Resource Consumption Analysis of Distributed Machine Learning Models for 6G Security

University of Turku
Faculty of Technology
Master of Science (Tech) Thesis
Information and Communication Technology (Cyber Security)
May 2024
Md Muzammal Hoque

Supervisors:
Mohammad Tahir (University of Turku)
Petri Sainio (University of Turku)
Ijaz Ahmad (VTT Technical Research Centre of Finland)

UNIVERSITY OF TURKU
Faculty of Technology

Md Muzammal Hoque: Resource Consumption Analysis of Distributed Machine Learning Models for 6G Security

Master of Science (Tech) Thesis, 66 p.
Information and Communication Technology (Cyber Security)
May 2024

Communications networks have become increasingly complex environments due to the massive increase in the number of communicating nodes and diverse services with unique requirements. Therefore, machine learning has become extremely important from the access to backhaul and core networks, as well as various technologies required for the smooth operations of different tasks and services within those networks. The overall complexity of networked environments and increasing volumes of data further complicates the network security landscape. Machine learning with its various techniques and tools, thus, has become vital for network security. In 6G network security, the promises of machine learning are vast, from preventive measures to detection to response and remediation. However, machine learning requires a huge amount of resources mainly due to the fact that machine learning operates on data and data volumes are consistently rising. This work studied and investigated the resource consumption of machine learning techniques used for network security to provide insights into the potential resource implications of deploying machine learning in 6G security.

The thesis explored a wide range of state-of-the-art resource-efficient Machine learning based security solutions to find out the key resources consumed by those solutions and the key enablers of resource efficiency for those solutions. In particular, the thesis focused on investigating the resource consumption of distributed learning for 6G networks in terms of computing, memory, bandwidth, energy, latency, and human resources. Distributed machine learning is highly relevant to the context of 6G, as it can meet the future 6G requirement of processing substantial amounts of data generated from numerous devices while preserving data privacy and security. The thesis presents an experimental and comparative analysis of the Federated Learning (FL) and Split Learning (SL) based network security solutions, which are the two most popular distributed learning in terms of resource consumption fingerprinting. The finding shows that both models perform well, while Federated Learning appears to have a slight edge over Split Learning in terms of precision and F1 score. However, the differences are quite small. In terms of resource consumption fingerprinting, we observed that both of them have their advantages and shortcomings. In terms of CPU usage, SL had higher CPU usage, while FL had higher peaks and variability. In terms of memory usage, FL was more memory efficient than the SL. Finally, SL was more time and power-efficient and had lower CO_2 emission.

Keywords: Security; 6G; Machine Learning; Network Security; distributed learning; 6G security; sustainability; DNN;Split Learning; Federated Learning; Resource Consumption; Resource Efficiency

Acknowledgements

I want to express my deep gratitude to my supervisor at the VTT Technical Research Centre of Finland, Ijaz Ahmad (PhD.), and my supervisors from the University of Turku, Tahir Mohammad (PhD.) and Petri Sainio. Their priceless advice, continuous assistance, and constructive feedback have been crucial in enhancing my comprehension of the topic and the meticulousness and profundity of this thesis. Their expertise and knowledge have illuminated every stage of this research, and their insights have challenged and extended my intellectual development. I am sincerely thankful for their patience and tremendous academic support. I am also grateful for the opportunity to contribute to the ongoing research in Machine Learning and 6G Network Security, particularly from the perspective of resource consumption and efficiency. This work has not only broadened my view but also instilled in me the confidence that it may contribute to paving the way toward a sustainable and innovative future. Hence, I would like to thank the VTT Technical Research Centre of Finland and the SUNSET-6G Project for providing me with this opportunity. Finally, I would like to thank all those who directly or indirectly guided me during this journey. Your support has been a source of strength and inspiration.

Contents

List of Figures

List of Tables

List of acronyms

5G	Fifth-Generation Mobile Technology
6G	Sixth-Generation Wireless Technology
AdaBoost	Adaptive Boosting
AI	Artificial Intelligence
ANN	Artificial and Neural Networks
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DBSCAN	Density-Based Clustering Algorithms
DDoS	Distributed Denial-of-Service
DL	Deep Learning
DLTs	Distributed Ledger Technologies
DNS	Domain Name System
DoS	Denial of Service
DRL	Deep Reinforcement Learning
DT	Decision Tree

FFNN	Feed-Forward Neural Network
FL	Federated Learning
GA	Genetic Algorithm
GPU	Graphics Processing Unit
GRU	Gated Recurrent Units
IDS	Intrusion Detection System
IoE	Internet of Everything
IoT	Internet of Things
IoV	Internet of Vehicles
KNN	K-Nearest Neighbors
KPIs	Key Performance Indicators
LDA	Linear Discriminant Analysis
LightGBM	Light Gradient Boosting Machine
LR	Logistic Regression
LSTM	Long Short Term Memory
M-TADS	Multi-Trust DoS Attack Detection System
MIMO	Multi-Input Multi-Output
ML	Machine Learning
MLP	Multi-Layer Perceptron
NB	Naive Bayes

NFV	Network Function Virtualization
NLP	Natural Language Processing
NN	Neural Network
PCA	Principal Component Analysis
QoE	Quality of Experience
QoS	Quality of Services
RAM	Random Access Memory
REDNN	Resource-Efficient Deep Neural Network
REFDL	Resource Efficient Federated Deep Learning
RF	Random Forest
RNN	Recurrent Neural Networks
SDN	Software Defined Networking
SL	Split Learning
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TPU	Tensor Processing Unit
VAE	Variational Autoencoder
VPN	Virtual Private Network

1 Introduction

Recently, we have been witnessing a revolution in multiple fields due to the development and application of Artificial Intelligence (AI) and, in particular, of Machine Learning (ML). Its extensive adoption in scientific and technological fields has paved the way for more data-driven decision-making models, like speech recognition, natural language processing (NLP), computer graphics, computer vision, and intelligent control. Additionally, it is also being utilized to solve network-related problems such as traffic engineering, routing, security, and resource allocation [1]. The Sixth Generation (6G) networks are expected to increase connectivity beyond 5G to the three-dimensional coverage of land, sea, and space by integrating novel technologies from the access to core networks and non-terrestrial networks [2]. Due to this, 6G networks will integrate a huge variety of devices, systems, and services [3]. Most of those devices, systems, and services will emerge with unique requirements. Moreover, 6G is expected to grow on almost all key performance indicators (KPIs) from 5G. Some of these KPIs will require a radical shift from the existing network architecture. For example, the existing 5G network is largely centralized, where most control network functions reside in the core network. Even though some of the core network functions can be pushed to the edge of the network, most functions still remain centralized. However, to meet the latency requirements of future services, most core network functions, such as the authentication function, will be pushed to the far edge of the network. The resulting network will be an extremely complex environment compared to the previous predecessors. ML, thus, will play a significant role in almost all aspects of the management and operations of 6G networks [4]. Therefore, the resulting 6G networks will be highly complicated environments needing AI and ML-based solutions for automated or semi-automated network-related decision-making, converting those decisions to configurations, and then deployment. AI and ML, thus, will be instrumental in not only enhancing but also in the very functioning of 6G networks and their technologies.

Similarly, the security of 6G communications networks will also become increasingly challenging due to the overall complex nature of networks [5]. To meet the needs of various applications, ML algorithms are progressing towards larger and computationally complex models, which require more computational resources and energy [6], [7]. Previously, while developing the ML algorithms, resource consumption did not get enough attention since most research focused on improving model accuracy. But recently, resource consumption of ML models has become a major research field for researchers since it has a significant impact on both economic and environmental costs [8]. Furthermore, the deployment of ML in communications networks has its own challenges [9]. When it comes to the implementation of security of communications networks, resource efficiency of ML solutions will be crucial. The main reason is that in all types of communication network transmission, resources like power and bandwidth are limited [10]. When an ML-based security solution is integrated into communications networks, resources are also consumed for ML and security operations. Therefore, researchers have increasingly given considerable attention to resource efficiency while developing ML models for various tasks. Moreover, there may be instances where the accuracy of the model may be sacrificed in order to optimize resource consumption.

1.1 Problem Statement

Native AI has been proposed for 6G. Native AI will require dedicating computing, memory, and energy resources to the 6G network infrastructure. To put facts in perspective, training large AI models requires a huge amount of resources. For example, training a large language model can consume hundreds of megawatts of hours of electricity, memory of several terabytes, and hundreds of powerful GPUs. Even though the consumption of resources of ML or AI can be smaller due to smaller tasks in 6G networks, the net impact can be much higher since 6G networks are considered more of an eco-system than a simple packet-forwarding infrastructure. For example, 6G will provide differentiated Quality of Service (QoS) for different verticals and services, such as digital healthcare [11] and vehicular networks [12].

Securing future networks will be extremely difficult because of the integration of the large array of mobile services that accompany users. The challenge is also due to the exponential growth in the number of resource-constraint devices like the Internet of Things (IoT) and the consolidation of various operators who may have differing and potentially conflicting security policies [5]. The security recommendation provided by the International Telecommunication Union-Telecommunication Standardization Sector (ITU-T) covers security from eight dimensions, they are authentication, access control, data confidentiality, non-repudiation, data integrity, communication security, privacy, and availability. Each of these aspects has its own implementation methods and resource costs. Achieving green security with them would need specific modifications to maintain the same level of security while reducing resource consumption. Although ML-based security solutions are potential candidates for meeting the network security challenges in many aspects of 6G, their resource consumption will be a key concern, and extensive research is needed to maintain the resource consumption of ML [3]. Hence, to establish a robust and reliable network security system for the 6G network, it is essential to ensure proper resource utilization. This will not only foster innovation but also promote sustainability in the coming years.

Therefore, the work in this thesis investigates the resource consumption of ML algorithms in network security. Additionally, different resource-efficient ML-based network security solutions are studied to find out the key enablers of the resource efficiency of those solutions. Finally, resource consumption measurement experiments of an ML-based network security solution are performed to provide important insights into the resource consumption of ML algorithms in network security solutions to study the impact on important resources through ML in future 6G networks.

1.2 Research Questions

In this thesis, our main goal is to answer three research questions, they are:

- **RQ1:** What are the key resources that are consumed by ML techniques used in network security?
- **RQ2:** How resource efficiency in network security is achieved in different ML techniques? and
- **RQ3:** How much resources are consumed by distributed learning techniques like Federated Learning (FL) and Split Learning (SL) and how do they perform in a network security test case?

1.3 Research Objectives

Given the unique characteristics of 6G, such as ultra-wide bandwidth, minimal delay, and a decentralized and intelligent network, ML is poised to play a crucial role in its operation. Therefore, this thesis aims to investigate the resource usage of ML in the context of 6G, with the goal of promoting sustainable practices and efficient utilization of resources. Specifically, the objectives of the thesis are:

1. Explore ML-based network security solutions to define Key Performance Indicators (KPIs) for resource usage
2. Identify key enablers of resource efficiency in ML-based security solutions
3. Evaluate resource consumption of deploying distributed ML techniques in 6G security

By doing so, this work contributes to the ongoing discussions and preparations for the advent of 6G and its implications for ML and network security.

1.4 Thesis Organization

The rest of the thesis is organized as follows:

- **Chapter 2** Describes the basic 5G and 6G concepts, their security challenges, basic machine learning concepts, their application in communication networks, and their challenges in integrating with 6G security.
- **Chapter 3** describes the resource consumption of ML in network security. Also, this chapter discusses the Key Performance Indicators (KPIs) for the resource consumption and efficiency of ML algorithms.
- **Chapter 4** describes different resource-efficient state-of-the-art ML-based network security solutions and how the resource efficiency was achieved there.
- **Chapter 5** describes the necessary experimental setup, use case, data preprocessing, model descriptions, model training, and other necessary steps for measuring the resource consumption fingerprint of the two most popular distributed learning techniques, FL and SL.

- **Chapter 6** discusses the findings of the experiment in detail, and provides recommendations for the application of FL and SL in various 6G use case scenarios.
- **Chapter 7** concludes the thesis with a discussion of the research questions and discuss of the future research direction.

2 Literature Review

This Chapter discussed some key background information, including an overview of the concepts of 5G and 6G, the requirements for 6G, the basic concept of ML and different ML approaches, the application of ML in network security, the importance of ML in 6G, and the challenges for ML in a 6G network system.

2.1 Introduction to 5G and the Need for 6G Network

Fifth-Generation Mobile Technology (5G) offers enhanced quality, high-speed data transfer, extensive coverage, minimal delay, high-dependability and cost-effective services. The services delivered by 5G can be divided into three types. Extreme Mobile Broadband (eMBB) is one of them, which enables high-speed internet access, moderate delay, increased bandwidth, augmented and virtual reality, and ultraHD video streaming. Another is Massive Machine-Type Communication (mMTC), designed for broadband and long-range machine communication. This type is cost-effective, has low power consumption, and provides high data rates and extended coverage, making it suitable for IoT applications. The final type is Ultra-Reliable Low Latency Communication (URLLC), known for its low latency, superior service quality, and ultra-high reliability. It facilitates services such as remote surgery, intelligent transport, smart grid, etc. [13].

The advent of the 5G wireless network introduced novel technological ideas to the wireless field, bridging the gap between communications networks and the conventional IT realm [14]. This includes the transformation and virtualization of networking technologies, which facilitated the launching of new wireless network applications and use cases. From the Massive Multi-Input Multi-Output (MIMO) technology at the physical layer to the enhancement of the application layer with ML capabilities, the network's functionalities have been significantly expanded. However, despite these advances, 5G inherited some limitations that prevent it from fulfilling the requirement of new services such as the Internet of Everything (IoE) [15].

The 6G communication network is expected to be introduced in 2030 and will make a significant jump beyond the current 5G technology. The aim will be to cater to the future needs of societies and services. That will revolve around intelligent, data-driven, and automated processes [16]. Presently, terrestrial communication is a fundamental aspect of 5G networks. However, in 6G, this dimension is expected to expand from terrestrial to underwater and aerial communication with several times greater capacity than the 5G [17]. This increased connectivity will pave the way for numerous advantages, for instance flying cars, underwater leisure activities, holographic telepresence, etc. [18]. Innovative and groundbreaking technologies like terahertz and optical communications, distributed AI based on end-user terminals, seamless coverage through combined terrestrial-satellite access technologies [2], and Distributed Ledger Technologies (DLTs) will be some of the key enablers for meeting the requirement of the new use cases and applications [19].

6G is anticipated to trigger a significant transformation in human interaction facilitated by enhanced context-aware devices, with new interfaces between humans and machines. However, the end devices that will provide these interfaces will evolve beyond simple data collectors to multiple synchronized entities operating together. Moreover, emerging technology like holographic data launching to millions of users simultaneously in real-time with extremely low latency will require data rates ranging from several gigabits per second up to terabits per second [18]. Hence, these services will require a strict Quality of Services (QoS), including latency, reliability, and bandwidth, which will pose a challenge for existing 5G networks [15]. Hence, with the emergence of sophisticated technologies, the 6G network is envisioned to transform into a more potent and efficient system than 5G, which will cater to the need for the existing services and pave the way for introducing groundbreaking services that have never been witnessed before [20].

2.2 Security Challenges in 5G and 6G

To meet the new technological demand for connected devices and diverse applications, a few concepts were introduced in 5G, such as Software Defined Networking (SDN), Cloud Computing, and Network Function Virtualization (NFV). However, these technologies also have their own security issues. For example, there are two entities in the cloud known as Mobility Management Entity (MME) and Home Subscriber Server (HSS). They store user's mobility handling, personal, and billing information accordingly. If any breach occurs in this area, the whole network will be ineffective. Moreover, in the SDN controllers, the network control logic is centralized, as a result, these controllers are the attractive target for the attackers for rendering the whole network down with resource exhaustion or Denial of Service (DoS) attacks. This challenge also applies to NFV hypervisors [5].

5G resource allocation optimization/orchestration utilizes AI/ML which provides built-in orchestration flexibility. However, this flexibility can lead to vulnerability and open the path for attackers to manipulate the configuration. Moreover, even though network slicing makes the 5G network more efficient in resource sharing and aids in allocating resources for different applications, it also involves security concerns. Additionally, since the edge is extended to end users, the attack surface increases as a result, and it becomes a favourite choice for cyber attacks. There also exists the complication related to the edge host security controls. Furthermore, the open-source activities in SDN/NFV also raise security concerns because there is a lack of documentation, level of support, intellectual property concerns, and more. Finally, there is also concern regarding to data security and privacy resulting from data centralization, and supply chain security from the utilization of commodity modular software and hardware [21].

The security threat landscape for 6G will be much wider since the network will be part of a system that includes space, air, and ground networks. The system will inherit vulnerabilities from 5G [5] and will obtain new threat vectors from newly integrated technologies from the infrastructure layer (e.g., radio technologies) up to the application layer (e.g., new verticals). For instance, the positioning of radio bands in extremely large MIMO systems operating at terahertz frequencies can be exposed to eavesdropping, jamming, and pilot contamination attacks [22]. Moreover, the openness of the network architecture, for example, Open Radio Access Networks (O-RAN), can significantly expand the attack surface in 6G [23].

The rise in connected devices and extreme use of open-source solutions will also significantly expand the attack surface in 6G. Additionally, since the security of a network system is dependent on its entities and interfaces, converging to the weakest point as these increase. Devices with poor security, especially IoT devices, will increase system vulnerability. Interfaces such as Network Exposure Function (NEF) or Mobile Edge Computing (MEC) will be more open to third parties, potentially leading to privacy invasion due to the failure to implement proper authentication and authorization measures [24].

The introduction of Further Enhanced Mobile Broadband (FeMBB) and its extreme data rates present challenges regarding traffic processing for security. Such challenges can be mitigated with distributed security solutions allowing for local and real-time traffic processing across various network sections, from the edge to the fundamental cloud services [25]. However, distributed solutions will have challenges regarding resource limitations, synchronization of different security approaches, and the extreme entanglement of core-network (mainly control) functions that can lead to signalling overhead. ML can provide a way forward in such cases.

