INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Enhancing Intrusion Detection Systems using Anomaly Detection and Imbalanced Learning Techniques

Tiago Manuel Pereira dos Santos Alves

Master in Data Science

Supervisor:
PhD Ana Maria Carvalho de Almeida, Associate Professor with Aggregation,
Iscte – Instituto Universitário de Lisboa

Supervisor:
PhD Luís Miguel Martins Nunes, Associate Professor,
Iscte – Instituto Universitário de Lisboa

September 2025

[This page is intentionally left blank.]

BUSINESS
SCHOOL

TECHNOLOGY
AND ARCHITECTURE

Quantitative Methods for Management Economics

Information Science and Technology

Enhancing Intrusion Detection Systems using Anomaly Detection and Imbalanced Learning Techniques

Tiago Manuel Pereira dos Santos Alves

Master in Data Science

Supervisor:
PhD Ana Maria Carvalho de Almeida, Associate Professor with Aggregation,
Iscte – Instituto Universitário de Lisboa

Supervisor:
PhD Luís Miguel Martins Nunes, Associate Professor,
Iscte – Instituto Universitário de Lisboa

September 2025

[This page is intentionally left blank.]

To Isabel and Bernardo.
In Loving Memory of my Parents.

[This page is intentionally left blank.]

Acknowledgment

This manuscript represents a significant milestone in my academic journey. If someone had told me ten years ago that I would achieve this, I would have said it was impossible.

First, I want to express my deepest gratitude to my supervisors, Professor Ana Maria de Almeida and Professor Luís Nunes, not only for their support throughout the development of this dissertation, but also for welcoming me into the fascinating world of imbalanced learning and anomaly detection.

"If you want to walk fast, walk alone; if you want to walk far, walk together." I could not have completed this demanding journey without the extraordinary support of my family. Their role, was crucial in maintaining my mental well-being and helping me overcome the numerous challenges that arose. Sometimes I believe they have more faith in me than I have in myself. To all of them, my heartfelt gratitude.

I extend special thanks to my employer, Siemens, for believing in my potential and supporting me in achieving this milestone. To all my colleagues and managers, thank you for providing the flexibility necessary to balance work commitments with academic pursuits.

A warm acknowledgment goes to my fellow master's students who embarked on this data science journey alongside me. Your companionship made this challenging path far more enjoyable and meaningful.

Finally, I express my appreciation to all the professors who, throughout this program, demonstrated patience and expertise in teaching us the essential concepts of data science, directly contributing to our successful completion of this academic journey.

This work was partially supported by Fundação para a Ciência e a Tecnologia, I.P. (FCT) [ISTAR Projects: UIDB/04466/2023 and UIDP/04466/2023].

[This page is intentionally left blank.]

Resumo

Ataques informáticos levam sempre a ameaças às redes informáticas colocando em risco a privacidade dos utilizadores, e por vezes até à fraude financeira. Tais ataques são difíceis de mitigar devido ao seu intrínseco desequilíbrio, onde os ataques apresentam uma mínima fração de todo o tráfego da rede informática. Nesta dissertação, o objetivo consiste em melhorar os sistemas de deteção de intrusões utilizando um dataset disponibilizado publicamente, HIKARI-2021. Este dataset é naturalmente desequilibrado e serve como ponto de partida para a deteção de ataques.

A metodologia utilizada está dividida em três partes, onde é inicialmente efetuada uma exploração do dataset para compreender a sua estrutura, são posteriormente gerados sintéticos para mitigar o desequilíbrio do dataset utilizando uma rede adversaria generativa orientada para dados tabulares, e por fim, é aplicado um conjunto de vários algoritmos para deteção de anomalias, incluindo *Isolation Forest, Local Outlier Factor, One-Class SVM, DBSCAN, e Elliptic Envelope.* Também são considerados os modelos tradicionais de aprendizagem de máquina, tais como *Random Forest, Gradient Boosting, Logistic Regression, e Naïve Bayes.* Para completar a metodologia de deteção de anomalias foram também consideradas redes neuronais, tais como, *Autoencoders* e *Deep Neural Networks.*

Os resultados finais demonstram que a metodologia aplicada melhora efetivamente a deteção de anomalias em redes informáticas, onde algoritmos como Gradient Boosting e Redes Neuronais atingiram resultados acima dos 99% em métricas como *F-1 Score* e Área Debaixo da Curva. A rede adversaria generativa para dados tabulares mostrou sucesso a criar dados sintéticos e preservando as correlações existentes entre as várias colunas.

Esta investigação ajuda a contribuir para a melhoria de deteção de ataques a redes informáticas demonstrando uma elevada eficácia na mitigação do desequilíbrio dos datasets e na melhoria na deteção destes mesmos ataques.

Palavras Chave: *Deteção de Anomalias*, *Dataset Desequilibrado*, *Redes Adversária Generativa*, *Cibersegurança*, *Sistemas de Deteção de Intrusões*

[This page is intentionally left blank.]

Abstract

Computer attacks always lead to threats to computer networks, putting user privacy at risk, and sometimes even leading to financial fraud. Such attacks are difficult to mitigate due to their intrinsic imbalance, where attacks represent a minimal fraction of all computer network traffic. In this dissertation, the objective consists of improving intrusion detection systems using a publicly available dataset, HIKARI-2021. This dataset is naturally imbalanced and serves as a starting point for attack detection.

The methodology used is divided into three parts, where initially an exploration of the dataset is performed to understand its structure, synthetic data is subsequently generated to mitigate the dataset imbalance using a generative adversarial network oriented toward tabular data, and finally, a set of various algorithms for anomaly detection is applied, including Isolation Forest, Local Outlier Factor, One-Class SVM, DBSCAN, and Elliptic Envelope. Traditional machine learning models are also considered, such as Random Forest, Gradient Boosting, Logistic Regression, and Naïve Bayes. To complete the anomaly detection methodology, neural networks were also considered, such as Autoencoders and Deep Neural Networks.

The final results demonstrate that the applied methodology effectively improves anomaly detection in computer networks, where algorithms like Gradient Boosting and Neural Networks achieved results above 99% in metrics such as F-1 Score and Area Under the Curve. The generative adversarial network for tabular data showed success in creating synthetic data while preserving the existing correlations between the various columns.

This research helps contribute to improving computer network attack detection by demonstrating high effectiveness in mitigating dataset imbalance and improving the detection of these same attacks.

KEYWORDS: *Anomaly Detection*, *Imbalanced Learning*, *Generative Adversarial Networks*, *Cybersecurity*, *Intrusion Detection Systems*

[This page is intentionally left blank.]

Contents

List of Figures

List of Tables

[This page is intentionally left blank.]

List of Acronyms

ACC:	Accuracy
AD:	Anomaly Detection
AE:	Auto Encoders
AI:	Artificial Intelligence
APT:	Advanced Persistent Threats
AUC:	Area Under Curve
CDF:	Cumulative Distribution Function
CF:	Categorical Features
CIDS:	Collaborative Intrusion Detection System
CNN:	Convolutional Neural Network
CRISP-DM:	Cross-Industry Standard Process for Data Mining
CTGAN:	Conditional Tabular Generative Adversarial Networks
DBSCAN:	Density-Based Spatial Clustering of Applications with Noise
DDoS:	Distributed Denial of Service
DL:	Deep Learning
DNN:	Deep Neural Network
DoS:	Denial of Service

EDA: Exploratory Data Analysis

EE: Elliptic Envelope

F-1S: F-1 Score

GAN: Generative Adversarial Network

GB: Gradient Boosting

GPU: Graphics Processing Unit

HIDS: Host-Based Intrusion Detection Systems

IL: Imbalanced Learning

IDS: Intrusion Detection Systems

IF: Isolation Forest

IIoT: Industrial Internet of Things

IoT: Internet of Things

IR: Imbalance Ratio

IT: Information Technology

LOF: Local Outlier Factor

LSTM: Long Short-Term Memory

LR: Linear Regression

MCMC: Markov Chain Monte Carlo

ML: Machine Learning

MLP: Multi-Layer Perceptron

NB: Naïve Bayes

NF: Numerical Features

NIDS: Network Intrusion Detection Systems

OCSVM: One-Class Support Vector Machine

PCA: Principal Components Analysis

PRE: Precision

PRISMA: Preferred Reporting Items for Systematic Reviews and Meta-Analysis

REC: Recall

RF: Random Forest

ROC: Receiver Operating Characteristic

RQ: Research Questions

SDN: Software Defined Network

SDV: Synthetic Data Vault

SMOTE: Synthetic Minority Over-Sampling Technique

[This page is intentionally left blank.]

CHAPTER 1

Introduction

In this introductory Chapter, we begin by exploring the motivation behind how Anomaly Detection (AD) and Imbalanced Learning (IL) are relevant to resolving one of the most complex topics in our current Information Technology (IT) world: cybersecurity, more precisely in the realm of Intrusion Detection Systems (IDS). The goal is to provide a high-level overview of the subject, then delve into the Research Questions (RQ) that we aim to answer in this dissertation. We also want to present the methodology that was implemented for the development of the work described.

1.1. Motivation

Anomaly detection—the identification of patterns that deviate from expected behavior—has become a subject of major importance across multiple research disciplines and application domains (Ghamry et al., 2024). In cybersecurity, particularly within Network Intrusion Detection Systems (NIDS), anomaly detection techniques play a crucial role in identifying potential security threats in increasingly complex network environments.

In today's interconnected digital landscape, most computers maintain constant internet connectivity, making them vulnerable to unauthorized access and malicious activities. NIDS were developed to address this vulnerability by monitoring network activity and detecting potential attacks (Hao & Yan, 2023). These systems typically comprise three core components (Fenanir et al., 2019):

- **Monitoring**: continuously observes network traffic patterns
- **Analysis and detection**: processes monitored data to identify potential intrusions
- **Alarm**: triggers alerts when suspicious activity is detected

However, implementing effective anomaly detection in network security faces a significant challenge: class imbalance. This represents one of the most challenging tasks in data mining and machine learning (Chen et al., 2024), as real-world network traffic datasets typically contain a vast majority of benign traffic with only a small fraction representing malicious activities. Traditional Machine Learning (ML) approaches often struggle with such imbalanced distributions, leading to poor detection rates for minority attack classes.

This dissertation addresses these challenges by conducting a comprehensive comparative study of three distinct methodological approaches to intrusion detection in imbalanced network environments. First we want to set the baseline for the dataset being studied using traditional ML models as well as Deep Learning (DL) models. The second approach employs Conditional Tabular Generative Adversarial Networks (CTGAN) to generate synthetic samples of minority attack classes, thereby balancing the dataset for traditional supervised machine learning algorithms. The third approach utilizes Isolation Forest (IF), an unsupervised anomaly detection algorithm that isolates anomalies by randomly selecting features and split values. Additionally, we will delve in another AD algorithms, such as Local Outlier Factor (LOF) which identifies anomalies based on local density deviations compared to neighboring points, One-Class Support Vector Machine (OCSVM), where the goal is learning the boundary around the normal data, Density-Based Spatial Clustering of Applications with Noise (DBSCAN), a clustering algorithm where the sparse points become noise, and Elliptic Envelope (EE), which is a covariance-based algorithm.

1.2. Research Questions

To address the critical challenge of training a model for NIDS detection, this dissertation aims to answer two distinct RQ that cover different paradigms in anomaly detection and synthetic data generation.

RQ1: Comparative Effectiveness Analysis

How does CTGAN-based synthetic data augmentation compare to traditional supervised learning baselines and unsupervised anomaly detection methods for network intrusion detection in a severely imbalanced datasets?

RQ2: Anomaly Detection Algorithms

How do reconstruction-based AD methods (Autoencoder) compare to distance-based (Isolation Forest, LOF and density-based DBSCAN approaches for detecting network attacks in an imbalanced cybersecurity datasets?

1.3. Document Structure

This dissertation is structured in six Chapters, including the current Chapter. In Chapter 2, we will bring an overview of the key technologies used in this dissertation, Chapter 3, a state-of-the-art will be made where the goal is to bring to discussion what approaches were made in the area of Imbalanced Learning and Anomaly Detection when applied to Intrusion Detection Systems and what are the limitations found, the following Chapter, 4, will bring the methodology applied to the chosen dataset, the problem formulation, and setting the baseline for starting the discussion in Chapter 5, and the results of the three distinctive approaches to intrusion detection. In the final Chapter (6), we will make a summary of the findings made in this dissertation. Two appendices were also included to provide additional information to support our analysis. Appendix A, will depict an high level overview for the literature review made during the research, and Appendix B aims to provide detailed analytical results that supports the findings in the main dissertation.

CHAPTER 2

Background

The goal of this Chapter is to bring additional context about the technologies and techniques that will be used in the course of this dissertation.

2.1. Cybersecurity and Intrusion Detection Systems

Schiliro (2023) States that, in its broadest sense, **Cybersecurity** includes the coordinated deployment of people, processes, and technical resources to preserve the integrity of information-management infrastructures. Cybersecurity is frequently invoked in policy and practice, and existing regulations require the prevention, detection, isolation, and resolution of threats.

Microsoft, in its Security 101 webpage[1], says that cybersecurity is the practice of protecting critical systems, data, and networks from digital attacks, such attacks can be:

- Malware - like viruses, worms, ransomware, spyware, etc.
- Phishing and social engineering attacks
- Identity Threats
- Business email compromise
- Denial of Service (DoS) Distributed Denial of Service (DDoS) attacks
- Advanced Persistent Threats (APT)
- Insider Threats

The urgency of such a unified perspective has intensified with the pervasive integration of the Internet of Things (IoT) and data-driven analytics into virtually every industry sector.

Identifying vulnerabilities and designing counter-strategies have become a central quality-improvement imperative for organizations to guarantee confidentiality, integrity, and availability in an environment where unauthorized access can impose severe operational and economic costs. One of the available solutions to enhance vulnerability detection is the **Intrusion Detection Systems**. This technology acts as a listen-only device and its primary goal is to monitor the network traffic and report the findings to a network administrator, who will have the accountability to decide if there is a threat or not.

According to Wanda (2020), Intrusion Detection can be defined as *the act of detecting actions that attempt to compromise the confidentiality, integrity, or availability of a resource.* An IDS differs from a firewall as the latter looks outward for intrusions in order to stop them before they happen. Figure 2.1 depicts how an IDS works and how it is implemented in a network architecture.

[1]https://www.microsoft.com/en-us/security/business/security-101/what-is-cybersecurity

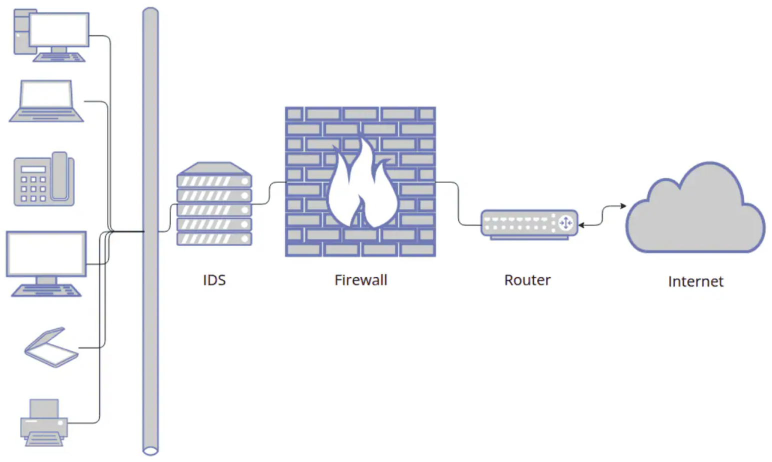


FIGURE 2.1. How an IDS works - taken from PaloAlto Networks

PaloAlto Networks[2], a global cybersecurity organization that offers an integrated platform that addresses the significant issue of maintaining digital security in an increasingly online-centric world[3] and a leader in Entry Point Protection solutions[4] states that there are 5 types of IDS: network-based, host-based, protocol-based, application protocol-based, and hybrid, however, the 2 most common IDS, and the ones considered for this dissertation, are:

- NIDS
  Coverage: Monitors entire protected network
  Deployment: Strategic infrastructure points, especially vulnerable subnets
  Monitoring: All traffic flow between network devices
  Analysis: Packet contents and metadata
- Host-Based Intrusion Detection Systems (HIDS)
  Coverage: Monitors individual computer/endpoint where installed
  Deployment: Directly on specific endpoints
  Monitoring: Both internal and external threats
  Analysis: Local traffic and logs of malicious activity

Within the contemporary cybersecurity framework, IDS serve as fundamental components for organizational network security. These systems fulfill a critical role by identifying and responding to unauthorized access attempts and malicious activities. Through continuous monitoring of network traffic patterns and systematic analysis of data anomalies, IDS provide early threat detection capabilities that enable organizations to implement proactive defensive measures and maintain network integrity.

[2]https://www.paloaltonetworks.com/
[3]https://www.gartner.com/reviews/market/network-firewalls/vendor/palo-alto-networks
[4]https://www.paloaltonetworks.com/blog/2025/07/named-a-leader-gartner-magic-quadrant/

2.2. Anomaly Detection

What is an anomaly? The primary goal is to discover rare and interesting patterns that deviate significantly from what we consider normal. Such patterns may be related to important events or issues requiring immediate attention.

Anomaly Detection is a crucial data analysis task that identifies unusual or abnormal patterns in datasets (Ahmed et al., 2016). This area has huge relevance in the area of statistics and ML, and its applications are applied in several domains, such as Network Security, but other relevant domains to take into consideration, and to name a few, are Fraud Detection, Medical Imaging, Industrial Damage Prevention, etc.

Anomaly Detection in cybersecurity, especially in IDS is one of the most prominent applications, as the systems are designed to identify malicious activities in computer networks and systems (Chandola et al., 2009). In this application domain, although anomaly detection can help to enhance the critical security protection as there are a multiple algorithmic approaches available, in Figure 2.2 we can see the most used algorithms and how they can handle an high-volume of data. On the other hand, such approach also poses some challenges, namely, high false alarm rates, the evolution of attack patterns, and the computational scalability issues (Chandola et al., 2009).

The field of AD has a rich variety of algorithmic approaches, each designed to address different types of data distributions and anomaly patterns. These can be categorized into statistical, distance-based, density-based, clustered, ensemble methods, and lastly in neural-network approaches. In Chapter 4 the chosen algorithms will be discussed.

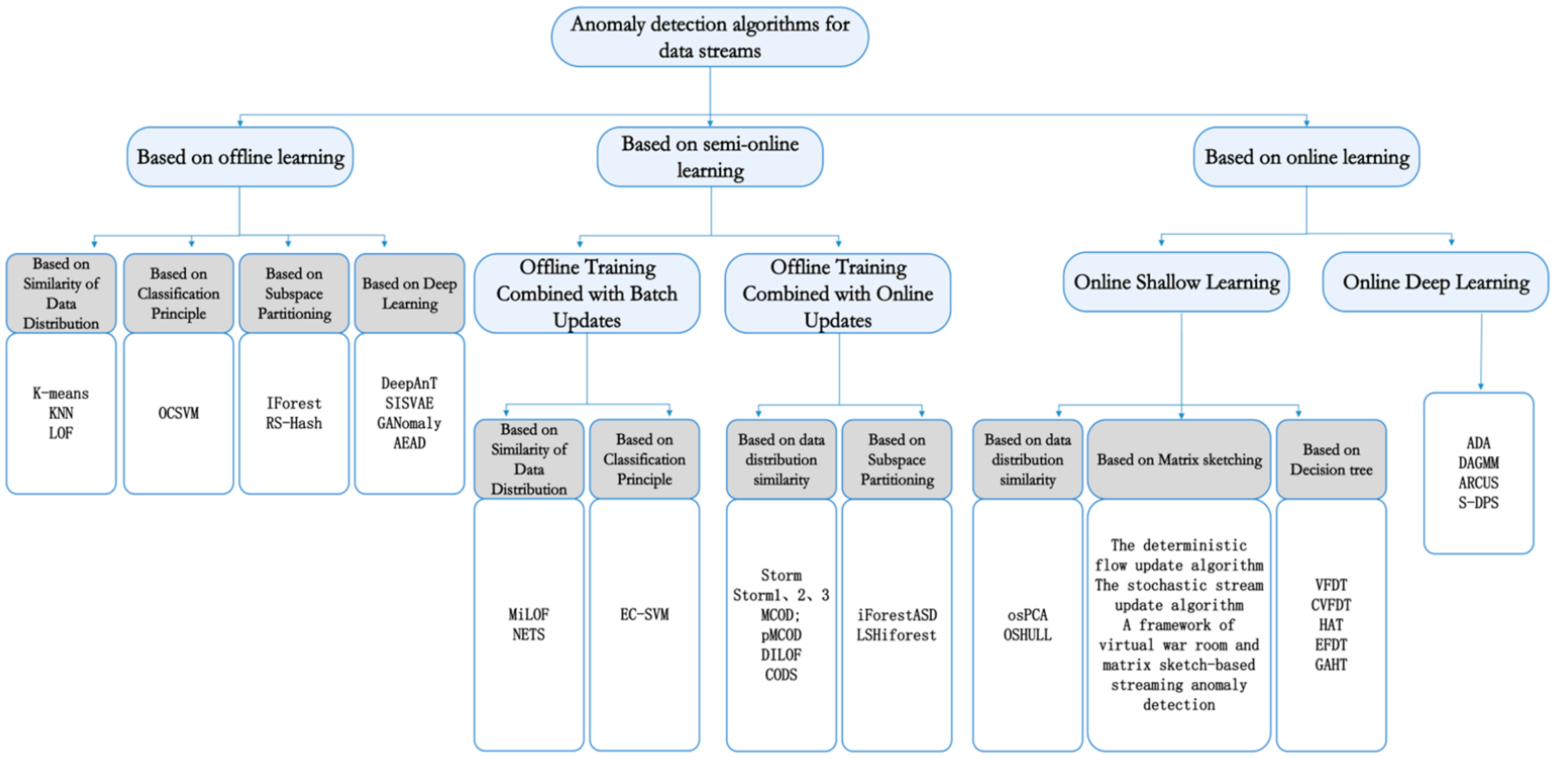

Anomaly detection algorithms for data streams
Based on offline learning
Based on semi-online learning
Based on online learning
Based on Similarity of Data Distribution
Based on Classification Principle
Based on Subspace Partitioning
Based on Deep Learning
Offline Training Combined with Batch Updates
Offline Training Combined with Online Updates
Online Shallow Learning
Online Deep Learning
K-means KNN LOF
OCSVM
IForest RS-Hash
DeepAnT SISVAE GANomaly AEAD
Based on Similarity of Data Distribution
Based on Classification Principle
Based on data distribution similarity
Based on Subspace Partitioning
Based on data distribution similarity
Based on Matrix sketching
Based on Decision tree
ADA DAGMM ARCUS S-DPS
MiLOF NETS
EC-SVM
Storm Storm1、2、3 MCOD; pMCOD DILOF CODS
iForestASD LSHiforest
osPCA OSHULL
The deterministic flow update algorithm The stochastic stream update algorithm A framework of virtual war room and matrix sketch-based streaming anomaly detection
VFDT CVFDT HAT EFDT GAHT

FIGURE 2.2. Anomaly Detection Algorithms - cortesy of (Lu et al., 2023)

2.3. Imbalanced Learning

Machine Learning and the data science landscape are evolving at a pace never seen before, and one of the most prominent challenges that data scientists and machine learning engineers face is the issue of imbalanced datasets. Such an event occurs when the distribution of classes within the dataset exhibits some skewness, whether significant or not. The skewness results in those classes having more instances than others, which, for the traditional ML algorithms, assume roughly equal distributions (Altalhan et al., 2025; Chen et al., 2024).

In a real-world scenario, the class balancing shows a behavior that goes in the opposite way, and that will create the data imbalance problem. This problem manifests across several domains and applications, mostly already mentioned in Section 2.2. The imbalanced learning problem in domains like healthcare, fraud detection, intrusion detection systems, etc. tend to exhibit a bias towards the majority class (which can be the number of healthy patients, trusted transactions, benign network traffic, etc.) often achieving high accuracy results while performing poorly on a minority-classes, which in fact those classes are the ones with the greatest interest, importance and the ones to bring value (Altalhan et al., 2025; Chen et al., 2024).

Adding focus to the bias on the majority class occurs because conventional algorithms are designed to minimize overall classification error, leading them to favor predictions that are statistically more likely to be correct based on the class frequency. Consequently, minority class instances are frequently misclassified or entirely overlooked, and this results in models that fail to capture critical patterns and relationships within the data (Chen et al., 2024).

As we can see in Figure 2.3, an imbalanced dataset shows its decision boundaries learned are tending to become increasingly biased towards the majority class (considering the decision boundaries generated), and that can lead to overlooking the data available in the minority class.

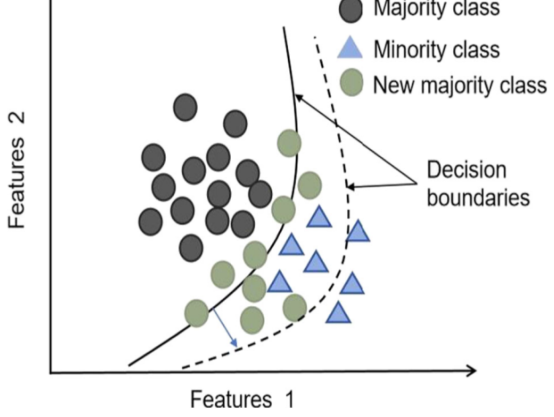


FIGURE 2.3. A binary imbalanced dataset - courtesy of Chen et al. (2024)

2.4. Generative Adversarial Networks

In this Section, we aim to explore the concept of Generative Adversarial Network (GAN) by establishing a framework and providing an overview of their groundbreaking applications in the field.

First introduced by Goodfellow et al. (2014), GAN, a framework that brought to the Artificial Intelligence (AI) realm a new application for generating synthetic data. The approach uses a two-player minmax game between a generator network (G) and a discriminator network (D). The generator learns to create synthetic data by mapping random noise to data space, while the discriminator learns to distinguish between real training data and generated data samples, making an adversarial training process. One of the used analogies to explain how GAN works is to imagine a criminal (generator[G]), who forges money, and a detective (discriminator [D]) who tries to catch him. The more authentic-looking the counterfeit money becomes, the better the detective must be at detecting it and vice versa. According to Goodfellow et al. (2014), the framework managed to eliminate the need for Markov Chain Monte Carlo (MCMC) sampling.

To summarize, the GAN framework is most straightforward to apply when the models are both multilayer perceptrons, Table 2.1, extracted from Langr and Bok (2019, p. 6) shows what the expected inputs, outputs, and goals are for each network:

Table 2.1. Generator and Discriminator Networks

	Generator	**Discriminator**
Input	A vector of random numbers	The discriminator receives input from two sources: • Real examples coming from the training dataset • Fake examples coming from the Generator
Output	Fake examples that strive to be as convincing as possible	Predicted probability that the input example is real
Goal	Generates fake data that is indistinguishable from members of the training dataset	Distinguish between the fake examples coming from the Generator and the real examples coming from the training dataset

The architecture of a GAN can be depicted in Figure 2.4 where the generator maps the vector of random numbers (z) in a latent space to create a high-dimensional $G(z)$, meaning, fake samples. The discriminator D, is expected to separate x (real samples), from $G(z)$

Since its first implementation by Goodfellow et al. (2014), where the MNIST[5], Toronto Face Database, and CIFAR-10[6] datasets were used, GANs have evolved and started to be applied in other scenarios, including the one being studied in this dissertation, cybersecurity. In cybersecurity, GANs can have a double purpose: They can be used to enhance the IDS by creating synthetic data to improve the baseline behavior model, or they can be used in a malicious way to generate adversarial attacks to evade IDS (Zhao et al., 2024). In our research, the articles reviewed consider both scenarios; however, there is a bigger focus on enhancing the IDS to create synthetic data to improve the baseline model.

