Deep Learning Model

Although, this section comes after unsupervised model section, decision of creating deep learning model started underlying supervised deep learning models. There are few researches and projects about this dataset covering this subsection since this dataset created at 1999 and this is one of the most popular datasets in order to understand the attack behaviors with dataset features. Intrusion detection systems based on this dataset for a while. Starting at that time with increasing attack types and attacks with different signatures created a need for automated detection system based on the most reliable attack features which was KDD'99 dataset.

As always creation of network started with implementing the dataset into our laboratory environment. Then attacks were separated with their feature values into a data frame named 'attacks'. This data frame will be the backbone of our deep learning model. Since every attack were labeled, training section of this network will create a high accuracy in testing process as can be seen in Figure 12.

```

pred = model.predict(X_test)
pred = np.argmax(pred,axis=1)
y_eval = np.argmax(y_test,axis=1)
score = metrics.accuracy_score(y_eval, pred)
print("Validation Score: {}".format(score))

3860/3860 [=====] - 4s 1ms/step
Validation Score: 0.9955062911923307

```

Figure 12: KDD'99 & NSL-KDD Supervised Results

6.2 Experiment / Case Study 2 (UNSW-NB15)

6.2.1 Unsupervised Deep Learning Model

In Section 2.3, machine learning and deep learning differences were mentioned. To see these differences' outcomes on result I used different machine learning models with using this dataset before creating unsupervised deep learning model. In order to create

use machine learning models, there are 4 different classification models have been used to train the dataset with different classified inputs. These classifications are Random Forest Classifier, Extra Trees Classifier, XGB Classifier, and LGBM Classifier. Before using these classifications, since this will be an unsupervised model label of each features outcome need to be dropped from the dataset. As a results of this classifiers in terms of evaluation, an F1 score is the mathematical midpoint between how well something was remembered and how accurately something was remembered. It's a metric for quantifying and comparing results.

- F1 Score: 0.952 - Random Forest Classifier
- F1 Score: 0.95 - Extra Trees Classifier
- F1 Score: 0.95 - XGB Classifier
- F1 Score: 0.951 - LGBM Classifier

When it comes to finding outliers in multivariate data, there is no better option than PyOD. These two terms, Outlier Detection and Anomaly Detection, are used interchangeably to describe this fascinating and difficult area of study. It includes more than 40 detection algorithms but in this case 9 of them were used to compare results and find the best result of them. As it can be seen in Figure 13, OCSVM (One Class SVM) has given the best result of given random values with more than 40 features of each attack row in the dataset. Result of % 55 of accuracy without any labels might look enough for anomaly detection with machine learning but when deep learning model created with using autoencoder (Figure 6) without batch normalization, it can be seen in Figure 14 model loss is almost %42. This indicates when the model investigates an attack row from the dataset big amount of value of the features are not being used or lost while trying to decide whether it's an attack or not.

```

Acc of train: 0.55653
F1_weighted of train: 0.47694
Acc of test: 0.54994
F1_weighted of train: 0.46759

```

Figure 13: UNSW-NB15 Machine Learning Unsupervised Best Accuracy

```

=====
Total params: 9,242
Trainable params: 9,242
Non-trainable params: 0
=====
Epoch 1/30
5798/5798 [=====] - 13s 2ms/step - loss: 42.2371 - val_loss: 41.1609
Epoch 2/30
5798/5798 [=====] - 10s 2ms/step - loss: 42.0878 - val_loss: 41.1609
Epoch 3/30
5798/5798 [=====] - 11s 2ms/step - loss: 42.0890 - val_loss: 41.1609
Epoch 4/30
5798/5798 [=====] - 11s 2ms/step - loss: 42.0879 - val_loss: 41.1609
Epoch 5/30
5798/5798 [=====] - 10s 2ms/step - loss: 42.0877 - val_loss: 41.1609
Epoch 6/30
5798/5798 [=====] - 11s 2ms/step - loss: 42.0890 - val_loss: 41.1609
Epoch 7/30
5798/5798 [=====] - 10s 2ms/step - loss: 42.0895 - val_loss: 41.1609
Epoch 8/30
5798/5798 [=====] - 11s 2ms/step - loss: 42.0885 - val_loss: 41.1609
Epoch 9/30
5798/5798 [=====] - 11s 2ms/step - loss: 42.0887 - val_loss: 41.1609
Epoch 10/30
5798/5798 [=====] - 11s 2ms/step - loss: 42.0884 - val_loss: 41.1609