2.3 Machine Learning

ML is a specialized field of AI focused on studying and developing statistical algorithms. These algorithms can learn from existing data and apply that knowledge to unseen or new data. This is how they can perform tasks without any explicit instruction. The term machine learning was coined by Frank Rosenblatt, a psychologist from Cornell University. He developed a machine that mimicked the operations of the human nervous system and was capable of identifying alphabetic characters in 1957 [26]. The machine is known as a **perceptron**, subsequently served as the blueprint for contemporary Artificial and Neural Networks (ANN). While the model's learning was close to the human and animal learning models [27]. In 1960, the multi-layer concept opened up the path for feed-forward neural networks and back-propagation. Later in 1967, Marcello Pelillo introduced the idea of the Nearest Neighbor Algorithm, marking the onset of pattern recognition. During the late 1970s and early 1980s, ML and AI went on a separate path. AI researchers abandoned research on neural networks and focused on knowledge-based logical approaches instead of algorithms, but up until that point, ML was utilized as a training program for AI.

ML types vary based on various aspects like input and output data type, approaches, and the problem in hand to solve. Additionally, different views and applications also lead to different classifications. Hence, it is not possible to refer to a completely acceptable taxonomy from the literature. A variety of frequently used ML approaches are discussed below.

2.3.1 Supervised Machine Learning

A **labelled dataset** is used to train the algorithm in supervised learning. A labelled dataset has both input and output parameters, and the model learns to map points between input and correct output [28]. Models developed with supervised learning can be classified using classification and regression techniques.

- **Classification:** The classification technique deals with categorical variable perdition that represents different labels or classes. It predicts distinct outcomes and segregates input data into various groups [28]. For instance, if a traffic flow is normal or attack flow, if an email is spam or not spam. Some of the common supervised ML algorithms are Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest (RF), Naive Bayes (NB), Logistic Regression (LR), and Decision Tree (DT) [29].

- **Regression:** The regression technique predicts a continuous response or target variables. It is applicable if the nature of the expected outcome is a real number or the data have a specific range. For example, predicting the sales of a product or stock market forecasting based on historical data. Some of the most common regression techniques are Lasso Regression, Ridge Regression, Polynomial Regression, Linear Regression, DT, and RF [30].

2.3.2 Unsupervised Machine Learning

In unsupervised Learning, the algorithm is trained with a **unlabeled dataset** to realize the inherent relationships or hidden patterns in data and react accordingly. When new data is introduced, the algorithm reacts with those data based on the presence or absence of those characteristics. In other words, unsupervised learning is used for data that includes only input attributes but lacks any corresponding output [31]. There are three main types of unsupervised learning. They are Clustering, Association, and Dimensionality Reduction.

- **Clustering:** It involves organizing data points into clusters based on their likeness. This is a suitable method to determine trends and connections in data without pre-labeled instances [32]. Some of the very common clustering algorithms are Hierarchical Clustering, K-means and k-medoid, Fuzzy C-means Clustering, Gaussian Mixture Model, Hidden Markov Models, and Subtractive Clustering.

- **Association:** It is the technique used to uncover the links between elements in a dataset. It establishes the rules suggesting that the occurrence of one item implies the occurrence of another item with a certain likelihood [33]. This technique is useful for market basket analysis helping companies to comprehend different product relationships and understand the customer behavior for cross-sell recommendation. Some of the very frequently used algoríthms for generating association rules are Eclat, Apriori, and FP-Growth.

- **Dimensionality reduction:** This technique is utilized when a dataset has an excessive number of features or dimensions. It lessens the dataset to a controllable size while striving to maintain the dataset's integrity as much as possible [34]. It is a very commonly used technique in dataset pre-processing. Some of the very common dimensionality reduction techniques are Autoencoders, Principal Component Analysis (PCA), and Singular Value Decomposition.

2.3.3 Semi-Supervised Machine Learning

This technique combines both supervised and unsupervised techniques by using both labelled and unlabeled data. It is applicable when relatively lower labelled data is available, and relatively higher unlabeled data are provided. It is a very suitable option when getting labelled data is difficult, resource-intensive, costly, and time-consuming [35]. There are various kinds of semi-supervised learning techniques, such as Label propagation, Self-training, Co-training, Generative Adversarial Networks (GANs), and Graph-based Semi-supervised Learning.

- **Self-training:** In this method, any supervised technique is used to train the method for classification or regression and deploy it in a semi-supervised manner. In other words, the model is trained with labelled data, and then the model is used to predict the label of unlabeled data.
- **Co-training:** Co-training is derived from and an improved version of self-training. It is utilized while a small portion of labelled data is available. Here two individual classifiers are trained with two different views of data. The views are features, and each view is independent of the class variable. Each view is sufficient to accurately predict the class of the sample data [36].
- **Graph-based Semi-supervised Learning** In this method, a graph is utilized to illustrate the connection between the data points. The graph is subsequently utilized to distribute labels from the data points that are labeled to those that are not. This method can efficiently encapsulate the inherent physical characteristics of intricate social interactions in real-world scenarios [37].

2.3.4 Reinforcement Learning

This method is a combination of ML and optimal control, which focuses on how intelligent agents can make decisions in a changing environment by maximizing the rewards they receive over time. It generally consists of a relationship between three elements, an agent, environment, and a goal. This relationship is often expressed with the concept of a Markov decision process (MDP) [38]. Here the RL agent interacts with an environment and learns about a problem. The RL agent receives the data about the current state of the environment and makes a decision based on the provided information. The RL agent receives the reward or punishment from the surrounding environment. Which encourages or discourages the RL agent from making the same decision in a similar state in the future. The process is repeated for every new state and hence RL agents learn to take actions within the environment to satisfy a specific goal. Some of the most common RL approaches are Q-learning, SARSA, and Deep Q-learning.

2.4 Centralized and Distributed ML

Due to the ever-evolving nature of data and devices, ML also needs to evolve. Training traditional ML algorithms requires centralizing data to a central server. However, there exists some laws and regulations that restrict data from being aggregated or shared directly to different regions [39]. Hence, distributed learning is utilized to address this issue. In this section, a brief discussion about Centralized Learning and Distributed Learning is presented.

2.4.1 Centralized Learning

Centralized machine learning is the traditional way of machine learning, where data from different sources are collected or gathered on a central server. The central server can act as a data warehouse, data lake, or lake warehouse. The data here is then processed and trained in the ML model in the central server. Training this type of ML requires an extensive amount of processing and memory resources. This trained model can then be used in the server or other nodes for operation. However, the models in the other nodes do not require a similar amount of memory and processing power as the server because they only perform inference. The advantages of this method are efficiency, simplicity, affordability, and consistency. However, the limitations of centralized learning are privacy issues, scalability, robustness, and adaptability [40].

2.4.2 Distributed Learning

The development of distributed machine learning has been geared towards managing extensive learning processes on massive data sets within the context of big data and distributed computing systems [40]. It allows training the ML model without requiring the data to be transmitted to a centralized server. This is relevant to future requirements of 6G as significant growth in data, such as from IoT devices, edge, and distributed sources, will make it challenging to centralize and process all the data on a single server. Also, with a growing need for data privacy, sensitive data must be confined to devices or within certain geographical regions. In such scenarios, distributed learning enables training models without centralizing all the data and addressing privacy and data sovereignty concerns. From a communication point of view, distributed learning results in lower overhead as the model parameters need to be shared without requiring the transmission of large amounts of data, therefore minimizing the use of bandwidth. For certain applications requiring low latency, such as autonomous vehicles, distributed learning can enable faster response times by training and processing the data closer to the sources.

2.5 Applications of ML in Communication Networks

ML is renowned for its contributions to scientific disciplines such as computer vision, NLP, and speech recognition and is now expanding its reach to numerous other areas. It has been attracting renewed interest in the networking domain. The data-driven nature of ML allows it to automatically understand the complexities of communication and networking environments and dynamically adjust protocols without the need for human intervention [41]. In communication networks, it is being used for various purposes. Fig. 2.1 shows an AI/ML-based network infrastructure for IoT applications. Some of the very common applications of ML are listed below:

- ***Information Cognition:*** Due to the complex nature of networks and the limitations of measurement tools and architecture, some types of data are difficult to access within an acceptable cost and granularity, even though data is the fundamental asset for ML networking. However, ML can evaluate a certain network state with its predictive ability [42].

- ***Traffic Prediction and Classification:*** Accurate traffic prediction is crucial for network routing, resource allocation, congestion control, and high-level streaming applications. Traditional traffic classification methods, like port-based and payload-based approaches, have limitations such as unfixed or reused port assignments and privacy issues, respectively. Therefore, ML-based statistical features have been extensively studied to address these gaps [42].

- ***Intelligent Routing:*** Routing is the process that determines the path from source to destination of any data packet. The performance of a network substantially depends on the efficiency of the routing algorithm [43]. There are many routing algorithms like Distance Vector, heuristic algorithms, and Shortest Path First (SPF). However, these traditional algorithms have some disadvantages, like high computational complexity, slow convergence speed, and long training time. Inefficient routing decisions can cause network congestion and packet loss because of the transmission delay, so dealing with the growing traffic and meeting the high user demand may become difficult[44]. Machine Learning can address this issue by enabling intelligent routing. Different ML-based solutions for intelligent routing are available in the literature such as, [45], [46], and [47].

- ***Resource Management and Network Adaptation:*** Improving network performance is key and relies heavily on network adaptation and resource management. However, challenges like Transmission Control Protocol (TCP) congestion control, routing, and traffic scheduling exist. Heuristic algorithms may not be sufficient due to the complexity of diverse system environments, noisy inputs, and optimization difficulties of tail performance. Deep learning (DL) can characterize the relationship between the input and output of a network system without human involvement, making it a promising solution for these issues. Various Deep-learning solutions for these issues are available in the literature [42].

- ***Network Performance Prediction and Configurations Extrapolation:*** The ability to predict performance is vital for decision-making in a variety of applications. This includes predicting the Quality of Experience (QoE) for videos, determining the location of Content Delivery Networks (CDN), selecting the best wireless channels, and extrapolating performance under different configurations. ML is an intuitive method for predicting system states, aiding in improved decision-making [42].

- ***Congestion Control:*** Network congestion is a significant issue for service providers as it negatively impacts the overall performance of the network. Congestion control maintains network stability, ensures fair use of resources, and keeps the packet loss ratio at an acceptable level [48]. With the massive development of widely used network technology, like 5G, Wi-Fi, data centers, and satellites, the diversity and complexity of the network transmission protocols and scenarios have increased significantly. Different types of congestion control algorithms are used in different scenarios, but finding a generic algorithm is difficult because of the variety of network scenarios and their intrinsic dynamics. But ML can provide a generic congestion control algorithm that supports different network scenarios [49].

- ***Fault management:*** It's unrealistic to expect network administrators or operators to have comprehensive knowledge about the entire network, including all applications and devices connected to it. Modern networks have become increasingly advanced and characterized by high levels of complexity and dynamism, which makes fault management even more challenging. However, ML can provide significant assistance in this area, aiding in the prediction, detection, and localization of faults, as well as in the mitigation of these issues [50].

- ***Developing Intelligent Architectures:*** AI/ML is also gaining popularity in developing intelligent architecture, which is necessary to meet the growing demands of modern networks and systems. Heilin et al. [51] proposed an AI-enabled intelligent Architecture for a 6G network to support diverse services smartly, guaranteeing reliable connectivity, and optimizing network performance.

- ***Edge Intelligence***

  The huge amount of data which are generated by edge devices is difficult to send for processing to the cloud due to the various quality of the channel, enormous energy consumption, privacy concerns, and traffic congestion. Hence, edge AI has become a potential solution by pushing the training and inference process at the edge to deal with this issue. However, there are some challenges, too, since the AI needs close cooperation between edge devices (smartphones, vehicles, base stations, etc.), which leads to a severe communication overhead. Shi. et. al [52] conducted a comprehensive survey on the recent development techniques which were done to overcome those communication challenges.

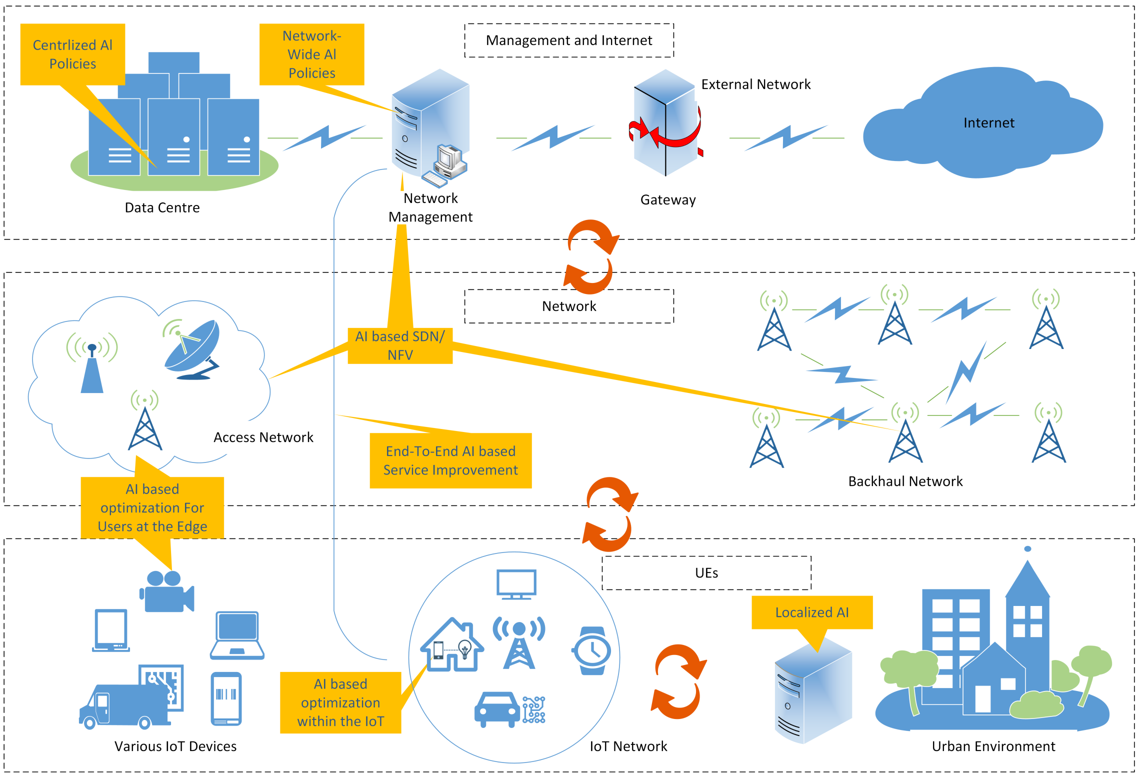


Figure 2.1: AI/ML Operations in AI-based Network Infrastructure.

2.6 Applications of ML in Network Security

The use of ML in network security is currently receiving significant attention in academic circles. The need for ML in various aspects of network security is becoming increasingly essential, given that a wide array of next-generation network services require automated decision-making processes for tasks such as security policy verification, conversion, deployment, and configuration. A crucial requirement for these services is the proactive implementation of security measures, such as access control and authentication, within specific time constraints. This is necessary to fulfil the primary service requirements. The authors in [4] discuss how ML is contributing to proactive security measures in communication networks and earning recognition by mitigating security risks and lapses. For example, ML is used in CAPTCHA to distinguish between humans and bots, in traffic classification to assist in intrusion detection and deep packet inspection, and so on. However, the authors also highlight some challenges, such as various attacks on ML itself, the lack of research focus on ML security, and the potential for ML to be leveraged to deploy and devise malicious attacks. Some other common applications of ML in network security are:

- Distributed denial-of-service (DDoS) Detection
- Network segmentation
- Intrusion Detection System (IDS)
- Anomaly Detection
- Botnet Detection
- Wireless security
- Security Traffic Management
- Mobile Malware Detection
- Distributed Domain Name System (DNS) Attack Detection.
- False data injection detection
- Web and Virtual Private Network (VPN) security

2.7 ML for Security in 6G

AI and ML will be instrumental in enhancing networks and developing novel waveforms of 6G. Additionally, the advancement in AI and ML will be propelled by 6G technology, which will leverage locally stored data on 6G sensors [53]. The fully automated networks hinge on preemptively identifying threats, implementing smart countermeasures, and ensuring that 6G networks are self-reliant. This section discusses the challenges for AI/ML in 6G security.

2.8 Challenges for ML in 6G Security

The driving force of the 6G will be the intrinsic intelligent connectivity within communication networks with enhanced AI and sophisticated networking technology [54]. The ML techniques, however, have several inherent challenges. Most ML techniques are highly centralized. Therefore, Large-scale deployment of the traditional ML and DL models is challenging in the context of 6G. This is because, in the traditional centralized ML, where a massive amount of data is collected and processed in a centralized server, resulting in performance bottlenecks and increased processing time. Moreover, data centralization leads to communication overhead, especially when dealing with geographically distributed data. Furthermore, centralized systems suffer from a single point of failure, leading to availability and privacy issues [55].

Additionally, though DL has demonstrated high efficiency in threat detection, context-aware security protection will be a requirement for future communication services with diversified data generated from various devices. The rapid growth of the IoT and the Internet of Vehicles (IoV) will also lead to several security challenges in 6G. Setting up appropriate security measures will be a complex issue, as implementing security measures often leads to a trade-off with network QoS [56].

Additionally, AI/ML is prone to various security risks. The most common attacks on AI/ML are evasion attacks, poisoning attacks, compromise of AI frameworks, ML API-based attacks, infrastructure physical attacks, and DoS attacks. DLTs have the potential to protect the integrity of AI/ML. However, the DLT is also subject to the eclipse attack and has software and end-user vulnerability [25].

