Intelligent Based Network Intrusion Detection System Using Deep Learning Methods and Bioinspired Metaheuristics

UNIVERSIDAD DE GRANADA

Mohammed Ahmad Mahmoud Jamoos

Supervisor: ANTONIO MIGUEL MORA GARCÍA

To obtain the Ph.D. degree as part of the
Programa de Doctorado en Tecnologías de la Información y la Comunicación

May 2025

Editor: Universidad de Granada. Tesis Doctorales
Autor: Mohammed Ahmad Mahmoud Jamoos
ISBN: 978-84-1195-901-8
URI: https://hdl.handle.net/10481/108350

I dedicate this dissertation to my loving wife and wonderful children. Your unwavering love, support, and patience have been my greatest source of strength and inspiration throughout this journey. This achievement belongs to all of us. ...

In loving memory of my dear older brother, Ezz El-Din, whose unwavering support and encouragement inspired me to pursue my studies from a young age. May Allah have mercy on his soul, grant him Jannah, and reward him abundantly for his kindness and guidance. ...

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Mohammed Ahmad Mahmoud Jamoos

May 2025

Acknowledgements

I would like to express my heartfelt gratitude to all those who supported me throughout this research journey.

First and foremost, I extend my deepest appreciation to my advisor, Prof. Antonio Miguel Mora García, for his invaluable guidance, patience, and encouragement. His expertise and unwavering support have been instrumental in the successful completion of this thesis.

I am also sincerely grateful to Professor Imad Abu Kishk, President of Al-Quds University, for his encouragement and support, which have greatly contributed to my ability to achieve this academic milestone.

My deepest gratitude goes to my family for their unwavering support, patience, and understanding throughout this endeavor. I dedicate this work to my beloved wife and my wonderful children—Karam, Ezz El-Din, Omar, and Fares—whose love and encouragement have been my greatest source of strength and motivation.

To my parents, whose inspiration, sacrifices, and unconditional love have guided me every step of the way, I dedicate this thesis with immense appreciation and gratitude.

This work would not have been possible without the support and encouragement of these incredible individuals. I am truly fortunate to have had the opportunity to work alongside such dedicated and supportive people.

Abstract

Imbalanced data refers to an unequal ratio of classes, resulting in one major class and one or more suppressed classes. This imbalance can significantly impact the classifications made by machine learning models, which tend to become skewed and unreliable as they favor the dominant class while often ignoring the smaller classes. Deep neural networks have been widely utilized for data synthesis in various fields. Recent research, however, indicates that the efficacy of deep learning models can be enhanced with more balanced datasets. For high-dimensional data augmentation, Generative Adversarial Networks (GANs) are considered among the most effective tools. GANs have been one of Artificial Intelligence's (AI) most exciting innovations in the past decade. They have revolutionized the deep learning paradigm, driving some of the most significant technological advances in AI history. GANs are considered a major breakthrough in AI because they can generate entirely new data rather than merely replicating the training data.

Intrusion Detection Systems (IDSs) play a crucial role in enhancing network security by continuously monitoring network traffic and host systems to detect suspicious activities and potential security threats. The growing prevalence of cyberattacks and vulnerabilities in network systems highlights the need for automated and sophisticated IDSs. These systems are designed to learn the normal behavior of network traffic, enabling them to accurately identify anomalies.

Machine learning (ML) methods have demonstrated their effectiveness in identifying malicious activity within network traffic. However, the increasing volume of data generated by IDSs poses significant security challenges, necessitating the development of more advanced network security measures. Traditional machine learning techniques rely on balanced datasets, yet many IDS datasets suffer from imbalanced class distributions. This imbalance adversely impacts the accuracy of ML techniques, leading to the potential oversight of suspicious activities and an increase in false positives in traditional IDSs. Importantly, in real-world network environments, malicious traffic is inherently rare compared to normal traffic. This natural imbalance reflects the actual operational conditions of an IDS, making it

essential to work with imbalanced datasets. Addressing this challenge is crucial to ensuring the effectiveness of ML-based detection systems in recognizing both frequent and rare attack patterns while minimizing false alarm.

To effectively address the challenge of imbalanced datasets in network intrusion detection, this thesis presents a novel model, TDCGAN—a tailored generative adversarial network (GAN) architecture designed to enhance the detection rates of minority classes while maintaining computational efficiency. Unlike conventional GAN-based approaches, TDCGAN integrates a unique election layer to refine classification accuracy and improve decision-making.

The TDCGAN architecture consists of a generator and three discriminators, each contributing to the model's ability to capture complex patterns within network traffic data. The election layer, strategically positioned at the final stage, aggregates the outputs of the discriminators to determine the most reliable classification outcome. This novel mechanism reduces misclassification rates and ensures that the model effectively differentiates between normal and attack instances, even in highly imbalanced datasets.

To rigorously assess the effectiveness of TDCGAN, the UGR'16 dataset—a widely recognized benchmark for network security with a large-scale network traffic dataset designed for Intrusion Detection System (IDS) research—is used for evaluation and benchmarking. Collected from a real-world Internet Service Provider (ISP) network over several months in 2016, the dataset includes both normal traffic and various types of cyberattacks, making it valuable for evaluating machine learning-based intrusion detection methods. A comprehensive comparison is conducted against various machine learning algorithms (such as the synthetic minority oversampling technique (SMOTE) , Random oversampling, nearest neighbor (ENN) SMOTEENN, Borderline-SMOTE , SVMSMOTE, CGAN (conditional generative adversarial network) and CTGAN (conditional tabular generative adversarial networks), demonstrating the superiority of the proposed model. The results reveal that TDCGAN significantly outperforms traditional oversampling techniques and other baseline approaches in detecting security threats, particularly under challenging class distribution conditions.

By integrating an election layer into the GAN architecture, TDCGAN not only enhances the detection of minority-class attacks but also introduces a scalable and adaptive framework for intrusion detection systems (IDS). The experimental findings validate the model's capa-

bility to mitigate biases inherent in IDS datasets, thereby contributing to more reliable and effective cybersecurity solutions.

TDCGAN generates synthetic attack samples, enhancing the representation of minority attack classes. By doing so, it helps ML models:

- **Improve Detection Accuracy:** More attack samples allow models to learn diverse attack patterns, reducing false negatives.
- **Balance Class Distributions:**Ensures the ML model does not become biased toward normal traffic.
- **Enhance Generalization:** Provides varied attack scenarios, improving the robustness of IDS models against new or rare threats

The performance of TDCGAN was evaluated on four benchmark IDS datasets: CIC-IDS2017, CSE-CIC-IDS2018, KDD Cup 99, and BOT-IOT. This evaluation involved applying four machine learning classifiers (Decision Tree, Random Forest, Multi-Layer Perceptron (MLP), and Naive Bayes) first to the imbalanced datasets and then to the balanced datasets, highlighting TDCGAN's ability to address data imbalance challenges effectively.

The integration of TDCGAN has demonstrated significant improvements in addressing class imbalance and enhancing classification performance.

Resumen

Los datos desbalanceados se refieren a una distribución desigual de clases, lo que resulta en una clase principal y una o más clases suprimidas. Este desequilibrio puede afectar significativamente las clasificaciones realizadas por los modelos de aprendizaje automático, los cuales tienden a volverse sesgados e ineficaces al favorecer la clase dominante y, a menudo, ignorar las clases menores. Las redes neuronales profundas han sido ampliamente utilizadas para la síntesis de datos en diversos campos. Sin embargo, investigaciones recientes indican que la eficacia de los modelos de aprendizaje profundo puede mejorarse con conjuntos de datos más equilibrados. Para la ampliación de datos de alta dimensión, las Redes Generativas Antagónicas (GANs) son consideradas una de las herramientas más efectivas. Las GANs han sido una de las innovaciones más emocionantes de la Inteligencia Artificial (IA) en la última década. Han revolucionado el paradigma del aprendizaje profundo, impulsando algunos de los avances tecnológicos más significativos en la historia de la IA. Se consideran un gran avance en IA porque pueden generar datos completamente nuevos en lugar de simplemente replicar los datos de entrenamiento.

Los Sistemas de Detección de Intrusiones (IDS) juegan un papel crucial en la mejora de la seguridad de las redes mediante la monitorización continua del tráfico de la red y los sistemas anfitriones para detectar actividades sospechosas y amenazas de seguridad potenciales. La creciente prevalencia de ciberataques y vulnerabilidades en los sistemas de red resalta la necesidad de IDS automatizados y sofisticados. Estos sistemas están diseñados para aprender el comportamiento normal del tráfico de red, lo que les permite identificar con precisión las anomalías.

Los métodos de aprendizaje automático (ML) han demostrado su eficacia en la identificación de actividades maliciosas dentro del tráfico de la red. Sin embargo, el volumen creciente de datos generados por los IDS plantea desafíos de seguridad significativos, lo que requiere el desarrollo de medidas de seguridad en redes más avanzadas. Las técnicas tradicionales de aprendizaje automático dependen de conjuntos de datos equilibrados, sin embargo, muchos conjuntos de datos de IDS sufren de distribuciones desbalanceadas de

clases. Este desequilibrio afecta negativamente la precisión de las técnicas de ML, lo que puede llevar a pasar por alto actividades sospechosas y aumentar los falsos positivos en los IDS tradicionales. Es importante señalar que, en los entornos de red del mundo real, el tráfico malicioso es inherentemente raro en comparación con el tráfico normal. Este desequilibrio natural refleja las condiciones operativas reales de un IDS, haciendo esencial trabajar con conjuntos de datos desbalanceados. Abordar este desafío es crucial para garantizar la efectividad de los sistemas de detección basados en ML para reconocer tanto patrones de ataque frecuentes como raros, al mismo tiempo que se minimizan las falsas alarmas.

Para abordar de manera efectiva el desafío de los conjuntos de datos desbalanceados en la detección de intrusiones en redes, esta tesis presenta un modelo novedoso, TDCGAN, una arquitectura de red generativa antagónica (GAN) diseñada para mejorar las tasas de detección de las clases minoritarias mientras mantiene la eficiencia computacional. A diferencia de los enfoques convencionales basados en GAN, TDCGAN integra una capa de selección única para refinar la precisión de clasificación y mejorar la toma de decisiones.

La arquitectura de TDCGAN consta de un generador y tres discriminadores, cada uno contribuyendo a la capacidad del modelo para capturar patrones complejos dentro de los datos del tráfico de red. La capa de selección, posicionada estratégicamente en la etapa final, agrega las salidas de los discriminadores para determinar el resultado de clasificación más confiable. Este mecanismo novedoso reduce las tasas de clasificación errónea y garantiza que el modelo diferencie de manera efectiva entre instancias normales y de ataque, incluso en conjuntos de datos altamente desbalanceados.

Para evaluar rigurosamente la efectividad de TDCGAN, se utiliza el conjunto de datos UGR'16, un punto de referencia ampliamente reconocido para la seguridad en redes con un conjunto de datos de tráfico de red a gran escala diseñado para la investigación de Sistemas de Detección de Intrusiones (IDS). Recopilado de una red de un Proveedor de Servicios de Internet (ISP) en el mundo real durante varios meses en 2016, el conjunto de datos incluye tanto tráfico normal como varios tipos de ciberataques, lo que lo hace valioso para evaluar los métodos de detección de intrusiones basados en aprendizaje automático. Se realiza una comparación exhaustiva con varios algoritmos de aprendizaje automático (como la técnica de sobremuestreo de minorías sintéticas (SMOTE), sobremuestreo aleatorio, vecino más cercano (ENN), SMOTEENN, Borderline-SMOTE, SVMSMOTE, CGAN (red generativa antagónica condicional) y CTGAN (redes generativas antagónicas condicionales tabulares)), demostrando la superioridad del modelo propuesto. Los resultados muestran que TDCGAN

supera significativamente las técnicas tradicionales de sobremuestreo y otros enfoques base en la detección de amenazas de seguridad, especialmente bajo condiciones desafiantes de distribución de clases.

Al integrar una capa de selección en la arquitectura de GAN, TDCGAN no solo mejora la detección de ataques de clases minoritarias, sino que también introduce un marco escalable y adaptable para los sistemas de detección de intrusiones (IDS). Los hallazgos experimentales validan la capacidad del modelo para mitigar los sesgos inherentes a los conjuntos de datos de IDS, contribuyendo así a soluciones de ciberseguridad más confiables y efectivas.

TDCGAN genera muestras sintéticas de ataque, mejorando la representación de las clases de ataque minoritarias. Al hacerlo, ayuda a los modelos de ML:

- **Mejorar la precisión de detección:** Más muestras de ataque permiten a los modelos aprender patrones de ataque diversos, reduciendo los falsos negativos.
- **Balancear las distribuciones de clases:** Asegura que el modelo de ML no se sesgue hacia el tráfico normal.
- **Mejorar la generalización:** Proporciona escenarios de ataque variados, mejorando la robustez de los modelos IDS frente a amenazas nuevas o raras.

El rendimiento de TDCGAN se evaluó en cuatro conjuntos de datos de referencia de IDS: CIC-IDS2017, CSE-CIC-IDS2018, KDD Cup 99 y BOT-IOT. Esta evaluación implicó aplicar cuatro clasificadores de aprendizaje automático (Árbol de Decisión, Random Forest, Perceptrón Multicapa (MLP) y Naive Bayes) primero a los conjuntos de datos desbalanceados y luego a los conjuntos de datos equilibrados, destacando la capacidad de TDCGAN para abordar los desafíos del desbalance de datos de manera efectiva.

La integración de TDCGAN ha demostrado mejoras significativas en el tratamiento del desbalance de clases y en el rendimiento de clasificación.

Table of contents

List of figures

List of tables

Acronyms

AI Artificial Intelligence

AIDS Anomaly Intrusion Detection System

ANN Artifical Neural Network

Bi-RNN Bidirectional Recurrent Neural Network

CNN Convolutional Neural Network

DBN Deep Belief Network

DL Deep Learning

DNN Deep Neural Network

DT Decision Tree

ENN Edited Nearest Neighbor

FAR False Alarm Rate

GAN Generative Adversarial Network

HIDS Host Intrusion Detection System

NIDS Network Intrusion Detection System

IDS Intrusion Detection System

KNN k-Nearest Neighbors

LSTM Legression models

LSTM Long Short-Term Memory

ML Machine Language

MLP Multilayer Perceptron

NB Naive Bayes

PCA Principal Component Analysis

VAEs Restricted Boltzmann Machines

RF Random forest

RNN Recurrent Neural Network

SIDS Signature Intrusion Detection System

SVM Support Vector Machines

TDCGAN Triple Discriminator Conditional Generative Adversarial Network

VAEs Variational Autoencoders

Chapter 1

Introduction

1.1 Background and Motivation

The rapid digitization of our world has changed the complexity of networks through the proliferation of smart devices, the Internet of Things (IoT) [5], cloud computing, and big data. Although they offer many benefits, however, they have introduced new vulnerabilities susceptible to cyber attacks. The uncontrolled rise in zero-day attacks and data breach incidents, the most recent report recorded over 14 billion records [6], have been critical reasons for embracing more cybersecurity solutions. Thus, intrusion detection systems have become critical defensive mechanisms in securing networks, providing important systems and tools that identify and act against threats [5, 7].

Cybersecurity emerged as a significant research area during the IT revolution, encompassing various protective measures such as antivirus software, firewalls, and IDSs to safeguard systems against internal and external attacks [8]. IDSs operate by monitoring network traffic for suspicious activities, thereby playing a crucial role in ensuring network security [9]. Originally proposed by Jim Anderson in 1980 [10], IDSs have evolved significantly, with numerous approaches developed to enhance their effectiveness. However, the rapid expansion of network infrastructure and the increasing complexity of network applications have led to a surge in data traffic, creating new vulnerabilities that cybercriminals can exploit. This growing threat landscape necessitates continuous advancements in IDS technologies to enhance their ability to detect both known and novel attacks while minimizing the (FAR).

IDSs can be classified based on detection methods or deployment strategies. Detection methods include anomaly-based IDS and signature-based IDS. Anomaly-based IDS detects intrusions by identifying deviations from normal system behavior, leveraging machine learn-

ing and statistical analysis to establish baselines and flag unusual activities as potential threats. In contrast, signature-based IDS identifies threats by comparing network traffic against a database of known attack patterns, making it effective against known threats but less capable of detecting emerging attacks. From a deployment perspective, IDSs are categorized as host-based IDS (HIDS) and network-based IDS (NIDS). HIDS is installed on individual devices to monitor system logs, file integrity, and local activities, providing detailed insights into host-specific attacks. NIDS, on the other hand, is deployed at network entry points such as gateways or routers, analyzing network traffic for malicious activity to detect large-scale attacks affecting multiple devices. Combining different IDS types enhances security by providing a more comprehensive defense against cyber threats [11].

Given the increasing sophistication of cyber threats, this study adopts a machine learning-driven anomaly detection approach, specifically leveraging Generative Adversarial Networks (GANs) to address the challenge of class imbalance in IDS datasets. Traditional IDS models struggle with imbalanced datasets, where minority-class attack instances are overshadowed by the overwhelming presence of normal traffic. The proposed Triple Discriminator Conditional Generative Adversarial Network (TDCGAN) model aims to enhance the detection of rare and emerging threats by generating synthetic samples that balance the dataset while maintaining detection efficiency.

A dedicated section in this dissertation will provide a detailed discussion of IDS taxonomy, evaluating their advantages and limitations to contextualize the choice of a GAN-based anomaly detection approach. This justification is crucial to demonstrating how TDCGAN overcomes existing IDS challenges and contributes to improving network security.

Various classification methods have been employed to detect anomalous data in IDSs, including decision trees, rule-based systems, support vector machines, naïve Bayes, and nearest neighbors. More recently, researchers have increasingly adopted machine learning (ML) techniques to enhance IDS capabilities in identifying malicious activities. ML, a subset of AI, enables efficient extraction of useful insights from large datasets [8]. Supervised ML methods have shown considerable effectiveness in detecting malicious payloads in well-labeled network traffic datasets. However, the increasing complexity of network infrastructure and the rapid growth of data traffic have introduced new security challenges, requiring continuous improvements in IDS performance. Researchers have therefore focused on refining IDS models to enhance their detection rates for both known and novel attacks while reducing FAR, which refers to instances where benign network activities are incorrectly flagged as

malicious. Reducing FAR is crucial to prevent unnecessary alerts and allow security teams to focus on genuine threats [12]. Unsupervised intrusion detection systems offer an alternative approach by eliminating the reliance on labeled data [9]. These methods detect anomalies by analyzing deviations from the training data, but they become less effective when dealing with imbalanced datasets. The inherent imbalance in network flow data remains a significant challenge in cybersecurity, even in widely used benchmark datasets, where normal traffic is often overrepresented compared to malicious activity. Addressing this imbalance is critical, as traditional ML and DL algorithms struggle with minority-class detection in multiclass classification scenarios [5, 7].

Severe class imbalance negatively impacts ML models, leading to biased learning where the majority class (benign traffic) dominates, and minority-class attacks are often misclassified. Existing oversampling methods, such as the Synthetic Minority Oversampling Technique (SMOTE) and the Adaptive Synthetic Sampling Approach (ADASYN), attempt to mitigate this issue by generating synthetic attack samples through interpolation [13]. However, these methods often fail to capture the true distribution of real-world attacks, resulting in unrealistic samples that do not significantly improve detection performance.

To overcome these limitations, Generative Adversarial Networks (GANs) have emerged as a novel approach to data generation. GANs consist of two neural networks—a generator and a discriminator—engaged in a competitive process where the generator creates synthetic data, and the discriminator attempts to distinguish it from real data. This adversarial training process enables GANs to generate highly realistic data distributions, making them increasingly valuable for various applications, including cybersecurity [14–16].

Intrusion detection systems (IDSs) continue to play a vital role in network security, with ML-based approaches showing significant potential in detecting malicious activities. However, the severe class imbalance in real-world datasets remains a major challenge, causing issues such as:

- **Bias Toward the Majority Class:** Traditional ML techniques tend to favor the majority class (benign traffic) due to an unequal class distribution. As a result, attacks are often misclassified, leading to high false negative rates, which reduce the reliability of IDSs.
- **Ineffective Synthetic Data Generation:** Methods such as SMOTE, Random Oversampling, and standard GANs attempt to mitigate class imbalance by generating synthetic attack samples. However, these approaches often fail to capture the true distribution

of real-world attacks, leading to unrealistic data that does not contribute to improved detection performance.

To address these challenges, we propose TDCGAN (Triple Discriminator Conditional GAN) as a novel solution for enhanced intrusion detection in imbalanced datasets. Our approach introduces key innovations that improve upon existing methods:

- **Generation of High-Quality, Realistic Attack Samples:** TDCGAN learns the temporal dependencies of attack patterns, ensuring that generated samples closely resemble real attack behaviors. This improves the overall quality of synthetic data, making it more useful for ML models.
- **Better Class Balance for Improved Detection:** By specifically focusing on generating minority-class (attack) samples, our model ensures a more balanced dataset. This leads to enhanced model learning and higher accuracy in detecting attacks without compromising on generalization.

Our research directly addresses the limitations of existing intrusion detection methods by bridging the gap between class imbalance and ML model performance. By generating high-quality attack samples, TDCGAN contributes to:

- More effective and robust IDS models capable of identifying diverse cyber threats.
- Improved reliability and trustworthiness of ML-based detection systems.
- A scalable approach that can be applied to various IDS datasets and ML models.

The success and growing adoption of adversarial networks have led to their modification for data balancing applications. Our research demonstrates the effectiveness of a new oversampling technique based on GANs, applied to IDSs to address data imbalance. The proposed TDCGAN model consists of one generator and three discriminators, with an additional election layer at the end. The generator utilizes random noise from a latent space to produce synthetic data that closely resembles real network traffic while attempting to evade detection by the discriminators. By leveraging this structure, TDCGAN effectively mitigates the challenge of high-class imbalance, as demonstrated through an analysis of the UGR'16 dataset [5, 7].

1.2 Problem Description

Data science involves a multi-step process beginning with data collection, followed by preparation, exploration, and modeling. However, diverse problem domains yield diverse

datasets, which may contain different issues that must be addressed before proceeding to data modeling.

Accurately addressing these problems can greatly improve the model's accuracy. Additionally, machine learning methods are commonly applied in intrusion detection systems [11, 17]. IDS is applied to monitor the network traffic and detect unauthorized access attempts by analyzing incoming and outgoing actions [18].

Recent cyberattacks suggest the demand for automated and intelligent IDSs. The majority of IDS systems are designed to learn the normal behaviors of network traffic, allowing them to detect behaviors that deviate from the norm, indicating anomalous or suspicious activity. In this case, machine learning methods are proven to be effective in detecting malicious payloads in network traffic. However, the volume of data generated by IDSs creates security risks and demands the need for enhanced network security measures. Traditional machine learning methods depend on balanced datasets. Unfortunately, many IDS datasets face limitations due to the presence of imbalanced datasets resulting in hindering the effectiveness of machine learning techniques to miss detection and false alarms in conventional IDSs [5].

1.3 Purpose and Objectives of the Study

The primary objective of this dissertation is to develop a novel model-based (GAN) framework designed to generate high-quality synthetic datasets, thereby enhancing the detection rate of the minority class in imbalanced intrusion detection datasets. The proposed model aims to address the inherent class imbalance challenge that significantly impacts the performance of machine learning (ML)-based intrusion detection systems (IDSs), where rare yet critical cyber threats are often misclassified due to their limited representation in training data.

Unlike conventional oversampling techniques such as Synthetic Minority Oversampling Technique (SMOTE) and Adaptive Synthetic Sampling (ADASYN), which often fail to preserve the distributional characteristics of real-world cyber threats, this study introduces a novel TDCGAN model. By generating realistic and diverse attack patterns, TDCGAN aims to rebalance the dataset without introducing synthetic artifacts that could degrade IDS performance. This will ensure that the enhanced dataset improves the generalization capability of machine learning classifiers, leading to a higher detection rate of minority-class attacks without compromising efficiency.

To achieve this overarching goal, the research will focus on the following key objectives and milestones:

1. **Design and Develop a Novel GAN Model:**Propose a GAN-based approach tailored to generating high-quality synthetic data that preserves the statistical properties of original IDS datasets. Ensure the model enhances classification performance by improving detection rates of minority-class attacks.

2. **Optimize the GAN Training Process for Cybersecurity Applications:** Fine-tune critical aspects such as loss functions, discriminator architectures, and stability mechanisms to mitigate issues like mode collapse and ensure robust convergence.

3. **Evaluate Model Performance Using Benchmark Datasets:** Assess the effectiveness of the proposed approach using widely recognized IDS datasets, including UGR'16, UNSW-NB15, CICIDS2017, KDD-Cup 99, CSE-CIC-IDS2018, and BOT-IOT.

4. **Apply and Compare Machine Learning Classifiers:** Implement traditional and DL based classifiers (e.g., DT, RF, MLP, and NB) to measure IDS performance before and after applying the proposed data rebalancing method.

5. **Analyze and Compare Classification Results:**Investigate how dataset imbalance affects detection performance and evaluate the extent of improvement gained through GAN-generated synthetic augmentation, with a focus on minority-class detection accuracy.

6. **Explore Scalability and Adaptability:**Examine the model's potential for real-time and dynamic network environments beyond static datasets, ensuring broader applicability in modern cybersecurity settings.

Through the development of this novel GAN-based data balancing approach, this research aims to contribute to the broader field of AI-driven cybersecurity by demonstrating how adversarial learning can enhance intrusion detection capabilities. The findings will offer deeper insights into practical and scalable solutions for mitigating class imbalance in IDS datasets.

1.4 Thesis Structure

This thesis is structured as follows:

Chapter 2, Background, provides an in-depth discussion on Intrusion Detection Systems (IDS), including the concept and taxonomy of IDS. It explores detection and deployment methods-based IDS and details the (ML) and (DL) algorithms commonly applied in IDS. This chapter also covers traditional data balancing techniques and delves into the use of Generative Adversarial Networks (GANs) for data balancing in IDS.

Chapter 3, State of the Art, discusses the challenges and current discussions in the field of IDS. It provides a summary of related works that highlight various approaches to solving the challenges in network intrusion detection and balancing data.

Chapter 4, Benchmark Datasets, focuses on the evaluation metrics and key benchmark datasets used for model assessment. The chapter introduces the UGR'16 dataset and other significant datasets, detailing how they contribute to the study of IDS.

Chapter 5, Proposed Model TDCGAN, presents the proposed model for intrusion detection, TDCGAN. It explains the data preparation process, the setup of the model, and the training phase. The chapter also provides experimental results, demonstrating the effectiveness of the model.

Chapter 6, Comparative Analysis of the TDCGAN Model, provides a detailed comparative analysis of the TDCGAN model. It discusses the materials and methods used, including datasets and machine learning techniques, and elaborates on the experimental setup, data selection, feature selection, and evaluation metrics. The chapter concludes with the results and discussion of the comparative experiments.

Chapter 7, Conclusions and Future Work, summarizes the key findings of the research, highlights the main contributions, and provides recommendations for future research directions. It also includes a list of publications that have resulted from this study.

Chapter 2

Background

Network attacks are continuously increasing with the rapid advancements in technology. To ensure the security of the network, intrusion detection systems have been implemented. However, intrusion detection systems face a leading problem in class imbalance. Class imbalance is problematic as it introduces bias in models, causing them to favor the majority class. This issue presents a serious problem in areas like network security and in medical diagnosis [19]. An imbalance happens when the majority class is prioritized, inhibiting the minority class. This results in the minority class having a minimal performance. In medical diagnosis, a model trained on specific data can prioritize common conditions and fail to identify life-threatening conditions.

Furthermore, often accuracy, an evaluation metric, fails to assess the performance of models on imbalanced datasets. This results in deceptive with outcomes., leading to decisions based on false representations. Accuracy and fairness are two important factors in addressing class imbalance. Machine learning has become a powerful tool, transforming many industries. These algorithms are designed to learn from the data and improve their performance through data analysis. Machine learning utilizes many algorithms and statistical models to enable computers to complete tasks independently. These algorithms are capable of identifying and learning data patterns. This encourages automated decision-making and enhances predictive analysis [19].

Machine learning has become a prominent tool for automated detection of minority classes. Dependently, the performance of these techniques is significantly influenced by the quality and balance of the training data. Imbalanced datasets, when one class greatly outnumbers the other, create an issue for classifier frameworks. Traditional machine learning algorithms are optimized for balanced datasets, and when faced with imbalanced data, often

prioritize the majority class, resulting in suboptimal performance for the minority class. According to [20], Classification accuracy can be achieved by classifying all samples as belonging to the majority class. Similarly, Sanober argued that resampling of the data is an effective way to alter the distribution of datasets that are not balanced. By resampling the data, the classifier's subsequent performance can be improved [21]. However, this is contingent on effectively removing noise, reducing the imbalance degree, minimizing information loss, and preserving informative samples. To combat this challenge, numerous data augmentation techniques have been implemented, such as SMOTE [22], ADASYN [23], B-SMOTE [24], CGAN [25], Vanilla GAN [26], WGAN [27], and SDG GAN [28]. In an effort to balance the data, these methods are able to create artificial additional data to balance the majority and minority class distribution.

This artificial data is designed to balance datasets and improve the performance of models. Traditional classification techniques can be problematic with imbalanced datasets, where one class, the positive, is more heavily represented than the other class, the negative. These techniques favor the majority class, leading to better performance for it compared to the minority class. Addressing this class imbalance requires the use of SMOTE [22]. The SMOTE technique is used for oversampling which works to balance the dataset by synthetically generating new minority class samples. Nonetheless, there are a number of disadvantages when noisy and irrelevant samples are generated. On the other hand, SMOTE can generate noisy data if classes overlap and when the data sets are high-dimensional. To address these challenges, Borderline SMOTE and ADASYN have been developed [24].These methods can create synthetic data that closely resemble the original data but fail to capture the underlying data distribution, especially in cases where feature extraction is difficult. Additionally, SMOTE's focus on minority class can lead to overfitting, particularly when the minority class is disproportionately underrepresented. This can result in the creation of synthetic samples that do not improve model performance and may exacerbate overfitting. The increasing availability of large datasets has driven the incorporation of ML techniques. GANs have emerged as a powerful tool for approximating real-world data distributions, offering significant advantages in terms of flexibility and reliability. Additionally, they are adaptable and easy to understand. Researchers can use GANs to generate realistic synthetic data while minimizing costs. This feature allows GANs to be a promising technology in the area of machine learning applications [22].

Our current study focuses on creating a model-based (GAN) called TDCGAN, which aims to improve the detection rate of the minority class in imbalanced datasets while maintaining

efficiency. The TDCGAN model has generator and discriminator networks. The UGR'16 dataset is used to evaluate the model. The performance of the TDCGAN model is evaluated using various machine learning algorithms.

2.1 Intrusion Detection Systems

An IDS is essential for safeguarding internet users from harmful network threats. This system continuously observes network traffic to identify unusual or suspicious behavior and generates alerts when such activity is detected. An IDS are one of the major research application problems in the computer security domain. With the increasing number of advanced network attacks, improvement of the traditional IDS techniques becomes a challenge [11].

2.1.1 IDS Concept

The rapid growth of network and internet technologies makes cybersecurity a crucial field of study. This field comprises numerous, protective measures, including antivirus software, firewalls, access control schemes and Intrusion Detection Systems (IDS). All aimed at protecting systems from internal and external threats while strengthening the security of information and communication systems. The primary goal of intrusion detection is to monitor network assets and identify anomalous behavior or misuse.

Over the years, numerous anti-intrusion technologies have been developed to stop Internet attacks. IDS is one of the seven anti-intrusion systems, along with avoidance, preemption, deterrence, deflection, detection, and countermeasures [29]. Accurate detection of an incursion is the most crucial of these aspects of intrusion prevention..

Intrusion refers to any attempt that leads to an adversary gaining unauthorized access to information, systems, or networks. This can jeopardize the confidentiality, integrity, or availability of information [30, 31]. Detection refers to the process of identifying unauthorized access to a system or network. This involves monitoring systems for any anomalies or suspicious activity.

Additionally, IDS is a security system that constantly monitors network traffic and hosts to detect any security violations or illegal activity. IDS generates alerts for any detected intrusion. and responds to these alerts [32]. Typically, the IDS are deployed near the network nodes with the purpose of monitoring network hosts and to allow the network traffic pass.

Different IDS “flavors” detect suspicious behavior in more than one way.

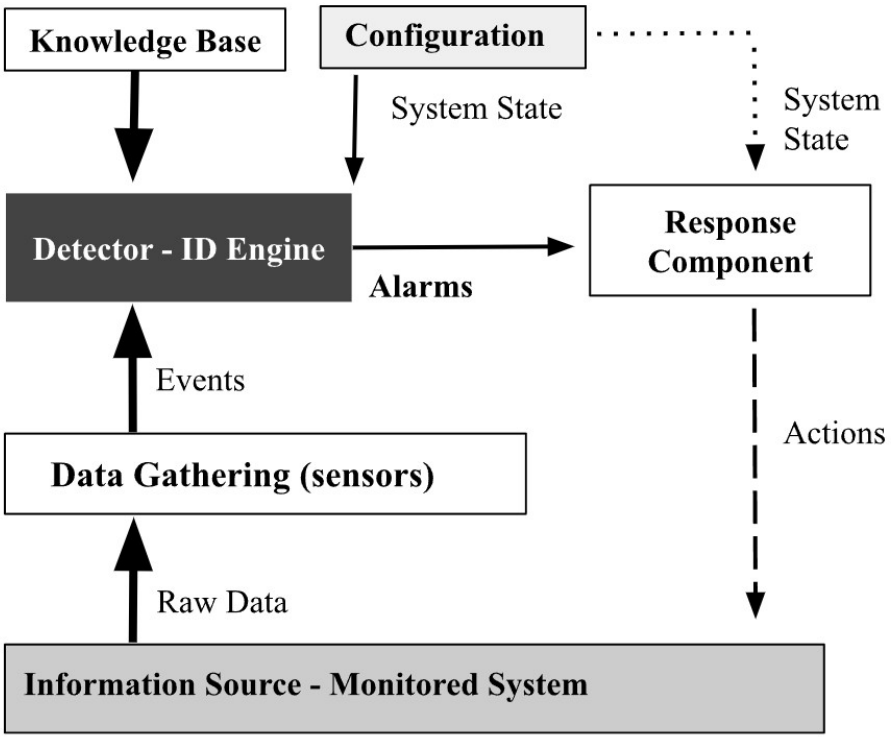


Fig. 2.1 Basic architecture of intrusion detection system (IDS).[1]

Figure 2.1 illustrates how the IDS system works. Initially, the data gathering device (the sensor) grabs data from the system it’s monitoring. That data then is passed to the detector, which looks through it to find anything that seems off or suspicious. The knowledge base holds all the relevant information, like attack patterns and data profiles, that has been cleaned up and organized by experts. The configuration device monitors the system’s status, ensuring everything is functioning properly. When the detector catches an intrusion, the response component takes action. It can either act automatically or, in some cases, request human intervention, depending on the setup [1, 33].

IDS can be categorized into two main types: (HIDS) and (NIDS). HIDS monitors the activity of individual devices, and NIDS analyzes network traffic at specific points to identify potential threats [34].

The concept of IDS has existed for nearly twenty years, but only recently has it seen a dramatic rise in popularity and incorporation into the overall information security infrastruc-

ture.
Introduced in Jim Anderson's seminal paper in 1980, the concept of (IDS) has been prominent in the development of network security. The exponential growth of network traffic and the increasing complexity of modern applications have demanded the development of more sophisticated IDS solutions. For this reason, researchers are focused on improving detection accuracy and minimizing (FAR) to address these developing risks.
Various classification methods, including decision trees, rule-based systems, naïve Bayes, and nearest neighbors, have been utilized in the domain of intrusion detection systems (IDS). Machine Learning (ML) has proven to be a very effective way to enhance the mechanism of IDS. By incorporating ML into an IDS, it can effectively detect and categorize various suspicious activities [11].

Furthermore, the advanced subset of ML, DL, has been shown to be capable of processing large datasets. DL architectures are adept at learning complex feature representations from raw data, which improves performance in various tasks. Additionally, DL approaches are needed to protect IDS from threats to network security.

2.1.2 IDS Taxonomy

IDS can be classified in two ways, either by the detection method used in the IDS or by the deployment method used in IDS [11]. The IDS is subclassified into two groups based on its detection methods: anomaly detection-based IDS and signature-based IDS. The second group is based from its deployment method-based IDS perspective, the IDS is classified as host-based IDS and network-based IDS [35]. The details are given in the next subsections.

