



UNIVERSIDADE D  
COIMBRA

Dylan Gonçalves Perdigão

TOWARDS AN INTRUSION DETECTION  
SYSTEM FOR SMART GRIDS  
A FEDERATED APPROACH

Dissertation in the context of the Master in Informatics Engineering,  
specialization in Intelligent Systems, advised by Prof. Dr. Pedro Henriques  
Abreu and Prof. Dr. Tiago José Cruz and presented to the Department of  
Informatics Engineering of the Faculty of Sciences and Technology of the  
University of Coimbra.


July 2023





FACULDADE DE  
CIÊNCIAS E TECNOLOGIA  
UNIVERSIDADE DE  
**COIMBRA**

DEPARTMENT OF INFORMATICS ENGINEERING

Dylan Gonçalves Perdigão 

# Towards an Intrusion Detection System for Smart Grids

A Federated Approach

Dissertation in the context of the Master in Informatics Engineering, specialization in Intelligent Systems, advised by Prof. Dr. Pedro Henriques Abreu and Prof. Dr. Tiago José Cruz, and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

July 2023



BIBTEX:

```
@mastersthesis{Perdigao2023MScThesis,  
  author = "Dylan Perdigão",  
  title = "Towards An Intrusion Detection System  
    For Smart Grids: A Federated Approach",  
  school = "University of Coimbra",  
  year = "2023"  
}
```

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with its author and that no quotation from the thesis and no information derived from it may be published without proper acknowledgement.

*Esta cópia da tese é fornecida na condição de que quem a consulta reconhece que os direitos de autor são pertença do autor da tese e que nenhuma citação ou informação obtida a partir dela pode ser publicada sem a referência apropriada.*



If something is important enough, even if the odds are against you, you should still do it.

---

*Elon Musk*





The work presented in this thesis was carried out within the Communications and Telematics (*CT*) group of the Centre for Informatics and Systems of the University of Coimbra (*CISUC*) in the context of the following project:

- *Smart5Grid - Automated 5G Networks and Services for Smart Grids.*  
FCT (P2020) POCI-01-0247-FEDER-047226

This work has been partially supported by grant nr 789550-1



## Acknowledgements *Agradecimentos*

I would like to express my deep gratitude to my two supervisors, Professor Doctor Pedro Henriques Abreu and Professor Doctor Tiago Cruz, for their invaluable guidance and support throughout the entire research process.

I am extremely grateful to my parents and grandparents for their love and support throughout my studies. Thank you for believing in me and for encouraging me to pursue my passions.

I would like to thank my friends for their support and encouragement during my studies. Your friendships have been a constant source of strength and motivation.

I would also like to thank Quantunna for becoming like a second family and for providing me breaks after the long and exhausting nights of work on this thesis.

Finally, I would also like to thank the University of Coimbra for providing the resources and materials necessary for this research, and the Department of Informatics Engineering, which has truly become a second home for me.

Thank you all for your help and support. This thesis would not have been possible without you.

*Gostaria de expressar minha profunda gratidão aos meus dois orientadores, Professor Doutor Pedro Henriques Abreu e Professor Doutor Tiago Cruz, pela valiosa orientação e pelo apoio ao longo de todo o processo de investigação.*

*Sou extremamente grato aos meus pais e avós pelo amor e pelo apoio durante meus estudos. Obrigado por acreditarem em mim e por me incentivarem a seguir as minhas paixões.*

*Gostaria de agradecer aos meus amigos pelo apoio e incentivo durante meus estudos. As vossas amizades foram uma fonte constante de força e motivação.*

*Gostaria também de agradecer à Quantunna por se tornar como uma segunda família e por me proporcionar momentos de descanso após as longas e exaustivas noites de trabalho nesta tese.*

*Por fim, gostaria de agradecer à Universidade de Coimbra pelos recursos e materiais disponibilizados para esta pesquisa, bem como ao Departamento de Engenharia Informática, que se tornou realmente uma segunda casa para mim.*

*Obrigado a todos pelas vossas ajudas e pelos vossos apoios. Esta tese não teria sido possível sem vocês.*



# Abstract

The emergence of 5G and the Internet of Things has prompted nations across the globe to develop smart grids, an advanced electrical grid infrastructure capable of producing and transporting energy in a more reliable, sustainable, and efficient way. In addition, smart grids allow monitoring in real-time the energy production, consumption, and distribution with smart meters. The grid's large amount of data enables intelligent energy flow management to end-point consumers. With renewable energy sources such as solar and wind, this management is crucial to balance electricity supply and demand efficiently, avoiding energy loss. However, smart grids face several challenges, such as costs for upgrading current infrastructures, interoperability with the traditional energy grids, or privacy and cybersecurity risks due to the large number of distributed devices providing data to the grid.

This work will focus on cybersecurity risks and, more specifically, anomaly detection. The proposed solution enables us to detect real-time cyberattacks in the smart grid through the network traffic between devices. Anomaly detection is performed through time-series classifiers and by attending to real and frequent problems, such as data imbalance and causality, using real-world network traffic datasets. To turn our simulated context more realistic, we decided to simulate a smart grid with a federated learning environment through the *Flower* framework. Similar to a conventional smart grid, the system comprises several distributed nodes communicating between them.

Two approaches are tested in the federated system. The first approach, as a baseline, only trains a time-series classifier. The second approach uses a data processing method, which oversamples minority class examples, increases data causality, and then trains the time-series classifier.

As a result, the proposed approach method presents significant classification performance in comparison to the baseline over datasets such as *IEC61850-Security*, *NSL-KDD*, *BOT-IoT*, and *UNSW-NB15* datasets. Compared to the reviewed literature, the simulation method and methodology perform better.

**Keywords:** Smart Grid · Anomaly Detection · Federated Learning · Imbalanced Data · Causal Inference · Time-Series Classification