Furthermore, in 6G, ML will also be responsible for processing and harvesting vast quantities of clients' private data, which will raise privacy concerns. Moreover, the ML is often criticized for lacking fairness and transparency. The substantial volume of training data can make it a prime target for attacks, and the inherent vulnerabilities of ML can make it susceptible to such attacks [57]. Moreover, distributed AI will be one of the key areas of 6G; hence, it needs to be IoE-compliant. It will make the 6G a sizable decentralized system that can make intelligent decisions at different levels. In that scenario, distributed ML is expected to be deployed extensively. However, there are some challenges related to distributed learning. For instance, some malicious users can upload poisonous models to train the entire model, and there are also some device-level security issues [58]. Furthermore, the 6G will have real-time edge intelligence, where the computation will be brought close to the data source. 6G is touted to possess the substantial capability to deliver seamless services to billions of smart devices, and the Edge node will be one of the network nodes in 6G. However, AI/ML algorithms are computationally intensive and consume a significant amount of power and computation resources, which is lacking in the edge node [6]. Hence, deploying resource-efficient security solutions will be a requirement for 6G while deploying at the edge. Finally, since AI/ML can be the indirect source of carbon footprint [7], and if they are the vital elements of 6G, deploying a Green ML-based Security System will be another crucial challenge in 6G.

3 Resource Consumption of ML in Network Security

This Chapter elaborates on the resource consumption by ML techniques in communications network security. It was mentioned in the previous chapter that, although ML is a promising technology for future network security, its extensive resource consumption is a significant challenge. Hence, to overcome this challenge extensive research and study is needed to maintain resource usage in ML in network security. Hence, this thesis attempted to find out the Key resources that are consumed by ML techniques. This chapter describes the process of finding those resources. It also provides a brief description of the resources that are consumed by ML techniques.

3.1 Key Performance Indicators (KPIs)

In this section, the thesis attempts to answer the first research question, **RQ1:** What are the key resources consumed by ML techniques used in network security? To address this question, an extensive literature review was conducted. Popular databases and search engines, including Google Scholar, Web of Science, Scopus, and Bing, were used to search for articles related to the theme of the work. The focus was on literature published by IEEE, ACM, Springer, and Elsevier, as these publishers have a strong reputation in academia and offer a portfolio of high-quality journals. Initially, the work focused on the importance of ML in communication networks to understand the basic principles of ML applications in these networks. It was found that ML has been extensively used across all layers of communication networks, from the physical layer up to the application layer. Subsequently, thorough research was conducted on the application of ML in network security. The application of ML in network security began much earlier than in other parts of communication networks [4]. For instance, the first application of ML was CAPTCHA.

Given the focus of the work, further research was conducted on the resources required for efficiently running ML in communication networks. In particular, searches were conducted to find the key resources considered in most of the state-of-the-art ML-based network security solutions. This thesis considers these resources the KPIs for resource-efficient ML-based network security solutions. For this purpose, articles on the state of the art were selected based on the following criteria:

- If the article is about a resource-efficient ML-based network security solution.
- Or, if the article discussed an ML-based network security solution that consumes less amount of resources.
- Or, if the article discussed any resource that an ML-based network security solution can improve.
- Or, if the author conducted a comparison of different ML algorithms where any algorithm was proven any kind of resource efficient.
- Or if the author conducted a comparison of any technique on ML model development that uses fewer resources.
- if the article is published not before 2017.

The literature study found different kinds of resources relevant to network operation and ML techniques deployment. These resources are intended to be improved or considered important by researchers when comparing or developing a network security solution. A list of those resources, along with corresponding network security solutions and the ML algorithm used, was prepared in this work. The list is presented in Table 3.1, where **E**, **P**, **M**, **T**, **B**, and **H** represent Energy/Power, Processing, Bandwidth, Memory, Time, and Human Involvement respectively.

Table 3.1: The Resource-Efficient ML-based Solutions for Network Security Concerning the KPI

Ref	**Security Type**	**ML Algorithm**	**Covered resources**					
			E	**P**	**M**	**T**	**B**	**H**
[59]	DDoS detection	Extreme Learning Machine		✓	✓	✓		
[60]	IDS	DT, RF, XGBoost (XGB) and KNN	✓			✓		
[61]	IDS	SVM and ANN				✓		
[62]	HTTP Flood Attacks prevention	SVM, Multi-layer perceptron (MLP), DT, and RF					✓	
[63]	IoT security monitoring	Resource Efficient Federated Deep Learning (REFDL)			✓	✓		
[64]	Intrusion detection engine	DT, NB, and kNN	✓	✓	✓			
[65]	DoS detection	Multi-Trust DoS Attack Detection System (M-TADS)	✓	✓		✓		

Table 3.1: The Resource-Efficient ML-based Solutions for Network Security Concerning the KPI

Ref	**Security Type**	**ML Algorithm**	**Covered resources**					
			E	**P**	**M**	**T**	**B**	**H**
[66]	Anomaly detection	Long Short Term Memory (LSTM), FL, LR, Gated Recurrent Units (GRU), Feed-Forward Neural Network (FFNN), Variational Autoencoder (VAE), Convolutional Neural Network (CNN), Vanilla Recurrent Neural Networks (RNN)	✓	✓	✓	✓	✓	
[67]	DDoS detection and prevention	RF, Light Gradient Boosting Machine (LightGBM), XGB, and Adaptive Boosting (AdaBoost)	✓	✓	✓	✓		
[68]	Anomaly detection	FL		✓	✓	✓		
[69]	IoT anomaly detection	FL, Deep Reinforcement Learning (DRL)				✓		
[70]	Anomaly detection	FL	✓	✓	✓		✓	
[71]	Anomaly-detection-based IDS	FL, GRU		✓		✓	✓	
[72]	WSN attack detection	SVM, MLP	✓			✓	✓	

Table 3.1: The Resource-Efficient ML-based Solutions for Network Security Concerning the KPI

Ref	**Security Type**	**ML Algorithm**	**Covered resources**					
			E	**P**	**M**	**T**	**B**	**H**
[73]	False data injection detection	Linear Discriminant Analysis (LDA), SVM				✓	✓	
[74]	IDS for 6LoWPAN	RL	✓	✓		✓		
[75]	Artificial jamming	FL (MLP)	✓			✓		
[76]	IoT security monitoring	Resource-Efficient Deep Neural Network (REDNN)			✓	✓		
[77]	IDS	DT, LSTM		✓		✓		
[78]	Anomaly detection	KNN, DT, SVM, NB, RF	✓		✓	✓		
[79]	DoS detection	Custom	✓	✓	✓	✓		
[80]	DoS detection	KNN, DT, SVM				✓		
[81]	DoS detection	lightweight variational Bayes algorithm	✓	✓		✓	✓	
[82]	DoS detection	LSTM		✓		✓	✓	
[83]	DDoS detection	REP Tree, Random Tree, RF, Decision Stump, PART		✓	✓			
[84]	IDS	SVM, RF, KNN			✓	✓		
[85]	DDoS mitigation	Cyclic queuing algorithm	✓	✓		✓		
[86]	DDoS detection	LSTM						✓
[87]	Botnet detection	LR, RF, KNN, SVM, XGB, DNN		✓	✓	✓		
[88]	Anomaly detection	Genetic Algorithm (GA)						✓

Table 3.1: The Resource-Efficient ML-based Solutions for Network Security Concerning the KPI

Ref	**Security Type**	**ML Algorithm**	**Covered resources**					
			E	**P**	**M**	**T**	**B**	**H**
[89]	IDS	KNN, LR, DT, RF, NB and ANN	✓					
[90]	Botnet detection	RL				✓		
[91]	Online phishing email detection	RL, Neural Network (NN)				✓		✓
[92]	Anomaly detection	RL		✓		✓		✓
[93]	IDS	RL				✓		
[94]	Security traffic management	DRL						✓
[95]	Mobile malware detection	Hybrid, SVM, C4.5CS, SVMCS				✓		
[96]	IDS	RF, XGB		✓				
[97]	IDS	NB, C4.5				✓		
[98]	IDS	KNN, SVM, DT				✓		
[99]	Botnet detection	LR						✓
[100]	IDS	SVM, KNN, K-means, SVM2, k-means2, SVM+k				✓		
[101]	IDS	Facebook's prophet	✓	✓		✓	✓	
[102]	IDS	Hybrid Feature Selection (Least Squares and SVM)		✓		✓		

Table 3.1: The Resource-Efficient ML-based Solutions for Network Security Concerning the KPI

Ref	**Security Type**	**ML Algorithm**	**Covered resources**					
			E	**P**	**M**	**T**	**B**	**H**
[103]	IDS	SVM, Mean-shift, ANN, LDA, RF, K-means, density-based clustering algorithms (DB-SCAN), and KNN				✓		
[104]	IDS	DT, RF, SVM, NB, CNN, KNN, ANN				✓		

According to the finding of this research attempt, while developing a resource-efficient security solution, most of the time, the researchers attempt to improve either one or multiple of the following resources: **Energy/Power**, **Processing**, **Bandwidth**, **Memory**, **Time**, and **Human Involvement**. However, Data and Storage are other important resources for an ML to function. According to the authors in [105], to learn the related pattern, ML processes and consumes data. Whereas storage is crucial to store the model after the completion of the training process. For large models, besides memory, a significant amount of storage is also required. For example, the GPT 3, a large and well-known language model, has over 175 billion parameters to store[106]. However, this work did not focus on these two resources. The reason is that in the search for resource-efficient network security solutions in this work, sufficient discussion was not found on improving these two resources.

3.2 Description of Resources Consumed by ML Techniques

The previous section mentioned that power, processing, memory, time, bandwidth, and human supervision are the key resources for any ML-based network security solution. This section briefly describes how and why this resource is required for ML techniques.

- ***Energy/Power:*** Machine learning learns from a large amount of data that requires computational power and energy. Various factors affect the energy consumption of an ML model, such as the size and structure of a model. Large and complex models have more parameters and require more data to train. So, the energy demand is higher for those kinds of models. Moreover, the volume of data is another factor, when there is more data to process, more energy will be required. Additionally, different algorithms have different convergence and stability, which affect energy consumption while optimizing. Furthermore, different optimization techniques also have different impacts on energy consumption [107]. Finally, the hardware platforms such as Central Processing Unit (CPU), Graphics Processing Unit (GPU), specialized chip, and software like TensorFlow, PyTorch, or Scikit-learn, which enables an ML model to run, also require energy. So, energy is a crucial resource for ML.

- ***Processing:*** Processing power is a crucial resource for ML. It may involve the CPU, GPU, Tensor Processing Unit (TPU), disk space for intermediate data or final model storage, memory usage, and others. The requirement for processing power can vary based on application needs and the nature of the algorithms. The CPU performs general-purpose processing, sequential processing, data preparation, and other tasks. It is suitable for training simple models, small effective batch sizes, and models limited by the host system's

I/O or networking bandwidth. Additionally, the CPU is much more effective and cost-efficient for non-deep learning ML. GPU consists of numerous simple processing units compared to CPU and can handle parallel computing and perform rapid mathematical calculations. It is suitable for medium and large models with larger effective batch sizes. Finally, TPU is an AI accelerator particularly designed for TensorFlow and provides extremely fast execution of calculation [108]. Various factors may influence the amount of processing resources required in ML. For example, the size of the training and prediction dataset. Larger datasets consume more computing power. Moreover, different ML algorithms have different complexity and require different levels of processing power. For instance, deep learning requires a great deal of computing resources. Additionally, the use of different frameworks and hardware also has different levels of processing consequences.

- ***Memory:*** Memory is the another crucial resource for ML. Generally, the memory in ML refers to the Random Access Memory (RAM). It temporarily stores the data that a computer processor can access quickly. Based on the requirement, different kinds of RAM are used in ML applications. The requirement for the amount of memory can depend on various factors. To process large datasets, larger memory is needed. Processing large and complex models also requires larger memory, etc. The fast-growing Machine Learning requires stronger support from the memory, like the memory with higher bandwidth, access speed, capacity, and reliability [106].

- ***Time:*** Managing time efficiently is the key to the success of a machine learning model. Time is crucial for the quality and accuracy of a modal, and various steps require time. For instance, data collection, data cleaning, pre-processing, training, and inference. The required training time for a model can vary based on different factors, such as the dataset size, model complexity, and computational resources. Moreover, the model accuracy of the ML model also improves with time because of its iterative nature. Additionally, time is a constraint in some real-world applications, like fraud detection, self-driving cars, and chat-bots. So, time is a vital resource for ML, and its proper utilization is crucial.

- ***Bandwidth:*** In computing, bandwidth or network bandwidth represents the maximum data transfer rate along a specific route [109]. It generally defines the number of bits, kilo-bits, megabits, or gigabits of data that can be sent in a single second. The network has more bandwidth and can send or receive more data. However, bandwidth is a scarce resource for networks, and the cost goes up with the increased utilization of bandwidth. Different communication devices and users compete and share the available bandwidth in a network. When an ML-based solution is deployed into a network, it also uses the network bandwidth. Because ML needs a large volume of data for real-time processing, it can strain high bandwidth.

- ***Human:*** Although ML paves the way towards automation, it is not a completely hands-off process. Humans are needed in every step to guide the key decisions. Human involvement is generally required in Data Collection and labelling, Data Cleaning and Pre-processing, Feature Engineering, Model Selection and Training, Deployment and Monitoring, and Ethics and Fairness. Additionally, humans also control the ML learning process, and some ML algorithms were developed from the inspiration of the human learning process [110].

4 Resource Efficient ML-Based Network Security Solutions

Despite the potential of 6G networks, it will introduce a range of security challenges because of the growing numbers of independence of smart services and their extensive involvement in daily life [111]. Some of the challenges will come from its extensive dimensions, such as the security system of underwater devices, which generally have lower access to resources like power than traditional land-based devices. Hence, lightweight security systems will be preferred for longer operations in such scenarios. Additionally, in traditional 5G, when the battery level of a device is low, it prefers a less complex security configuration to conserve power. Considering those situations, the security protocols in the 6G network should be tailored in a flexible and evolving way for different situations [112]. Although security was hardly researched from the sustainability point of view for the previous generation of wireless network technology, the trend is expected to change for the 6G [3]. Hence, the desired level of resource utilization and the environmental cost with acceptable accuracy will get some serious attention while deploying future network security. Furthermore, ML is expected to play a key role in securing 6G networks, but it also requires resources for operation, as discussed in the previous chapters. Hence, ML must be resource-efficient in developing sustainable security systems for 6G. In this Chapter, the thesis tries to answer the second question **RQ2:** How resource efficiency in network security is achieved in different ML-based solutions? For this purpose, the literature collected in answering **RQ1** in Chapter 3 was studied and examined carefully. Various remarks from those articles were taken, such as:

- Which kind of datasets were used in those solutions?
- Which ML algorithm performed best with that solution?
- How was the performance of the best ML algorithm in that solution?
- What types of validation methods were used?
- Which step influenced achieving resource efficiency for that solution?

After extracting this information, the information was arranged according to the KPIs and corresponding tables.

4.1 Energy/Power

Energy consumption of ML is becoming a major concern not only because of its scarcity but also because of its environmental cost. The energy consumption in ML happens for training, inference, and operating the ML hardware, including the data center overhead. According to [113], the energy consumption of ML can be reduced significantly by selecting efficient ML models, using processors optimized for ML training, and computing in the cloud rather than on-premise. Moreover, in Google, even though the use of ML from 2019-2021 increased significantly, the amount of energy used did not increase that much because of algorithmic and hardware improvements. ML researchers are continuously improving the efficiency of ML models in various fields by innovating model architectures and algorithms. This section discusses various efforts by researchers to reduce the energy consumption of ML-based network security solutions. It also discusses how energy efficiency was achieved in different solutions. Table 4.1 shows the comprehensive analysis of these solutions.

Table 4.1: Methods for Efficient Energy Consumption

Ref	**ML Algorithms***	**Datasets**	**Resource Efficiency Achieved by**	**Validation Techniques**	**Performance**			
					Acc.	**Pr.**	**Re.**	**FS**
[65]	LSTM	-	Offloading Computation, Algorithmic Improvement	Cross-validation	94	93	94	90
[60]	RF	CICIDS2017	Data Aggregation, Distributing Computations	Cross-validation	99.9	99.9	99.9	99.9
[72]	MLP	Custom	Efficient Load Distribution, Feature Extraction, Algorithmic Improvement	-	84-89%	-	-	-
[64]	DT	Custom	Feature extraction, Feature Selection, Hardware Implementation	Stratified Sampling	99.1	-	-	-
[75]	FL (MLP)	MNIST	Optimal Pairing, Algorithmic Improvement	Comparative Analysis	79	-	-	-
[81]	Hybrid	KDD 99	Distributed Processing, Algorithmic improvement	Cross-validation	86.5	94.2	-	-
[89]	NB	Custom	-	Comparative Analysis	97	99	81	89

ML Algorithms*: Best Performing ML Algorithms, Accuracy: Acc., Precision: Pr., Recall: Re, Fi-score: FS.

4.2 Processing/Computation

Efficient Computation/Processing is essential for any ML-based network security solution. If the solution uses too much computation power, it may not be able to process network traffic in real-time, and malicious activities may remain undetected. As a result, the whole system's performance can be degraded and unstable. Hence, efficient processing power utilization is crucial. Table 4.2 presented various processing power efficient network security solutions. It also included different datasets, validation methods, and the way processing power efficiency was achieved in those solutions.

4.3 Memory

Efficient memory utilization is very important for any ML-based solution, and it is also notably a relevant context for network security applications. Since ML models are becoming computationally complex and large, their memory requirements also increase significantly. As a result, it can be a severe challenge to implement them in resource-constrained environments. Hence, the ML models need to be developed to utilize the available memory efficiently and not compromise the system's performance. Table 4.3 shows some of the ML-based memory-efficient network security solutions and how the memory efficiency was achieved in those solutions.