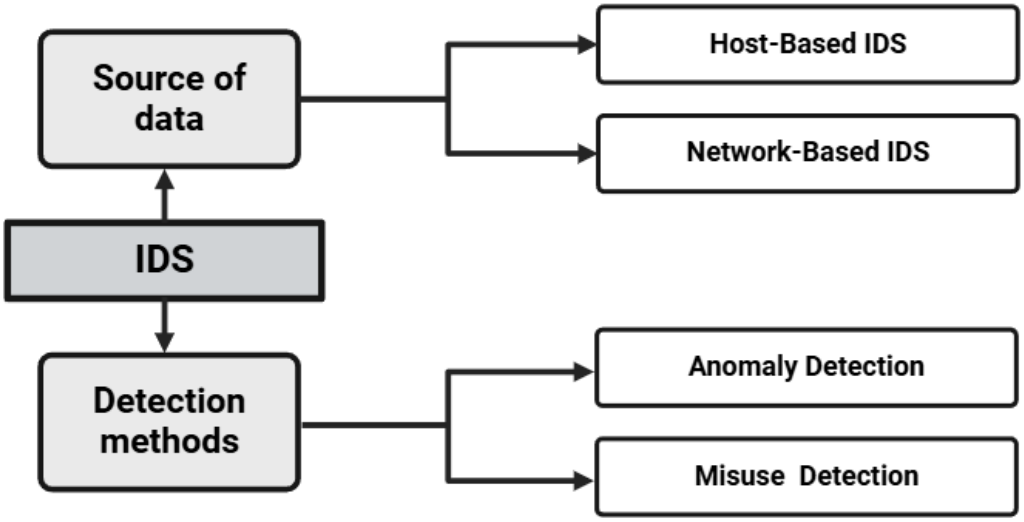


Fig. 2.2 IDS taxonomies.(Source: Author own creation)

Figure 2.2 breaks down how Intrusion Detection Systems (IDS) are organized. There are two types based on how they detect threats: anomaly-based IDS, which looks for behavior different from the normal, and signature-based IDS, which detects threats by matching them to known patterns. On the other hand, IDS can also be classified based on where they're used: host-based IDS that monitor individual computers or devices, and network-based IDS, which keep track of traffic across the whole network. This provides a straightforward overview of how IDS systems operate in different ways, depending on the threats they aim to detect and their deployment.

2.1.2.1 Detection Method-Based IDS

The detection method-based IDS is further subdivided into two primary categories: Signature-based IDS (SIDS) and Anomaly-based IDS (AIDS).

- Signature-Based IDS (SIDS): The term 'signature-based detection' is also known as misuse detection or abuse detection, which works by storing a signature for each attack pattern in the database and comparing any suspicious data patterns with these stored signatures to detect signature-based attacks [34, 2]. For any known attack,(SIDS) can detect it with high accuracy. However, for unknown or new attacks, the system fails to identify the attack pattern as it cannot be matched with any of the stored signature patterns in the database. Another disadvantage for SIDS is that it is a resource consuming system for large signature databases, the comparison between

stored signatures and data packet may increase time complexity which reduces overall performance [36].

- Anomaly-Based IDS (AIDS): which is also called behaviour-based IDS, depends on creating a profile for each normal activity and defines the abnormal activity as the degree of deviation from the normal activity profile [37]. "Despite AIDS' ability to detect abnormal attacks, it suffers from a high (FAR) due to its inability to accurately determine the boundaries between normal and abnormal attack profiles [2, 38].

Abnormal behavior and system faults can be detected using these two techniques: Abuse Detection and Anomaly Detection.

Both of these detection systems are for the purpose of strengthening network security, however, they are distinct in their function and the way they detect suspicious behaviour. Abuse Detection is precise in detecting familiar attacks by comparing them with previous network traffic patterns of known attacks. However, unexpected attacks are detected Anomaly Detection. Anomaly detection is considered to be most accurate in detecting normal and even unforeseen attacks. Additionally, like any other techniques, both of these methods have their own strengths and weaknesses as their mechanisms are different from each other. Abuse detection is considered to be weak against attacks it has not previously observed indicating its limitations. Whereas anomaly detection has been studied to be susceptible to false positives [2].

Table 2.1 Differences between Signature-Based Detection and Anomaly-Based Detection.

Metric	**Signature-Based Detection**	**Anomaly-Based Detection**
Detection Proficiency	Minimal false positives; High false negatives.	Low false negatives; High false positives.
Detection Efficiency	High, decreasing as the signature database size increases.	Varies with the complexity of the model.
Reliance on Domain Knowledge	Almost all detections rely on domain knowledge.	Minimal, with feature design being the only aspect reliant on subject matter expertise.
Unknown Attack Detection	Only detects known attacks.	Detects known and unknown attacks.
Interpretation	Excellent interpretive skills and domain knowledge-based design.	Produces just detection results and has poor interpreting skills.

Table 2.1 compares Signature-Based and Anomaly-Based Detection methods. Signature-Based Detection has high efficiency but struggles with unknown attacks and heavily relies on domain knowledge. In contrast, Anomaly-Based Detection detects both known and unknown attacks but has a higher (FAR) and lower interpretability. This highlights the trade-off between accuracy and adaptability in intrusion detection systems [39].

2.1.2.2 Deployment Method-Based IDS

From the perspective of Classification Based on Data Source, IDS can be classified into two main types: (HIDS) and (NIDS) [35].

- Host-Based IDS (HIDS): can watch the actions of important objects, they have the ability to precisely locate intrusions and launch replies (e.g., sensitive files, programs, and ports), it is deployed separately on every single host in the network, responsible on monitoring the activities of this host and detecting any suspicious behavior. The main disadvantage of this type is the extra-processing overhead that results from the deployment of IDS at each host in the network which reduces the overall performance of the system. Major servers or switches are typically used to deploy a network-based IDS [40].
- Network-Based IDS (NIDS): the IDS is deployed over the network and is responsible for monitoring the traffic in the network to detect any attack that passes through the network in real-time. NIDS is deployed in the major hosts in the network, and it can be applied in different operating system environments. A summary of the main advantages and disadvantages of each type of IDS is given in Table 2.2 [2].

As previously mentioned, Host-Based Intrusion and Network-Based Intrusion are two types of IDSs. Host-Based Detection is installed on each server and works by monitoring individual hosts for unusual behaviour. Network Based Intrusion tracks the monitor network traffic and is installed on network devices. For high network performance and security, both IDS systems are implemented together.

2.1.3 ML and DL Algorithms for IDS

ML can be further classified into three types based on the learning method: supervised, unsupervised, and reinforcement learning. To train a computer with our own categorization

Table 2.2 Differences between Host-Based and Network-Based IDSs.

Metric	**Host-Based IDS**	**Network-Based IDS**
Data Source	Operating system and application event logs.	Network traffic.
Deployment	Every host; dependent on operating systems; challenging to deploy.	Important network nodes; deployable.
Precision of Detection Knowledge	Low, many logs must be processed.	High, real-time attack detection rate.
Traceability of Intrusions	Track the intrusion process using the system call pathways.	Utilize IP addresses and timestamps to track the location and date of an intrusion.
Constraint	Unable to examine network behavior.	Only monitor traffic going through a certain network segment.

scheme, supervised learning is often used to overcome classification challenges. Unsupervised learning seems to be more challenging. The goal is to teach the computer how to accomplish a task that we have not explicitly instructed it to perform. It is a useful tool for detecting structure in unsupervised learning, as it represents the statistical properties of the entire set of input formats. Instead of being explicitly taught, reinforcement learning entails the learning agent interacting with the environment and learning by doing by seeing the effects of its actions [11].

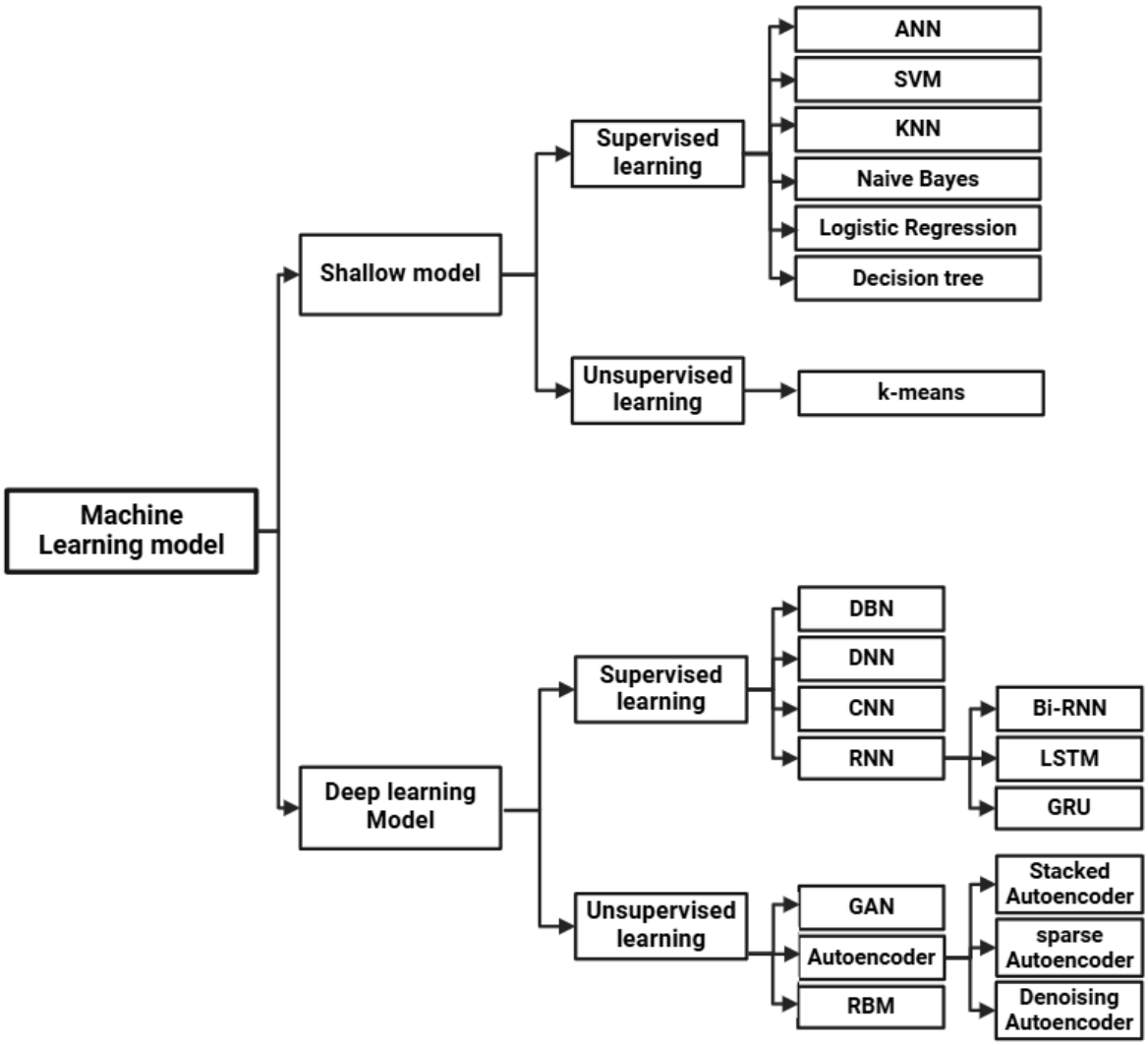


Fig. 2.3 Models for machine learning [2].

According to above Figure 2.3,Machine Learning Models are categorized into two models: Shallow and DL. Shallow and DL models can be subcategorized into Supervised and Unsupervised learning, Supervised learning under the Shallow model include, but are not limited to: Logistic Regression, NB, DT, ANN, etc… Unsupervised learning of the Shallow method includes K-means.

On the contrary, Supervised learning under the DL model category includes: DBN, DNN, CNN, and RNN. RNN includes subcategories: Bi-RNN, LSTM, and GRU. Additionally, Unsupervised learning for the DL model includes: GAN, RBM, and Autoencoder. Autoencoder has three categories: Stacked Autoencoder, Sparse Autoencoder, and Denoising Autoencoder

[2, 41] .

The two essential categories of machine learning are: supervised and unsupervised learning. Classification is the primary function in supervised learning and intrusion detection systems; yet, the human labeling of data is both expensive and time-consuming. The primary obstacle to supervised learning is a lack of adequate labeled data. Unsupervised learning, on the other hand, extracts significant features from unlabeled data, hence aiding in the acquisition of training data. Unsupervised learning approaches typically exhibit worse performance in detection relative to supervised learning methods [42].

In general, machine learning methods can be classified into two main groups:

2.1.3.1 Traditional machine learning algorithms

Shallow learning, also known as traditional machine learning algorithms, include several methods that have been observed to upgrade IDS. It includes Artificial Neural Network (ANN), Support Vector Machine (SVM), K-Nearest Neighbor (KNN), naïve Bayes, regression models (LR), decision trees, clustering, in addition to mixed and hybrid techniques [43].

- **Artificial Neural Network (ANN):** ANN are meant to familiarize the structure and function of human brains. They consist of an input, output, and one or more hidden layers. These are interconnected layers. Units in neighboring strata are completely interconnected. ANNs are powerful tools that learn and represent complex patterns. Training ANNs is a time-consuming operation due to backpropagation algorithms which is not suitable for the training of deep networks. An ANN is so distinct from DL models and belongs to the category of shallow models [44].

- **Support Vector Machines (SVM):** The strategy of SVM is to find the hyperplane farthest from the nearest data points of both classes. This method allows SVMs to reach high accuracy with limited data. Nonetheless, SVMs can be sensitive to noise and outliers near the decision boundary. To address the challenge of nonlinear separable data, kernel functions are used to project the data into a higher dimensional feature space. Both SVMs as well as other machine learning methods commonly use kernel techniques [45].

- **k-Nearest Neighbors (KNN):** is a machine learning classifier that predicts the class of a data based on the idea of feature similarity. KNN identifies a sample based on its distance from the neighbours. The parameter k affects KNN performance [46].

- **Naïve Bayes:** is a simple but effective tool for classification that's based on Bayes' theorem. The "naive" part comes from its assumption that all features are independent, which isn't always true. Still, it tends to work really well, especially in areas like text classification. It works by looking at the features of the data, calculating how likely each class is, and then picking the class with the highest probability [47–49].

- **Decision Tree (DT):** is a popular supervised learning method used for both classification and regression tasks. It works by splitting the data into different regions based on feature values. The structure of a (DT) consists of nodes, branches, and leaves. Each node represents a feature, the branches show the rules for splitting, and the leaves represent the final class or outcome. The tree is automatically built by selecting the best features, and pruning is applied to remove unnecessary branches [50].Random RF and XGBoost are more advanced algorithms that build on decision trees by using multiple trees to make more accurate predictions [51, 52].

- **Multi-Layer Perceptron (MLP):** is a type of neural network that works by processing data through multiple layers. These layers include an input layer, one or more hidden layers, and an output layer. Each node in a layer is connected to every node in the next layer, with each connection having a weight that determines how important that connection is. The network learns by adjusting these weights to reduce the error between its predictions and the actual results, a process that happens through backpropagation and gradient descent. To put it simply, an MLP is a neural network that passes data from the input layer, processes it in one or more hidden layers, and then gives an output. The hidden layers are responsible for figuring out the patterns in the data before it's passed along to make a prediction [53–55].

- **K-means:** is a widely used clustering algorithm, where "K" represents the number of clusters and "means" refers to the average attributes within each cluster. The algorithm relies on distance as a measure of semantic similarity, grouping data objects closer in proximity into the same cluster. As the distance between two data points decreases, their likelihood of being assigned to the same cluster increases. While K-means performs effectively with linear data, its performance declines with nonconvex datasets, often yielding suboptimal results. Moreover, the algorithm's outcomes are influenced by its initial conditions and the choice of the parameter K. Determining the optimal value for K typically requires multiple iterations and repeated testing to achieve reliable results [2].

- **Ensemble:** A sensible approach to improving classification accuracy is to integrate multiple weak classifiers into a single robust classifier. This involves combining multiple learning algorithms, each with its strengths and weaknesses, and using ensemble methods where several classifiers are trained, and their outputs collectively determine the final prediction through a voting mechanism. Hybrid methods further enhance this approach by incorporating multiple stages, each utilizing a distinct classification model. Researchers are increasingly exploring ensemble and hybrid classifiers, as these methods often deliver superior performance compared to individual classifiers in classification tasks. A critical aspect of this process lies in selecting the appropriate classifiers to combine and determining the optimal strategy for their integration [56, 57].

Table 2.3 Benefits and Drawbacks of Various Shallow Models.

Algorithms	Advantages	Drawbacks	Improvement Measures
Artificial Neural Network	Capable of handling nonlinear data with exceptional fitting capabilities.	Prone to overfitting, vulnerable to getting trapped in local optima, and requiring extensive training time.	Improved optimization techniques, activation functions, and loss functions.
Support Vector Machines	Acquire insights from a simple training set with strong generative capabilities.	Performs poorly on tasks with large datasets or diverse classifications, with kernel function parameters being highly sensitive.	Parameter optimization aided by Particle Swarm Optimization (PSO).
k-Nearest Neighbor	Applicable to large datasets, efficient for nonlinear data processing, and robust against noise.	Extended testing times, low accuracy for minority classes, and high sensitivity to the parameter K.	Trigonometric inequality reduces comparison time, Particle Swarm Optimization (PSO) optimizes parameters, and Synthetic Minority Over-Sampling Technique (SMOTE) balances datasets.
Naïve Bayes	Resilient to noise and capable of incremental learning.	Weak performance related to attributes.	Latent variables have been incorporated to relax the assumption of independence.

Table 2.3 continued on next page

Table 2.3 (continued)

Algorithms	Advantages	Drawbacks	Improvement Measures
Linear Regression	Fast to train and straightforward; automatically scales features.	Struggles with nonlinear data; suitable for handling oversized datasets.	Data regularization to prevent overfitting.
Decision Tree	Automated feature selection with strong interpretability.	Classification results tend to favor the majority class, overlooking the correlations within the data.	SMOTE-balanced datasets; latent variables added.
K-means	Excellent scalability, simple and fast training, capable of fitting large datasets.	Performs poorly with non-convex data, but capable of initialization and sensitive to the K parameter.	Improved initialization technique.
Multi-Layer Perceptron (MLP)	Handles nonlinear data effectively.	Time-consuming and prone to local optima issues.	Optimizing the network architecture and using techniques to improve performance and avoid overfitting.

Table 2.3 demonstrates Shallow Learning Models' advantages and disadvantages as well as areas of improvement. For example, the benefits of NB include: simplicity in assuming independence between features and efficiency in training and prediction. However, it can be sensitive to features that do not show in the training data, which makes up its disadvantage. Another example is (DT), where its advantages include: clear to understand, efficient in handling categorical and numerical data. Its disadvantages include: bias towards majority classes, and delicacy towards small changes in the training data. The figure below shows the nature of each of the Shallow learning methods, their strengths and weaknesses.

2.1.3.2 Deep learning algorithms

Different deep networks make up DL models. The DL algorithms include many layers in the architecture to the characteristics of the data and learn more features on their own. Among them, autoencoders, and generative adversarial networks (GANs) are unsupervised learning

models. Deep neural networks (DNNs), recurrent neural networks (RNNs), convolutional neural networks (CNNs), are supervised learning models [11].

Since 2015, there has been a significant increase in research on DL based (IDSs). DL models autonomously acquire visual features from raw data, including photos and texts, without requiring manual feature engineering. Consequently, DL techniques can be employed in an end-to-end manner. DL methodologies substantially surpass shallow models when used on large datasets [11, 2].

- **Recurrent Neural Network (RNN):** is a DNN that consists of input layer, output layer and hidden layer with one or more feedback loops. The hidden layer contains states and memory blocks to store, remember and process past data for a long period of time [57].
- **Convolution Neural Networks (CNN):** is a DL algorithm that consists of convolutional and pooling layers. CNN is designed to operate with multi-dimensional image data [58].
- **Autoencoder:** A specific type of neural network with three components: encoder, code and decoder. The encoder compresses the data to produce code and decoder reconstructs the input using this code [59].
- **Generative Adversarial Network (GAN):** is a generative modelling that uses DL algorithm such as CNN to extract patterns from input data and use it to generate new samples [26, 60]
- **Deep Neural Network (DNN):** An approach called "tier pretraining and fine-tuning" can be used to construct DNNs with many layers, DNN is trained by initially learning its parameters from unsupervised learning, which is known as an unsupervised learning stage. Labeled data are subsequently used to refine the network's performance. This unsupervised learning stage plays a prominent role in the remarkable success of DNNs [2].

Table 2.4 show the comparison of various DL models based on their strengths, weaknesses, and best use cases.

Table 2.4 Comparison of Various Deep Learning Models.

Algorithms	Strength	Weakness	Suitable Data Types	Supervised or Unsupervised	Functions
Recurrent Neural Networks (RNN)	Remember the previous information and use them to predict future.	Performance depends on the time lag value.	Raw data.	Supervised.	Sequence data supervised feature extraction; classification.
Convolutional Neural Networks (CNN)	Powerful in detecting and extracting complex features.	Performance depends on the kernel size. Time consuming for large data.	Raw data.	Supervised.	Matrices supervised feature extraction.
Autoencoder	Reduce dimensionality of the data.	Time consuming for large data.	Raw data; feature vectors.	Unsupervised.	Feature extraction; feature reduction; denoising.
Generative Adversarial Network (GAN)	Learn the internal representation of the data.	Optimizing the network requires many trial-and-error attempts.	Raw data.	Unsupervised.	Unsupervised data augmentation; adversarial training.
Deep Neural Network (DNN)	Ability to automatically learn complex, hierarchical representations of data.	High computational cost and resource requirements; training time-consuming.	Raw data; feature vectors.	Supervised.	Feature extraction; classification.

DL methods are robust and powerful in the learning ability because of its internal structure with multi hidden layers that enable the model to extract a useful feature from a complex and huge dataset. The performance of DL methods is superior to ML methods. However,

there are distinct properties between DL and ML methods that influence the decision of which method to implement. These differences can be summarized as follows [11]:

- **Running time:** because of multi hidden layer architecture, DL methods consume more running time for learning.
- **Hyperparameters tuning:** DL methods contain more hyperparameters that need to be tuned efficiently before training the model.
- **Learning capacity:** DL methods are robust and can learn from a large volume of data because of its complex structure.
- **Interpretability:** DL methods are black-box models that generate the output without an interpretation.

2.2 Data Balancing Techniques in IDS

2.2.1 Overview of traditional methods

Data balancing in Intrusion Detection Systems (IDS) is essential due to the prevalence of class imbalances in real-world datasets, wherein instances of attacks (minority class) are markedly less than normal traffic (majority class). This imbalance can significantly affect the efficacy of machine learning algorithms, since they often exhibit bias towards the majority class, resulting in elevated false negatives for attacks. Conventional data balancing methods seek to mitigate this issue by either oversampling the minority class, undersampling the majority class, or employing a hybrid approach of both strategies.

Oversampling entails duplicating instances from the minority class to equilibrate the class distribution, a technique frequently employed in methods such as SMOTE (Synthetic Minority Over-sampling Technique), which creates synthetic samples by interpolation between existing minority class instances. Conversely, undersampling diminishes the quantity of cases from the majority class, which can be effective but may result in the loss of significant data. Hybrid methodologies, such as SMOTE-Tomek, integrate both oversampling and undersampling techniques to enhance classifier efficacy while preserving data diversity. Nonetheless, these conventional techniques possess limitations, including the potential for overfitting due to oversampling and the forfeiture of valuable data through undersampling. Notwithstanding these limitations, they continue to be fundamental methods for tackling class imbalance challenges in Intrusion Detection Systems (IDS).

- **The synthetic minority oversampling technique (SMOTE):** SMOTE has proven its efficiency across various applications and among several fields since its publication in 2002. The SMOTE technique is a method used in oversampling that creates artificial instances from the lower class. It is introduced to reduce the shortcomings faced by the random over sampling method by creating a training set that is either synthetically balanced or close to balance in distribution, which is subsequently utilized for classifier training. It performs significantly better with small datasets, and takes a substantial amount of time to produce artificial data points in large datasets. However, oversampling big datasets also allows for SMOTE's efficiency to decrease. Moreover, the chance of overlapping data points in the minority class is strong when artificial data points are created [22].

Each sample, x_i, is selected as a root sample selection. When that is completed, the process randomly selects a sample from the nearest K neighbors of the same class as x_i is chosen as an auxiliary example.

This step is known as Auxiliary Sample Selection. After that, linear interpolation between x_i and x_j is performed n times, using Equation (2.1) to produce new synthetic samples. These processes result in the production of n synthetic samples.

$$x_{\text{new,attr}} = x_{i,\text{attr}} + \text{rand}(0,1) \times (x_{ij} - x_{i,\text{attr}}) \tag{2.1}$$

In this equation, $x_{i,\text{attr}}$ refers to the value of the attribute attr for the i-th sample in the minority class. The term $\text{rand}(0,1)$ represents a randomly generated number between 0 and 1. $x_{ij,\text{attr}}$ denotes the j-th nearest neighbor of sample x_i. A new sample, x_{new}, is created by interpolating between x_i and x_{ij}. As shown in Equation (2.1), the new sample x_{new} is obtained by blending the values of x_i and x_{ij} [13, 22].

SMOTE involves generating synthetic instances for the minority class by interpolating existing minority class instances.

- **Random oversampling:** Random oversampling is a technique used to moderate class imbalance by randomly duplicating instances from the minority class. This increases the sample size and creates artificial diversity, which may result negatively as it can hinder the model's ability to differentiate between real-world data and artificial data. The duplication process is repeated until class balance is achieved. Adding random oversampling to small datasets may not require additional data generation. However, by creating the same copies of the data from the minority class samples, the model may be delicate towards new and unseen data [61].

- **SMOTEENN, SMOTE with (ENN):** ENN is an under sampling method that removes noisy and irrelevant samples from the dataset. This technique identifies the majority class very near to the minority class samples. Class imbalance can obstruct the performance of learning systems, even when the minority class is underrepresented. SMOTE-ENN combines the techniques to result in a more balanced and informative dataset. SMOTE is used first to oversample the minority class, then ENN is applied to remove noisy majority class samples. This combined method allows for an improved performance of machine learning models on imbalance datasets [62].

- **Borderline-SMOTE (oversample technique using Borderline-SMOTE):** Borderline SMOTE, an advanced version of SMOTE and ADASYN, focus on creating synthetic samples for the minority class in close proximity to the decision boundary between classes. These synthetic samples allow the model to learn the decision boundary more precisely resulting in an enhanced classification performance. This approach improves classification accuracy. By employing B SMOTE in the training phase, researchers were able to observe the exact boundary between each of the classes which enhanced predictions.

$$P = \{P_1, P_2, \ldots, P_{\text{pnum}}\}, \quad N = \{n_1, n_2, \ldots, n_{\text{nnum}}\} \tag{2.2}$$

In the Equation, the entire training set is assumed as T i.e., both P and N. The entire minority class is supposed as P and the majority class is assumed as N. Moreover, P_{pnum} represents the total number of minority instances, and n_{nnum} represents the total number of majority cases. Moreover, for each P_i where $i = 1, 2, 3, \ldots, P_{\text{pnum}}$ in the minority class P, its m nearest neighbors are calculated from the entire training set T.

Overall, B-SMOTE is effective in targeting the decision boundary, which can then enhance the model's ability in differentiating between classes [24].

- **SVM SMOTE: combines the support vector machine (SVM) with SMOTE:** The purpose of SVM SMOTE is to determine the decision boundary between the majority and minority class samples. It uses the support vectors from the SVM model rather than K neighbours to identify the nearest neighbours. However, SVM SMOTE technique focuses on the decision boundary by generating synthetic samples to improve classification accuracy. A positive feature of this technique includes its ability to handle complex non-linear decision boundaries. Moreover, imbalanced datasets are more common in real world datasets.This method can also assist to mitigate overfitting. Overall, SVM SMOTE is efficient in handling class imbalance and in improving the performance of machine learning models on imbalanced datasets [63].

- **SMOTE-Tomek Links:** Tomek Links is an undersampling technique used by removing majority class samples near the minority class sample. Removing these samples can clarify the decision boundary between classes and clear the distinction between the majority and minority classes. The samples removed are often noisy or misclassified, Overall, Tomek Links denotes a technique used to detect pairs of closest

neighbors within a dataset that exhibit dissimilar classes [64]. Eliminating either one or both instances from these pairs, particularly those from the majority class, results in a reduction in noise or ambiguity within the decision boundary of the training dataset.

- **SMOTE-NC (synthetic minority over-sampling technique for nominal and continuous):** is used to oversample data with categorical features. SMOTE-NC is designed to effectively generate new synthetic datasets that represent the underlying data distribution. This leads to an enhanced model performance on imbalance datasets. This technique is used to address class imbalance in datasets including numerical and categorical features. It works by first identifying the minority class, then randomly assigns a minority class instance, identifies the nearest K neighbours of the selected minority class instance, and finally, generates a new synthetic sample for each of the nearest K neighbours. Its numerical feature can linearly interpolate between the values of the current minority class and its neighbour. the categorical feature assigns the closet categorical value from the nearest neighbour. However, overusing SMOTE-NC can lead to overfitting, allowing the model to be too specialized towards the synthetic data. If the minority class is noisy, the new synthetic samples may intensify the noise hindering results. A leading advantage of SMOTE-NC is its ability to effectively address class imbalance and provide representative data to the model which contributes to a better performance [65].

2.2.2 GANs for Data Balancing

Generative Adversarial Networks, otherwise known as GANs, are a class of AI algorithms that are designed to produce data similar to real-world examples. GANs utilize an innovative framework involving two neural networks: a generator and a discriminator. These networks engage one another. The purpose of the generator is to generate synthetic data that is very alike to real-world data; and the purpose of the discriminator is to distinguish between the synthetic data and the real-world data. Through the iterative process, the generator learns to refine its output, creating new highly realistic data instances.

2.2.2.1 What are Generative Adversarial Networks

Generative Adversarial Networks, or GANs, comprise a class of machine learning methods designed for unsupervised learning tasks. It is an innovative AI tool that can create realistic data without supervision. Additionally, the GAN model requires two neural networks: a

generator and a discriminator. The generator produces a batch of samples, which, along with real examples from the domain, are fed to the discriminator. The discriminator then classifies them as either real or fake. Subsequently, the discriminator undergoes updates to improve its ability to distinguish between real and fake samples in the subsequent round. Additionally, the generator receives updates based on its success or failure in deceiving the discriminator with its generated samples. The two models then engage in an adversarial process, reminiscent of a zero-sum game. The discriminator works to distinguish real and synthetic data, accurately, and receives a reward for accurate classifications and a penalty for incorrect classifications, this engagement between the two models allows for an iterative learning process, leading to the production of highly realistic generated data [66].

As previously mentioned, GAN is an unsupervised learning method where both its generator and discriminator are trained at the same time. This process is continuous as the generator strives to create synthetic data and the discriminator works to classify the generator's data: synthetic and authentic data [67].

2.2.2.2 History of GANS

In 2014, Ian Goodfellow and his colleagues introduced GANs. Such a model changed the production of hyper realistic data,images, videos, and texts. This transformed the field of AI and DL. The foundational paper titled "Generative Adversarial Nets" was presented at the Neural Information Processing Systems (NeurIPS) conference, displaying the novel concept of training two neural networks concurrently in a game-theoretic scenario. GANs were developed to overcome the limitations of early generative models, like (VAEs) and Restricted Boltzmann Machines (RBMs) [26]. The abilities of these earlier models were limited in creating high quality data. GAN is based on a competitive training process that strives to achieve results when the Generator produces synthetic data statistically identical to real-world data in which the Discriminator is unable to distinguish between the two types of data. Since their introduction, GANs have been expanded to many applications. GANs can be used to produce synthetic patient data, create artwork, produce new gaming environments and much more. However, considering the enhancements of GANs, their development has raised ethical questions concerning its potential in producing harmful and misleading data. These concerns are continually addressed by experts in the field of AI. These experts have also ensured the ethical use of GAN technology [68].

2.2.2.3 How GANS Work: An Overview

Generative Adversarial Networks (GANS) have outstandingly transformed models in AI primarily through the adaptation of adversarial training. This method requires two neural networks that are trained at the same time: a generator and a discriminator.

- **Generator (G):** The Generator is a neural network that takes random noise as input and work to produce synthetic data that highly resembles real-world data. At first, the generator's output can be low in quality, however, through iterative learning, it learns to produce indistinguishable samples from synthetic data.
- **Discriminator (D):** The Discriminator is a neural network that functions as the adversary in the GAN model. It receives data, generated and real, from the Generator and must accurately classify the two types of data. Eventually, the discriminator learns from its errors when misclassifying and gradually enhances its ability to classify between real and generated data.

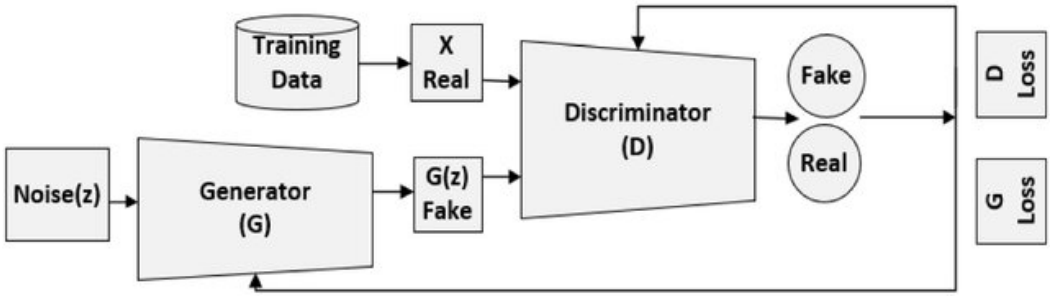


Fig. 2.4 Basic architecture of Generative Adversarial Networks (GAN)[3].

As shown in Figure 2.4, the generator works to create data similar to a real dataset and the discriminator must distinguish between the two datasets. The generator uses random noise as input and generates synthetic data samples. Its counterpart, the discriminator network (D), receives either the generated data or an authentic sample and outputs a probability, signifying how likely it is that the given sample is real. Both models must work to achieve precision, to the point where the generator creates data that cannot be differentiated by the discriminator [26].

Training a GAN requires an alternate process switching from improving the discriminator and the generator. At first, the discriminator is trained to classify real data from generated data and this is achieved by probability of assigning the correct label to both the real and

generated data. At the same time, the generator is optimized to produce data the discriminator cannot recognize [3]. The formal representation of a GAN can be seen through its objective function, often described as a min-max game:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))] \tag{2.3}$$

Here, x represents a real data instance, and z represents the latent space vectors used by G to generate data. The first term $\mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)]$ calculates the expectation that D assigns the correct label to real data, while the second term $\mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]$ computes the expectation that D assigns the correct label to generated data.

2.2.2.4 GANS vs Other Generative Models

GANs are prominent in the field of machine learning due to their efficiency in producing high quality data. However, GANs are not the only class capable of data generation; there are other significant models that are able to generate data. Models such as: (VAEs), Restricted Boltzmann Machines (RBMSs), and Autoregressive models like PixelRNN and PixelCNN [26, 69].

- **Variational Autoencoders (VAEs):** VAEs encode and decode units into latent space. They can also generate new datasets similar to the original data, however they are considered to be not as effective when creating detailed images, in comparison to GANs.
- **Restricted Boltzmann Machines (RBMs):** RBMs, a type of generative model, that uses a hidden layer of units to model to learn the distribution of visible units, Compared to GANs and VAEs, RBMs are not as proficient in generating high quality and differentiated outputs.
- **Autoregressive Models (PixelRNN and PixelCNN):** These models create images pixel-by-pixel. PixelRNN uses recurrent neural networks and PixelCNN uses convolutional neural networks. However, when producing large images, they are computationally challenging.

Additionally, GANs are different from these models because of the adversarial training process.
This rigorous process promotes the generator to create hyper realistic data. Its advantages include:

- **High-Quality Generation:** GANs are effective in producing high quality and detailed data.
- **Flexibility:** GANs are versatile models capable of handling various types of data, images, and video synthesis to text-to-text generation.
- **Improved Sample Diversity:** GANs are able to create numerous ranges of outputs.

However, GANs can face challenges in training instability and mode collapse. These challenges happen when the generator creates limited numbers of data. Methods like Wasserstein loss, batch normalization, and architectural adjustments are used to address these shortcomings, pushing GANs to outperform other generative models.