```

Figure 14: UNSW-NB15 Unsupervised Accuracy Loss

6.2.2 Supervised Deep Learning Model

After seeing the huge loss in unsupervised model with this dataset, same classification models have been used to start this supervised deep learning model. But this time I decided to use CatBoost library to see how much it can increase classification value. After using CatBoost boosting algorithm on the dataset accuracy increased %0.01 at the end.

```
RFC = RandomForestClassifier(n_estimators=150, random_state=42, n_jobs=-1)
RFC = RFC.fit(X_train,y_train)
pred = RFC.score(X_test, y_test)
name = str(type(RFC)).split(".")[1][:-2]
print("Acc: %0.5f for the %s" % (pred, name))
Acc: 0.95166 for the RandomForestClassifier
```

Figure 15: UNSW-NB15 Machine Learning Supervised Best Accuracy Before CatBoost

```
import catboost
CBC = catboost.CatBoostClassifier(iterations=3000, eval_metric='AUC', use_best_model=True, task_type='CPU')
CBC = CBC.fit(X_train,y_train, eval_set=(X_test, y_test))
pred = CBC.score(X_test,y_test)
name = str(type(CBC)).split(".")[1][:-2]
print("Acc: %0.5f for the %s" % (pred, name))
Acc: 0.95306 for the CatBoostClassifier
```

Figure 16: UNSW-NB15 Machine Learning Supervised Best Accuracy After CatBoost

In order to create deep learning models and decrease the training and testing duration while increasing accuracy of the network, low correlated features need to be eliminated from the dataset. To find these low correlations, correlation map can be examined (Figure 17).

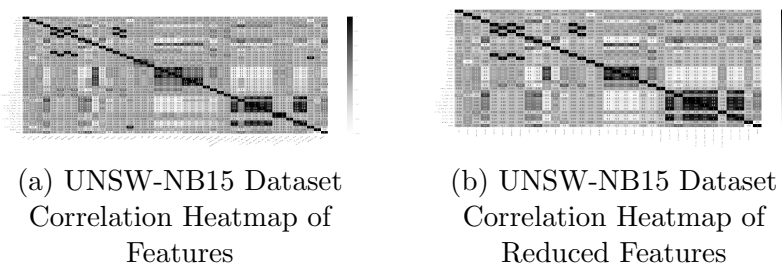


Figure 17: UNSW-NB15 Dataset Correlation Matrix Before & After

But as mentioned in the Methodology 3 part in this dataset number of features are higher than usual, we will not be able to decide which one is correlated and which one is not. To make this decision easier, low correlated features have been printed related to 'attack_cat' feature.

After eliminating the low correlated feature our correlation map will be look like this Figure 17. Still, this dataset's features are unbelievable high even after elimination. We can make guess that our deep learning model will have high loss values even it can achieve high accuracy in this dataset. The number of attack types in this dataset creates this need number features. Every attack type uses different type of feature as a signature. At the end attack types and feature numbers are directly proportional in the dataset.

To understand the importance of feature in this dataset, feature importance library has been used and it can be find in Figure 18. According to this feature importance graph, we have to eliminate half of the features in order to create an accurate network but in this case our model will not be able to detect most of the attack types and it will be like one type of intrusion detection system.