4.4 Bandwidth

Machine learning in network security often comes with a high amount of bandwidth consumption, which can degrade the network performance and increase the operational cost. Few researchers have attempted to develop some network security solutions that can achieve effective security measures with efficient bandwidth uti-

Table 4.2: The Methods for Efficient CPU(Processing Units) Consumption

Ref	ML Algorithms*	Datasets	Resource Efficiency Achieved by	Validation Techniques	Performance			
					Acc.	**Pr.**	**Re.**	**FS**
[67]	RF	Slowloris (custom)	Feature Selection, Algorithmic Improvement	Cross-validation	99.9	99.5	99.9	99.8
[73]	SVM+LDA	Custom	Algorithmic Improvement, Dimensionality Reduction	Cross-validation	99	-	-	-
[74]	RL	Custom	ARL, Feature Extraction, Feature Engineering, Feature Selection	Empirical: Black-box Grey-box	96.3 92.2	96.3 92.2	96.3 92.2	96.3 92.2
[77]	DT	UNSW-NB15	Snort Rule Generation, Feature Extraction	Cross-validation	99.8	99.7	99.9	99.8
[82]	LSTM	CICIDS2017	Feature Selection, Period-wise Detection, Packet-wise Detection	Cross-validation	-	99 99.4	99 100	99 99.7
[85]	Cyclic Queuing	A5M B5M	Algorithmic Improvement, VAE, Cyclic queuing	Holdout	-	-	-	-
[96]	RF XGB	ISCXIDS2012	Feature Engineering, Using a Small Subset of Dataset	ROC AUC	92 99.9	-	-	-
[87]	DNN	Custom	Packet Aggregation, Pre-trained Model Using	-	85	-	-	85

ML Algorithms*: Best Performing ML Algorithms, Accuracy: Acc., Precision: Pr., Recall: Re, Fi-score: FS.

Table 4.3: Methods for Efficient Memory Consumption

Ref	ML Algorithms*	Datasets	Resource Efficiency Achieved by	Validation Techniques	Performance			
					Acc.	Pr.	Re.	FS
[63]	REFDL	MNIST	Pruning, Tuning, Simulated Micro-batching, Algorithmic Improvement	-	95.1	-	-	-
[76]	REDNN	Danmini Doorbell Ecobee Thermostat Ennio Doorbell Philips B120N10 PT-737E PT-838 SNH-1011 XCS-1002 XCS-1003 Kitsune Wustl	Optimization: Micro-batching, Pruning, Parameter Regularization, Weight Averaging	Hold-out	95.1 93.4 88.9 84.1 88.1 92.5 86.1 94.7 97.7 84.1 94.3	-	-	-
[83]	PART	CICIDS2017	Feature Selection	Holdout	99.7	-	-	-
[79]	Custom	-	Algorithmic Improvement	Empirical analysis	100	-	-	-
[84]	RF	Bot-IoT	Feature Selection, Reduction	Cross-validations	99.8	-	-	-
[114]	RF	Custom	Feature Selection	Holdout	93.1	-	-	-
[87]	XGB	Custom	Packet Aggregation, Pre-Trained Model Using	-	85	-	-	85

ML Algorithms*: Best Performing ML Algorithms, Accuracy: Acc., Precision: Pr., Recall: Re, Fi-score: FS.

lization. Table 4.4 shows some of the bandwidth-efficient ML-based network security solutions and how the bandwidth efficiency was achieved in those solutions.

4.5 Human Supervision

Human supervision is needed in various ML steps until it can make decisions. Even after the deployment of an ML model, proper human monitoring is crucial to ensure the model is performing as accurately as it is supposed to. The extensive expense of employing skilled ML engineers in tech industries is also a significant hurdle [115]. However, some of the tasks that require human involvement are nowadays replaced by ML-based solutions to some degree, such as data collection and labeling, data cleaning and pre-processing, feature engineering, etc. Table 4.5 presents some of the ML-based network security solutions where human involvement was reduced.

4.6 Time

Time is crucial for time-sensitive applications such as network security. For instance, the models used in IDS should have relatively short training time because the model needs to be retrained to classify new types of attacks or to adapt to the changing nature of previously seen attacks [61]. Additionally, it also needs to have less inference time for real-time detection and better user experience. Consequently, researchers have put forth significant effort, as documented in the literature, to reduce the training and execution time of ML-based network security solutions. Table 4.6 presented some time-efficient ML-based network security solutions and how time efficiency was achieved in those solutions.

Table 4.4: The Methods for Efficient Bandwidth Consumption

Ref	**ML Algorithms***	**Datasets**	**Resource Efficiency Achieved by**	**Validation Techniques**	**Performance**			
					Acc.	**Pr.**	**Re.**	**FS**
[62]	RF	CIC DoS	Implementation of ML with SDN	cross-validation	96+	-	-	92+
[68]	FL	Gas Pipeline	Distributed Learning, AE, Feature Selection, Fourier Transform, Algorithmic Improvement	Cross-validation	-	96.8	100	98.4
		SWaT				93.9	97.8	95.8
		HAI				89.4	100	94.4
		Power Demand				92.9	94.4	93.6
		ECGs				98.6	100	99.3
		Respiration				95.8	97.3	96.5
		Gesture				93.3	99.1	96.1
		Space shuttle				97.4	100	98.7
		NYC taxi				94.2	100	97
[70]	FL	Space shuttle	VAE and Decoder, LSTM	-	-	100	100	100
		Respiration				93.1	55.3	69.4
		Gesture				52.7	100	69.1
		Nyc taxi				96.1	100	97.9
		ECG				100	100	100
		Power demand				73.6	91.0	81.4
		SCADA				96.1	99.8	97.9

ML Algorithms*: Best Performing ML Algorithms, Accuracy: Acc., Precision: Pr., Recall: Re, Fi-score: FS.

Table 4.5: The Methods for Human Involvement Reduction

Ref	**ML Algorithms***	**Datasets**	**Resource Efficiency Achieved by**	**Validation Techniques**	**Performance**			
					Acc.	**Pr.**	**Re.**	**FS**
[86]	LSTM	Smart Home 1	Auto Hyperparameter Tuning, Active Learning Techniques	Cross-validation	-	97	89.7	92.9
[88]	Genetic Algorithm	-	Unsupervised Learning	Evaluation Metrics	96.5	95.2	76.5	84.8
[92]	RL	Custom	Auto Parameter Adjustment	-	96.5	95.2	76.5	84.8
[93]	RL	NSL KDD UNSW-NB15 AWID	Algorithmic Improvement, Dynamic Evolving of NN	hold out	98.6	98.2	99.07	98.6
[91]	RL	Custom	Dynamic Evolving of NN	Cross-validation	98.6	98.21	99.1	98.6

ML Algorithms*: Best Performing ML Algorithms, Accuracy: Acc., Precision: Pr., Recall: Re, Fi-score: FS.

Table 4.6: The Methods for Efficient Time Consumption

Ref	**ML Algorithms***	**Datasets**	**Resource Efficiency Achieved by**	**Validation Techniques**	**Performance**			
					Acc.	**Pr.**	**Re.**	**FS**
[69]	FL(DRL)	-	Decentralized Training	Stratified Sampling		-	-	
[61]	ANN	UNSW-NB-15	Feature Engineering	Holdout Method	92	93	92	92
[66]	FL (LR)	KDD 99 NSL-KDD CIDDS-001	Distributed Learning	Holdout Method	96 92 91	61 64 70	93 95 91	-
[71]	FL, GRU	Custom	Device-Type-Specific Anomaly Detection, Feature Selection	Cross-validation	95.4	-	-	-
[80]	DT	Custom	Feature selection, Hyper-parameters Fine Tuning, Label Encoder, Heuristic Rules	Holdout	100	100	100	100
[116]	DT MLP	UNSW-NB15	Feature Selection Feature Extraction	-	-	80.18 79.90	76.6 71.4	78.4 75.4
[90]	RL	-	Network Traffic Reduction, Feature Extraction, Feature Reduction	Cross-validation	98.3	-	-	98.9
[100]	SVM2 k-means2 SVM+k	NSL-KDD	Multi-level Tweak, Feature Selection, Normalization	Holdout	97.9 97.2 93.9	97 92 95	99 94 98	98 96 96
[101]	Facebook Prophet	CSECICIDS2018	Feature Selection	Cross-validation	98.3	-	-	98.9
[102]	Hybrid	-	Feature Selection, Algorithmic Improvement	Cross-validation	98.3	-	-	98.9
[103]	RF	KDD99 NSL-KDD UNSW-NB15 CIC-IDS-2017	-	Holdout	99.9 99.9 97.4 99.8	-	-	-
[117]	NB	KDD 99	Feature Selection: Dense FR Feature Selection: Sparse FR	Cross-validation	94.6 94.9	94 94.8	98.2 98.2	96 96.4
[98]	DT	Custom	-	Holdout	98.9	99.5	97.7	98.6
[59]	ELM	NSL-KDD	Adaptive Feature Selection	Cross-validation	91.7	-	-	-
[118]	DT Bagging	NSL-KDD KDD+Kyoto 2006	Feature Selection, Algorithmic Improvement	Cross-validation	84.5 99.8	-	-	-
[78]	RF (low-end) DT (high-end)	Custom	Feature Construction, Data Reduction	Cross-validation	92.9 92.2	94.7 96.5	-	96.3 95.8
[104]	NB	MQTT UNSW-NB15 IoTID20	Feature Selection and Extraction	Holdout	64.1 77.4 25.2	59.99 86 100	80 90 20	74 73 98
[87]	SVM	Custom	Packet Aggregation, Pre-trained Model Using	-	86	-	-	86

ML Algorithms*: Best Performing ML Algorithms, Accuracy: Acc., Precision: Pr., Recall: Re, Fi-score: FS.

5 Resource Consumption of Distributed ML in Network Security

In this chapter, an experiment was conducted to answer **RQ3**. Two popular distributed learning methods were investigated: FL and SL. The experiment was conducted focusing on resource consumption fingerprinting in terms of memory, CPU utilization, power consumption, and potential CO_2 emission while considering the accuracy of the models. The chapter presents these two ML models briefly, and then the required experimental setup and experiment procedures are described.

5.1 Federated Learning

Federated learning [119] is a method for training machine learning models on data distributed across multiple devices without centralizing or sharing the raw data. Initially, each client is given a copy of a pre-existing base model. This model is then further trained using the client's private data. The updated models from each client device are then sent back to a central server. The server combines the updated model parameters to create a new, global model. This global model is then distributed back to each client, and the process repeats in cycles until the model is fully trained. One of the key benefits of this process is that it does not require sharing raw data with the server, thereby preserving the client's privacy. Furthermore, federated learning allows for simultaneous training across multiple clients, which is a significant advantage. A general overview of the Federated Learning process is depicted in Fig. 5.1. However, one potential challenge with this approach is that the clients must train the entire model, which can be problematic for client devices with limited resources [120].

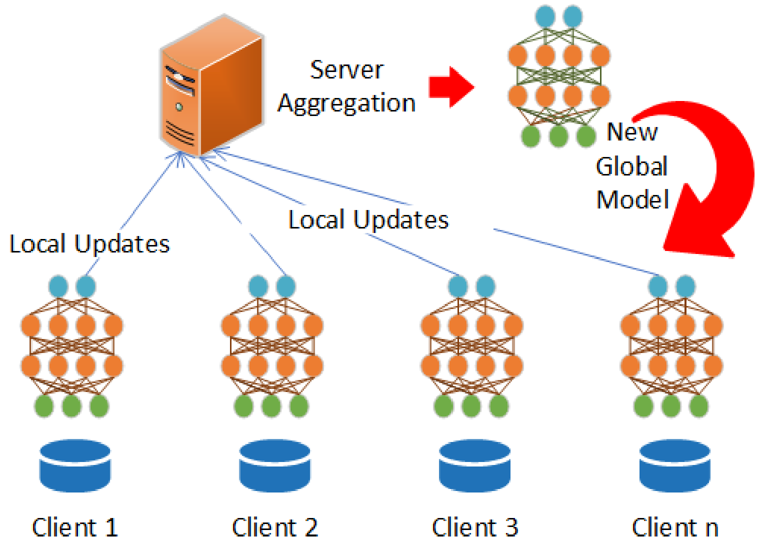


Figure 5.1: A High-level Presentation of FL Mechanism.

5.2 Split Learning

SL involves dividing a deep neural network into several sections, each training by a different client. This differs from FL, where the entire model is trained on the client device. In split learning, only a portion of the neural network is trained on the user's end, while the rest is trained on the server. Hence, the clients are relieved from the burden of the resource-intensive part of the computation, which is crucial for resource-constrained devices. In SL, the model architecture is divided into layers, and each client maintains the weights of what is learned until a specific layer, known as a split layer. The server hosts the remaining layers. SL reduces the communication payload size that must be sent during distributed training. This is because only the activation from the split layer needs to be sent from each client to the server. Similarly, only the gradient from the layer following the split layer needs to be sent from the server to the client [121]. Split learning is considered to offer more excellent privacy protection than FL. A general training process of split learning is illustrated in Fig. 5.2. However, unlike FL, training in split learning is sequential. Only one client interacts with the server at any given time, which could result in significant training overhead when there are many clients.

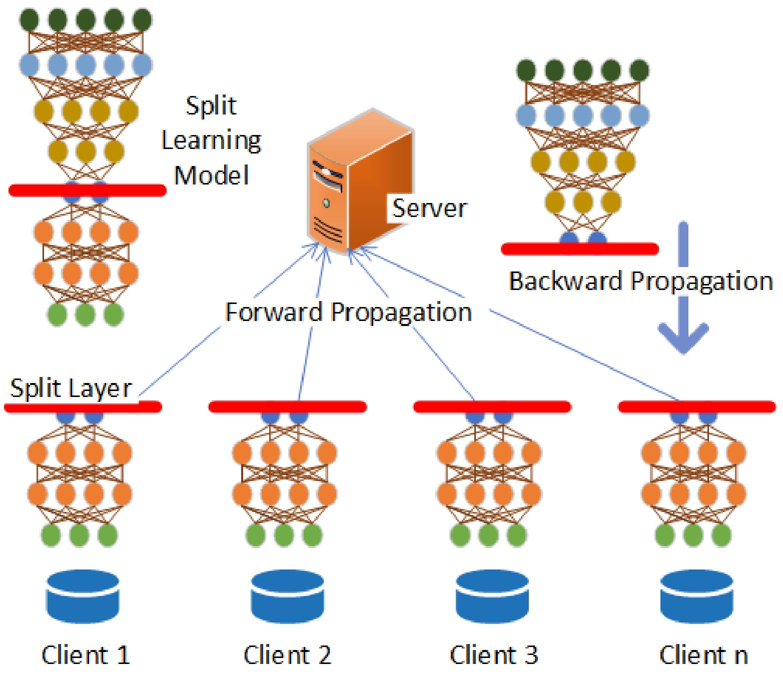


Figure 5.2: A High-level Presentation of SL Mechanism.

5.3 Experimental Setup and Model Training

This section describes the considered use case, the experimental environment, the used dataset for the experiment and its specification, the process regarding data preparation, the nature of ML models used, the matrices that are used for performance comparison, and the possible resource measurement tools that can be used in the experiment.

5.3.1 Use Case

6G networks are poised to be the cornerstone of future communication systems, catering to various sectors, including V2X, IoT, healthcare, and public and emergency services. As such, it's crucial to safeguard these pervasive networks from different types of attacks. A DDoS attack, for instance, which is relatively simple to carry out, can lead to large-scale disruption of various essential services, directly or indirectly. Moreover, ML models are also vulnerable to DoS attacks arising from input flooding, resource exhaustion, model overload, and data manipulation. These attacks can lead to increased resource usage, drain system resources, and result in unresponsiveness, posing significant threats to the reliability and security of 6G networks. Distributed Machine Learning-based DDoS detection solutions, such as FL and SL, can be employed to identify and mitigate these malicious requests that aim to cause DDoS. In these scenarios, the distributed ML model can leverage data and computational power from various devices involved while preserving user privacy, as the raw data is not shared with the server.

A variety of devices, including drones, IoT devices, smartphones, smart helmets, cameras, autonomous vehicles, and others, can be equipped with distributed ML. It's essential to ensure minimal resource consumption for training on these devices. Hence, these devices can continue to perform their primary functions without any hindrance. Fig 5.3 illustrates a conceptual scenario where a diverse range of devices are used to train a global model using distributed ML for 6G networks. Additionally, it's essential to keep resource consumption to a minimum to reduce the carbon footprint and foster a sustainable green network.

5.3.2 Experimental Environment

The evaluation was carried out on a high-performance system featuring an x86-64 architecture and a 12th Gen Intel® Core™ i9-12900K CPU. This system boasts 24

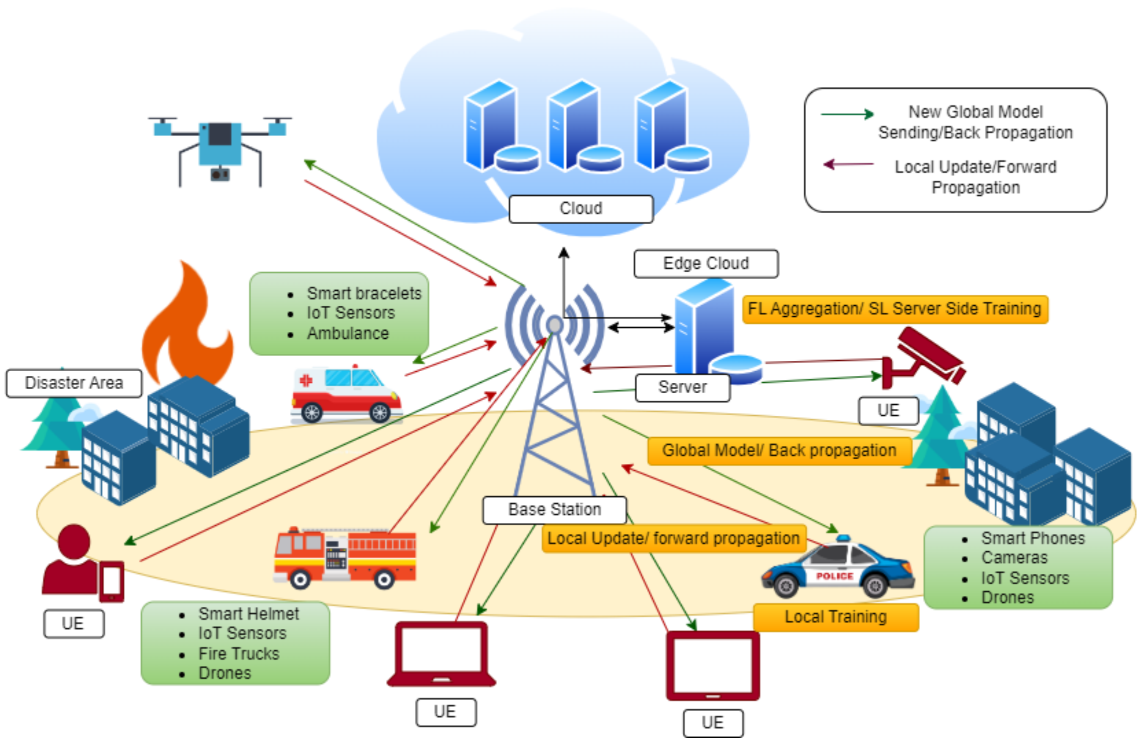


Figure 5.3: A Generalized 6G Network Representation with Key Relevant Network and User Segments.