Like any machine learning model, GANs have their advantages and disadvantages. Each of these generating models are different from the other in their operation and results. Therefore, deciding which model to work with depends on considering factors like quality of output, and complexity of the data.

2.2.2.5 Types of Generative Adversarial Networks

Since they were first introduced, GANs have undergone immense changes and advancements, including:

- **Conditional GANs (cGANs):** Are a type of GANs that are able to generate specific data according to the given characteristics. This limits GAN to creating data only within the specific boundaries [25].
- **Deep Convolutional GANs (DCGANS):**This specific type of GANs uses convolutional layers in the generator and discriminator. This allows it to be effective in producing high quality images [3].
- **Wasserstein GANs (WGANS):** This kind of GANs works to enhance the stability of training and the quality of the synthetic data [70].
- **Cycle GANs:** are a type of GANs that allow for unpaired image-to-image translation, meaning a model can be trained to convert images from one domain to another without the need of other images [71].

DCGANS (Deep Convolutional GANs), Conditional GANS (CGANs), CycleGANS, Pix2Pix GANs, SDG GAN, StyleGANS, Wasserstein GANS (WGANS), BigGANs, GANs for Text-to-Image Synthesis, 3D GANS, Super-Resolution GANS (SRGANs), Adversarial

Autoencoders, NSGAN, LSGAN, and CTGAN are GAN variants that introduce innovations either in the architectural design, the training algorithm, or the objective function, tackling specific challenges inherent to the original GAN formulation [69].

Understanding these variants is essential for leveraging the full potential of generative adversarial networks in a wide array of applications. Below, we will discuss most of these GAN types:

1. **CGANs (Conditional GANS):** Conditional Generative Adversarial Networks, CGANs, include a conditional component to the generative process. This component enables the generative model to produce a dataset based on the given input characteristics transforming the standard of the GAN to a conditional one, hence the name [25]. The conditional variable can include: class labels, text description, and even images.

   Additionally, CGAN shares a similarity to the general model of GAN, as it has a generator and a discriminator. However, both receive an additional input, which comprises the condition.

   The applications of CGANs are abundant, including but not limited to:

   - Image-to-image translation: where the condition y can be an image from another domain, guiding the generator to produce a corresponding image in the target domain.
   - Photo inpainting: where the condition y can be a partially masked image, and the generator learns to fill in the missing parts.
   - Text-to-image synthesis: where y is a textual description that guides the generation of image content.

   Introducing conditionality into GANs has significantly widened their applicability and performance in generating conditioned data. By effectively using auxiliary information as conditions, CGANs can be precise and produce controlled data in the generator [25].

2. **CTGAN (Conditional Tabular Generative Adversarial Training) :** CTGAN is a DL model that is designed to create synthetic tabular data that mimics real world

data. They work by using a GAN architecture, and similarly, it has two components: a Generator G and a Discriminator D. CTGAN is able to effectively generate fabricated data very similar to real world data and can be used to protect sensitive data. A study titled, "Modeline Tabular Data Using Conditional GAN" [72], researched a flexible and strong model to understand the distribution of columns using complicated distributions. The authors of this study suggest their conditional generator model and training-by-sampling can overcome the imbalance training data.

3. **CTGAN (DCGANS (Deep Convolutional GANs):** Deep Convolutional Generative Adversarial Networks, DCGANs, were first introduced by Radford, Metz, and Chintala in 2015 [68]. By incorporating convolutional neural networks, CNNs, into the generator and discriminator. DCGANs utilize the power of CNNs in image feature extraction. Additionally, this model addresses stability and image quality challenges found in earlier GANs, enforcing DCGANs to be highly effective for image generation tasks.

   The architecture of DCGANS is characterized by several key modifications to the conventional GAN framework:

   - Replacing fully connected layers with convolutional layers in the generator and discriminator allows the network to effectively capture spatial differences in images,
   - Using batch normalization in the generator and discriminator stabilizes the training by normalizing the input to every unit. A mean of zero and a variance of one is required.
   - Instead of pooling layers, the model utilizes strided convolutions to minimize image size in the discriminator and transposed convolution for upsampling in the generator.
   - Using ReLU activation in the generator and the LeakyReLU activation in the discriminator.

4. **SDG GAN:** The generator and discriminator of the SDG GAN [28] are both convolutional channels with an MLP design in the SDG GAN framework. A standard GAN's generator seeks to produce fake data that closely resemble the true distribution

[73]. Synthetic Data Generation GAN has the capacity to outperform density-based oversampling methods and enhances the classification ability of benchmark datasets and real fraud datasets.

In addition, original GAN uses regular loss, however, instead of using regular loss, the SDG GAN utilizes feature matching loss to improve GAN training. The feature matching objective is central to the SDG GAN approach, which uses a conditional GAN method to calculate the conditional distribution of the data based on class labels. This allows the generator to sample from the minority class by conditioning on the minority class label. To reduce the statistical differences between the actual data and the generated data, the generator's objective function is adjusted to focus on matching the features of real data rather than just fooling the discriminator. Meanwhile, the discriminator is trained in the same way as in a standard GAN setup. The original SDG GAN demonstrates that feature matching is an effective strategy for stabilizing the training process, particularly when regular GANs may face instability [28, 73].

5. **WGANs (Wasserstein GANs):** WGANs, a variant of GANs, address some of the stability and mode collapse issues present in traditional GANs. This is addressed by using Wasserstein distance as the loss function which provides a meaningful and stable measure for differentiating between real and generated data. The Wasserstein distance provides a meaningful and continuous measure of the difference between real and synthetic data. To encourage effective training, this architecture alters the loss role of the default application and uses a weight clip [27, 74].

6. **NS GAN (Non-saturating GANs:** Non-saturating GANs [75] is an improved version of GANs [26]. The form of GAN that is most widely used as a benchmark in research and practical applications is non-saturating GAN (NS GAN). However, the NS GAN algorithm lacked theoretical justifications [28], like other GANs such as WGAN. The loss function performs poorly in practice; despite being outstanding for theoretical results. The GAN has difficulty converging, stabilising its training, and offering a range of samples.
   The aforementioned loss function for generator should not be trained, but rather improved gradients from prior training should be utilized [28]. In the adversarial game, the discriminator tries to classify which data is fake or real. On the other hand, the objective of generator is to fool discriminator into thinking that the fake data it produces is

real data. According to [26] maximizing the objective of discriminator after swapping data (NS GAN) gives better results than minimizing the objective of discriminator (Saturating GAN) directly.

7. **LS GAN Least (Squares GANs):** The foremost benefit of LS GANs [76] is that unlike conventional GANs, where there is nearly no loss for samples that lie on the correct side of the decision boundary, LS GANs can penalize samples although they are rightly classified. The other benefit is that the decision boundary can produce more and more gradients when updating the G, this then lessens the issue of the vanishing gradient [77]. This design shows that the model trains more steadily and is better able to handle the gradient vanishing problem than the vanilla method by using the least square error as the loss. The aim of generator is to learn the distribution over data. The generator begins from sampling input variables from a Gaussian distribution then maps the input variables to data space via a differentiable network. In addition, discriminator is a classifier that tries to recognize the instance from training data or from Generator.

8. **CycleGANS (Cycle-Consistent Adversarial Networks):** CycleGANS enables image-to-image translation without required paired examples. This can convert images from one domain to another like enhancing photos, transferring styles, and more without the need of corresponding images from both domains. The model of CycleGANS consists of two generators and two discriminators. The generators are responsible for translating images from domain A to domain B and vice versa, while the discriminators aim at distinguishing between translated images and real images in each domain.

   This creates two mapping functions:

$$G : A \rightarrow B \quad \text{and} \quad F : B \rightarrow A$$

   Cycle GANs include a cycle consistency loss to ensure that images are translated from domain A to domain B can be translated back to domain A with minimal loss of information. However, the loss measures how well the network preserves the essential information during the translation and reconstruction process.

   The Cycle GAN's function is to combine adversarial losses for both forward and backward translation with a cycle consistency loss. Similarly, the adversarial loss for

the transformer F and its discriminator D is defined in a parallel manner [71].

9. **Pix2Pix GANS:** Pix2Pix GANS are a type of GANs that specialize in image-to-image translation tasks. Through using paired sets, these types of GANs can learn to map input images to output images. They can be efficient in performing tasks like converting sketches to photographs, changing day scenes to night, or colorizing black and white images [78].

   The Pix2Pix GAN consists of a generator and a discriminator. The generator takes a source image and generates a target image. The discriminator determines whether the generated picture is real or fake. The aim of the generator is to train the generator to produce synthetic but realistic images that cannot be determined by the discriminator. Unlike the traditional discriminator that classifies the entire image, the PatchGAN classifies image patches, this allows for the model to focus on high frequency information which will lead to high quality generated images.

10. **StyleGANs (Style-based Generative Adversarial Networks):** Style-based Generative Adversarial Networks, StyleGANs, [120]. This class of GANs have significantly advanced in image synthesis. They enable separate control over high level attributes, medium-level features, and fine details in image generation [79]. StyleGAN architecture has two main components: the mapping network and the synthesis network.The mapping network transforms a random input vector into an intermediate latent space . The synthesis network then generates the output image, using the space control style at each generation level. This allows for precise manipulation of the image attributes from shape to color and texture.

    One of the StyleGAN's important innovations is Adaptive Instance Normalization, AdaIN. AdaIN adjusts the mean and variance of the features in each synthesis network layer based on style parameters from the space which allows style control at different levels of detail when the image is synthesized. StyleGANs are not only used for image production. Their manipulation style allows them to be implemented in fields like facial attribute manipulation, fashion modification, and photorealistic landscapes. The enhanced models of StyleGAN2 and StyleGAN3 have addressed challenges in image artifacts and further refined the quality and control of the generated images.

11. **BigGANS (Large Scale GANS):** BigGANs, a recent advancement in GANs, have achieved prominent success in generating high quality and detailed images at a variety of resolutions. BigGANs can introduce training optimization by utilizing DL and massive datasets to improve traditional GANs. BigGANS follow the general structure of GANS which includes a generator and discriminator. BiGANs highlight depth and scale when using networks more developed than earlier models [80].

    The important features of BigGANS include:

    - Larger Models: Larger models enhance BiGANs ability to learn and produce complex patterns.
    - Large Batch Sizes: Large batches can improve the stability and quality by providing accurate gradient estimates and better distribution characteristics.
    - Hinge Loss Function: BigGANs uses hinge loss, which provides more stable gradient flow and faster convergence during large scale training.
    - Class Conditional Generation: By incorporating class labels into the generator, BigGANs can produce specific image classes which enables targeted image synthesis.
    - Batch Normalization: Batch Normalization in both the generator and discriminator stabilizes training especially for large scale networks and datasets.

12. **Progressive GANs (ProGANs):** The advent of ProGANs is one example of an improvement that has been made to GAN networks [81]. During training, the generator and discriminator in a progressive GAN (ProGAN) gradually get better and larger, according to the theory behind the network [68].

    Initially, ProGANs are trained with a low-quality image that fails to capture important details. As the training progresses, the resolution is increased. The resolution is enhanced because the ProGAN is able to collect finer information with each iteration's additional layers. One drawback of these methods is the time it takes to train, because there are more rounds involved. The table 2.5 below clearly shows the strengths and weaknesses of each model

Table 2.5 A Comparison of Various GAN Models.

GAN Model	Strength	Weakness
GAN	High-quality image generation; supports supervised, unsupervised, and semi-supervised learning; continual learning.	Privacy concerns; highly dependent on hyperparameters; mode collapse.
CGAN	Conditional generation; high-quality outputs; versatile; data augmentation.	Training instability; requires large datasets; mode collapse.
CTGAN	Handles mixed data types; good for imbalanced data; supports conditional generation.	Computationally intensive; struggles with high-dimensional or temporal data.
DCGAN	Hierarchical feature learning; less prone to mode collapse; good with images.	Poor performance with text; hard to explain generation process.
SDG GAN	Enhances data diversity; good for imbalance; stabilizes training.	High computational cost; requires large datasets; sensitive to config.
WGAN	Stable training; smooth optimization; high-quality outputs.	Expensive to train; slower convergence; tuning critic is hard.
NSGAN	Sharp outputs; simplified loss.	Mode collapse; training instability.
LSGAN	Stabilizes training; reduces vanishing gradients.	Still may collapse; computationally intensive.
CycleGAN	Unpaired image translation; no need for paired data; effective for style transfer.	Long training; prone to collapse with geometric transformations.
Pix2Pix GAN	Effective paired translation; detailed image output.	Needs paired data; sensitive to dataset size and quality.
StyleGAN	High-quality and diverse outputs; captures various styles.	Memory intensive; impractical for limited-resource devices.
BigGAN	High-resolution images; scalable.	Long training time; high resource needs.
ProGAN	Realistic images with fine details; stable via progressive training.	Expensive and long training; sensitive to hyperparameters.

2.2.2.6 Future Directions in GAN Research

GANs, over the years, have advanced immensely. Their functions and applications can be studied for enhancements A challenge in GAN is instability in training that leads to mode collapse. To prevent mode collapse. researchers have explored adjusting loss functions and adding regularization terms. Since GANs are very effective in generating high quality images, they face challenges in scaling to higher resolutions. Enhanced training techniques could be implemented to increase their capabilities with higher resolutions. Developing their controllability and interpretability is crucial to enhance the applications of GANs. Techniques have to be created to manipulate the latent space, condition the input in generating and understand the factors that produce the outputs.

Additionally, more work can be done to optimize efficiency. Current GAN models that produce high-quality data are computationally intensive, requiring intricate work. By implementing ethical policies, the risks associated with wrongful use and misinformation can be regulated [26].

The ongoing advancements of GAN show their potential in advancing generative modeling and also in exploring new approaches with a positive societal and technological impact. GANs can further develop and continue to make substantial contributions to the AI field.

2.2.3 GAN for data balancing in IDS

Recent developments in Intrusion Detection Systems (IDSs) have spurred fresh enthusiasm in tackling the ubiquitous problem of class imbalance. Attack samples are far less in normal datasets as well as in real-world environments. The efficient identification of cyber threats suffers from this imbalance since IDSs are biassed towards the majority class (benign samples), which usually results in poor performance in spotting unusual but important attack events. As a result, this problem has been the main focus of research and drives initiatives to create methods able to manage the difference between attack and benign samples [82].

The concept of employing GAN models to address the class imbalance problem is introduced by Lee and Park in reference [83]. In general, GAN is an unsupervised learning technique rooted in DL and generates synthetic data that closely resembles the existing data. The authors in this work used GAN to effectively tackle fitting issues, class overlaps, and noise through the process of resampling by explicitly defining the desired rare class. To evaluate the classifier's performance, the re-sampled data are trained using the widely

adopted machine learning technique called RF. The proposed solution demonstrates superior performance compared to the methods currently utilized [82].

Huang and colleagues [84] developed a solution to address the problem of class imbalance in Intrusion Detection Systems (IDS) by introducing Imbalanced Generative Adversarial Networks (IGANs). Their method involves modifying the GAN generator with a filter to focus on generating samples for the minority class, which is often underrepresented in attack data. They incorporated this model into an IDS framework, which operates in three stages: feature extraction, applying IGAN, and using a (DNN) for classification. Testing their model on three different datasets, including the CIC-IDS-2017 dataset, they achieved impressive results, with over 99% accuracy in both F1-score and AUC. However, the study didn't provide much detail on the specific GAN architecture used and didn't test the model's ability to withstand adversarial attacks, which leaves room for further research.

Seo, Song, and Kim introduced a GAN-based intrusion detection system (GIDS). Due to the limited number of known attack signatures for vehicle networks, the author employs the concept of generating unknown attacks during the training process to enable the IDS to effectively handle various types of attacks. In the context of vehicle IDS, accuracy is of the utmost importance to ensure driver safety, as any false positive error could have serious consequences. Traditional IDS approaches are inadequate for dealing with numerous new and undiscovered attacks that may arise. The proposed GAN-based IDS solution successfully detects four previously unknown attacks [85].

Another IDS solution based on GAN was proposed in [86] introduced an advanced approach to address the class imbalance issue in Intrusion Detection Systems (IDS). Their system follows four key steps: preprocessing, GAN training, autoencoder training, and classification. In the GAN training phase, they used a stable version of GAN called BEGAN to generate samples for the minority class. These synthetic samples were then processed using an autoencoder, which aids in detecting abnormal data patterns. The researchers trained various DL models, such as DNNs, CNNs, and LSTMs, on both the original and synthetic data to improve detection accuracy. Their method showed significant improvements, especially in detecting rare attack types like R2L and probe attacks in the NSL-KDD dataset. However, the study didn't evaluate the system's performance in real-world distributed network settings or its resilience to adversarial attacks, leaving these areas open for future research.

Proposing a Modified Conditional Generative Adversarial Network (MCGAN), Babu and Rao [87] tackled the problem of class imbalance in (NIDS). One of the few studies to precisely describe the GAN design, this model especially creates synthetic samples for underrepresented classes using a customized Conditional GAN (CGAN) architecture. The method uses a linear-correlation-based feature selecting technique and combined optimizer for feature extraction. Using a Bidirectional Long Short-Term Memory (Bi-LSTM) approach, a classification accuracy of noteworthy 95.6% on the NSL-KDD dataset is obtained.

The study referenced in [88] they used SYN-GAN, a GAN-based system meant to create totally synthetic datasets instead of only augmenting already existing ones, the authors addressed the difficulties of data shortage and imbalance in IoT security. Their method removes the need for labeled data by addressing the absence of benchmark datasets and the high expenses related with data collecting in IoT contexts. Testing SYN-GAN on synthetic datasets taken from UNSW-NB15, NSL-KDD, BoT-IoT, it was successful in training an IDS. The results revealed that models trained on synthetic datasets performed comparable to those trained on genuine datasets, therefore indicating the possibilities of this approach for handling IoT security issues.

The fast development of cloud computing has made security a major area of research especially in relation to class imbalance problems in cloud-based intrusion detection systems (IDSs). Vu et al. [89] approached this problem by creating a model combining two specialized GANs for data balancing with a (DNN) for detection. Leveraging a Conditional Denoising Adversarial Autoencoder (CDAAE), the first GAN creates focused minority class samples. To generate samples close to decision boundaries and hence improve classification accuracy, the second GAN, CDAAE-KNN combines CDAAE with a (KNN) algorithm. On the expanded datasets the researchers used Support Vector machines (SVMs), Decision Trees (DTs), and Random Forests (RFs) to assess their model. On more than one IDS datasets benchmark, experimental findings showed how well the model reduced class imbalance and raised detection performance.

By use of a single GAN, Chkirbene et al. [90] addressed the class imbalance in cloud network intrusion detection. Crucially for balancing the dataset, they created a machine learning-based secure network model that automatically sets GAN parameters including the number of inner learning steps for the discriminator. Particularly for rare classes, their tests on the UNSW and NSL-KDD datasets demonstrated considerable increases in classification accuracy. The results showed that their model detected assaults in cloud environments

better than current techniques, therefore underlining the efficiency of GANs in tackling the difficulties of class imbalance in IDS.

Most of the research mentioned above focuses on using GANs for signature-based IDS, which poses a significant problem as it requires large amounts of data that are not always readily available or up to date [82]. As a result, many researchers believe that unsupervised or weakly supervised approaches are more effective and realistic, making them better suited to handle the rapidly evolving nature of network attacks. Wang et al. [91] introduced a novel methodology for anomaly-based intrusion detection systems (IDS) through the integration of Generative Adversarial Networks (GANs) with vision transformers. The GAN produces synthetic samples to enhance the dataset, which is subsequently subjected to min-max normalization. The enriched data is input into a transformer model utilized for anomaly prediction in network traffic. Their assessment of the CIC-IDS-2017 dataset illustrates the efficacy of this method in achieving data equilibrium and ensuring precise forecasts. This approach provides a viable alternative for enhancing the precision and resilience of Intrusion Detection Systems in identifying network intrusions.

Geiger et al. [92] introduced TadGAN, a framework employing GANs for anomaly identification in time-series data. This model employs an LSTM network to capture temporal dependencies in the data and is trained with a cycle consistency loss to assure precise reconstruction of time-series data. The researchers presented novel algorithms for calculating reconstruction errors, essential for assessing the anomaly score. Their comprehensive experiments, utilizing datasets from NASA, Yahoo, Numenta, Amazon, and X, evidenced that TadGAN surpasses current methodologies regarding F1-score, while also markedly diminishing the false positive rate a prevalent issue for IDSs managing time-series data. This open-source approach possesses significant promise for enhancing anomaly detection studies in time-series data. The works discussed in this section are part of the comparative summaries presented in table 2.5 and 2.6.

The shortage of suitable and up-to-date datasets is a significant challenge in the IoT domain, prompting the adoption of anomaly-based IDSs for IoT environments [93].Ullah and Mahmoud [94] created an anomaly-based IDS architecture employing Conditional Generative Adversarial Networks (CGANs) to mitigate class imbalance in Internet of Things (IoT) networks.Their methodology presented three varieties of CGAN: one-class CGAN (ocCGAN) for generating samples from a single class, binary-class CGAN (bcCGAN) for two classes, and multi-class CGAN for multiple classes. These models utilize distance-based

computations to accurately assign class labels to the generated data. Assessments of seven benchmark IoT datasets revealed an exceptional average accuracy of 98.01% for intrusion detection, highlighting the framework's effectiveness in balancing datasets and enhancing detection rates for underrepresented attack categories.

Research on usage of GANs in hybrid Intrusion Detection Systems (IDSs) emphasizes the benefits of combining anomaly-based and signature-based methods [95]. This hybrid approach leverages signature-based precision in detecting known attacks and anomaly-based adaptability to identify new threats, enhancing overall detection accuracy and coverage. Mari et al. [96] Studying GANs in hybrid IDSs, they found that they can generate adversarial attack samples to make IDSs more resilient and that they can also balance datasets. They used ANNs, KNNs, and RFs for classification on the NSL-KDD dataset. The application of GANs allowed for the generation of synthetic assault samples that were highly realistic, leading to an improvement in detection accuracy. They found that specific kinds of attacks, such denial-of-service attacks, were more prevalent in the dataset, and their method was able to detect these attacks with 90% accuracy. This study demonstrates that GANs are very useful for training and stress testing IDSs.

Aldhaheri and Alhuzali [97] introduced the SGAN-IDS architecture to combat evolving and real-time assault scenarios and improve intrusion detection systems (IDSs). Combining self-attention methods with generative adversarial networks (GANs) is the essence of this design. Enhanced intrusion detection systems are the end aim of this design. This model can bypass the currently active intrusion detection systems by using self-attention to create synthetic assault samples of sufficient quality. Applying this method to the CIC-IDS-2017 dataset provided actual evidence of its usefulness. The experiment demonstrated a 15.93% decrease in the detection rates of hostile assaults compared to the previous results. This shows that the approach has the ability to evaluate and enhance the robustness of intrusion detection systems (IDS) against more complex threats.

Strickland et al. [98] proposed a hybrid methodology for network intrusion detection that integrates Generative Adversarial Networks (GANs) with Deep Reinforcement Learning (DRL). They employed GANs to produce synthetic attack samples, which were subsequently utilized to train the DRL model for both binary and multiclass attack categorization. The GAN was trained on the NSL-KDD dataset, yielding findings that demonstrated a substantial enhancement in the detection of minority classes, hence improving the F1-score and detection rate. Their solution, by tackling the class imbalance issue, shown enhanced accuracy in

detecting infrequent attack types, hence highlighting the efficacy of synthetic data in IDS applications.

The table below 2.6 provides an overview of related work, highlighting most recent GAN models used, the datasets employed for training, the accuracy metrics evaluated, and the domain of each study. This comparison allows for a better understanding of the advancements in the field and sets the foundation for the proposed method in this thesis.

Generative Adversarial Networks (GANs) have garnered considerable attention in recent years owing to their proficiency in learning data characteristics via a competitive interaction between the generator and the discriminator. This interactive game-like framework allows GANs to replicate intricate data distributions, rendering them very advantageous for data production tasks. Generative Adversarial Networks (GANs) have demonstrated significant progress in multiple fields, including picture generation [26], speech synthesis [108], and text generation [68]. Consequently, researchers from many fields are progressively using GAN-based methodologies due to their capacity to produce high-quality synthetic data and enhance model efficacy in data-deficient settings [109]. The capacity to produce realistic and varied data renders GANs an optimal selection for our model, particularly in the realm of network intrusion detection, where the generation of synthetic network traffic helps mitigate data imbalance and enhance detection precision.

Traditional GAN-based methods have been widely used for generating synthetic samples in various domains, including intrusion detection. However, most existing approaches are designed for general-purpose data augmentation rather than being specifically optimized for cybersecurity applications. TDCGAN is a tailored DL framework that directly addresses the challenges posed by imbalanced IDS datasets.Key Advantages Over Existing GAN Approaches:

- **Targeted Minority Class Augmentation:** Many existing GANs generate synthetic data for all classes, which does not effectively address class imbalance. TDCGAN focuses exclusively on minority-class augmentation, generating high-quality attack samples while preserving the natural distribution of benign traffic.
- **Higher Detection Performance:** Compared to other synthetic data generation techniques (e.g., SMOTE, CTGAN, and CGAN), TDCGAN demonstrates higher accuracy, precision, recall, and F1-score in intrusion detection tasks. Experimental results show that classifiers trained with TDCGAN-augmented datasets achieve significantly better detection rates for rare attacks.

Table 2.6 Summary of GAN-Based IDS Literature.

Related Work	GAN Model	Dataset	Accuracy Metrics	Year	Domain
[99]	Standard (GAN)	KDDCup99	Acc.: 97.76%, Prec.: 97.85%, Rec.: 98.12%, F1.: 97.98%	2020	Class imbalance in anomaly-based IDS
[84]	Imbalanced GAN (IGAN)	NSL-KDD, CIC-IDS-2017, UNSW-NB15	Acc.: 84.45%, Prec.: 84.50%, Rec.: 96.13%, F1.: 89.92%	2022	Class imbalance
[100]	Wasserstein GAN (WGAN)	NSL-KDD, KDDCup99	Acc.: 97.93%, Prec.: 97.87%, Rec.: 98.50%, F1.: 98.18%	2023	Class imbalance in anomaly-based IDS
[101]	Standard (GAN)	NSL-KDD, KDDCup99	Acc.: < 98%	2023	Class imbalance in anomaly-based IDS
[87]	Modified Conditional GAN (MC-GAN)	NSL-KDD	Acc.: 95.16%, Prec.: 94.21%, F1.: 96.7%	2023	Class imbalance in anomaly-based IDS
[88]	Synthetic GAN (SYN-GAN)	Kitsune, NSL-KDD	Acc.: 95.16%, Prec.: 94.21%, F1.: 96.7%	2024	Class imbalance in anomaly-based IDS
[102]	Dynamic Distributed GAN (DDGAN)	CICDDoS2019, ToN-IoT	Not provided	2024	Class imbalance in anomaly-based IDS
[103]	Standard (GAN)	ToN-IoT, NSL-KDD	Acc.: 94.45%, F1.: 93.6%	2022	Class imbalance in anomaly-based IDS, IoT distributed
[104]	Standard GAN + encoder (N-GAN)	CIC-IDS2017	Det.: 81.3%, AUC.: 82%	2022	Class imbalance in anomaly-based IDS
[105]	Generative Local Metric Learning (GLML)	NSL-KDD, CICIDS 2017	Acc.: 90.19%, Det.: 81%	2020	Class imbalance in hybrid IDS
[106]	Conditional tabular GAN (CTGAN)	NSL-KDD, CICIDS 2017	Rec.: 86%, F1.: 76%	2023	Class imbalance in hybrid IDS
[107]	Conditional tabular GAN (CTGAN)	NSL-KDD, CICIDS 2017	Rec.: 86.1%	2022	Hybrid IDS, military network protection
[94]	Conditional GAN (MC-GAN)	Bot-IoT, NSL-KDD, KDDCup99	Prec.: 94.92%, Rec.: 97.36%, Det.: 97%	2021	Class imbalance in anomaly-based IDS

Note: Acc. = Accuracy, Prec. = Precision, Rec. = Recall, F1. = F1-Score, Det. = Detection Rate, AUC. = ROC AUC

Chapter 3

Sate of the art

In this Chapter, a set of recent related works that apply machine learning methods to the IDS are summarized and analysed. The literature review papers are collected for the years from 2018 to 2025 with a more than 35 research papers. Table **??** identifies each work by answering the following questions:

1. What are the machine learning methods used by each work to improve IDS?
2. Which datasets are used by each work to examine the performance and approve the results?
3. What are the metrics used for evaluation purposes?
4. What are the contributions of each proposed work?

The publisher's name and the publication year of each research paper are mentioned also in the table. Each research paper is then analysed and the advantages and disadvantages are listed in the next section. Finally, we identified the future scope of the IDS based on machine learning methods research by highlighting the challenges of the design of an efficient model in terms of accuracy, training time and security issues.

Strength and Weakness of the Proposed Literature Works for IDS This section investigates the previous related works by listing the strengths and weaknesses points of each work.Table 3.2 shows the Strength and weakness points of each related work.

3.1 Discussion and Challenges

A. Results:

The use of state-of-the-art DL methods is more efficient than traditional machine learning methods and has led to better cyber security strategies that perform better in data analysis. The increase in the size of the dataset led to less accuracy in the multi-class classification attacks. Traditional machine learning methods become incompatible with high-dimensionality learning data. DL methods are powerful due to its primary characteristics: Hierarchical feature representations and Long-term dependency learning.This result can be noticed from the related works that used DL methods and compared their performance with traditional machine learning methods [9, 10, 12].

Internet of things architecture consists of several layers. The network layer is responsible for transferring packet data between hosts. It is vulnerable to many security threats. Many security frameworks have been proposed in the literature to address security issues [38, 110]. Most of these frameworks require the consideration of storage as well as the computational power of IoT devices. A combination of IDS with DL algorithms can offer an intelligent solution to address security threats and prevent attacks in the IoT environments by keeping in mind the shortcoming of IoT devices.

New security solutions such as Blockchain technology can be integrated with DL methods and IDS in order to enhance security and offer confidentiality, trust, integrity, privacy, etc [110].

Feature selection techniques can be used by DL methods to enhance the performance of IDS. The features are important for the prediction and their importance in the whole datasets are not equal. The complexity of training time can be reduced by using the feature selection technique and efficient work can be proposed with high accuracy at the same time [50, 30].

Data balancing is an important issue to be taken into consideration when dealing with datasets of intrusion detection. There are many powerful techniques for data sampling. Testing the performance of IDS based machine learning algorithms is done by finding the outcome of different statistical metrics. The mostly used metrics for that are accuracy, precision, F1-score and recall [8, 10, 32, 110].

A good dataset plays a vital role in model training. The up-to-date dataset contains more information about new attacks. The problem of unbalancing in the dataset can be solved by

oversampling techniques to improve the class distribution of the dataset. The GAN method is used in the literature that captures the real data distribution and then generates a specific type of data (attack) to reduce imbalance [35].

Ensemble learning aims to find the best set of classifiers and the best way to combine them to improve the overall performance of classification. It can be a good choice to deal with big data size then can be divided into small sets where each set can be trained by a single classifier. The performance of ML and DL methods in IDS is the superior performance of other traditional methods. Most of the related works that used ML and DL methods in the IDS achieved a high accuracy rate and improved the generalization ability.

In general, the use of DL methods is more efficient than traditional machine learning methods, while the performance varies between them depending on the type of the model and the optimization techniques used to optimize it.

B. Challenges:

1. Designing a secure system based on the use of DL methods does not necessarily guarantee all the security issues like integrity, trust, transparency, confidence, etc. ML and DL methods have been used previously in several security applications [111]. The IDS aims to prevent cyber security from different attacks while DL methods play an important role in supporting IDS with solutions to successfully secure the system from known and unknown threats efficiently in terms of accuracy, training time, etc. Choosing the related aspect to be achieved in the security system depends on the domain application, quality of data, the method used, and the data engineer's experience [112].

2. Each model from the previous studies has its own strength and weakness, and has been evaluated over one or two datasets which make it suitable for a particular type of attack. The models are not generalizable due to outdated datasets.

3. A good dataset plays a vital role in model training. The IDS data obtained from a real environment regularly contains a huge amount of normal behavior with a minority of attacks behavior data. Thus, the number of attacks is imbalanced. The imbalanced dataset affects the model performance and will result in poor recognition of attacks as the model will pay attention to normal behaviors. Therefore, the imbalanced dataset of IDS became a major challenge. The general technique to increase minority in the dataset is oversampling.

4. Most of the previous research relies on the old dataset to evaluate the performance of the proposed system. The old datasets contain old traffic and do not represent a real and recent attack scenario. Therefore, using recent datasets to evaluate IDS would be more efficient.

5. An efficient, dynamic and lightweight model design is a challenge for the IDS of the IoT environment where the memory storage and battery lifetime are big issues to be considered in the IoT devices.

6. The large volume of data represents a big challenge in the IDS design. The data are generated from different resources. Therefore, structured, unstructured, and semi-structured are included. This requires an efficient technique to analyze and manage various large quantities of data.

7. The datasets require passing through a set of pre-processing operations before providing it in any DL classifier model [113]. Changing the scale and distribution of input data may only be useful for the model that depends on the calculations of weighted sums, such as the neural network. The scale method may result in sensitive input data that can change the model results.

8. Tuning the neural Network Parameters is an (NP) problem that may require the run of several attempts in order to find the optimal value of each parameter that optimizes the model performance [114]. There is no optimal method to be used for that. Most of the previous research depends on the trial-and-error method which is time-consuming and may exhaust resources.

9. Most of the literature research papers that are published in the ML and DL based IDS domain focused on the improvement of NIDS. Few types of research focused on the other types of IDS (HIDS, SIDS, and AIDS).

10. Most of the dataset attributes are not clearly described in a way that can be easily understood by the researchers [115]. A well-described input and output attributes of the dataset are critical and helpful in the designing of IDS to achieve meaningful progress in performance.

C. IDS Based ML/DL Methods Framework:

Based on the analysis of the previous literature research works, a framework for the application of ML/DL methods on the IDS is proposed. The framework shows the pipeline of the IDS works divided into two main phases: The data preparation phase and the model design phase. In the data preparation phase, the IDS dataset is chosen and passes through a set of pre-processing steps. In the model design phase, a ML/DL method is used to be applied in the classification task of IDS. The aim of using ML/DL method is to increase detection accuracy and reduce error rate. The details of the framework phases are as follows: In the data preparation phase, cleaning, normalization, and transformation are the main steps that convert the dataset into a consistent form with no missing values. Normalization is an important step that converts data into a suitable range. The dataset is divided into a subset of three: training, testing, and validation sets.

In the model design phase, the type of ML/DL method to be used for the classification task is chosen. As summarized in the literature, there are many methods that can be used for this task [116]. The choice of the best method depends on the application domain of IDS. The tuning of the ML/DL model is essential to configure its setting to the best that guarantees the best results and accuracy.

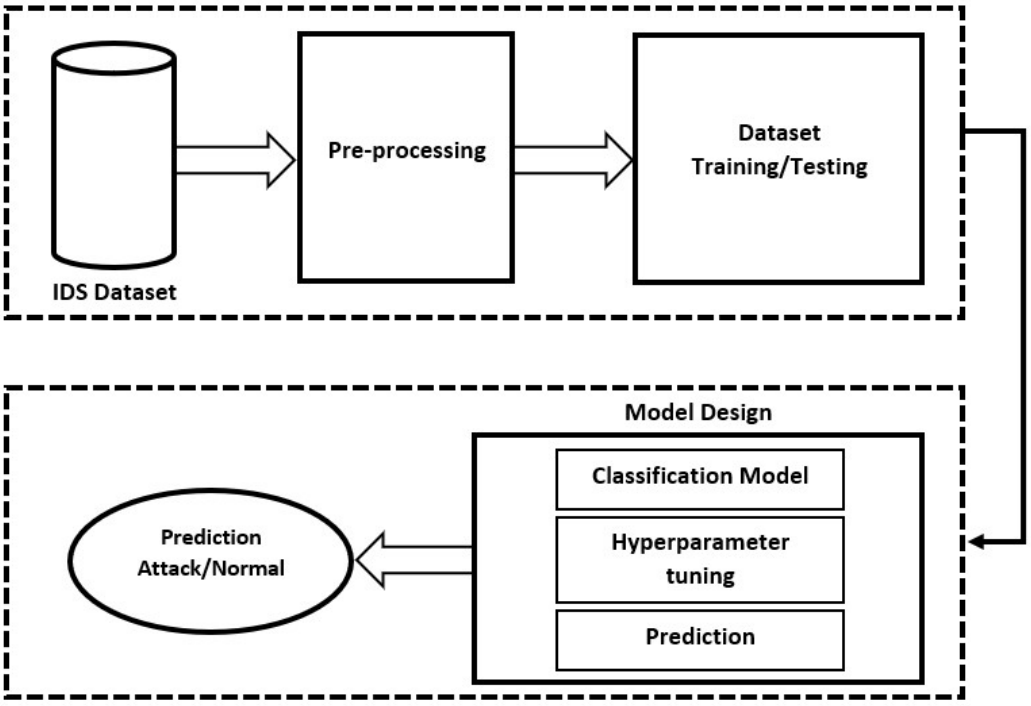


Fig. 3.1 IDS based ML/DL methods framework.(Source: Author own creation)

3.2 Related Works

Many experts in the field conducted various types of research on these topics. Alzahrani and Alenazi [117] conducted a research, in 2021 titled "Designing a NIDS Based on Machine Learning for Software Defined Networks." They used three machine learning methods: DT, RF, and XGBoost; which were compared to determine the most effective. Their research had a result of 95.95% accuracy. One of the strengths in this study was that the proposed model was investigated over a multi-class classification. Another study is [118], "Sequential Model Based IDS for IoT Servers Using DL Methods", done by Zhong et al. 's, in 2021. This study concluded that their proposed sequential model-based IDS was able to effectively detect suspicious anomalies in network traffic, higher than F1-score in traditional methods. They utilized GRU, Text-CNN, and MLP for machine learning methods. However, the limitations of this research circulated around the complexity and computational value of the GRU structure. Both Alzahrani and Alenazi's study, and Zhong et al. 's 2021 research intended to enhance network security through advanced intrusion detection methods. The first study relied on traditional machine learning, whereas the latter investigated the potential of DL models to identify anomalies in IoT servers.