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Context and Motivation . . . . .	2
1.2	Research Goals . . . . .	3
1.3	Research Contributions . . . . .	4
1.4	Document Structure . . . . .	4
<b>2</b>	<b>Background Knowledge</b>	<b>5</b>
2.1	Federated Learning . . . . .	5
2.1.1	Phases of Federated Learning . . . . .	6
2.1.2	Data Partitioning . . . . .	7
2.1.3	Federation Scale . . . . .	8
2.1.4	Aggregation Strategies for Federated Learning . . . . .	9
2.1.5	Privacy Mechanisms . . . . .	13
2.1.6	Frameworks for Federated Learning . . . . .	13
2.1.7	Summary . . . . .	14
2.2	Imbalanced Data . . . . .	15
2.2.1	Distance Metrics . . . . .	15
2.2.2	Types of Examples in Minority Classes . . . . .	18
2.2.3	Imbalanced Data Processing Methods . . . . .	19
2.2.4	Summary . . . . .	22
2.3	Causal Inference . . . . .	23
2.3.1	Causal Discovery . . . . .	24
2.3.2	Causality Evaluation . . . . .	24
2.3.3	Refutation Methods . . . . .	25
2.3.4	Summary . . . . .	26
2.4	Time-Series Classifiers . . . . .	26
2.4.1	Shapelet-based . . . . .	27
2.4.2	Distance-based . . . . .	28
2.4.3	Interval-based . . . . .	29
2.4.4	Dictionary-based . . . . .	30
2.4.5	Hybrid Approaches . . . . .	30
2.4.6	Summary . . . . .	31
2.5	Conclusions . . . . .	31
<b>3</b>	<b>Literature Review</b>	<b>33</b>
3.1	Datasets for Intrusion Detection Systems . . . . .	33
3.1.1	Network-based Datasets . . . . .	33
3.1.2	IoT-based Datasets . . . . .	34

3.1.3	Smart Grid-based Datasets . . . . .	34
3.1.4	Summary . . . . .	35
3.2	Intrusion Detection Systems in Smart Grids . . . . .	35
3.2.1	Non-Distributed Approaches . . . . .	35
3.2.2	Distributed Approaches . . . . .	36
3.2.3	Summary . . . . .	37
3.3	Conclusions . . . . .	37
<b>4</b>	<b>Experimental Setup</b>	<b>39</b>
4.1	Dataset Collection . . . . .	40
4.2	Proposed Approach . . . . .	42
4.3	Evaluation Metrics . . . . .	44
4.4	Federated System . . . . .	45
4.5	Conclusions . . . . .	47
<b>5</b>	<b>Results and Discussion</b>	<b>49</b>
5.1	First Experimental Phase . . . . .	49
5.1.1	Time-Series Classifier Selection . . . . .	49
5.1.2	Structural Causal Model Discovery . . . . .	50
5.1.3	Summary . . . . .	51
5.2	Second Experimental Phase . . . . .	51
5.2.1	Attack Detection . . . . .	52
5.2.2	Denial of Service Detection . . . . .	53
5.2.3	Detection of Other Attacks . . . . .	54
5.2.4	Summary . . . . .	56
5.3	Conclusions . . . . .	56
<b>6</b>	<b>Conclusion</b>	<b>59</b>
	<b>References</b>	<b>61</b>
	<b>Appendix A Datasets</b>	<b>77</b>
	<b>Appendix B Results Tables</b>	<b>87</b>



# Acronyms

- AE** auto-encoder. 36, 37
- AMI** advanced metering infrastructure. 1, 26
- ATE** average treatment effect. 24, 25, 26, 43, 46, 51
- BOSS** bag-of-SFA-symbols. 30, 31
- cBOSS** contractable BOSS. 30, 39
- CD** cross-device. 8, 14
- CIF** canonical interval forest. 29, 31, 39
- CNN** convolutional neural network. 12, 36, 37
- CS** cross-silo. 8, 14
- DAG** directed acyclic graph. 24, 32, 50
- DDoS** distributed DoS. 34, 41
- DER** distributed energy resource. 1, 2, 36
- DHCP** dynamic host configuration protocol. 82
- DL** deep learning. 6, 9, 37
- DNN** deep neural network. 9, 36, 37
- DNS** domain name system. 41, 53, 81, 82, 85
- DoS** denial of service. 4, 33, 34, 35, 40, 41, 51, 53, 54, 56, 59
- DP** differential privacy. 13, 14
- DrCIF** diverse representation CIF. 29, 31
- DT** decision tree. 35, 36, 37
- DTW** dynamic time wrapping. 28
- EE** elastic ensemble. 28, 30, 31
- FL** federated learning. 3, 5, 6, 8, 9, 10, 12, 13, 33, 36, 37, 59
- FPR** false positive rate. 44

**FTP** file transfer protocol. 33, 34, 41, 83, 84, 86

**GAN** generative adversarial network. 36, 37

**GOOSE** generic object oriented substation event. 2, 34

**GRU** gated recurrent unit. 36, 37

**GUI** graphical user interface. 3, 47, 59

**HE** homographic encryption. 13, 14

**HEOM** heterogeneous euclidean-overlap metric. 17, 22, 23

**HFL** horizontal federated learning. 7, 8, 9, 10, 14

**HTTP** hypertext transfer protocol. 34, 41, 81, 82, 83, 84, 85, 86

**HVDM** heterogeneous value difference metric. 17, 18, 19, 20, 22, 42, 43

**ICMP** internet control message protocol. 34, 41, 78, 81, 82, 85

**IDS** intrusion detection system. 3, 33, 35, 37

**IED** intelligent electronic device. 1, 2, 34

**IIoT** industrial internet of things. 33, 34, 35

**IMAP** internet message access protocol. 33, 41

**IoT** internet of things. 1, 2, 3, 6, 9, 33, 34, 35, 36, 37

**IP** internet protocol. 3, 33, 41, 42, 45, 78, 81, 82, 83, 84, 85, 86

**IR** imbalance ratio. 15

**ITE** individual treatment effect. 24, 25

**KNN** K-nearest neighbor. 15, 19, 20, 22, 35, 37

**LR** logistic regression. 36, 37

**LSTM** long short-term memory. 12

**MDS** multidimensional scaling. 18, 19

**MITM** man-in-the-middle. 34, 41

**ML** machine learning. 5, 6, 13, 15, 23, 29, 37

**MQTT** message queuing telemetry transport. 3, 45, 46, 47, 59, 81

**NB** naive Bayes. 36, 37

**NCR** neighborhood cleaning rule. 19

**NN** neural networks. 12, 35, 36, 37

- PF** proximity forest. 28, 31
- PSO** particle swarm optimization. 35, 36, 37
- RF** random forest. 35, 36, 37
- RISE** random interval spectral ensemble. 29, 30, 31, 39
- RNN** recurrent neural network. 36, 37
- ROC** receiver operating characteristic. 44
- SCM** structural causal model. 3, 4, 23, 24, 25, 56
- SEM** structural equation model. 24
- SFA** symbolic Fourier approximation. 30
- SGD** stochastic gradient descent. 11
- SMC** secure multiparty computation. 13, 14
- SSH** secure shell. 34, 41, 82
- SSL** secure sockets layer. 82, 85
- STC** shapelet transform classifier. 27, 30, 31
- SVM** support vector machine. 35, 36, 37
- TCP** transmission control protocol. 33, 34, 41, 42, 78, 79, 80, 81, 82, 83, 84, 85, 86
- TDE** temporal dictionary ensemble. 31
- TFL** transfer federated learning. 7, 8
- TNR** true negative rate. 19
- TPR** true positive rate. 19, 44
- TSC** time-series classifier. 27, 37
- TSF** time-series forest. 29, 30, 31, 39, 51, 56, 59
- UDP** user datagram protocol. 34, 41, 78, 81, 82, 83, 84, 85, 86
- URI** uniform resource identifier. 81, 85
- VFL** vertical federated learning. 7, 8, 9, 14
- XSS** cross-site scripting. 34, 41