CPUs and 16 cores per socket. The CPU speed fluctuates between 800 and 6700 MHz, and it supports VT-x virtualization. In addition, it has an L1d cache of 384 KiB, an L1i cache of 256 KiB, and an L2 cache of 10 MiB. The system is also equipped with 64GB of RAM. The experiments were performed using Anaconda PyTorch and Jupyter Notebook, tools that are highly regarded in the ML community for their robustness, flexibility, and interactive capabilities in development, execution, and sharing of experiments.

5.3.3 Dataset

This work utilized the DDoS evaluation dataset (CIC-DDoS2019), a benchmark dataset prepared by the Canadian Institute for Cybersecurity [122]. The dataset has 50,063,112 entries, where 50,006,249 instances represent DDoS attacks, and

56,863 instances represent benign (normal) behavior. The dataset is stored in 11 packages, where it has 12 types of DDoS attacks, which include NetBIOS, MSSQL, WebDDoS, LDAP, DNS, SNMP, NTP, UDP, SSDP, Syn, TFTP, and UDP Lag. For the experiment, 50,000 instances from 11 packages were randomly chosen from the dataset, resulting in 550000 instances. The resulting dataset from the original contains both attack instances and normal instances. However, there were only 5330 normal data while the rest, 544670, were different attack types mentioned earlier.

5.3.4 Data Pre-Processing and Feature Extraction

Data preparation is a crucial step for training any machine learning model. The real-world data can be inconsistent and may have missing values, duplicates, outliers, or errors. The dataset was inspected and handled for infinite, NaN, and negative values. Some features, such as source and destination IPs, timestamps, etc., which were considered less informative, were removed from the dataset to enhance the training process and decrease the data size. The *StandardScaler* was employed to standardize the dataset's features by removing the mean and scaling to unit variance. Additionally, the PCA method was used to extract the most significant features, resulting in 30 features. After the feature extraction, the target variables were binary encoded, with all the attack data labeled as 0 and the Normal data labeled as 1. The data was split into a training and testing set at a ratio of 75:25. The training dataset was balanced with the Synthetic Minority Oversampling Technique (SMOTE), which generates synthetic data using a linear interpretation between the minority class samples and their K-nearest neighbors. After balancing, the training set had 816928 instances with nearly equal numbers of DDoS and Normal data. The test dataset was imbalanced with 136206 Attack data and 1294 Normal data out of 137500 instances.

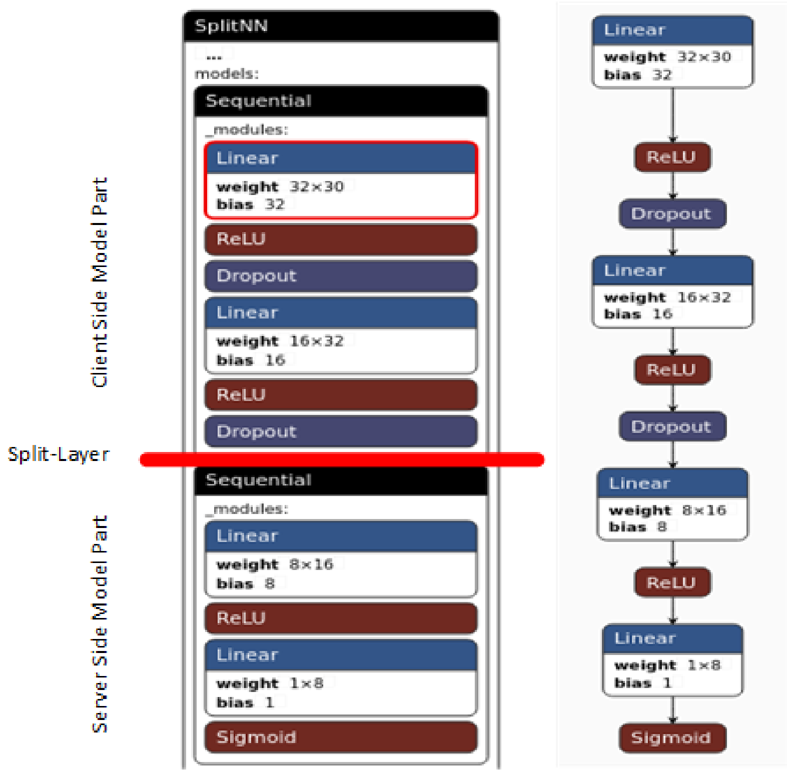
5.3.5 Model Training Setup

The experimental evaluations were conducted using the FL and SL source code adapted from [123]. The experiments involved a total of 32 client nodes, with the number of clients used for training at a time varying from 10, 16, and 24. It's assumed that the clients are spread across the network, and each client model is trained with a distinct dataset. However, the study did not take into account any data loss resulting from network congestion or channel interference. A similar neural network structure for both models (SL and FL) is implemented, as shown in Fig 5.4. In the case of FL, each client trains an MLP model, as shown in the figure 5.4. In the case of SL, for each selected client, a split Neural Network model is created with the client and server models. Both models are feed-forward neural networks that incorporate dropout regularization and utilize the ReLU activation function in their hidden layers. The inclusion of a dropout layer mitigates the problem of overfitting by randomly nullifying a portion of the input units during each update in the training phase. The model's parameters were updated during the training process using the Adam optimization algorithm, with a learning rate set at 0.001. The models were trained with the training dataset for five epochs for 50 rounds. After each round of training, the model is tested with a test dataset, and its performance is assessed. This process is repeated for all 10, 16, and 24 batches of clients for training out of the 32 clients to measure the time, CPU usage, memory usage, power consumption, and CO2 emission for the entire process.

5.3.6 Performance Metrics

The DDoS attack detection was considered as a binary problem, where the 1 represents the normal flow, while the 0 represents an attack. Four possible unique outcomes can be found in ML models. They are:

- ***True Positive (TP):*** When the ML model correctly predicts the positive or



(a) SL Model (b) FL Model

Figure 5.4: Considered FL and SL Models Structure for the Experiment.

DDoS attack in this case.

- ***True Negative (TN):*** When the model correctly predicts negative or Normal flow in this case.
- ***False Positive (FP):*** When the model incorrectly predicts positive but the actual label is negative.
- ***False Negative (FN):*** When the model incorrectly predicts negative but the actual label is positive.

While comparing the FL and SL models, the standard performance matrices Accuracy, Precision, Recall, and F1-Score were used.

- ***Accuracy:*** It is the ratio of correct predictions to the total number of predictions. It is defined as:

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

- ***Precision:*** It is the proportion of the correctly predicted positive assumption made by the classifier. It is defined as:

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

- ***Recall:*** It is the proportion of actual positive prediction that is predicted correctly. It is defined as:

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

- ***F1-Score:*** It is the weighted average of precision and recall. It is useful to understand the rate of false positives and false negatives in the prediction model.

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

We also evaluate them from the resource efficiency points of view in terms of CPU usage, memory usage, required time, energy consumption, and CO2 emission.

5.3.7 Resource Consumption Measurement Tools

For the measurement of the resource consumption of ML algorithms, various tools, and libraries are available in the literature. Hence, a list of the potential tools to measure those resources is prepared shown in Table 5.1. Although the experiment utilized only a single tool for each corresponding resource, it is expected that the prepared list will aid in further research on resource consumption in the ML research field.

Table 5.1: Tools to Measure the Resource Consumption

Energy	**Processing**	**Memory**	**Time**	**Bandwidth**
Tegrastats [68] Makerhawk UM25C USB Tester [70] ARM Stream Line [124] Powmon [125] Intel Power Gadget [126] McPAT [127] PAPI [128] Eco2AI[7] Altera's Power Monitor Tool [64] Custom Power Measurement Platform [64]	Tegrastats [68] Conky [66] Linux Command "top" [87] Resmon [129] Simple System Monitor [67]	Tegrastats [68] Conky [66] Memory-profiler [101] Resmon [129] Simple System Monitor [67] Tracemalloc [87] Fil Memory Profiler [130] Sciagraph Profiler [131]	Tegrastats [68] IPython-autotimer [101] Conky [66] Time[132]	Bmon [70] Iftop [66]

6 Results and Discussion

One of the main goals of this work was to study the resource consumption of SL and FL algorithms in DDoS detection, their change in performance in highly imbalanced test datasets with various batch sizes of clients, and the impact of the dataset on various resource consumption. The results of the experiments from Chapter 5 are analyzed and presented in this chapter.

6.1 CPU Usage

The CPU and Memory usage was measured by using the Python library *psutil*, which provides an interface for measuring system resources like CPU, disk, memory, and network. To measure CPU and memory usage, a separate background thread was created which runs until the stop_event is triggered. This thread runs concurrently with the main process and continuously samples and stores the CPU and Memory usage of the FL and SL. The *cpu_percent* method of the *psutil* library with "interval=0" was used for getting the real-time and instantaneous CPU percentage usage of the Python script. There is no delay between the measurements, providing an immediate snapshot of CPU usage. According to the result, the average CPU usage for the FL 10 clients batch is 8.10%, while for SL, it is slightly higher at 8.52%. This indicates that, on average, SL utilises more CPU resources than FL. The maximum CPU usage for FL is 17.4%, which is significantly higher than the average. This indicates that certain operations in FL are particularly resource-intensive. On the other hand, for SL, it is 13.3%, which is closer to the average. However, the standard deviation of the CPU usage for FL and SL are 2.78 and 2.91, indicating that, even though SL has a lower maximum CPU usage than FL, its CPU usage values vary more around its mean than FL.

For 16 clients batch, The average CPU usage for FL is 7.78%, while for SL, it is significantly lower at 5.22%. This indicates that, on average, SL utilises less CPU resources than FL. However, the maximum CPU usage for FL and SL was 29.6% and 33.3%. Also, for SL, the highest spike to this maximum value was only for the first second, and then it stabilized and never went that up. After that, the maximum value for SL was 7.4%. The minimum CPU usage for FL is 0.0%, and for SL, it is 4.4%, indicating that SL has a higher baseline CPU usage than FL. Furthermore, the standard deviation for FL is 5.36, and for SL is significantly lower at 1.39, indicating that FL's CPU usage has more variation than SL. Finally, for 24 clients, the average CPU usage for FL is 8.38%, while for SL, it is slightly higher at 8.81%. This indicates that, on average, SL utilises more CPU resources than FL. Whereas the maximum CPU usage for FL is 18.7%, and for SL, it is 12.2%, and the standard deviation for FL is 2.28, and for SL, it is 1.10. Hence, while SL has high average CPU usage, FL has a higher peak (resource-intensive operations) and more variability. Fig 6.1 shows the CPU usage of the FL and SL.

6.2 Memory Usage

For memory usage, *memory_ info().rss* from *psutil* was used to retrieve the Resident Set Size (RSS) memory usage of the current process for every second during the entire process. The result was converted from bytes to megabytes and stored in *memory_ usage* list. In terms of memory usage, it was observed that SL's memory usage increased linearly over time, whereas FL's memory usage increased until a certain point, after which it remained nearly constant. For instance, in FL with 10,

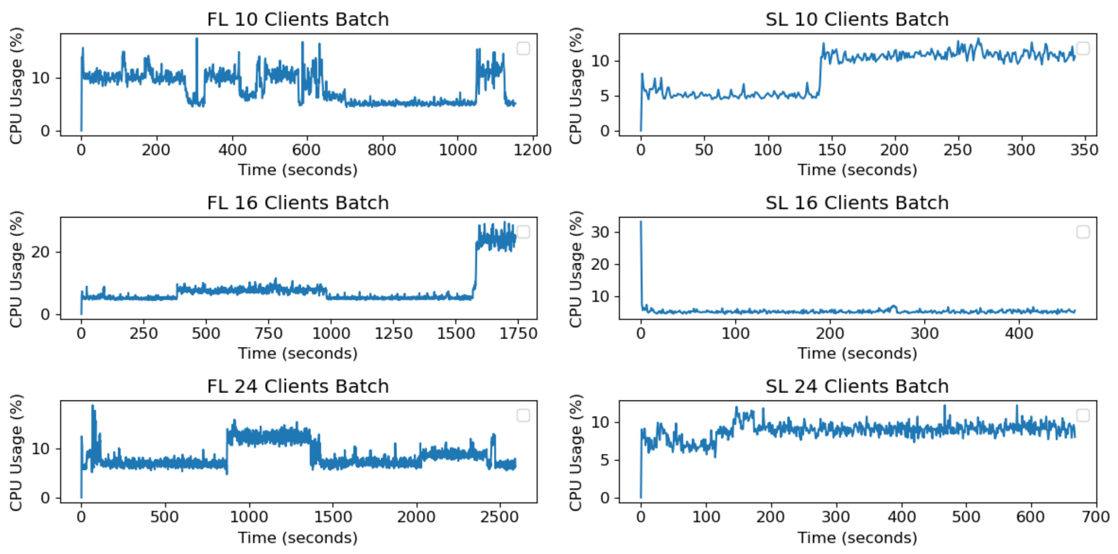


Figure 6.1: CPU Usage Over Time.

16, and 24 client batches, the memory usage increased to 1974.59 MB, 2000.15 MB, and 1960.54MB at the initial stage of execution, and they became nearly constant at that point for the rest of the time. Whereas, SL used significantly more memory than FL in all the scenarios. For example, memory usage for SL increased with time for all the scenarios and reached 5997.21 MB, 8286.375 MB, and 11335.62MB at the end of 50 rounds for 10, 16, and 24 clients, respectively, which is 3.04, 4.14 and 5.78, times more memory usage than FL. Fig. 6.2 shows the memory usage comparison of the FL and SL.

6.3 Power Consumption and CO2 Emission

For measuring power consumption, Eco2AI python library [7] was used. This tool can accumulate statistics about power consumption and CO_2 emission while running a code. The experiment result found that the power consumption and CO_2 emission are significantly higher for FL than SL for all scenarios. For instance, while the training batch size was 10 clients, the power consumption of SL was approximately

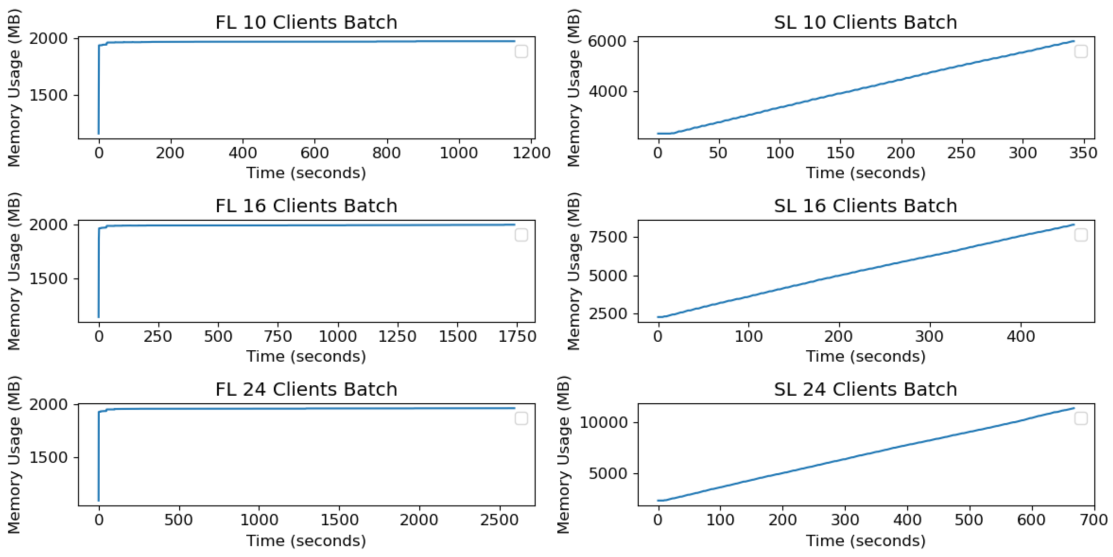


Figure 6.2: Memory Usage Over Time.

3.14 times less than FL, while the CO_2 emission was approximately 3.16 times less than FL. In the case of 16 clients, the power consumption of SL is approximately 3.45 times less than FL, and CO_2 emission is approximately 3.43 times less than FL. Finally, in the case of 24 batch clients, SL consumes approximately 3.48 times less power than FL and approximately 3.46 times less CO_2 emission than FL. The comparison results are shown in Fig. 6.3.