Additionally, Mahbooba et al. conducted a research in 2021, titled, "Trust in Intrusion Detection Systems: An Investigation of Performance Analysis for Machine Learning and DL Models" [119]. This research utilized machine learning methods: DT, KNN, RF, NB, LSTM, and GRU; and datasets: WSN-DS and KDD-Cup 99. They suggested the use of AI-solutions in IDS can promote trust and precision. The limitations of this study included that the result showed no advantage of either the traditional machine learning or DL among the chosen datasets. To add to that, Muhammad Ashfaq Khan [120] wrote an article, in 2021 named, "Hybrid Convolutional Recurrent Neural NIDS". The machine learning methods he used were: CNN and RNN; and the datasets he used were: CSE-CIC-IDS2018. His proposed model resulted in an accuracy rate of 97.75% against high malicious attack detection rate. A disadvantage of this study was that it was only tested against a single database, however, the advantage was that class imbalance was dealt with. Authors of another study, Yao et al.[121] wrote, "IDS in the Advanced Metering Infrastructure: A Cross-Layer-Feature-Fusion CNN-LSTM-Based Approach". This study was published in 2021and used LSTM and CNN as machine learning methods. Their datasets consist of KDD-Cup 99 and NSL-KDD. Their experimental results showed that the proposed model outperformed both single DL components and previous models. A disadvantage of this study is that the detection effect of the U2R attack was not ideal.

"Hybrid Model for Improving the Classification Effectiveness of Network Intrusion Detection" is a research article written by Dutta et al. [122]. Their machine learning approach utilized: Classical AutoEncoder and (DNN); and used data sets: UNSW-NB15. Their proposed model increased accuracy and improved generalization ability. To add to that, an article published in 2021, named, "IDS for the Internet of Things Based on Blockchain and Multi-Agent Systems" found that using DL Methods enhances IDS of IoT Devices[123]. The authors used a machine learning method, ANN, and a dataset, NSL-KDD. A study published in 2020 [124] titled, "NIDS Using Supervised Learning Paradigm" proposed a method suitable for real-time IDS with an accuracy of 76.96%. The machine learning method they used is MLP, and the datasets they used is UNSW-NB15. Moreover, an additional study published by Guojie Liu and Jianbiao Zhang in 2020 [123] suggests that their proposed model improves the accuracy, check rate, and minimizes false positives. They used a machine learning base: CNN; and dataset: KDDCUP99 and NSL-KDD. Furthermore, in 2020, Tongtong Su, Huazhi Sun, Jinqi Zhu, Sheng Wang, and Yabo Li [125]published the study, "BAT: DL Methods on Network Intrusion Detection Using NSL-KDD Dataset". The machine learning methods they used are: LSTM and CNN. Their study had a result of 84.24% accuracy.

Furthermore, a study published in 2020, titled, "An Integrated IDS Using Correlation-based Attribute Selection and Artificial Neural Network" had an accuracy of 97.49 and specificity of 99.31 for NSL-KDD; and an accuracy of 96.44 and specificity of 98.4 for UNSW-NB. The machine learning method they used is ANN, and the datasets included are: NSL-KDD and UNSW-NB15 [126]. This research proved that the integration of CFS and ANN increases precision, however, a drawback was that it was time consuming. Another study published in 2020 called, "CNN-Based Network Intrusion Detection Against Denial-of-Service-Attacks" had an accuracy of 99% for KDD-CUP 99 and 91.5% for CIC-IDS2018. They used CNN as the machine learning method and KDD-CUP 99 and CSE-CIC-IDS2018 for datasets [127]. A drawback to this study is that the increase in the number of convolutional layers increased the complexity of the training. Additionally, the research published in 2020 by Bambang Susilo and Riri Fitri Sari [18] proved that RF and CNN increases accuracy. In this research they used RF, SVM, CNN, and MLP as machine learning methods. For the datasets they used BoT-IoT. Guoling Zhang, Xiaodan Wang, Rui Li, and Yafei Song researched "Network Intrusion Detection Based on Conditional Wasserstein Generative Adversarial Network and Cost-Sensitive Stacked Autoencoder". The machine learning methods they used are: CWGAN and CSSAE. The datasets they used are: NSL-KDD and UNSW-NB15. This proposed model improves the detection accuracy of small and unfamiliar attacks.

The drawback to this research was that an old database was to evaluate the performance [74].

A study titled, "An Improved BIGAN Based Approach for Anomaly Detection", found that the suggested approach increased the performance of BiGan on anomaly detection tasks. This study was published in 2020 and conducted by S. Emre Alptekin and M. Oguz Kaplan. A disadvantage to this is that the suggested approach only works for binary classification of attacks rather than multiclass classification [128]. Also, another study, published in 2022, titled, "Network Traffic Anomaly Detection Using PCA and BiGAN" showed that an improvement in the performance and training time was achieved. They used BiGAN as a machine leaving method and KDD-CUP 99 as a dataset [129]. The authors of this study used an old dataset to evaluate this performance, but also investigated the importance of feature reduction in improving the overall performance.

Another research [130] showed that GAN can improve performance by balancing the imbalanced dataset and proposed model to improve accuracy. This research is titled, "Generative Adversarial Networks Assisted Intrusion Detection Systems". The authors used GAN in their machine learning method, and NSL-KDD as their dataset. The model they proposed solved the imbalance problem in the dataset by using GANs, however, it was considered to be time consuming. Another research, "Toward a Lightweight IDS for the Internet of Things" .The authors used SVM as a machine learning method. Their research showed that the SVM-based IDS can perform satisfactorily in detection of attacks. The drawback in this research is that only one type of attack is investigated in the research [131].Another research within this area of study is, "A Novel Two-Stage DL Model for Efficient Network Intrusion Detection" Their research used the SAE machine learning method and KDD-CUP 99 and UNSW-NB 15 for their datasets. Their research resulted in a recognition rate of 99.996% for KDD-CUP 99 and 89.134% for UNSW-NB 15. An advantage of this research is that the proposed model learns the feature representation to avoid overfitting and increase accuracy [132].

Additionally, a research by Bahram Hajimirzaei and Nima Navimiour [133] showed that their suggested method has a 2.23% improvement in correctly classified instances and a decrease in incorrectly classified instances. These researchers used MLP, Heuristic Algorithm, and Fuzzy Clustering Algorithm. The dataset used is NSL-KDD. The addition of two algorithms to ANN was costly. Moreover, a study that showed high accuracy with DNN for binary and multiclass classification, was in "Intrusion Detection Using Big Data and DL Techniques". This study was conducted by Osama Faker and Erdogan Dogdu. The authors used DNN, RF and GBT for the machine learning methods. For datasets, they used

UNSW-NB15 and CICIDS2017. The benefits of this research includes that the proposed model integrated big data technologies and DL techniques to improve IDS. A drawback was noted that better feature selection techniques can be used [134].

Furthermore, a study by Xiao Y. and Xiao X. concluded that their experimental results show that IDS based on the S-ResNet achieves better performance in terms of accuracy and recall compared to the existing IDSs. They used ResNets for their machine learning method and NSL-KDD for their datasets [135].

Moreover, a study conducted by Khater, et al. showed a result of 94% accuracy, 95% recall, and 92% F1-Measure in ADFA-LD and 74% accuracy, 74% recall, and 74% F1-Measure in ADFA-WD. In this study, they used MLP as a machine learning method and ADFA-LD and ADFA-WD as datasets. A disadvantage stated that a more efficient learning algorithm could have been used instead of Backpropagation [136]. Another study titled "Towards Deep Learning- Driven Intrusion Detection for the Internet of Things" suggested that the IDS can detect real-world intrusions effectively. This study was done by Geethapriya Thamilarasu and Shiven Chawla. For the machine learning method, they used DNN; and used a dataset of 5 million network transactions from the six sensors distributed in a smart home network simulation. This approach did not require prior knowledge of captured network payload binaries, traffic signatures, or compromised node address. However, other types of attacks against the IoT were not investigated [137]. "Network Intrusion Detection Based on Deep Learning", published in 2019 showed that the results of their proposed method has a significant improvement over the traditional machine learning accuracy. In this research, they used RBM as a machine learning method and KDD-CUP 99 as a dataset. An advantage of this method is that a (DNN) increases detection rate, and a disadvantage was that an old dataset was used to evaluate the performance [138]. The article, "IDS for NSL-KDD Dataset Using Convolutional Neural Networks", was published by Yalei Ding and Yuqing Zhai in 2018. They used CNN as a machine learning method and NSL-KDD as datasets. Their research resulted in their proposed IDS model being superior in multi-class classification to the performance of models based on traditional machine learning methods. They also suggested that using a multi-stage feature with CNN improves detection rate [139].

Another study conducted in 2018 titled, "Improving performance of intrusion detection systems using ensemble methods and feature selection", published by Ngoc Tu Pham, et al. These researchers used bagging and boosting ensembles as a machine learning method, and NSL-KDD as datasets. They reached a conclusion that the bagging ensemble model produced

the best performance in terms of both classification accuracy and FAR when working with the subset of 35 selected features. The authors also suggested that ensemble learning improves classification accuracy with IDS [140], The article, “Design and Implementation of IDS using Convolutional Neural Network for DoS Detection”, by Nguyen, et al. showed an accuracy of 99.87% They used CNN as a machine learning method and KDD Cup 99 as dataset [141]. Another research published in 2018 by Li Z. and Qin Z., “A Semantic Parsing Based LSTM Model for Intrusion Detection,” dictates that their proposed model outperforms most of the standard classifiers and solves anomaly detection. They used LSTM as a machine learning method and NSL-KDD as a dataset. However, an old dataset was used to analyse the performance [142].

The study titled "Addressing the Imbalanced Data Problem with Generative Adversarial Networks for Intrusion Detection," conducted by Yilmaz et al., recommended using the UGR’16 dataset in an intrusion detection model. The research explores the application of GANs to address class imbalance in intrusion detection datasets. Their proposed approach employs GANs to generate synthetic samples for the minority class, with the goal of enhancing the performance of a Multi-layer Perceptron (MLP) model [99].

Additionally, a study titled, “Hybrid Deep Learning Based Attack Detection for Imbalanced Data Classification” found that a new IDS system based on CNN and LSTM Networks can improve the security of IoT devices. This study was done in 2022 by Almarshdi et al.The authors suggest that to enhance IDSs’ precision an architecture combining CNNs and LSTMs networks must be implemented, Their proposed model was evaluated on the UNSW-NB15 dataset and was compared to a basic CNN model and traditional machine learning classifier. The model scored a 92.10% accuracy on the UNSW-NB15 dataset and 89.90% for the basic CNN model. All in all, the results of this research show the prospect of a hybrid-CNN-LSTM architecture to improve performance in IoT systems [143].

In Table 3.1, A Comprehensive review of the most recent articles that used ML and DL to improve IDS with highlighting the strengths and weaknesses of each work. Based on the analysis of the literature review research papers, a framework for the application of ML and DL in the IDS is proposed. Table 3.2, showes the Strength and weakness points of each related work.

Table 3.1 Summary of related work research papers.

Citation	ML Method	Dataset	Metrics	Year	Results
Alzahrani and Alenazi [117]	DT, RF, XG-Boost	NSL-KDD	Acc, Prec, Rec, F1	2021	Accuracy of 95.95
Zhong, et al. [118]	GRU, Text-CNN and MLP	KDD-Cup 99 and ADFA-LD	F-Score	2021	The proposed method achieved higher F1-score than traditional methods
Mahbooba, et al. [119]	DT, KNN, RF, NB and LSTM, GRU	WSN-DS and KDD-Cup 99	Accuracy, precision, recall, and F-Score	2021	The using of AI-solutions in IDS can enhance trust and accuracy
Khan [120]	CNN and RNN	CSE-CIC-IDS2018	Accuracy, TP, TN, FP, FN	2021	The proposed model achieved high malicious attack detection rate accuracy of up to 97.75
Yao, et al. [121]	LSTM and CNN	KDD-Cup 99 and NSL-KDD	Accuracy, Precision, DR, F-score and FPR	2021	The experimental results demonstrated that the performance of the proposed model was superior to those of a single DL component and models proposed in previous studies.

Table 3.1 continued on next page

Table 3.1 (continued)

Citation	ML Method	Dataset	Metrics	Year	Results
Dutta, et al. [122]	Classical AutoEncoder and DNN	UNSW-NB15	Precision, recall, accuracy, F-score, False Positive Rate (FPR), ROC curve	2020	The proposed model increases accuracy and improves generalization ability.
Liang, et al. [123]	ANN	NSL-KDD	Accuracy, precision, recall and F-Score	2020	The using of DL methods enhances IDS of IoT device.
Mebawondua, et al. [124]	MLP	UNSW-NB15	Accuracy and False Alarm Rate	2020	The proposed method is suitable for real-time IDS with accuracy of 76.96
Liu and Zhang [144]	CNN	KDD-CUP 99 and NSL-KDD	Accuracy, Precision, Recall, F–Score and Error Rate	2020	The proposed model improves the accuracy and check rate, reduces the false positive rate.
Tang, et al. [145]	SAE and DNN	NSL-KDD	Accuracy, Precision, Recall and F–Score	2020	Accuracy 87.74 and 82.14 (binary-classification and multi-classification)
SU, et al. [125]	LSTM and CNN	NSL-KDD	Accuracy, TPR and FPR	2020	Accuracy of 84.25

Table 3.1 continued on next page

Table 3.1 (continued)

Citation	ML Method	Dataset	Metrics	Year	Results
Thaseen, et al. [126]	ANN	NSL-KDD and UNSW-NB15	Accuracy and Specificity	2020	Accuracy of 97.49 and Specificity of 99.31for NSL-KDD Accuracy of 96.44 and Specificity of 98.4 for UNSW-NB
Kim, et al. [127]	CNN	KDD-CUP 99 and CSE-CIC-IDS2018	Accuracy, Precision, Recall and F1-score	2020	Accuracy of 99 for KDD-CUP 99 and 91.5 for CIC-IDS2018
Susilo and Sari [18]	RF, SVM, CNN and MLP	BoT-IoT	Accuracy and Precision	2020	RF and CNN increases accuracy.
ZHANG, et al. [74]	CWGAN and CSSAE	NSL-KDD and UNSW-NB15	Accuracy and F-Score	2020	The proposed model improves the detection accuracy of minority attacks and unknown attacks
Kaplan and Alptekin [128]	BiGAN	KDD-CUP 99	Accuracy, Precision, Recall and F-Score	2020	The proposed approaches increased the performance of BiGAN on anomaly detection task.
Patil, et al. [129]	BiGAN	KDD-CUP 99	Accuracy, Precision, Recall and F1-score	2020	An improvement in the performance and training time is achieved.

Table 3.1 continued on next page

Table 3.1 (continued)

Citation	ML Method	Dataset	Metrics	Year	Results
Shahriar, et al. [130]	GAN	NSL-KDD	Precision, Recall, F1-score and Confusion matrix	2020	GAN improves performance by balancing the imbalanced dataset and proposed model improve accuracy
JAN, et al. [131]	SVM	Simulation Dataset of 100 normal samples and 100 intruded samples according to Poisson Distribution and CICIDS2017	Accuracy, TPR, TNR, FPR and FNR	2019	The SVM-based IDS can perform satisfactorily in detection of attacks
Khan, et al. [132]	SAE	KDD-CUP 99 and UNSW-NB15	Accuracy, Precision, Recall, F–Score and FAR	2019	Recognition rate of 99.996 for KDD-CUP 99 and 89.134 for UNSW-NB15

Table 3.1 continued on next page

Table 3.1 (continued)

Citation	ML Method	Dataset	Metrics	Year	Results
Hajimirzaei and Navimipour [133]	MLP, Heuristic Algorithm and Fuzzy Clustering Algorithm	NSL-KDD	MAE, RMSE, and the kappa statistic	2019	The proposed method shows a 2.23 improvement in correctly-classified instances and a decrease in incorrectly classified instances
Faker and Dogdu [134]	DNN, RF and GBT	UNSW-NB15 and CICIDS2017	Accuracy	2019	The results show a high accuracy with DNN for binary and multiclass classification.
Xiao and Xiao [135]	ResNets	NSL-KDD	Accuracy, recall and F-Score	2019	The experimental results show that the IDS based on the S-ResNet achieves better performance in terms of accuracy and recall compared to the existing IDSs.
Khater, et al. [136]	MLP	ADFA-LD and ADFA-WD	Accuracy, recall and F-Score	2019	94 Accuracy, 95 Recall, and 92 F1-Measure in ADFA-LD and 74 Accuracy, 74 Recall, and 74 F1-Measure in ADFA-WD

Table 3.1 continued on next page

Table 3.1 (continued)

Citation	ML Method	Dataset	Metrics	Year	Results
Thamilarasu and Chawla [137]	DNN	dataset of 5 million network transactions from the six sensors distributed in a smart home network simulation	Precision, Recall, F – Score	2019	The proposed IDS can detect real-world intrusions effectively.
Peng, et al. [138]	RBM	KDD-CUP 99	Accuracy, FPR	2019	The results show that the proposed method has a significant improvement over the traditional machine learning accuracy.
Ding and Zhai [139]	CNN	NSL-KDD	Accuracy, TPR and FPR	2018	The experimental results show that the performance of proposed IDS model is superior in multi-class classification to the performance of models based on traditional machine learning methods.

Table 3.1 continued on next page

Table 3.1 (continued)

Citation	ML Method	Dataset	Metrics	Year	Results
Pham, et al. [140]	Bagging and Boosting ensemble	NSL-KDD		2018	The bagging ensemble model produced the best performance in terms of both classification accuracy and FAR when working with the subset of 35 selected features.
Nguyen, et al. [141]	CNN	KDD Cup 99	Accuracy	2018	Accuracy of 99.87
Li and Qin [142]	LSTM	NSL-KDD		2018	Proposed model outperforms most of the standard classifier and solve anomaly detection
Yilmaz, et al. [99]	GAN	UGR'16	Accuracy, recall and F-Score	2020	Proposed GAN MODEL shows that a balanced attack sample dataset, produces more accurate results than an imbalanced one.
Thockchom, et al. [146]	Ensemble method	CICIDS-2017 KDD99 and UNSW-NB15	precision, recall, F1 score, accuracy and confusion matrix	2023	proposed model has been shown to improve the overall performance significantly.

Table 3.1 continued on next page

Table 3.1 (continued)

Citation	ML Method	Dataset	Metrics	Year	Results
Almarshdi, et al. [143]	CNN and LSTM	UNSW-NB15	precision, recall, F1 score, accuracy	2023	The proposed model achieved 92.10% accuracy.
Sattarpour, et al. [147]	BERT (Bidirectional Encoder Representations from Transformers)	CICIDS-2017	Accuracy, Precision, Recall,F1 Score	2024	The BERT-based IDS outperforms traditional methods, achieving high accuracy, precision, recall, and F1 score, making it highly effective for IoT environments.
Nandanwar and Katarya [148]	CNN and LSTM	KDD99	Accuracy, Precision, Recall,F1 Score	2024	The proposed deep learning-based IDS using CNN and LSTM networks shows improved performance in accuracy matrix effectively detecting intrusions in Industrial IoT environments.

Table 3.1 continued on next page

Table 3.1 (continued)

Citation	ML Method	Dataset	Metrics	Year	Results
Yaras and Dener [149]	CNN and LSTM	KDD99	Accuracy, Precision, Recall,F1 Score, Confusion Matrix	2024	The hybrid DL model proposed by Yaras and Dener (2024) significantly improves intrusion detection in IoT environments, achieving higher accuracy, precision, recall, and F1 score compared to traditional methods.

Table 3.2 Strength and weakness points of each related work.

Related Work (Citation)	Strength	Weakness
Alzahrani, and Alenazi [117]	The proposed model is investigated over multi-class classification.	The proposed model used an older dataset for evaluation, more benchmark cyber security datasets can be used to achieve generalization. Traditional machine learning classification methods are used instead of state-of-the-art DL methods.
Zhong, et al. [118]	Deep learning methods (state-of-the-art) are used to design the model.	The structure of GRU is complex and computationally expensive. The integrity of the data was limited.
Mahbooba, et al. [119]	The proposed model investigated how to enhance the trust in IDS.	The performance-based comparison shows no superiority of one model (traditional machine learning and deep learning) among the chosen datasets.
Khan [120]	The problem of class imbalance was handled.	The proposed model was tested on a single dataset.
Yao, et al. [121]	The proposed model guarantees AMI communication security.	The detection effect of the U2R attack was not ideal.
Dutta, et al. [122]	Feature engineering method was used to increase performance of IDS.	The proposed model did not address the problem of the classification of multiple attack families.
Liang, et al. [123]	Applied new technology, Blockchain and multi-agent for IDS.	Computational power problem was not considered in the system design. Testing the performance was done over an old dataset.
Mebawondua, et al. [124]	Using Gain ratio technique for feature selection and MLP for classification proposed a lightweight IDS.	The study fails to test the performance of the model using different numbers of attributes.

Table 3.2 continued on next page

Table 3.2 (continued)

Related Work (Citation)	**Strength**	**Weakness**
Liu and Zhang [144]	Applied a multi-class classification.	Old datasets were used for experimental evaluations.
Tang, et al. [145]	The proposed model deals with high dimensionality data.	Old dataset was used for experimental evaluations.
SU, et al. [125]	The proposed model can capture features of network traffic more comprehensively.	Does not evaluate the performance in terms of time complexity.
Thaseen, et al. [126]	The integration of CFS and ANN increases accuracy.	Time consumed is high.
Kim, et al. [127]	New types of data were used to train the model by generating a set of images from existing numerical samples in the datasets.	Increasing number of convolutional layers increases complexity of the training.
Susilo and Sari [18]	The authors investigated the model hyperparameters (number of epochs and batch size). Batch size speeds up the calculation process.	An old dataset was used to evaluate the performance.
ZHANG, et al. [74]	The proposed model solves the imbalance problem in the dataset by using GAN technology.	An old dataset was used to evaluate the performance.
Kaplan and Alptekin [128]	The dependency between generators and discriminators is reduced to force generator to produce more reliable data and improve the performance.	The proposed approach only considers binary classification of attacks while anomaly detection requires multi-class classification.
Patil, et al. [129]	The authors investigated the importance of feature reduction in improving the overall performance.	An old dataset was used to evaluate the performance.

Table 3.2 continued on next page

Table 3.2 (continued)

Related Work (Citation)	Strength	Weakness
Shahriar, et al. [130]	The proposed model solves the imbalance problem in the dataset by using GAN technology.	The proposed model was time consuming.
JAN, et al. [131]	The lightweight Ness measures of proposed algorithm is proven in terms of CPU time execution.	One type of attacks is investigated in the proposed work.
Khan, et al. [132]	The proposed model learns the feature representation to avoid overfitting and increase accuracy. The imbalanced datasets were treated in the proposed work.	Less accuracy achieved for UNSW-NB15 dataset with complicated types of attack compared with KDD-99 dataset.
Hajimirzaei and Navimipour [133]	Proposed new method for IDS in cloud environment that classifies instances correctly.	The addition of two algorithms to ANN was costly.
Faker and Dogdu [134]	The proposed model integrated big data technologies and deep learning techniques to improve IDS.	Better feature selection technique can be used.
Xiao and Xiao [135]	The simplified residual block prevents overfitting and improves the generalization ability of the model. The imbalanced datasets were treated in the proposed work.	The performance of the proposed model in terms of training time is not mentioned in the work.
Khater, et al. [136]	Lightweight intrusion detection model with less computational complexity achieved via n-gram transformation for feature selection.	More efficient state-of-the-art learning algorithms can be used instead of Backpropagation.

Table 3.2 continued on next page

Table 3.2 (continued)

Related Work (Citation)	**Strength**	**Weakness**
Thamilarasu and Chawla [137]	No prior knowledge of captured network payload binaries, traffic signatures, or compromised node address are needed for the proposed approach.	Other types of attacks against the IoT including location-dependent attacks are not investigated.
Peng, et al. [138]	DNN increases detection rate.	An old dataset was used to evaluate the performance.
Ding and Zhai [139]	Using multi-stage feature with CNN improves detection rate.	The FPR of Denial-of-Service attack was not satisfying.
Pham, et al. [140]	Ensemble learning improves classification accuracy in IDS.	An old dataset was used to evaluate the performance.
Nguyen, et al. [141]	An investigation about normalization technique for data input was provided with a comparison between performance of each case.	The proposed model investigated one type of attacks.
Li and Qin [142]	The semantic representation of network data was used with DL LSTM method which improves classification accuracy.	An old dataset was used to evaluate the performance.

The review of existing intrusion detection and data augmentation techniques has provided several key insights into the limitations and challenges faced by current approaches:

- **Imbalanced IDS Datasets and Their Impact on Detection Performance:**Most IDS datasets exhibit significant class imbalance, where attack samples are greatly underrepresented compared to benign traffic. This imbalance can severely hinder the detection capabilities of traditional machine learning models, as they tend to favor the majority class, resulting in low detection rates for rare and critical attacks.

- **Remaining Gaps in Intrusion Detection and Data Augmentation:**Despite the advances made in the field of intrusion detection and the use of synthetic data generation techniques, several gaps remain unaddressed, particularly in the handling of class imbalance. While oversampling techniques can increase the number of attack samples, they do not always lead to improvements in detection accuracy. Moreover, some (GAN)-based solutions suffer from issues such as mode collapse, where they generate a limited variety of attack samples. This narrow scope reduces the model's ability to learn from diverse attack scenarios, ultimately impairing its capacity to detect a wide range of emerging threats.

These identified shortcomings highlight the necessity for a more robust data balancing strategy. In this context, TDCGAN offers a promising solution. It addresses these challenges by generating realistic, diverse, and high-quality attack samples, thereby enhancing the accuracy and resilience of intrusion detection models and improving their capability to detect both known and novel threats.

Chapter 4

Benchmark Datasets

Since the advent of IDS technologies, significant efforts have been made to develop robust datasets for their evaluation [4]. Early endeavors produced what are now referred to as reference datasets, datasets widely used in numerous research studies, which have provided valuable insights to the research community. Over time, many newer datasets have been introduced, with several made publicly available for research purposes. However, a review of the literature reveals that some of these datasets, created decades ago, may no longer effectively address modern threats due to the rapidly evolving nature of cyberattacks. This section highlights the evaluation metrics commonly employed by researchers to assess their proposed methodologies. Additionally, it provides a comprehensive summary and analysis of the most prominent benchmark IDS datasets, shedding light on their strengths, limitations, and relevance to current cybersecurity challenges[99].
The following subsection provides a detailed summary and analysis of the benchmark IDS datasets used in this study. It will outline their key features, attack categories, and the role they play in assessing the effectiveness of the proposed model. This analysis ensures that the chosen datasets are relevant and suitable for evaluating network intrusion detection in realistic, imbalanced scenarios.

4.1 UGR'16 Dataset

The UGR'16 dataset [4], developed by the University of Granada in 2016, captures real network traffic from a medium-sized ISP between March and June 2016. During July and August, simulated attacks such as DoS, botnet activity, and port scanning were intentionally executed on the same ISP to generate additional traffic for testing purposes. The dataset includes NetFlow traffic flows with nearly 17 billion unique connections, of which over 98% represent normal traffic, making it highly imbalanced. To prepare the dataset, advanced

anomaly detection and network attack identification methods were applied, assigning labels to each record to indicate the type of attack, if any. Due to its large scale and recent data, UGR'16 is a valuable and up-to-date resource for training AI models and (NIDS) [150]. The performance of the proposed model will use the UGR '16 dataset. This dataset consists of various netflow v9 collectors that are tactically placed in the network of a Spanish Internet Service Provider (ISP) as shown in figure 4.1. This internet service provider makes the internet available for private networks like houses and business. Key features of the ISP network infrastructure are [5, 150]:

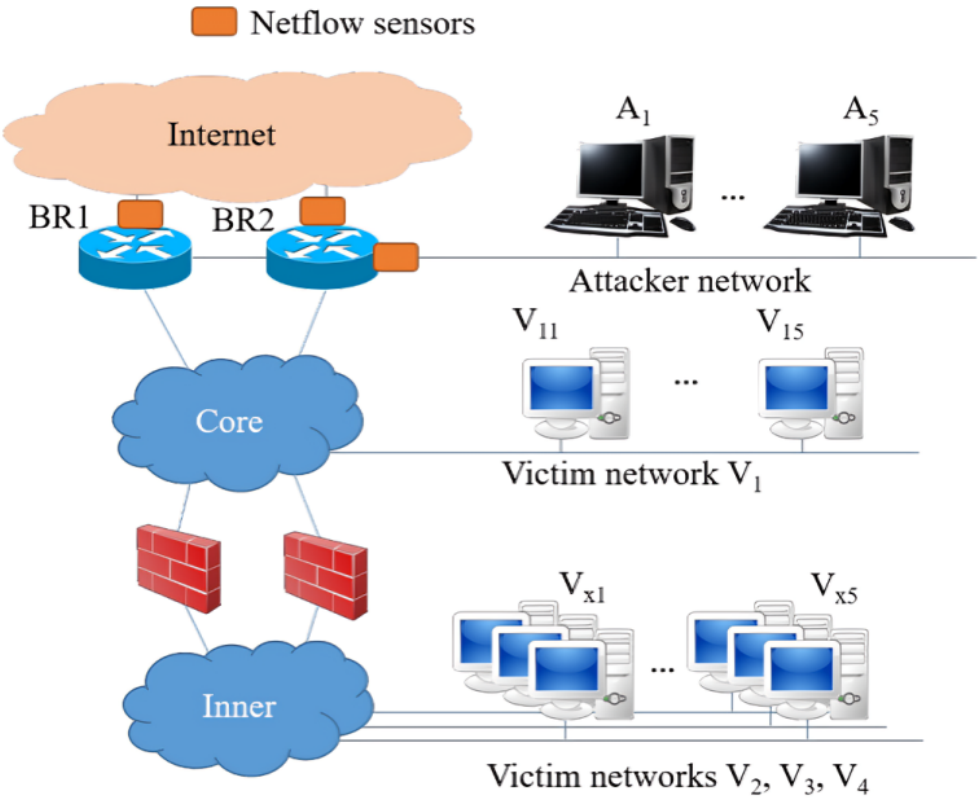


Fig. 4.1 Topology of the network of a Spanish Internet Service Provider (ISP) [4].

- Netflow probes are installed on the egress interfaces of the redundant border routers, BR1 and BR2, which grant internet connectivity. This configuration allows the collection of all inbound and outbound connections.
- The ISP consists of two distinct subnetworks: the core network, where services are exposed with the lack of firewall protection, and the inner network, where the firewall protection is offered to clients.

- Present at the highest level, is a five-node attacker network (A1-A5).
- Five victim machines (V11-V15) are established in the core network alongside authentic clients in V1 for data collection.
- An additional subset of 15 victim machines operates within the inner networks as follows: a victim network, V2, with machines from V21 to V25; a victim network, V3, with machines from V31 to V35; and a victim network, V4, with machines from V41 to V45. The dataset is separated into two: a calibration dataset and a test dataset. The calibration dataset has uncontrolled, unlabeled attacks and a general network traffic dataset collected from March through June 2016. The testing dataset has information collected in July and August 2016, used for model verification. Both datasets are bundled into weekly tar files of about 14 GB each. The calibration dataset has a total of 17 files, while the testing dataset has 6 files. IP addresses were anonymized using CryptoPan prefix-preserving anonymization method [151] which uses the nfanon tool [152]. Table 4.1 contains the list of different attacks with their corresponding labels in the UGR'16 dataset.

Table 4.1 List of attacks in UGR'16 dataset.

Attack	Label Description	Description
DoS11	Dos	One-to-one DoS (denial of service) attack, where the attacker A1 attacks the victim V21
DoS53s	Dos	The five attackers A1–A5 attack three of the victims, each one at a different network
DoS53a	Dos	The attacks are executed as in DoS53s, but now every victim is sequentially selected
Scan11	Scan11	One-to-one scan attack, where the attacker A1 scans the victim V41
Scan44	Scan14	Four-to-four scan attack, where the attackers A1, A2, A3 and A4 initiate a scan at the same time to the victims V21, V11, V31 and V41
Botnet	Nerisbotnet	Mixing botnet captures recorded elsewhere in a controlled environment with our background traffic

Table 4.1 continued on next page

Table 4.1 (continued)

Attack	Label Description	Description
IP in blacklist	Blacklist	It is an attack of class signature
UDP Scan	Anomaly-udpscan	Depending on the source port of the connection, each victim host is scanned through a specific range of 60 ports
SSH Scan	Anomaly-sshscan	An anomaly attack
SPAM	Anomaly-spam	An anomaly attack

Two-hour batches of artificial attack traffic were hence generated, during which each of the various attack types was carried out during the sessions. Two distinct alternatives were used in scheduling the execution of attack types [5]:

- Planned scheduling: every attack within the batch is executed at a predetermined and known time, which is determined by an offset from the initial batch time, denoted as t0.
- Random scheduling: the initial time for the execution of each of the attacks is randomly selected between t0 + 00h00m and t0 + 01h50m, thus restricting the total duration of the batch to a maximum of two hours.

The UGR '16 dataset, constructed from packet and flow data, has a total of about 16.9 billion anonymous network traffic flows. The network flow features extracted from the authentic network traffic are shown in Table 4.2.

Table 4.2 UGR'16 dataset network flow features.

Number	Feature Name	Type
1	Timestamp	date - time
2	Flow duration	continuous numeric
3	Source IP Address	categorical
4	Destination IP Address	categorical
5	Source port number	discrete numeric

Table 4.2 continued on next page

Table 4.2 (continued)

Number	Feature Name	Type
6	Destination port number	discrete numeric
7	Protocol	categorical
8	Flag	categorical
9	Forwarding status	numeric
10	Source type of service	discrete numeric
11	Total Number of packets	continuous numeric
12	Total number of bytes	continuous numeric
13	Class (Label)	categorical

The UGR'16 dataset is separated into 23 compressed files, each corresponding to a specific week. Of these, 16 files are allocated to the calibration class, while the remaining 6 are assigned to the test class. Each file, approximately 14 GB in compressed format, can be downloaded in CSV format.

Table 4.3 provides a summary of the types of attacks for each dataset, including UGR'16.

Table 4.3 Attacks type for each dataset for IDS.

Dataset	Attack's type
ADFA-LD	Hydra-FTP, Hydra-SSH, Adduser, Java-Meterpreter, Meterpreter, Webshell
ADFA-WD	password brute-force, java based meterpreter, add latest superuser, C100 Webshell and linux meterpreter payload
BoT-IoT	BENIGN, Service scanning, OS Fingerprinting, DDoS TCP, DDoS UDP, DDoS HTTP, DoS TCP, DoS UDP, DoS HTTP, Keylogging, Data theft
CICIDS2017	botnet (Ares), cross-site-scripting, DoS (executed through Hulk, GoldenEye, Slowloris, and Slowhttptest), DDoS (executed through LOIC), heartbleed, infiltration, SSH brute force, SQL injection
CSE-CIC-IDS2018	Heartbleed, Brute-force, DoS attack, Web attack, Infiltration attack, Botnet attack, DDoS attack, and Heartleech
KDD-Cup 99	DoS, privilege escalation (remote-to-local and user-to-root), probing.
NSL-KDD	DoS, privilege escalation (remote-to-local and user-to-root), probing.