# List of Figures

1.1	Smart grid structure . . . . .	2
2.1	Phases of federated learning . . . . .	6
2.2	Data partitioning for federated learning . . . . .	8
2.3	Examples of imbalanced data over 100 samples . . . . .	15
2.4	MDS visualizations of the minority classes over the <i>Ecoli</i> dataset . . .	19
2.5	Neighborhood cleaning rule over 1000 samples with $K = 3$ . . . . .	19
2.6	SPIDER3 over 1000 samples with $K = 5$ . . . . .	21
2.7	ADASYN over 1000 samples with $K = 5$ . . . . .	21
2.8	SMOTE over 1000 samples with $K = 5$ . . . . .	22
2.9	Causal graph presenting the different variables . . . . .	23
2.10	Different propensity score methods . . . . .	25
4.1	Experimental phases . . . . .	39
4.2	Attack injection strategy . . . . .	40
4.3	Preprocessing and processing flow . . . . .	44
4.4	Representation of the three-level architecture . . . . .	45
4.5	Representation of the whole system . . . . .	46



# List of Tables

2.1	Summary of federation scale . . . . .	9
2.2	Summary of federated learning frameworks . . . . .	14
2.3	Notation for time-series classifiers . . . . .	27
2.4	Complexity comparison between time-series classifiers . . . . .	31
3.1	Comparison of datasets for intrusion detection systems . . . . .	35
3.2	Summary of research for intrusion detection systems . . . . .	37
4.1	Description of attacks per dataset . . . . .	41
5.1	Time-series classification in binary scenario . . . . .	50
5.2	Treatment variables inferred from <i>NOTEARS</i> . . . . .	51
5.3	Refutation of structural causal models . . . . .	51
5.4	Performance in detecting attacks . . . . .	52
5.5	Kolmogorov-Smirnov normality test for attack scenarios . . . . .	52
5.6	Wilcoxon signed-rank test for attack scenarios . . . . .	53
5.7	Performance in detecting DoS attacks . . . . .	53
5.8	Kolmogorov-Smirnov normality test for DoS attack scenarios . . . . .	54
5.9	Wilcoxon signed-rank test for DoS attack scenarios . . . . .	54
5.10	Performance in detecting other <i>IEC61850-Security</i> attacks . . . . .	55
5.11	Kolmogorov-Smirnov normality test for other <i>IEC61850-Security</i> at- tack scenarios . . . . .	55
5.12	Wilcoxon signed-rank test for other <i>IEC61850-Security</i> attack scenarios	56
A.1	BOT-IoT Dataset Description . . . . .	78
A.2	CIC-IDS-2017 Dataset Description . . . . .	79
A.3	CSE-CIC-IDS-2018 Dataset Description . . . . .	80
A.4	Edge-IIoT Dataset Description . . . . .	81
A.5	IEC61850-Security Dataset Description . . . . .	81
A.6	IoT-23 Dataset Description . . . . .	82
A.7	ISCX-IDS-2012 Dataset Description . . . . .	82
A.8	KDD99 Dataset Description . . . . .	83
A.9	NSL-KDD Dataset Description . . . . .	84
A.10	TON-IoT Dataset Description . . . . .	85
A.11	UNSW-NB15 Dataset Description . . . . .	86
B.1	Time-Series Classification in Binary Scenario (Full Table) . . . . .	88
B.2	Time-Series Classification in Multiclass Scenario (Full Table) . . . . .	89





# List of Algorithms

1	HVDM distance . . . . .	43
2	Classification of minority class examples . . . . .	43



# Chapter 1

## Introduction

The rapid advancement of technology and the internet of things (IoT) has led to the development of smart grids, an advanced version of the traditional electrical power grid, which is designed to improve the reliability, sustainability, and efficiency of the power grid by detecting, responding, and adapting to changes in the power system [33]. The current state of the art in smart grids relies heavily on technologies such as advanced metering infrastructure (AMI), which uses wireless communication to enable remote meter reading and improve the accuracy and reliability of power usage data.

The increasing availability of 5G networks offers the potential to support new applications and capabilities in smart grids, such as improved support for distributed energy resources (DERs) and enhanced grid automation and control. Typically a smart grid architecture (Figure 1.1) is composed of three kinds of entities such as:

- **Energy production plants** are the facilities that generate electricity through solar, wind, thermal, and nuclear energy;
- **Substations** are facilities where the voltage of the electricity is increased or decreased, allowing it to be transmitted over long distances. They also help to distribute electricity to different parts of the grid;
- **Final consumers** are the homes, businesses, and industries that receive energy. They are equipped with smart meters to communicate with the grid to adjust the real-time energy flow sent to the infrastructure.

Smart grids deal with small distributed generation sources, such as homes with photovoltaic panels injecting power into the grid, redirecting the energy flow where the energy is needed. It contrasts with a conventional grid where energy production is centralized into a unique power plant [118]. To manage all this energy traffic, the smart grid comprises an AMI, which monitors power consumption, using remote communication to send readings from smart meters. This is one of the key technologies currently used to improve the accuracy and reliability of power usage data [34].

In a substation, there are intelligent electronic devices (IEDs), which are devices that monitor and control electrical equipment, such as circuit breakers and transformers.

IEDs can be used to automate and optimize the operation of a substation and provide real-time monitoring and control capabilities of the energy flow. The information is exchanged using the generic object oriented substation event (GOOSE) protocol. As the data is sent in real-time, the IEDs can quickly respond to changes in the substation and make the necessary adjustments to keep the grid stable and reliable.