[5]https://www.tensorflow.org/datasets/catalog/mnist
[6]https://www.cs.toronto.edu/ kriz/cifar.html

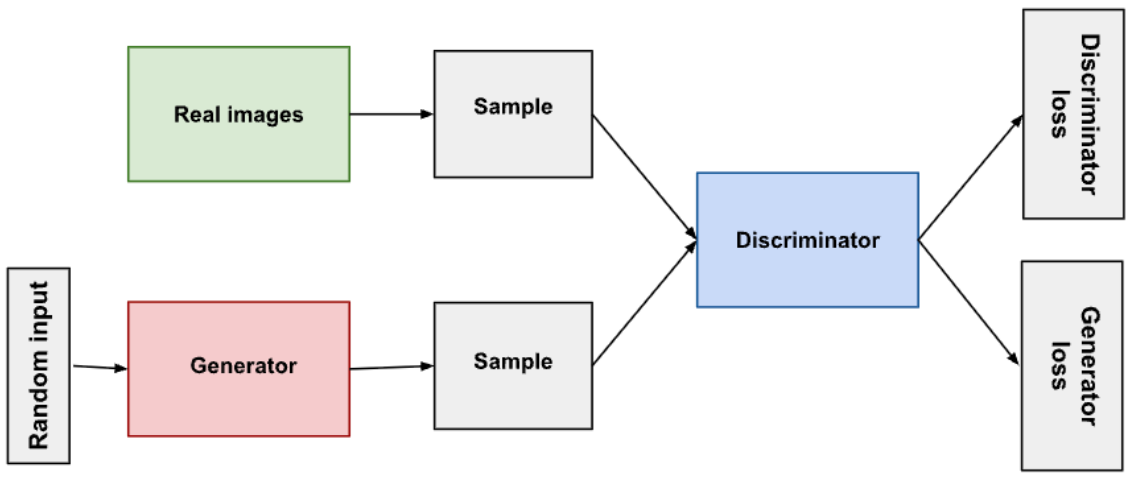


FIGURE 2.4. GAN Architecture

CHAPTER 3

Literature Review

The main purpose of this Chapter is to identify and review the state-of-the-art of the study that will be conducted. Reviewing what has already been studied allows us to get more scientific knowledge that exists in the area that is being investigated, in this case, anomaly detection and imbalanced learning.

This Chapter is structured into three distinct Sections. The first Section establishes the eligibility criteria and outlines the methodology employed for the literature search, including query development and acceptance criteria that support the research framework. The second Section presents a comprehensive state-of-the-art review of the most relevant articles identified for this study. Finally, the third Section examines existing research gaps and discusses how this dissertation addresses these limitations to advance the field.

3.1. Eligibility Criteria

This Section will describe the research method used and its eligibility criteria. The goal is to explain what research was conducted to answer the Research Questions asked in Section 1.2.

The Preferred Reporting Items for Systematic Reviews and Meta-Analysis (PRISMA) is an elaborate set of guidelines developed to help researchers conduct and report high-quality systematic reviews and meta-analyses. Think of it as a road map that guides researchers through every step of their review process, ensuring transparency and reproducibility. PRISMA, at its core, consists of a 27-item checklist and a four-phase flow diagram that helps researchers trace their search and selection process. The checklist covers the essential elements to be included in the review rationale, systematic search strategies, selection of studies, data collection methods, and synthesis of results. Just as a chef follows a recipe to ensure consistent results, following PRISMA guidelines enables researchers to produce systematic reviews that meet established quality standards. Since its introduction in 2009, updated in 2020, PRISMA has become the gold standard in systematic review reporting, helping readers understand exactly how the researchers arrived at their conclusions and enabling other scientists to replicate or build upon their work Page et al. (2021).

Since the PRISMA methodology was chosen for this literature review. The recommended flow diagram has been used to enhance transparency and a proper reproduction in systematic reviews and meta-analyses. It offers a diagrammatic way of representing the process of the literature search, clearly depicting how researchers move from initial database searches to the final selection of studies for review. The flowchart in Figure 3.1

supports the literature review by providing a standardized method for documenting the number of records identified, screened, and included or excluded at each stage, along with the reasons for exclusions.

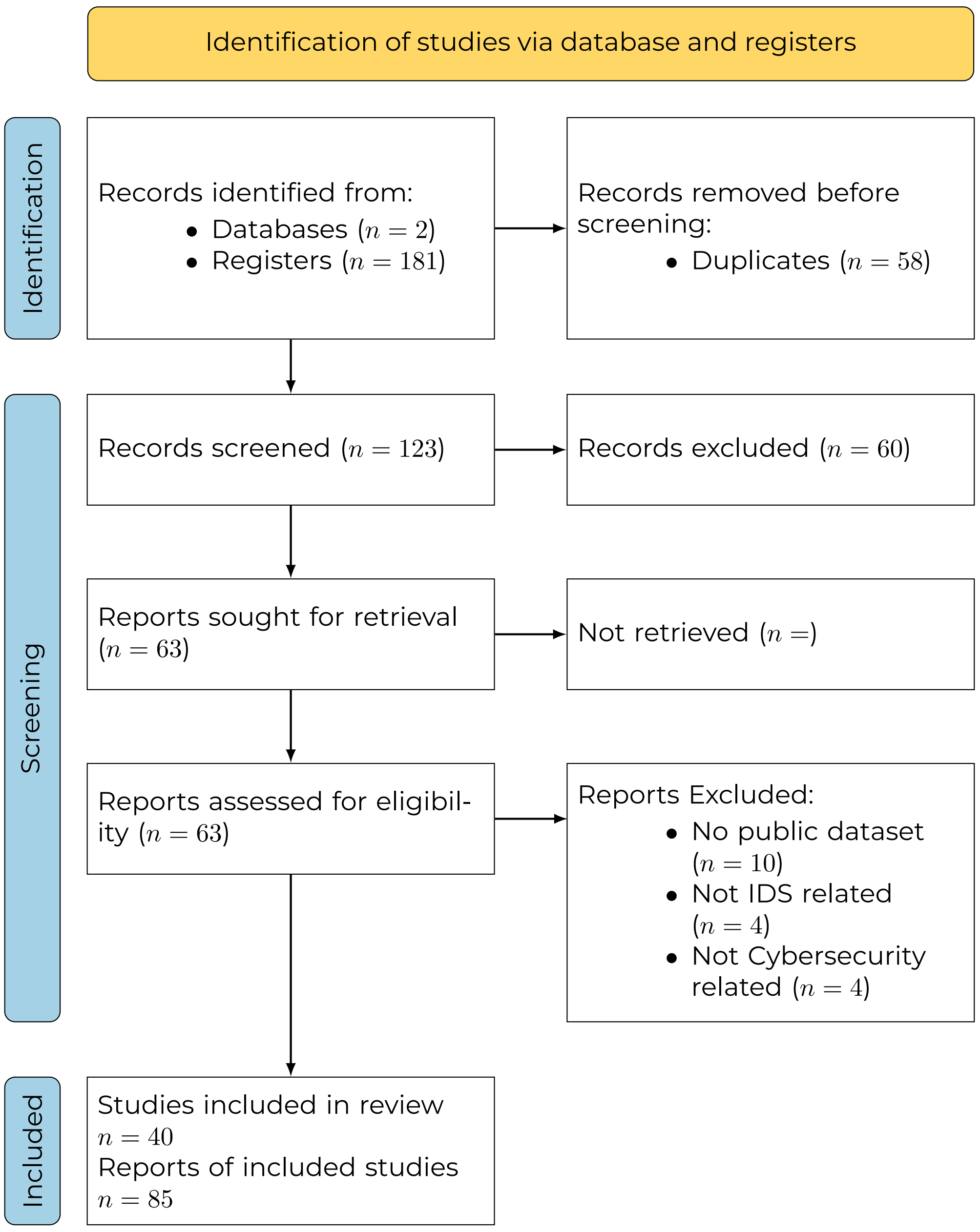

Identification of studies via database and registers
Identification
Records identified from:
• Databases ($n = 2$)
• Registers ($n = 181$)
Records removed before screening:
• Duplicates ($n = 58$)
Screening
Records screened ($n = 123$)
Records excluded ($n = 60$)
Reports sought for retrieval ($n = 63$)
Not retrieved ($n =$)
Reports assessed for eligibility ($n = 63$)
Reports Excluded:
• No public dataset ($n = 10$)
• Not IDS related ($n = 4$)
• Not Cybersecurity related ($n = 4$)
Included
Studies included in review $n = 40$
Reports of included studies $n = 85$

FIGURE 3.1. PRISMA Workflow

We developed a specific search query to answer our research questions, which will be discussed further in this Section, and Zotero[1] was used to manage the search results and remove duplicates, and initially found 181 articles to screen. We then excluded, 58 articles which were duplicated in both databases. After removing the duplicates, we removed 60 articles that did not meet our criteria for acceptance (see Table 3.2) or did not answer our research questions listed.

[1]https://www.zotero.org/

This rigorous screening left us with 63 articles (one of the articles was retraced) that were relevant and eligible for full-text review, which formed the basis of our systematic review.
We subsequently excluded 18 studies based on predetermined criteria: ten lacked publicly accessible datasets, four fell outside the intrusion detection systems (IDS) domain, and four addressed non-cybersecurity topics.

After this thorough analysis, a total of 45 articles ensured that we have a comprehensive and relevant literature base to answer our research questions. Apart from these 45 articles, it was also considered 40 previous studies which supported the knowledge earned during the development of this dissertation.

3.1.1. Search and Selection

The following databases for academic research and publication were used: (1) SCOPUS and (2) Web of Science.

These two leading academic research databases provide access to scholarly literature worldwide. While Scopus, launched by Elsevier in 2004 (“Scopus | Abstract and citation database | Celebrating 20 years of innovation”, n.d.), covers more than twenty eight thousand and three hundred (28,300) journals across various disciplines, Web of Science, maintained by Clarivate Analytics (“Web of Science Core Collection | Clarivate”, n.d.), includes over twenty two thousand (22,000) scientific publications. Both databases serve as comprehensive sources for academic research, but they have distinct characteristics in terms of coverage and functionality.

Having documents indexed by either database offers significant benefits for researchers. Articles published in Scopus or Web of Science gain increased visibility and recognition among peers. This visibility and recognition may become a leverage when applying for research grants, fellowships, and scholarships.

To ensure that we were able to get a wide set of results for the research questions made, a set of conditions were defined in order to get a proper research query, Table 3.1 shows how the query was developed:

Table 3.1. Query Development

Concept	“Anomaly Detection” AND “Imbalanced Learning” AND "NIDS"
Population	"Network Intrusion Detection" OR IDS OR Cybersecurity
Context	"Outlier Detection" OR "Class Imbalance" OR "Abnormal Behavior" OR "GAN" OR "SMOTE" OR "Oversampling"
Document Type	Article
Limit To	Articles from 2017 to 2024 (7 Years)
Language	English

Since the questions we want to answer are related to the effectiveness of synthetic data as well as class imbalance mitigation, our research began in 2016. Still, the first result shown was in 2021, according to the criteria described in Table 3.1, which was written by Novaes et al. (2021) where it was proposed a system detection and defense against

DDoS attacks in Software Defined Network (SDN) environments. Although if we go back to where the GAN were implemented for the first time, we can find the Generative Adversarial Nets (Goodfellow et al., 2014) as mention in the previous Section 2.4, where an adversarial networks framework that operates through a competitive training paradigm wherein a generative model attempts to produce synthetic samples that are indistinguishable from authentic data. To ensure we were able to find the studies related to synthetic data creation, it was also considered additional criteria to the context, such as Synthetic Minority Over-Sampling Technique (SMOTE), first discussed in Chawla et al. (2011) and *Oversampling*, to bring a broader result to our investigation.

The outcome queries were the following:

- SCOPUS

```
 TITLE-ABS-KEY ( ( "anomaly detection" OR "outlier detection"
OR "abnormal behavior" ) AND ( "imbalanced learning" OR "
imbalanced datasets" OR "imbalanced learning" OR gan OR "class
 imbalance" OR "data imbalance" OR smote OR oversampling ) AND
 ( "network intrusion detection" OR ids OR cybersecurity ) )
AND PUBYEAR > 2016 AND PUBYEAR < 2026 AND ( LIMIT-TO ( DOCTYPE
 , "ar" ) ) AND ( LIMIT-TO ( LANGUAGE , "English" ) )
```

- Web of Science

```
 (‘‘anomaly detection’’ OR ‘‘outlier detection’’ OR ‘‘abnormal
 behavior’’) AND (‘‘imbalanced datasets’’ OR ‘‘class imbalance
’’ OR ‘‘data imbalance’’ OR ‘‘imbalanced learning’’ OR GAN OR
SMOTE OR oversampling) AND (‘‘network intrusion detection’’ OR
 IDS OR cybersecurity) (All Fields) and 2025 or 2024 or 2023
or 2022 or 2021 or 2020 or 2019 or 2018 or 2017 (Publication
Years) and Article (Document Types)
```

3.1.2. The Criteria for Acceptance

To ensure the review does not compromise the researcher bias, the Table 3.2 shows the inclusion criteria and its justification to consider an article to answer the research questions.

TABLE 3.2. Criteria Acceptance

Item	Criteria	Justification
01	Must be Cybersecurity related	Cybersecurity problem to resolve
02	Dataset must be accessible	Must be able to be replicated using a public dataset
03	Must use IDS, NIDS or HIDS	Ensure that network attacks are studied
04	Must solve Anomalies	Ensure that an Anomaly can be found
05	Must use Imbalanced datasets	Ensure the datasets have a minority class

Based on the criteria above, we have come up with the following overview of the articles: For **item 01** and **item 03** in Figure 3.2 we can understand that Intrusion Detection

Systems and its alternatives, such as NIDS, and Collaborative Intrusion Detection System (CIDS) have the biggest part of the articles studied, however we can see that IoT is starting to gain some traction in the cybersecurity area, especially in the Industrial Internet of Things (IIoT) where the volume of data is huge and the risk of that data become a target for a cybersecurity attack poses a great risk (Lu et al., 2024).

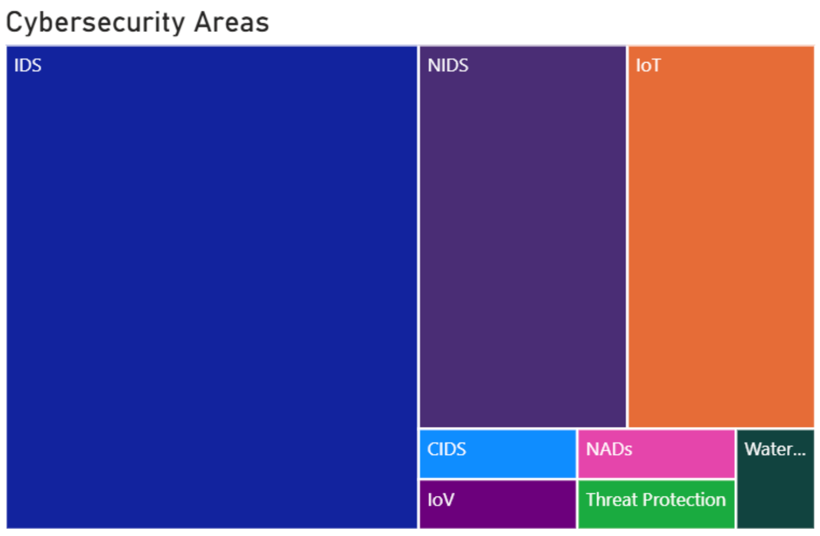


FIGURE 3.2. Treemap of Cybersecurity Areas Studied

The **item 02** and **item 05** from the acceptance criteria state that a dataset must be accessible; this is a recurring challenge, as the cybersecurity community should help the machine learning community to tackle one of the key obstacles, which is the lack of appropriate datasets Amit et al. (2018). Although this is true because companies tend not to share their network activity data, when it is cybersecurity-related, this topic becomes more prominent. During our research, we have found that most of the articles reviewed used a public dataset; some exceptions were found where the authors used a proprietary dataset. For the articles considered for this study, the Figure 3.3 shows the top ten datasets used, we have also added the publication year to understand what is the trend.

We also have learned that NSL-KDD and KDDcup99 share the same structure, but KDDcup99 is an improved version of the NSL-KDD where the redundant records in training and testing sets were removed, and in that way, the classifier model used will have a reduced chance to become biased toward the frequent records (Tavallaee et al., 2009). About the UNSW-NB15[2], this dataset is kindly made available by the University of New South Wales, and it has nine types of attacks, with the goal of reflecting the real network traffic and modern low-footprint attacks.

To complete the analysis of the Table 3.2, the biggest percentage of the articles considered for this study aim to solve anomalies, exceptions made for surveys that were considered as they are relevant to understand the state-of-the-art of anomaly detection.

[2]https://research.unsw.edu.au/projects/unsw-nb15-dataset

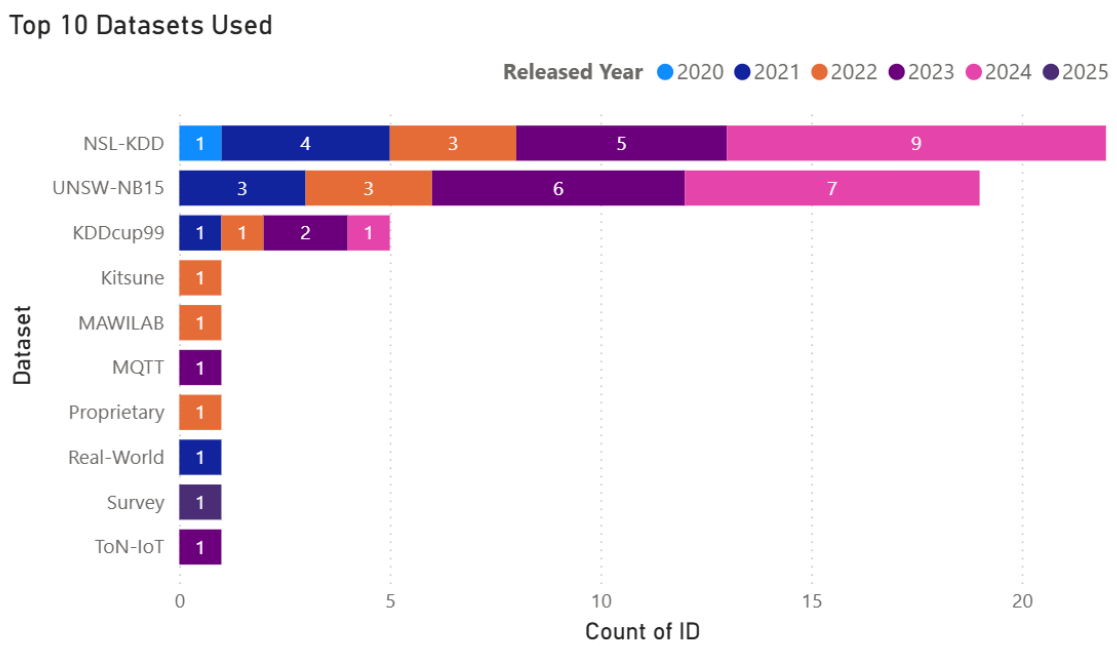


FIGURE 3.3. Top 10 datasets used per year of publication

3.2. State of the Art

Although all the 45 articles stated in Section 3.1 brought relevant knowledge for our study and to understand what is being discussed in the AD and IL area, there were 14 articles that stood out to help us develop our approach that will be discussed in Chapter 4.

We will start with Li et al. (2024), where the authors propose a GAN-based anomaly detection model for NIDS using GAN and convolutional enconders / decoders. The authors wer able to improve the performance compared to the traditional ML algorithms in four distinct datasets. The major limitation was the parameter tuning, also it was found that the dataset size will increase the computational complexity. Also in the area of GANs, Benaddi et al. (2022) has proposed a DRL-GAN where combining a Distributional Reinforcement Learning with Wasserstein GAN for anomaly detection in IIoT. The authors were able to achieve an accuracy above the 99% either in binary or multi-class classification othe DS2OS dataset. Although the results were very promising, it is relevant to know that high computational requirements will be needed for the training phase.

Duy et al. (2021) on the other hand proposed a framework where the goal was to deceive an IDS using a GAN while preserving the functional features of the attack traffic. A Wasserstein GAN with function preserving algorithms and a black-box organization was used and the outcomes were very positive on two distinct datasets where the framework managed to deceive 9 different ML / DL algorithms while maintaining the functional behavior of network traffic. The need to use extensive computational resources and potential detection by advanced IDS were the limitations found during the study.

In the healthcare security domain, Ashok et al., 2024 introduced GANA-AO, a hybrid deep learning model combining Generative Adversarial Networks and Autoencoders enhanced with Adam optimization for Social Medical Public Healthcare systems. The authors achieved impressive results with 98.33% accuracy and 98.67% True Positive Rate on the IoT-23 dataset, significantly outperforming baseline Convolutional Neural Network (CNN) and Autoencoder models. However, the study was limited to single dataset evaluation, raising concerns about scalability to larger healthcare networks.

Rookard and Khojandi (2024) conducted a comprehensive comparison of unsupervised machine learning approaches, including a novel GAN-based method with RBM feature engineering across traditional and software-defined networking environments. Their study revealed that Isolation Forest and Deep Belief Networks generally outperformed other models, while their GAN approach showed competitive performance specifically on the CIC-IDS2017 dataset. The main limitation was inconsistent GAN performance across different datasets and challenges in model generalization to zero-day attacks.

For 5G-enabled metaverse applications, Ding et al. (2022) proposed a hybrid intrusion detection model combining Wasserstein GAN with gradient penalty, Deep Autoencoder, and Random Forest. Their approach achieved exceptional accuracy rates of 99.8% for binary classification and 99.6% for multi-class classification on the InSDN dataset. The model successfully addressed dataset imbalance through GAN-generated synthetic samples and improved training efficiency via DAE dimensionality reduction. Nevertheless, the evaluation was limited to simulated SDN environments, and performance may degrade with real-world network complexity.

In the area of adversarial deep learning for DDoS detection, Novaes et al. (2021) developed a GAN-based framework operating in near-real-time with 1-second analysis intervals. Their system achieved superior performance compared to CNN, Long Short-Term Memory (LSTM), and Multi-Layer Perceptron (MLP) approaches, with 99.78% accuracy on emulated SDN data and 94.3% accuracy with 95.54% F1-score on the CICDDoS 2019 dataset. The major limitation was performance variability with different attack complexity levels and extensive parameter tuning requirements for optimal convergence.

Fu et al. (2023) presented a Wasserstein GAN-based approach with encoder-generator-discriminator architecture, incorporating spectral normalization and gradient penalty for stable training. Their GANAD model achieved superior performance across three datasets, with 97.49% precision and 97.55% F1-score on KDDCUP99, demonstrating effectiveness through novel anomaly scoring combining reconstruction and discriminator losses. However, computational complexity increases significantly with dataset size, and GAN training instability remains a concern.

In the automated machine learning domain, Xu et al. (2023) proposed a comprehensive framework using SMOTE for class balancing and mutual information for feature selection, followed by AutoML for optimal algorithm selection. Their approach achieved remarkable 99.79% accuracy on the KDDcup99 dataset, with AutoML automatically selecting Bagged Ensemble of Decision Trees as the optimal classifier. The framework successfully improved detection rates for minority attack classes, though evaluation was limited to a single dataset, raising generalization concerns.

Sun et al. (2022) addressed multi-class classification challenges by combining Borderline SMOTE with information gain ratio-based feature selection. Their framework demonstrated optimal performance with Decision Tree algorithm using 62 features, achieving improved detection rates particularly for minority Web Attack classes. The study revealed that performance gains plateau beyond 62 features, indicating effective feature subset selection, though performance varied significantly across different attack types.

For addressing class imbalance and overlap issues, Zoghi and Serpen (2024) developed an ensemble classifier integrating Balanced Bagging, XGBoost, and Random Forest with Hellinger Distance Decision Trees. Their novel threshold-adjustment algorithms achieved 97.80% binary classification accuracy with 98.26% sensitivity for attack detection, and multi-class classification reached up to 99.73% accuracy. The approach effectively handled class overlap problems, but computational complexity and hyperparameter dependency present implementation challenges.

Alfoudi et al. (2022) introduced Enhanced DBSCAN (EDBSCAN) with cosine similarity for dynamic intrusion detection, incorporating an evolving process based on inter-cluster distance measures. Their approach achieved a mean silhouette score of 0.87, substantially outperforming standard DBSCAN, with accuracy rates of 86.82%, 79.10%, and 90.03% on KDDTest, KDDTest-21, and UNSW-NB15 datasets respectively. Significant improvements were observed for minority classes, though manual parameter selection and computational complexity of the evolving clustering process remain limitations.

Xu et al. (2021) proposed a 5-layer autoencoder architecture with optimized preprocessing methodology, incorporating 95th percentile outlier removal and Mean Absolute Error reconstruction loss. Their model achieved 90.61% accuracy and 92.26% F1-score on the NSL-KDD dataset, demonstrating that data preprocessing has the most significant impact on performance compared to model architecture. However, the study was limited to the NSL-KDD dataset, and the extensive preprocessing requirements may not generalize effectively to other network environments.

Finally, Dlamini and Fahim (2021) introduced a Conditional Generative Adversarial Network-based Data Generative Model incorporating KL-divergence for minority class improvement in anomaly detection. Their approach significantly enhanced classifier performance on minority classes, achieving superior weighted F1-scores compared to SMOTE and traditional oversampling methods on both NSL-KDD and UNSW-NB15 datasets. Despite consistent improvements across different classifier families, GAN training instability and computational complexity for large datasets remain significant challenges for practical deployment.

Table 3.3 depicts an high-level overview for the articles discussed in this Section, and for an overall high-level overview of the articles reviewed, you can consult Table A.1 available in appendix A

Table 3.3. Summary of the state of the art

Author & Year	**Methodology**	**Algorithms**	**Key findings**	**Limitations**
Duy et al. (2021)	Adversarial sample generation with function preservation	Wasserstein GAN (WGAN)	Successfully bypassed 9 ML/DL algorithms while preserving traffic functionality	Limited to specific attacks, ethical concerns
Novaes et al. (2021)	Adversarial training with GAN	GAN with adversarial training	Superior to CNN/LSTM/MLP, 95.54% F1-score on CICDDoS 2019	Performance varies with attack complexity
Xu et al. (2021)	5-layer autoencoder with optimized preprocessing	5-layer AE + MAE + 95th Percentile Outlier Removal	90.61% accuracy, 92.26% F1-score on NSL-KDD	Limited dataset evaluation, preprocessing dependency
Dlamini and Fahim (2021)	Conditional GAN for minority class generation	cGAN + KL-divergence + Synthetic Data Generation	Superior minority class performance, consistent improvements	GAN training instability, computational complexity
Benaddi et al. (2022)	DRL combined with WGAN-GP	Distributional RL + WGAN-GP	99.05% accuracy in binary classification, superior detection of specific attack types	High computational resources, single dataset evaluation
Ding et al. (2022)	Hybrid GAN-DAE-RF model	WGAN + Deep Autoencoder + Random Forest	99.8% binary, 99.6% multi-class accuracy on InSDN dataset	Simulated environment only, dataset dependency
Sun et al. (2022)	Borderline SMOTE with feature selection	Borderline SMOTE + Information Gain Ratio + DT/RF/KNN	Optimal performance with 62 features, improved minority class detection	Performance variation across attack types
Alfoudi et al. (2022)	Enhanced DBSCAN with cosine similarity	EDBSCAN + Cosine Similarity + Fuzzy Clustering	0.87 mean silhouette, substantial minority class improvement	Manual parameter selection, computational complexity
Fu et al. (2023)	WGAN with encoder and novel scoring	WGAN + Spectral Normalization + Gradient Penalty	Outperformed existing GAN methods on 3 datasets	Training instability, computational complexity
Xu et al. (2023)	Data-driven AutoML approach	SMOTE + MI + AutoML (Bagged Ensemble)	99.79% accuracy with automated algorithm selection	Limited dataset evaluation, generalization challenges
Li et al. (2024)	Flow-based GAN with adversarial learning	FlowGANAnomaly (GAN + Flow Encoder)	Significantly improved performance on 4 datasets with uniform feature space mapping	High computational complexity, parameter tuning requirements
Ashok et al. (2024)	Hybrid GAN-Autoencoder with Adam optimization	GANA-AO (GAN + Autoencoder + Adam)	98.33% accuracy, 98.67% TPR on IoT-23 dataset	Single dataset evaluation, scalability concerns
Rookard and Khojandi (2024)	Comparative unsupervised ML analysis	GAN, DBN, RBM, OCSVM, I-Forest	I-Forest and DBN outperformed, GAN competitive on CIC-IDS2017	Inconsistent GAN performance, generalization challenges

Continued on next page

Table 3.3 (continued)

Author & Year	Methodology	Algorithms	Key findings	Limitations
Zoghi and Serpen (2024)	Ensemble with threshold adjustment	BB + XGBoost + RF-HDDT + Novel Algorithms	97.80% binary accuracy, 98.26% attack sensitivity	Computational complexity, hyperparameter dependency

3.3. Research Gaps

The comprehensive analysis of the 45 articles examined in Section 3.1 reveals the current state of AD, IL, and IDS, with particular emphasis on the emerging application of Generative Adversarial Networks for addressing data imbalance challenges. While studies such as Li et al. (2024) demonstrated significant improvements using GAN-based approaches and Dlamini and Fahim (2021) showed superior performance on minority classes, the literature analysis reveals several critical limitations that existing approaches fail to address adequately. The identified research gaps, derived from systematic evaluation of current methodologies and their reported limitations including computational complexity, dataset dependency, and generalization challenges, provide the foundation for the present dissertation's objectives and methodological framework.