Table 4.3 continued on next page

Table 4.3 (continued)

Dataset	Attack's type
UNSW-NB15	backdoors, DoS, exploits, fuzzers, generic, port scans, reconnaissance, shellcode, spam, worms.
WSN-DS	Four types of Denial of Service (DoS) attacks: Blackhole, Grayhole, Flooding, and Scheduling attacks.
DARPA	Denial of Service (DoS), Probe, Remote-to-Local (R2L) and User-to-Root (U2R).
ADFA-2013	Password brute-force attacks, Java-based meterpreter reverse shells, Linux-based meterpreter reverse shells, File upload attacks and Webshell-based attacks.
UGR'16	DoS11, DoS53s, DoS53a, Scan11, Scan44, Botnet, IP in blacklist, UDP Scan, SSH Scan and SPAM.
Kyoto 2006+	DoS (Denial of Service), Scanning attacks, Infiltration attacks, Malware attacks, Data theft and Web-based attacks.
NF-UQ-NIDS	DDoS (Distributed Denial of Service), Ransomware, Backdoor, Injection attacks, Cross-Site Scripting (XSS), Brute Force, SQL Injection and Port Scanning.

4.2 Other Datasets for IDS

The experiments in any research of the IDS domain require a network-based data representation. Benchmark datasets are a good choice for the evaluation and comparison between different IDS. The dataset contains labelled data (for supervised learning) which can be classified to either attack data or normal data. There are many representative datasets for the IDS. This section provides a literature survey of the existing IDS benchmark datasets that are used in the literature research with the properties for each one [11, 153].

The following section discusses prominent datasets, focusing on significant attributes within the field.

- **ADFA-WD dataset**: ADFA-WD dataset Australian Defence Force Academy Windows Data was developed in 2013 by the Australian Centre for Cyber Security at UNSW Canberra. It is a rich source for IDS research. ADFA-WD dataset provides labeled data, real-world scenarios, and system call level. Network attacks pertain to real-world scenarios allowing the dataset to be very realistic and challenging. Each system call trace is labeled with the corresponding attack allowing this dataset to be very specific [154]. This dataset also focuses on system call traces which provide details on process behaviours. Analyzing the system call traces of various attacks can help researchers better understand attack tactics to create effective countermeasures. This dataset includes 17,000 traces, 833 normal activity records and 12,426 attack traces that mimic contemporary attack scenarios like zero-day attacks, password brute force, privilege escalation, and web-based exploitation. This dataset provides a foundation for detailed behavioral analysis from capturing system calls for Windows systems in benign and malicious activities.

  Additionally, this dataset was created due to the unavailability of IDS datasets for Windows OS. A study published in 2016 titled, “Windows Based Data Sets for Evaluation of Robustness of Host Based Intrusion Detection Systems (IDS) to Zero-Day and Stealth Attacks” explored the reasons for its development. First, the creators wanted to provide a standard for the Windows IDS research community to be able to compare and analyze numerous techniques. Second, they wanted to simulate real-world attacks by adding zero-day attacks. Third, They wanted to create an easily accessible dataset for research purposes. This dataset provides a realistic and challenging environment for testing and developing IDS methodologies.

- **ADFA-LD dataset**: Australian Defence Force Academy - Learning Dataset was developed in 2013 by the by the Australian Centre for Cyber Security at UNSW Canberra is ADFA-LD [155]. ADFA-LD is a dataset designed for evaluating the performance of intrusion detection systems (IDS) targeted at the Linux platform. The dataset comprises 7,724 system call traces, of which 355 traces are for training, 1,827 for validation, and 5,542 for attack scenarios. The dataset encompasses more than 205 million system calls. It contains real attack scenarios such as privilege escalation, information disclosures, and denial of service, thus providing a conclusive and rigorous environment for IDS research. The Dataset is labeled, thus allowing the utilization of supervised techniques in machine learning, while it appeals to system call traces that offers a detailed understanding of behaviors. Upon investigation of this trace, the researchers will be able to gain information on some attack techniques, thus elaborating countermeasures. ADFA-LD is thus a valuable addition in the growth of intrusion detection research- especially for Linux based systems.

- **Bot-IoT dataset** Bot-IoT Dataset was introduced in 2019 for research in IoT security [156]. This dataset utilizes IoT Devices like fitness trackers and cameras. It is also provided in pcap, Argus, and CSV formats for flexibility. Additionally, it includes a number of botnet attacks like DDoS, DoS, data exfiltration, and command-and-control communication. This also allows researchers to understand the attack tactics to produce counterattacks. All in all, this dataset is able to guide researchers to enhance network security by detecting attacks on IoT Network [157].

- **CICIDS2017 dataset**

  CICIDS2017 is a well-known dataset used for evaluating IDS. This dataset was first introduced in 2017 and contains about 2.8 million records with more than 80 network features. CICIDS2017 collects real-world network traffic and gives labeled data. The attacks it collects range from DDos, DoS, port scanning, web attacks, and botnet attacks. It is available in formats: PCAP and CSV. The expansiveness and profundity of this dataset make it an important resource for concentrating on the complexities of organization intrusion detection [158]. Researchers often use the CIC-IDS2017 dataset to benchmark the performance of their intrusion detection algorithms due to its diversity and comprehensiveness [159, 160]. Furthermore, utilizing the CICIDS2017 dataset, experts can further develop and strengthen IDS.

- **CSE-CIC-IDS2018 dataset** CSE-CIC-IDS2018 is an extensive network traffic dataset developed by the Communications Security Establishment (CSE) and the Canadian Institute for Cybersecurity (CIC) in 2018 [161]. It comprises several real-world scenarios related to botnet attacks, DDoS attacks, web attacks and more and includes labeled network traffic allowing it to be appropriate for supervised machine learning mechanisms.This allows for a better analysis and training of the model. Additionally, it has 80 features and 16.2 million records. The network traffic it captures include benign and malicious activities [158].

- **KDD99 dataset** KDD99 dataset is a well known dataset in the field of IDS. Developed in1999 by the Defense Advanced Research Projects Agency (DARPA) to analyze the performance of many IDS techniques. It includes a diverse range of networks like TCP and UDP connections, HTTP requests, and FTP transfers, including five million records and 41 features. KDD-Cup 99 was developed for the KDD Cup 1999 competition, which focuses on creating effective techniques for detecting unauthorized network access [162]. The dataset features a substantial amount of network traffic data gathered from a simulated military network environment. It encompasses a range of network attacks, including DoS attacks, probing attacks, and user-to-root attacks, alongside normal traffic [163].

- **NSL-KDD dataset** Network Security Laboratory Knowledge Discovery and Data Mining (NSL-KDD) was designed to address the limitations of the KDD99 dataset. [10] It was developed in 2009 by Academics at the University of New Brunswick to strengthen NIDS. It has 125,973 records and 41 features. An important optimization feature is its ability to reduce redundancy in the dataset. Eliminating redundancy allows for a more efficient analysis and model training. This dataset also selects a more balanced dataset unlike KDD99. It accommodates both supervised and unsupervised learning methods. It is designed to depict authentic network traffic alongside several attack types, including denial of service (DoS), probing, and remote-to-local (R2L) attacks [163].

- **UNSW-NB15 dataset** The UNSW-NB15 dataset is intended as a large set of network intrusion and detection under bench-testing of its older datasets. This is a simulated real-world setting that captures various forms of modern attacks such as fuzzers, backdoors, DoS, exploits, and many other threats. It offers 2.5 million records and 49

features. The dataset is labeled, lending itself to the training of supervised learning methods, and provides a good number of extracted features, hence unpacking the nature of network traffic in detail. Additionally, this will give the opportunity to the researchers to develop and evaluate intrusion detection systems with better detection and response capabilities toward evolving cyber threats [164].

- **WSN-DS dataset** The WSN-DS is a dedicated resource for intrusion detection in Wireless Sensor Networks (WSNs). It has 374,661 data across 19 characteristics. This dataset addresses four main Denial of Service (DoS) attacks: Black Hole, Grayhole, Flooding, and Scheduling. The WSN-DS dataset consists of 19 features provided with 374,661 records, rendering this a comprehensive dataset for training and evaluating intrusion detection models.Therefore, as it simulates real-world WSN environments and retains key network metrics, this dataset allows researchers to formulate stronger and better security solutions for WSNs [165]. The information is Table 11 gives the properties of each dataset.

The table 4.4 summarizes the main standards applied in order to evaluate and classify several cybersecurity Datasets. These criteria offer a disciplined framework for comprehending the traits and fit of any dataset for various research uses. Access type describes whether access to the dataset requires payment or permission or is publically available. Year denotes the first version of the dataset's releasing year. In supervised learning, data labels—which indicate whether the observations are categorized as normal, abnormal, or linked with known attacks—help define the tasks involved. Crucially for model development, balanced reveals if the dataset exhibits an equal distribution of attack and normal events. Features reveals the focus of the data by indicating whether the dataset reflects host or network traffic. By means of traffic packet, flow data, or feature extraction from traffic flows, collected data reveals if the dataset gathers such information, therefore providing knowledge of the data collecting technique. Traffic Source separates real-world traffic captures from synthetic datasets produced under controlled circumstances. Finally, duration describes the length of time the network traffic of the dataset was captured over, therefore influencing the relevance and scalability of the dataset for training and evaluation. This structure guarantees a thorough awareness of the features of every dataset, hence enabling better-informed selections on which datasets to investigate intrusion detection systems for [150].

Table 4.4 Benchmark datasets for IDS.

Dataset	Access	Year	Label	Bal.	Feat.	Data	Source	Dur.
ADFA-LD [155]	Public	2013	Yes	No	H	Flows	R	5 D
ADFA-WD [154]	Protected	2014	Yes	No	H	Flows	R	5 D
BoT-IoT [156]	Public	2018	Yes	No	N	Flows	R	4 D
CICIDS-2017 [158]	Public	2017	Yes	No	N	Flows	R	5 D
CSE-CIC-IDS2018 [161]	Public	2018	Yes	No	N	Flows	R	7 D
KDD-Cup99 [162]	Public	1998	Yes	No	N	Features	S	3 M
NSL-KDD [163]	Public	1998	Yes	Yes	N	Features	S	N.S.
UNSW-NB15 [164]	Public	2015	Yes	No	N	Features	S	2 W
WSN-DS [165]	Public	2016	Yes	No	N	Packets	R	N.S.
DARPA [166]	Public	1998	Yes	No	N	Packets	S	7 W
ADFA-2013 [167]	Protected	2013	Yes	No	H	Packets	R	N.S.
UGR'16 [4]	Public	2016	Yes	No	N	Flows	R	6 M
Kyoto-2006+ [163]	Public	2006	Yes	No	N	Features	R	9 Y
NF-UQ-NIDS [168]	Public	2021	Yes	No	N	Flows	S	N.S.
Botnet [169]	Public	2010	Yes	Yes	N	Packets	S	N.S.

Key: D = Day, W = Week, M = Month, Y = Year, N.S. = Not Specified; H = Host, N = Network, R = Real, S = Synthetic; Feat. = Features, Data = Collected Data, Source = Traffic Source, Bal. = Balanced

Chapter 5

Proposed Model TDCGAN

Intrusion Detection Systems (IDS) are vital for upholding network security. They work by continuously tracking network traffic and host systems for potential security threats and malicious activities. The alarming rate in cyberattacks necessitates the development for automated and sophisticated IDSs. These systems can learn normal network behaviours, enabling them to identify any deviations from the norm, which can be indicative of anomalous or malicious behavior. Moreover, empirical studies show the efficacy of machine learning algorithms in their ability to detect malicious payloads within network traffic. However, the growing volume of data generated by IDSs creates a significant security risk, emphasizing the demand for more firm network security measures. The demonstration of traditional machine learning methods is heavily dependent on the dataset and its balanced distribution of classes. These imbalanced class distributions are a common issue in IDS datasets. Consequently, this imbalance can impact the accuracy of the machine learning models, resulting in poor detection and high rates of false alarms in traditional IDSs. To address this challenge, in this thesis we propose a novel model-based (GAN) named TDCGAN, which aims to improve the detection rate of the minority class in imbalanced datasets while maintaining efficiency. This TDCGAN model contains a generator and three discriminators, with an election layer incorporated at the end of the architecture. This allows for the selection of the optimal outcome from the discriminators' outputs [5].

To evaluate the proposed TDCGAN model, its performance is compared against various machine learning methods while using the UGR '16 dataset as a benchmark. Experimental findings reveal TDCGAN as an effective solution to addressing the challenge of imbalance dataset. Outperforming traditional oversampling methods, TDCGAN leverages the strengths of GANs and an election layer to achieve superior performance in detecting security threats within imbalanced IDS datasets.

5.1 Introduction

Data Science involves multiple stages, from data collection to preparation, exploration, and modeling. Since datasets vary across domains, the data-gathering process often introduces issues requiring careful handling to ensure accuracy before modeling. Addressing data quality is critical for effective data-driven solutions, particularly in security applications like Intrusion Detection Systems (IDSs). Machine learning (ML) has become a key enabler for IDSs by analyzing network activity, identifying unauthorized access attempts, and improving detection rates. Both supervised and unsupervised learning approaches have been explored, with unsupervised methods offering the advantage of detecting novel attacks without labeled data. However, imbalanced datasets—where one class significantly outnumbers others—can impact accuracy, necessitating effective data augmentation techniques[170].

To address the class imbalance challenge in IDS datasets, this thesis proposes a novel oversampling method based on Generative Adversarial Networks (GANs), called TDCGAN. The model consists of one generator and three discriminators, with an election layer selecting the best classification outcome. The generator produces synthetic samples from random noise to mimic real data, while each discriminator, configured with different architectures and parameters, evaluates the generated data. The election layer enhances classification performance by aggregating outputs, similar to ensemble learning. The generator is a deep multi-layer perceptron (MLP) with four hidden layers, incorporating an embedded layer for feature restructuring, ReLU activation, and dropout regularization [18, 11].

Each discriminator in TDCGAN employs distinct configurations to improve robustness. The first has three hidden layers, each with 100 neurons and a 10% dropout rate, while the second features five layers with varying neuron counts and 40% dropout. The third has four layers with decreasing neuron sizes and 20% dropout, using Leaky ReLU for activation. Two output layers are used—one with softmax and another with sigmoid activation, optimizing classification through binary and categorical cross-entropy loss functions. The model is evaluated using the UGR'16 dataset, chosen for its real-world network traffic and contemporary attack scenarios. Compared to traditional oversampling methods like SMOTE and ADASYN, GAN-based approaches offer a more dynamic framework for generating synthetic samples, making them highly effective for IDS applications[171, 172].

5.2 Data Preparation

The UGR'16 dataset, used in this research, is a massive collection of network traffic data as it contains a total of 16.9 billion records. Although DL algorithms are very strong, they require huge numbers such as the use of well-built CPUs, sufficient memory, and dedicated GPUs for data processing and efficient model training. This has led to the selective use of a part of the UGR'16 dataset which could provide some relief from resource constraints. In addition to this, the chosen part is comprehensive in terms of network traffic types of different scales, thus providing a representative sample for the development and evaluation of models.

The subset selection-all of which was for all kinds of attacks-used some measures to obviate any imbalanced distributions or biases. The measures were:

- **Stratified sampling**: The stratified sampling techniques in the subset selection ensured that each type of attack was proportionate to the total numbers in the overall collection. This showed that such attacks were sufficiently balanced in the subset.
- **Class-balancing**: These steps included the additional processes-by oversampling in minority classes, undersampling in majority classes, and so on-that ensured class balance in the subset and the prevention of this imbalanced distribution.
- **Randomization**: Randomization was used to eliminate any potential bias in the process of the subset selection. It was ensured that the selection was not on any specific predetermined order or biases.

The subset selection process intended a range of representative samples for attack types and implemented taking proper consideration of imbalances and possible biases to ensure quality analysis. The selected subset was then preprocessed, including removal of missing as well as duplicate instances. The entire detailed properties of the final subset are found in Table 5.1.

In network security, normal traffic is very vastly outnumbering malicious traffic, which creates an imbalanced dataset. Such imbalance poses a serious challenge in terms of machine learning, as models will be biased towards the majority class (normal traffic) [173].

Table 5.1 UGR'16 subset details.

From	To	Class Label	Counts	Percentage
27 July 2016	31 July 2016	background	197,185	98.5%
27 July 2016	31 July 2016	Dos	1169	0.6%
27 July 2016	31 July 2016	scan44	578	0.3%
27 July 2016	31 July 2016	blacklist	545	0.3%
27 July 2016	31 July 2016	nerisbotnet	227	0.1%
27 July 2016	31 July 2016	anomaly-spam	170	0.1%
27 July 2016	31 July 2016	scan11	126	0.1%

This thesis has considered major extra classes in case the dataset records contain mostly only the class equal to background. The other class labels are oversampled to obtain a balanced subset of the UGR'16 dataset. The original number of records and classes of the selected subset are given in Table 5.1.

Since machine learning algorithms operate with numerical data, certain categorical features in the dataset must be encoded, including protocol, source IP, destination IP, and class label. One-hot encoding is used for this purpose, converting categorical data into a numerical format suitable for machine learning models. This technique represents each category as a binary vector, where only one element is "1" (active) and the rest are "0." Additionally, the dataset is scaled using the MinMaxScaler from the Scikit-learn library, ensuring that all values fall within the range of (0,1).

Feature importance is assessed using a RF classifier, which measures importance based on the Mean Decrease in Impurity (MDI). MDI is a feature importance metric that indicates how much a feature reduces uncertainty (impurity) when used for splitting at decision nodes. It is crucial for feature selection and dimensionality reduction. The importance value for each feature is determined by summing the number of splits involving that feature across all decision trees, weighted by the number of samples each split affects.

The MDI values of the most numerically significant features in the UGR '16 dataset are shown in Figure 5.1. The relevance of showing these MDI values is to compare feature importance, helping researchers focus on meaningful attributes, explaining the model's behavior for transparency, and optimizing performance by potentially removing redundant

or irrelevant features. In the proposed model, all features are included in the process, with Source IP identified as the most critical feature.

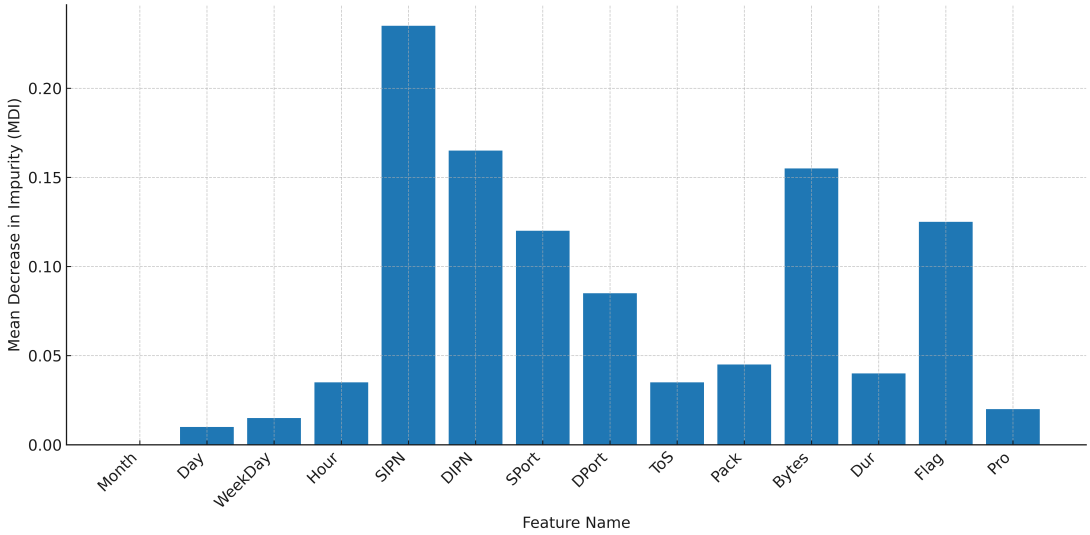


Fig. 5.1 The highest numerical features of the UGR'16 dataset based on the mean decrease in impurity (MDI).(Source: Author own creation)

MDI results help evaluate and interpret feature importance in ML model. Features with higher MDI values contribute more to reducing uncertainty (impurity) in the model. Helps prioritize the most influential features in classification or regression tasks.

5.3 Setup of Proposed Model

GAN is a DL method in machine learning used for creating new data. Specifically, it aims at learning input data without supervision and producing new samples from the original dataset. Many researchers have applied GAN into different applications, such as computer vision [174], time-series applications [175], health [176], etc. for passing marks as well as outperforming data generation. As many improvements and versions for the GAN are proposed, in order to fit it with the application domain and increase the performance and model accuracy [177, 178], this thesis proposes a new version of GAN called triple discriminator conditional generative adversarial networks (TDCGANs) as an augmentation tool to generate new data for the UGR'16 dataset with the aim to restore balance in the dataset by increasing minor attack classes.

The TDCGAN model consists of one generator and three discriminators. The generator takes random noise from a latent space as input and generates raw data that closely resembles

the real data, aiming to avoid detection by discriminators. This is due to the fact that each of the diversely architecture-discriminators has many and different parameter settings, each presenting with its own kind of (DNN). Each discriminator's role is to extract features from the output of the generator and classify the data with varying levels of accuracy for each of them. The end of architecture TDCGAN adds an election layer that takes the outputs from three discriminators and performs an election procedure for achieving the result with much precision in classification in the form of an ensemble method. The model aims to classify the input data into two classes, one normal flow for background traffic represented by 0 and the other anomaly flow for attack data represented with 1. Additionally, in the case of anomaly flow, the model classifies it as its specific class type. Figure 5.2 shows the workflow of the proposed TDCGAN model. The details of the generator and each discriminator, including their architectures and parameter settings, are provided below.

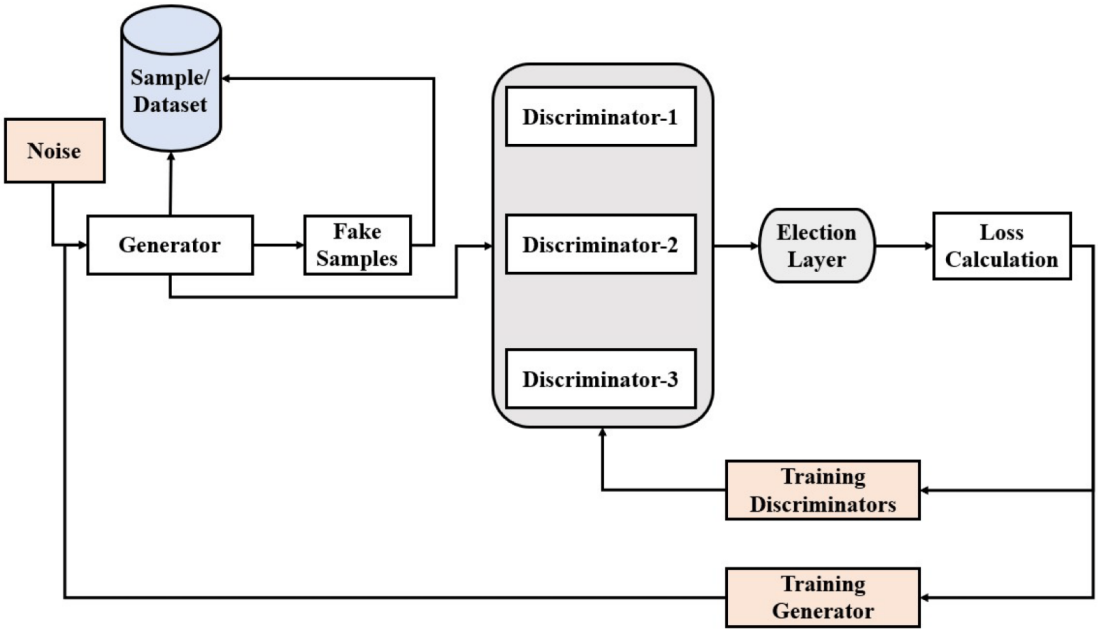


Fig. 5.2 Workflow of TDCGAN model. (Source: Author own creation)

The generator model is designed as a deep Multi-Layer Perceptron (MLP), which is a type of artificial neural network (ANN) consisting of multiple layers of neurons, and is widely used in machine learning tasks such as classification, regression, and pattern recognition. The MLP includes an input layer, an output layer, and four hidden layers. The generator initially samples a point from the latent space. The latent space is a multi-dimensional hypersphere of normal distributed points, where each variable is drawn from the distribution of the data in the dataset. An embedded layer inside the generator creates a vector representation for

the generated point, and through the training process, the generator learns to map points from the latent space into some specific output data, which also differs every time the model is trained. Then, to push it a step further, from a latent space Random points are used to generate new data using random points in the latent space. This means that these points will generate specific data. The discriminator carries out the necessary work of identifying the new data generated by the generator against the actual true data distribution. GAN is an unsupervised learning method. The simultaneous training of both generator and discriminator is adopted as in [66]. The output of the generator produces a batch of samples, meanwhile real cases from the domain are presented to the discriminator for judgment as to whether a sample is "real" or "fake". Thereupon, it is updated in terms of improvement in ability to distinguish real and fake samples during the next round. The generator, however, is updated based on whether it managed to trick the discriminator with its generated samples or not.

Thus, the two models form competitive relations and adversarial behavior among themselves in a game-theoretic context. In these relationships within a zero-sum game, the same implies that a fully functioning discriminator that classifies correctly between real and fake samples would earn rewards or receive no alteration to its model parameters by definition, while the generator receives a heavy penalty because of major adjustments made to its model parameters.

On the other hand, if the generator is successful in tricking the discriminator, then it would get a reward or the parameters of its model would remain unchecked. While the discriminator is, otherwise, subject to penalties. This is the standard GAN framework.

In addition to comparing the earlier models, it proposed a new TDCGAN model in which the generator essentially consumes points from the latent space and evolves data for the real data distribution found in the dataset using fully connected layers with four hidden layers, one input layer and one output layer. The discriminators try to classify the data from the class into the class using the fully connected MLP network.

MLP is popularly followed while using neural networks [179, 57]. The biggest factor is fast computing, simple implementation, and performance within smaller training datasets.

The generator model in this thesis learns to create new data that is similar to the attack traffic in the UGR ’16 data set, while the discriminator needs to distinguish between the real data coming from the dataset and the ones generated by the generator. Both generator

and discriminator models are conditioned by class labels during the training phase. This conditioning enables the generation of minor class data belonging to a specific class label by the generator model when utilized in isolation. The complete TGCGAN model can then be formed from all generators and three discriminator's models into one larger model.

Discriminators will have to train separately where the weight of each model is set as non-trainable in the TDCGAN model. This makes for updating the weights of the generator model only during training. This trainability modification applies only when training the TDCGAN model but not training independently for the discriminator. So here, the TDCGAN model is employed to train model weights of generators using the output and error computed by the models of discriminators. Therefore, a point in latent space would be given to the TDCGAN model. The generator model generates data on this input and feeds that data on the discriminator model. The model divergence is whether data are real or fake from the discriminator model then in the case of fake data classifies them according to their type of attack. A set of vectors (z) created randomly through a Gaussian distribution is fed into the generator, which maps them into G(z), having the same dimensions as the dataset. The discriminators take the output from the generator and try to classify it. The loss gets calculated between seen data and predicted data for an update of the weights of the generator and ensures that only the generator weights are being updated.

The cross-entropy loss function is deployed here to estimate the difference between actual data and predicted data, which is mathematically represented in the following manner:

$$[\text{Loss}]_{CE} = -\frac{1}{N}\sum_{n=1}^{N}[y_i \cdot \log(p(y_i)) + (1-y_i)\cdot\log(1-p(y_i))] \tag{5.1}$$

- y_i is the true label, with 1 representing malicious traffic and 0 representing normal traffic.
- $p(y_i)$ is the predicted probability of the observation i, calculated by the sigmoid activation function.
- N is the total number of observations in the batch.

The generator model has four hidden layers. The first hidden layer has a total of 256 neurons which works on a rectified linear unit (ReLU) activation function. Between the hidden layers a very short embedded layer is placed to efficiently transfer the high-dimension input data to the lower-dimension space for the neural network learning of the data relationship

and its efficient processing.

Thus, 128 neurons make up the second hidden layer, whereas the third and fourth comprise 64 and 32 neurons respectively, with an ReLU activation function employed with all of them and a regularization dropout of 20% is added to prevent overfitting at last. Softmax activation with 14 neurons in the output layer captures the features of the dataset.

Subsequent to the generator specification, we define the architecture of each discriminator in the model that we are proposing. Hence, each discriminator is thus an MLP, but with a different number of hidden layers, different number of neurons, and different dropout percentage, among others.The first discriminator is composed of 3 hidden layers with 100 neurons for each and 10% dropout regularization. The second discriminator has five hidden layers: 64, 128, 256, 512, and 1024 neurons. The dropout percentage is 40%. This most recent discriminator has 4 levels in hidden layers containing 512, 256, 128, and 64 neurons for each layer followed by a dropout percentage of 20%.

LeakyReLU(alpha=0.2) is used as an activation function for hidden layers in the discriminators. Each of the discriminators has two output layers with the Softmax activation function for one output layer, and the second output layer is associated with the Sigmoid activation function. The first output layer's loss is binary cross-entropy, while the second output layer's loss is categorical cross-entropy. The output is then taken from each of the discriminators and fed into the last layer of the model, which is where the final decision is made to get the best result.

A TDCGAN model is defined as a large model that is the combination of a generator model and three different discriminator models. That larger model is used for training the weights in the generator model using the output and error defined by the discriminators. The discriminators are trained separately by feeding real input coming from the dataset. This model runs for 1000 epochs, taking input with a batch size of 128. The optimizer is Adam with a learning rate of .0001. The proposed system has a capacity in its generator to train until it creates a new set of data samples behaving as the real distribution of the original data.

However, it does not often work well in real scenarios. It is necessary to maintain the relationships within the feature sets of the generated dataset by the generator and the dataset used by the discriminator may not even be exactly the same. This always leads to instability

during the generator training.

Discriminators in most cases typically converge very quickly during the early training process without the generator optimally shaping it as per the discriminator's learning outcome. This problem is solved by introducing a modification in the training scheme related to network intrusion detection applications by testing out the usage of three discriminators at the end with different architectural specifications and trying to eliminate the scenario of an early optimal discriminator, thus obtaining a more balanced generator-discriminator training environment.

5.4 Training Phase

The GAN framework implements a training method whose central objective is much for the generator to generate illusory data that, as much as possible, is similar to the real data and much for the discriminator to learn enough to be able to differentiate real samples from the fakes.

The generator and the discriminator will keep on undergoing training until the discriminator can no more distinguish between data as real or fake. This should mean, therefore, that the generated network can estimate the sample distribution in data and reach the Nash equilibrium.

To evaluate the model with accuracy, the usual split would be among training and testing data, enabling proper predictions on data not seen before. The training set will be used for model fitting, while the test set will be used to measure predictive precision of the trained model. In this way, data is separated into 70 percent for training and validation and into 30 percent for testing. The training set has minor class data and other class data. The minor class is used for data generation in the TDCGAN model. The generator is taught to model the anomaly sample distribution (minor class), keeping the discriminator constant; this output will be used as input to the discriminator for predicting it. Then comes the error calculation and updating the weight of the generator, and this continues until the discriminator will no longer be able to say whether input data came from the generator's output or the real anomaly dataset. In training, we ensure that all architectures undergo an equal number of epochs, and the weights from the final epoch are selected for use in generating artificial attack samples.

Furthermore, we start with iterative training and then the generator finally helps generate attack samples. Finally, these samples are incorporated into the training samples. In this way oversampling the very minor classes in the data set in the training phase. The model is then evaluated for its performance using the test data.

To compare the two groups of classifiers, the performance of our proposed classifier was established and tested against a test data set using accuracy and precision matching.

5.5 Experimental Results

Within this section, we methodically plan and execute a sequence of experiments, and subsequently analyze the obtained results.

5.5.1 Experimental Setup

Our experiments were carried out on the Python Colab Jupyter notebook that runs in the browser with the integrated free GPUs and freely installed Python libraries. The system setup is shown in Table 5.2.

Table 5.2 System environment specifications.

Unit	Description
Processor	Intel® Xeon®
CPU	2.30 GHz with No.CPUs 2
RAM	12 GB
OS	Ubuntu 22.04.4 LTS
PACKAGES	textbfTensorFlow 2.6.0®

5.5.2 Performance Metrics

In order to evaluate the effectiveness of our model, we use performance metrics like classification accuracy, precision, recall, and F1 score. For the purpose of measuring the total classification of data samples, we consider all predictions made by the model using the following equation:

The accuracy metric (Acc) at the output level is measured in terms of all the predictions made by the model from the data samples correctly classified:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{5.2}$$

where TP is the true positive, which represents the number of truly predicted anomalies; TN is the true negative, which indicates the number of truly predicted normal instances; FP is the false positive indicator that denotes the number of normal instances that are incorrectly classified as anomalies; and FN is the false negative indicator that indicates the number of the number of anomalies that are misclassified as normal. For a given class, precision, which quantifies the accuracy of its correct predictions, is calculated as the ratio of accurately predicted samples to the total number of predicted samples, as defined in the following equation:

$$Precision = \frac{TP}{TP + FP} \tag{5.3}$$

Recall, which is known as the true positive rate (TPR), is used to determine the ratio of correctly predicted samples of a particular class to the total number of instances within the same class as given by the following equation:

$$TPR(Recall) = \frac{TP}{TP + FN} \tag{5.4}$$

Finally, the F1 score computes the balance between precision and recall, evaluating the trade-off between the two metrics as given in the following equation:

$$F1score = 2 * \left(\frac{Precision\ x\ Recall}{Precision + Recall} \right) \tag{5.5}$$

5.5.3 Experimental Results and Analysis

The performance of the TDCGAN model is evaluated on the testing dataset. The previous metrics are used to evaluate and compare the results. The results after training the TDCGAN model for UGR'16 dataset balancing is given in Table 5.3.

To accurately evaluate our model's performance, the dataset is typically split into training and test sets to ensure reliable predictions on unseen data. The training set is used for model learning, while the test set assesses the predictive accuracy of the trained model. In this setup, 70% of the dataset is allocated for training and validation, while the remaining 30% is

reserved for testing.

Within the training set, data is further divided into the minor class and other categories. The TDCGAN model focuses on the minor class to generate synthetic data. During training, the generator learns to replicate the distribution of anomaly data (minor class) while the discriminator remains fixed. The generator's output is then passed to the discriminator for classification. The prediction error is computed, and the generator's weights are updated accordingly.

Training continues until the discriminator can no longer differentiate between real anomaly samples and synthetic ones. To ensure consistency, all architectures undergo the same number of training epochs, and the final epoch's weights are used to generate artificial attack samples.

Table 5.3 Performance evaluation metrics score for TDCGAN model.

Accuracy	Precision	F1 score	Recall
0.95	0.94	0.94	0.96

These performance metrics provide insights into the effectiveness of the model, particularly in terms of how well it distinguishes between normal and attack traffic. A 95% accuracy indicates that the model correctly classifies 95 out of 100 samples, which is generally high. A precision of 94% means that 94% of the samples classified as attacks are indeed attacks, but 6% are false positives. A 96% recall means that 96% of the real attacks in the dataset are correctly identified. A score of 0.94 indicates that the model maintains a good trade-off between avoiding false positives (high precision) and capturing all attacks (high recall).

The high accuracy (0.95), precision (0.94), recall (0.96), and F1-score (0.94) indicate that the model is performing well. Low false positives (precision 0.94) and high attack detection (recall 0.96) make the model suitable for intrusion detection.

Figure 5.3 shows the loss function while training the model for different numbers of epochs: 200, 400, 600, 800 and 1000.