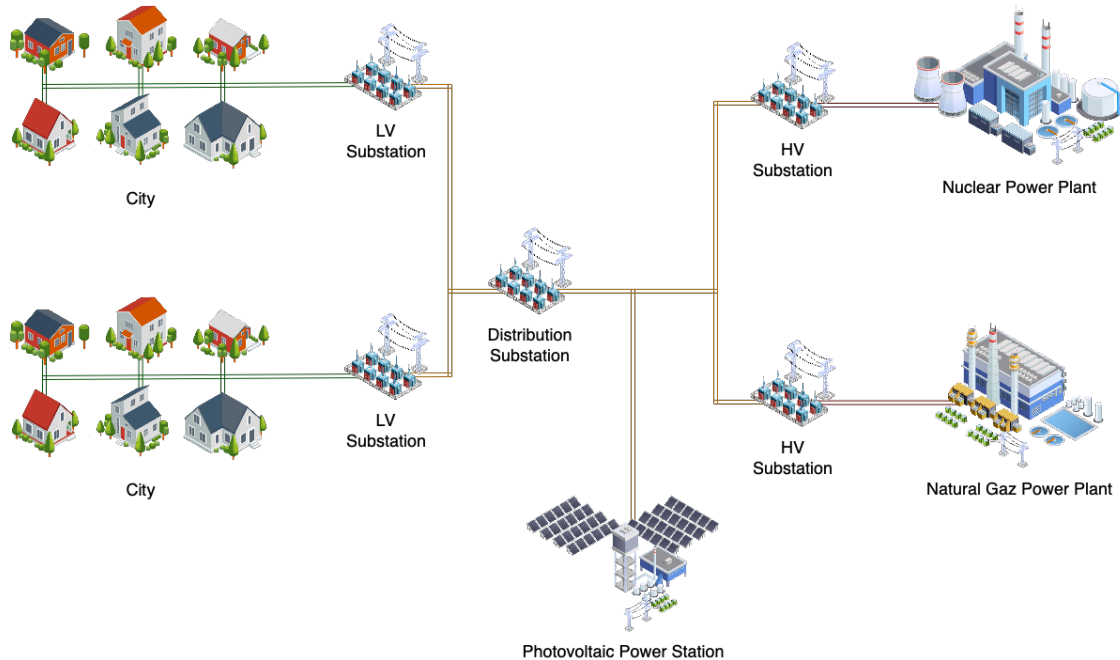


Figure 1.1: Smart grid structure  
[Individual widgets provided by Macrovector [79]]

## 1.1 Context and Motivation

With the increasing availability of 5G networks, smart grids have the potential to support new applications and capabilities, such as improved support for DERs and enhanced grid automation and control [31]. However, smart grids are challenged with some issues [118] such as:

- The management and synchronization of homemade solar energy production with the grid;
- The optimization of renewable energy production and storage depending on the energy demand;
- The management of remote communication between systems and smart meters;
- Their vulnerability to cyberattacks with all the IoT devices;
- The privacy of the data exchanged between devices;
- The storage and processing of large amounts of data.

The main challenge is developing an efficient intrusion detection system to prevent anomalies and mitigate consumer consequences. Furthermore, anomaly detection is crucial in smart grids as it can help detect and mitigate potential threats that may result from cyberattacks or technical failures conducting, in the worst cases, into blackouts. Focusing on network traffic generated by the IoT devices of the grid, these anomalies can be characterized by suspicious communications between internet protocol (IP) addresses, ports, or unusual network packet information [15].

## 1.2 Research Goals

In this work, we developed an intrusion detection system for the smart grid context, capable of detecting in real-time the network traffic anomalies using a distributed environment. Federated learning (FL) is the key paradigm used in this work since it enables the distribution of the nodes to simulate the smart grid architecture, where nodes communicate with each other [81]. However, anomaly detection problems, such as detecting attacks in a system, are affected by several problems, such as causality or poor data quality due to imbalanced data.

Despite the good performances delivered by federated-based intrusion detection systems (IDSs) proposed in the literature, a very small number of these approaches provide real-time information about data imbalance and causality, enabling the understanding of poor classification results. Towards a deeper analysis of the system's data, the main goal of this work is to:

*Develop strategies to predict cyberattacks  
in a federated-based context*

Following this goal, a federated system was developed to simulate the smart grid, in which were used two different approaches:

- The **first approach** (baseline) trains the classifier with the received data and predicts the cyberattacks;
- The **second approach** (ours) performs a processing step on the received data, handling data imbalance and causality problems before training the classifier.

This federated system is monitored by a graphical user interface (GUI), which shows real-time metrics from each node, such as classification performance, data imbalance, and causality. The communication between the federated system and the GUI is done via message queuing telemetry transport (MQTT) protocol [8], which sends those metrics about the system asynchronously.

The experiments in this work attempt are separated into two phases where:

- The **first phase** searches for a time-series classifier that fits the need for fast and efficient classification for the smart grid context, but also for the structural causal model (SCM) representing dataset's variables relationships;

- The **second phase** evaluates the two approaches in different anomaly detection scenarios, using the classifier and the SCM from the previous phase. The first scenario only detects if there is a cyberattack or not using a smart grid-oriented dataset, *IEC61850-Security*, and a binary dataset often used in the literature as *NSL-KDD*. The second scenario consists in detecting individually different attacks as denial of service (DoS) attacks present in *BOT-IoT*, *IEC61850-Security*, and *UNSW-NB15* datasets, but also message suppression, data manipulation, or control manipulation attacks present only in *IEC61850-Security* dataset.

The results obtained by these experiments enable comparing the two approaches (baseline and our data processing) and understanding if the proposed approach surpasses the baseline.

### 1.3 Research Contributions

The work developed during this thesis resulted in the following works:

- Dylan Perdigão, Tiago Cruz, Paulo Simões, and Pedro Henriques Abreu. "*Federated Intrusion Detection System for Smart Grids using Causality*". In Proceedings of ACM International Conference on Information and Knowledge Management (CIKM 2023).  
– *Paper, submitted on the 2<sup>nd</sup> June 2023*
- Dylan Perdigão, Tiago Cruz, and Pedro Henriques Abreu. "*Towards An Intrusion Detection System For Smart Grids: A Federated Approach*". In the 22<sup>nd</sup> European Conference on Cyber Warfare and Security (ECCWS 2023).  
– *Poster, presented on the 23<sup>rd</sup> June 2023, "Best Poster Award" [55]*

### 1.4 Document Structure

The remainder of this thesis is organized as follows. Chapter 2 overviews the background knowledge supporting this work, presenting key concepts such as federated learning, distance metrics, imbalanced data, causality, and time-series classifiers. Chapter 3 highlights datasets and research on intrusion detection systems for smart grids. Chapter 4 describes the architecture of the experimental design for anomaly detection in the federated-based environment, including the proposed strategies. Chapter 5 showcases the results and discusses them. Finally, Chapter 6 concludes the study, highlights the key findings, and suggests directions for future research in this area.

# Chapter 2

## Background Knowledge

This chapter introduces the concepts and technologies used in this work. First, federated learning is described with data partitioning methods, federation scale, aggregation strategies, privacy mechanisms, and frameworks in Section 2.1. After that, the section presents distance metrics used for multiple purposes in this work. Afterward, Section 2.2 discusses imbalanced data, which reviews different approaches to classify minority classes and treat this problem. Thereafter, causality is another topic that affects data and is discussed in Section 2.3. Then, different algorithms are presented in Section 2.4 for classification. Finally, Section 2.5 overviews the whole chapter.