The application of Generative Adversarial Networks for synthetic data generation in intrusion detection systems remains significantly under-explored despite their demonstrated potential across various domains. While Dlamini and Fahim (2021) introduced a Conditional GAN-based approach achieving superior performance on minority classes, and Ding et al. (2022) demonstrated exceptional results with WGAN integration, most existing studies continue to rely on traditional oversampling techniques such as SMOTE rather than leveraging the advanced capabilities of GANs. The limited exploration of GAN-generated synthetic intrusion data represents a critical research gap, particularly concerning their effectiveness in creating high-fidelity samples for rare attack types that are typically underrepresented in training datasets. Current approaches fail to fully investigate how GAN-generated data impacts the robustness and generalization capabilities of AD models when deployed against previously unseen threats. This limitation is particularly concerning given the evolving nature of cybersecurity threats, where the ability to synthesize realistic attack patterns could significantly enhance detection capabilities. The effectiveness of using synthetic data for both training and evaluation of IDS models, especially regarding their capacity to generalize to zero-day attacks, remains largely unexplored and represents a fundamental challenge in advancing anomaly detection methodologies.

This research will address the aforementioned gap, by developing a framework where CTGAN applications in IDS anomaly detection. The proposed methodology aims to achieve the two critical dimensions of the research gap: (1) rigorous synthetic data quality assessment, and (2) compare the performance against anomaly detection algorithms.

Furthermore, this research contributes to advancing the anomaly detection methodologies by exploring the CTGAN synthetic data augmentation and unsupervised learning in IDS context.

[This page is intentionally left blank.]

CHAPTER 4

Materials and Methods

Cross-Industry Standard Process for Data Mining (CRISP-DM) proposes a domain and tool-agnostic, hierarchical methodology for data mining that structures projects into six iterative phases (Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment), to improve planning, communication, and quality across stakeholders (Wirth & Hipp, 2000).

Following this framework, the goal of this Chapter is to examine the materials used in this dissertation. We will describe the dataset, ML models used, strategies to generate synthetic data, and evaluate the performance of our findings. The Business understanding was already discussed in Chapter 2.

To perform all the steps mentioned in this Subsection, we have used a HP Mobile Workstation Fury 15 G7 powered by an Intel(R) Core(TM) i7-10850H processor, 32Gb RAM and a graphics card made by Nvidia, the model of the graphics card is Quadro T2000 with Max-Q Design. Not only for sanity check but also to run the tests that needed more computational power, we also have used Google Colab[1]. The programming language used in the local environment is python version 3.12.3[2], running in Jupyter Notebook[3] on a WSL virtual environment on Ubuntu 24.04[4]. Host Operating System is Windows 11 pro 22631. Google Colab versioning may be different.

4.1. Data Understanding

One of the major challenges in the cybersecurity domain is to find a good dataset, as already discussed in Section 3.1.2. For this dissertation, we will be using a recently published dataset, HIKARI-2021 (Ferriyan et al., 2021).

HIKARI-2021 is a modern IDS dataset mixing real background traffic with **labeled**, encrypted synthetic attacks over HTTPS/TLS 1.2 to reflect today's encryption-dominant networks. It specifies clear content and process requirements (complete capture, payload handling, selective anonymization via Crypto-PAn, ground-truth, reproducibility) and publishes procedures to regenerate data. Traffic was captured from CMS targets (WordPress, Joomla, Drupal), with benign user-like browsing via Selenium and attacks including browser brute force and XML-RPC brute force. Features (88 total) are Zeek-derived, as we can see in Table 4.1, which depicts the features extracted. This dataset

[1]https://colab.google/
[2]https://www.python.org/
[3]https://jupyter.org/
[4]https://learn.microsoft.com/en-us/windows/wsl/about

also extends the CICIDS-2017's 80 flow features with identifiers and TLS-relevant statistics, enabling encrypted-traffic analysis without decryption. Chosen here to provide up-to-date, encryption-aware context data with transparent labeling and a realistic synthetic/real mix for robust IDS evaluation.

The authors of the dataset have several versions available; we have chosen to use the most recent version, 1.4.0 (Ferriyan et al., 2022). This dataset's last update was April 15, 2022, and based on our literature review, the dataset was not used in any of the articles eligible for acceptance, which became an excellent candidate to study how imbalanced learning and anomaly detection can be put in practice.

TABLE 4.1. List of features

ID	Feature	ID	Feature	ID	Feature
0	Unnamed: 0				
1	uid	30	flow_CWR_flag_count	59	flow_iat.avg
2	originh	31	flow_ECE_flag_count	60	flow_iat.std
3	originp	32	fwd_pkts_payload.min	61	payload_bytes_per_second
4	responh	33	fwd_pkts_payload.max	62	fwd_subflow_pkts
5	responp	34	fwd_pkts_payload.tot	63	bwd_subflow_pkts
6	flow_duration	35	fwd_pkts_payload.avg	64	fwd_subflow_bytes
7	fwd_pkts_tot	36	fwd_pkts_payload.std	65	bwd_subflow_bytes
8	bwd_pkts_tot	37	bwd_pkts_payload.min	66	fwd_bulk_bytes
9	fwd_data_pkts_tot	38	bwd_pkts_payload.max	67	bwd_bulk_bytes
10	bwd_data_pkts_tot	39	bwd_pkts_payload.tot	68	fwd_bulk_packets
11	fwd_pkts_per_sec	40	bwd_pkts_payload.avg	69	bwd_bulk_packets
12	bwd_pkts_per_sec	41	bwd_pkts_payload.std	70	fwd_bulk_rate
13	flow_pkts_per_sec	42	flow_pkts_payload.min	71	bwd_bulk_rate
14	down_up_ratio	43	flow_pkts_payload.max	72	active.min
15	fwd_header_size_tot	44	flow_pkts_payload.tot	73	active.max
16	fwd_header_size_min	45	flow_pkts_payload.avg	74	active.tot
17	fwd_header_size_max	46	flow_pkts_payload.std	75	active.avg
18	bwd_header_size_tot	47	fwd_iat.min	76	active.std
19	bwd_header_size_min	48	fwd_iat.max	77	idle.min
20	bwd_header_size_max	49	fwd_iat.tot	78	idle.max
21	flow_FIN_flag_count	50	fwd_iat.avg	79	idle.tot
22	flow_SYN_flag_count	51	fwd_iat.std	80	idle.avg
23	flow_RST_flag_count	52	bwd_iat.min	81	idle.std
24	fwd_PSH_flag_count	53	bwd_iat.max	82	fwd_init_window_size
25	bwd_PSH_flag_count	54	bwd_iat.tot	83	bwd_init_window_size
26	flow_ACK_flag_count	55	bwd_iat.avg	84	fwd_last_window_size
27	fwd_URG_flag_count	56	bwd_iat.std	85	bwd_last_window_size
28	bwd_URG_flag_count	57	flow_iat.min	86	attack_category
29	flow_CWR_flag_count	58	flow_iat.max	87	Label

To get additional information about the dataset, an Exploratory Data Analysis (EDA) has been applied. EDA[5] is a method to analyze datasets and to summarize their main characteristics. It supports discovering patterns, spotting anomalies, and assessing model assumptions before formal analysis.

Figure 4.1 depicts the results of dataset EDA, this is a binary distribution within the dataset, this means that data is split into two types of labels: 0, for benign traffic, which refers to when the network was not attacked, and 1, for malicious traffic, which refers to an attack or attempt to attack the network. The values are shown by volume and proportion. Figure 4.2 depicts the same approach but with a proportional overview.

[5]https://www.ibm.com/think/topics/exploratory-data-analysis

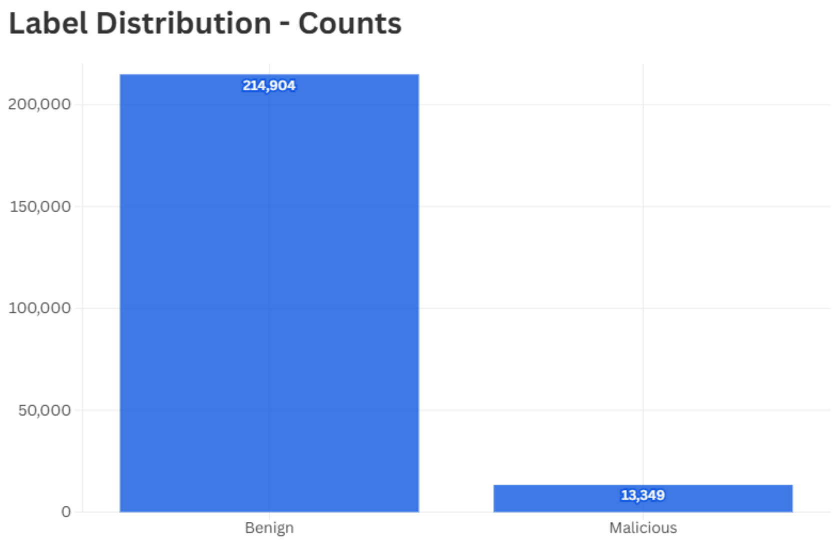


FIGURE 4.1. EDA for binary classification - Absolute values

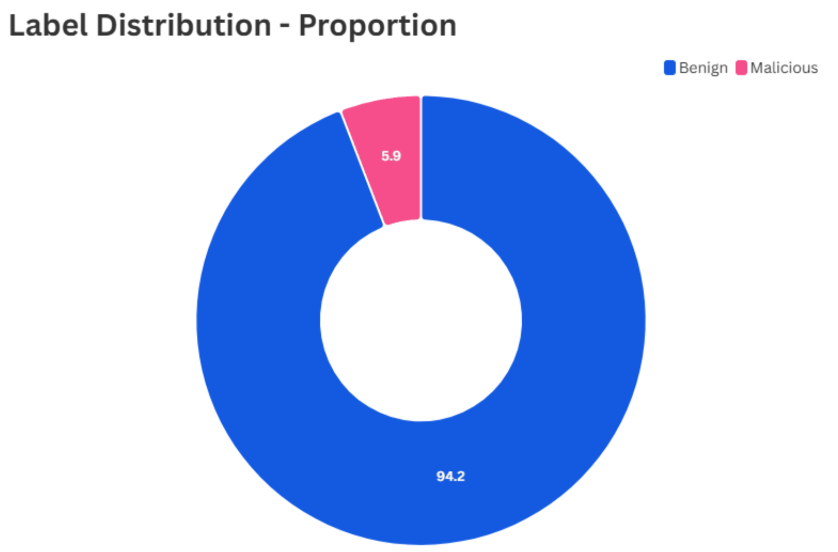


FIGURE 4.2. EDA for binary classification - Proportion

Figure 4.3 show the attack categories available in the dataset by volume and by logarithmic scale. There are three types of malicious traffic (Bruteforce, Bruteforce-XML, and XMRIGCC CryptoMiner), and one category for benign traffic.

By looking at the results of EDA in Figure 4.2, we can understand that the HIKARI-2021 dataset version is highly imbalanced, either on a binary level (Label) or in a multi-class level (Attack Categories). To assess such an imbalance, Table 4.2 shows the distribution analysis by Label and by Attack Category.

To calculate the Imbalance Ratio (IR) (Kulkarni et al., 2021), we applied the equation 4.1 where we consider the *Majority Class* as the Benign traffic (0) and the *Minority Class* the Malicious traffic (1).

$$IR = \frac{MajorityClass}{MinorityClass} \tag{4.1}$$

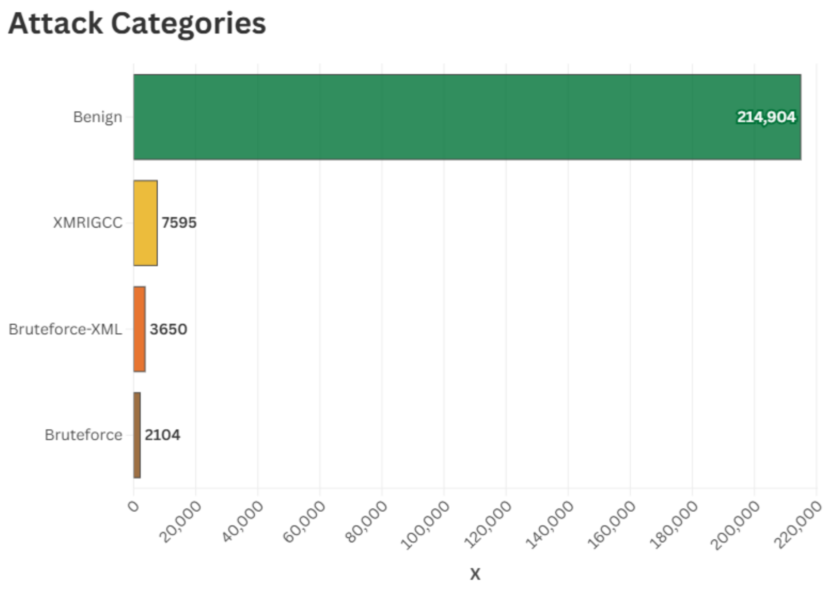


FIGURE 4.3. EDA for a multi-class classification

TABLE 4.2. Variable Distribution

Variables	**Type**	**No. of samples**	**Percentage**
Label	Benign (0)	214.904	94,15%
	Malicious (1)	13.349	5,85%
	Benign	214.904	94,15%
	XMRIGCC	7.595	3,33%
Attack Category	Bruteforce-XML	3.650	1,60%
	Bruteforce	2.104	0,92%

The IR calculated is $16, 10 : 1$, which means that there are 16,10 benign events (Majority Class) for 1 malicious event (Minority Class). Considering such high IR, we can confirm that the HIKARI-2021 dataset is an eligible dataset to answer our research questions as mentioned in the Section 1.2.

To further understand the dataset characteristics and identify potential data quality issues, we conducted a correlation analysis and examined features with extreme values. Figure B.1 presents the correlation matrix of the 50 most important features.

Before delving into the data preparation to start modeling the dataset to meet our goals, we felt we needed to have a better understanding of the dataset; therefore, we have split the features in Table 4.1 into five different categories: temporal, volume, protocol, behavioral, and statistical. We have excluded for this categorization the features that contains IDs, and the label features (already reviewed in Table 4.2) The Figure 4.4 shows the distribution for each category (4.4a) and the number of features for the same categories (4.4b) the Table 4.3 shows a summary of the dataset including the features that shows outliers above the 5% mark.

Table 4.3. Feature Analysis Summary

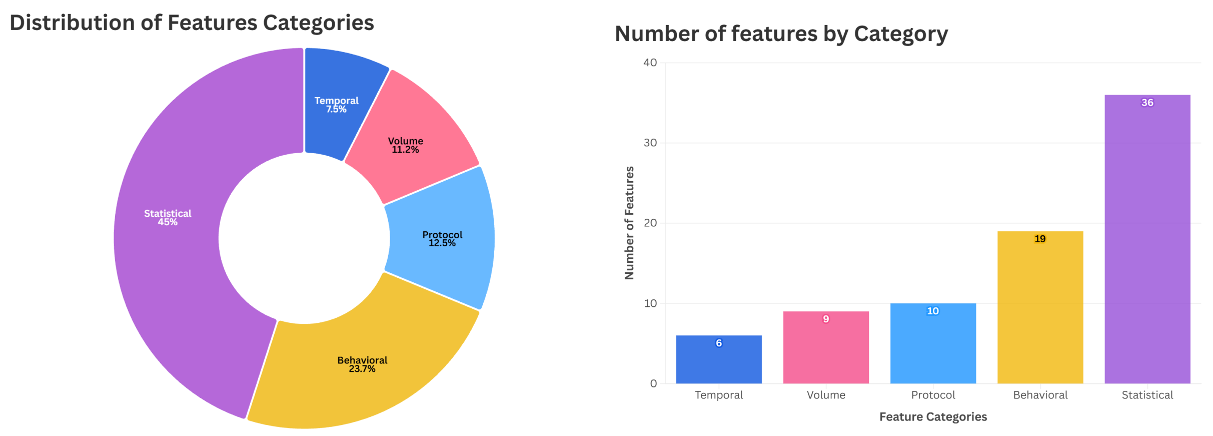
Metric	Count
Total Features	80
Numerical Features	79
Temporal Features	6
Volume Features	9
Protocol Features	10
Behavioral Features	19
Statistical Features	36
Features with Outliers (>5%)	37
Zero Minimum Features	79
High Variance Features	68

4.2. Data Preparation

The purpose of this Section is to prepare the dataset for the modeling phase, which will be discussed in Section 4.3. Our research, as discussed in Section 1.1, will be split into three phases:

(1) Run a baseline - The goal is to understand how the Traditional ML models and DL models perform in finding anomalies in an imbalanced dataset
(2) Apply GAN to generate malicious synthetic data - After balancing the dataset, the goal is to measure the performance of finding anomalies compared with the baseline
(3) Use Anomaly Detection algorithms - The goal is to understand if Anomaly Detection algorithms are more efficient than the baseline and GAN.

To ensure the dataset is ready to fulfill the requirements, we have searched for features with missing values, and it turned out that no missing values were found in the dataset.



(A) Distribution of Feature Categories

(B) Number of Features by Category

Figure 4.4. Feature Categorization

We also have considered the Numerical Features (NF) and removed the Categorical Features (CF), the categorical features were removed from dataset, as they are one-hot encoded in CTGAN, creating sparse vectors. That way, the discriminator can easily distinguish between real and synthetic data by detecting the sparsity patterns in these encoded vectors, making it harder for the generator to produce realistic synthetic data. The CF descriptions are depicted in Table 4.4.

Table 4.4. Categorical Features

Feature	**Description**
Unnamed: 0	Index column, usually created during CSV export. Not a real feature
uid	Unique identifier for each flow/session
originp	Source port (origin port)
Response	Destination port (responding port)
originh	Source IP address (origin host)
Response	Destination IP address (responding host)

4.3. Modeling

In this Section, we will cover the model plan and techniques used to help answer the RQs presented in Section 1.2. This Section will be split into three Subsections to match the CRISP-DM listed in Section 4.2.

4.3.1. Baseline

Baseline models provide a critical reference point for evaluating advanced ML algorithms, establishing minimum performance thresholds that complex models must surpass to justify their implementation costs. They enable systematic assessment of model improvements, preventing unnecessary complexity while guiding efficient resource allocation in the development process. By offering clear performance benchmarks, baselines help to identify meaningful advancements and streamline iterative model refinement.

According to our literature review in Chapter 3, we established our baseline using four traditional ML models and one DL model. Figure 4.5 depicts the top 10 most frequently used models within the accepted articles. Rather than selecting the top 5 models listed, we aimed for a balanced choice, resulting in the models shown in Table 4.5.

Figure 4.5. Top 10 Models per year of article

The selected baseline models represent diverse algorithmic approaches: ensemble methods (RF, GB), probabilistic approaches (NB), linear methods (LR), and deep learning (DNN). This diversity ensures comprehensive performance comparison across different model families, from interpretable linear models to complex neural architectures, providing robust benchmarks for subsequent imbalanced learning techniques.

All models were implemented with fixed random seeds (seed=42) to ensure reproducibility. The traditional ML models utilized scikit-learn's[6] default hyperparameters as

[6]https://scikit-learn.org/stable/

Table 4.5. Machine Learning and Deep Learning Models Configuration Summary

Model Name	Description	Hyperparameters
Random Forest (RF)	Ensemble learning method using multiple decision trees for classification	Default Random Forest parameters
Gradient Boosting (GB)	Ensemble method that builds models sequentially to correct prediction errors	Default Gradient Boosting parameters
Linear Regression (LR)	Linear model for binary and multi'class classification using logistic function	Default Logistic Regression parameters
Naïve Bayes (NB)	Probabilistic classifier based on Bayes' theorem with strong independence assumptions	Default Gaussian Naive Bayes parameters
Deep Neural Network (DNN)	Multi-layered neural network trained with PyTorch framework	3 Hidden Layers: 128, 64, 32 neurons Activation: ReLU with 30% Dropout Output: Linear layer with CrossEntropyLoss Optimizer: Adam (lr=0.001) Batch size: 256 Early stopping: patience=10 epochs Random seed: 42 for reproducibility

baseline configurations, while the DNN employed PyTorch's[7] automatic differentiation and Graphics Processing Unit (GPU) acceleration when available. Training incorporated comprehensive monitoring including loss tracking, early stopping mechanisms, and performance visualization to prevent overfitting and ensure optimal convergence.

The dataset was split into a train-test with a ratio of 80:20, which means that 80% of the dataset will be used for testing and the remaining 20% will be used to measure the performance of the models in finding network attacks. Figure 4.6 depicts the class distribution by dataset split and the percentage of minority class distributed per split, Table 4.6 shows a consolidated summary of the dataset split.

Table 4.6. Data Components Summary

Component	Size	Structure	Notes
Training Data	182.602 rows	88 columns	10.672 minority (4,7% of total)
Testing Data	45.651 rows	88 columns	2,677 minority (1,2% of total)
Training Majority Class	75,3% of total	88 columns	Dominant Class
Tota Minority Class	5,8% of total	88 columns	Target: 1
Numerical Features	Continuous Variables	88 columns	
Categorical Features	Discrete Variables	88 columns	
Processed Data	10.672 rows	82 columns	Training Data - Malicious

4.3.2. Synthetic Data Generation

One of the techniques to mitigate a dataset imbalance consists in oversampling the minority class, the most known approach is SMOTE (Chawla et al., 2011), and SMOTE principle consists in replicating existing examples until the number is equal to the majority class. Kim (2024) used this technique to improve detection rates without labeled attack

[7]https://pytorch.org/

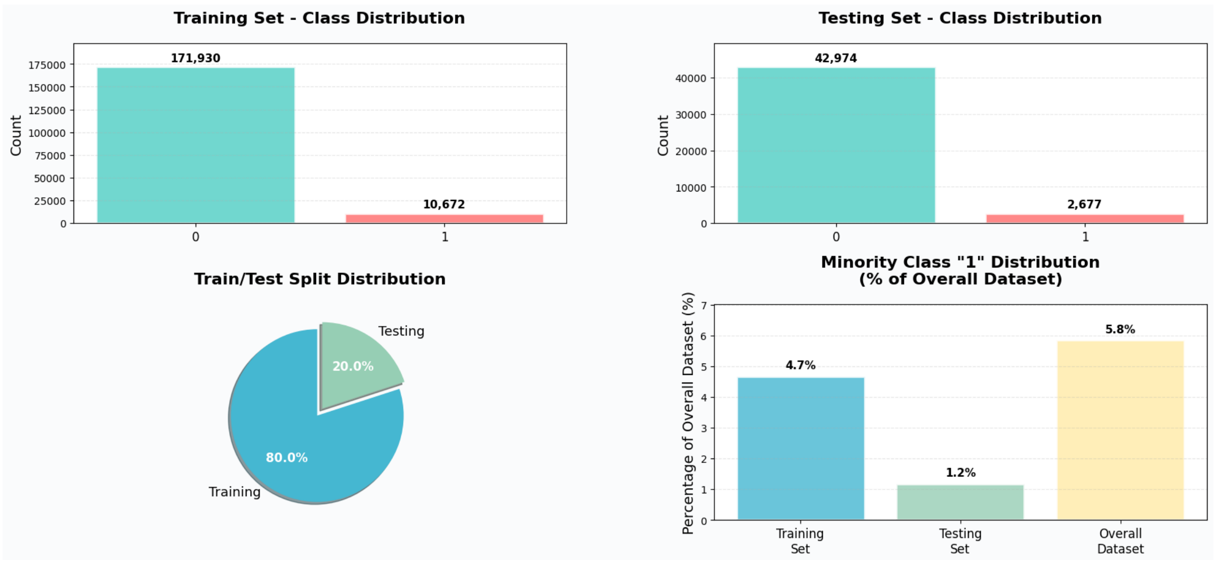


FIGURE 4.6. Dataset Train / Test Split approach

data, Sun et al. (2022) had a primary focus on the borderline SMOTE where a combination of random under-sampling (another imbalance dataset mitigation technique), and Borderline SMOTE allowed to achive an accuracy of 99% on the CIC-IDS2017 dataset. Hacilar et al., 2024, has applied three different oversampling approaches, SMOTE, SMOTETomek and ADASYN and combined with an Autoencoder for feature extraction. The results were very promising for Intrusion Detection datasets.

Generative Adversarial Networks are gaining traction in synthetic data generation, as already discussed in Section 2.4 we have covered the structure of a GAN and how the "game" between the Generator and Discriminator is dealt to ensure the fake data is generated. GANs have been evolving since their first implementation in 2014, Ding et al. (2022) and Fu et al. (2023) have used a Wasserstein-GAN with Gradient Penalty (WGAN-GP) to find the Nash equilibrium training considering the Wasserstein distance[8]. Li et al. (2023) for the Abnormal Traffic Detection has used Denoising Autoencoder plus an Hybrid GAN (DAE-GAN), where the DAEs worked as pseudoanomalies generator the GAN discriminator provided a binary result.

Standard GANs also brings value to our research and some good approaches were made, such as, Chen et al., 2022, have used a standard GAN to test the evasion success rate. Novaes et al., 2021 managed to improve the detection of DDoS attacks in a SDN.

4.3.3. Data Generation and Evaluation

We generated synthetic malicious data for dataset balancing, using CTGAN proposed by Xu et al. (2019). This approach was specifically designed to address the unique challenges of synthetic tabular data generation that traditional GANs struggle with, including mixed data types (continuous and discrete columns), non-Gaussian multimodal distributions, and severe class imbalance in categorical features. CTGAN introduces several key

[8]https://arize.com/glossary/wasserstein-distance/

innovations: mode-specific normalization using Variational Gaussian Mixture Models to handle multimodal continuous distributions, and a conditional generator with training-by-sampling that uses log-frequency sampling to ensure adequate representation of minority classes during training. The authors demonstrated that CTGAN outperformed Bayesian network baselines on 87.5% of evaluated datasets, making it particularly effective for highly imbalanced datasets where traditional methods fail to adequately represent minority classes—a critical requirement for malicious data augmentation in cybersecurity applications.

Figure 4.7 illustrates the synthetic data generation methodology employed for the HIKARI-2021 dataset. The objective of this approach is to generate synthetic malicious samples to address class imbalance within the training dataset.

The proposed methodology encompasses the following components:

Synthetic Data Generation Strategy: The generation process is exclusively applied to malicious samples, representing the minority class within the dataset. This approach operates under the assumption that benign samples maintain sufficient representational quality and therefore do not require synthetic augmentation.

Theoretical Foundation: The methodology employs class separation techniques to isolate minority and majority classes within the training data, thereby enabling targeted synthetic data generation. This approach is theoretically justified as minority classes often exhibit distinct distributional characteristics that require specialized handling to prevent being overwhelmed by majority class patterns during the synthesis process. The separation of classes mitigates the risk of distributional dominance, wherein prevalent benign patterns may suppress the distinctive features inherent in malicious samples.

Test Data Preservation: The test dataset remains unmodified throughout the synthetic data generation process. This preservation ensures the integrity of performance evaluation metrics for the machine learning and deep learning models established in the baseline methodology (Section 4.3.1).

This targeted synthetic data generation strategy addresses the inherent class imbalance characteristic of cybersecurity datasets, where malicious instances typically constitute a significantly smaller proportion compared to benign samples, thereby enabling more robust model evaluation and performance assessment.