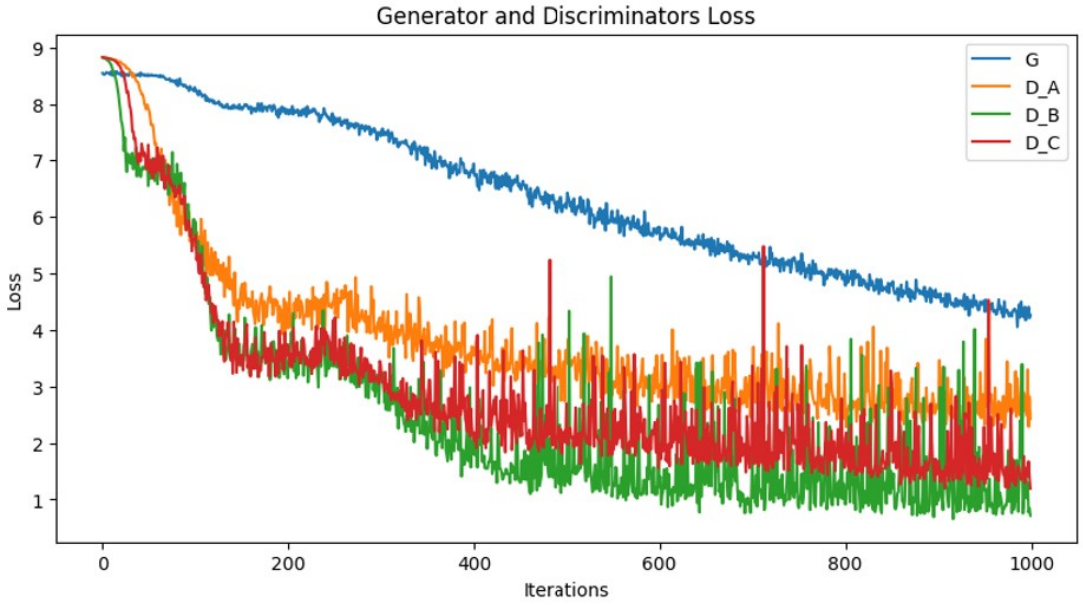


Fig. 5.3 The loss function components in the TDCGAN model: G represents the generator, D_A is the first discriminator, D_B is the second discriminator, and D_C is the third discriminator.(Source: Author own creation)

We compare the performance of the TDCGAN model for data balancing on the testing dataset with some resampling methods. The methods are as follows:

1. The synthetic minority oversampling technique (SMOTE) is a method for oversampling that produces artificial instances from the minor class. Its purpose is to create a training set that is either synthetically balanced or close to balance in terms of class distribution, which is subsequently utilized for classifier training. We use the implementations provided in the imbalanced-learn Python library, which provides a range of resample techniques that can be combined for evaluation comparison [22].

2. Random oversampling is used, which randomly duplicates the instances from the minor class [61].

3. SMOTE-ENN is a hybrid approach that combines SMOTE, which generates synthetic samples for the minority class through linear interpolation, with (ENN), which reduces the majority class by removing noisy samples, thereby improving class balance and data quality. [62].

4. Borderline-SMOTE is an enhanced oversampling technique based on SMOTE that focuses on minority instances near the decision boundary, generating new samples to improve class distribution.[24].

5. SVM-SMOTE integrates Support Vector Machines (SVM) with SMOTE, using support vectors as the basis for generating synthetic samples, making it an alternative to Borderline-SMOTE for better class separation. [63].

6. SMOTE-Tomek Links which denotes a technique used to detect pairs of closest neighbors within a dataset that exhibit dissimilar classes. Eliminating either one or both instances from these pairs, particularly those from the majority class, results in a reduction in noise or ambiguity within the decision boundary of the training dataset [64].

7. SMOTE-NC (synthetic minority over-sampling technique for nominal and continuous) which extends SMOTE to handle datasets with both categorical and continuous features by interpolating continuous values and using a mode-based approach for categorical attributes to ensure data consistency. [65].

8. CGAN (Conditional Generative Adversarial Network) is a type of GAN that generates data under a conditional setting, where both the generator and discriminator receive additional information, such as class labels or attributes, to control the output. This makes cGANs useful for structured data synthesis, ensuring generated samples belong to specific categories. [25].

9. CTGAN (Conditional Tabular GAN) is a GAN-based model designed for generating tabular data, which often includes mixed data types (continuous and categorical) and imbalanced distributions, using the principles of CGAN. [72].

   The results are listed in Table 5.4 and shown in Figure 5.4.

To ensure the reliability of our results, we accounted for the stochastic nature of the applied techniques by conducting multiple runs of the experiments. Since many of these techniques are inherently stochastic, their outcomes can vary between runs due to randomness in different stages of the process.

To mitigate this variability, we repeated the data generation and model training processes several times, computing the mean and standard deviation of the evaluation metrics. This approach helped reduce the impact of random initialization, synthetic sample generation, and adversarial training dynamics in GAN-based models.

Table 5.4 Performance evaluation metrics score for TDCGAN model and other resampling methods.

Model	Accuracy	Precision	F1 Score	Recall
SMOTE	0.88	0.86	0.87	0.91
Random Oversampling	0.85	0.89	0.90	0.88
SMOTE-ENN	0.86	0.89	0.90	0.89
Borderline SMOTE	0.84	0.87	0.87	0.88
SVMSMOTE	0.89	0.90	0.91	0.89
SMOTE-Tomek Links	0.90	0.87	0.89	0.87
SMOTE-NC	0.85	0.88	0.86	0.85
CGAN	0.83	0.83	0.83	0.83
CTGAN	0.76	0.76	0.76	0.76
TDCGAN	0.95	0.94	0.94	0.96

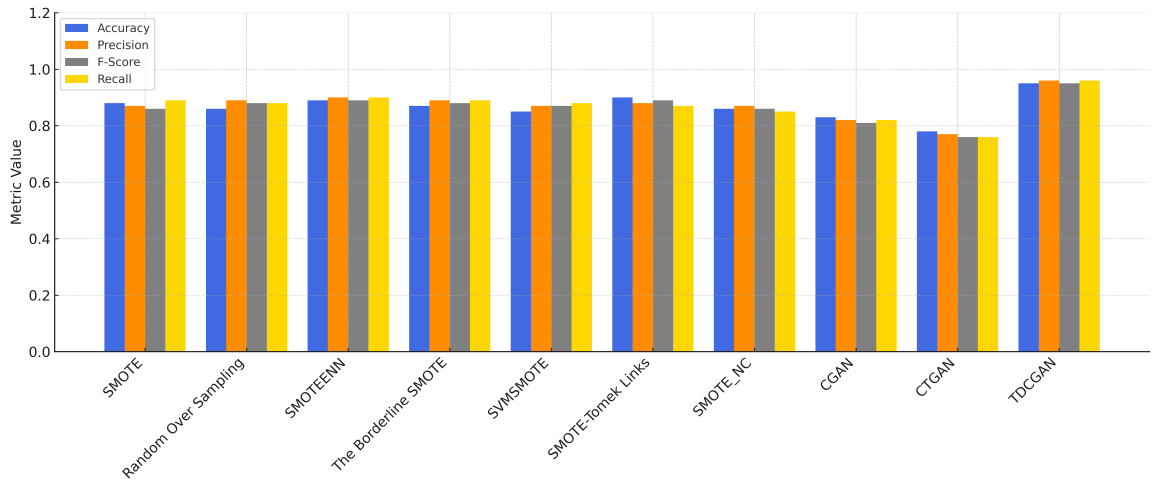


Fig. 5.4 Performance evaluation metrics score for TDCGAN model and other resampling methods.(Source: Author own creation)

The table 5.4 presents the performance of various oversampling techniques for handling imbalanced data in an IDS. The results are evaluated using Accuracy, Precision, F1-score, and Recall, with the TDCGAN model achieving the highest overall performance. Below is a detailed analysis of these results:

- SMOTE (Accuracy: 0.88, F1-score: 0.87, Recall: 0.91): SMOTE performs well in recall (0.91), indicating strong attack detection, but its precision (0.86) suggests a moderate number of false positives.

- Random Oversampling (Accuracy: 0.85, F1-score: 0.90, Precision: 0.89): High precision (0.89) means fewer false alarms, but slightly lower recall (0.88) suggests that some attacks might be missed.
- SMOTEENN (Accuracy: 0.86, F1-score: 0.90, Recall: 0.89): Similar to Random Oversampling, SMOTEENN maintains a balance between precision and recall, making it a stable choice for IDS applications.
- Borderline SMOTE (Accuracy: 0.84, F1-score: 0.87, Recall: 0.88): Slightly lower overall performance, but recall remains high, meaning it effectively detects attacks.
- SVMSMOTE (Accuracy: 0.89, F1-score: 0.91, Precision: 0.90): This method achieves a strong balance, with high precision (0.90) and recall (0.89), making it effective at identifying attacks while minimizing false positives.
- SMOTE-Tomek Links (Accuracy: 0.90, F1-score: 0.89, Precision: 0.87): It shows a balance between metrics, but slightly lower precision (0.87) indicates some false positives.
- SMOTE-NC (Accuracy: 0.85, F1-score: 0.86, Recall: 0.85):Performs similarly to standard SMOTE but has slightly lower recall, meaning it may miss more attacks.
- CGAN (Accuracy: 0.83, F1-score: 0.83, Precision: 0.83, Recall: 0.83):CGAN shows consistent but lower performance compared to traditional oversampling techniques. This suggests that it struggles with effectively generating diverse attack samples.
- CTGAN (Accuracy: 0.76, F1-score: 0.76, Precision: 0.76, Recall: 0.76):The lowest performance among all methods, indicating that CTGAN may not effectively model the minority attack class in this dataset.
- TDCGAN (Accuracy: 0.95, F1-score: 0.94, Precision: 0.94, Recall: 0.96):The best-performing method overall.High recall (0.96) indicates that almost all attacks are detected.High precision (0.94) ensures minimal false alarms. Balanced F1-score (0.94) confirms that the model is reliable and not biased towards any class.

This indicates that TDCGAN successfully generates synthetic attack samples, enhancing the model's robustness when dealing with imbalanced datasets.

Chapter 6

Comparative Analysis of the TDCGAN Model

With the increasing network throughput and rising security threats, intrusion detection systems (IDSs) have gained significant attention in computer science. While DL has been widely used in IDS development, a major limitation persists—the high imbalance in IDS datasets, where normal traffic samples significantly outnumber attack samples. This imbalance creates a bias in classification models, particularly DL-based approaches, leading to poor detection performance for minority attack classes.
Although various methods have been proposed to address this issue, TDCGAN is a recently introduced GAN-based approach specifically designed to tackle data imbalance in IDS datasets. Building upon previous research, this experiment aims to evaluate the effectiveness of TDCGAN in balancing data across four benchmark IDS datasets: CIC-IDS2017, CSE-CIC-IDS2018, KDD-Cup 99, and BOT-IOT. Furthermore, the study assesses the impact of data balancing on classification accuracy by comparing the performance of four different machine learning classifiers on both imbalanced and balanced datasets. The results highlight significant improvements in classification accuracy across all classifiers after applying TDCGAN, demonstrating its effectiveness in mitigating the challenges of dataset imbalance in IDS applications.

6.1 Introduction

The rapid growth of connected devices and complex networks has increased the risk of cyberattacks, highlighting the need for effective Intrusion Detection Systems (IDS) to handle imbalanced network traffic data [180]. Traditional machine learning algorithms struggle with

the disproportionate amount of normal traffic versus attacks, leading to biased models. To address this, techniques like SMOTE and ADASYN oversampling are used, but Generative Adversarial Networks (GANs) offer a more advanced approach by generating synthetic data. [181].

This study introduces TDCGAN, a novel GAN-based approach designed to address class imbalance in IDS datasets. Unlike traditional oversampling techniques, TDCGAN employs a triple-discriminator architecture with an election layer, enhancing classification accuracy through an ensemble-like selection process [5]. By applying TDCGAN to multiple benchmark IDS datasets (CIC-IDS2017, CSE-CIC-IDS2018, KDD-Cup 99, and BOT-IOT) and evaluating its impact on machine learning classifiers, this study demonstrates significant improvements in detecting minority-class attacks, reinforcing the necessity of advanced data balancing techniques in cybersecurity.

The TDCGAN generator uses random noise to produce synthetic data that mimics real data, while each discriminator, with unique architectures and parameters, extracts features and classifies outputs. The election layer selects the best discriminator based on classification accuracy, similar to an ensemble method. By combining multiple network architectures, TDCGAN enhances performance and effectively addresses class imbalance, particularly in UGR '16 dataset analysis [5].

To assess the effectiveness of the proposed GAN model, TDCGAN, across various datasets, this chapter introduces an evaluation framework. The general framework includes subjecting the TDCGAN model to multiple IDS datasets for a comprehensive performance evaluation. These datasets include CIC-IDS2017, CSE-CIC-IDS2018, KDD-cup 99, and BOT-IOT datasets. Afterward, a comparative analysis of four machine learning methods was conducted for intrusion detection classification tasks over these four datasets. The methods are: DT, RF, MLP, and NB.

This analysis was done both prior to and after employing data balancing methods. The results demonstrated that the performance of the machine learning model substantiating data balance showed an upgrade; hence, justifies the need to address issues of class imbalance in machine learning models even when the baseline performance appears to be good enough. By balancing the dataset, the machine learning model's ability to generalize and accurately classify instances from both classes improved significantly. This is crucial when accurate identification of both classes, including the minority class, is equally important, ensuring

balanced performance.

The key contributions of this chapter are summarized as follows:

1. Apply the TDCGAN model to address data imbalance in four intrusion detection datasets.
2. Employ different machine learning techniques to test intrusion detection classification across four benchmark datasets before and after data balancing.
3. Compare the performance of the utilized models.

6.2 Evaluation of TDCGAN on Diverse IDS Datasets

This study evaluates the effectiveness of the TDCGAN model for data balancing and intrusion detection using four diverse datasets: CIC-IDS2017, CSE-CIC-IDS2018, KDD-cup 99, and BOT-IOT. These datasets provide a comprehensive testbed to assess the model's performance across different network traffic scenarios.

1. **CIC-IDS2017 dataset:** The CIC-IDS2017 dataset has become one of the best-recognized datasets for intrusion detection systems performance evaluation in cybersecurity. Foots in the foreground at this dataset are network traffic in real-world situations that include normal traffic as well as various types of attacks, such as denial of service (DoS), distributed denial of service (DDoS), brute force, and botnet attacks. This can be considered a realistic measure of network traffic environments, both benign and malicious, for training and testing intrusion detection models. Owing to its comprehensive and diverse content, researchers now submit the CIC-IDS2017 dataset for performance evaluations of their intrusion detection algorithm [158].
2. **CSE-CIC-IDS2018 dataset:** The new CSE-CIC-IDS2018 corpus has labeled flows of network traffic with a number of attributes, such as time stamp, source/destination IPs, ports, protocols, and the attack type attached to each hyperparameter. This allows for examination in fine detail and the training of the model. Seven types of attacks are found in the dataset: Brute-force, Heartbleed, Botnet, DoS, DDoS, Web, and Internal Network infiltration [158].
3. **KDD-cup 99 dataset:** KDD-Cup 99 was intended for the KDD Cup 1999 competition, which aimed at delivering truly effective techniques for the detection of unauthorized access to the network. As expected, it has a large amount of network traffic data coming

from a simulated environment Miller and the reputed military field just because it composes and comprises well over 100 different kinds of attacks, including, DoS attacks, probing attacks, and user-to-root attacks, alongside large normal traffic [163].

4. **BOT-IOT dataset:** The comprehensive BOT-IoT dataset is solely dedicated to research in the area of IoT security. This consists of network traffic data recorded by a real-world IoT environment, portraying different IoT devices, such as an IP camera, smart thermostats, and a wearable fitness tracker. The dataset contains benign traffic and its aberrancies, which could be found in attacks concerning botnets on IoT devices [157].

6.2.1 Machine Learning Methods

In light of this, the four methods we have selected for this study considering their effectiveness as demonstrated by a plethora of previously conducted studies [182–185]. Our comparison is going to test the four aforementioned methods of data balancing and intrusion detection against our proposed model, TDCGAN, introduced in [5]. This is a brief synopsis of each model.

1. **Decision Tree:** Decision trees have been the most commonly used supervised learning technique for both classification and regression. In their method, the space of features is repeatedly partitioned into discrete regions based on input feature values. A decision criterion node in the tree indicates a feature, and its branches illustrate the likely result of the decision made from it [51, 52].

2. **Random Forest:** Random forests are an ensemble learning approach that builds several individual decision trees at training time and predicts with them using voting or averaging. Every tree is created using a random subset from the training data and is fed into a random subset of features, hence controlling overfitting and helping to enhance generalization performance [186, 187].

3. **Multilayer Perception (MLP):** An MLP, or multi-layer perceptron, is a kind of artificial neural network that is fully feedforward, and whose deep architecture has its layers remaining such that there is at least one output and one hidden layer sandwiched between an input layer and a number of hidden layers. Every node reaches every node in the adjacent layers, and weights are assigned for every connection. Therefore it uses backpropagation and gradient descent for training to learn the optimal weights that minimize a given loss function [53, 54].

4. **Naive Bayes:** NB is a classifier based on the probability model. It applies Bayes' theorem and the premise that all features are independent, which is the "naive" part of the model. NB can be very useful, particularly in text classification, despite its naivety. Compute now the probability of each class against the input features and return the one with the maximum probability [48, 49].

As proposed in [5], the generator utilizes points from the latent space to generate data that matches the distribution of real data in the dataset.

This process involves employing fully connected layers comprising an input layer, four hidden layers, and an output layer. The generator produces the data, $\hat{y}$, by applying the function $f(x,w)$ to the dataset features, where w represents the optimized weights of the neural network. The output z_j^L of the j-th neuron in the L-th layer is calculated as:

$$z_j^L = f\left(\sum_{i=1}^{n} w_{ij}^L x_i + b_j^L\right) \tag{6.1}$$

The notation w_{ji}^L represents the weight of the connection between the computing neuron and its i-th input in the preceding layer, and b_j^L is a bias parameter. The hidden layer activation function uses $\sigma(\cdot)$. On the other hand, the discriminators classify the data into classes using a fully-connected MLP network.

The generator model is trained to generate new data that resemble the minority class in each dataset, while the discriminators work to classify whether the data coming from the dataset is real or synthesized by the generator. The resultant loss is computed against both real and generated data, and the loss only serves to update the parameters of the generator. None of the parameters are adjusted other than those of the generator.

The cross-entropy loss function measures the difference between the real and generated data, as expressed in the following equation:

$$\text{Loss}_{CE} = -\frac{1}{N}\sum_{n=1}^{N}\left[y_i \cdot \log(p(y_i)) + (1-y_i)\cdot \log(1-p(y_i))\right] \tag{6.2}$$

where y_i is the real data, $p(y_i)$ is the predicted data calculated by the sigmoid activation function, and N is the number of observations in the batch.

To ensure transparency and enable a fair comparison, this chapter utilizes the same architecture that was proposed previously in [5]. The model is trained for 1000 epochs with a batch size of 128. The Adam optimizer is used, with a learning rate of 0.0001.
The TDCGAN model allows the generator to train until it produces synthetic data samples that closely match the distribution of the original dataset.

These models were selected based on their established performance in similar tasks and are being compared with our proposed model, TDCGAN, which introduces a novel architecture specifically designed for data balancing and intrusion detection tasks.

6.3 Experiments Setup

The workflow has been simplified for the working method of this work, and it is shown in Figure 6.1 below. This study flow chart shows an iterative study that has put four different data sets through the application of the TDCGAN model with the intent of examining the outcome in performance of classification. Here is the detailed explanation of the process.

- **Data Iteration:** The whole model consists of four iterations, during which different datasets are analyzed. Each of the datasets is sampled in that each iteration goes through a serious pre-processing to make the data consistent.
- **Data Pre-processing:**This step includes a series of pre-processing operations such as: Error Elimination and Gap Filling: The selected dataset undergoes pre-processing to eliminate errors, fill gaps, remove outliers, and discard irrelevant data types. Simple linear interpolation is employed to fill in any missing values.
- **Feature Encoding:** The dataset is processed to machine learning algorithms by encoding the categorical features via one-hot encoding.

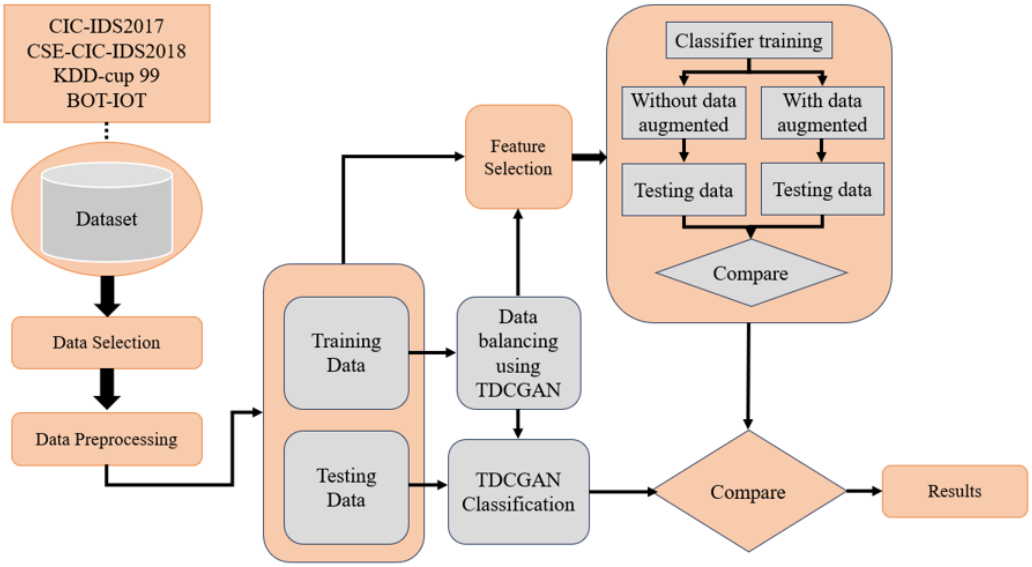


Fig. 6.1 The flowchart of the Experimental setup. (Source: Author own creation)

- **Scaling:** The dataset is scaled using the MinMaxScaler from Scikit-learn, normalizing the values to the range [0,1].
- **Feature Selection (PCA):** PCA is applied to the dataset. PCA reduces the dimensionality of the data while retaining the most significant features, thereby enhancing computational efficiency and potentially improving model performance.
- **Data Splitting:** The dataset is split into two parts: 70% of the data are allocated to the training set, while the remaining 30% is used as the testing set. We divided the dataset into 70% for training and 30% for testing to ensure the reliability of our model's evaluation.

  The training would be held on most of the data while the test would be done on a separate unseen data. This allows us to avoid overfitting and ensure that the performance evaluation of the model is done on data not seen before, which provides an accurate measure of its ability to generalize. A 70% and 30% division is a common practice in machine learning, ensuring that model or technique comparisons can take place in a very robust, unbiased manner.
- **TDCGAN Model Application:** The TDCGAN model has utilized training data from a particular dataset to learn how synthetic samples are generated. These samples form a better-balanced training set. Classifier training and evaluation: To begin with, the classifier will be trained using only non-TDCGAN samples, and tested with original

training data. The classifier will be finally tested on the same dataset after being trained with TDCGAN-generated samples.

- **Performance Comparison:** The impact of TDCGAN generation data augmentation on the respective dataset classification performances is then assessed through a comparison of the performance metrics from both scenarios; that is, training with TDCGAN-generated samples and without these samples.

- **Final Evaluation:** This last step, thus, involves the comparison of evaluation measures among all classifiers and all datasets to determine the best classifier and measure the overall success of the TDCGAN model in improving classification performance.

The pseudo code for the overall proposed approach is offered in Figure 6.2.

```
1. datasets = [dataset1, dataset2, dataset3, dataset4]
2. for dataset in datasets:
    a. dataset = eliminate_errors_and_fill_gaps(dataset)
    b. dataset = one_hot_encode(dataset)
    c. dataset = min_max_scale(dataset)
    d. dataset = apply_PCA (dataset)
    e. training_set, testing_set = split_data (dataset, train_size=0.7)
    f. tdgan_model = train_TDCGAN (training_set)
    g. synthetic_samples = tdgan_model. generate_samples ()
    h. augmented_training_set = training_set + synthetic_samples
    i. classifier = train_classifier(training_set)
    j. performance_without_TDCGAN = evaluate_classifier (classifier, testing_set)
    k. classifier = train_classifier(augmented_training_set)
    l. performance_with_TDCGAN = evaluate_classifier (classifier, testing_set)
    m. compare performance (performance_without_TDCGAN,
       performance_with_TDCGAN)
    n. store results (performance_without_TDCGAN, performance_with_TDCGAN)
3. end for
4. final_comparison = compare_across_datasets ()
5. identify_best_classifier(final_comparison)
6. evaluate_TDCGAN_effectiveness(final_comparison)
```

Fig. 6.2 The pseudo code of the proposed approach.(Source: Author own creation)

In the next subsection, we describe the sample selection of the data and the process of preparing the data. The feature selection for each dataset and the detailed setup of the experiment will now be discussed in the following sections.

6.3.1 Data Selection and Preparation

In this section, we experiment on four different data sets. The first step for any of the experiments is to extract the sample from every dataset shown above. DL algorithms consume a lot of hardware resources like CPU, Memory, and GPU for data processing and training. Hence to keep the selected sample small, it includes all categories such as normal and anomalous traffic from each data set.

To ensure that the dataset samples are uniformly distributed across attack and non-attack instances, the authors repeat this selection for each dataset ten times.

The subset selection process including all attack types was executed with adequate measures to counter imbalanced distributions and biases. These measures include:

- **Stratified sampling:**The subset selection utilized stratified sampling techniques to preserve a proportional representation of each attack type, ensuring a balanced distribution of attacks within the subset.
- **Class balancing:** Additional measures were implemented to balance the representation of different attack types in the subset. These actions might include oversampling the minority classes or undersampling the majority classes to address issues related to imbalanced distributions.
- **Randomization:**Randomization techniques were used during the subset selection process to reduce potential biases. This approach ensured that the selection was not influenced by any specific order or predetermined biases.

After every successive sample selection round, we plot all the data distributions in each sample and compare them for equality. This will prove that the random selection included all possible observations existing within the dataset. The next step is that each sample passes a series of pre-processing steps where it is harmonized, error elimination, gap filling, outliers' removal, and irrelevant data types. We filled all the gaps present in the dataset by simple linear interpolation. Some dataset features require encoding as a preparation for machine learning algorithm processes. One-hot encoding is employed for this purpose. Subsequently, the dataset undergoes scaling using the MinMaxScaler from the Scikit-learn library, which scales the values to the interval [0,1].

6.3.2 Feature Selection

PCA is undergoing its functionality on feature selection in a very robust manner; PCA transforms the original variables into another set of new uncorrelated variables known as principal components. On one hand, PCA reduces the dimension of the data and manages to retain the maximum variance, which results in reduced computational load and enhanced performance of the model [188]. PCA leads us to the most significant patterns and structures in a dataset and supports improved analysis and modeling in succeeding stages through a chosen and optimized suite of features. The top numerical features of the datasets about their PCA scores are provided in the Figures 6.3 - 6.6.

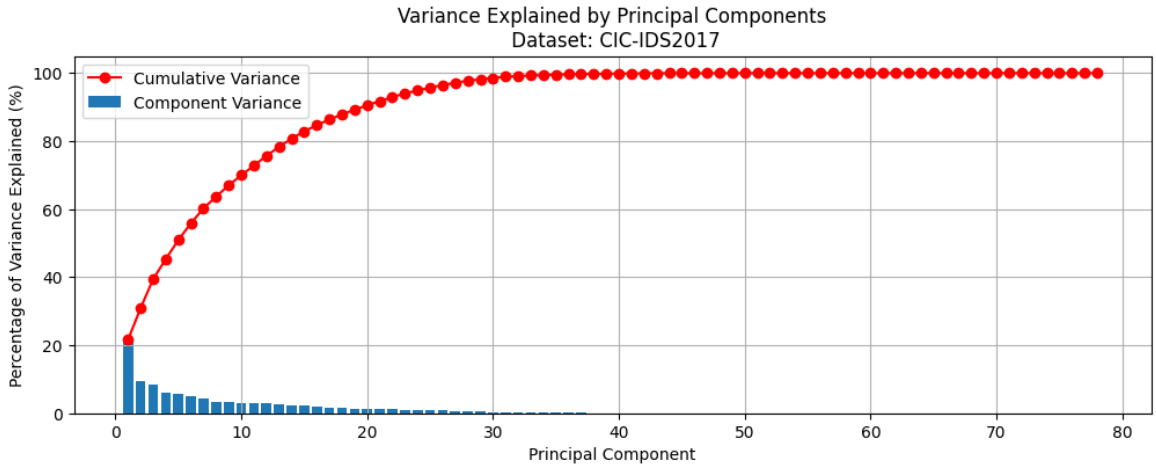


Fig. 6.3 Principal Component Analysis of the CIC-IDS2017 Dataset.(Source: Author own creation)

The figure 6.3 illustrates the PCA applied to a subset of the CIC-IDS2017 dataset, which originally contains 78 features. After retaining over 97% of the variance, the data is reduced to 30 principal components.

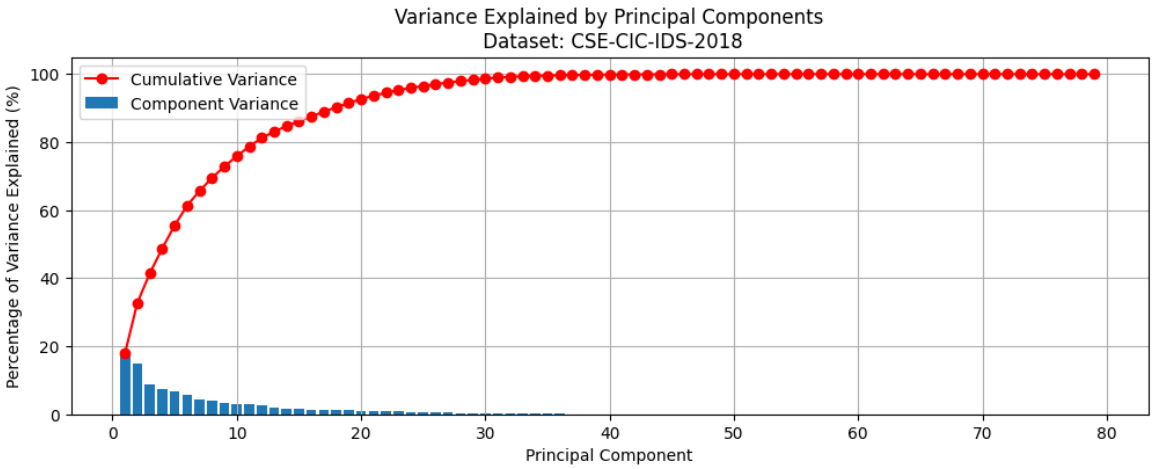


Fig. 6.4 Principal Component Analysis of the CSE-CIC-IDS-2018 Dataset.(Source: Author own creation)

The figure 6.4 illustrates the PCA applied to a subset of the CSE-CIC-IDS-2018 dataset, which originally contains 79 features. After retaining over 97% of the variance, the data is reduced to 30 principal components.

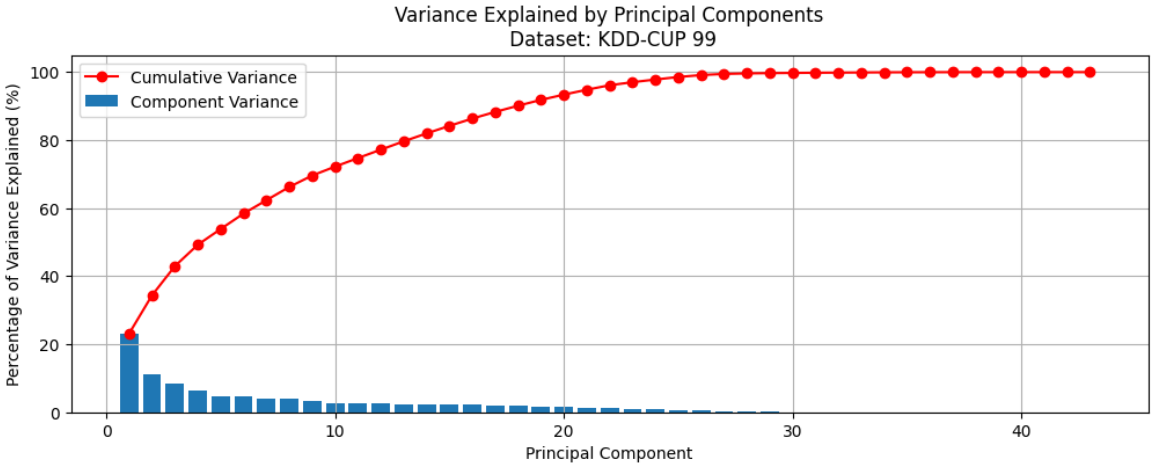


Fig. 6.5 Principal Component Analysis of the KDD-CUP 99 Dataset.(Source: Author own creation)

The figure 6.5 illustrates the PCA applied to a subset of the KDD-CUP 99 dataset, which originally contains 43 features. After retaining over 97% of the variance, the data is reduced to 26 principal components.

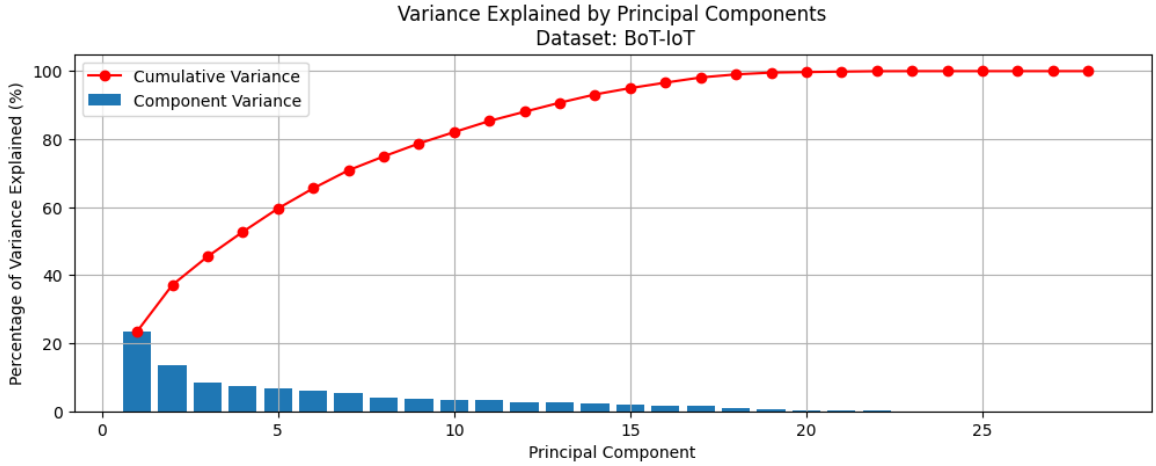


Fig. 6.6 Principal Component Analysis of the BoT-IoT Dataset.(Source: Author own creation)

The figure 6.6 illustrates the PCA applied to a subset of the BoT-IoT dataset, which originally contains 28 features. After retaining over 97% of the variance, the data is reduced to 18 principal components.

Table 6.1 Attack Label Distribution Across Datasets

Dataset	Attack Type	Count
CIC-IDS2017	Benign	2,273,097
	DOS Hulk	231,073
	PortScan	158,930
	DDoS	128,027
	DOS GoldenEye	10,293
	FTP-Patator	7,938
	SSH-Patator	5,879
	DoS Slowloris	5,796
	DoS Slowhttptest	5,499
	Bot	1,966
	Web Attack Brute Force	1,507
	Web Attack XSS	652
	Infiltration	367
	Web Attack SQL Injection	21
	Heartbleed	11
CSE-CIC-IDS2018	Benign	1,222,612
	DDOS attack-HOIC	137,157
	DOS attack-Hulk	92,325
	Bot	57,067
	FTP-Bruteforce	38,881
	SSH-Bruteforce	37,403
	Infiltration	32,470
	DOS SlowHTTPTest	27,902
	DOS GoldenEye	8,284
	DDOS attack-LOIC-UDP	343
	Brute Force-Web	113
	Brute Force-XSS	49
	SQL Injection	16
KDD-CUP 99	Smurf	2,807,886
	Neptune	1,072,017
	Normal	972,780

Continued on next page

Table 6.1 – Continued from previous page

Dataset	Attack Type	Count
	Satan	15,892
	Ipsweep	12,481
	Portsweep	10,413
	Nmap	2,316
	Back	2,203
	Warezclient	1,020
	Teardrop	979
	Pod	264
	Guess_passwd	53
	Buffer_overflow	30
	Land	21
	Warezmaster	20
	Imap	12
	Rootkit	10
	Loadmodule	9
	Ftp_write	8
	Multihop	7
	Phf	4
	Perl	3
	Spy	2
BOT-IoT	UDP	3,170,060
	TCP	2,549,206
	Service_Scan	117,170
	OS_Fingerprint	28,479
	HTTP	3,902
	Normal	707
	Keylogging	100
	Data Exfiltration	11

The table 6.1 presents the class distribution for each dataset in terms of attack types. The first column lists the dataset names, while the second column details the distribution of attack classes, specifying each attack type individually. The third column provides the count of instances for each attack type.