### 2.1 Federated Learning

Federated Learning [81] was proposed by *Google* researchers in 2016. The concept proposes a system that allows solving machine learning problems in a distributed way. In particular, the objective is to minimize the function  $f(w)$  (typically, it can be the loss function, but it can be any function that solves a specific problem), with the model parameters  $w$ , over  $N$  clients:

$$\min_w f(w) = \frac{1}{N} \sum_{i=1}^N f_i(w) = \sum_{k=1}^K \frac{n_k}{n} F_k(w) \quad (2.1)$$

where:

$$F_k(w) = \frac{1}{n_k} \sum_{i \in p_k} f_i(w) \quad (2.2)$$

First, several local nodes (i.e., client devices) train their model locally to preserve data privacy. Then the model computed in each local node is aggregated in a central node (i.e., central server) that will allow updating the local nodes without directly exchanging data. The advantage of FL is that it is scalable since the training of the machine learning (ML) model is distributed into multiple devices in contrast

with centralized learning. This gives FL privacy-preserving capabilities during the training of the ML model since it is done locally on the local node (in the case of deep learning (DL), only the weights are shared with the central node). Although IoT devices like smartphones have, in some cases, little processing power, low bandwidth, and limited storage, they produce sensitive data that should not be transmitted to centralized servers. This data serves to train more accurately their local ML models in a federated system. Note that there is a primary advantage compared to traditional centralized training since the performance of the ML model is better where the data is not independent and identically distributed or where the data is heterogeneous. This means that the data produced in the local node can be utterly different from node to node, and in a centralized context, it will underfit the model. Hopefully, FL solves this problem.

Regarding our smart grid context, federated learning enables the grid architecture simulation, where the model's distributed training is done through nodes representing the substations.

The remainder of this section presents the phases of FL in Section 2.1.1. Then, the concept of data partitioning is explained in Section 2.1.2. Next, the scalability of the FL system is explained in Section 2.1.3. Afterward, Section 2.1.4 gives an overview of aggregation strategies. After that, privacy is discussed in Section 2.1.5, followed by a review of the FL frameworks in Section 2.1.6. Finally, this section is summarized in Section 2.1.7

### 2.1.1 Phases of Federated Learning

Federated Learning encompasses three phases [13] exemplified in Figure 2.1, which are the initialization, the model's local training, and the clients' aggregations.

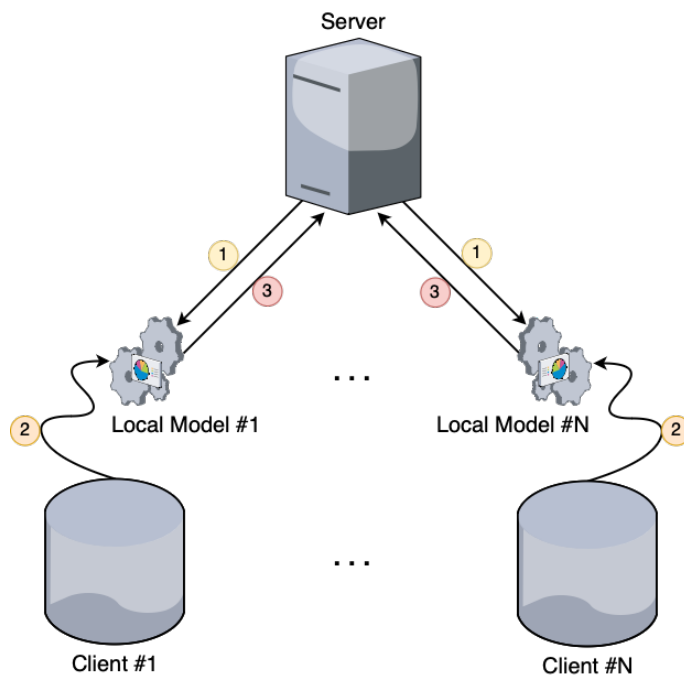


Figure 2.1: Phases of federated learning



## Initialization

The first phase is initialization, where the server selects the clients participating in the federated learning round. The server sends its global model and an initial gradient to the participating clients.

## Local Training

The second phase is local training, where the clients update their local models (in the first round, the model will be identical to the global model). Then, each client trains their local models on its data to send it back to the server.

## Aggregation

The third phase is aggregation, where the server aggregates the models from the clients using a strategy such as FedAvg. For instance, the client model weights are transmitted to the server using this strategy, averaging all weights received. Finally, the process begins again for the subsequent federated learning round.

### 2.1.2 Data Partitioning

A federated system can have its data partitioned in three ways, according to their feature space (i.e., columns) or sample space (i.e., rows), as shown in Figure 2.2. The first is horizontal federated learning (HFL), the second is vertical federated learning (VFL), and the third is transfer federated learning (TFL).

#### Horizontal Federated Learning

Horizontal federated learning shares the same feature space but has a different sample space. It is a simple solution to avoid data leaks because only the parameters of the local and global models are communicated. However, it has the disadvantage of needing more communication resources, computational power, and storage memory. Therefore, some research has been done to reduce communication costs [52]. Examples of HFL include wake-word detection (call voice assistants in smart homes by voice when saying a sentence but with different voices), next-word prediction, and recommendation systems.

#### Vertical Federated Learning

Vertical federated learning shares the same sample space but a different feature space. Only one client stores all the data labels, this client being a guest party or passive party client. Unlike the HFL, the guest client receives the model from the other clients (host party clients) and returns the value of the gradients to update the local models. The advantage of this system is to use fewer communication resources,

which depend on the number of data samples, ensuring the same performance. Some algorithms using VFL are SecureBoost or FedBCD. For example, this system cooperates in training a model with their datasets by hospitals and insurance companies can use or by banks together with e-commerce companies so that from transactions, they can estimate personalized loans based on online shopping behavior.

## Transfer Federated Learning

Transfer federated learning combines HFL and VFL. In this case, the sample and feature space are different, as shown in Figure 2.2c. In this system, the value of the different feature spaces is standardized for local training. With encryption, the system has to use a random mask to guarantee privacy in the gradient exchange. For example, TFL is used for disease diagnosis in which multiple patients from multiple hospitals and multiple countries (sample space) using multiple medications (feature space) are used. The objective is to standardize the data to enrich the model.