To leverage what Xu et al. (2019) proposed, we have used the Synthetic Data Vault (SDV) python library (Patki et al., 2016), which already offers the CTGAN Synthesizer(“CTGANSynthesizer | Synthetic Data Vault”, 2025) that allows us to create synthetic data. This library also offers the ability to evaluate the data quality of the synthetic data created. To ensure the quality of synthetic data, we have also developed a comprehensive data comparison between the generated data and the real data; this will be discussed in Chapter 5. The design of the CTGAN is depicted in the Table 4.7.

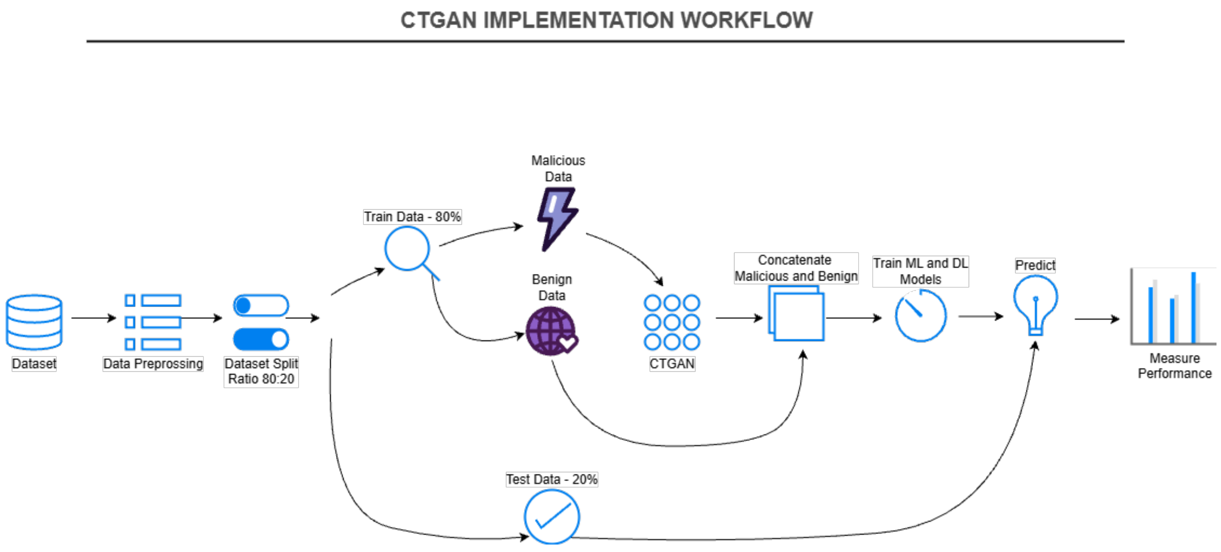


FIGURE 4.7. Synthetic Data Generation using CTGAN

TABLE 4.7. CTGAN Model Summary

Parameter	Value	Description
Epochs	50 to 10000	Number of training epochs for the GAN model
Cuda	True	Enables GPU acceleration for faster testing
enforce_rounding	True	Ensures integer columns remain integers in synthetic data
verbose	True	Enables detailed training progress output

Principal Components Analysis (PCA)[9] was considered as a strategy to implement before the CTGAN generation, where its main goal is to establish a comprehensive baseline understanding of the original dataset's intrinsic structure (see Figure B.2 in Appendix B). Implementing this strategy, we managed to set the quantitative benchmarks against which synthetic data quality can be measured objectively.

Understand the quality of the synthetic data is a relevant task to understand if the generated keeps the same structure as the original data and there is no major deviations, to ensure we were able to assess the synthetic data quality, we have created a post-generation evaluation framework where we measure the the statistical validation using the Kolmogorov-Smirnov tests to assess the similarity between the original and synthetic data (see Figure B.3 in Appendix B). Also, we used the PCA benchmark generated for the original dataset state and generated a comparison step with the goal of quantifying the structural preservation in a reduced-dimensional space. The analysis of the principal component was also considered to assess whether synthetic data maintains a similar variance pattern.

To support all the steps mentioned above, a set of visualizations and exploratory analysis was put in place. Cumulative Distribution Function (CDF) comparisons were used to reveal the distributional differences across the entire feature range (see Figure B.4

[9]https://www.ibm.com/think/topics/principal-component-analysis

in Appendix B). Using CDF after the synthetic data generation was beneficial to assess the distributional shifts, variance differences, and the overall similarity.

Inspired by the Python library table-evaluator[10], A cumulative sum (cumsum) analysis has also been considered for the post-synthetic data generation. This approach, instead of using the traditional absolute values, uses both axes as percentages to enable a direct comparison across the dataset features, size, and scales (see Figure B.5 in Appendix B). To improve the analysis, an area between curves has also been implemented using a trapezoidal integration, providing a scalar metric that allows for quantifying distributional similarity.

By combining both approaches (CDF and cumsum), we were able to enhance the statistical rigor, as well as we can provide a comprehensive distributional assessment, establish quantitative benchmarks, and address the fundamental challenges in synthetic data quality while maintaining computational efficiency.

4.3.4. Post-Generation EDA for Synthetic Data Quality Assessment

To complete the methodology made in Section 4.3.3 and to ensure the synthetic data quality matches the expectations, another EDA has been applied but this time focused only on the synthetic data . With this approach we managed to perform a comprehensive data preview and structural comparison by comparing the generated data with the original data by retrieving a set of samples (first five rows; last five rows; and a set of 10 random rows). A statistical summary integration has also been considered having in mind the ability to profile the distribution of the dataset by measuring the central tendency, dispersion metrics and range characteristics. This EDA can be seen in the code developed to support this study which was made available through a Github[11] repository and the link is made available in our Appendix B.

To support the EDA, a dashboard with structural and distributional analysis was put in place, also a correlation structure comparison has been included. This correlation structure comparison addresses a critical aspect of synthetic data fidelity and such analysis is essential to understand the statistical dependencies, the correlation structure and the feature interactions. To complete the correlation analysis, we have considered a R-squared (R^2) correlation metric due to its simplicity and ability to provide a bounded interpretation, intuitive meaning, and comparative benchmarking. We will delve in the outcome of the correlation comparison in the Chapter 5.

4.3.5. Anomaly Detection Algorithms

In this last part of this Section, we will apply a set of unsupervised, density-based and covariance-based Anomaly Detection algorithms. As discussed in Section 2.2 our goal is to find potential attacks in a computer network.

[10]https://pypi.org/project/table-evaluator/
[11]https://github.com/

A complete, unsupervised-to-semisupervised approach will be made keeping the same strategy as previously mentioned. Like the baseline in Section 4.3.1 and 4.3.3 the dataset will be split into 80% for training and the remaining 20% for testing (see Figure 4.6), just like the baseline, the categorical features were removed (see Table 4.4). Five classical Anomaly Detection algorithms were used and to complete our analysis an Autoencoder using pytorch was also considered for this research.

During our Literature Review, we have learned the most common anomaly detection algorithms used were the IF, LOF and DBSCAN.

Kim (2024) used IF combined with LOF in an hybrid unsupervised learning with autoencoders, Fernando et al. (2024) brought the the computer efficiency on the unsupervised algorithms and the IF shown a superior computational efficiency compared to other models, LOF included. Alfoudi et al. (2022) proposed a new evolving process based on cluster distance measures for DBSCAN, and managed to improve the minority class detection.

We have decided also to include the OCSVM and EE, as OCSVM uses the normal data (benign traffic in our case), and identifies any deviations from the baseline created. EE on the other hand creates an elliptical boundary around the normal network traffic with any traffic patterns falling outside this ellipse are flagged as potential attacks.

Table 4.8 describes the algorithms used and their definitions and Table 4.9 depicts the optimization strategy used to find the best hyperparameter setup

TABLE 4.8. Implemented Anomaly-Detection Algorithms

IF: Isolation Forest, LOF: Local Outlier Factor, OCSVM: One-Class Support Vector Machine, DBSCAN: Density-Based Spacial Clustering of Applications with Noise, EE: Elliptic Envelope

Algorithm	**Core Principle**
IF	Randomly isolates data points; anomalies need fewer splits for isolation
LOF	Measures local density; points in significantly lower-density regions are flagged as outliers
OCSVM	Learns a hypersurface enclosing normal data; points outside are considered anomalies
DBSCAN	Forms clusters in dense regions; points not assigned to any cluster (*noise*) are anomalies
EE	Fits a robust covariance ellipse around normal data; samples outside the ellipse are anomalies
Autoencoder Neural Network	Learns to reconstruct normal patterns; high reconstruction error signals an anomaly

TABLE 4.9. Optimization Strategy Used Across Algorithms

Aspect	**Specification**
Multiple contamination scenarios	Contamination rates tested: 0.01, 0.03, 0.05, 0.10, 0.15
Cross-validation approach	Each algorithm evaluated across all contamination settings; best model selected via internal metrics
GPU acceleration	CUML libraries employed when available to speed up training and inference

Following the approach established in Section 4.3.1, we employed the Python library scikit-learn for implementing traditional anomaly detection algorithms (acIF, LOF, OCSVM, DBSCAN, and EE) and utilized PyTorch for developing the Auto Encoders (AE) algorithm.

Neloy and Turgeon (2024) demonstrates that autoencoders have emerged as particularly powerful tools for anomaly detection due to their ability to learn complex latent representations and reconstruct normal samples with high fidelity. This generative modeling approach addresses a critical challenge inherent in anomaly detection: the severe class imbalance between anomalous and normal instances, where anomalies typically represent less than 1–5% of real-world datasets. Unlike supervised methods that require extensive labeled anomalous examples, autoencoders operate under the unsupervised paradigm, learning to compress normal data into lower-dimensional latent representations while preserving essential structural information.

The fundamental principle underlying AE-based anomaly detection centers on reconstruction error as the anomaly score. During training, the autoencoder learns to minimize reconstruction loss (typically mean squared error or binary cross-entropy) exclusively on normal samples, effectively modeling the underlying data distribution of the inlier class. At inference time, normal samples are reconstructed with low error due to their similarity to training patterns, while anomalies (arising from different generative mechanisms) exhibit significantly higher reconstruction errors. This reconstruction-based scoring mechanism provides a natural, threshold-based classification approach: samples exceeding a predefined reconstruction error threshold are classified as anomalous, with higher scores indicating greater likelihood of anomalous behavior.

The architectural flexibility of autoencoders enables adaptation to specific domain requirements and data characteristics. Neloy and Turgeon (2024) systematically evaluated 11 autoencoder variants, demonstrating that Self-Adversarial Variational Autoencoders (adVAE) achieved superior performance with Receiver Operating Characteristic (ROC)-Area Under Curve (AUC) scores of 0.93 on MNIST and 0.87 on Fashion-MNIST, while simpler architectures like Probabilistic Autoencoders provided competitive results (ROC-AUC ≈ 0.89) with significantly reduced training time. This performance spectrum allows practitioners to balance detection accuracy against computational efficiency based on operational constraints.

Figure 4.8 illustrates the canonical autoencoder architecture, comprising an encoder network that maps input data to a compressed latent representation, and a decoder network) that reconstructs the inpup. The bottleneck design forces the network to learn meaningful feature abstractions while discarding redundant information, creating a framework where normal patterns are efficiently encoded and anomalous patterns exhibit poor reconstruction fidelity due to their deviation from learned representations.

The computational efficiency of autoencoders represents another significant advantage over traditional anomaly detection methods. While ensemble methods like Isolation

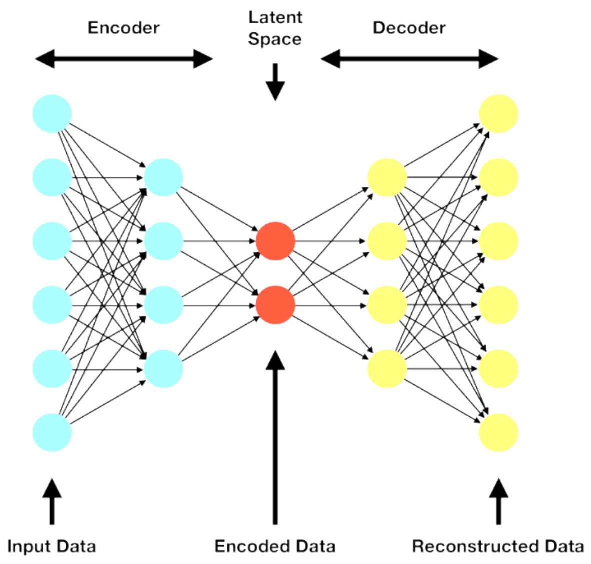


FIGURE 4.8. Architecture of an Autoencoder

Forest require substantial memory for storing multiple decision trees, and density-based approaches like LOF suffer from quadratic complexity in the number of samples, autoencoders (once trained) provide constant-time inference with linear memory requirements. Training times reported by Neloy and Turgeon (2024) range from approximately 20 minutes for PAE to 1 hour 45 minutes on standard datasets, making them practical for large-scale deployment scenarios.

The AE developed for this dissertation is shown in Table 4.10.

TABLE 4.10. Autoencoder Architecture and Training Strategy

Aspect	**Specification**
Encoding layers	Progressive dimensionality reduction 128 → 64 → 32 with BatchNormalization and Dropout regularization
Training strategy	Exclusive training on normal samples (label = 0) to learn baseline patterns
Anomaly detection	Samples producing high reconstruction error are classified as anomalies
Threshold selection	Percentile thresholds tested: $90^{th}, 95^{th}, 97^{th}, 98^{th}, 99^{th}, 99.5^{th}$

To support the approach implemented, a comprehensive anomaly detection dashboard has been created to evaluate the results of the developed algorithms.

CHAPTER 5

Results and Discussion

Continuing with the CRISP-DM methodology, after the modeling stage, we reach the evaluation, the fifth critical stage. In this Chapter, we will determine which models give the best answer to our research questions and what strategic actions should follow.

5.1. Metrics

The assessment of binary classification models relies on the systematic analysis of prediction outcomes against a known baseline. After running the model predictions for the test dataset, a performance evaluation is conducted through the construction of a confusion matrix, which categorizes all the classification results in four distinct outcomes.

This confusion matrix helps to provide a comprehensive understanding by organizing predictions according to the following classifications:

- True Positives (TP)
  Instances where positive cases are correctly predicted as positive
- False Positives (FP)
  Instances where negative cases are incorrectly predicted as positive
- True Negatives (TN)
  Instances where negative cases are correctly predicted as negative
- False Negatives (FN)
  Instances where positive cases are incorrectly predicted as negative

Table 5.1 depicts the structure of a confusion matrix

Table 5.1. Binary Classification Confusion Matrix

	Positive Prediction	**Negative Prediction**
Positive Class	True Positive (TP)	False Negative (FN)
Negative Class	False Positive (FP)	True Negative (TN)

From the confusion matrix, we can derive a set of additional performance metrics to assess the algorithm's effectiveness. The ones being considered for our dissertation are **Accuracy (ACC)**, **Precision (PRE)**, **Recall (REC)**, **F-1 Score (F-1S)**. To make the analysis more comprehensive, we have also incorporated the **AUC** of the **ROC** curve as a graphical performance assessment tool. The ROC curve visualizes the relationship between the True Positive Rate (Sensitivity) and the False Positive Rate (1-specificity), illustrating the trade-off between sensitivity and specificity across various classification thresholds. The higher AUC values indicate superior discriminative performance, with perfect classification achieving an AUC of 1.0.

Accuracy , as defined in Equation 5.1, represents one of the most fundamental metrics for assessing classification model performance. This metric quantifies the proportion of correctly predicted instances relative to the total number of observations in the dataset, providing an overall measure of model effectiveness.

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

Precision , shown in Equation 5.2, serves as a critical classification evaluation measure. This metric calculates the ratio of correctly identified positive cases to all instances predicted as positive by the model, indicating the model's ability to avoid false positive classifications.

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

Recall , presented in Equation 5.3, measures the model's capability to identify positive instances correctly. It represents the proportion of actual positive cases that were successfully detected by the classifier, reflecting the model's sensitivity to positive examples.

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

F1-Score , defined in Equation 5.4, provides a balanced assessment by computing the harmonic mean of precision and recall. This metric ranges from 0 to 1, where higher values indicate superior model performance, making it particularly valuable for evaluating models on imbalanced datasets.

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

5.2. Baseline Performance

In Section 4.2, we described the approach to set the baseline (Section 4.3.1). The goal is to set the benchmark for the following approaches taken. Figure 5.1 depicts the performance for the five models used. The charts are split per number of epochs[1], to understand if the increase in epochs improved the metrics. For the current approach, the only model where epochs were implemented is the Deep Neural Network. Table 5.2 supports the charts, including the ROC-AUC performance for each model.

To provide additional visual context for the model performance comparison, Figure 5.2 presents the ROC curves for all baseline models. The ROC-AUC analysis reinforces the quantitative results presented in Table 5.2, where the visual representation of the true positive rate versus false positive rate clearly demonstrates the superior discriminative ability of Random Forest (AUC = 90.83%) and Gradient Boosting (AUC = 87.15%) models, while highlighting the poor performance of the Neural Network model, which performs at random chance level.

[1]An epoch is the number of times the entire training dataset passes through the model during training

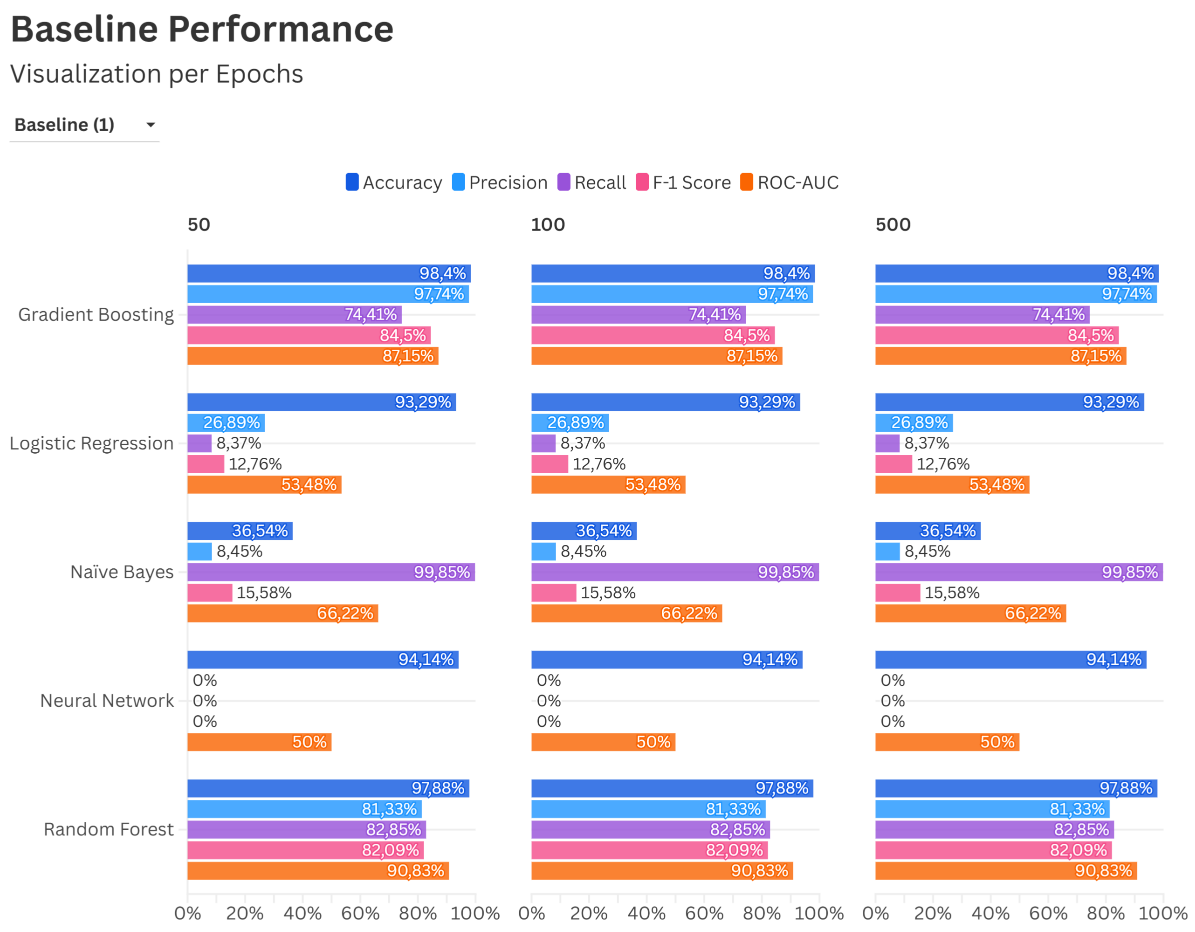


Figure 5.1. Baseline Performance - Per Epochs

Table 5.2. Baseline Model Performance Results

Step	Model	Ep.	Acc.(%)	Prec.(%)	Rec.(%)	F-1(%)	AUC(%)
Baseline(1)	Gradient Boosting	50	98.40	97.74	74.41	84.50	87.15
Baseline(1)	Gradient Boosting	100	98.40	97.74	74.41	84.50	87.15
Baseline(1)	Gradient Boosting	500	98.40	97.74	74.41	84.50	87.15
Baseline(1)	Random Forest	50	97.88	81.33	82.85	82.09	90.83
Baseline(1)	Random Forest	100	97.88	81.33	82.85	82.09	90.83
Baseline(1)	Random Forest	500	97.88	81.33	82.85	82.09	90.83
Baseline(1)	Naïve Bayes	50	36.54	8.45	99.85	15.58	66.22
Baseline(1)	Naïve Bayes	100	36.54	8.45	99.85	15.58	66.22
Baseline(1)	Naïve Bayes	500	36.54	8.45	99.85	15.58	66.22
Baseline(1)	Logistic Regression	50	93.29	26.89	8.37	12.76	53.48
Baseline(1)	Logistic Regression	100	93.29	26.89	8.37	12.76	53.48
Baseline(1)	Logistic Regression	500	93.29	26.89	8.37	12.76	53.48
Baseline(1)	Neural Network	50	94.14	0.00	0.00	0.00	50.00
Baseline(1)	Neural Network	100	94.14	0.00	0.00	0.00	50.00
Baseline(1)	Neural Network	500	94.14	0.00	0.00	0.00	50.00

The Deep Neural Network is the worst model to be used during the baseline, achieving zero percentage in precision, recall, and F-1 Score, which means that it has only learned how to predict the majority class (accuracy showed a 94.14% performance). We have run until 500 epochs, but the model stopped at epoch 29 as no major improvement was found

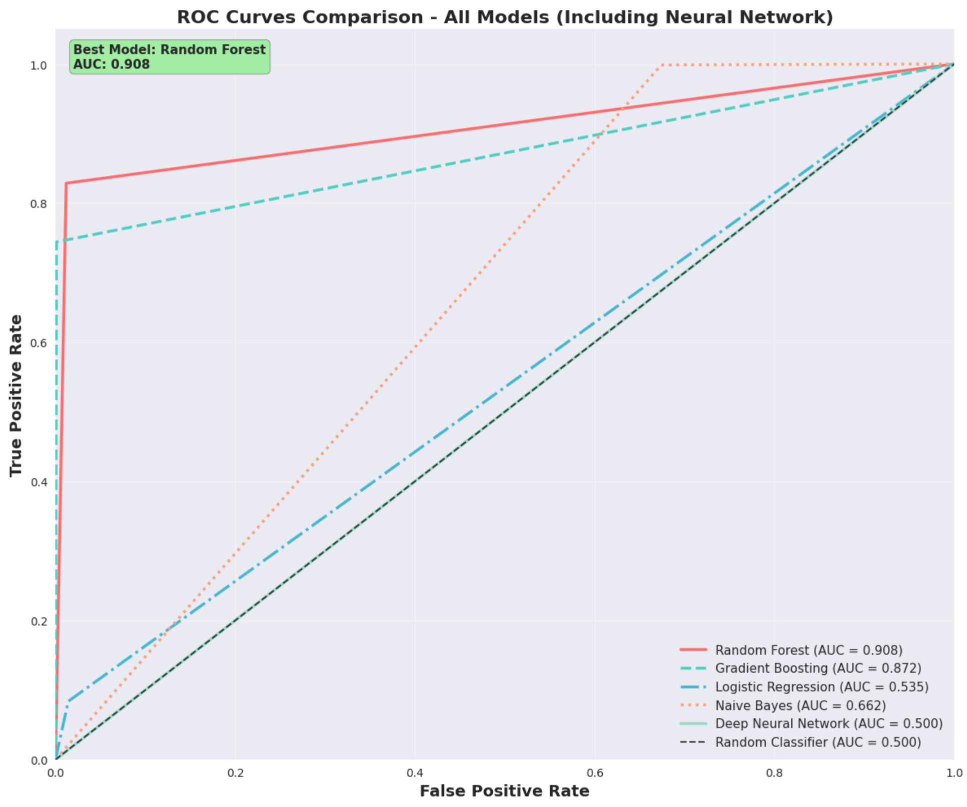


FIGURE 5.2. ROC Curves Comparison for Baseline Models

from that epoch onwards. The Neural Network ROC-AUC also showed the worst result of 50%, where it performed no better than random chance in distinguishing between classes.

Random Forest is the model with the most balanced performance throughout all the metrics, which, in an imbalanced dataset like HIKARI-2021, is a crucial approach. This model was able to perform above 80% on precision, recall, and F-1 Score, while achieving the highest AUC score of 90.83%.

Gradient Boosting is the model with the best F-1 Score, slightly outperforming Random Forest. Although Gradient Boosting has lower recall performance, the sacrifice is justified to achieve a better precision score (97.74%), which is preferable in an imbalanced dataset. Naïve Bayes and Logistic Regression are unsuitable for this imbalanced anomaly detection problem.

5.3. CTGAN Performance

In this Section, we will prepare the dataset to start answering our RQs made in Section 1.2. We will evaluate the performance of the same models used in the baseline, but augmenting the dataset with synthetic data generated by CTGAN.

This Section will be split into two Sections to measure the performance of all the steps applied until the generated data is completed and included in the dataset.

5.3.1. Synthetic Data Performance

We will start to assess the quality of the synthetic data by looking at the metrics made available by the Python library we are using, SDV. SDV provides a set of metrics available to assess the quality of the data. We have considered the following metrics for this dissertation:

- Data Validity (Diagnostic Report)

  This property applies metrics based on the column types. This yields a separate score for every column. The final Data Validity score is the average of all columns.
- Data Structure (Diagnostic Report)

  This property applies the TableStructure metric to each Table of the dataset. This checks to see that there is the same set of column names in the synthetic vs. the real data.
- Column Shape Score (Quality Report)

  The shape of a column describes its overall distribution. The higher the score, the more similar the distributions of real and synthetic data.
- Column Pair Trends (Quality Report)

  The trend between two columns describes how they vary in relation to each other, for example, the correlation. The higher the score, the more the trends are alike.

For each group (Diagnostic and Quality), the final scores are the average of both metrics, where the higher score means better quality of the synthetic data. We have also included an additional analysis, which is the PCA Quality Score. With this metric, we can measure how well the synthetic data preserves the distributional patterns of the original data in the first two principal components; the higher the score, the better the similarity between original and synthetic data. Table 5.3 depicts the performance per number of epochs.

Table 5.3. CTGAN Performance Metrics Across Training Epochs

Epochs	**Data Validity**	**Data Structure**	**Overall Score (Average)**	**Column Shape Score**	**Column Pair Trends**	**Overall Score (Average)**	**PCA Quality Score**
50	100%	100%	100%	77,47%	82,35%	79,91%	67,00%
100	100%	100%	100%	78,75%	84,26%	81,50%	77,50%
500	100%	100%	100%	78,95%	87,56%	83,25%	84,40%
1000	100%	100%	100%	74,01%	88,25%	81,13%	85,20%
5000	100%	100%	100%	80,34%	87,59%	83,97%	85,10%
10000	100%	100%	100%	80,90%	87,67%	84,29%	85,10%

By looking at the Table, the higher the number of epochs used, the better the score we get, regarding the data validity and data structure. Independent of the number of epochs, it shows an excellent performance and never shows a score below 100%, which means that CTGAN can maintain the fundamental structure properties of the original data. This shows robust data generation without any corruption or invalid entries.

Looking to the other group—Quality Report—the column shape score shows an increasing trend with the number of epochs; however, at the 1000 epochs mark, there was a performance dip, which could suggest a potential overfitting at this point. The column pair trends keep the same increasing trend per number of epochs, which means that CTGAN is effective in capturing the column relationships and correlations.