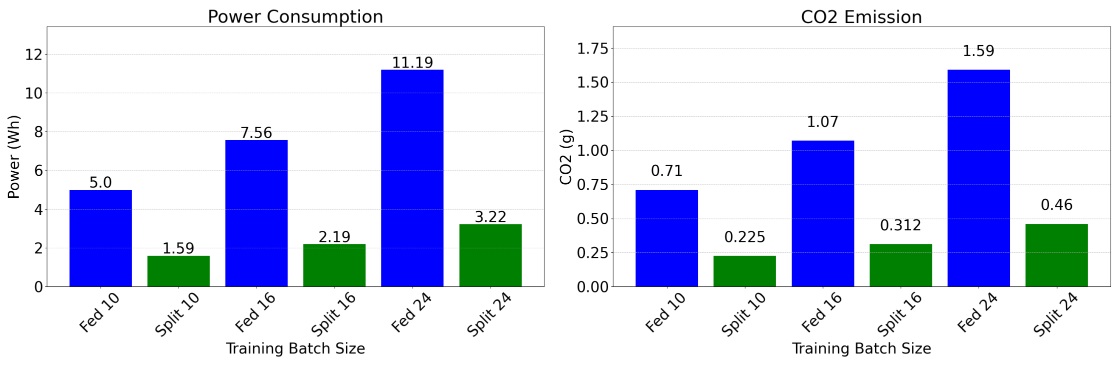


Figure 6.3: Energy Consumption and CO2 Emission.

6.4 Turnaround Time for Execution

For measuring the turnaround time, the Python library Time[132] was used. This is the total time required to complete the 50 training rounds and the corresponding testing. The experiment finding shows that FL requires significantly more time than the SL for all the scenarios. For instance, for 10 client batches, FL required approximately 3.37 times more time than the FL. For 16 and 24 client batches, it required 3.79 and 3.89 times more time, as shown in Fig. 6.3.

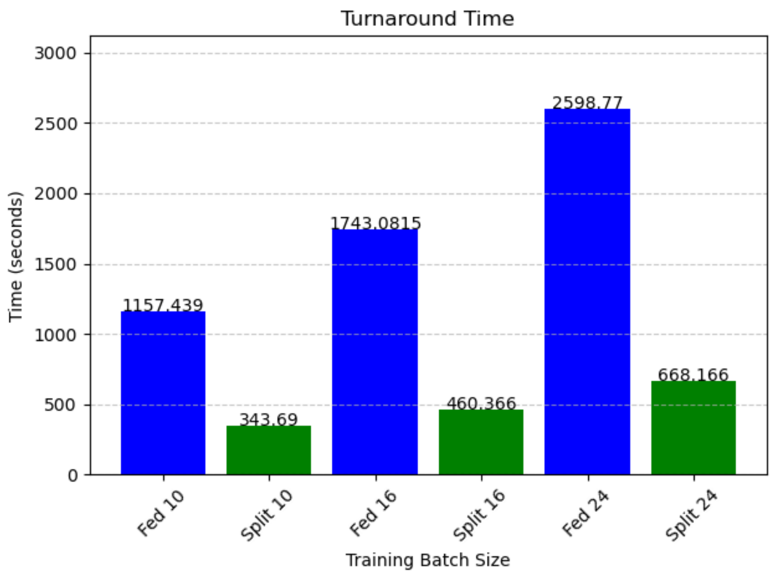


Figure 6.4: Turnaround Time for Execution.

6.5 Performance Evaluation

Along with the resource consumption, the performance of FL and SL is also compared in DDoS detection. Where the numbers of client batches were varied to 10, 16, and 24 clients during training for 32 clients. From the result, we observed that both FL and SL demonstrated high performance in DDoS detection. The average accuracy for both methods is quite high (> 0.996) across all batch sizes, indicating that both methods are highly accurate. Table 6.1 shows the performance of both methods with different numbers of clients. FL had slightly lower false positives than SL and better precision. Also, the precision improved with the increase in client batch size and with the increment in rounds shown in Fig 6.5. In the case of Recall, both methods performed very well and were close to 1, indicating that both methods are excellent at detecting positive instances. Finally, the F1 score of FL is slightly better than the SL for the entire period for all scenarios. This suggests that FL is slightly better at maintaining a balance between precision and recall.

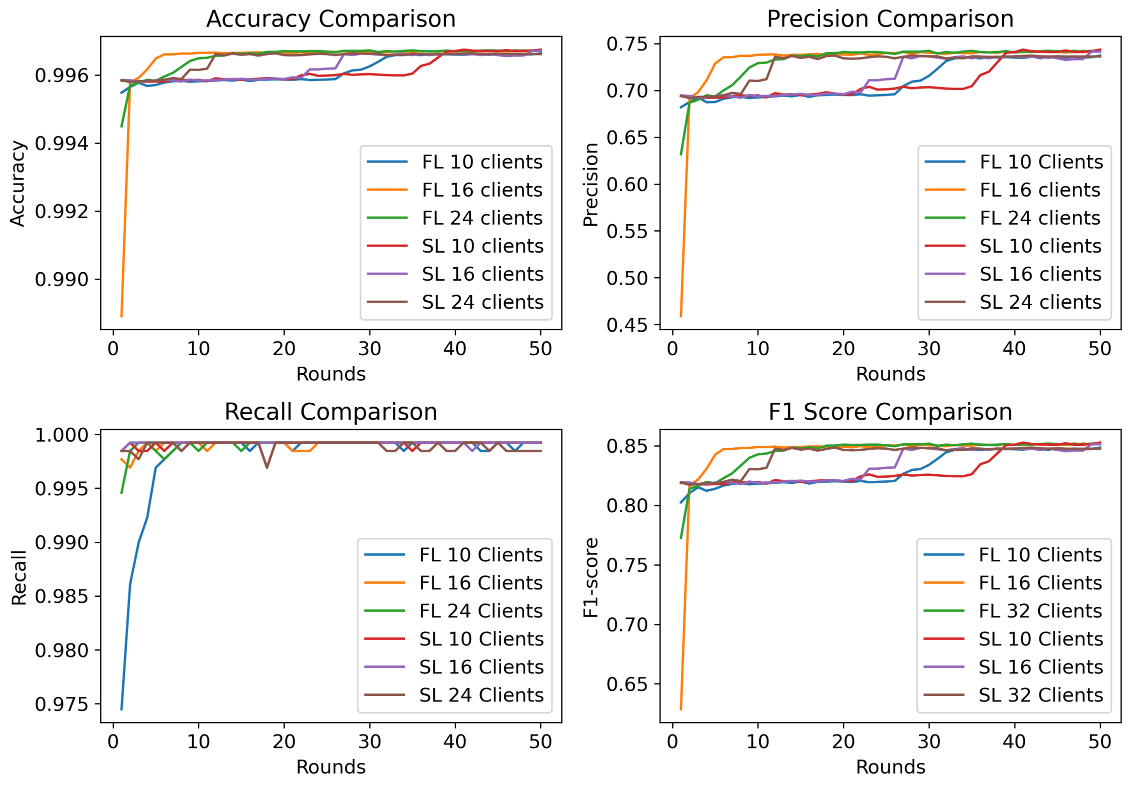


Figure 6.5: Performance Comparison of FL and SL.

Table 6.1: The Performance Comparison of FL and SL

Models	Metrics			
	Avg. Accu	Avg. Pre	Avg. Rec	Avg. F1 Sco
Fed 10	0.9962	0.7109	0.9980	0.8302
Split 10	0.9961	0.7098	0.9991	0.8298
Fed 16	0.9965	0.7312	0.9990	0.8437
Split 16	0.9962	0.7158	0.9992	0.8339
Fed 24	0.9965	0.7314	0.9990	0.8444
Split 24	0.9965	0.7276	0.9989	0.8419

6.6 Impact of Data on the Resources and Performance

When the size of client batches increases, the amount of data needed to process also increases. Both FL and SL can handle greater client batch sizes without noticeably sacrificing their performance, as evidenced by the models' nearly identical performance when the client batch size increases in terms of scalability. Whereas for resource consumption, it was observed that the amount of resource consumption increases significantly with the increase in client batch size, especially memory, time, and power. This is expected since the larger batch size means more data to process, which naturally requires more computational resources. Based on the experiment evaluation, SL consistently used fewer resources than FL for all the batch sizes, indicating better resource efficiency. However, it was also observed that the memory usage for SL was much higher than for FL. For FL, it was nearly constant for all the batch sizes, while SL increased significantly with the batch size.

6.7 Recommendations from Analysis

Devices operating on 6G networks will have varying resource requirements, including power, processing, memory, bandwidth, and other types of resources, in addition to data privacy requirements. It was discussed in the earlier sections that FL and

SL train models in distributed manners without the requirement of centralizing the raw data, and hence they safeguard users' privacy. Also, this approaches reduce the computational load on the devices. However, it was observed in the experiment that, although both techniques showed nearly similar performance in terms of DDoS detection, their resource consumption fingerprint was significantly different. For instance, SL showed superior resource efficiency in terms of (time, power consumption, and CO_2). This is because FL needs to train the entire client model, while SL trains only a part of the model. Hence SL will be suitable for large-scale networks like IoT or time-sensitive applications such as V2X. Nonetheless, in scenarios where data is unevenly distributed among the devices, or when the complexity of the model cost and communication is not substantial, Federated Learning can be utilized. Ultimately, the choice of distributed learning type for a 6G network will depend on various aspects like computational capabilities, attack nature, communication overhead, privacy needs, and other factors.

7 Conclusion and Future Work

As the advent of the 6G era approaches, extensive research is being conducted on the role of ML in various sectors of 6G. To contribute to the ongoing research and to participate in the advancement of 6G, the thesis work investigated the resource consumption and resource efficiency of ML techniques. The first objective of the work was to find out the key resources that are required in ML-based network security solutions, and the thesis work studied literature, prepared a list of those resources, and presented the reason why and how those resources are required for ML techniques. The second objective was to investigate the resource-efficient ML-based network security solutions and find out the key enablers of resource efficiency in those solutions. The thesis studied a wide range of literature and prepared a list of possible ways to achieve resource efficiency in a network security solution. in a network security solution. Additionally, the work also collected the possible resources that can be useful for this purpose. Those resources include the list of better-performing ML algorithms in different use cases and datasets for different kinds of network security solutions. It also listed different useful tools that can be used to measure the resource consumption of ML. Finally, the third objective was to evaluate the possible resource consumption of deploying ML techniques for 6G, and this thesis conducted an experiment based on a DDoS detection use case. The answers to the research questions mentioned at the beginning of the thesis are discussed below:

7.1 Result of RQ1

RQ1: What are the key resources that are consumed by ML techniques used in network security?

According to our findings, while ML techniques are deployed in a network security solution, they consume different resources. The focus of the thesis has been on Power/Energy, Bandwidth, Processing, Memory, time, and required human involvement to supervise the ML model, its selection, or the whole process. There are also other resources that are involved, for instance in the acquisition of data, its storage, and the relevant processing of these data.

7.2 Result of RQ2

RQ2: How resource efficiency in network security is achieved in different ML techniques?

In various network security solutions, the researchers intend to improve resource usage, including energy/power, processing, memory, time, bandwidth, and human involvement. The researchers used various techniques to achieve resource efficiency while developing an approach. For instance, to achieve efficient energy consumption, techniques such as feature selection, feature extraction, hardware implementation, offloading computation, hyperparameter optimization, distributed learning, and some sorts of algorithmic improvement techniques were used. For efficient processing utilization, feature extraction, feature selection, feature engineering, snort rule generation, period-wise-detection, packet-wise-detection, VAE, cyclic queuing, and using a small chunk of the dataset were used. For efficient memory utilization, pruning, tuning, simulated micro-batching, hyper-parameter optimization, parameter regularization, weight-averaging, feature selection, and reduction techniques were used. To improve the time efficiency, feature engineering, parameter tuning, decentralized training, feature selection, feature extraction, hysteretic rule integration, reduction, multi-level tweak, normalization, algorithmic improvement, etc., were used. For efficient bandwidth consumption, an efficient transformer, a fourier mixing sublayer, distributed training, VAE, feature selection, feature extraction, and reduction were used. Finally, for human involvement reduction, auto hyperparameter tuning, active learning, unsupervised learning, dynamic evolving of NN, etc., were used.

7.3 Result of RQ3

RQ3: How much resources are consumed by distributed learning techniques like Federated Learning (FL) and Split Learning and how do they perform in a network security test case?

This work's third goal was to provide insight into how resources might be consumed in ML-based distributed network security solutions for 6G. For this purpose, the work presented a comparative analysis of resource usage of a DDoS detection system using FL and SL. It was observed from the experiment that both techniques can detect DDoS attacks very effectively with very high accuracy. Although the detection performance is pretty similar for both techniques, their resource usage differs significantly, and both require a considerable amount of resources. In terms of CPU usage, SL consumed a higher average CPU, while FL had a higher peak and variability. In the case of memory, SL consumed significantly more memory than FL, while FL's memory usage was more consistent. However, SL was more power and time-efficient and emitted lower CO_2.

7.4 Limitation of the Thesis Work

This work has certain limitations, for instance, it was mentioned that data and storage are also two important resources for ML-based network security solutions, but the thesis work did not include them in the KPIs. Moreover, although the work considers human involvement an important resource for ML-based network security solutions, it did not find any scale or tool to measure human involvement in machine learning. Furthermore, the experiments that conducted here, were done on a single machine instead of distributed machines. Additionally, the resource consumption was measured for all the clients and servers altogether, not separately. Hence, the work could not determine how much resources the server consumes and how much each client consumes in FL and SL. Moreover, the CO_2 emission can be different based on the source of energy. For example, fossil fuel produces significantly higher CO_2, while renewable energy produces minimal CO_2. However, the Eco2ai considered the global average emission coefficient for calculating the emission, which is 436.529 kg/MWh. Also, hence the work did not deploy the technique in any network, it excluded the bandwidth measurement from the experiment, and it could not measure the amount of human involvement in this work. Finally, the measurement tools and the libraries used in this experiment also consumed some resources, but the work did not exclude those resources from the measurements.

Despite limitations, the experiments serve as a good foundation for further research in the area that is still in the early stages, providing several intriguing insights into the resource consumption of distributed ML for security in 6G and beyond. Such results are of significant importance as they will help design a resource-efficient, sustainable 6G network with a lower overall carbon footprint, which has become a key consideration in tackling climate change.

7.5 Future work

The future work includes establishing a test bed and conducting a series of experiments using various centralized and distributed machine learning-based network security solutions. There is also an aim to measure the resource requirements for each client device and investigate whether an attack or compromise on any node affects the resources of other nodes. Furthermore, there will be a strive to design a use case-specific resource-efficient SL or FL algorithm that performs optimally for that particular use case, and deploying that solution into the test bed. Furthermore, it is also aimed to develop generalized techniques for detecting a combination of security threats and then providing protection against those security threats.

References

[1] L. Cui, S. Yang, F. Chen, Z. Ming, N. Lu, and J. Qin, “A survey on application of machine learning for internet of things”, *International Journal of Machine Learning and Cybernetics*, vol. 9, no. 8, pp. 1399–1417, 2018. DOI: 10.1007/s13042-018-0834-5.

[2] I. Ahmad, J. Suomalainen, P. Porambage, A. Gurtov, J. Huusko, and M. Höyhtyä, “Security of satellite-terrestrial communications: Challenges and potential solutions”, *IEEE Access*, vol. 10, pp. 96 038–96 052, 2022. DOI: 10.1109/ACCESS.2022.3205426.

[3] I. Ahmad, A. Mämmelä, M. M. Mowla, A. Flizikowski, M. A. B. Abbasi, D. Zelenchuk, and M. Sinaie, “Sustainability in 6G networks: Vision and directions”, in *2023 IEEE Conference on Standards for Communications and Networking*, 2023, pp. 202–208. DOI: 10.1109/CSCN60443.2023.10453200.

[4] I. Ahmad, S. Shahabuddin, H. Malik, E. Harjula, T. Leppänen, L. Lovén, A. Anttonen, A. H. Sodhro, M. Mahtab Alam, M. Juntti, A. Ylä-Jääski, T. Sauter, A. Gurtov, M. Ylianttila, and J. Riekki, “Machine learning meets communication networks: Current trends and future challenges”, *IEEE Access*, vol. 8, pp. 223 418–223 460, 2020. DOI: 10.1109/ACCESS.2020.3041765.

[5] I. Ahmad, S. Shahabuddin, T. Kumar, J. Okwuibe, A. Gurtov, and M. Ylianttila, “Security for 5G and beyond”, *IEEE Communications Surveys Tutorials*, vol. 21, no. 4, pp. 3682–3722, 2019. DOI: 10.1109/COMST.2019.2916180.

[6] I. Ahmad, S. Shahabuddin, T. Sauter, E. Harjula, T. Kumar, M. Meisel, M. Juntti, and M. Ylianttila, "The challenges of artificial intelligence in wireless networks for the internet of things: Exploring opportunities for growth", *IEEE Industrial Electronics Magazine*, vol. 15, no. 1, pp. 16–29, 2021. DOI: `10.1109/MIE.2020.2979272`.

[7] S. A. Budennyy, V. D. Lazarev, N. N. Zakharenko, A. N. Korovin, O. A. Plosskaya, D. V. Dimitrov, V. S. Akhripkin, I. V. Pavlov, I. V. Oseledets, I. S. Barsola, I. V. Egorov, A. A. Kosterina, and L. E. Zhukov, "Eco2ai: Carbon emissions tracking of machine learning models as the first step towards sustainable AI", *Doklady Mathematics*, vol. 106, no. 1, S118–S128, 2022, ISSN: 1531-8362. DOI: `10.1134/S1064562422060230`.

[8] A. E. Brownlee, J. Adair, S. O. Haraldsson, and J. Jabbo, "Exploring the accuracy – energy trade-off in machine learning", in *2021 IEEE/ACM International Workshop on Genetic Improvement*, 2021, pp. 11–18. DOI: `10.1109/GI52543.2021.00011`.

[9] J. Suomalainen, A. Juhola, S. Shahabuddin, A. Mämmelä, and I. Ahmad, "Machine learning threatens 5G security", *IEEE Access*, vol. 8, pp. 190 822–190 842, 2020. DOI: `10.1109/ACCESS.2020.3031966`.