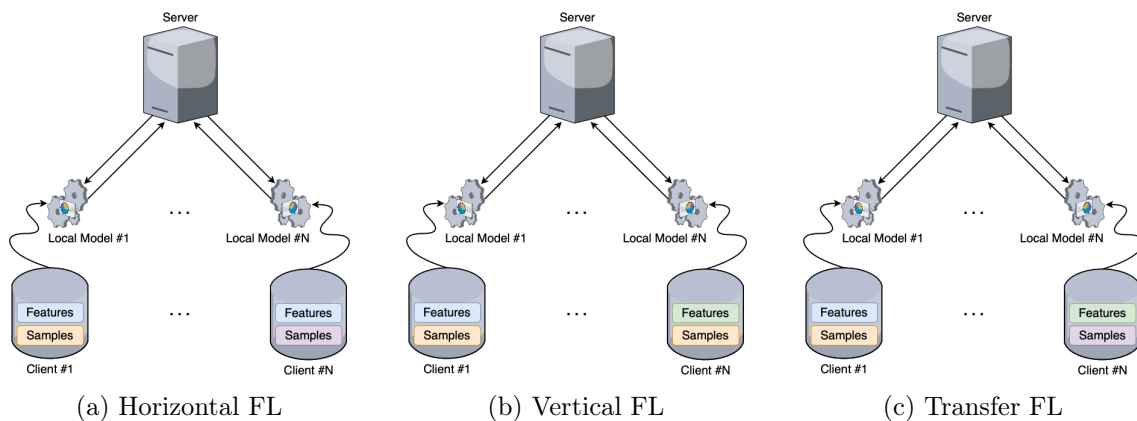


Figure 2.2: Data partitioning for federated learning

### 2.1.3 Federation Scale

This section discusses the distinction between cross-device (CD) and cross-silo (CS) systems, which are two ways of scaling the FL system [59]. This topic is essential for this work since smart grids are large distributed systems, and the system needs a federation scale that fits its goals. The main difference between these two types of systems is their scalability, which leads to various other differences in their characteristics. A summary of the federation scales for FL for both types of systems is provided in Table 2.1.

#### Cross-Device

Cross-device is defined by the case that there are many clients but a small amount of data for training due to the limited computational power. The clients also have more problems transmitting their models to an aggregation server, making them more susceptible to failures. For example, on smartphones, in the case of keyboard

suggestion, clients do not have many resources to train the model, but the server has the processing power.

## Cross-Silo

Cross-silo is defined by the small number of clients in the federation for processing a large amount of data. Consequently, they require more computational power. Unlike for cross-device, the clients in cross-silo are referenced. Therefore, it allows for better control of the clients to avoid failure scenarios. For example, the server would be a company that owns several data centers worldwide. In this case, each data center is a client.

Table 2.1: Summary of federation scale

Category	Subcategory	Cross-Device	Cross-Silo
Clients	Scalability	>100	<100
	Computational Power	Low	High
	Availability	Low	High
	Reliability	Low	High
	Indexing	✗	✓
Data Partitioning	HFL	✓	✓
	VFL	✗	✓
Example	-	IoT Device	Data Center

## 2.1.4 Aggregation Strategies for Federated Learning

The aggregation strategies are important since they define how the server aggregates the clients' models. Most of the strategies are applied to DL since it works with weights, but it is possible to adapt them into our context for smart grids. This section overviews the current state-of-the-art for aggregation strategies for FL.

### FedAvg

FedAvg is the first federated aggregation algorithm [81] proposed by the researchers who developed FL. It averages the model updates from multiple nodes in a FL setting. The approach is robust for unbalanced, non-independent, and identically distributed data distributions. Mainly used for deep neural network (DNN), the weights  $\vec{w}$  are updated as follows:

$$\vec{w} \leftarrow \vec{w} - \sum_{k=1}^K \frac{n_k}{n} \Delta \vec{w}_k \quad (2.3)$$

Where  $n_k$  is the number of examples for the client  $k$ , consequently:

$$n = \sum_{k=1}^K n_k \quad (2.4)$$

**FedAvgM**

FedAvgM (*Federated Averaging with Server Momentum*) is an extension of the FedAvg algorithm [51] that introduces momentum  $\beta$  to the model updates to improve the convergence rate. The weights are computed as follows:

$$\vec{w} \leftarrow \vec{w} - \vec{v} \quad (2.5)$$

Where  $\vec{v}$  is:

$$\vec{v} \leftarrow \beta \vec{v} + \sum_{k=1}^K \frac{n_k}{n} \Delta \vec{w}_k \quad (2.6)$$

**FedAsync**

FedAsync (*Asynchronous Federated Optimisation*) is a federated optimization algorithm [131] that uses an asynchronous approach to train models on non-independent and identically distributed data, improving flexibility and scalability. It allows the nodes to update their local models independently and asynchronously for a restricted family of non-convex problems. The strategy converges quickly and often outperforms synchronous optimization algorithms such as FedAvg.

**FedBCD**

FedBCD (*Federated Block Coordinate Descent*) is an aggregation algorithm [75] that performs several gradient updates in client nodes before communication with the server. It is designed for FL scenarios where the data is partitioned by feature rather than by sample (HFL). It allows parties to perform multiple local gradient updates, reducing communication overhead independently. Consequently, the communication cost decreases while maintaining the performance of the collaborative model.

**FedProx**

FedProx (*Federated Proximal Gradient Descent*) is a generalization of FedAvg [71]. It is a federated optimization algorithm that uses a proximal term to the local sub-problem to limit the impact of local variable updates. The new objective  $h_k$  to minimize is:

$$\min_w h_k(w) = F_k(w) + \frac{\mu}{2} \|w - w^t\|^2 \quad (2.7)$$

It improves the convergence and demonstrates significantly more stable and accurate convergence behavior relative to FedAvg.

## SCAFFOLD

SCAFFOLD (*Stochastic Controlled Averaging for Federated Learning*) is a federated aggregation algorithm [60] that uses a communication-efficient variant of stochastic gradient descent (SGD) to train models. It aims to reduce the communication rounds and is not affected by data heterogeneity, besides being faster than FedAvg. The update direction for the server model  $c$  and the update direction for each client  $c_i$  are estimated. The difference between these update directions is the following:

$$\Delta c_i = c - c_i \quad (2.8)$$

These differences are used to estimate the *client drift*, which measures how much the local model updates deviate from the global model updates. This estimation of the *client drift* is then used to correct the local updates, helping to improve the convergence of the model.

## FedOpt

FedOpt (*Adaptive Federated Optimization*) is a general strategy [103] for federated learning that is compatible with various adaptive optimizers as *AdaGrad* [105], *Yogi* [132], and *Adam* [62], resulting respectively in the following *FedAdaGrad*, *FedYogi*, *FedAdam*. Adaptive optimizers adjust their learning rates based on the knowledge of past iterations, which can improve the convergence of the model.