PCA quality score shows a significant improvement with the increasing training, where the score keeps improving until the 5000 epochs, where it reached a plateau of 85.10%.

Another metric taken into consideration is the CTGAN Loss Function. Loss functions are an area of active research, and many approaches have been proposed for different uses in CTGAN. CTGAN has formulated custom loss functions[2] for the purposes of creating synthetic data.

$$\mathcal{L}_D = \frac{1}{m}\sum_{i=1}^{m}[D(x'^{(i)}) - D(x^{(i)})] \tag{5.5}$$

$$\mathcal{L}_G = -\frac{1}{m}\sum_{i=1}^{m}[D(x'^{(i)})] + H \tag{5.6}$$

Here, x represents the real data and x' represents the synthetic data. Accordingly, D(x) is the discriminator's output given the real data, and D(x') is for the synthetic. Finally, H is the cross-entropy score and is always positive.

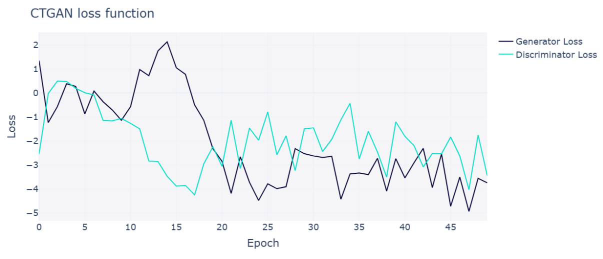
The discriminator is learning to produce low values if the data is synthetic and high values if it is real. The range of these values is dependent on linear transformations and dimensions of the input data.

Figures 5.3 and 5.4 depicts the Loss functions for 50, 100, 500, 1000, 5000, and 10000 epochs. For the initial phase (50-100 epochs), both generator and discriminator losses show instability and high variance, which is a typical behavior in the early learning phase; however, the generator keeps its loss values in the negative area, which is an expected behavior. For the 500 epoch mark, there is an intermediate convergence with a significant oscillation decrease compared with the initial phase. This behavior suggests the CTGAN is on the right path, but additional training will be needed to achieve the right convergence.

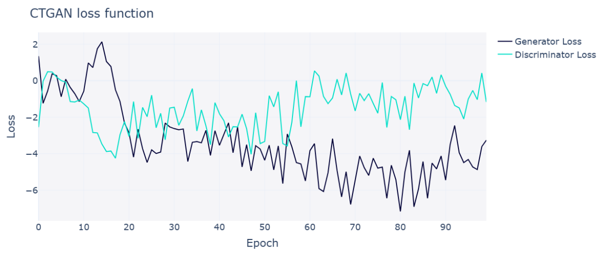
When reaching the 1000 epochs mark, we can see a progressive stabilization with a reduced loss variance; this stabilization helps to achieve the high-quality synthetic data generation expected to balance the dataset.

For the final phase (5000 - 10000 epochs as depicted in Figure 5.4), the convergence is the most promising one, where we can see both losses stabilizing and keeping consistent values throughout the epochs. According to SDV documentation, the optima stopping occurs when the generator loss stabilizes at negative values while discriminator loss oscillates around zero, which indicates a successful adversarial equilibrium. The 5000 epoch mark proved to be the most stable. The 10000 epoch mark keeps the same pattern, but no major improvement is shown.

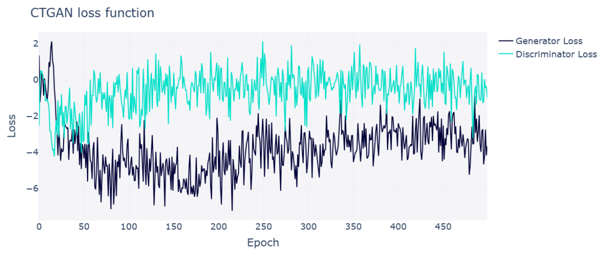
[2]https://github.com/sdv-dev/SDV/discussions/980



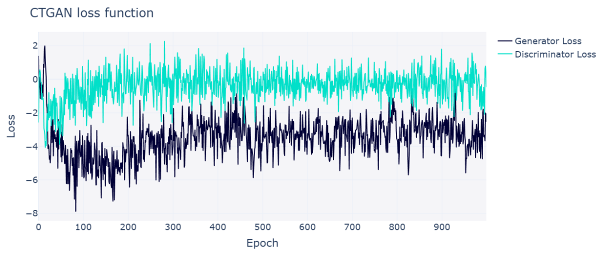
50 Epochs: Initial CTGAN variation



100 Epochs: Early stabilization

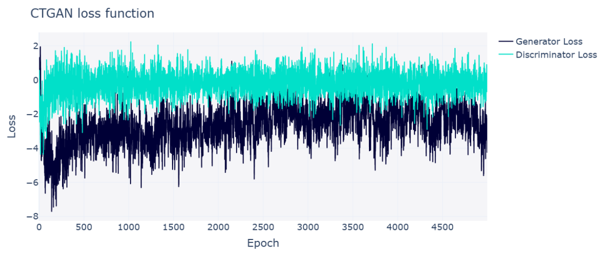


500 Epochs: Defined loss descent

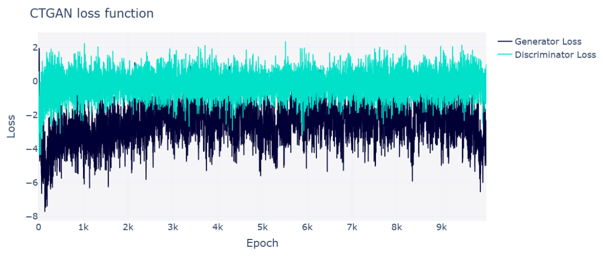


1000 Epochs: Fluctuation narrows

FIGURE 5.3. CTGAN Loss Function for epochs 50, 100, 500, and 1000.



5000 Epochs: Long-term stability become clear



10000 Epochs: Full convergence of loss curves

FIGURE 5.4. CTGAN Loss Function for epochs 5000 and 10000.

To have a deeper overview of how the synthetic data generation was reaching the expected quality, a correlation structure comparison has been put in place. The goal is to compare and visualize how similar the synthetic data correlations are to the real data correlations. Figure 5.5 shows the correlation matrix of the real data, and Figure 5.6 the correlation matrix for the synthetic data, by comparing the quality of the generated data by the number of epochs applied within CTGAN.

Analysis of the training progression reveals that the synthetic data's correlation structure remained largely stable after the 1,000-epoch mark, with a notable improvement in accuracy observed at 10,000 epochs. While the overall correlation structure at this stage closely mimics the real data, specific anomalies persist. For instance, the feature *bwd_pkts_per_sec* exhibits a contradictory trend in the synthetic dataset compared to its real counterpart.

This discrepancy is attributed to the model's difficulty in handling the severe class imbalance of the training data, which contained minimal malicious samples. This likely led to a form of mode collapse, where the generator failed to learn the long-tailed distributions of certain features and instead produced samples clustered around the median. This hypothesis is supported by the feature distribution plots (Figures B.6–B.8 in Appendix B) and a feature-by-feature statistical comparison (Table 5.4). The latter confirms a drastic reduction in standard deviation for the affected features, indicating the model's inability to capture their true variability.

TABLE 5.4. Feature-by-Feature Comparison (at 10000 epochs)

Feature	Real_Mean	Synthetic_Mean	Mean_Diff_%	Real_Std	Synthetic_Std	Std_Diff_%
originp	28156.5682	26115.0378	-7.25%	22956.4496	21785.4893	-5.10%
responp	739.1025	652.0133	-11.78%	725.9902	670.2998	-7.67%
fwd_pkts_tot	16.4217	16.9038	+2.94%	22.7457	21.7030	-4.58%
bwd_pkts_tot	13.7037	14.8205	+8.15%	22.2852	22.9653	+3.05%
fwd_data_pkts_tot	5.0916	5.5029	+8.08%	5.7794	4.3883	-24.07%
bwd_data_pkts_tot	11.3280	12.8606	+13.53%	20.3616	21.7448	+6.79%
fwd_pkts_per_sec	5108.7929	1763.5093	-65.48%	98515.3048	7764.3932	-92.12%
bwd_pkts_per_sec	6.3295	6.4665	+2.16%	109.3450	11.5165	-89.47%
flow_pkts_per_sec	5115.1224	1747.4850	-65.84%	98515.1591	7405.4212	-92.48%
down_up_ratio	0.3833	0.4357	+13.68%	0.4442	0.4709	+6.01%
fwd_header_size_tot	487.4854	532.9888	+9.33%	746.6150	774.5798	+3.75%
fwd_header_size_min	13.2912	14.2390	+7.13%	6.0998	7.0688	+15.89%
fwd_header_size_max	22.7256	24.0185	+5.69%	16.8338	16.9666	+0.79%
bwd_header_size_tot	442.0004	496.1689	+12.26%	715.9190	755.6080	+5.54%
bwd_header_size_min	13.9187	14.9708	+7.56%	15.8650	15.9752	+0.69%

5.3.2. CTGAN with Baseline Models Performance

Synthetic data has already been generated, Figure 5.7 depicts the final class distribution after the steps mentioned in Section 4.3.2. The training dataset imbalance has been mitigated, and the testing dataset keeps the imbalanced aspect as expected.

To evaluate the impact of synthetic data, the models from the baseline study were trained on the CTGAN-generated dataset. As shown in Figure 5.8, this approach led to

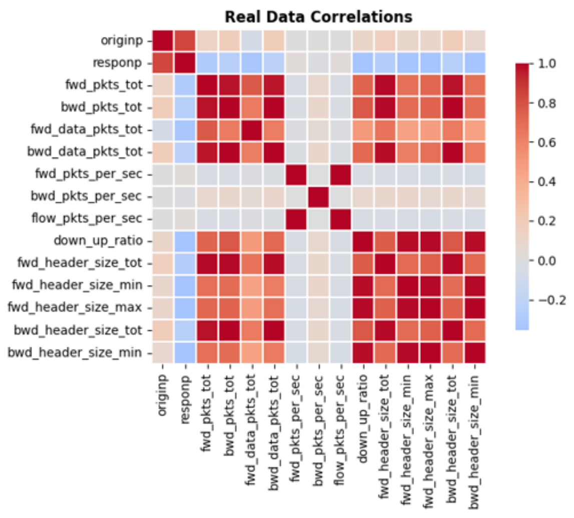
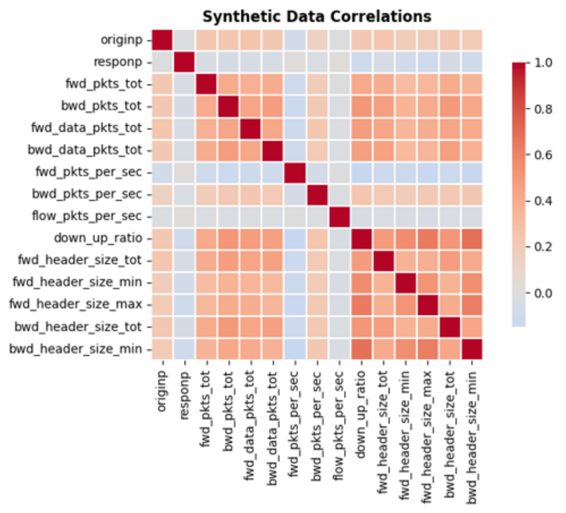
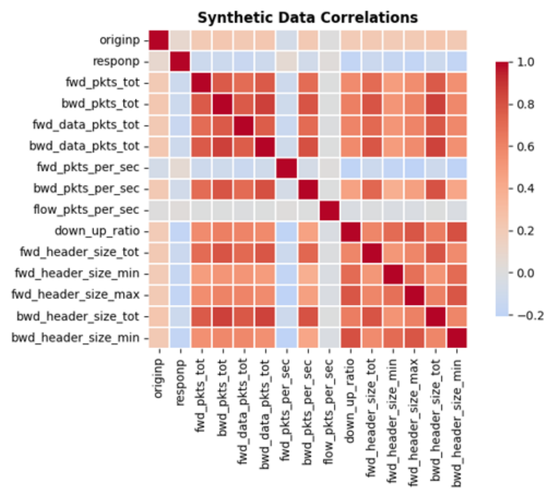


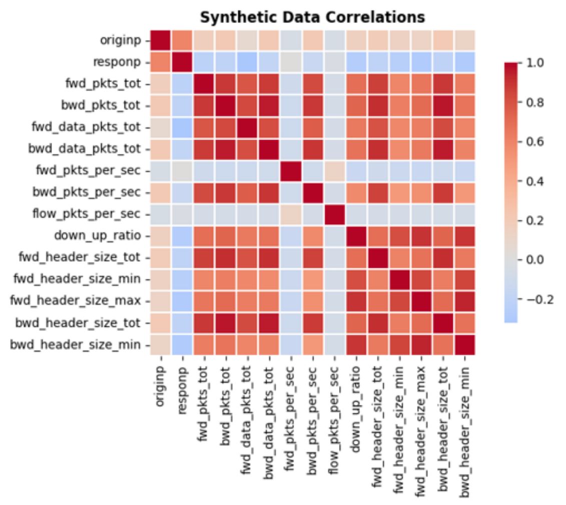
FIGURE 5.5. Correlation Structure Real Data



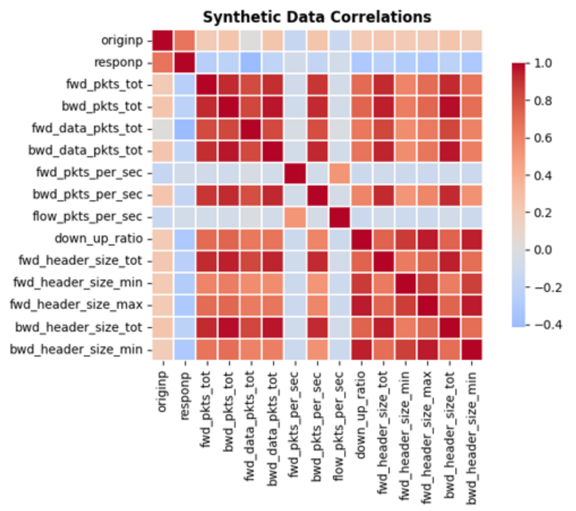
(A) 50 Epochs



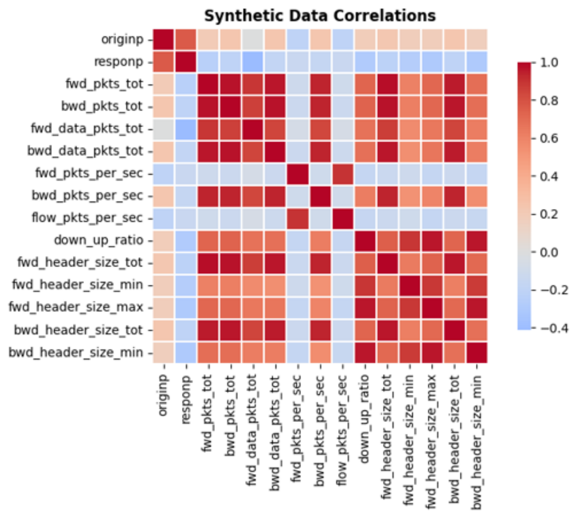
(B) 100 Epochs



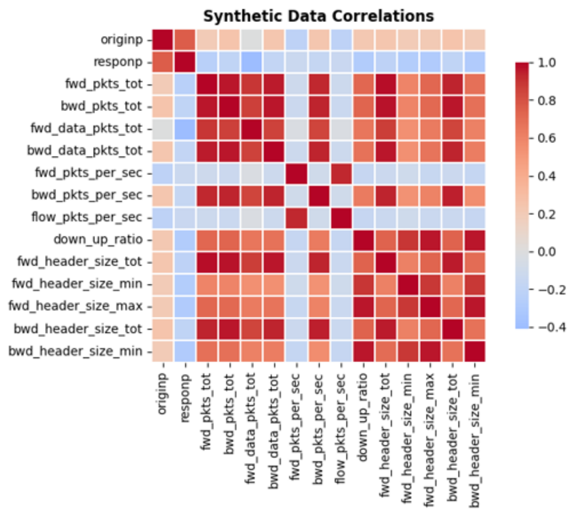
(C) 500 Epochs



(D) 1000 Epochs



(E) 5000 Epochs



(F) 10000 Epochs

FIGURE 5.6. Correlation Structure Synthetic Data - Per Epochs

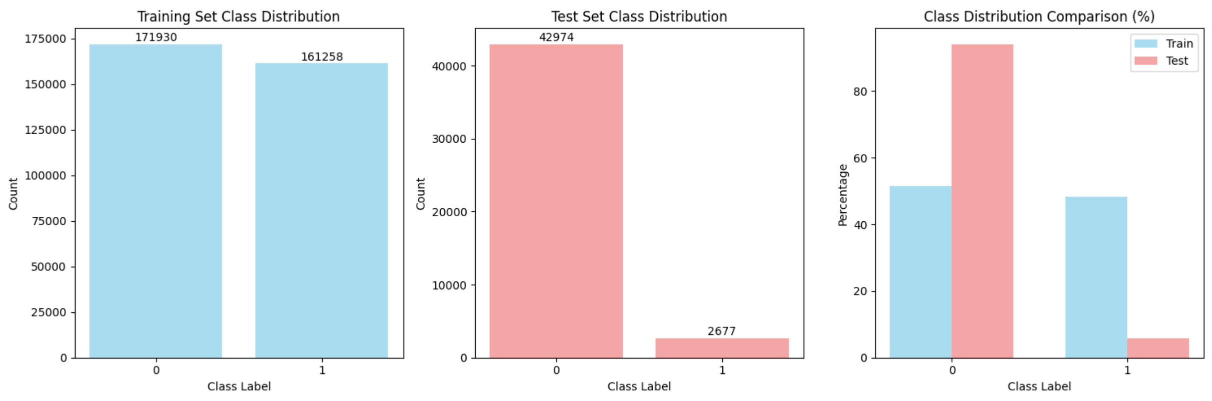


FIGURE 5.7. Class Distribution Analysis after synthetic data generation

a substantial improvement in predictive performance across all models compared to their baseline counterparts.

The most dramatic transformation was observed in the Deep Neural Network. In the baseline, this model failed entirely to predict anomalies (F-1 Score of 0.00% and ROC-AUC of 50.00%), whereas with synthetic data, it became a top-performing model, achieving an F-1 Score above 98% and a ROC-AUC exceeding 99%. Similarly, models that were previously unusable, such as Naïve Bayes and Logistic Regression, also benefited significantly. Their F-1 Scores increased from 15.58% and 12.76% to 78.47% and 97.44%, respectively, demonstrating the effectiveness of the synthetic data in resolving severe class imbalance.

Furthermore, models that already performed well in the baseline also saw significant gains. The F-1 Score for Gradient Boosting increased from 84.50% to a peak of 96.59%, while the Random Forest model's score rose from 82.09% to 98.15%. These results indicate that balancing the dataset with CTGAN enhances performance even for robust ensemble methods.

Finally, the analysis highlights that the number of CTGAN training epochs is a critical hyperparameter. For instance, the Gradient Boosting model's performance peaked when using data from a CTGAN trained for 500 epochs (96.59% F-1 Score) and subsequently declined with further training, indicating that optimal epoch selection is crucial for maximizing downstream model performance. Table 5.5 depicts the overall model performance.

As already discussed above but in a summarized way, we found relevant making a deeper analysis of ROC-AUC further confirms the benefits of training on CTGAN-generated data. The ROC-AUC metric is particularly valuable as it evaluates a model's ability to distinguish between classes across all possible classification thresholds, making it a robust indicator of performance, especially with imbalanced datasets.

Figure 5.9 (scaling was adjusted to start at 80%, to make the visualization easier to understand), after training on synthetic data, all models demonstrated excellent discriminatory power, with most achieving ROC-AUC scores well above 98%. Notably, the Gradient Boosting and Deep Neural Network models achieved near-perfect scores of 99.47%

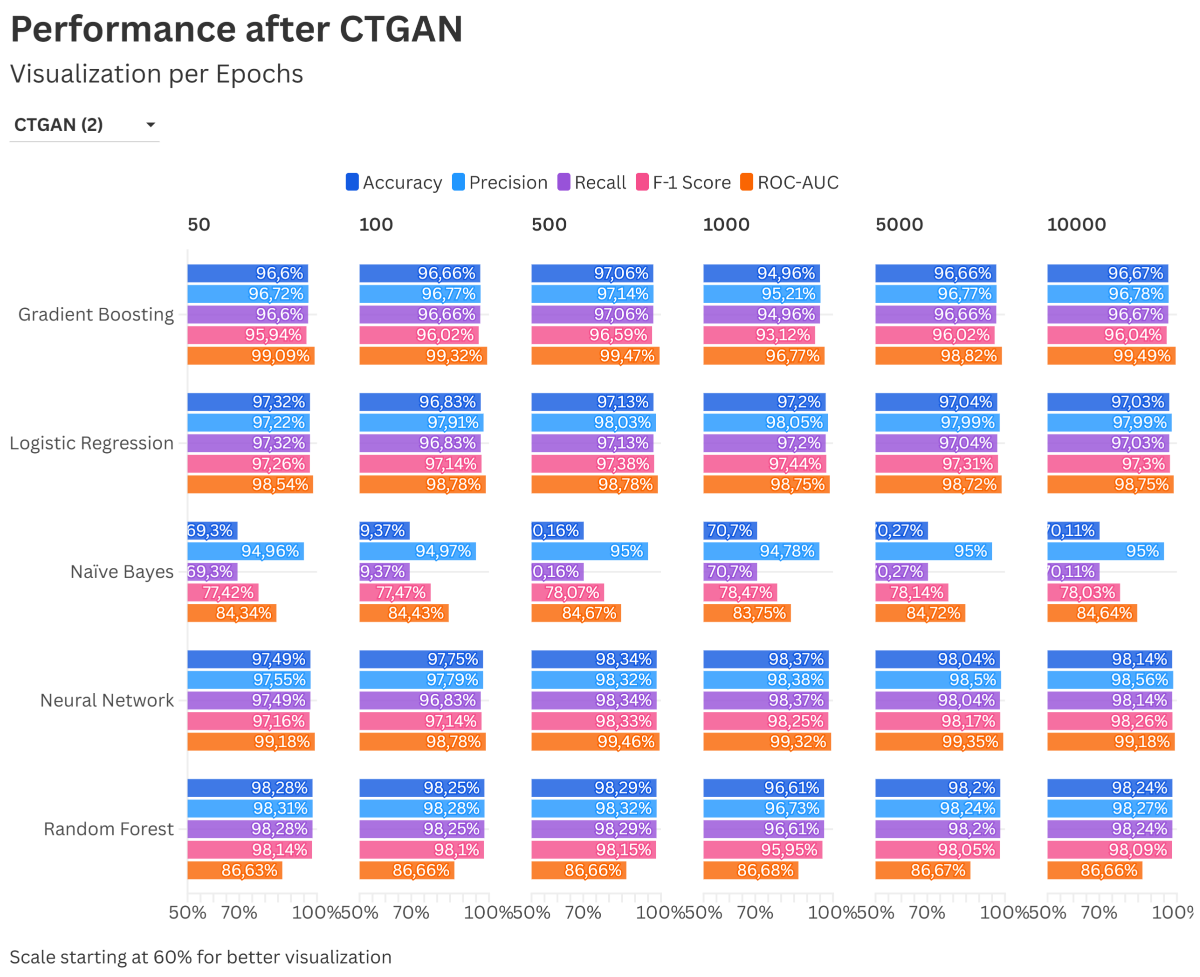


FIGURE 5.8. Performance after Synthetic Data Augmentation using CT-GAN

and 99.46%, respectively. This represents a substantial improvement from the baseline, where models like the Neural Network were no better than random chance (50.00% AUC). The consistently high ROC-AUC scores across different models underscore the quality of the synthetic data in creating a feature distribution that allows classifiers to effectively separate positive and negative instances without being biased by a majority class.

5.4. Anomaly Detection Algorithms Performance

In this Section we will discuss the performance of Anomaly Detection algorithms implemented in this dissertation, this is the last step of the analysis made, and the goal is to understand if using specific Anomaly Detection algorithms are beneficial for IDSes datasets like HIKARI-2021.

In Section 4.3.5 we already discussed which AD algorithms will be used, and since the hyperparameter tuning is a very important step to ensure the best performance of the algorithms in place, Table 4.9 depicted the optimization strategy for the implemented algorithms as well as Table 4.10 depicts the AE architecture and training strategy.

The final outcome from the strategies applied are shown in Table 5.6

TABLE 5.5. Performance Metrics for CTGAN+Baseline Models with Different Epochs

Step	Model	Ep.	Acc.(%)	Prec.(%)	Rec.(%)	F-1(%)	AUC(%)
CTGAN (2)	Gradient Boosting	50	96.60	96.72	96.60	95.94	99.09
CTGAN (2)	Gradient Boosting	100	96.66	96.77	96.66	96.02	99.32
CTGAN (2)	Gradient Boosting	500	97.06	97.14	97.06	96.59	99.47
CTGAN (2)	Gradient Boosting	1000	94.96	95.21	94.96	93.12	96.77
CTGAN (2)	Gradient Boosting	5000	96.66	96.77	96.66	96.02	98.82
CTGAN (2)	Gradient Boosting	10000	96.67	96.78	96.67	96.04	99.49
CTGAN (2)	Logistic Regression	50	97.32	97.22	97.32	97.26	98.54
CTGAN (2)	Logistic Regression	100	96.83	97.91	96.83	97.14	98.78
CTGAN (2)	Logistic Regression	500	97.13	98.03	97.13	97.38	98.78
CTGAN (2)	Logistic Regression	1000	97.20	98.05	97.20	97.44	98.75
CTGAN (2)	Logistic Regression	5000	97.04	97.99	97.04	97.31	98.72
CTGAN (2)	Logistic Regression	10000	97.03	97.99	97.03	97.30	98.75
CTGAN (2)	Naïve Bayes	50	69.30	94.96	69.30	77.42	84.34
CTGAN (2)	Naïve Bayes	100	69.37	94.97	69.37	77.47	84.43
CTGAN (2)	Naïve Bayes	500	70.16	95.00	70.16	78.07	84.67
CTGAN (2)	Naïve Bayes	1000	70.70	94.78	70.70	78.47	83.75
CTGAN (2)	Naïve Bayes	5000	70.27	95.00	70.27	78.14	84.72
CTGAN (2)	Naïve Bayes	10000	70.11	95.00	70.11	78.03	84.64
CTGAN (2)	Neural Network	50	97.49	97.55	97.49	97.16	99.18
CTGAN (2)	Neural Network	100	97.75	97.79	96.83	97.14	98.78
CTGAN (2)	Neural Network	500	98.34	98.32	98.34	98.33	99.46
CTGAN (2)	Neural Network	1000	98.37	98.38	98.37	98.25	99.32
CTGAN (2)	Neural Network	5000	98.04	98.50	98.04	98.17	99.35
CTGAN (2)	Neural Network	10000	98.14	98.56	98.14	98.26	99.18
CTGAN (2)	Random Forest	50	98.28	98.31	98.28	98.14	86.63
CTGAN (2)	Random Forest	100	98.25	98.28	98.25	98.10	86.66
CTGAN (2)	Random Forest	500	98.29	98.32	98.29	98.15	86.66
CTGAN (2)	Random Forest	1000	96.61	96.73	96.61	95.95	86.68
CTGAN (2)	Random Forest	5000	98.20	98.24	98.20	98.05	86.67
CTGAN (2)	Random Forest	10000	98.24	98.27	98.24	98.09	86.66

TABLE 5.6. Anomaly Detection Models Configuration

Step	Model	Hyperparameter	Hyperparameter Value	Epochs / Estimators
Anomaly Detection (3)	Isolation Forest	Contamination	15%	500
Anomaly Detection (3)	Local Outlier Factor	Contamination	1%	20
Anomaly Detection (3)	One-Class SVM	nu	15%	–
Anomaly Detection (3)	DBSCAN	Neighbor Distance	100%	10
Anomaly Detection (3)	Elliptical Envelope	Contamination	1%	–
Anomaly Detection (3)	Autoencoder	Percentile	95%	500

The comparative analysis that Figures 5.10 and 5.11, with Table 5.7 to support, reveals a significant performance variations across the six AD models. LOF and EE achived the highest accuracy rates (93.51% and 93.10% respectively), but the recall and f-1 score shows a low performance which means those models are not effective detecting anomalies.