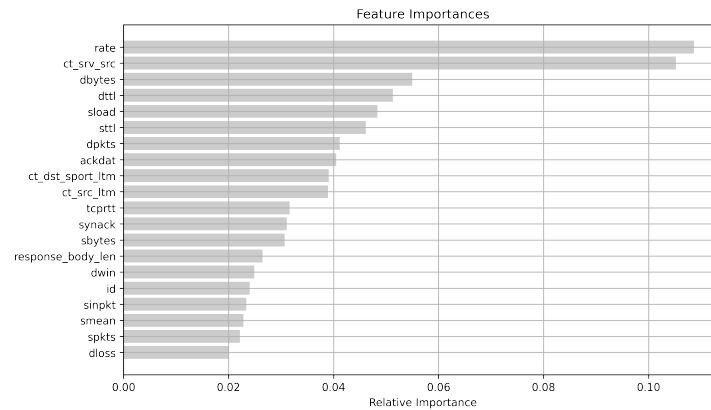


Figure 18: UNSW-NB15 Dataset Feature Importance Graph

As we guessed before our deep learning model achieved %93.5 of accuracy while losing %13 amount of data (Figure 19).

```

eval_model = classifier.evaluate(X_train, y_train)
print(eval_model)

eval_model = classifier.evaluate(X_test, y_test)
print(eval_model)

predictions = classifier.predict(X_test)
predictions = (predictions>0.80)

mse = np.mean(np.power(X_test - predictions, 2), axis=1)
error_df = pd.DataFrame({'reconstruction_error': mse, 'true_class': y_test.values.reshape(1,-1)[0]})
error_df.describe()

6442/6442 [=====] - 5s 778us/step - loss: 0.1317 - accuracy: 0.9346
[0.1317436397075653, 0.9345681071281433]
1611/1611 [=====] - 1s 783us/step - loss: 0.1328 - accuracy: 0.9345
[0.13280874490737915, 0.9345299601554871]

```

Figure 19: UNSW-NB15 Supervised Accuracy with ANN

6.3 Experiment / Case Study 3 (CIC-IDS-2017)

Last part of this experimental process is ends with creating a deep learning model with CIC-IDS-2017 dataset. The difference between this dataset and UNSW-NB15 dataset is, CIC-IDS-2017 dataset comes in different portion of pcap files seperated based attack types. To start with investigation web attack file has been chosen to implement in this process. This will help us to understand main differences and needs while creating deep learning models and real importance of feature elimination based on our intentions. Like in every other experiment this dataset features needs to be eliminated based on their correlation and relativity about attacks. I started eliminating features based on their relativity before looking at their correlations. Eliminated features are;

- Flow ID
- Source IP, Source Port

- Destination IP, Destination Port
- Protocol, Timestamp

These features are not relevant to web attacks, they are just indicating the IP addresses and connection protocols between those destinations. After this elimination we need to look at the correlation map as always like in Figure 20.

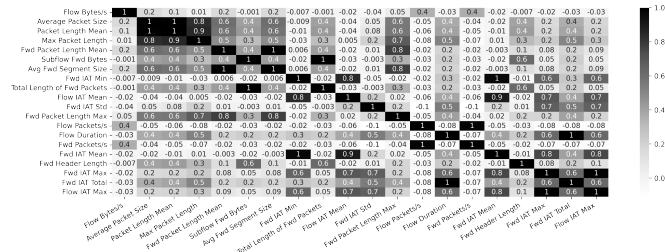


Figure 20: CIC-IDS-2017 Dataset Correlation Heatmap of Features

Even, this correlation matrix is readable than Figure 17, feature importance library has been used to reduce the mistakes of elimination. After creating an importance graph with feature importance library, can be seen in Figure 22, low correlated features are eliminated from the dataset and final correlation matrix is created with reduced features for visual purposes (Figure 21).