[10] M. Nazir, A. Sabah, S. Sarwar, A. Yaseen, and A. Jurcut, "Power and resource allocation in wireless communication network", *Wireless Personal Communications*, vol. 119, no. 4, pp. 3529–3552, 2021. DOI: `10.1007/s11277-021-08419-x`.

[11] I. Ahmad, Z. Asghar, T. Kumar, G. Li, A. Manzoor, K. Mikhaylov, S. A. Shah, M. Höyhtyä, J. Reponen, J. Huusko, and E. Harjula, "Emerging technologies for next generation remote health care and assisted living", *IEEE Access*, vol. 10, pp. 56 094–56 132, 2022. DOI: `10.1109/ACCESS.2022.3177278`.

[12] D. P. Moya Osorio, I. Ahmad, J. D. V. Sánchez, A. Gurtov, J. Scholliers, M. Kutila, and P. Porambage, “Towards 6G-enabled internet of vehicles: Security and privacy”, *IEEE Open Journal of the Communications Society*, vol. 3, pp. 82–105, 2022. DOI: 10.1109/OJCOMS.2022.3143098.

[13] R. Dangi, P. Lalwani, G. Choudhary, I. You, and G. Pau, “Study and investigation on 5G technology: A systematic review”, *Sensors*, vol. 22, no. 1, 2022, ISSN: 1424-8220. DOI: 10.3390/s22010026.

[14] W. Saad, M. Bennis, and M. Chen, “A vision of 6G wireless systems: Applications, trends, technologies, and open research problems”, *IEEE Network*, vol. 34, no. 3, pp. 134–142, 2020. DOI: 10.1109/MNET.001.1900287.

[15] I. Ahmad, F. Rodriguez, J. Huusko, and K. Seppänen, “On the dependability of 6G networks”, *Electronics*, vol. 12, no. 6, 2023, ISSN: 2079-9292. DOI: 10.3390/electronics12061472.

[16] M. Giordani, M. Polese, M. Mezzavilla, S. Rangan, and M. Zorzi, “Toward 6G networks: Use cases and technologies”, *IEEE Communications Magazine*, vol. 58, no. 3, pp. 55–61, 2020. DOI: 10.1109/MCOM.001.1900411.

[17] E. Calvanese Strinati, S. Barbarossa, J. L. Gonzalez-Jimenez, D. Ktenas, N. Cassiau, L. Maret, and C. Dehos, “6G: The next frontier: From holographic messaging to artificial intelligence using subterahertz and visible light communication”, *IEEE Vehicular Technology Magazine*, vol. 14, no. 3, pp. 42–50, 2019. DOI: 10.1109/MVT.2019.2921162.

[18] J. R. Bhat and S. A. Alqahtani, “6G ecosystem: Current status and future perspective”, *IEEE Access*, vol. 9, pp. 43 134–43 167, 2021. DOI: 10.1109/ACCESS.2021.3054833.

[19] S. J. Nawaz, S. K. Sharma, B. Mansoor, M. N. Patwary, and N. M. Khan, “Non-coherent and backscatter communications: Enabling ultra-massive con-

nectivity in 6G wireless networks", *IEEE Access*, vol. 9, pp. 38 144–38 186, 2021. DOI: 10.1109/ACCESS.2021.3061499.

[20] W. Jiang, B. Han, M. A. Habibi, and H. D. Schotten, "The road towards 6G: A comprehensive survey", *IEEE Open Journal of the Communications Society*, vol. 2, pp. 334–366, 2021. DOI: 10.1109/OJCOMS.2021.3057679.

[21] A. Dutta and E. Hammad, "5G security challenges and opportunities: A system approach", in *2020 IEEE 3rd 5G World Forum*, 2020, pp. 109–114. DOI: 10.1109/5GWF49715.2020.9221122.

[22] V.-L. Nguyen, P.-C. Lin, B.-C. Cheng, R.-H. Hwang, and Y.-D. Lin, "Security and privacy for 6G: A survey on prospective technologies and challenges", *IEEE Communications Surveys Tutorials*, vol. 23, no. 4, pp. 2384–2428, 2021. DOI: 10.1109/COMST.2021.3108618.

[23] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Understanding O-RAN: Architecture, interfaces, algorithms, security, and research challenges", *IEEE Communications Surveys Tutorials*, vol. 25, no. 2, pp. 1376–1411, 2023. DOI: 10.1109/COMST.2023.3239220.

[24] M. Liyanage, A. Braeken, S. Shahabuddin, and P. Ranaweera, "Open RAN security: Challenges and opportunities", *Journal of Network and Computer Applications*, vol. 214, p. 103 621, 2023, ISSN: 1084-8045. DOI: 10.1016/j.jnca.2023.103621.

[25] P. Porambage, G. Gür, D. P. Moya Osorio, M. Livanage, and M. Ylianttila, "6G security challenges and potential solutions", in *2021 Joint European Conference on Networks and Communications 6G Summit*, 2021, pp. 622–627. DOI: 10.1109/EuCNC/6GSummit51104.2021.9482609.

[26] F. Rosenblatt, *The Perceptron, a Perceiving and Recognizing Automaton Project Para*. Cornell Aeronautical Laboratory, 1957, vol. 85, Issues 460-461 of Report.

[27] A. L. Fradkov, "Early history of machine learning", *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 1385–1390, 2020, ISSN: 2405-8963. DOI: 10.1016/j.ifacol.2020.12.1888.

[28] T. Jiang, J. L. Gradus, and A. J. Rosellini, "Supervised machine learning: A brief primer", *Behavior Therapy*, vol. 51, no. 5, pp. 675–687, 2020, ISSN: 0005-7894. DOI: 10.1016/j.beth.2020.05.002.

[29] R. Choudhary and H. K. Gianey, "Comprehensive review on supervised machine learning algorithms", in *2017 International Conference on Machine Learning and Data Science*, 2017, pp. 37–43. DOI: 10.1109/MLDS.2017.11.

[30] P. C. Sen, M. Hajra, and M. Ghosh, "Supervised classification algorithms in machine learning: A survey and review", in *Emerging Technology in Modelling and Graphics*, Springer Singapore, 2020, pp. 99–111, ISBN: 978-981-13-7403-6. DOI: 10.1007/978-981-13-7403-6_11.

[31] H. Taherdoost, "Classification of machine learning algorithms", in *Advances in Data Computing, Communication and Security*, Springer Nature Singapore, 2022, pp. 417–422, ISBN: 978-981-16-8403-6. DOI: 10.1007/978-981-16-8403-6_38.

[32] H. U. Dike, Y. Zhou, K. K. Deveerasetty, and Q. Wu, "Unsupervised learning based on artificial neural network: A review", in *2018 IEEE International Conference on Cyborg and Bionic Systems*, 2018, pp. 322–327. DOI: 10.1109/CBS.2018.8612259.

[33] K. J. Cios, R. W. Swiniarski, W. Pedrycz, L. A. Kurgan, K. J. Cios, R. W. Swiniarski, W. Pedrycz, and L. A. Kurgan, "Unsupervised learning: Associa-

tion rules", *Data mining: A knowledge discovery approach*, pp. 289–306, 2007. DOI: 10.1007/978-0-387-36795-8_10.

[34] F. Ros and R. Riad, "Dimensionality reduction", in *Feature and Dimensionality Reduction for Clustering with Deep Learning*. Springer Nature Switzerland, 2024, pp. 11–25, ISBN: 978-3-031-48743-9. DOI: 10.1007/978-3-031-48743-9_2.

[35] X. Zhu and A. Goldberg, "Overview of semi-supervised learning", in *Introduction to Semi-Supervised Learning*. Springer Cham, 2009, ch. 2, pp. 09–19, ISBN: 978-3-031-01548-9. DOI: 10.1007/978-3-031-01548-9_2.

[36] X. Zhu and A. B. Goldberg, "Co-training", in *Introduction to Semi-Supervised Learning*. Springer International Publishing, 2009, pp. 35–42, ISBN: 978-3-031-01548-9. DOI: 10.1007/978-3-031-01548-9_4.

[37] X. Zhu, "Semi-supervised learning", in *Encyclopedia of Machine Learning*, C. Sammut and G. I. Webb, Eds. Springer US, 2010, pp. 892–897, ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_749.

[38] D. J. Joshi, I. Kale, S. Gandewar, O. Korate, D. Patwari, and S. Patil, "Reinforcement learning: A survey", in *Machine Learning and Information Processing*, Springer Singapore, 2021, pp. 297–308, ISBN: 978-981-33-4859-2. DOI: 10.1007/978-981-33-4859-2_29.

[39] J. Liu, J. Huang, Y. Zhou, X. Li, S. Ji, H. Xiong, and D. Dou, "From distributed machine learning to federated learning: A survey", *Knowledge and Information Systems*, vol. 64, no. 4, pp. 885–917, Apr. 2022. DOI: 10.1007/s10115-022-01664-x.

[40] D. Naik and N. Naik, "The changing landscape of machine learning: A comparative analysis of centralized machine learning, distributed machine learning and federated machine learning", in *Advances in Computational Intelli-*

gence Systems, Springer Nature Switzerland, 2024, pp. 18–28, ISBN: 978-3-031-47508-5. DOI: `10.1007/978-3-031-47508-5_2`.

[41] P. Chemouil, P. Hui, W. Kellerer, Y. Li, R. Stadler, D. Tao, Y. Wen, and Y. Zhang, "Special issue on artificial intelligence and machine learning for networking and communications", *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1185–1191, 2019. DOI: `10.1109/JSAC.2019.2909076`.

[42] M. Wang, Y. Cui, X. Wang, S. Xiao, and J. Jiang, "Machine learning for networking: Workflow, advances and opportunities", *IEEE Network*, vol. 32, no. 2, pp. 92–99, 2018. DOI: `10.1109/MNET.2017.1700200`.

[43] M. Cicioğlu and A. Çalhan, "HUBsFLOW: A novel interface protocol for SDN-enabled WBANs", *Computer Networks*, vol. 160, pp. 105–117, 2019, ISSN: 1389-1286. DOI: `10.1016/j.comnet.2019.06.007`.

[44] B. Dai, Y. Cao, Z. Wu, Z. Dai, R. Yao, and Y. Xu, "Routing optimization meets machine intelligence: A perspective for the future network", *Neurocomputing*, vol. 459, pp. 44–58, 2021, ISSN: 0925-2312. DOI: `10.1016/j.neucom.2021.06.093`.

[45] M. K. Awad, M. H. H. Ahmed, A. F. Almutairi, and I. Ahmad, "Machine learning-based multipath routing for software defined networks", *Journal of Network and Systems Management*, vol. 29, no. 2, p. 18, 2021. DOI: `10.1007/s10922-020-09583-4`.

[46] H. Yao, T. Mai, X. Xu, P. Zhang, M. Li, and Y. Liu, "Networkai: An intelligent network architecture for self-learning control strategies in software defined networks", *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4319–4327, 2018. DOI: `10.1109/JIOT.2018.2859480`.

[47] V. Saxena, J. Jaldén, J. E. Gonzalez, M. Bengtsson, H. Tullberg, and I. Stoica, "Contextual multi-armed bandits for link adaptation in cellular networks", in *Proceedings of the 2019 Workshop on Network Meets AI & ML*, ser. NetAI'19, Association for Computing Machinery, 2019, pp. 44–49, ISBN: 9781450368728. DOI: 10.1145/3341216.3342212.

[48] M. A. Ridwan, N. A. M. Radzi, F. Abdullah, and Y. E. Jalil, "Applications of machine learning in networking: A survey of current issues and future challenges", *IEEE Access*, vol. 9, pp. 52 523–52 556, 2021. DOI: 10.1109/ACCESS.2021.3069210.

[49] H. Jiang, Q. Li, Y. Jiang, G. Shen, R. Sinnott, C. Tian, and M. Xu, "When machine learning meets congestion control: A survey and comparison", *Computer Networks*, vol. 192, p. 108 033, 2021, ISSN: 1389-1286. DOI: 10.1016/j.comnet.2021.108033.

[50] R. Boutaba, M. A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, and O. M. Caicedo, "A comprehensive survey on machine learning for networking: Evolution, applications and research opportunities", *Journal of Internet Services and Applications*, vol. 9, no. 1, p. 16, 2018. DOI: 10.1186/s13174-018-0087-2.

[51] H. Yang, A. Alphones, Z. Xiong, D. Niyato, J. Zhao, and K. Wu, "Artificial-intelligence-enabled intelligent 6G networks", *IEEE Network*, vol. 34, no. 6, pp. 272–280, 2020. DOI: 10.1109/MNET.011.2000195.

[52] Y. Shi, K. Yang, T. Jiang, J. Zhang, and K. B. Letaief, "Communication-efficient edge AI: Algorithms and systems", *IEEE Communications Surveys Tutorials*, vol. 22, no. 4, pp. 2167–2191, 2020. DOI: 10.1109/COMST.2020.3007787.

[53] A. Bang, K. K. Kamal, P. Joshi, and K. Bhatia, “6G: The next giant leap for AI and ML”, *Procedia Computer Science*, vol. 218, pp. 310–317, 2023, International Conference on Machine Learning and Data Engineering, ISSN: 1877-0509. DOI: 10.1016/j.procs.2023.01.013.

[54] C. D. Alwis, A. Kalla, Q.-V. Pham, P. Kumar, K. Dev, W.-J. Hwang, and M. Liyanage, “Survey on 6G frontiers: Trends, applications, requirements, technologies and future research”, *IEEE Open Journal of the Communications Society*, vol. 2, pp. 836–886, 2021. DOI: 10.1109/OJCOMS.2021.3071496.

[55] Y. Liu, X. Yuan, Z. Xiong, J. Kang, X. Wang, and D. Niyato, “Federated learning for 6G communications: Challenges, methods, and future directions”, *China Communications*, vol. 17, no. 9, pp. 105–118, 2020. DOI: 10.23919/JCC.2020.09.009.

[56] N. Kato, B. Mao, F. Tang, Y. Kawamoto, and J. Liu, “Ten challenges in advancing machine learning technologies toward 6G”, *IEEE Wireless Communications*, vol. 27, no. 3, pp. 96–103, 2020. DOI: 10.1109/MWC.001.1900476.

[57] Y. Sun, J. Liu, J. Wang, Y. Cao, and N. Kato, “When machine learning meets privacy in 6G: A survey”, *IEEE Communications Surveys Tutorials*, vol. 22, no. 4, pp. 2694–2724, 2020. DOI: 10.1109/COMST.2020.3011561.

[58] M. U. Afzal, A. A. Abdellatif, M. Zubair, M. Q. Mehmood, and Y. Massoud, “Privacy and security in distributed learning: A review of challenges, solutions, and open research issues”, *IEEE Access*, vol. 11, pp. 114 562–114 581, 2023. DOI: 10.1109/ACCESS.2023.3323932.

[59] A. R. Yusof, N. I. Udzir, A. Selamat, H. Hamdan, and M. T. Abdullah, “Adaptive feature selection for denial of services (DoS) attack”, in *2017 IEEE Conference on Application, Information and Network Security*, 2017, pp. 81–84. DOI: 10.1109/AINS.2017.8270429.

[60] P. H. Mirzaee, M. Shojafar, H. Bagheri, T. H. Chan, H. Cruickshank, and R. Tafazolli, "A two-layer collaborative vehicle-edge intrusion detection system for vehicular communications", in *2021 IEEE 94th Vehicular Technology Conference*, 2021, pp. 1–6. DOI: 10.1109/VTC2021-Fall52928.2021.9625388.

[61] N. Aboueata, S. Alrasbi, A. Erbad, A. Kassler, and D. Bhamare, "Supervised machine learning techniques for efficient network intrusion detection", in *2019 28th International Conference on Computer Communication and Networks*, 2019, pp. 1–8. DOI: 10.1109/ICCCN.2019.8847179.

[62] R. Mohammadi, C. Lal, and M. Conti, "HTTPScout: A machine learning based countermeasure for HTTP flood attacks in SDN", *International Journal of Information Security*, Dec. 2022. DOI: 10.1007/s10207-022-00641-3.

[63] I. Zakariyya, H. Kalutarage, and M. O. Al-Kadri, "Resource efficient federated deep learning for IoT security monitoring", in *Attacks and Defenses for the Internet-of-Things*, W. Li, S. Furnell, and W. Meng, Eds., Springer Nature Switzerland, 2022, pp. 122–142.

[64] E. Viegas, A. O. Santin, A. França, R. Jasinski, V. A. Pedroni, and L. S. Oliveira, "Towards an energy-efficient anomaly-based intrusion detection engine for embedded systems", *IEEE Transactions on Computers*, vol. 66, no. 1, pp. 163–177, 2017. DOI: 10.1109/TC.2016.2560839.

[65] E. Gyamfi and A. Jurcut, "M-TADS: A multi-trust DoS attack detection system for mec-enabled industrial loT", in *2022 IEEE 27th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks*, 2022, pp. 166–172. DOI: 10.1109/CAMAD55695.2022.9966900.

[66] J. Jithish, B. Alangot, N. Mahalingam, and K. S. Yeo, "Distributed anomaly detection in smart grids: A federated learning-based approach", *IEEE Access*, vol. 11, pp. 7157–7179, 2023. DOI: 10.1109/ACCESS.2023.3237554.

[67] S. J. Rani, I. Ioannou, P. Nagaradjane, C. Christophorou, V. Vassiliou, S. Charan, S. Prakash, N. Parekh, and A. Pitsillides, "Detection of DDoS attacks in D2D communications using machine learning approach", *Computer Communications*, vol. 198, pp. 32–51, 2023, ISSN: 0140-3664. DOI: `10.1016/j.comcom.2022.11.013`.

[68] H. T. Truong, B. P. Ta, Q. A. Le, D. M. Nguyen, C. T. Le, H. X. Nguyen, H. T. Do, H. T. Nguyen, and K. P. Tran, "Light-weight federated learning-based anomaly detection for time-series data in industrial control systems", *Computers in Industry*, vol. 140, p. 103 692, 2022, ISSN: 0166-3615. DOI: `10.1016/j.compind.2022.103692`.