Isolation Forest on the other hand, shows that is the most balanced performer, where most metrics (exception made to precision) become the best combination in detecting anomalies. Local Outlier Factor, although the best contaminator setting was at 1%, it shows poor performance in recall and f-1 score, also fails in detecting the most actual anomalies.

OCSVM is the second most balanced model, showing moderate performance across the metrics, but with low precision performance.

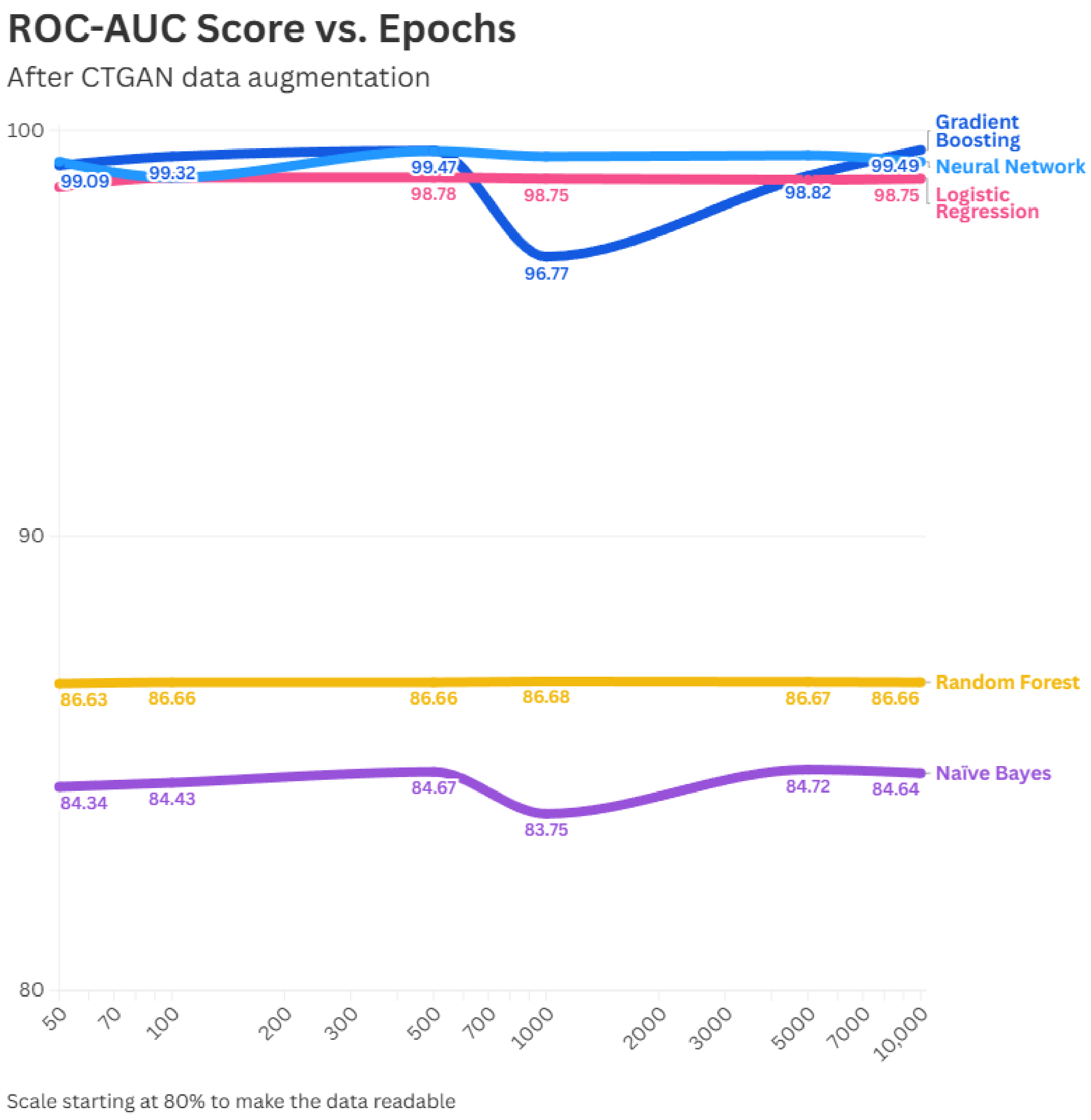


FIGURE 5.9. ROC-AUC model performance per epochs

DBSCAN and Autoencoder were worst performant models, suggesting that these models are not a good fit for the anomaly detection task being studied.

TABLE 5.7. Performance Metrics of Anomaly Detection Models

Model	**Acc.(%)**	**Prec.(%)**	**Rec.(%)**	**F-1(%)**	**AUC(%)**
Isolation Forest	87,60	28,46	73,66	41,06	81,06
Local Outlier Factor	93,51	18,82	3,21	5,49	51,17
One-Class SVM	82,95	11,47	28,39	16,34	57,37
DBSCAN	6,54	5,80	98,02	10,95	49,43
Elliptical Envelope	93,10	0,42	0,07	0,13	49,48
Autoencoder	89,18	0,44	0,37	0,40	47,54

The results shows that traditional statistical methods outperform more sophisticated approaches (like Autoencoder) for the HIKARI-2021 dataset.

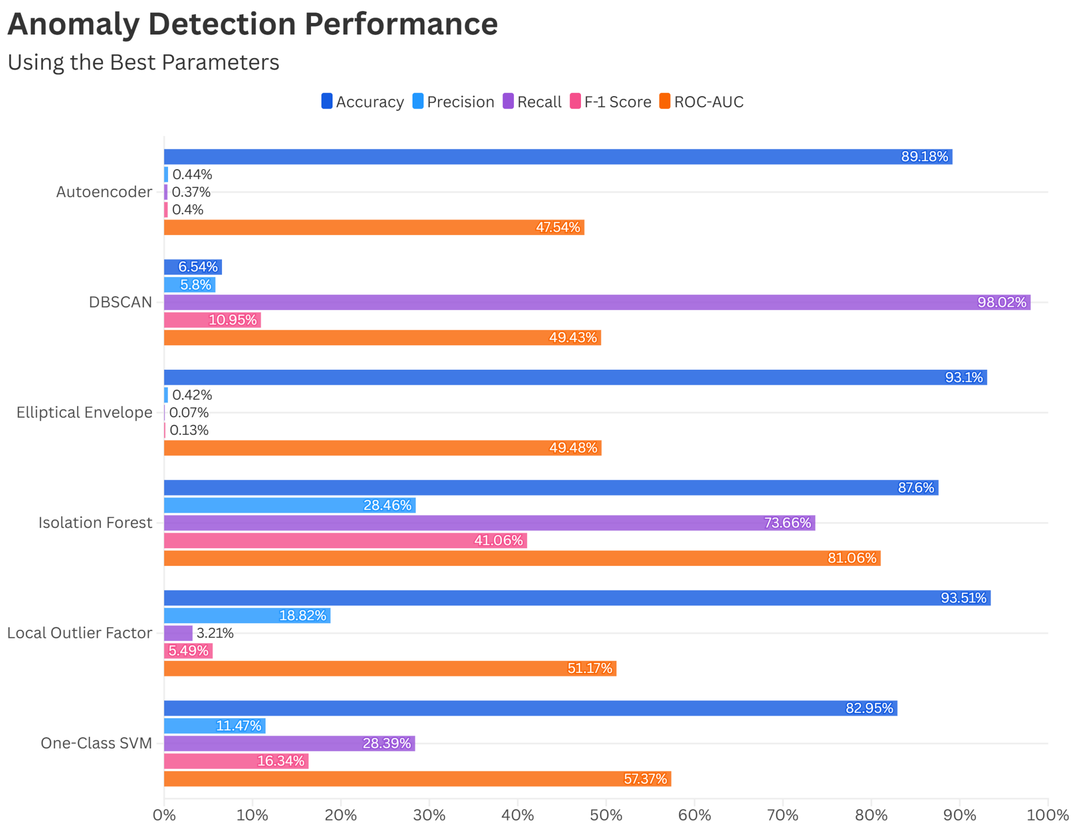


FIGURE 5.10. Anomaly Detection Models Performance

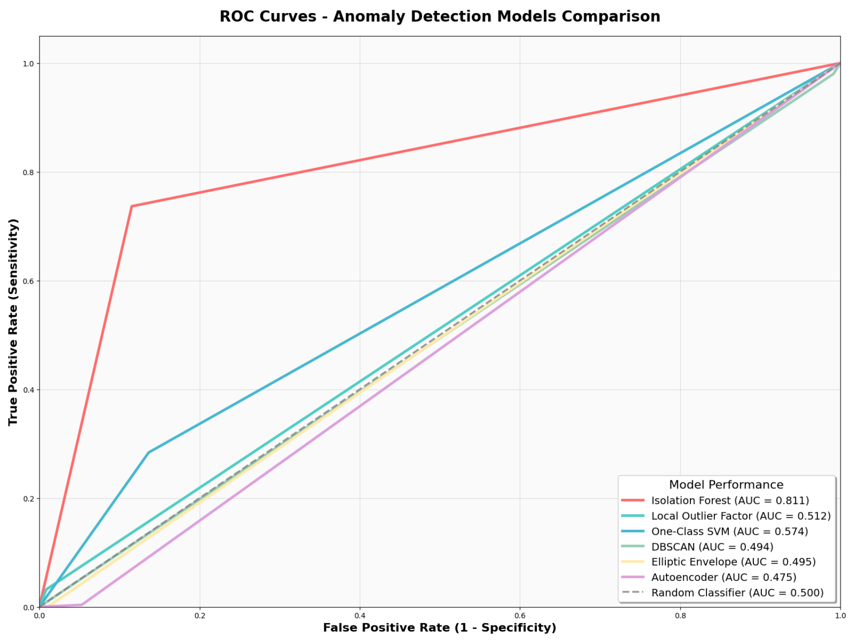


FIGURE 5.11. Anomaly Detection Algorithms Comparison

CHAPTER 6

Conclusion and Future Work

This dissertation explored the effectiveness of generating synthetic data to balance imbalanced datasets, specifically in the area of Intrusion Detection Systems. The research investigated whether dataset balancing would be more effective for detecting anomalies compared with traditional anomaly detection algorithms, and successfully answers the research questions while accomplishing the stated goals by providing a comprehensive understanding for predicting anomalies in imbalanced datasets.

Using a framework that generated synthetic data through CTGAN exclusively for the training split of the HIKARI-2021 dataset, and comparing these results with baseline models and anomaly detection algorithms, the synthetic data generation approach demonstrated exceptional performance for anomaly detection. While other models achieved satisfactory results in certain areas, none were able to match the performance that CTGAN achieved.

The HIKARI-2021 dataset represents a highly imbalanced dataset with a 16:1 ratio. We systematically evaluated baseline machine learning models, CTGAN synthetic data augmentation, and multiple anomaly detection algorithms to answer fundamental questions about optimal approaches for cybersecurity threat detection.

The study sets the stage for future work on unbalanced cybersecurity data and gives professionals clear, research-backed advice on how to build strong intrusion detection systems. Moving from understanding data to developing methods and putting them into practice shows how important it is to have careful experimental setups in cybersecurity research.

6.1. Key Findings and Contributions

Our research demonstrates that CTGAN synthetic data generation significantly improves baseline performance. The optimal configuration using 10,000 training epochs achieved F1-Score improvements of 15-20% across all baseline models. Contrary to our expectations, sophisticated anomaly detection algorithms showed variable effectiveness, with Isolation Forest emerging as the most reliable unsupervised approach, outperforming deep-learning models such as autoencoders and density-based clustering algorithms like DBSCAN. This confirms that the HIKARI-2021 dataset contains traffic anomalies characterized by isolation properties rather than reconstruction or clustering behaviors.

6.2. Research Questions Answered

Regarding RQ1, we demonstrated that applying CTGAN with traditional supervised learning models achieved superior anomaly detection results in the studied dataset compared with the baseline approach. CTGAN synthetic data augmentation, as stated in Section 6.1, demonstrates substantial improvements over the baseline approach when we are working with imbalanced datasets.

Also, the synthetic data augmentation has shown consistency during all the training cycles; however, Logistic Regression and Naïve Bayes were in the baseline approach, showing poor performance. With synthetic data augmentation, they have shown a massive improvement, and, although they are not a match for Gradient Boosting or Deep Neural Networks, they now behave as good classifiers.

Although the training cycles show an increase in performance within the higher amount of epochs, there are models that at the mark of 500-1000 epochs are already showing a ROC-AUC performance above 99%, such as Gradient Boosting or Deep Neural Networks. This efficiency can become significant for Intrusion Detection Systems, where rapid model deployment and retraining capabilities are vital to keep the systems available.

For RQ2, we determined that anomaly detection algorithms cannot match the effectiveness of CTGAN combined with supervised machine learning models in the studied case. Distance-based anomaly detection methods are the strongest performers, with Isolation Forest achieving the highest F-1 Score and ROC-AUC scores. For the density-based algorithms, although none of the ones used within this dissertation achieve relevant scores, OCSVM has stood out among the other models, but failed to capture the minority class patterns effectively.

This concludes that for the current research, unsupervised anomaly detection algorithms do not fit the purpose for Intrusion Detection Systems. Isolation Forest shows potential for deployment due to its low computational requirements and high accuracy scores.

6.3. Limitations

A significant limitation of this research is the focus on binary classification (benign vs. malicious traffic) rather than multi-class attack type classification. While the HIKARI-2021 dataset contains three distinct attack categories, our methodology aggregated all malicious samples into a single "attack" class. This approach resulted in some synthetic samples that did not accurately match the real data distribution from the original dataset.

Another limitation is the focus on a single dataset. The results presented in this dissertation may differ when applied to different datasets, limiting the generalizability of our findings.

Regarding technical constraints, training CTGAN with high epoch values requires significant computational resources (e.g., 10,000 epochs required approximately 6 hours to complete).

6.4. Future Work

Building upon the identified limitations, future research should include a multi-class approach to detect specific attack types rather than employing binary classification, which may improve synthetic data quality generation. Additionally, evaluating this framework across different datasets would ensure broader applicability of our findings.

A comparative study evaluating the effectiveness of lightweight anomaly detection algorithms in resource-constrained environments would provide valuable insights into the trade-offs between computational efficiency and detection accuracy.

[This page is intentionally left blank.]

References

Abdulganiyu, O, Ait Tchaoucht, T, Ezziyyani, M, & Benslimane, M. (2024). XIDINTV: XGBoost-based intrusion detection of imbalance network traffic via variational auto-encoder [Publisher: Lviv Polytechnic National University]. *Mathematical Modeling and Computing*, *11*(4), 930–945. https://doi.org/10.23939/mmc2024.04.930

Abdulhammed, R, Faezipour, M, Abuzneid, A, & Abumallouh, A. (2019). Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic [Publisher: Institute of Electrical and Electronics Engineers Inc.]. *IEEE Sensors Letters*, *3*(1). https://doi.org/10.1109/LSENS.2018.2879990

Ahmed, M, Naser Mahmood, A, & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, *60*, 19–31. https://doi.org/10.1016/j.jnca.2015.11.016

Ahsan, R, Shi, W, Ma, X, & Lee Croft, W. (2022). A comparative analysis of CGAN-based oversampling for anomaly detection [Publisher: John Wiley and Sons Inc]. *IET Cyber-Physical Systems: Theory and Applications*, *7*(1), 40–50. https://doi.org/10.1049/cps2.12019

Alabrah, A. (2023). An efficient NIDPS with improved salp swarm feature optimization method [Publisher: MDPI]. *Applied Sciences (Switzerland)*, *13*(12). https://doi.org/10.3390/app13127002

Alfoudi, A, Aziz, M, Alyasseri, Z, Alsaeedi, A, Nuiaa, R, Mohammed, M, Abdulkareem, K, & Jaber, M. (2022). Hyper clustering model for dynamic network intrusion detection [Publisher: John Wiley and Sons Inc]. *IET Communications*. https://doi.org/10.1049/cmu2.12523

Al-Shehari, T, Rosaci, D, Al-Razgan, M, Alfakih, T, Kadrie, M, Afzal, H, & Nawaz, R. (2024). Enhancing insider threat detection in imbalanced cybersecurity settings using the density-based local outlier factor algorithm [Publisher: Institute of Electrical and Electronics Engineers Inc.]. *IEEE Access*, *12*, 34820–34834. https://doi.org/10.1109/ACCESS.2024.3373694

Altalhan, M, Algarni, A, & Turki-Hadj Alouane, M. (2025). Imbalanced data problem in machine learning: A review. *IEEE Access*, *13*, 13686–13699. https://doi.org/10.1109/ACCESS.2025.3531662

Amit, I, Matherly, J, Hewlett, W, Xu, Z, Meshi, Y, & Weinberger, Y. (2018). Machine learning in cyber-security - problems, challenges and data sets [Publisher: arXiv Version Number: 3]. https://doi.org/10.48550/ARXIV.1812.07858

Ashok, W, Gujar, S, Mujawar, S, Sakhare, N, Pareek, V, & Bendale, S. (2024). Securing the digital frontier: The role of technology in social medical public healthcare security [Publisher: Jacobs Verlag]. *South Eastern European Journal of Public Health*, *23*, 52–62. https://doi.org/10.52710/seejph.487

Benaddi, H, Jouhari, M, Ibrahimi, K, Ben Othman, J, & Amhoud, E. (2022). Anomaly detection in industrial IoT using distributional reinforcement learning and generative adversarial networks. *Sensors*, *22*(21). https://doi.org/10.3390/s22218085

Chandola, V, Banerjee, A, & Kumar, V. (2009). Anomaly detection: A survey. *ACM Comput. Surv.*, *41*(3), 15:1–15:58. https://doi.org/10.1145/1541880.1541882

Chawla, NV, Bowyer, KW, Hall, LO, & Kegelmeyer, WP. (2011). SMOTE: Synthetic minority over-sampling technique [Publisher: arXiv Version Number: 1]. https://doi.org/10.48550/ARXIV.1106.1813

Chen, J, Gao, X, Deng, R, He, Y, Fang, C, & Cheng, P. (2022). Generating adversarial examples against machine learning-based intrusion detector in industrial control systems. *IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING*, *19*(3), 1810–1825. https://doi.org/10.1109/TDSC.2020.3037500

Chen, W, Yang, K, Yu, Z, Shi, Y, & Chen, CLP. (2024). A survey on imbalanced learning: Latest research, applications and future directions. *Artificial Intelligence Review*, *57*(6), 137. https://doi.org/10.1007/s10462-024-10759-6

Christopher, V, Aathman, T, Mahendrakumaran, K, Nawaratne, R, De Silva, D, Nanayakkara, V, & Alahakoon, D. (2021). Minority resampling boosted unsupervised learning with hyperdimensional computing for threat detection at the edge of internet of things. *IEEE ACCESS*, *9*, 126646–126657. https://doi.org/10.1109/ACCESS.2021.3111053

CTGANSynthesizer | synthetic data vault. (2025, June 3). Retrieved July 8, 2025, from https://docs.sdv.dev/sdv/single-table-data/modeling/synthesizers/ctgansynthesizer

De Araujo-Filho, P, Naili, M, Kaddoum, G, Fapi, E, & Zhu, Z. (2023). Unsupervised GAN-based intrusion detection system using temporal convolutional networks and self-attention. *IEEE Transactions on Network and Service Management*, *20*(4), 4951–4963. https://doi.org/10.1109/TNSM.2023.3260039

Ding, S, Kou, L, & Wu, T. (2022). A GAN-based intrusion detection model for 5g enabled future metaverse. *MOBILE NETWORKS & APPLICATIONS*, *27*(6), 2596–2610. https://doi.org/10.1007/s11036-022-02075-6

Dlamini, G, & Fahim, M. (2021). DGM: A data generative model to improve minority class presence in anomaly detection domain. *Neural Computing and Applications*, *33*(20), 13635–13646. https://doi.org/10.1007/s00521-021-05993-w

Duy, PT, Tien, LK, Khoa, NH, Hien, DTT, Nguyen, AGT, & Pham, VH. (2021). DIGFuPAS: Deceive IDS with GAN and function-preserving on adversarial samples in SDN-enabled networks. *Computers & Security*, *109*, 102367. https://doi.org/10.1016/j.cose.2021.102367

Fenanir, S, Semchedine, F, & Baadache, A. (2019). A machine learning-based lightweight intrusion detection system for the internet of things. *Revue d'Intelligence Artificielle*, *33*(3), 203–211. https://doi.org/10.18280/ria.330306

Fernando, GP, Florina, A, & Liliana, CB. (2024). Evaluation of the performance of unsupervised learning algorithms for intrusion detection in unbalanced data environments [Publisher: Institute of Electrical and Electronics Engineers Inc.]. *IEEE Access*, *12*, 190134–190157. https://doi.org/10.1109/ACCESS.2024.3516615

Ferriyan, A, Thamrin, AH, Takeda, K, & Murai, J. (2021). Generating network intrusion detection dataset based on real and encrypted synthetic attack traffic [Publisher: MDPI AG]. *Applied Sciences*, *11*(17), 7868. https://doi.org/10.3390/app11177868

Ferriyan, A, Thamrin, AH, Takeda, K, & Murai, J. (2022, April 15). HIKARI-2021: Generating network intrusion detection dataset based on real and encrypted synthetic attack traffic. https://doi.org/10.5281/ZENODO.6463389

Fu, J, Wang, L, Ke, J, Yang, K, & Yu, R. (2023). GANAD: A GAN-based method for network anomaly detection [Publisher: Springer]. *World Wide Web*, *26*(5), 2727–2748. https://doi.org/10.1007/s11280-023-01160-4

Fu, Y, Du, Y, Cao, Z, Li, Q, & Xiang, W. (2022). A deep learning model for network intrusion detection with imbalanced data. *ELECTRONICS*, *11*(6). https://doi.org/10.3390/electronics11060898

Ghamry, FM, El-Banby, GM, El-Fishawy, AS, El-Samie, FEA, & Dessouky, MI. (2024). A survey of anomaly detection techniques. *Journal of Optics*, *53*(2), 756–774. https://doi.org/10.1007/s12596-023-01147-4

Goodfellow, IJ, Pouget-Abadie, J, Mirza, M, Xu, B, Warde-Farley, D, Ozair, S, Courville, A, & Bengio, Y. (2014, June 10). Generative adversarial networks. https://doi.org/10.48550/arXiv.1406.2661

Gutiérrez, O, Núñez, J, Avila, M, & Caro, A. (2024). A detailed study of resampling algorithms for cyberattack classification in engineering applications. *PEERJ COMPUTER SCIENCE*, *10*. https://doi.org/10.7717/peerj-cs.1975

Hacilar, H, Aydin, Z, & Güngör, V. (2024). Network intrusion detection based on machine learning strategies: Performance comparisons on imbalanced wired, wireless, and sfwr-eotaedfieind i e i nd networking (SDN) network ratffifics fi cs. *TURKISH JOURNAL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCES*, *32*(4).

Haddaway, NR, Page, MJ, Pritchard, CC, & McGuinness, LA. (2022). *PRISMA2020* : An r package and shiny app for producing PRISMA 2020-compliant flow diagrams, with interactivity for optimised digital transparency and open synthesis. *Campbell Systematic Reviews*, *18*(2), e1230. https://doi.org/10.1002/cl2.1230

Hammad, M, Hewahi, N, & Elmedany, W. (2022). MMM-RF: A novel high accuracy multinomial mixture model for network intrusion detection systems. *COMPUTERS & SECURITY*, *120*. https://doi.org/10.1016/j.cose.2022.102777

Hao, Y, & Yan, S. (2023). A new method for intrusion detection in computer networks using computational intelligence algorithms. *International Journal of Advanced Computer Science and Applications*, *14*(9), 1170–1184. https://doi.org/10.14569/IJACSA.2023.01409122

Harini, R, Maheswari, N, Ganapathy, S, & Sivagami, M. (2023). An effective technique for detecting minority attacks in NIDS using deep learning and sampling approach. *ALEXANDRIA ENGINEERING JOURNAL*, *78*, 469–482. https://doi.org/10.1016/j.aej.2023.07.063

Hooshmand, M, Huchaiah, M, Alzighaibi, A, Hashim, H, Atlam, ES, & Gad, I. (2024). Robust network anomaly detection using ensemble learning approach and explainable artificial intelligence (XAI) [Publisher: Elsevier B.V.]. *Alexandria Engineering Journal*, *94*, 120–130. https://doi.org/10.1016/j.aej.2024.03.041

Idrissi, H, & Kartit, A. (2024). Network intrusion detection using combined deep learning models: Literature survey and future research directions. *IAENG International Journal of Computer Science*, *51*(8), 998–1010. https://www.scopus.com/inward/record.uri?eid=2-s2.0-85200902020&partnerID=40&md5=15ac23ca920f5fd7509ccb86cb544ed9

Jin, F, Chen, M, Zhang, W, Yuan, Y, & Wang, S. (2021). Intrusion detection on internet of vehicles via combining log-ratio oversampling, outlier detection and metric learning [Publisher: Elsevier Inc.]. *Information Sciences*, *579*, 814–831. https://doi.org/10.1016/j.ins.2021.08.010

Khan, F, Shah, A, Alshammry, N, Saif, S, Khan, W, Malik, M, & Ullah, Z. (2024). Balanced multi-class network intrusion detection using machine learning [Publisher: Institute of Electrical and Electronics Engineers Inc.]. *IEEE Access*, *12*, 178222–178236. https://doi.org/10.1109/ACCESS.2024.3503497

Khan, M, & Alkhathami, M. (2024). Anomaly detection in IoT-based healthcare: Machine learning for enhanced security [Publisher: Nature Research]. *Scientific Reports*, *14*(1). https://doi.org/10.1038/s41598-024-56126-x

Kim, K. (2024). An effective anomaly detection approach based on hybrid unsupervised learning technologies in NIDS. *KSII Transactions on Internet and Information Systems*, *18*(2), 494–510. https://doi.org/10.3837/tiis.2024.02.012

Kulkarni, A, Chong, D, & Batarseh, FA. (2021). Foundations of data imbalance and solutions for a data democracy [Version Number: 1]. https://doi.org/10.48550/ARXIV.2108.00071

Langr, J, & Bok, V. (2019). *GANs in action: Deep learning with generative adversarial networks*. Manning Publications.