To enhance the reliability of our comparison, we divided each sample of the dataset into two parts: the first 70% served as the training set, while the remaining 30% constituted the testing set.

6.3.3 Experimental Setup

The distribution of classes among the said datasets is shown in Table 1. For the experiment, first of all, the TDCGAN trains itself on the training data of each of these datasets to produce samples and improve them. Thereafter, this output becomes available for the data balancing evaluation problem. Generated samples are visually inspected to ensure that they are similar enough to real data and capture important aspects of the problem scenario. The classification performance of the TDCGAN is then evaluated using all test data of every dataset. Following this, each classifier is evaluated and compared by running each method at least twice on each dataset:

1. **Training without TDCGAN-generated samples:** Train the classifier using only the original training data, without adding any TDCGAN-generated samples, then evaluate its performance on the testing data.
2. **Training with TDCGAN-generated samples:** Train the classifier using the original training data augmented with TDCGAN-generated samples, then evaluate its performance on the same testing data set. The performance metrics (accuracy, precision, recall, and F1-score) for each classifier can be compared here to analyze the effects of data augmentation with TDCGAN-generated samples on the classification performance.

The hypothesis states that if a classifier trained with TDCGAN samples does better than a classifier trained without the TDCGAN methodology, then it could be inferred that the TDCGAN samples were tally improving the generalization of the classifier to data never seen before or improving the classifier's ability to get the underlying data distribution. By contrast, if little or no improvement occurs or results in degradation of performance, these observations may indicate that the TDCGAN-generated samples are not relevant for the task under consideration or that they add some noise or bias to the classifier's behavior.

The last operation consists of comparing the evaluation metrics of each classifier with respect to the TDCGAN model for the classification task applied to the testing data of each dataset. This comparison lets you assess the classification performance of each model and then identify the best-performing model.

6.4 Results and Discussion

6.4.1 Experimental Results

To assess data balancing for each dataset after using TDCGAN model, we compared the class distribution before and after data balancing; the results are shown in Table 6.2

Table 6.2 Compact visualization of class distribution before and after data balancing using the TDCGAN model. (Source: Author's own creation)

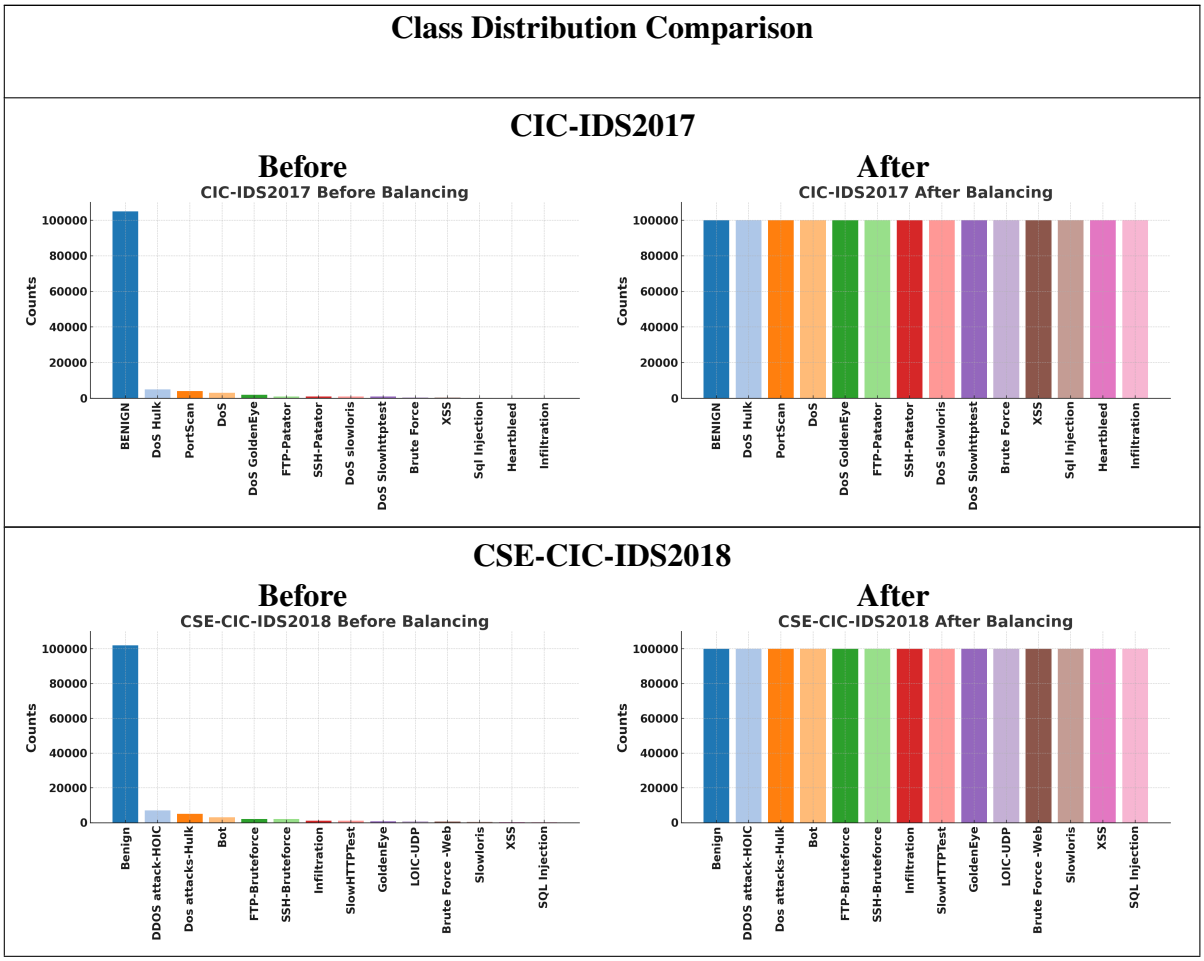
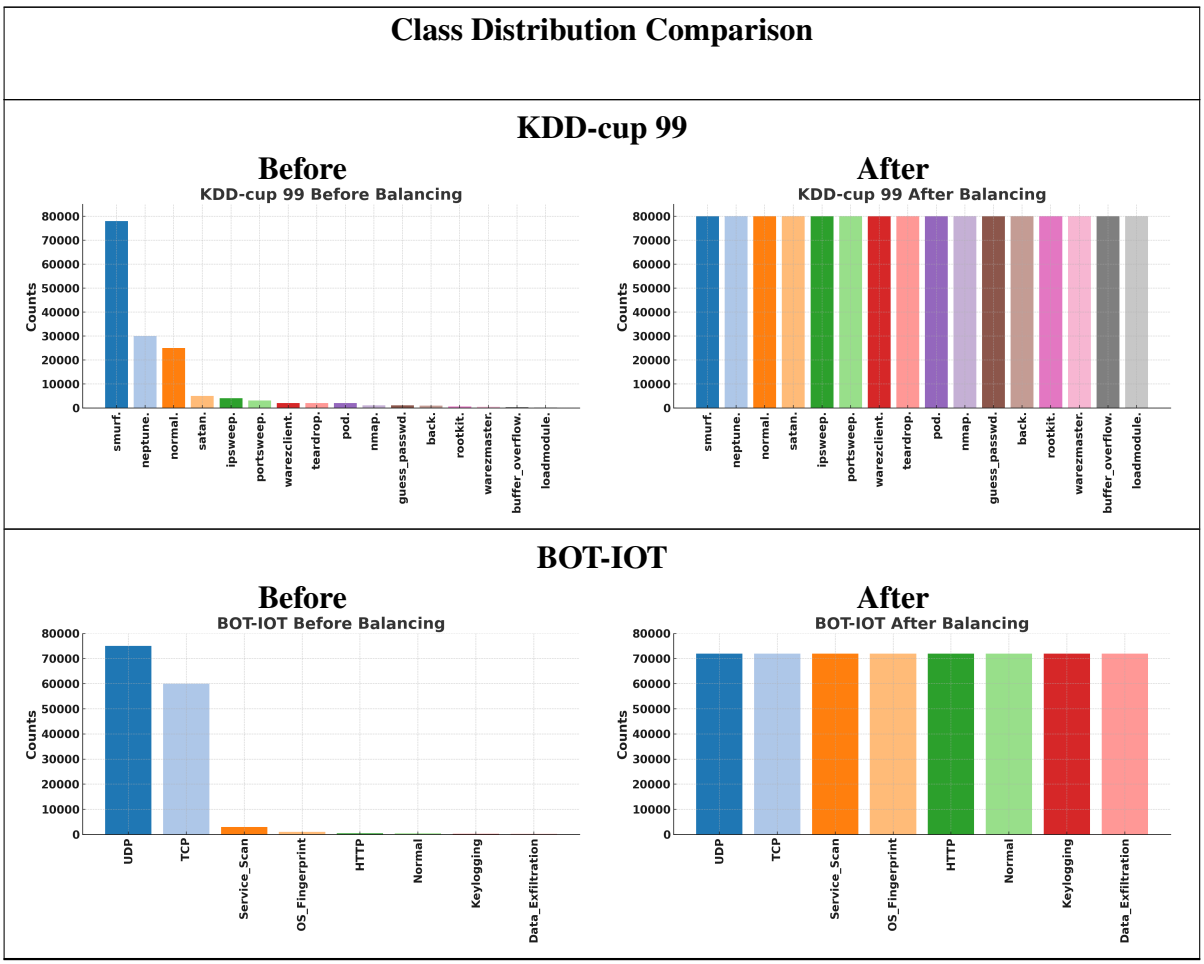


Table 6.2 continued on next page

Table 6.2 (continued)



Then, TDCGAN is trained with the testing data for every dataset to put to the test its efficacy in classification issues. Most importantly, this determines the model's efficacy in categorizing data instances, which is computed after the balancing procedure. After applying the TDCGAN model, we perform very intensive evaluations to assess its accuracy, precision, recall, and F1 score metrics. These results afford meaningful insights into how well the model handles test data, clearly understood as to its classification efficiencies in practical applications. The results are presented in Table 6.3.

In our evaluation of the effectiveness of data balancing techniques applied to TDCGAN across four individual datasets, we have followed an experimental design that provides thorough assessments of both the training data pre- and post-data balancing process using each machine learning classifier. This easily clarifies the overall impact that any data balancing activity might have on classifiers' output. Next, we will use testing data to immediately compare classifier performance before and after data-balancing. Hence, we will assess the

efficacy of TDCGAN in terms of improving classification accuracy and robustness with different classifiers over multiple datasets. The results are displayed in the tables below.

Table 6.3 The performance metrics for the TDCGAN model on testing data after data balancing

Dataset	Accuracy	Precision	Recall	F1-Score
CIC-IDS2017	0.96	0.97	0.99	0.98
CSE-CIC-IDS2018	0.95	0.95	0.97	0.98
KDD-CUP 99	0.95	0.96	0.95	0.96
BOT-IOT	0.96	0.95	0.95	0.96

Table 6.4 The performance metrics for decision tree before data balancing.

Dataset	Accuracy	Precision	Recall	F1-Score
CIC-IDS2017	0.70	0.92	0.86	0.89
CSE-CIC-IDS2018	0.69	0.89	0.82	0.86
KDD-CUP 99	0.75	0.85	0.63	0.73
BOT-IOT	0.72	0.83	0.79	0.80

Table 6.5 The performance metrics for decision tree after data balancing.

Dataset	Accuracy	Precision	Recall	F1-Score
CIC-IDS2017	0.85	0.92	0.90	0.90
CSE-CIC-IDS2018	0.77	0.91	0.89	0.86
KDD-CUP 99	0.88	0.89	0.78	0.82
BOT-IOT	0.88	0.92	0.91	0.90

Table 6.6 The performance metrics for random forest before data balancing.

Dataset	Accuracy	Precision	Recall	F1-Score
CIC-IDS2017	0.68	0.83	0.77	0.72
CSE-CIC-IDS2018	0.61	0.79	0.78	0.80
KDD-CUP 99	0.70	0.82	0.75	0.82
BOT-IOT	0.81	0.87	0.88	0.90

Table 6.7 The performance metrics for random forest after data balancing.

Dataset	Accuracy	Precision	Recall	F1-Score
CIC-IDS2017	0.78	0.80	0.83	0.79
CSE-CIC-IDS2018	0.82	0.84	0.86	0.87
KDD-CUP 99	0.88	0.90	0.89	0.89
BOT-IOT	0.90	0.89	0.91	0.91

Table 6.8 The performance metrics for MLP before data balancing.

Dataset	Accuracy	Precision	Recall	F1-Score
CIC-IDS2017	0.82	0.84	0.85	0.87
CSE-CIC-IDS2018	0.84	0.88	0.82	0.84
KDD-CUP 99	0.83	0.82	0.84	0.82
BOT-IOT	0.85	0.83	0.81	0.82

Table 6.9 The performance metrics for MLP after data balancing.

Dataset	Accuracy	Precision	Recall	F1-Score
CIC-IDS2017	0.92	0.90	0.93	0.91
CSE-CIC-IDS2018	0.93	0.92	0.94	0.91
KDD-CUP 99	0.92	0.91	0.92	0.92
BOT-IOT	0.92	0.93	0.94	0.94

Table 6.10 The performance metrics for Naive Bayes before data balancing.

Dataset	Accuracy	Precision	Recall	F1-Score
CIC-IDS2017	0.60	0.71	0.72	0.69
CSE-CIC-IDS2018	0.65	0.67	0.70	0.68
KDD-CUP 99	0.70	0.75	0.80	0.82
BOT-IOT	0.72	0.74	0.78	0.77

Table 6.11 The performance metrics for Naive Bayes after data balancing.

Dataset	Accuracy	Precision	Recall	F1-Score
CIC-IDS2017	0.80	0.85	0.82	0.81
CSE-CIC-IDS2018	0.73	0.81	0.73	0.76
KDD-CUP 99	0.85	0.87	0.85	0.87
BOT-IOT	0.80	0.82	0.81	0.83

To assess the performance of the TDCGAN model relative to other machine learning models in terms of classification accuracy across four datasets, a comparative analysis of these models is conducted. The results are shown in Figures 6.7 - 6.10

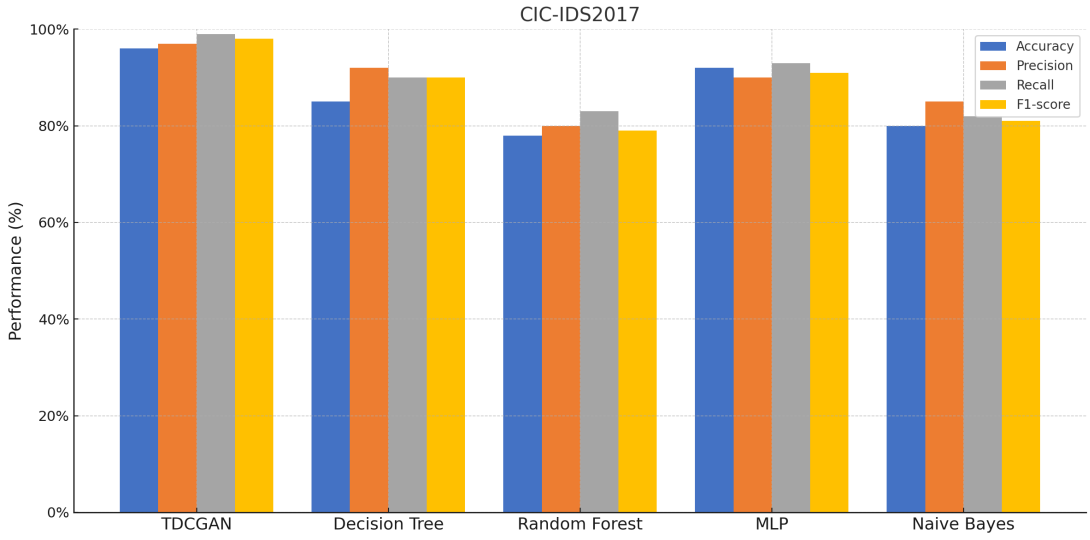


Fig. 6.7 CIC-IDS2017.(Source: Author own creation)

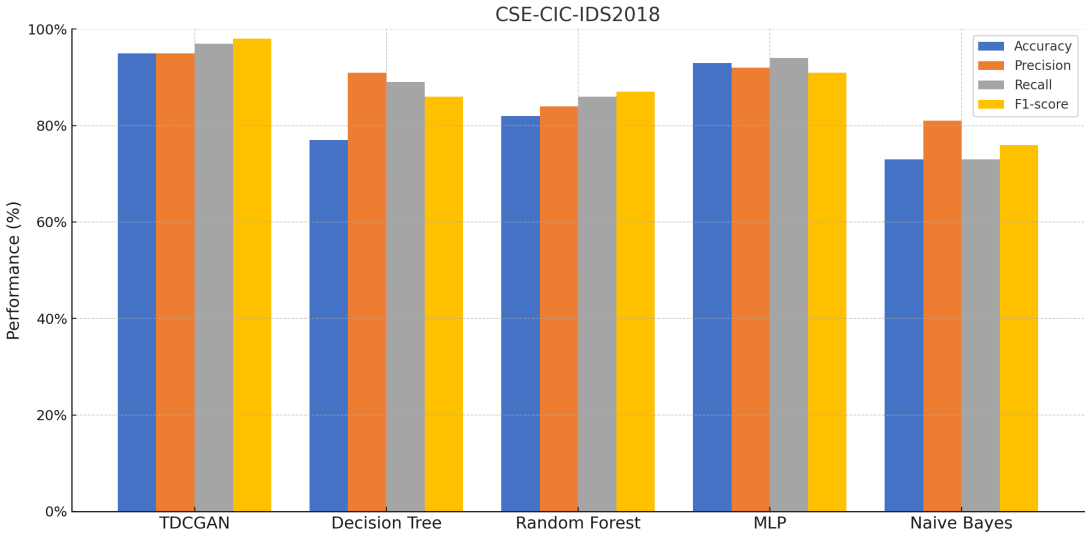


Fig. 6.8 CSE-CIC-IDS2018.(Source: Author own creation)

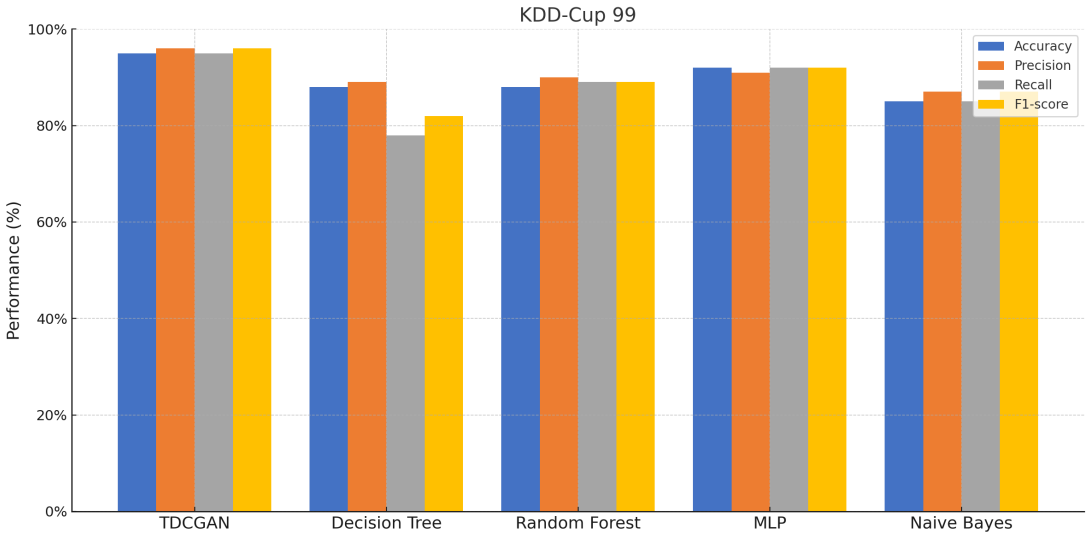


Fig. 6.9 KDD-cup 99.(Source: Author own creation)

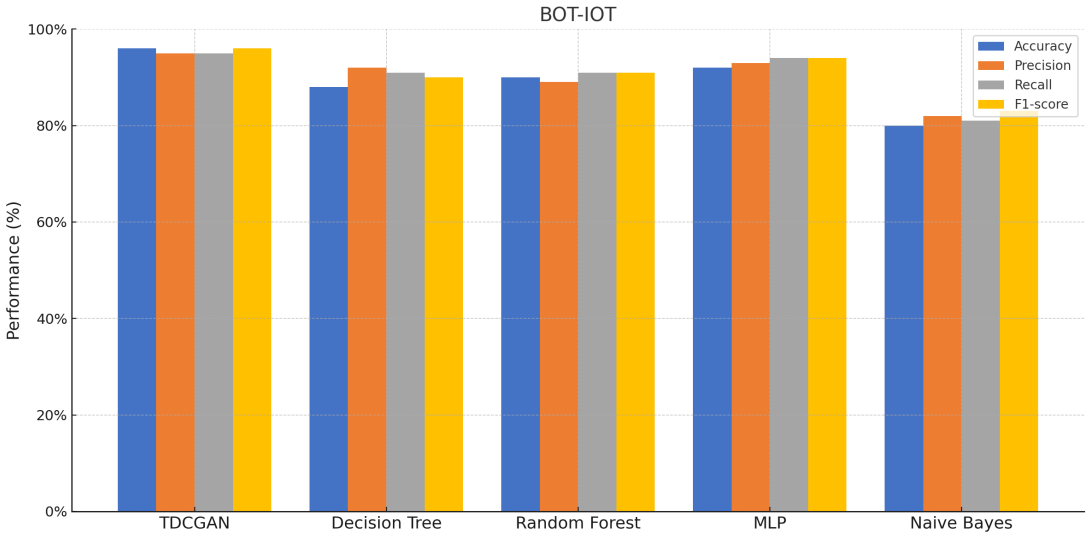


Fig. 6.10 BOT-IOT.(Source: Author own creation)

6.4.2 Discussion

Before the data was balanced, the decision tree model gave a good prediction performance on the datasets, with the accuracy ranging from 0.69 to 0.75. However, there were clear differences in performance metrics like recall and F1, especially regarding the KDD-cup 99 data set. The recall here was lower, at 0.63, implying that the model had some difficulties in correctly classifying a sample into the positive class. After data balancing, there is a definite enhancement in the performance of the DT model across all datasets. Generally, the accuracy has increased, and such improvements occurred regarding precision, recall, and F1 measures. For instance, the recall in the CICIDS2017 dataset improved from 0.86 to 0.90, indicating that this model has become better at correctly identifying positives. Similar improvements are noted in all other datasets. The RF also showed varied performances across the datasets. This model achieved relatively high accuracies; however, considerable differences exist in terms of datasets regarding precision, recall, and F1 scores. As an example, in the CSE-CIC-IDS2018 data set, the precision value was 0.79, meaning that only 79% of positive class instances were correctly identified by the model, which may render the model undesirable based on the application. After data balancing, there is a gain on the RF model with respect to performance improvement in each dataset. Notably, accuracy has improved greatly, and there have been substantial increases in precision, recall, and F1-score as well. For example, recall improved from 0.78 to 0.83 in the CICIDS2017 dataset, which shows a considerable improvement of positive class instance identification.

Concisely, DT model performance became good before data balancing, whether with datasets having an accuracy of 0.69 to 0.75. A sound performance in terms of metrics like recall and F1 score was observed. This can be seen clearly from instances like the one for KDD-cup 99, where the accuracy measure was relatively lower at a recall of 0.63, meaning that the machine has a little defect in indicating data belonging to the positive class. After this balancing of data, the model has exhibited a marked improvement in performance for all datasets in DT learning. Accuracy has generally increased, while metrics such as precision, recall, and F1 have been improved significantly. For example, the recall in the CICIDS2017 dataset rose from 0.86 to 0.90, which implies the model became better in positive class identification. Nearly identical improvements can be seen in all other datasets. The RF also gave different performances over the datasets. While it provided relatively high accuracies, the difference in precision, recall, and F1 score among the different datasets was substantial. For instance, the precision in the CSE-CIC-IDS2018 dataset was 0.79, denoting the fact that only 79% of the positive class instances were identified correctly by the model, which might not be very desirable based on its application. After performing data balancing, there was a huge gain in the performance of the RF model on all datasets. There was a very substantial increment in accuracies and significant increases in precision, recall, and F1-score too. For instance, in the CICIDS2017 dataset, recall improved from 0.78 to 0.83, indicating quite an improvement toward correct identification of positive class instances.

The MLP model showed a very good performance on every data set. The accuracies varied from 0.82 to 0.85. The model possessed good precision, recall, and F1-score values, meaning it was able to classify instances from both classes accurately. For example, the precision was 0.84 in the case of the CIC-IDS2017 dataset, meaning that 84% of the instances classified as positive instances were true positives These metrics were equally high for recall and F1-score, suggesting that the model excels in identifying most of the positive cases while minimising its false negative rate. Even at its best, data balancing significantly improved the MLP's effectiveness. After balancing the data, a notable improvement in accuracy, precision, recall, and F1 score was noticed across the datasets. For example, in the CICIDS2017 dataset, accuracy improved from 0.82 to 0.92, a marked increase in the ability of the model to accurately categorise examples from both classes. Apart from a clear improvement in accuracy, precision, recall, and F1-score also improved, thus ensuring a more balanced and efficient classification, especially for the minority class.

The MLP model showed very good performance on each of the datasets. Accuracies ranged between 0.82 and 0.85. The model possessed good precision, recall, and F1-score

values, indicating that it was able to classify instances from both classes correctly. For example, the model achieved a 0.84 precision in the case of the CIC-IDS2017 dataset: 84% of instances that were classified as positive instances were true positives. These values were also high for recall and F1-score, suggesting that it does fairly well in identifying most positive cases while keeping its false negative rate low. Data balancing further improved the effectiveness of the MLP model even at its present superior strength. Following data balancing, the noticeable improvement in accuracies, precision, recalls, and F1 scores was heard across all datasets. For instance, the accuracy on the CICIDS2017 dataset improved from 0.82 to 0.92, indicating a marked increase in the ability of the model to correctly classify instances from both classes. Similarly, precision, recall, and F1 score also improved and indicated a more balanced and accurate classification, especially for the minority class.

The NB model performed moderately on all the datasets without balancing the data, with accuracies ranging between 0.60 and 0.72. While the model has almost all good recall values, meaning its ability to identify positive classes has been better, its precision values and F1 scores are relatively low, indicating high false positives and a less optimal balance of precision and recall.

And after data balancing, the impact on the performance of the NB model is evident in all datasets. The accuracies improved overall, and measures such as precision, recall, and F1-score experienced considerable improvements too. In the case of the CIC-IDS2017, the accuracy had risen from 0.60 to 0.80, evidencing a considerable improvement regarding overall performance in the classification of the model. Precision, recall, and F1-score also improved, indicating a more balanced and accurate classification across both classes.

All the learning machine models, like DT, RF, MLP, and NB, significantly improved due to the data balancing. This points to the huge contribution of data balancing by the TDC-GAN model in enhancing classification accuracy and metrics precision, recall, and f1-score. Indeed, TDCGAN generated synthetic instances of the minority class while respecting the original distribution of data and making class representation more exhaustive in learning the machine against fairer and accurate predictions. Aggregation for class representation helps in producing a balanced dataset, hence a more balanced learning environment. All the improvements across all the models and datasets make a stronger case for the performance of the TDCGAN method in transforming the nature of an imbalanced data set problem into a more meaningful one for machine learning classifiers.

In summary of these results, all the models perform better on the other three datasets than on the BOT-IOT dataset. Overall, TDCGAN produced slightly better results than all the other models, while the NB model appeared to underperform compared to the other models tested.

The NB model states that all the features in the dataset are conditionally independent. In other words, it is supposed to find complex relationships between features that probably will clash with this notion and thus will decrease the performance for the NB model. For instance, in specific network intrusion detection tasks, interdependence can be seen between packet size, protocol type, and connection duration; this correlation is significant to distinguish normal traffic from malicious traffic. These properties may be responsible for the low performance of the NB model on the BOT-IOT dataset in capturing interdependencies in the dataset features. Although these might not be captured in the output class, TDCGAN, which generates synthetic samples while handling a severe class imbalance, might also generate these and contribute to overall effectivity as it incorporates NB or any other model.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

The imbalanced distribution of attacks in historical network traffic presents a significant challenge for intrusion detection systems (IDSs) based on traditional machine learning methods. These methods often struggle to effectively address the issue of imbalanced learning. In response, this thesis introduces a novel technique called TDCGAN, a technology based on generative adversarial networks (GANs), specifically designed to tackle the problem of imbalanced datasets in IDS. The proposed TDCGAN model consists of a generator and three discriminators, all implemented using multi-layer perceptron (MLP) networks. This architecture allows the generator to generate synthetic data closely resembling real network traffic, while the discriminators aim to differentiate between genuine and attack traffic. To further enhance the TDCGAN framework, an additional layer is added at the end of the network to select the optimal outcome from the outputs produced by the three discriminators, enhancing the overall performance of the model. To evaluate the effectiveness of the proposed approach, the UGR'16 dataset, widely used in IDS research, is utilized for testing and evaluation purposes. A subset of the dataset is extracted and divided into training and testing sets. The experimental results showcase the outstanding performance of the proposed TDCGAN model across various evaluation metrics, including accuracy, precision, F1 score, and recall. Additionally, a comparison is made with other oversampling machine learning techniques, highlighting the superiority of the proposed method. While balancing datasets can be beneficial, it is important to note that it might not always be necessary or feasible, especially in cases where the class imbalance reflects the real-world distribution. In many real-world applications, the distribution of classes is often imbalanced. For instance, fraud detection, disease diagnosis, or rare event prediction typically involve imbalanced datasets. By balancing the dataset during training, the model learns to handle these imbalances and

becomes more effective in addressing real-world scenarios. Additionally, balancing the dataset should be performed carefully to avoid introducing artificial patterns or losing valuable information from the original data.

In order to assess the efficacy of integrating the TDCGAN model for data balancing task and to demonstrate the impact of data balancing on enhancing classification accuracy and improving machine learning performance, we conducted a comparative analysis of four machine learning models: DT, RF, MLP, and NB, across various datasets. Utilizing diverse benchmark IDS datasets, through our work we implemented the TDCGAN model to address class imbalance. Subsequently, the machine learning models were utilized to classify the data both before and after applying data balancing techniques, aiming to assess the resulting effects on classification performance.

Before data balancing, the performance of various machine learning models, including DT, RF, MLP, and NB, exhibited inconsistencies across datasets. Notably, metrics such as recall and F1-score displayed discrepancies, indicating challenges in accurately identifying positive class instances. However, after employing the TDCGAN model for data balancing, substantial improvements were observed across all models and datasets. The augmentation of the dataset with synthetic instances generated by TDCGAN led to notable enhancements in accuracy, precision, recall, and F1-score. For instance, in the CIC-IDS2017 dataset, recall surged from 0.86 to 0.90 for the DT model, showcasing a significant improvement in positive class identification. Similar enhancements were observed across other models, highlighting the effectiveness of TDCGAN in mitigating class imbalance and enhancing overall classification performance. The consistent improvements seen in the performance of machine learning models post-data balancing strongly indicate the efficacy of the TDCGAN approach. By generating synthetic instances of the minority class while preserving the underlying data distribution, TDCGAN effectively addressed class imbalance issues present in the datasets. This facilitated a more balanced representation of both classes, enabling the models to learn more effectively and make fairer and more accurate predictions. The substantial enhancements across metrics such as precision, recall, and F1-score underscore the significant role played by TDCGAN in refining the classification capabilities of the machine learning models. Thus, the thesis highlights the importance of employing advanced data balancing techniques like TDCGAN to improve the performance of machine learning models in scenarios with imbalanced datasets.

In summary, while the integration of TDCGAN has demonstrated significant improvements in addressing class imbalance and enhancing classification performance, it also introduces certain challenges. The addition of TDCGAN increases the complexity of the model development process, requiring careful tuning of additional hyperparameters, which demands extensive experimentation and expertise. Furthermore, the implementation of TDCGAN adds computational overhead, increasing the time and resources needed for generating synthetic instances and integrating them into the training data. These factors highlight the need for a balanced approach, where the benefits of using TDCGAN are carefully weighed against the added complexity and resource requirements

7.2 Publications derived from the thesis

Four papers originating from this thesis have been published: three in well-recognized, peer-reviewed journals and one presented at reputable conference, as outlined below:

- Jamoos, M.; Mora, A.M.; Alkhanafseh, M.; Surakhi, O. "A Comprehensive Survey for Machine Learning and Deep Learning Applications for Detecting Intrusion Detection." Proceedings of the 22nd International Arab Conference on Information Technology, Muscat, 2021, pp. 1–13.

- Surakhi, O.; Mora, A.M.; Jamoos, M.; Alkhanafseh, M. "The Intrusion Detection System by Deep Learning Methods: Issues and Challenges." The International Arab Journal of Information Technology 2022, 19(3A), 501–513. https://doi.org/10.34028/iajit/19/3a/10.

- Jamoos, M.; Mora, A.M.; AlKhanafseh, M.; Surakhi, O. A New Data-Balancing Approach Based on Generative Adversarial Network for Network Intrusion Detection System. Electronics 2023, 12, 2851.

- Jamoos, M.; Mora, A.M.; AlKhanafseh, M.; Surakhi, O. "A Comparative Analysis of the TDCGAN Model for Data Balancing and Intrusion Detection." Signals 2024, 5, 580–596. https://doi.org/10.3390/signals5030032.

In the first and second paper, the work presents a comprehensive survey on the use of Machine Learning (ML) and Deep Learning (DL) techniques in Intrusion Detection Systems (IDS) to enhance detection accuracy and minimize error rates. Recent research studies published between 2018 and 2021 on the application of ML and DL in IDS are thoroughly analyzed and summarized. The first one, published in the in 22nd International Arab Conference on Information Technology which held in Muscat-Oman. The second one was published

in The International Arab Journal of Information Technology, which holds a Q2 ranking in the global Scopus database. The journal's CiteScore has risen to 2.6.

The third paper, which serves as the cornerstone of this work, introduces our model-based generative adversarial network (GAN) named TDCGAN. This model is designed to enhance the detection rate of the minority class in imbalanced datasets while ensuring efficiency. The paper was published in the Electronics journal by MDPI, which held a Q2 ranking in JCR and had an impact factor of 2.6 at the time of publication.

The final paper was published in Signals, an MDPI journal with a Q2 ranking and indexed in Scopus. This study evaluates the performance of the TDCGAN model by applying it to balance data across four benchmark IDS datasets: CIC-IDS2017, CSE-CIC-IDS2018, KDD Cup 99, and BOT-IOT. Subsequently, four machine learning algorithms were utilized to classify the balanced data. The results showed a significant improvement in classification accuracy for each classifier following the implementation of the TDCGAN model for data balancing.

7.3 Future Work

While this thesis demonstrates the effectiveness of the TDCGAN model in addressing class imbalance and improving classification performance in intrusion detection systems (IDS), there remains a significant opportunity to extend and refine this work. One of the most critical avenues for future research involves the practical integration of the TDCGAN framework into real-time IDS environments. Although the model has proven successful in balancing datasets and enhancing the performance of traditional machine learning classifiers in static settings, its application to dynamic and real-time scenarios poses unique challenges. These include computational overhead, latency, and the need to adapt to evolving attack patterns and network conditions. Future studies could explore methods to optimize the computational efficiency of the TDCGAN framework, such as developing lightweight architectures or leveraging hardware acceleration techniques. Additionally, a dynamic data balancing strategy, where synthetic data is generated and incorporated into the training process adaptively, could enhance the model's practicality and relevance in real-world applications. This approach would enable IDS systems to remain effective even as data distributions change over time.

Moreover, extending the use of TDCGAN to address complex attack scenarios, such as zero-day attacks or multi-stage intrusions, represents another promising direction. Such research could focus on generating synthetic data that captures nuanced attack patterns, thereby improving the ability of IDS systems to detect and respond to advanced threats. By prioritizing these key areas, future work can build upon the foundation laid by this thesis, pushing the boundaries of what is achievable in intrusion detection and further enhancing the security of network systems.