[69] X. Wang, S. Garg, H. Lin, J. Hu, G. Kaddoum, M. Jalil Piran, and M. S. Hossain, "Toward accurate anomaly detection in industrial internet of things using hierarchical federated learning", *IEEE Internet of Things Journal*, vol. 9, no. 10, pp. 7110–7119, 2022. DOI: `10.1109/JIOT.2021.3074382`.

[70] T. T. Huong, T. P. Bac, D. M. Long, T. D. Luong, N. M. Dan, L. A. Quang, L. T. Cong, B. D. Thang, and K. P. Tran, "Detecting cyberattacks using anomaly detection in industrial control systems: A federated learning approach", *Computers in Industry*, vol. 132, p. 103 509, 2021, ISSN: 0166-3615. DOI: `10.1016/j.compind.2021.103509`.

[71] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DÏoT: A federated self-learning anomaly detection system for IoT", in *2019 IEEE 39th International Conference on Distributed Computing Systems*, 2019, pp. 756–767. DOI: `10.1109/ICDCS.2019.00080`.

[72] P. Gulganwa and S. Jain, "Base station model selection using machine learning technique for wireless sensor network", *Wireless Personal Communica-*

tions, vol. 132, no. 2, pp. 1225–1239, 2023. DOI: 10.1007/s11277-023-10655-2.

[73] P. Sen and S. Waghmare, "Machine learning based intrusion detection system for real-time smart grid security", in *2021 13th IEEE PES Asia Pacific Power Energy Engineering Conference*, 2021, pp. 1–6. DOI: 10.1109/APPEEC50844.2021.9687802.

[74] A. M. Pasikhani, J. A. Clark, and P. Gope, "Adversarial RL-based IDS for evolving data environment in 6LoWPAN", *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 3831–3846, 2022. DOI: 10.1109/TIFS.2022.3214099.

[75] T. Wang, N. Huang, Y. Wu, J. Gao, and T. Q. S. Quek, "Latency-oriented secure wireless federated learning: A channel-sharing approach with artificial jamming", *IEEE Internet of Things Journal*, vol. 10, no. 11, pp. 9675–9689, 2023. DOI: 10.1109/JIOT.2023.3234422.

[76] I. Zakariyya, H. Kalutarage, and M. O. Al-Kadri, "Towards a robust, effective and resource efficient machine learning technique for IoT security monitoring", *Computers Security*, vol. 133, p. 103 388, 2023, ISSN: 0167-4048. DOI: 10.1016/j.cose.2023.103388.

[77] L. Buschlinger, R. Rieke, S. Sarda, and C. Krauß, "Decision Tree-based rule derivation for intrusion detection in safety-critical automotive systems", in *2022 30th Euromicro International Conference on Parallel, Distributed and Network-based Processing*, 2022, pp. 246–254. DOI: 10.1109/PDP55904.2022.00046.

[78] A. Ahmed, S. Hameed, M. Rafi, and Q. K. A. Mirza, "An intelligent and time-efficient DDoS identification framework for real-time enterprise networks:

SAD-F: Spark based anomaly detection framework", *IEEE Access*, vol. 8, pp. 219 483–219 502, 2020. DOI: 10.1109/ACCESS.2020.3042905.

[79] J. J. Kponyo, J. O. Agyemang, G. S. Klogo, and J. O. Boateng, "Lightweight and host-based denial of service (DoS) detection and defense mechanism for resource-constrained IoT devices", *Internet of Things*, vol. 12, p. 100 319, 2020, ISSN: 2542-6605. DOI: 10.1016/j.iot.2020.100319.

[80] R. Al Rahbani and J. Khalife, "IoT DDoS traffic detection using adaptive heuristics assisted with machine learning", in *2022 10th International Symposium on Digital Forensics and Security*, 2022, pp. 1–6. DOI: 10.1109/ISDFS55398.2022.9800786.

[81] W. He, Y. Liu, H. Yao, T. Mai, N. Zhang, and F. R. Yu, "Distributed variational bayes-based in-network security for the internet of things", *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6293–6304, 2021. DOI: 10.1109/JIOT.2020.3041656.

[82] Z. Shi, J. Li, and C. Wu, "Deepddos: Online DDoS attack detection", in *2019 IEEE Global Communications Conference*, 2019, pp. 1–6. DOI: 10.1109/GLOBECOM38437.2019.9013186.

[83] M. I. Kareem and M. N. Jasim, "DDoS attack detection using lightweight partial decision tree algorithm", in *2022 International Conference on Computer Science and Software Engineering*, 2022, pp. 362–367. DOI: 10.1109/CSASE51777.2022.9759824.

[84] Y. Katsura, A. Endo, M. Kakiuchi, I. Arai, and K. Fujikawa, "Lightweight intrusion detection using multiple entropies of traffic behavior in IoT networks", in *2022 IEEE Global Conference on Artificial Intelligence and Internet of Things*, 2022, pp. 138–145. DOI: 10.1109/GCAIoT57150.2022.10019223.

[85] R. Yaegashi, E. Takeshita, and Y. Nakayama, "Two-stage DDoS mitigation with variational auto-encoder and cyclic queuing", in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 5421–5426. DOI: 10.1109/ICC45855.2022.9838914.

[86] S. Mergendahl and J. Li, "Rapid: Robust and adaptive detection of distributed denial-of-service traffic from the internet of things", in *2020 IEEE Conference on Communications and Network Security*, 2020, pp. 1–9. DOI: 10.1109/CNS48642.2020.9162278.

[87] H. Gandhi and V. Ribeiro, "Packet batching for reducing system resource consumption for botnet detection using network traffic analysis", in *2022 14th International Conference on COMmunication Systems NETworkS*, 2022, pp. 1–6. DOI: 10.1109/COMSNETS53615.2022.9668511.

[88] A. H. Hamamoto, L. F. Carvalho, L. D. H. Sampaio, T. Abrão, and M. L. Proença, "Network anomaly detection system using genetic algorithm and fuzzy logic", *Expert Systems with Applications*, vol. 92, pp. 390–402, 2018, ISSN: 0957-4174. DOI: 10.1016/j.eswa.2017.09.013.

[89] N. Tekin, A. Acar, A. Aris, A. S. Uluagac, and V. C. Gungor, "Energy consumption of on-device machine learning models for IoT intrusion detection", *Internet of Things*, vol. 21, p. 100 670, 2023, ISSN: 2542-6605. DOI: 10.1016/j.iot.2022.100670.

[90] M. Alauthman, N. Aslam, M. Al-kasassbeh, S. Khan, A. Al-Qerem, and K.-K. Raymond Choo, "An efficient reinforcement learning-based botnet detection approach", *Journal of Network and Computer Applications*, vol. 150, p. 102 479, 2020, ISSN: 1084-8045. DOI: 10.1016/j.jnca.2019.102479.

[91] S. Smadi, N. Aslam, and L. Zhang, "Detection of online phishing email using dynamic evolving neural network based on reinforcement learning", *Decision*

Support Systems, vol. 107, pp. 88–102, 2018, ISSN: 0167-9236. DOI: 10.1016/j.dss.2018.01.001.

[92] Y. Xu, N. Chen, H. Zhang, and B. Liang, "Adaptive anomaly detection strategy based on reinforcement learning", in *Data Science*, Springer Singapore, 2018, pp. 493–504. DOI: 10.1007/978-981-13-2206-8_40.

[93] K. Sethi, E. Sai Rupesh, R. Kumar, P. Bera, and Y. Venu Madhav, "A context-aware robust intrusion detection system: A reinforcement learning-based approach", *International Journal of Information Security*, vol. 19, no. 6, pp. 657–678, 2020. DOI: 10.1007/s10207-019-00482-7.

[94] Q. Jin and L. Wang, "Intranet user-level security traffic management with deep reinforcement learning", in *2019 International Joint Conference on Neural Networks*, 2019, pp. 1–8. DOI: 10.1109/IJCNN.2019.8852447.

[95] Z. Chen, Q. Yan, H. Han, S. Wang, L. Peng, L. Wang, and B. Yang, "Machine learning based mobile malware detection using highly imbalanced network traffic", *Information Sciences*, vol. 433-434, pp. 346–364, 2018, ISSN: 0020-0255. DOI: 10.1016/j.ins.2017.04.044.

[96] Q.-V. Dang, "Studying machine learning techniques for intrusion detection systems", in *Future Data and Security Engineering*, T. K. Dang, J. Küng, M. Takizawa, and S. H. Bui, Eds., Springer International Publishing, 2019, pp. 411–426. DOI: 10.1007/978-3-030-35653-8_28.

[97] P.-J. Chuang and S.-H. Li, "Network intrusion detection using hybrid machine learning", in *2019 International Conference on Fuzzy Theory and Its Applications*, 2019, pp. 1–5. DOI: 10.1109/iFUZZY46984.2019.9066223.

[98] T. Öztürk, Z. Turgut, G. Akgün, and C. Köse, "Machine learning-based intrusion detection for SCADA systems in healthcare", *Network Modeling Analysis*

in Health Informatics and Bioinformatics, vol. 11, no. 1, p. 47, 2022. DOI: 10.1007/s13721-022-00390-2.

[99] R. Bapat, A. Mandya, X. Liu, B. Abraham, D. E. Brown, H. Kang, and M. Veeraraghavan, "Identifying malicious botnet traffic using logistic regression", in *2018 Systems and Information Engineering Design Symposium*, 2018, pp. 266–271. DOI: 10.1109/SIEDS.2018.8374749.

[100] H. Mliki, A. H. Kaceam, and L. Chaari, "Intrusion detection study and enhancement using machine learning", in *Risks and Security of Internet and Systems*, Springer International Publishing, 2020, pp. 263–278. DOI: 10.1007/978-3-030-41568-6_17.

[101] A.-R. Al-Ghuwairi, Y. Sharrab, D. Al-Fraihat, M. AlElaimat, A. Alsarhan, and A. Algarni, "Intrusion detection in cloud computing based on time series anomalies utilizing machine learning", *Journal of Cloud Computing*, vol. 12, no. 1, p. 127, 2023, ISSN: 2192-113X. DOI: 10.1186/s13677-023-00491-x.

[102] A. K. Balyan, S. Ahuja, S. K. Sharma, and U. K. Lilhore, "Machine learning-based intrusion detection system for healthcare data", in *2022 IEEE VLSI Device Circuit and System*, 2022, pp. 290–294. DOI: 10.1109/VLSIDCS53788.2022.9811465.

[103] M. Leon, T. Markovic, and S. Punnekkat, "Comparative evaluation of machine learning algorithms for network intrusion detection and attack classification", in *2022 International Joint Conference on Neural Networks*, 2022, pp. 01–08. DOI: 10.1109/IJCNN55064.2022.9892293.

[104] N. Y. Jien, M. Tahir, M. Dabbagh, Y. K. Meng, and A. Farooq, "Performance evaluation of machine learning algorithms for intrusion detection in IoT applications", in *2022 IEEE International Conference on Artificial Intelligence*

in Engineering and Technology, 2022, pp. 1–6. DOI: 10.1109/IICAIET55139.2022.9936863.

[105] I. H. Sarker, "Machine learning: Algorithms, real-world applications and research directions", *SN Computer Science*, vol. 2, no. 3, p. 160, 2021. DOI: 10.1007/s42979-021-00592-x.

[106] A. Jiang and E. F. Haratsch, "Machine learning: Enabling and enabled by advances in storage and memory systems", *IEEE BITS the Information Theory Magazine*, pp. 1–12, 2023. DOI: 10.1109/MBITS.2023.3314392.

[107] M. Gutiérrez, M. Á. Moraga, and F. García, "Analysing the energy impact of different optimisations for machine learning models", in *2022 International Conference on ICT for Sustainability*, 2022, pp. 46–52. DOI: 10.1109/ICT4S55073.2022.00016.

[108] G. S. Nikolić, B. R. Dimitrijević, T. R. Nikolić, and M. K. Stojcev, "A survey of three types of processing units: CPU, GPU and TPU", in *2022 57th International Scientific Conference on Information, Communication and Energy Systems and Technologies*, 2022, pp. 1–6. DOI: 10.1109/ICEST55168.2022.9828625.

[109] D. Comer, *Computer Networks and Internets*. Pearson/Prentice Hall, 2009, ISBN: 9780136061274. [Online]. Available: https://books.google.fi/books?id=tm-evHmOs3oC.

[110] E. Mosqueira-Rey, E. Hernández-Pereira, D. Alonso-Ríos, J. Bobes-Bascarán, and Á. Fernández-Leal, "Human-in-the-loop machine learning: A state of the art", *Artificial Intelligence Review*, vol. 56, no. 4, pp. 3005–3054, 2023. DOI: 10.1007/s10462-022-10246-w.

[111] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y.-J. A. Zhang, "The roadmap to 6G: AI empowered wireless networks", *IEEE Communications Magazine*, vol. 57, no. 8, pp. 84–90, 2019. DOI: 10.1109/MCOM.2019.1900271.

[112] S. Shen, C. Yu, K. Zhang, J. Ni, and S. Ci, "Adaptive and dynamic security in AI-empowered 6G: From an energy efficiency perspective", *IEEE Communications Standards Magazine*, vol. 5, no. 3, pp. 80–88, 2021. DOI: 10.1109/MCOMSTD.101.2000090.

[113] D. Patterson, J. Gonzalez, U. Hölzle, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D. R. So, M. Texier, and J. Dean, "The carbon footprint of machine learning training will plateau, then shrink", *Computer*, vol. 55, no. 7, pp. 18–28, 2022. DOI: 10.1109/MC.2022.3148714.

[114] U. J. Otokwala and A. Petrovski, "Ensemble common features technique for lightweight intrusion detection in industrial control system", in *2023 IEEE 6th International Conference on Industrial Cyber-Physical Systems*, 2023, pp. 1–6. DOI: 10.1109/ICPS58381.2023.10128040.

[115] S. Han and W. J. Dally, "Bandwidth-efficient deep learning", in *Proceedings of the 55th Annual Design Automation Conference*, ser. DAC '18, Association for Computing Machinery, 2018, ISBN: 9781450357005. DOI: 10.1145/3195970.3199847.

[116] V.-D. Ngo, T.-C. Vuong, T. Van Luong, and H. Tran, "Machine learning-based intrusion detection: Feature selection versus feature extraction", *Cluster Computing*, 2023, ISSN: 1573-7543. DOI: 10.1007/s10586-023-04089-5.

[117] G. P. Dubey and D. R. K. Bhujade, "Optimal feature selection for machine learning based intrusion detection system by exploiting attribute dependence", *Materials Today: Proceedings*, vol. 47, pp. 6325–6331, 2021, SI: TIME-2021, ISSN: 2214-7853. DOI: 10.1016/j.matpr.2021.04.643.

[118] P. Ghosh, Z. Alam, R. R. Sharma, and S. Phadikar, "An efficient SGM based IDS in cloud environment", *Computing*, vol. 104, no. 3, pp. 553–576, 2022. DOI: 10.1007/s00607-022-01059-4.

[119] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning", *Knowledge-Based Systems*, vol. 216, p. 106 775, 2021, ISSN: 0950-7051. DOI: 10.1016/j.knosys.2021.106775.

[120] S. Otoum, N. Guizani, and H. Mouftah, "On the feasibility of split learning, transfer learning and federated learning for preserving security in its systems", *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 7, pp. 7462–7470, 2023. DOI: 10.1109/TITS.2022.3159092.

[121] P. Vepakomma and R. Raskar, "Split learning: A resource efficient model and data parallel approach for distributed deep learning", in *Federated Learning: A Comprehensive Overview of Methods and Applications*. Springer International Publishing, 2022, pp. 439–451. DOI: 10.1007/978-3-030-96896-0_19.

[122] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy", in *2019 International Carnahan Conference on Security Technology*, 2019, pp. 1–8. DOI: 10.1109/CCST.2019.8888419.

[123] P. R. Desai, *Credit Card Fraud Detection using Federated Learning and Split Learning*, Accessed: 2023-10-10. [Online]. Available: https://github.com/PR-Desai2226/Credit-card-fraud-detection-using-Federated-Learning-and-Split-Learning.

[124] *Getting started with streamline: Introduction to streamline*, Accessed: 2024-01-04, 2024. [Online]. Available: https://developer.arm.com/documentation/

101816/0707/Getting-started-with-Streamline/Introduction-to-Streamline.

[125] ARM, *Powmon: Power monitoring technologies*, Accessed: 2024-01-04, 2024. [Online]. Available: https://www.arm.ecs.soton.ac.uk/technologies/powmon/.

[126] P. C. K. Timothy McKay, *Intel power gadget*, Accessed: 2024-01-04, 2024. [Online]. Available: https://www.intel.com/content/www/us/en/developer/articles/tool/power-gadget.html.

[127] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures", in *Proc. 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, 2009, pp. 469–480. DOI: 10.1145/1669112.1669172.

[128] V. M. Weaver, M. Johnson, K. Kasichayanula, J. Ralph, P. Luszczek, D. Terpstra, and S. Moore, "Measuring energy and power with PAPI", in *2012 41st International Conference on Parallel Processing Workshops*, 2012, pp. 262–268. DOI: 10.1109/ICPPW.2012.39.

[129] C. Hope, *Resmon - computer hope*, Accessed: 2014-01-04, 2024. [Online]. Available: https://www.computerhope.com/jargon/r/resmon.htm.

[130] *Fil: A memory profiler for python*, Accessed: 2024-01-19. [Online]. Available: https://pythonspeed.com/fil/docs/index.html.

[131] *Sciagraph: Identify speed and memory bottlenecks in python data processing workflows*, Accessed: 2024-01-19. [Online]. Available: https://www.sciagraph.com/.

[132] *Time — time access and conversions*, Accessed: 2024-01-19. [Online]. Available: https://docs.python.org/3/library/time.html.