Li, Z, Chen, S, Dai, H, Xu, D, Chu, CK, & Xiao, B. (2023). Abnormal traffic detection: Traffic feature extraction and DAE-GAN with efficient data augmentation. *IEEE Transactions on Reliability*, *72*(2), 498–510. https://doi.org/10.1109/TR.2022.3204349

Li, Z, Wang, P, & Wang, Z. (2024). FlowGANAnomaly: Flow-based anomaly network intrusion detection with adversarial learning. *Chinese Journal of Electronics*, *33*(1), 58–71. https://doi.org/10.23919/cje.2022.00.173

Lu, T, Wang, L, & Zhao, X. (2023). Review of anomaly detection algorithms for data streams. *Applied Sciences*, *13*(10), 6353. https://doi.org/10.3390/app13106353

Lu, Y, Chai, S, Suo, Y, Yao, F, & Zhang, C. (2024). Intrusion detection for industrial internet of things based on deep learning [Publisher: Elsevier B.V.]. *Neurocomputing*, *564*. https://doi.org/10.1016/j.neucom.2023.126886

Neloy, AA, & Turgeon, M. (2024). A comprehensive study of auto-encoders for anomaly detection: Efficiency and trade-offs. *Machine Learning with Applications*, *17*, 100572. https://doi.org/https://doi.org/10.1016/j.mlwa.2024.100572

Novaes, M, Carvalho, L, Lloret, J, & Proença, M. (2021). Adversarial deep learning approach detection and defense against DDoS attacks in SDN environments. *FUTURE GENERATION COMPUTER SYSTEMS-THE INTERNATIONAL JOURNAL OF ESCIENCE*, *125*, 156–167. https://doi.org/10.1016/j.future.2021.06.047

Page, MJ, McKenzie, JE, Bossuyt, PM, Boutron, I, Hoffmann, TC, Mulrow, CD, Shamseer, L, Tetzlaff, JM, Akl, EA, Brennan, SE, Chou, R, Glanville, J, Grimshaw, JM, Hróbjartsson, A, Lalu, MM, Li, T, Loder, EW, Mayo-Wilson, E, McDonald, S, . . . Moher, D. (2021). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ*, n71. https://doi.org/10.1136/bmj.n71

Patki, N, Wedge, R, & Veeramachaneni, K. (2016). The synthetic data vault. *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 399–410. https://doi.org/10.1109/DSAA.2016.49

Rookard, C, & Khojandi, A. (2024). Unsupervised machine learning for cybersecurity anomaly detection in traditional and software-defined networking environments [Publisher: Institute of Electrical and Electronics Engineers Inc.]. *IEEE Transactions on Network and Service Management*. https://doi.org/10.1109/TNSM.2024.3490181

Sadhwani, S, Manibalan, B, Muthalagu, R, & Pawar, P. (2023). A lightweight model for DDoS attack detection using machine learning techniques. *APPLIED SCIENCES-BASEL*, *13*(17). https://doi.org/10.3390/app13179937

Schiliro, F. (2023). Towards a contemporary definition of cybersecurity [Version Number: 1]. https://doi.org/10.48550/ARXIV.2302.02274

Scopus | abstract and citation database | celebrating 20 years of innovation [Www.elsevier.com]. (n.d.). Retrieved December 22, 2024, from https://www.elsevier.com/products/scopus/20-years-of-discovery

Shou, D, Li, C, Wang, Z, Cheng, S, Hu, X, Zhang, K, Wen, M, & Wang, Y. (2023). An intrusion detection method based on attention mechanism to improve CNN-BiLSTM model. *COMPUTER JOURNAL*, *67*(5), 1851–1865. https://doi.org/10.1093/comjnl/bxad105

Singh, G, & Khare, N. (2021). GSFI_smote: A hybrid multiclass classifier for minority attack detection in internet of things network [Publisher: Inderscience Publishers]. *International Journal of Ad Hoc and Ubiquitous Computing*, *38*(1), 45–61. https://doi.org/10.1504/ijahuc.2021.119085

Sivasubramanian, A, Devisetty, M, & Bhavukam, P. (2024). Feature extraction and anomaly detection using different autoencoders for modeling intrusion detection systems [Publisher: Springer Nature]. *Arabian Journal for Science and Engineering*, *49*(9), 13061–13073. https://doi.org/10.1007/s13369-024-08951-5

Song, J, Wang, X, He, M, & Jin, L. (2023). CSK-CNN: Network intrusion detection model based on two-layer convolution neural network for handling imbalanced dataset [Publisher: MDPI]. *Information (Switzerland)*, *14*(2). https://doi.org/10.3390/info14020130

Sun, Y, Que, H, Cai, Q, Zhao, J, Li, J, Kong, Z, & Wang, S. (2022). Borderline SMOTE algorithm and feature selection-based network anomalies detection strategy [Publisher: MDPI]. *Energies*, *15*(13). https://doi.org/10.3390/en15134751

Tavallaee, M, Bagheri, E, Lu, W, & Ghorbani, AA. (2009). A detailed analysis of the KDD CUP 99 data set. *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 1–6. https://doi.org/10.1109/CISDA.2009.5356528

Wanda, P. (2020). A survey of intrusion detection system. *International Journal of Informatics and Computation*, *1*(1), 1. https://doi.org/10.35842/ijicom.v1i1.7

Wang, S, Xu, W, & Liu, Y. (2023). Res-TranBiLSTM: An intelligent approach for intrusion detection in the internet of things. *COMPUTER NETWORKS*, *235*. https://doi.org/10.1016/j.comnet.2023.109982

Wankhade, N, & Khandare, A. (2023). Optimization of deep generative intrusion detection system for cloud computing: Challenges and scope for improvements. *EAI ENDORSED TRANSACTIONS ON SCALABLE INFORMATION SYSTEMS*, *10*(6). https://doi.org/10.4108/eetsis.3993

Web of science core collection | clarivate. (n.d.). Retrieved December 22, 2024, from https://clarivate.com/academia-government/scientific-and-academic-research/research-discovery-and-referencing/web-of-science/web-of-science-core-collection/

Wirth, R, & Hipp, J. (2000). Crisp-dm: Towards a standard process model for data mining. *Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining*.

Xu, H, Sun, Z, Cao, Y, & Bilal, H. (2023). A data-driven approach for intrusion and anomaly detection using automated machine learning for the internet of things. *Soft Computing*, *27*(19), 14469–14481. https://doi.org/10.1007/s00500-023-09037-4

Xu, L, Skoularidou, M, Cuesta-Infante, A, & Veeramachaneni, K. (2019). Modeling tabular data using conditional GAN [Version Number: 2]. https://doi.org/10.48550/ARXIV.1907.00503

Xu, W, Jang-Jaccard, J, Liu, T, Sabrina, F, & Kwak, J. (2022). Improved bidirectional GAN-based approach for network intrusion detection using one-class classifier. *COMPUTERS*, *11*(6). https://doi.org/10.3390/computers11060085

Xu, W, Jang-Jaccard, J, Singh, A, Wei, Y, & Sabrina, F. (2021). Improving performance of autoencoder-based network anomaly detection on NSL-KDD dataset. *IEEE Access*, *9*, 140136–140146. https://doi.org/10.1109/ACCESS.2021.3116612

Zhao, X, Fok, K, & Thing, V. (2024). Enhancing network intrusion detection performance using generative adversarial networks. *COMPUTERS & SECURITY*, *145*. https://doi.org/10.1016/j.cose.2024.104005

Zhu, Y, Cui, L, Ding, Z, Li, L, Liu, Y, & Hao, Z. (2022). Black box attack and network intrusion detection using machine learning for malicious traffic. *COMPUTERS & SECURITY*, *123*. https://doi.org/10.1016/j.cose.2022.102922

Zoghi, Z, & Serpen, G. (2024). Building an intrusion detection system on UNSW-NB15: Reducing the margin of error to deal with data overlap and imbalance. *Concurrency and Computation: Practice and Experience*, *36*(25). https://doi.org/10.1002/cpe.8242

Zola, F, Segurola-Gil, L, Bruse, J, Galar, M, & Orduna-Urrutia, R. (2022). Network traffic analysis through node behaviour classification: A graph-based approach with temporal dissection and data-level preprocessing [Publisher: Elsevier Ltd]. *Computers and Security*, *115*. https://doi.org/10.1016/j.cose.2022.102632

[This page is intentionally left blank.]

APPENDIX A

Complete Literature Review

This appendix presents a comprehensive systematic literature review of recent advances in network intrusion detection systems, with particular emphasis on approaches addressing class imbalance challenges in cybersecurity datasets. The literature synthesis encompasses 45 peer-reviewed publications spanning the period from 2019 to 2024, representing the current state-of-the-art in anomaly detection and machine learning methodologies applied to network security.

Table A.1. Summary of identified publications

Author & Year	**Methodology**	**Algorithms**	**Key findings**	**Limitations**
Abdulhammed et al. (2019)	Deep learning and machine learning approaches with various sampling techniques on imbalanced network traffic	Deep Neural Networks, Random Forest, Voting, Variational Autoencoder, SMOTE	Achieved up to 99.99% accuracy using Random Forest with proper handling of imbalanced classes	Limited evaluation on contemporary attack types; potential overfitting with oversampling techniques
Singh and Khare (2021)	Hybrid multiclass classifier for minority attack detection using feature selection and oversampling	GSFI_SMOTE (Random Forest with grid-search cross-validation, feature importance, SMOTE)	Outperforms existing methods with better minority attack detection rates; applicable to resource-limited fog/edge infrastructure	Limited to UNSW-NB15 dataset only; specific to IoT network environments; requires periodic model updates
Xu et al. (2021)	5-layer autoencoder with optimized data preprocessing and reconstruction loss	5-layer AE [122-32-5-32-122] with MAE loss and 95th percentile outlier removal	Achieved 90.61% accuracy and 92.26% F1-score on NSL-KDD; data preprocessing has largest performance impact	Limited to NSL-KDD; needs validation on real-world large-scale environments
Dlamini and Fahim (2021)	Data generative model to improve minority class presence	DGM based on conditional GAN + KL-divergence	7% improvement over original dataset; outperforms SMOTE and existing methods in F-score	Model complexity increases with data scale; performance varies with activation functions
Christopher et al. (2021)	Enhanced Geometric SMOTE with Growing Self Organizing Map for edge computing threat detection	EG-SMOTE, Growing Self Organizing Map, Hyperdimensional Computing	F-score improvements: KDD99 (0.9360), NSL-KDD (0.9647), CICIDS2017 (0.9999), BoT-IoT (0.9445)	Requires fine-tuning for different datasets; complexity in handling dynamic network environments

Continued on next page

Table A.1 (continued)

Author & Year	Methodology	Algorithms	Key findings	Limitations
Novaes et al. (2021)	Adversarial Deep Learning approach using GAN framework for DDoS attack detection and defense in SDN environments	Generative Adversarial Network (GAN), Deep Neural Network discriminator, Event-Condition-Action (ECA) mitigation model	Achieved 99% accuracy on both emulated SDN environment and CICDDoS2019 dataset with near real-time response ($\leq$ 1 second)	Focus primarily on DDoS attacks; requires continuous model updates; high computational overhead for GAN training
Jin et al. (2021)	Novel oversampling strategy based on logarithmic ratio (OBLR), outlier detection, and metric learning for Internet of Vehicles	Log-ratio-based oversampling, Local Outlier Factor (LOF), distance metric learning, genetic algorithm for feature selection, LightGBM	98.51% accuracy with 0.82% false alarm rate on UNSW-NB15; highest performance on vehicle-specific datasets (ROAD, Car-Hacking, CAN-intrusion)	Complex multi-step preprocessing; computationally intensive feature selection process
Duy et al. (2021)	Black-box adversarial attack framework using Wasserstein GAN to generate adversarial samples against ML-based IDS	Wasserstein GAN (WGAN), function-preserving adversarial sample generation, black-box attack methodology	Successfully reduced detection rates of various ML-based IDS by generating adversarial samples while preserving attack functionality	Limited to specific attack types; requires knowledge of target IDS feature extraction methods
Sun et al. (2022)	Borderline SMOTE with information gain ratio feature selection	Borderline SMOTE, Information Gain Ratio, KNN, Decision Tree, Random Forest	Decision Tree with 62 features achieved best overall performance; feature selection improved efficiency while maintaining accuracy	More features do not necessarily ensure better performance; limited to CIC-IDS2017 dataset evaluation
Xu et al. (2022)	Bidirectional GAN with relaxed training dependency and one-class classifier	Improved BiGAN with independent generator/discriminator training	Achieved 92% F1-score on NSL-KDD and 99% F1-score on CIC-DDoS2019; reduces training overheads with simpler loss function	Limited to specific datasets; requires careful tuning of training iteration ratios between generator and discriminator
Chen et al. (2022)	Adversarial examples generation against ML-based IDS in ICS	Optimal solution attack (Opt. attack) and GAN attack	Both optimal solution and GAN algorithm achieve 80% higher probability to evade detector while maintaining same attack effect	Complex optimization with continuous/discontinuous variables; requires specific ICS domain knowledge
Alfoudi et al. (2022)	Hyper clustering model for dynamic network intrusion detection	EDBSCAN (Enhanced DBSCAN + cosine similarity)	Mean silhouette score of 0.87; reduces MSE from 0.66 to 0.13; achieves 86.82%–90.03% accuracy across datasets	Requires manual parameter tuning (α, θ variables); complexity increases with data volume

Continued on next page

Table A.1 (continued)

Author & Year	**Methodology**	**Algorithms**	**Key findings**	**Limitations**
Ahsan et al. (2022)	Comparative analysis of CGAN-based oversampling for anomaly detection	AEGAN (Adversarial Environment RL + CGAN)	CGAN consistently outperforms other oversampling techniques; LR+CGAN achieves F1-score of 0.985	Algorithm-level approaches show limited improvement when combined with data-level approaches
Fu et al. (2022)	Deep learning model for network intrusion detection with imbalanced data	DLNID (CNN + Attention + Bi-LSTM + ADASYN)	90.73% accuracy and 89.65% F1-score; ADASYN yields 2.09% improvement over SMOTE	Performance degrades on multiclass tasks; U2R category prone to misclassification
Ding et al. (2022)	Hybrid intrusion detection using GAN with deep autoencoder and random forest for 5G metaverse environments	GAN, Deep Autoencoder (DAE), Random Forest	Achieved 99.8% and 99.6% accuracy for binary and multiclass classification, respectively	High computational cost; complexity in GAN training; limited to specific IoT scenarios
Benaddi et al. (2022)	Distributional Reinforcement Learning combined with GAN for Industrial IoT anomaly detection	Distributional Reinforcement Learning, GAN, Wasserstein GAN	Enhanced detection of minority attack classes; effective on DS2OS dataset with improved F1-scores	High training time; computationally resource-intensive; limited to specific IoT environments
Zola et al. (2022)	Graph-based approach with temporal dissection and novel data-level preprocessing techniques for node behavior classification	Traffic Dispersion Graphs (TDGs), R-hybrid and SM-hybrid preprocessing, Graph Convolutional Networks (GCN), Neural Networks	Improved supervised node behavior classification performance; outperformed traditional anomaly detection methods	Limited to specific temporal snapshot sizes; computational complexity increases exponentially with graph size
Hammad et al. (2022)	Multinomial Mixture Modeling with Random Forest using correlation feature selection and T-SNE dimensionality reduction	Multinomial Mixture Modeling (MMM), Expectation-Maximization (EM), Median Absolute Deviation (MAD), Random Forest, CFS, T-SNE, SMOTE	Achieved 99.98% accuracy with 0.02% false positive rate on CSE-CIC-IDS2018 dataset	Limited evaluation on single dataset; high dependency on proper feature selection and preprocessing
Zhu et al. (2022)	Black-box attack methodology using GAN to generate forged packets and defense system combining GAN with GRU for spatial-temporal feature learning	Wasserstein GAN for packet generation, Gated Recurrent Unit (GRU), spatial-temporal feature fusion	Achieved 90% evasion rate against multiple anomaly detectors; proposed defense system significantly reduced evasion effectiveness	Attack method limited to timing-based features; defense system requires extensive training data

Continued on next page

Table A.1 (continued)

Author & Year	Methodology	Algorithms	Key findings	Limitations
Xu et al. (2023)	Data balancing (SMOTE + undersampling), MI feature selection, Auto-ML for model search on KDDCup99	Bagged ensemble of decision trees (auto-tuned)	Lightweight edge-deployable model hit 99.7% multiclass accuracy after balancing and feature pruning	KDDCup99 is dated/less complex; Auto-ML requires high compute during tuning
Sadhwani et al. (2023)	Lightweight IoT network protection	ExtraTrees Classifier, SMOTE, Logistic Regression, Random Forest, Naïve Bayes, ANN, KNN	Random Forest achieved 100% accuracy on TON-IOT; Naïve Bayes achieved 100% accuracy on BOT-IOT with reduced training time	Limited to 15 features; model complexity varies across datasets; need for real-time deployment validation
Shou et al. (2023)	CNN-BiLSTM with attention mechanism	NCR-SMOTE, Recursive Feature Elimination with Extra Random Tree, CNN-BiLSTM-Attention	84.5% accuracy on UNSW-NB15; 98.3% accuracy on CSE-CIC-IDS2018; attention mechanism improved performance over single models	High computational complexity; limited scalability for real-time applications
De Araujo-Filho et al. (2023)	Unsupervised GAN with TCN and self-attention for DDoS detection	Temporal Convolutional Network (TCN), Self-attention, WGAN	Outperformed FID-GAN and ALAD with AUCROC close to 1; 3.8x faster detection time than state-of-the-art baselines	Limited to DDoS attack scenarios; requires edge computing deployment for optimal performance
Wankhade and Khandare (2023)	Survey and proposed cloud-based deep generative IDS	Deep generative models, ensemble methods, feature selection techniques	Comprehensive analysis of cloud-based attacks; proposed system addresses simultaneous attack detection including DDoS, CSCA, SQL injection	Theoretical framework without full implementation; limited real-world cloud environment testing
Fu et al. (2023)	GAN-based method for network anomaly detection	GANAD (WGAN + gradient penalty + spectral normalization)	Superior to state-of-the-art methods; precision up to 97.49%; reduced time consumption	Requires careful hyperparameter tuning; performance varies across datasets; complex architecture
Song et al. (2023)	Two-layer CNN with Cluster-SMOTE + K-means for handling imbalanced datasets	CSK-CNN, Cluster-SMOTE, K-means	Achieved high detection rates with 98.6% precision on NSL-KDD and 98.5% on UNSW-NB15; effectively handles class imbalance in NID	High computational complexity; requires parameter tuning for different datasets
Li et al. (2023)	Abnormal traffic detection using pseudoanomalies generated by multiple DAEs with adversarial training	DAE-GAN, Multiple Denoising Autoencoders, Binary Discriminator	98.6% precision on NSL-KDD; 98.5% on UNSW-NB15; effective early detection capability	Complex model architecture; high computational overhead during training

Continued on next page

Table A.1 (continued)

Author & Year	**Methodology**	**Algorithms**	**Key findings**	**Limitations**
Alabrah (2023)	Feature selection using Improved Salp Swarm Algorithm with SMOTE-Tomek balancing	ISSA, SMOTE-Tomek, Random Forest, Decision Tree, AdaBoost	Improved performance on UNSW-NB15 dataset; automated feature selection reduces dimensionality	Computational overhead for optimization algorithms; dependency on parameter tuning
Li et al. (2023)	Abnormal traffic detection using pseudoanomalies generated by multiple DAEs with adversarial training	DAE-GAN, Multiple Denoising Autoencoders, Binary Discriminator	98.6% precision on NSL-KDD; 98.5% on UNSW-NB15; effective early detection capability	Complex model architecture; high computational overhead during training
Harini et al. (2023)	Triple-layered hybrid approach for anomaly-based NID with weighted DNN, CNN+LSTM, and XGBoost	Weighted DNN (WDNN), CNN, LSTM, XGBoost, Chi-squared feature selection, One-Sided Selection (OSS), ADASYN	Overall accuracy of 97.94% on NSL-KDD, 98.3% on CICIDS-2017, and 97.9% on CIDDS-001; improved minority attack detection	High computational complexity; requires dataset-specific parameter tuning; some minority classes still show low detection rates
Wang et al. (2023)	Hybrid deep model combining ResNet, Transformer, and BiLSTM for spatial-temporal features in IoT networks	ResNet for spatial features, Transformer encoder with multi-head attention, BiLSTM for temporal features, SMOTE-ENN for balancing	Accuracy of 90.99% on NSL-KDD, 99.15% on CIC-IDS2017, and 99.56% on MQTTset; improved detection by 1–10% over baselines	High computational cost; complex architecture; limited scalability for real-time applications
Zhao et al. (2024)	GAN-based data augmentation; compared three GAN variants on CIC-IDS2017	Vanilla GAN, WGAN, CTGAN	Synthetic Botnet samples boosted IDS precision/recall; WGAN and Vanilla GAN produced closest data and largest performance gains	Little gain for tiny classes (e.g., Infiltration); CTGAN less effective on simple distributions; diminishing returns with very large augmentation
Zoghi and Serpen (2024)	Ensemble IDS with novel threshold-adjustment algorithms to curb class imbalance/overlap on UNSW-NB15	Balanced Bagging, XGBoost, RF-HDDT	Ensemble plus threshold tuning raised sensitivity and specificity, beating prior UNSW-NB15 results	Multi-class confusion among attacks still high; some classes keep notable FN; framework adds tuning complexity
Hacilar et al. (2024)	Performance comparison on imbalanced datasets	SMOTE, ROS, ADASYN, SMOTE-Tomek with XGBoost, Random Forest, KNN	Best results: UNSW-NB15 [0.9356, 0.9289, 0.9328, 0.07597], AWID [0.997, 0.9995, 0.9999, 0.0171], InSDN [0.9998, 0.9996, 0.9998, 0.0012]	Need for comprehensive cybersecurity standards and workforce training; computational resource requirements

Continued on next page

Table A.1 (continued)

Author & Year	Methodology	Algorithms	Key findings	Limitations
Gutiérrez et al. (2024)	Detailed resampling algorithms study	SMOTE, Borderline1-SMOTE, ADASYN, Random Undersampling, Cluster Centroids, NearMiss, RENN, Tomek Links	I-Forest and DBN performed better overall; two-stage classification model achieved superior results compared to single models	Limited to specific datasets; need for real-world validation in diverse IoT deployments
Ashok et al. (2024)	Hybrid deep learning for healthcare security	GANA-AO (GAN + Autoencoders + Adam optimization)	Achieved 98.33% accuracy and 98.67% TPR on IoT-23 dataset; outperformed baseline CNN and Autoencoder models	Limited to healthcare IoT environments; requires integration with existing healthcare systems
Rookard and Khojandi (2024)	Unsupervised ML comparison across traditional and SDN environments	GAN, DBN, RBM, OCSVM, Isolation Forest	I-Forest and DBN performed best overall; GAN outperformed some benchmarks on CIC-IDS2017; good generalizability across datasets	GAN struggled with certain datasets; limited to specific attack types; need for real-world SDN validation
Idrissi and Kartit (2024)	Literature survey of combined deep learning approaches for network intrusion detection	Various hybrid models (CNN + BiLSTM, GANomaly + CNN, etc.)	Combined deep learning solutions address feature extraction and class imbalance better than traditional approaches	Most existing GAN-based methods designed for images/videos; not optimized for network traffic characteristics
Kim (2024)	Hybrid anomaly detection combining deep learning and traditional ML	Conv1D-DAE + SAE + Isolation Forest + LOF with VAE oversampling	Achieved 88.2% precision and 94.6% recall with oversampling; effective threshold setting without human judgment	Limited to NSL-KDD dataset; performance heavily dependent on normal data availability
Khan and Alkhathami (2024)	Machine learning for IoT healthcare security using balanced dataset	Random Forest, AdaBoost, Logistic Regression, Perceptron, DNN with SMOTE balancing	Random Forest achieved ~99.55% accuracy under both reduced and all feature spaces; improved computational response time	Limited to CIC IoT dataset; concerns about real-time attack detection and response capabilities
Khan et al. (2024)	Balanced multi-class network intrusion detection using SMOTE-Tomek Links	Decision Tree, Random Forest, XGBoost, KNN, Naive Bayes, Logistic Regression, AdaBoost	Decision Tree and AdaBoost achieved 96.37% accuracy and 96.33% F1-score for multi-class classification	Dataset based on simulated environment; may not capture real-world traffic variability and complexity; computational scalability concerns
Al-Shehari et al. (2024)	Insider threat detection in imbalanced cybersecurity settings	DBLOF (Density-Based Local Outlier Factor)	Outstanding F-score of 98%; detection rate of 98% on CERT r4.2 dataset	Limited to specific insider threat scenarios; requires contamination parameter tuning

Continued on next page

Table A.1 (continued)

Author & Year	**Methodology**	**Algorithms**	**Key findings**	**Limitations**
Fernando et al. (2024)	Performance evaluation of unsupervised learning algorithms in unbalanced environments	K-means++, DBSCAN, LOF, Isolation Forest	K-means++ achieved 95% purity; Isolation Forest excelled in efficiency (10% CPU usage vs 16% for others)	Performance varies significantly across datasets; computational complexity issues
Sivasubramanian et al. (2024)	Feature extraction and anomaly detection using different autoencoders	CAAE-DNN (Convolutional Auto-Encoder + Attention + DNN)	79.18% accuracy; smooth epoch convergence; improved stacked autoencoder performance	Performance degrades on multiclass vs binary classification; limited U2R class improvement
Li et al. (2024)	Flow-based anomaly NID with adversarial learning	FlowGANAnomaly (GAN with flow encoder/decoder)	Superior performance on four datasets; effective anomaly scoring combining reconstruction and discriminator losses	Complex multi-component architecture; time-consuming training; variable performance across attack types
Abdulganiyu et al. (2024)	XGBoost combined with Variational Autoencoder for imbalanced network traffic classification	XGBoost, Variational Autoencoder, SMOTE	Achieved 99.79% accuracy on NSL-KDD and 99.89% on CSE-CIC-IDS2018; effective minority class detection	High resource consumption; increased training time compared to traditional methods
Hooshmand et al. (2024)	Ensemble learning combining SMOTE oversampling with K-means undersampling and explainable AI (XAI) using SHAP	SKM (SMOTE + K-means), XGBoost, SHAP, Denoising Autoencoder (DAE) for feature selection	Detection rate of 99.01% and 97.49% for binary and multiclass on UNSW-NB15; 99.37% and 99.22% on NSL-KDD with enhanced interpretability	Computationally expensive for large datasets; requires extensive hyperparameter tuning
Lu et al. (2024)	Deep neural network with global-local parallel subnetworks, hierarchical clustering-based undersampling, and optimal feature selection	Hierarchical clustering with VSS Gain (VG-AGHC), entity embedding, greedy feature selection (GRFECV), parallel CNN architecture	Improved detection performance on imbalanced datasets; better minority class recognition compared to traditional methods	Complex multi-stage preprocessing; computationally intensive hierarchical clustering; limited scalability

[This page is intentionally left blank.]

APPENDIX B

Supporting Information

This appendix presents some supporting documentation and detailed analytical results that supports the findings in the main dissertation, Figure B.1 depicts the correlation matrix for the first 50 of 87 features.

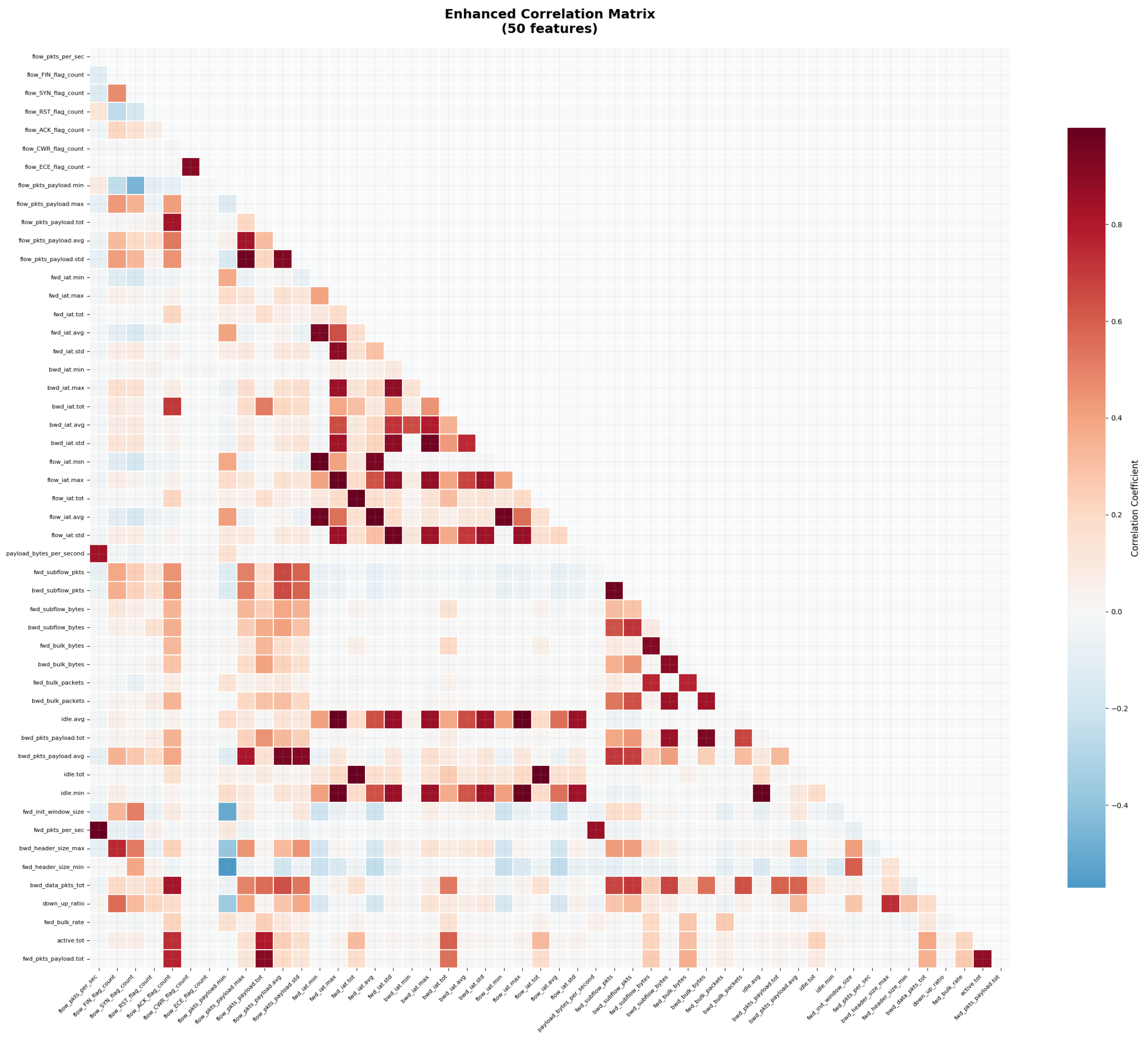


Figure B.1. Correlation Matrix - 50 Features

Figure B.2, depicts the PCA analysis before applying the CTGAN and their variances across ten components and with a detailed overview for the first 4. The heatmap allowed us to understand which feature contributes to each principal component.