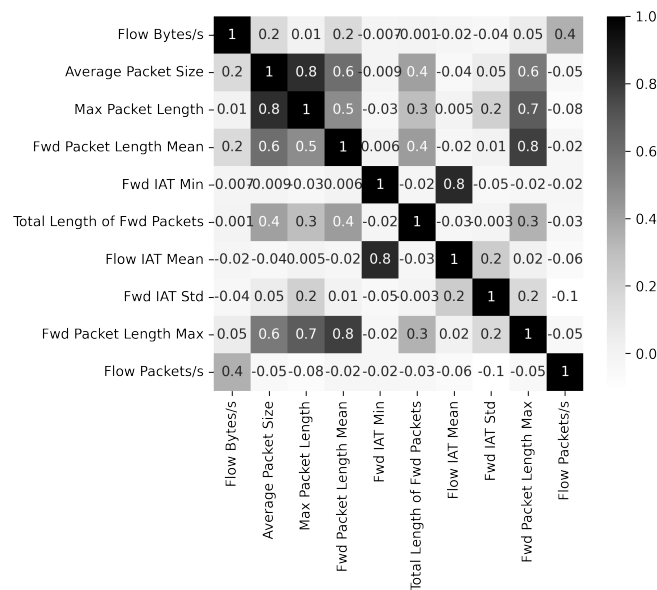


Figure 21: CIC-IDS-2017 Dataset Correlation Heatmap of Reduced Features

Deep learning model created based on Figure 7, and to fasten the training process a monitor variable was added for EarlyStopping. EarlyStopping will help our network to stop when best values are reached. This is important while training network with big datasets like this to decrease duration of learning process.

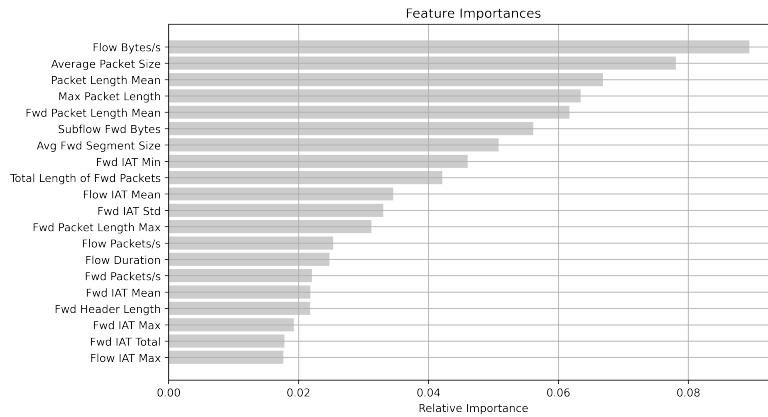


Figure 22: CIC-IDS-2017 Dataset Feature Importance Graph

As can be seen below Figure 23, network’s accuracy reached to %97 with %0.08 loss. In this experiment having a one-sided dataset created a perfect environment for feature elimination with feature importance graph. This difference is the main and most important findings in this research to understand the differences between UNSW-NB15 dataset and CIC-IDS-2017 dataset.

```

model = Sequential()
model.add(Dense(42, activation='relu', input_dim=dim))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.02))
model.add(Dense(42, activation='relu' ))
model.add(Dropout(0.02))
model.add(Dense(25, activation='relu'))
model.add(Dropout(0.02))
model.add(Dense(12, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
monitor = EarlyStopping(monitor='loss', min_delta=1e-3, patience=5, verbose=1, mode='auto', restore_best_weights=True)
history = classifier.fit(X,y_train, batch_size=64, epochs=100, validation_data=(X_t,y_test), callbacks=[
Epoch 1/100
88/88 [=====] - 1s 4ms/step - loss: 0.5756 - accuracy: 0.7153 - val_loss: 0.347
0 - val_accuracy: 0.9421
88/88 [=====] - 0s 3ms/step - loss: 0.0862 - accuracy: 0.9654 - val_loss: 0.088
2 - val_accuracy: 0.9697
Epoch 36: early stopping