References

[1] Aleksandar Lazarevic, Vipin Kumar, and Jaideep Srivastava. *Intrusion Detection: A Survey*, volume 5, pages 19–78. 01 2005.

[2] Mayank Swarnkar and Shyam Singh Rajput. *Artificial Intelligence for Intrusion Detection Systems*. CRC Press, 2023.

[3] Hamed Alqahtani, Manolya Kavakli-Thorne, and Gulshan Kumar. Applications of generative adversarial networks (gans): An updated review. *Archives of Computational Methods in Engineering*, 28:525–552, 2021.

[4] Gabriel Maciá-Fernández, José A. Camacho, Roberto Magán-Carrión, Pedro García-Teodoro, and Roberto Therón. UGR'16: A New Dataset for the Evaluation of Cyclostationarity-Based Network IDSs. *Computers & Security*, 73:411–424, 2018.

[5] Mohammad Jamoos, Antonio M. Mora, Mohammad AlKhanafseh, and Ola Surakhi. A new data-balancing approach based on generative adversarial network for network intrusion detection system. *Electronics*, 12(13), 2023.

[6] Yang Li, Zhengming Li, and Mengyao Li. A comprehensive survey on intrusion detection algorithms. *Computers and Electrical Engineering*, 121:109863, 2025.

[7] Mohammad Jamoos, Antonio M. Mora, Mohammad AlKhanafseh, and Ola Surakhi. A comparative analysis of the tdcgan model for data balancing and intrusion detection. *Signals*, 5(3):580–596, 2024.

[8] P Saskia Bayerl, Ruža Karlović, Babak Akhgar, and Garik Markarian. *Community Policing-A European Perspective*. Springer, 2017.

[9] Jie Li, Yanpeng Qu, Fei Chao, Hubert PH Shum, Edmond SL Ho, and Longzhi Yang. Machine learning algorithms for network intrusion detection. *AI in Cybersecurity*, pages 151–179, 2019.

[10] J. Anderson. Computer security threat monitoring and surveillance. Technical report, James P. Anderson Company, 1980.

[11] Ola Surakhi, Antonio García, Mohammed Jamoos, and Mohammad Alkhanafseh. The intrusion detection system by deep learning methods: issues and challenges. 2022.

[12] Donald Michie, David J Spiegelhalter, Charles C Taylor, and John Campbell. *Machine learning, neural and statistical classification*. Ellis Horwood, 1995.

[13] Damien Dablain, Bartosz Krawczyk, and Nitesh V Chawla. Deepsmote: Fusing deep learning and smote for imbalanced data. *IEEE Transactions on Neural Networks and Learning Systems*, 34(9):6390–6404, 2022.

[14] Olusola O Abayomi-Alli, Robertas Damaševičius, Atika Qazi, Mariam Adedoyin-Olowe, and Sanjay Misra. Data augmentation and deep learning methods in sound classification: A systematic review. *Electronics*, 11(22):3795, 2022.

[15] Shahnawaz Ayoub, Yonis Gulzar, Jaloliddin Rustamov, Abdoh Jabbari, Faheem Ahmad Reegu, and Sherzod Turaev. Adversarial approaches to tackle imbalanced data in machine learning. *Sustainability*, 15(9):7097, 2023.

[16] Lijyun Huang, Kate Ching-Ju Lin, and Yu-Chee Tseng. Resolving intra-class imbalance for gan-based image augmentation. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pages 970–975. IEEE, 2019.

[17] Mohammad Y AlKhanafseh and Ola M Surakhi. Vanet intrusion investigation based forensics technology: A new framework. In *2022 International Conference on Emerging Trends in Computing and Engineering Applications (ETCEA)*, pages 1–7. IEEE, 2022.

[18] Bambang Susilo and Riri Fitri Sari. Intrusion detection in iot networks using deep learning algorithm. *Information*, 11(5):279, 2020.

[19] Yuan Zhou, Fang Dong, Yufei Liu, Zhaofu Li, JunFei Du, and Li Zhang. Forecasting emerging technologies using data augmentation and deep learning. *Scientometrics*, 123:1–29, 2020.

[20] Wei Xue and Jing Zhang. Dealing with imbalanced dataset: A re-sampling method based on the improved smote algorithm. *Communications in Statistics-Simulation and Computation*, 45(4):1160–1172, 2016.

[21] Sumaya Sanober, Izhar Alam, Sagar Pande, Farrukh Arslan, Kantilal Pitambar Rane, Bhupesh Kumar Singh, Aditya Khamparia, and Mohammad Shabaz. An enhanced secure deep learning algorithm for fraud detection in wireless communication. *Wireless Communications and Mobile Computing*, 2021(1):6079582, 2021.

[22] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[23] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pages 1322–1328. Ieee, 2008.

[24] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer, 2005.

[25] Mehdi Mirza. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[26] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[27] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[28] Charitos Charitou, Simo Dragicevic, and Artur d'Avila Garcez. Synthetic data generation for fraud detection using gans. *arXiv preprint arXiv:2109.12546*, 2021.

[29] Lawrence R Halme and R Kenneth Bauer. Aint misbehaving: A taxonomy of anti-intrusion techniques. In *Proceedings of the 18th national information systems security conference*, pages 163–172, 2000.

[30] Pedro Garcia-Teodoro, Jesus Diaz-Verdejo, Gabriel Maciá-Fernández, and Enrique Vázquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. *computers & security*, 28(1-2):18–28, 2009.

[31] Srinivas Mukkamala, Guadalupe Janoski, and Andrew Sung. Intrusion detection using neural networks and support vector machines. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*, volume 2, pages 1702–1707. IEEE, 2002.

[32] Dorothy E Denning. An intrusion-detection model. *IEEE Transactions on software engineering*, (2):222–232, 1987.

[33] Hervé Debar, Marc Dacier, and Andreas Wespi. Towards a taxonomy of intrusion-detection systems. *Computer Networks*, 31(8):805–822, 1999.

[34] Ahmed Ahmim, Makhlouf Derdour, and Mohamed Amine Ferrag. An intrusion detection system based on combining probability predictions of a tree of classifiers. *International Journal of Communication Systems*, 31(9):e3547, 2018.

[35] Theuns Verwoerd and Ray Hunt. Intrusion detection techniques and approaches. *Computer communications*, 25(15):1356–1365, 2002.

[36] Mueen Uddin, Azizah Abdul Rahman, Naeem Uddin, Jamshed Memon, Raed A Alsaqour, and Suhail Kazi. Signature-based multi-layer distributed intrusion detection system using mobile agents. *Int. J. Netw. Secur.*, 15(2):97–105, 2013.

[37] Wei Ma. Analysis of anomaly detection method for internet of things based on deep learning. *Transactions on Emerging Telecommunications Technologies*, 31(12):e3893, 2020.

[38] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[39] Vikash Kumar, Ditipriya Sinha, Ayan Kumar Das, Subhash Chandra Pandey, and Radha Tamal Goswami. An integrated rule based intrusion detection system: analysis on unsw-nb15 data set and the real time online dataset. *Cluster Computing*, 23:1397–1418, 2020.

[40] Peyman Kabiri and Ali A Ghorbani. Research on intrusion detection and response: A survey. *Int. J. Netw. Secur.*, 1(2):84–102, 2005.

[41] Enamul Kabir, Jiankun Hu, Hua Wang, and Guangping Zhuo. A novel statistical technique for intrusion detection systems. *Future Generation Computer Systems*, 79:303–318, 2018.

[42] Zeeshan Ahmad, Adnan Shahid Khan, Cheah Wai Shiang, Johari Abdullah, and Farhan Ahmad. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, 32(1):e4150, 2021.

[43] Hongyu Liu and Bo Lang. Machine learning and deep learning methods for intrusion detection systems: A survey. *applied sciences*, 9(20):4396, 2019.

[44] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Abubakar Malah Umar, Okafor Uchenwa Linus, Humaira Arshad, Abdullahi Aminu Kazaure, Usman Gana, and Muhammad Ubale Kiru. Comprehensive review of artificial neural network applications to pattern recognition. *IEEE access*, 7:158820–158846, 2019.

[45] L. L. Hsu, C. J. Lin, and L. Wei. A survey of support vector machine applications in the field of machine learning. *Computational Intelligence*, 2010.

[46] Youqiang Zhang, Guo Cao, Bisheng Wang, and Xuesong Li. A novel ensemble method for k-nearest neighbor. *Pattern Recognition*, 85:13–25, 2019.

[47] Hafza A Mahmood and Soukaena H Hashem. Network intrusion detection system (nids) in cloud environment based on hidden naïve bayes multiclass classifier. *Al-Mustansiriyah Journal of Science*, 28(2):134–142, 2018.

[48] Irina Rish et al. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46. Seattle, WA, USA;, 2001.

[49] Daniel Lowd and Pedro Domingos. Naive bayes models for probability estimation. In *Proceedings of the 22nd international conference on Machine learning*, pages 529–536, 2005.

[50] S. Chary and B. Rama. A survey on comparative analysis of decision tree algorithms in data mining. *International Journal of Advanced Scientific Technologies, Engineering and Management Sciences*, 3(1):91–95, 2017.

[51] Pooja Gulati, Amita Sharma, and Manish Gupta. Theoretical study of decision tree algorithms to identify pivotal factors for performance improvement: A review. *Int. J. Comput. Appl*, 141(14):19–25, 2016.

[52] Mrinal Pandey and Vivek Kumar Sharma. A decision tree algorithm pertaining to the student performance analysis and prediction. *International Journal of Computer Applications*, 61(13):1–5, 2013.

[53] Martha A Zaidan, Ola Surakhi, Pak Lun Fung, and Tareq Hussein. Sensitivity analysis for predicting sub-micron aerosol concentrations based on meteorological parameters. *Sensors*, 20(10):2876, 2020.

[54] Xinyue Wang, Zhicheng Cai, and Chenglei Peng. X-mlp: A patch embedding-free mlp architecture for vision. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2023.

[55] Marius-Constantin Popescu, Valentina E Balas, Liliana Perescu-Popescu, and Nikos Mastorakis. Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, 8(7):579–588, 2009.

[56] Ola M Surakhi, Martha Arbayani Zaidan, Sami Serhan, Imad Salah, and Tareq Hussein. An optimal stacked ensemble deep learning model for predicting time-series data using a genetic algorithm—an application for aerosol particle number concentrations. *Computers*, 9(4):89, 2020.

[57] O. Surakhi, S. Serhan, and I. Salah. On the ensemble of recurrent neural network for air pollution forecasting: Issues and challenges. *Advances in Science, Technology and Engineering Systems Journal*, 5(2):512–526, 2020.

[58] AW Salameh and OM Surakhi. An optimized convolutional neural network for handwritten digital recognition classification. *Journal of Theoretical and Applied Information Technology*, 98(21):3494–3503, 2020.

[59] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. Variational autoencoder for deep learning of images, labels and captions. *Advances in neural information processing systems*, 29, 2016.

[60] Nilesh Pandey and Andreas Savakis. Poly-gan: Multi-conditioned gan for fashion synthesis. *Neurocomputing*, 414:356–364, 2020.

[61] Firuz Kamalov, Ho-Hon Leung, and Aswani Kumar Cherukuri. Keep it simple: random oversampling for imbalanced data. In *2023 Advances in Science and Engineering Technology International Conferences (ASET)*, pages 1–4. IEEE, 2023.

[62] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29, 2004.

[63] R. Srinivasan and S. Rajasekaran. Analysis of machine learning algorithm: Smote with svm. *International Journal of Novel Research and Development*, 8(5):631–636, 2023. Accessed December 28, 2024.

[64] Elsie Fezeka Swana, Wesley Doorsamy, and Pitshou Bokoro. Tomek link and smote approaches for machine fault classification with an imbalanced dataset. *Sensors*, 22(9):3246, 2022.

[65] Mimi Mukherjee and Matloob Khushi. Smote-enc: A novel smote-based method to generate synthetic data for nominal and continuous features. *Applied system innovation*, 4(1):18, 2021.

[66] M Durgadevi et al. Generative adversarial network (gan): A general review on different variants of gan and applications. In *2021 6th International Conference on Communication and Electronics Systems (ICCES)*, pages 1–8. IEEE, 2021.

[67] Emilija Strelcenia. *A New Generative Adversarial Network for Improving Classification Performance for Imbalanced Data*. PhD thesis, Bournemouth University, 2024.

[68] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. arxiv 2015. *arXiv preprint arXiv:1511.06434*, 2015.

[69] Y Bengio. Learning deep architectures for ai, 2009.

[70] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[71] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2223–2232, 2017.

[72] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. *Advances in neural information processing systems*, 32, 2019.

[73] Altyeb Altaher Taha and Sharaf Jameel Malebary. An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine. *IEEE access*, 8:25579–25587, 2020.

[74] Guoling Zhang, Xiaodan Wang, Rui Li, Yafei Song, Jiaxing He, and Jie Lai. Network intrusion detection based on conditional wasserstein generative adversarial network and cost-sensitive stacked autoencoder. *IEEE access*, 8:190431–190447, 2020.

[75] Matt Shannon, Ben Poole, Soroosh Mariooryad, Tom Bagby, Eric Battenberg, David Kao, Daisy Stanton, and RJ Skerry-Ryan. Non-saturating gan training as divergence minimization. *arXiv preprint arXiv:2010.08029*, 2020.

[76] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017.

[77] Ishan Sohony, Rameshwar Pratap, and Ullas Nambiar. Ensemble learning for credit card fraud detection. In *Proceedings of the ACM India joint international conference on data science and management of data*, pages 289–294, 2018.

[78] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[79] Tero Karras. A style-based generator architecture for generative adversarial networks. *arXiv preprint arXiv:1812.04948*, 2019.

[80] Andrew Brock. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[81] Tero Karras. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[82] Monirah Al-Ajlan and Mourad Ykhlef. A review of generative adversarial networks for intrusion detection systems: Advances, challenges, and future directions. *Computers, Materials & Continua*, 81(2), 2024.

[83] JooHwa Lee and KeeHyun Park. Gan-based imbalanced data intrusion detection system. *Personal and Ubiquitous Computing*, 25(1):121–128, 2021.

[84] Shuokang Huang and Kai Lei. Igan-ids: An imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks. *Ad Hoc Networks*, 105:102177, 2020.

[85] E. Seo, H.M. Song, and H.K. Kim. GIDS: GAN based intrusion detection system for in-vehicle network. In *Proceedings of the 2018 16th Annual Conference on Privacy, Security and Trust (PST)*, pages 1–6, Belfast, Ireland.

[86] Cheolhee Park, Jonghoon Lee, Youngsoo Kim, Jong-Geun Park, Hyunjin Kim, and Dowon Hong. An enhanced ai-based network intrusion detection system using generative adversarial networks. *IEEE Internet of Things Journal*, 10(3):2330–2345, 2022.

[87] Kunda Suresh Babu and Yamarthi Narasimha Rao. Mcgan: modified conditional generative adversarial network (mcgan) for class imbalance problems in network intrusion detection system. *Applied Sciences*, 13(4):2576, 2023.

[88] Saifur Rahman, Shantanu Pal, Shubh Mittal, Tisha Chawla, and Chandan Karmakar. Syn-gan: A robust intrusion detection system using gan-based synthetic data for iot security. *Internet of Things*, 26:101212, 2024.

[89] Ly Vu, Quang Uy Nguyen, Diep N Nguyen, Dinh Thai Hoang, and Eryk Dutkiewicz. Deep generative learning models for cloud intrusion detection systems. *IEEE Transactions on Cybernetics*, 53(1):565–577, 2022.

[90] Zina Chkirbene, Habib Ben Abdallah, Kawther Hassine, Ridha Hamila, and Aiman Erbad. Data augmentation for intrusion detection and classification in cloud networks. In *2021 International Wireless Communications and Mobile Computing (IWCMC)*, pages 831–836. IEEE, 2021.

[91] Zhurong Wang, Jing Zhou, and Xinhong Hei. Network traffic anomaly detection based on generative adversarial network and transformer. In *The International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, pages 228–235. Springer, 2022.

[92] Alexander Geiger, Dongyu Liu, Sarah Alnegheimish, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Tadgan: Time series anomaly detection using generative adversarial networks. In *2020 ieee international conference on big data (big data)*, pages 33–43. IEEE, 2020.

[93] Ankit Thakkar and Ritika Lohiya. A review of the advancement in intrusion detection datasets. *Procedia Computer Science*, 167:636–645, 2020.

[94] Imtiaz Ullah and Qusay H Mahmoud. A framework for anomaly detection in iot networks using conditional generative adversarial networks. *IEEE Access*, 9:165907–165931, 2021.

[95] Vajiheh Hajisalem and Shahram Babaie. A hybrid intrusion detection system based on abc-afs algorithm for misuse and anomaly detection. *Computer Networks*, 136:37–50, 2018.

[96] Andrei-Grigore Mari, Daniel Zinca, and Virgil Dobrota. Development of a machine-learning intrusion detection system and testing of its performance using a generative adversarial network. *Sensors*, 23(3):1315, 2023.

[97] Sahar Aldhaheri and Abeer Alhuzali. Sgan-ids: self-attention-based generative adversarial network against intrusion detection systems. *Sensors*, 23(18):7796, 2023.

[98] Caroline Strickland, Muhammad Zakar, Chandrika Saha, Sareh Soltani Nejad, Noshin Tasnim, Daniel J Lizotte, and Anwar Haque. Drl-gan: A hybrid approach for binary and multiclass network intrusion detection. *Sensors*, 24(9):2746, 2024.

[99] Ibrahim Yilmaz, Rahat Masum, and Ambareen Siraj. Addressing imbalanced data problem with generative adversarial network for intrusion detection. In *2020 IEEE 21st international conference on information reuse and integration for data science (IRI)*, pages 25–30. IEEE, 2020.

[100] Cheolhee Park, Jonghoon Lee, Youngsoo Kim, Jong-Geun Park, Hyunjin Kim, and Dowon Hong. An enhanced ai-based network intrusion detection system using generative adversarial networks. *IEEE Internet of Things Journal*, 10(3):2330–2345, 2022.

[101] Yamarthi Narasimha Rao and Kunda Suresh Babu. An imbalanced generative adversarial network-based approach for network intrusion detection in an imbalanced dataset. *Sensors*, 23(1):550, 2023.

[102] D Suja Mary, L Jaya Singh Dhas, AR Deepa, Mousmi Ajay Chaurasia, and C Jaspin Jeba Sheela. Network intrusion detection: An optimized deep learning approach using big data analytics. *Expert Systems with Applications*, 251:123919, 2024.

[103] S Balaji and S Sankara Narayanan. Dynamic distributed generative adversarial network for intrusion detection system over internet of things. *Wireless Networks*, 29(5):1949–1967, 2023.

[104] Auwal Sani Iliyasu and Huifang Deng. N-gan: a novel anomaly-based network intrusion detection with generative adversarial networks. *International Journal of Information Technology*, 14(7):3365–3375, 2022.

[105] Mahdis Saharkhizan, Amin Azmoodeh, Hamed HaddadPajouh, Ali Dehghantanha, Reza M Parizi, and Gautam Srivastava. A hybrid deep generative local metric learning method for intrusion detection. *Handbook of Big Data Privacy*, pages 343–357, 2020.

[106] Matthieu Mouyart, Guilherme Medeiros Machado, and Jae-Yun Jun. A multi-agent intrusion detection system optimized by a deep reinforcement learning approach with a dataset enlarged using a generative model to reduce the bias effect. *Journal of Sensor and Actuator Networks*, 12(5):68, 2023.

[107] Marc Chalé and Nathaniel D Bastian. Generating realistic cyber data for training and evaluating machine learning classifiers for network intrusion detection systems. *Expert Systems with Applications*, 207:117936, 2022.

[108] Ryuichi Yamamoto, Eunwoo Song, and Jae-Min Kim. Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6199–6203. IEEE, 2020.

[109] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018.

[110] T Saranya, S Sridevi, C Deisy, Tran Duc Chung, and MKA Ahamed Khan. Performance analysis of machine learning algorithms in intrusion detection system: A review. *Procedia Computer Science*, 171:1251–1260, 2020.

[111] Ola M Surakhi and Mohammad Y AlKhanafseh. Review on the application of blockchain technology to compact covid-19 pandemic. In *2021 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT)*, pages 193–198. IEEE, 2021.

[112] O. Surakhi, A. García, M. Jamoos, and M. Alkhanafseh. A comprehensive survey for machine learning and deep learning applications for detecting intrusion detection. In *22nd International Arab Conference on Information Technology*, pages 1–13, 2021.

[113] Ola Surakhi, Martha A Zaidan, Pak Lun Fung, Naser Hossein Motlagh, Sami Serhan, Mohammad AlKhanafseh, Rania M Ghoniem, and Tareq Hussein. Time-lag selection for time-series forecasting using neural network and heuristic algorithm. *Electronics*, 10(20):2518, 2021.

[114] Pak Lun Fung, Martha Arbayani Zaidan, Ola Surakhi, Sasu Tarkoma, Tuukka Petäjä, and Tareq Hussein. Data imputation in in situ-measured particle size distributions by means of neural networks. *Atmospheric Measurement Techniques*, 14(8):5535–5554, 2021.

[115] Adamu I Abubakar, Haruna Chiroma, Sanah Abdullahi Muaz, and Libabatu Baballe Ila. A review of the advances in cyber security benchmark datasets for evaluating data-driven based intrusion detection systems. *Procedia Computer Science*, 62:221–227, 2015.

[116] V. Sharmila, S. Kannadhasan, A. R. Kannan, P. Sivakumar, and V. Vennila, editors. *Challenges in Information, Communication and Computing Technology: Proceedings of the 2nd International Conference on Challenges in Information, Communication, and Computing Technology (ICCICCT 2024), April 26th 27th, 2024, Namakkal, Tamil Nadu, India*. CRC Press, 1st edition, 2024.

[117] Abdulsalam O Alzahrani and Mohammed JF Alenazi. Designing a network intrusion detection system based on machine learning for software defined networks. *Future Internet*, 13(5):111, 2021.

[118] Ming Zhong, Yajin Zhou, and Gang Chen. Sequential model based intrusion detection system for iot servers using deep learning methods. *Sensors*, 21(4):1113, 2021.

[119] Basim Mahbooba, Radhya Sahal, Wael Alosaimi, and Martin Serrano. Trust in intrusion detection systems: an investigation of performance analysis for machine learning and deep learning models. *Complexity*, 2021(1):5538896, 2021.

[120] Muhammad Ashfaq Khan. Hcrnnids: Hybrid convolutional recurrent neural network-based network intrusion detection system. *Processes*, 9(5):834, 2021.

[121] Ruizhe Yao, Ning Wang, Zhihui Liu, Peng Chen, and Xianjun Sheng. Intrusion detection system in the advanced metering infrastructure: a cross-layer feature-fusion cnn-lstm-based approach. *Sensors*, 21(2):626, 2021.

[122] Vibekananda Dutta, Michał Choraś, Rafał Kozik, and Marek Pawlicki. Hybrid model for improving the classification effectiveness of network intrusion detection. In *13th International Conference on Computational Intelligence in Security for Information Systems (CISIS 2020) 12*, pages 405–414. Springer, 2021.

[123] Chao Liang, Bharanidharan Shanmugam, Sami Azam, Asif Karim, Ashraful Islam, Mazdak Zamani, Sanaz Kavianpour, and Norbik Bashah Idris. Intrusion detection system for the internet of things based on blockchain and multi-agent systems. *Electronics*, 9(7):1120, 2020.

[124] J Olamantanmi Mebawondu, Olufunso D Alowolodu, Jacob O Mebawondu, and Adebayo O Adetunmbi. Network intrusion detection system using supervised learning paradigm. *Scientific African*, 9:e00497, 2020.

[125] Tongtong Su, Huazhi Sun, Jinqi Zhu, Sheng Wang, and Yabo Li. Bat: Deep learning methods on network intrusion detection using nsl-kdd dataset. *IEEE Access*, 8:29575–29585, 2020.

[126] I Sumaiya Thaseen, J Saira Banu, K Lavanya, Muhammad Rukunuddin Ghalib, and Kumar Abhishek. An integrated intrusion detection system using correlation-based attribute selection and artificial neural network. *Transactions on Emerging Telecommunications Technologies*, 32(2):e4014, 2021.

[127] Jiyeon Kim, Jiwon Kim, Hyunjung Kim, Minsun Shim, and Eunjung Choi. Cnn-based network intrusion detection against denial-of-service attacks. *Electronics*, 9(6):916, 2020.

[128] M Oguz Kaplan and S Emre Alptekin. An improved bigan based approach for anomaly detection. *Procedia Computer Science*, 176:185–194, 2020.

[129] Rajlaxmi Patil, Rajshekhar Biradar, Vinayakumar Ravi, Poornima Biradar, and Uttam Ghosh. Network traffic anomaly detection using pca and bigan. *Internet Technology Letters*, 5(1):e235, 2022.

[130] Md Hasan Shahriar, Nur Imtiazul Haque, Mohammad Ashiqur Rahman, and Miguel Alonso. G-ids: Generative adversarial networks assisted intrusion detection system. In *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 376–385. IEEE, 2020.

[131] Sana Ullah Jan, Saeed Ahmed, Vladimir Shakhov, and Insoo Koo. Toward a lightweight intrusion detection system for the internet of things. *IEEE access*, 7:42450–42471, 2019.

[132] Farrukh Aslam Khan, Abdu Gumaei, Abdelouahid Derhab, and Amir Hussain. A novel two-stage deep learning model for efficient network intrusion detection. *Ieee Access*, 7:30373–30385, 2019.

[133] Bahram Hajimirzaei and Nima Jafari Navimipour. Intrusion detection for cloud computing using neural networks and artificial bee colony optimization algorithm. *Ict Express*, 5(1):56–59, 2019.

[134] Osama Faker and Erdogan Dogdu. Intrusion detection using big data and deep learning techniques. In *Proceedings of the 2019 ACM Southeast conference*, pages 86–93, 2019.

[135] Yuelei Xiao and Xing Xiao. An intrusion detection system based on a simplified residual network. *Information*, 10(11):356, 2019.

[136] Belal Sudqi Khater, Ainuddin Wahid Bin Abdul Wahab, Mohd Yamani Idna Bin Idris, Mohammed Abdulla Hussain, and Ashraf Ahmed Ibrahim. A lightweight perceptron-based intrusion detection system for fog computing. *applied sciences*, 9(1):178, 2019.

[137] Geethapriya Thamilarasu and Shiven Chawla. Towards deep-learning-driven intrusion detection for the internet of things. *Sensors*, 19(9):1977, 2019.

[138] Wang Peng, Xiangwei Kong, Guojin Peng, Xiaoya Li, and Zhongjie Wang. Network intrusion detection based on deep learning. In *2019 International Conference on Communications, Information System and Computer Engineering (CISCE)*, pages 431–435. IEEE, 2019.

[139] Yalei Ding and Yuqing Zhai. Intrusion detection system for nsl-kdd dataset using convolutional neural networks. In *Proceedings of the 2018 2nd International conference on computer science and artificial intelligence*, pages 81–85, 2018.

[140] Ngoc Tu Pham, Ernest Foo, Suriadi Suriadi, Helen Jeffrey, and Hassan Fareed M Lahza. Improving performance of intrusion detection system using ensemble methods and feature selection. In *Proceedings of the Australasian computer science week multiconference*, pages 1–6, 2018.

[141] Sinh-Ngoc Nguyen, Van-Quyet Nguyen, Jintae Choi, and Kyungbaek Kim. Design and implementation of intrusion detection system using convolutional neural network for dos detection. In *Proceedings of the 2nd international conference on machine learning and soft computing*, pages 34–38, 2018.

[142] Zhipeng Li and Zheng Qin. A semantic parsing based lstm model for intrusion detection. In *Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part IV 25*, pages 600–609. Springer, 2018.

[143] Rasha Almarshdi, Laila Nassef, Etimad Fadel, and Nahed Alowidi. Hybrid deep learning based attack detection for imbalanced data classification. *Intelligent Automation & Soft Computing*, 35(1), 2023.

[144] Guojie Liu and Jianbiao Zhang. Cnid: research of network intrusion detection based on convolutional neural network. *Discrete Dynamics in Nature and Society*, 2020(1):4705982, 2020.

[145] Chaofei Tang, Nurbol Luktarhan, and Yuxin Zhao. Saae-dnn: Deep learning method on intrusion detection. *Symmetry*, 12(10):1695, 2020.

[146] Ngamba Thockchom, Moirangthem Marjit Singh, and Utpal Nandi. A novel ensemble learning-based model for network intrusion detection. *Complex & Intelligent Systems*, 9(5):5693–5714, 2023.

[147] Shiva Sattarpour, Ali Barati, and Hamid Barati. Ebids: efficient bert-based intrusion detection system in the network and application layers of iot. *Cluster Computing*, 28(2):1–21, 2025.

[148] Himanshu Nandanwar and Rahul Katarya. Deep learning enabled intrusion detection system for industrial iot environment. *Expert Systems with Applications*, 249:123808, 2024.

[149] Sami Yaras and Murat Dener. Iot-based intrusion detection system using new hybrid deep learning algorithm. *Electronics*, 13(6):1053, 2024.

[150] Joaquín Gaspar Medina-Arco, Roberto Magán-Carrión, Rafael Alejandro Rodríguez-Gómez, and Pedro García-Teodoro. Methodology for the detection of contaminated training datasets for machine learning-based network intrusion-detection systems. *Sensors*, 24(2):479, 2024.

[151] Jinliang Fan, Jun Xu, Mostafa H Ammar, and Sue B Moon. Prefix-preserving ip address anonymization: measurement-based security evaluation and a new cryptography-based scheme. *Computer Networks*, 46(2):253–272, 2004.

[152] P. Haag. NFDUMP-NetFlow Processing Tools, 2011. Accessed on 16 June 2023.

[153] Siamak Layeghy, Marcus Gallagher, and Marius Portmann. Benchmarking the benchmark—comparing synthetic and real-world network ids datasets. *Journal of Information Security and Applications*, 80:103689, 2024.

[154] Waqas Haider, Gideon Creech, Yi Xie, and Jiankun Hu. Windows based data sets for evaluation of robustness of host based intrusion detection systems (ids) to zero-day and stealth attacks. *Future Internet*, 8(3):29, 2016.

[155] School of Engineering and Information Technology, UNSW, Australia. ADFA Linux Data Set (ADFA-LD) Cyber Security Benchmark Dataset, 2021. Accessed: 2021.

[156] Nickolaos Koroniotis, Nour Moustafa, Elena Sitnikova, and Benjamin Turnbull. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Generation Computer Systems*, 100:779–796, 2019.

[157] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. Ddos in the iot: Mirai and other botnets. *Computer*, 50(7):80–84, 2017.

[158] Iman Sharafaldin, Arash Habibi Lashkari, Ali A Ghorbani, et al. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, 1:108–116, 2018.

[159] Ranjit Panigrahi and Samarjeet Borah. A detailed analysis of cicids2017 dataset for designing intrusion detection systems. *International Journal of Engineering & Technology*, 7(3.24):479–482, 2018.

[160] Zafar Iqbal Khan, Mohammad Mazhar Afzal, and Khurram Naim Shamsi. A comprehensive study on cic-ids2017 dataset for intrusion detection systems. *International Research Journal on Advanced Engineering Hub (IRJAEH)*, 2(02):254–260, 2024.

[161] CSE-CIC-IDS2018, 2022. Last visited: 2022.

[162] S. Stolfo, W. Fan, W. Lee, and A. Prodromidis. KDD Cup 1999 Dataset, 2018. Last visited: 2018.

[163] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. A detailed analysis of the kdd cup 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications*, pages 1–6. Ieee, 2009.

[164] Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 military communications and information systems conference (MilCIS)*, pages 1–6. IEEE, 2015.

[165] Iman Almomani, Bassam Al-Kasasbeh, and Mousa Al-Akhras. Wsn-ds: a dataset for intrusion detection systems in wireless sensor networks. *Journal of Sensors*, 2016(1):4731953, 2016.

[166] Richard Lippmann, Joshua W Haines, David J Fried, Jonathan Korba, and Kumar Das. The 1999 darpa off-line intrusion detection evaluation. *Computer networks*, 34(4):579–595, 2000.

[167] Gideon Creech and Jiankun Hu. Generation of a new ids test dataset: Time to retire the kdd collection. In *2013 IEEE wireless communications and networking conference (WCNC)*, pages 4487–4492. IEEE, 2013.

[168] Mohanad Sarhan, Siamak Layeghy, Nour Moustafa, and Marius Portmann. Netflow datasets for machine learning-based network intrusion detection systems. In *Big Data Technologies and Applications: 10th EAI International Conference, BDTA 2020, and 13th EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, December 11, 2020, Proceedings 10*, pages 117–135. Springer, 2021.

[169] Elaheh Biglar Beigi, Hossein Hadian Jazi, Natalia Stakhanova, and Ali A Ghorbani. Towards effective feature selection in machine learning-based botnet detection approaches. In *2014 IEEE Conference on Communications and Network Security*, pages 247–255. IEEE, 2014.

[170] Xinxing Zhao, Kar Wai Fok, and Vrizlynn LL Thing. Enhancing network intrusion detection performance using generative adversarial networks. *arXiv preprint arXiv:2404.07464*, 2024.

[171] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[172] Hui Su, Xiaoyu Shen, Pengwei Hu, Wenjie Li, and Yun Chen. Dialogue generation with gan. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[173] Samuel Ndichu, Tao Ban, Takeshi Takahashi, and Daisuke Inoue. Ai-assisted security alert data analysis with imbalanced learning methods. *Applied Sciences*, 13(3):1977, 2023.

[174] Zhengwei Wang, Qi She, and Tomas E Ward. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021.

[175] Wenqian Jiang, Yang Hong, Beitong Zhou, Xin He, and Cheng Cheng. A gan-based anomaly detection approach for imbalanced industrial time series. *IEEE Access*, 7:143608–143619, 2019.

[176] Yun Yang, Fengtao Nan, Po Yang, Qiang Meng, Yingfu Xie, Dehai Zhang, and Khan Muhammad. Gan-based semi-supervised learning approach for clinical decision support in health-iot platform. *Ieee Access*, 7:8048–8057, 2019.

[177] Xin Wang, Hui Guo, Shu Hu, Ming-Ching Chang, and Siwei Lyu. Gan-generated faces detection: A survey and new perspectives. *ECAI 2023*, pages 2533–2542, 2023.

[178] Xuan Xia, Xizhou Pan, Nan Li, Xing He, Lin Ma, Xiaoguang Zhang, and Ning Ding. Gan-based anomaly detection: A review. *Neurocomputing*, 493:497–535, 2022.

[179] Shin Fujieda, Atsushi Yoshimura, and Takahiro Harada. Local positional encoding for multi-layer perceptrons. *arXiv preprint arXiv:2309.15101*, 2023.

[180] Ravi Vinayakumar, Mamoun Alazab, K Padannayil Soman, Prabaharan Poornachandran, Ameer Al-Nemrat, and Sitalakshmi Venkatraman. Deep learning approach for intelligent intrusion detection system. *Ieee Access*, 7:41525–41550, 2019.

[181] Ruchi Bhatt and Gaurav Indra. Detecting the undetectable: Gan-based strategies for network intrusion detection. *International Journal of Information Technology*, pages 1–7, 2024.

[182] Ibomoiye Domor Mienye, Yanxia Sun, and Zenghui Wang. Prediction performance of improved decision tree-based algorithms: a review. *Procedia Manufacturing*, 35:698–703, 2019.

[183] Rifkie Primartha and Bayu Adhi Tama. Anomaly detection using random forest: A performance revisited. In *2017 International conference on data and software engineering (ICoDSE)*, pages 1–6. IEEE, 2017.

[184] Abdulrahman Jassam Mohammed, Muhanad Hameed Arif, and Ali Adil Ali. A multilayer perceptron artificial neural network approach for improving the accuracy of intrusion detection systems. *IAES International Journal of Artificial Intelligence*, 9(4):609, 2020.

[185] Jie Gu and Shan Lu. An effective intrusion detection approach using svm with naïve bayes feature embedding. *Computers & Security*, 103:102158, 2021.

[186] Stacey J Winham, Robert R Freimuth, and Joanna M Biernacka. A weighted random forests approach to improve predictive performance. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 6(6):496–505, 2013.

[187] Lukas Schoppa, Markus Disse, and Sophie Bachmair. Evaluating the performance of random forest for large-scale flood discharge simulation. *Journal of Hydrology*, 590:125531, 2020.

[188] Arnaz Malhi and Robert X Gao. Pca-based feature selection scheme for machine defect classification. *IEEE transactions on instrumentation and measurement*, 53(6):1517–1525, 2004.