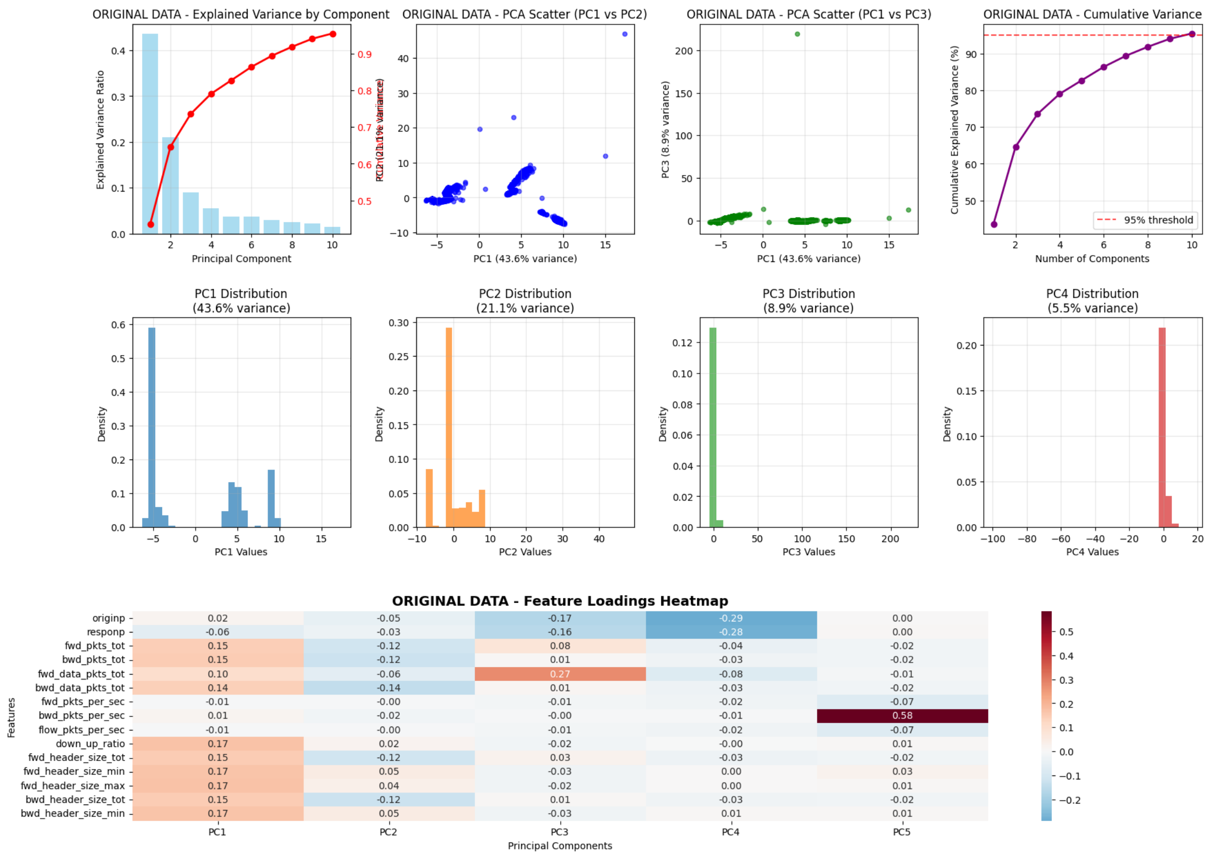
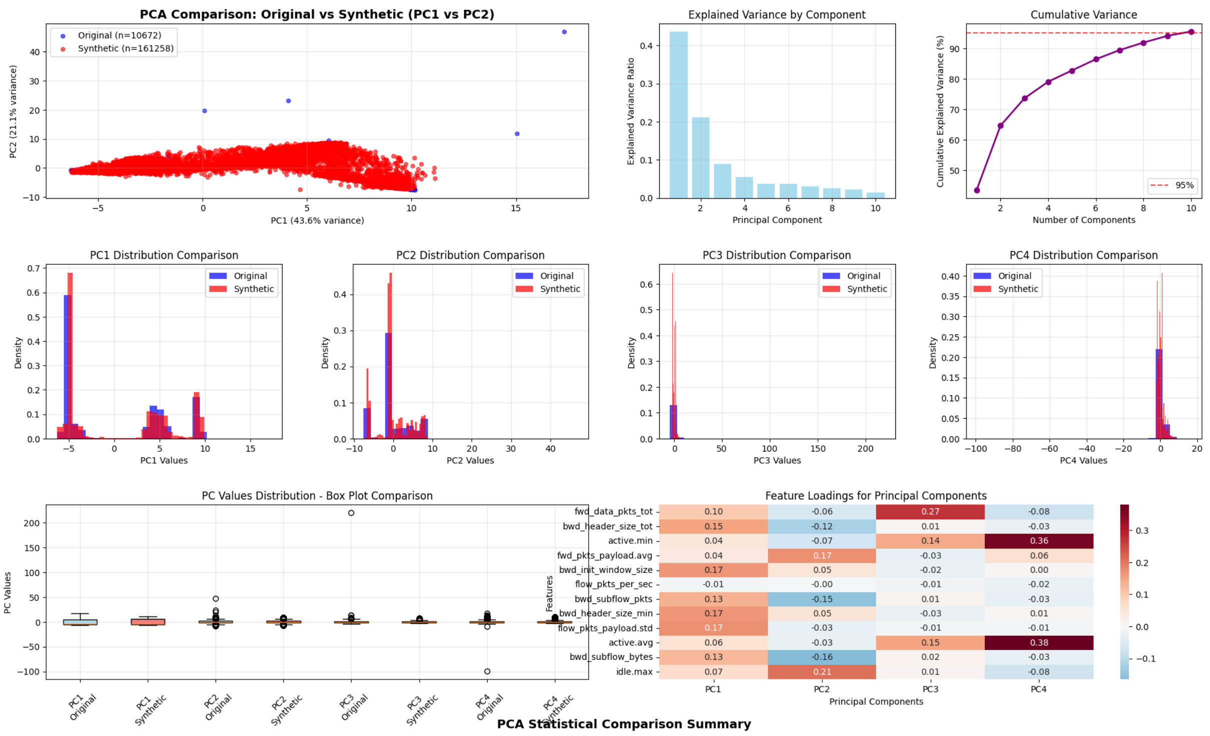


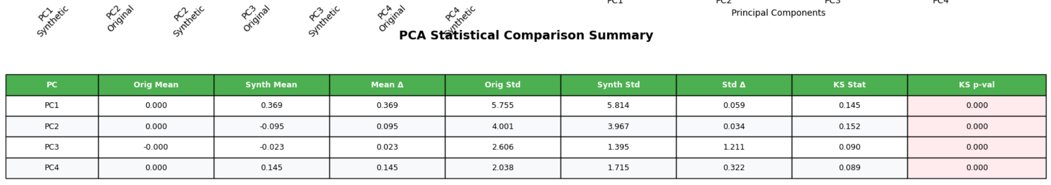
FIGURE B.2. PCA Structure before CTGAN implementation

Figure B.3 follows the same comprehensive structure as Figure B.2 but presents the evaluation after implementing synthetic data generation with 10,000 training epochs. The scatter plot depicts the first two principal components (PC1 vs PC2) in the PCA comparison, while maintaining the explained variance ratio visualization for each component.

The Figure incorporates several enhanced analytical components for data quality assessment. Distribution comparisons between original and synthetic data are visualized through histograms for the first four principal components, enabling direct evaluation of distributional similarity. The feature loadings heatmap maintains the same interpretive approach as previous analyses, revealing the contribution weights of original features to each principal component using a diverging color scheme (red-blue) centered at zero.

Additionally, box plots are integrated to provide complementary statistical visualization of the principal component value distributions, offering insights into data spread, quartiles, and potential outliers across both datasets. The comprehensive statistical overview Table at the bottom presents quantitative metrics including mean values, standard deviations, absolute differences, and Kolmogorov-Smirnov test statistics with corresponding p-values for formal hypothesis testing.




PCA Statistical Comparison Summary

PC	Orig Mean	Synth Mean	Mean Δ	Orig Std	Synth Std	Std Δ	KS Stat	KS p-val
PC1	0.000	0.369	0.369	5.755	5.814	0.059	0.145	0.000
PC2	0.000	-0.095	0.095	4.001	3.967	0.034	0.152	0.000
PC3	-0.000	-0.023	0.023	2.606	1.395	1.211	0.090	0.000
PC4	0.000	0.145	0.145	2.038	1.715	0.322	0.089	0.000

FIGURE B.3. PCA Comparison after 10000 Epochs of CTGAN generation

Figure B.4 shows in a detailed way how the data distribution and CDF is made comparing the real data against the synthetic data generated per feature (first 9 shown).

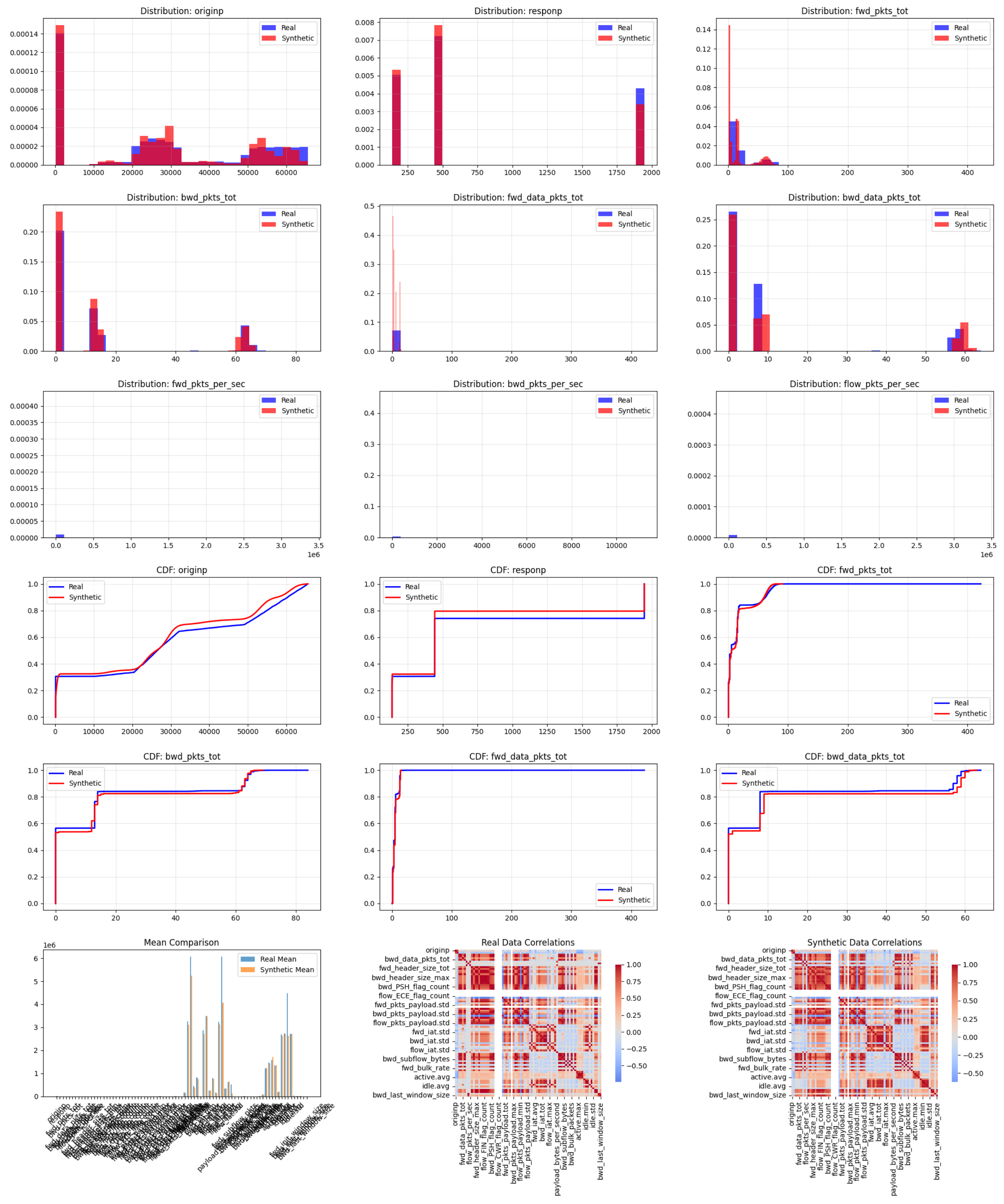


FIGURE B.4. Cumulative Distribution Function and Correlation comparison after 10000 Epochs of CTGAN

Figure B.5 shows in a detailed way how the data cumulative sum (by percentage) is made comparing the real data against the synthetic data generated per feature (first 12 shown).

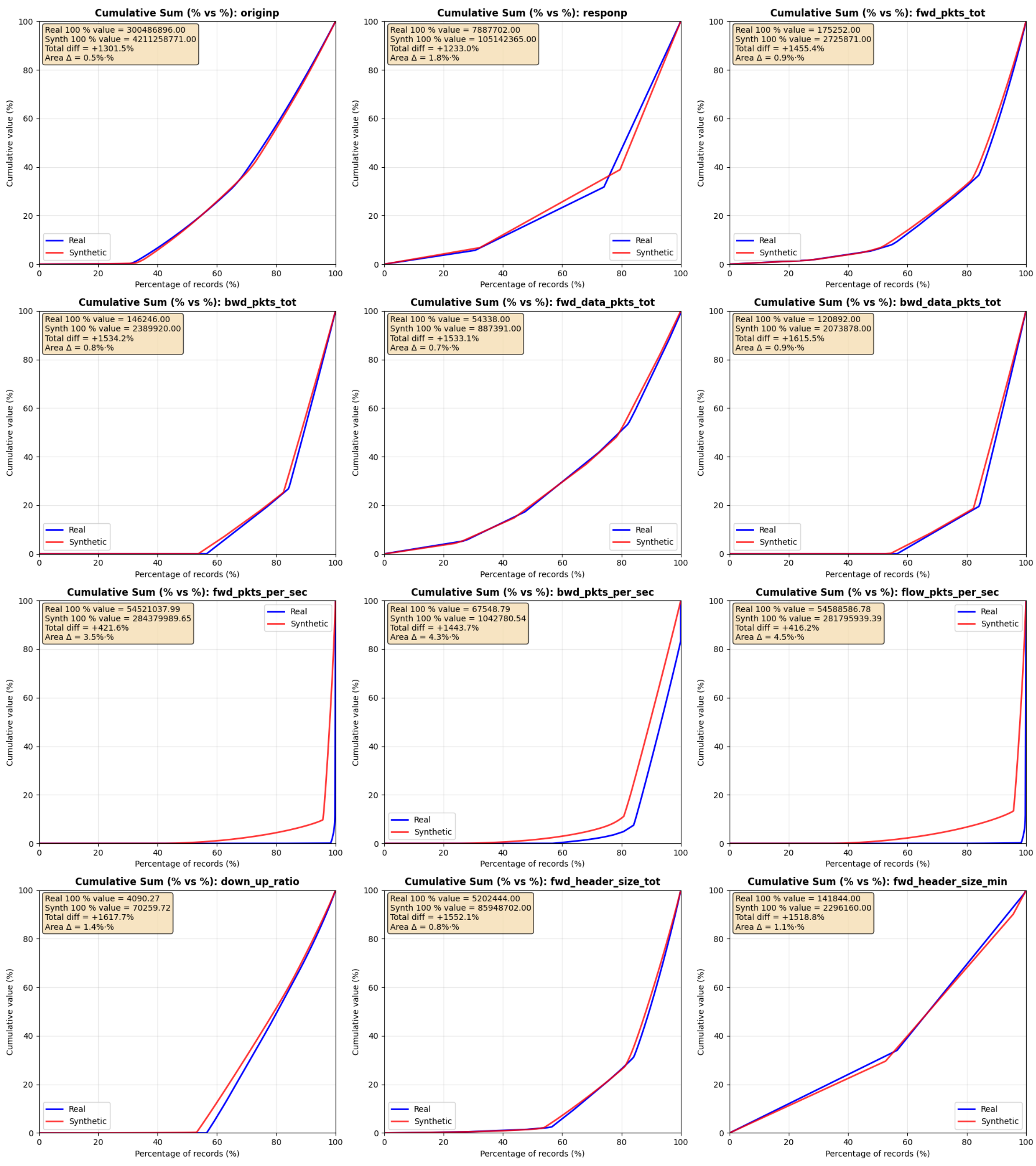


FIGURE B.5. Cumulative Sum after 10000 Epochs (first 12 features)

Figures from B.6–B.8 supports the analysis made in Section 5.3.1 where the goal is to have an overview of the new correlation that showed after creating synthetic data.

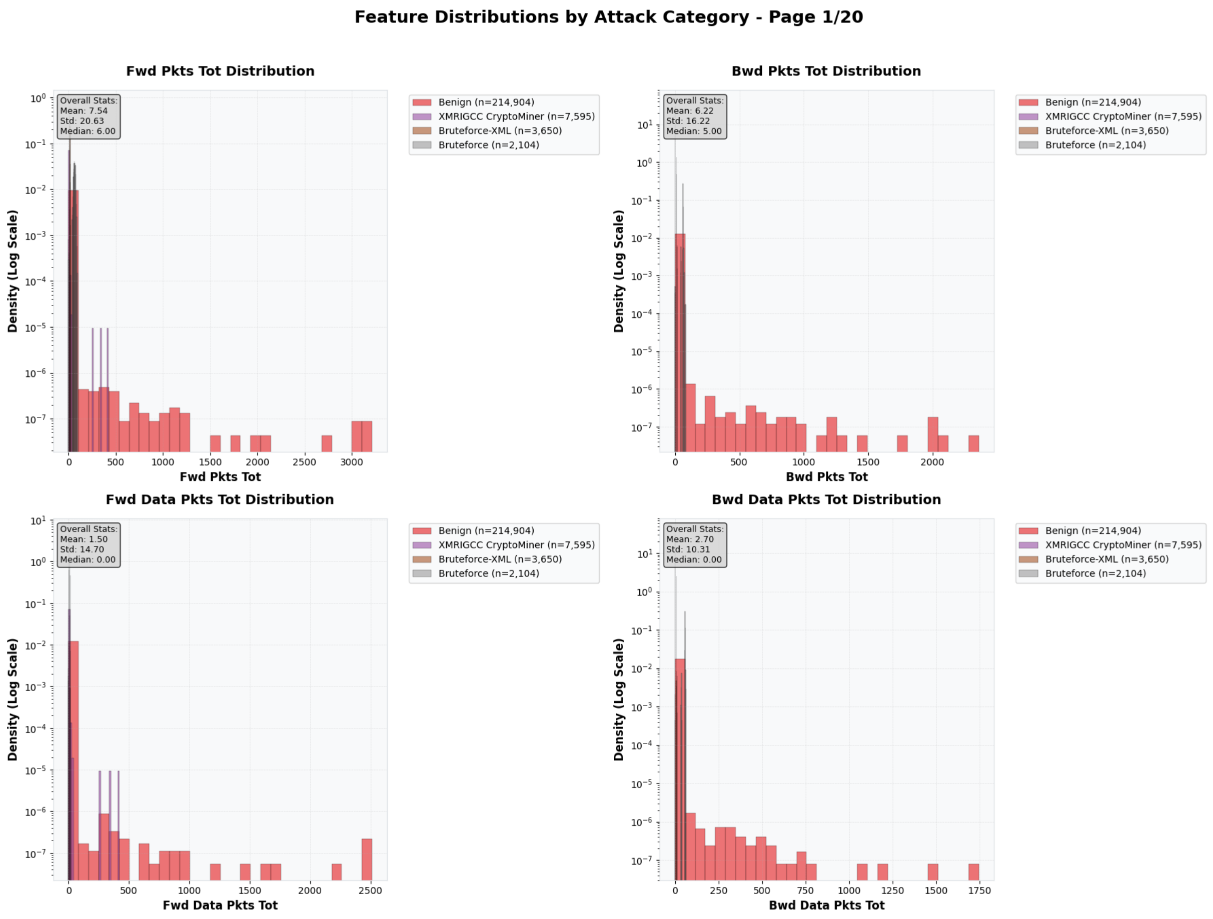


FIGURE B.6. Affected Features Distribution by attack category

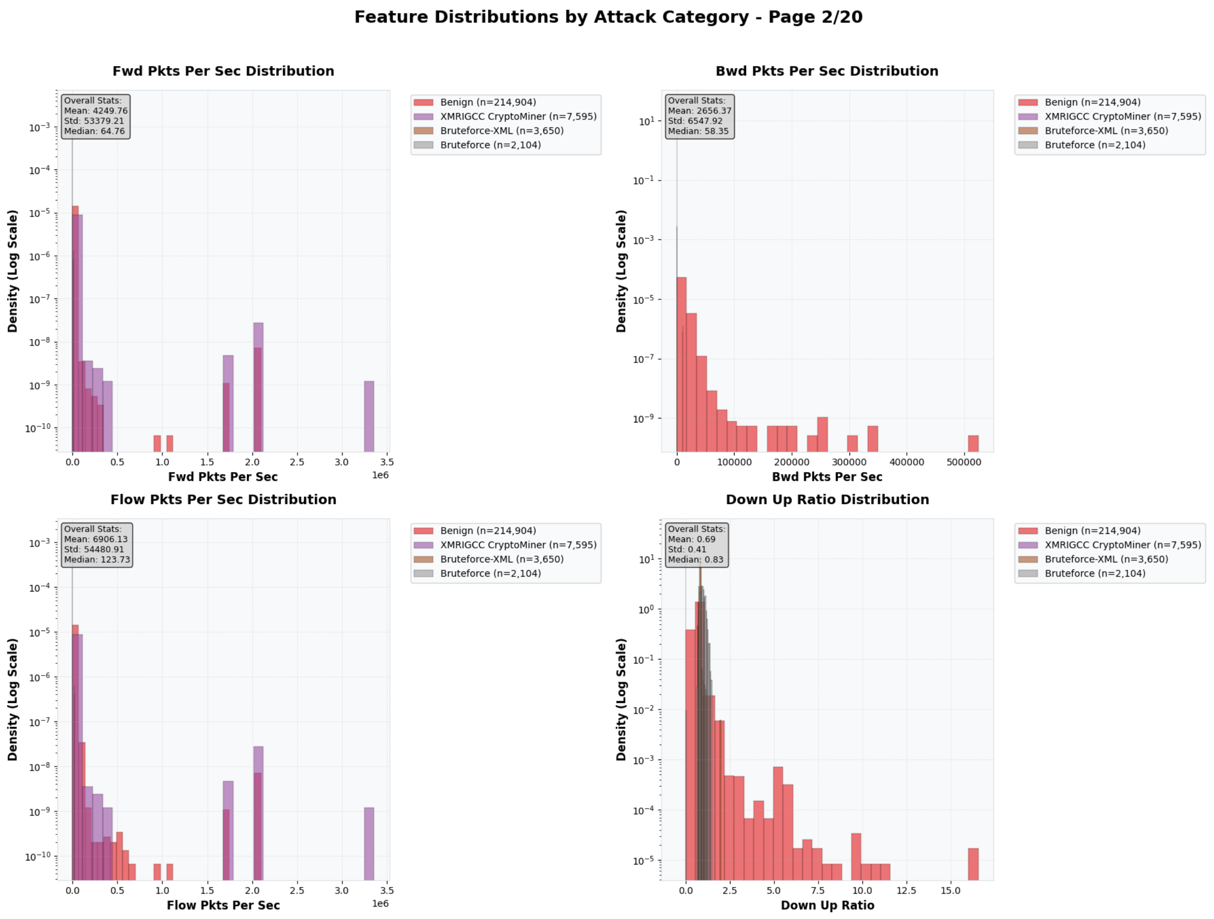


FIGURE B.7. Affected Features Distribution by attack category

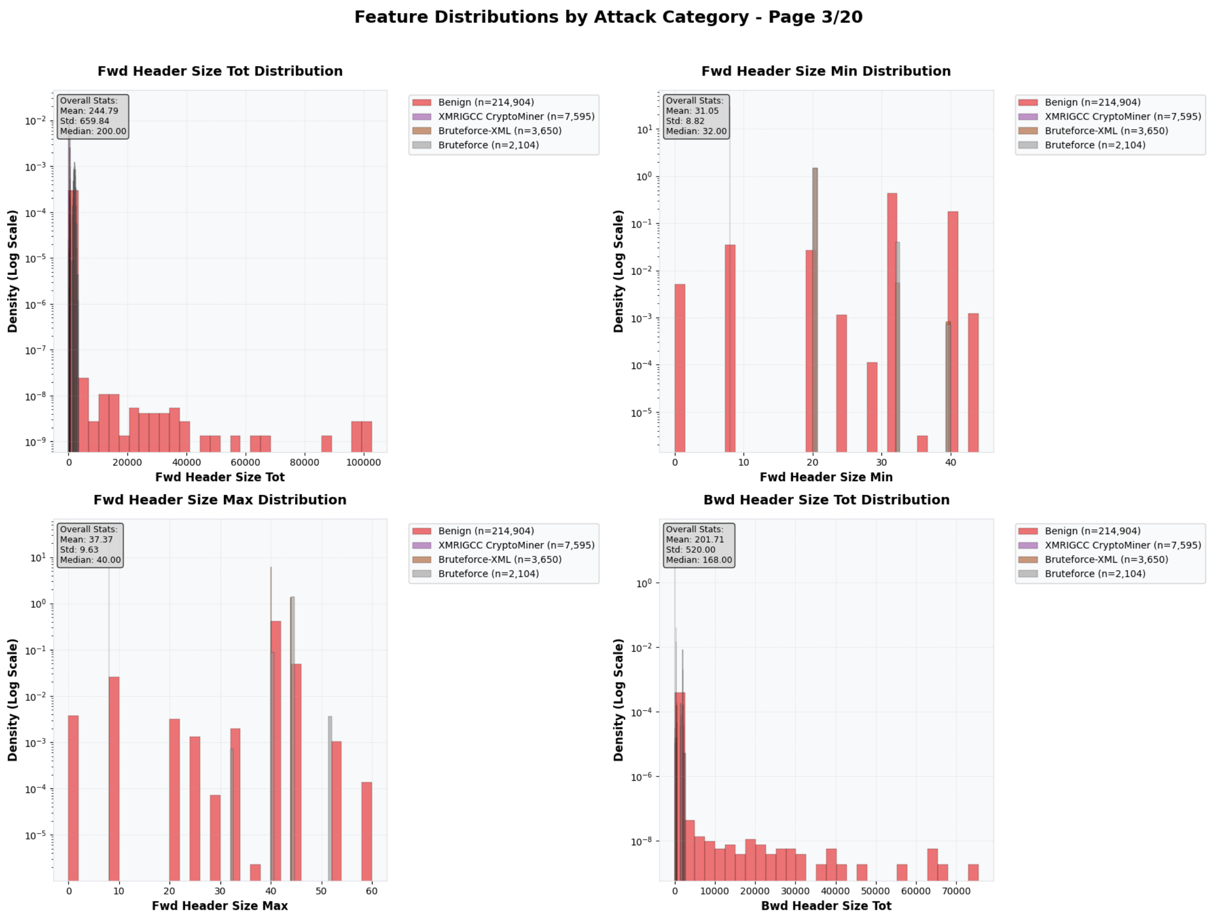


FIGURE B.8. Affected Features Distribution by attack category

Code Availabiltiy

More information and additional visualizations can be consulted in the code that supports this dissertation.

The code can be found in our Github repository:

- https://github.com/tmpsaatiscte/anomalydetectionmasters