```

Figure 23: CIC-IDS-2017 Dataset Accuracy with ANN

6.4 Discussion

As a results of all findings starting with the first experimental process of KDD’99 dataset. Even KDD’99 dataset was created in 1999 with bunch of different attack types deep learning models can achieve the impossible amount of accuracy. This accuracy is so high to be real in today’s networks as experienced in Case Study 2 (6.2). All datasets were chosen well connected to each other some way to understand the main differences of today’s network needs to identify attacks and today’s detection systems to learn the attack features. The main difference between KDD’99 and UNSW-NB15 is not just the realase year. They both have huge amount of data and they contain a lot of attack types, but UNSW-NB15 dataset contains too much related features to those attack types. In time attacks are started to effect different part of network, that’s why intrusion detection system are started to detect those attacks with looking their effect on network. In order to understand the difference between creating a neural network with lots of attack types

in every classification and lots of attack types with one classification, UNSW-NB15 and CIC-IDS-2017 datasets were investigated in Case 2 and Case 3 (6.3). As a result of the Case 3, it was containing only web attacks in the dataset with almost same amount of features with UNSW-NB15 dataset. But it's accuracy almost double with nearly %0 of loss value. The reason of this mainly UNSW-NB15 dataset contains every type of attack with every type of features in one dataset. This is creating a diversion in deep learning network while training the system. In my opinion, accuracy results might go higher if Bayesian Optimization applied to CatBoost on Case 2 but for the reasons have discussed about UNSW-NB15 dataset, I don't think loss values can be decreased to almost %0 like in Case 3.

7 Conclusion and Future Work

In the conclusion of main research question (Section 1) and aim of this project have been achieved and answered successfully. It is generally possible to create deep learning models to identify zero-day attacks in networks but as mentioned in 6.4 there are obstacles to achieve high success rates. Elimination of those obstacles rely of well seperated datasets in future works for intrusion detection systems and systems that work simultaneously with network flow. IDSs can achieve great success with using simple ANNs if data is well established and separated depending on attack types on individual datasets. After feeding systems' networks individually with different attack types, neural networks can learn differences better and seperate attacks from each other. To make systems ready for today's vulnerabilities networks can be fed with CIC-IDS-2017 dataset after this feeding process to make attacks more accurate neural network can be fed with CIC-IDS-2018 dataset. This way neural network can identify differences between attacks recorded on 2017 and 2018. By doing this identification networks will be ready for similar attacks and even small changes in each attack type. The neural network that created should be tested in future works using real network connection while capturing network flows to see identification duration of whole system. If this work can achieve high success rate with very low latency it can be transformed into a device connected to a cloud server and used like an external firewall. By the time of using those devices connected into different networks all around the world, all attack data that is collected by this device can be transferred into cloud server to feed the neural network to keep it updated for new type of attacks. This way neural network can be kept updated while growing it's learning capabilities.

References

- Al-Daweri, M., Zainol Ariffin, K. A., Abdullah, S. and Senan, M. (2020). An analysis of the kdd99 and unsw-nb15 datasets for the intrusion detection system, *Symmetry* **12**: 1666.
- Andreucut, M. (2022). Attack vs benign network intrusion traffic classification.
URL: <https://arxiv.org/abs/2205.07323>
- Baldi, P. (2011). Autoencoders, unsupervised learning and deep architectures, *Proceedings of the 2011 International Conference on Unsupervised and Transfer Learning Workshop - Volume 27*, UTLW'11, JMLR.org, p. 37–50.