## FedAttOpt

FedAttOpt (*Federated Attentive Optimisation*) is a federated optimization algorithm [56] that uses attention mechanisms to weight the contribution of each client’s model parameters to the global model parameters on the central server. The attention score for each client model  $a_k$  is:

$$a_k = \sum_{j=1}^n \frac{e^{s_k}}{e^{s_j}} \quad (2.9)$$

It is calculated using a scaled dot-product  $s_k$  of the client model parameters  $\theta_k$  and the global model parameters:

$$s_k = \frac{\theta^T \theta_k}{\sqrt{n}} \quad (2.10)$$

The distance between the two sets of model parameters at time  $t$  is calculated using a p-norm distance metric:

$$d_k = \|\theta_t - \theta_{t+1}^k\|_p \quad (2.11)$$

The overall objective is to minimize the loss function, which is a weighted sum of

the distances between the global model and the client models, with the weights determined by the attention scores:

$$L = \sum_{k=1}^m a_k d_k \quad (2.12)$$

It performs better than FedAvg in terms of perplexity and communication cost.

### FedMA

FedMA (*Federated Matched Averaging*) is a layer-wise federated aggregation algorithm [125] for neural networks (NNs) such as convolutional neural networks (CNNs) and long short-term memories (LSTMs). FedMA gathers the weights of the first layer of the model from the clients and performs one-layer matching to obtain the first layer weights of the federated model. The server then broadcasts these weights to the clients, who train all consecutive layers on their datasets, keeping the matched federated layers frozen. This process is then repeated up to the last layer, for which a weighted averaging is performed based on the class proportions of data points per client. It reduces the variance of the model updates and improves the convergence rate of the model.

### SAFL

SAFL (*Simulated Annealing-based Federated Learning*) introduces the concept of simulated annealing [101] to the federated learning process. Simulated annealing is an optimization technique that allows a device to choose its local model with a high probability in the early stages of training when the global model is still immature. As training continues and the global model becomes more mature, the probability of using the local model decreases. Using simulated annealing, SAFL can improve the model's convergence speed and classification accuracy compared to FedAvg.

### FedMGDA+

FedMGDA+ (*Federated Multiple Gradient Descent Algorithm Plus*) aims to optimize multiple objectives [53], including fairness among users and robustness against malicious adversaries. FedMGDA+ is designed to be simple and has fewer hyperparameters to tune than other approaches. It is also guaranteed to converge to Pareto stationary solutions, which are optimal solutions that cannot be improved without sacrificing the performance of at least one of the participating users.

### FedBuff

FedBuff (*Federated Learning with Buffered Asynchronous Aggregation*) is a novel method [100] for FL introduced by *Meta AI* that aims to combine the best properties of both synchronous and asynchronous FL. It uses a buffer to store client

updates and aggregates them in a way that is agnostic to the optimizer’s choice. This allows FedBuff to scale more efficiently than synchronous FL while still being compatible with privacy-preserving technologies such as Differential Privacy and Secure Aggregation [12].

### 2.1.5 Privacy Mechanisms

Federated learning has privacy mechanisms that can be important for the smart grid context since it is a critical infrastructure. The three mechanisms presented in this section are the differential privacy (DP), the homomorphic encryption (HE), and the secure multiparty computation (SMC).

#### Differential Privacy

Differential privacy is a privacy-preserving technique used to protect sensitive personal data. It works by adding noise to the data, which makes it difficult to determine the specific values of any individual’s data. This noise helps protect each individual’s privacy while allowing the data to be used for statistical analysis. However, it is essential to note that adding noise to data can also negatively impact the quality of the statistical analysis. Therefore, balancing the trade-off between privacy protection and data quality is crucial when using DP [24, 30, 45].

#### Homomorphic Encryption

Homomorphic encryption is used to protect the privacy of the trained model by allowing the server to aggregate updates from multiple clients without learning the contents of the updates. HE can provide strong privacy guarantees in federated learning, as it ensures that the server cannot learn the contents of the updates. However, it can also introduce significant overhead in computation and communication, as the encrypted updates may be larger and more computationally expensive to work with than their unencrypted counterparts [32, 135].

#### Secure Multiparty Computation

Secure multiparty computation can secure client updates in the FL system. This is achieved by encrypting the client updates on the client side, such that the central server can only perform computations on the encrypted updates but cannot access the raw data. This ensures no information leakage from the client updates at the central server [21, 87].

### 2.1.6 Frameworks for Federated Learning

Several frameworks are available for federated learning, as listed in Table 2.2. These frameworks vary in compatibility with ML frameworks, scalability, data partition-

ing, communication protocols, and privacy considerations. Among these options, *Flower* stands out for its ongoing development and comprehensive documentation. Additionally, it is highly customizable and has a supportive community on *Slack* for addressing questions and providing suggestions.

Table 2.2: Summary of federated learning frameworks

Name	Year	Compatibility			Federation Scale		Nodes	Data Partitioning		Topology	Proto.	Priv.	Ref.
		<i>pytorch</i>	<i>tf</i>	<i>sklearn</i>	<i>CS</i>	<i>CD</i>		<i>HFL</i>	<i>VFL</i>				
TFF	2017	✗	✓	✗	✗	✓	-	✓	✗	Centralized	gRPC	DP HE	[1]
PySyft	2018	✓	✓	✗	✓	✓	-	✓	✓	Centralized	RPC WS HE	DP SMC	[137]
LEAF	2019	✗	✓	✗	✓	✓	-	✓	✗	Centralized	-	-	[17]
PaddleFL	2019	✗	✗	✗	✓	✓	>100	✓	✓	Centralized	gRPC ZMQ	DP HE	[78]
FL&DP	2020	✓	✓	✓	✓	✗	-	✓	✗	Centralized	-	DP	[104]
FedML	2020	✓	✗	✗	✓	✓	>100	✓	✓	Centralized Distributed Hierarchical	RPC MPI SP	-	[48]
FATE	2021	✓	✓	✓	✓	✓	-	✓	✓	Centralized	gRPC	DP HE SMC	[74]
OpenFed	2021	✓	✓	✗	✓	✓	-	✓	✗	Centralized Distributed Hierarchical	-	-	[23]
Flower	2022	✓	✓	✓	✓	✓	1000	✓	✗	Centralized	gRPC Ray	DP	[10]
FedLab	2022	✓	✗	✗	✓	✓	100	✓	✗	Centralized Distributed Hierarchical	-	-	[134]
OpenFL	2022	✓	✓	✗	✓	✗	-	✓	✓	Centralized	gRPC	-	[36]
FLUTE	2022	✓	✗	✗	✗	✓	50000	✓	✗	Centralized	Gloo NCCL	DP	[43]

## 2.1.7 Summary

This section presented different aspects of federated learning. First, the federated process is explained, where the local nodes have their own data and train a model aggregated at the upper nodes. Then the different data partitions, such as horizontal, vertical, and transfer federated learning, were presented. Afterward, two federation scales were presented, which are cross-device and cross-silo. Next, different aggregation strategies were presented, with FedAvg being the most popular. Thereafter, privacy mechanisms are presented, such as differential privacy, homographic encryption, and secure multiparty computation. Finally, different frameworks are presented with different characteristics.