- Bhuyan, M. H., Bhattacharyya, D. K. and Kalita, J. K. (2014). Network anomaly detection: Methods, systems and tools, *IEEE Communications Surveys Tutorials* **16**(1): 303–336.
- Bilge, L. and Dumitras, T. (2012). Before we knew it: An empirical study of zero-day attacks in the real world, *Proceedings of the 2012 ACM Conference on Computer and Communications Security, CCS '12*, Association for Computing Machinery, New York, NY, USA, p. 833–844.
URL: <https://doi.org/10.1145/2382196.2382284>
- Bishop, C. M. (2006). Pattern recognition and machine learning, pp. X–728.
- Chapman, C. (2016). Chapter 1 - introduction to practical security and performance testing, in C. Chapman (ed.), *Network Performance and Security*, Syngress, Boston, pp. 1–14.
URL: <https://www.sciencedirect.com/science/article/pii/B9780128035849000019>
- Chen, C.-M., Chen, Y.-L. and Lin, H.-C. (2010). An efficient network intrusion detection, *Comput. Commun.* **33**(4): 477–484.
URL: <https://doi.org/10.1016/j.comcom.2009.10.010>
- Comar, P. M., Liu, L., Saha, S., Tan, P.-N. and Nucci, A. (2013). Combining supervised and unsupervised learning for zero-day malware detection, *2013 Proceedings IEEE INFOCOM*, pp. 2022–2030.
- Deng, L. and Yu, D. (2014). Deep learning: Methods and applications, *Found. Trends Signal Process.* **7**(3–4): 197–387.
URL: <https://doi.org/10.1561/20000000039>
- Frei, S. (2009). *Security Econometrics - The Dynamics of (In)Security*, PhD thesis.
- Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep learning*, MIT press.
- Hamid, Y., Muthukumarasamy, S. and Ranganathan, B. (2016). Ids using machine learning -current state of art and future directions, *British Journal of Applied Science and Technology* **15**: 1–22.
- Hebb, D. O. (1949). The organization of behavior: A neuropsychological theory.
URL: <https://doi.org/10.1002/sce.37303405110>
- Hettich, S. and Bay, S. D. (1999). The uci kdd archive [<http://kdd.ics.uci.edu>]. irvine, ca: University of california, department of information and computer science.
- Hinton, G. E. (2009). Deep belief networks, *Scholarpedia* **4**(5): 5947. revision #91189.
- Ioulianou, P., Vasilakis, V., Moscholios, I. and Logothetis, M. (2018). A signature-based intrusion detection system for the internet of things, *Information and Communication Technology Form*, AUT.
URL: <https://eprints.whiterose.ac.uk/133312/>
- Ioulianou, P., Vassilakis, V. and Moscholios, I. (2018). A signature-based intrusion detection system for the internet of things.

- Janiesch, C., Zschech, P. and Heinrich, K. (2021). Machine learning and deep learning, *Electronic Markets* **31**(3): 685–695.
URL: <https://doi.org/10.1007/s12525-021-00475-2>
- Jordan, M. I. and Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects, *Science* **349**(6245): 255–260.
URL: <https://www.science.org/doi/abs/10.1126/science.aaa8415>
- Khraisat, A., Gondal, I., Vamplew, P. and Kamruzzaman, J. (2019). Survey of intrusion detection systems: techniques, datasets and challenges, *Cybersecurity* **2**(1): 20.
URL: <https://doi.org/10.1186/s42400-019-0038-7>
- Kurniabudi, Stiawan, D., Darmawijoyo, Bin Idris, M. Y., Bamhdi, A. M. and Budiarto, R. (2020). Cicides-2017 dataset feature analysis with information gain for anomaly detection, *IEEE Access* **8**: 132911–132921.
- LeCun, Y., Bengio, Y. and Hinton, G. (2015). Deep learning, *Nature* **521**(7553): 436–444.
URL: <https://doi.org/10.1038/nature14539>
- Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998). Gradient-based learning applied to document recognition, *Proceedings of the IEEE* **86**(11): 2278–2324.
- Liao, H.-J., Richard Lin, C.-H., Lin, Y.-C. and Tung, K.-Y. (2013). Intrusion detection system: A comprehensive review, *Journal of Network and Computer Applications* **36**(1): 16–24.
URL: <https://www.sciencedirect.com/science/article/pii/S1084804512001944>
- Louridas, P. and Ebert, C. (2016). Machine learning, *IEEE Software* **33**: 110–115.
- Madani, A., Arnaout, R., Mofrad, M. and Arnaout, R. (2018). Fast and accurate view classification of echocardiograms using deep learning, *npj Digital Medicine* **1**(1): 6.
URL: <https://doi.org/10.1038/s41746-017-0013-1>
- Moustafa, N., Creech, G. and Slay, J. (2017). *Big Data Analytics for Intrusion Detection System: Statistical Decision-Making Using Finite Dirichlet Mixture Models*, Springer International Publishing, Cham, pp. 127–156.
URL: https://doi.org/10.1007/978-3-319-59439-2_5
- Moustafa, N. and Slay, J. (2015). Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set), *2015 Military Communications and Information Systems Conference (MilCIS)*, pp. 1–6.
- Moustafa, N. and Slay, J. (2016). The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set, *Information Security Journal: A Global Perspective* **25**(1-3): 18–31.
URL: <https://doi.org/10.1080/19393555.2015.1125974>
- Moustafa, N., Slay, J. and Creech, G. (2019). Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks, *IEEE Transactions on Big Data* **5**(4): 481–494.
- of New Brunswick, U. (2010). Nsl-kdd dataset.
URL: <https://www.unb.ca/cic/datasets/nsl.html>

- Oja, E. (1982). Simplified neuron model as a principal component analyzer, *Journal of Mathematical Biology* **15**(3): 267–273.
URL: <https://doi.org/10.1007/BF00275687>
- Rumelhart, D. E., Hinton, G. E. and Williams, R. J. (1986). *Learning Internal Representations by Error Propagation*, MIT Press, Cambridge, MA, USA, p. 318–362.
- Sarhan, M., Layeghy, S., Moustafa, N. and Portmann, M. (2021). NetFlow datasets for machine learning-based network intrusion detection systems, *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, Springer International Publishing, pp. 117–135.
URL: https://doi.org/10.1007/978-3-030-72802-1_9
- Schneier, B. (2000). Crypto-gram.
URL: <https://www.schneier.com/crypto-gram/archives/2000/0915.html>
- Seo, W. and Pak, W. (2021). Real-time network intrusion prevention system based on hybrid machine learning, *IEEE Access* **9**: 46386–46397.
- Sharafaldin., I., Habibi Lashkari., A. and Ghorbani., A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization, *Proceedings of the 4th International Conference on Information Systems Security and Privacy - ICISSP*, INSTICC, SciTePress, pp. 108–116.
- Sheikhpour, R., Sarram, M. A., Gharaghani, S. and Chahooki, M. A. Z. (2017). A survey on semi-supervised feature selection methods, *Pattern Recognition* **64**: 141–158.
URL: <https://www.sciencedirect.com/science/article/pii/S0031320316303545>
- Society, I. (2019).
URL: <https://www.internetsociety.org/resources/ota/2019/2018-cyber-incident-breach-trends-report/>
- Uppuluri, P. and Sekar, R. (2001). Experiences with specification-based intrusion detection, in W. Lee, L. Mé and A. Wespi (eds), *Recent Advances in Intrusion Detection*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 172–189.
- Venter, H. S. and Eloff, J. H. P. (2003). A taxonomy for information security technologies, *Comput. Secur.* **22**: 299–307.
- Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., Zhu, H., Gao, M., Hou, H. and Wang, C. (2018). Machine learning and deep learning methods for cybersecurity, *IEEE Access* **6**: 35365–35381.
- Zeng, H., Liu, Z. and Cai, H. (2020). Research on the application of deep learning in computer network information security, *Journal of Physics: Conference Series* **1650**(3): 032117.
URL: <https://doi.org/10.1088/1742-6596/1650/3/032117>
- Çakır, B. (2019). *Zero-day attack detection with deep learning*, Master’s thesis, Middle East Technical University.