The system developed in this work enables the simulation of the smart grid’s substations in a distributed way. In this work, the data used has the same features along the simulation, which means the simulation uses a horizontal partitioning of the data. The system is defined as cross-silo for the federation scale since the number of nodes is relatively small, needs availability, and must be reliable. Regarding the aggregation strategies, the simulation system will aggregate the node’s data instead of local models, as done in traditional federated learning. The framework used for the simulation environment is *Flower*.



## 2.2 Imbalanced Data

Data imbalance is a common issue in ML, which can be particularly challenging when working with anomaly detection tasks because the classification model suffers from bias, leading to inappropriate classification results. In anomaly detection tasks, which detect unexpected events in a system like a smart grid, the positive class (i.e., the anomalies) are often minoritarian towards the majority class. This can result in poor performance in the minority class and many classification errors. One important concept to understand when working with imbalanced data is the imbalance ratio (IR), which measures the proportion of the number of minority samples  $n_{\min}$  and the total number of samples  $n$  (Eq. 2.13).

$$IR = \frac{n_{\min}}{n} \quad (2.13)$$

Figure 2.3 shows examples of imbalanced data, where an IR of 50% corresponds to a balanced dataset (i.e., the number of samples of all classes are the same). On the other hand, the smaller the IR, the smaller the number of samples in the minority class.

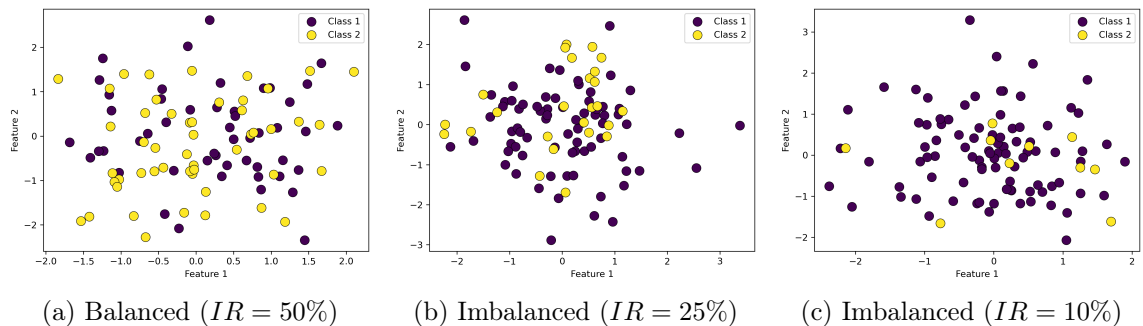


Figure 2.3: Examples of imbalanced data over 100 samples

The remainder of this section presents different distance metrics in Section 2.2.1. Then, the different types of examples of minority classes are explained in Section 2.2.2. Next, different methods of handling imbalanced data are described in Section 2.2.3. Finally, this section is summarized in Section 2.2.4.

### 2.2.1 Distance Metrics

Distance metric functions are essential in ML for supervised and unsupervised learning to compute the distance between points (instances) in data. Each distance metric has better characteristics for a specific task, such as classification or clustering. If two points are close, then they are similar. On the other side, they are different if the points are far. Distance metrics are essential for this work to compute the K-nearest neighbors (KNNs) to classify data [80, 106]. However, for this context, points can be characterized by continuous (i.e., lengths, temperatures) or nominal

(i.e., colors, categories) attributes. Consequently, exploring a metric that satisfies both attribute types is necessary.

### Euclidean Distance

The Euclidean distance (Eq. 2.14) is a well-known metric, namely to compare the similarity between measures using common properties, for instance, height, width, or depth of parts.

$$D_{euclidean}(\vec{x}, \vec{y}) = \sqrt{\sum_{a=1}^m (x_a - y_a)^2} \quad (2.14)$$

$\vec{x}$  and  $\vec{y}$  are two input vectors to be compared, and  $m$  is the number of attributes.

Note that if the square root is removed, another commonly used metric is obtained called Squared Euclidean distance since the closest point is still the nearest point with or without the squared root.

### Manhattan Distance

The Manhattan distance (Eq. 2.15), also called city-block distance, is preferred for attributes with different properties, for instance, age, weight, and gender of a patient.

$$D_{manhattan}(\vec{x}, \vec{y}) = \sum_{a=1}^m |x_a - y_a| \quad (2.15)$$

It requires less computation than the Euclidean distance since it does not compute the square root on each iteration.

### Minkowski Distance

The Minkowski distance (Eq. 2.16) is a generalization of Euclidean and Manhattan distances.

$$D_{minkowski}(\vec{x}, \vec{y}) = \left[ \sum_{a=1}^m |x_a - y_a|^r \right]^{1/r} \quad (2.16)$$

Where  $p$  is the distance of order between two points.  $p = 2$  corresponds to the Euclidean distance, and  $p = 1$  corresponds to the Manhattan distance.

### Chebychev Distance

The Chebychev distance (Eq. 2.17) is another specific case of the Minkowski distance when  $p \rightarrow +\infty$ . It results in the biggest attribute difference.

$$D_{chebychev}(\vec{x}, \vec{y}) = \max_{a=1}^m |x_a - y_a| \quad (2.17)$$

Similarly, if  $p \rightarrow -\infty$ , it results in the smallest attribute difference.

## Mahalanobis Distance

The Mahalanobis distance (Eq. 2.18) is a more complex metric, considering the covariance between dimensions, which makes the distance invariant to scaling operations.

$$D_{mahalanobis}(\vec{x}, \vec{y}) = \sqrt{(\vec{x} - \vec{y})^T C^{-1} (\vec{x} - \vec{y})} \quad (2.18)$$

$C$  is the covariance matrix, and consequently  $C^{-1}$  is its inverse matrix. Note that the symbol  $^T$  means the vector is transposed.

## HEOM Distance

The above-presented metrics are more suited for continuous attributes but not both simultaneously. The heterogeneous euclidean-overlap metric (HEOM) from Equation 2.19 tries to satisfy this issue, computing a normalized version of the Euclidean distance for continuous attributes and computing overlaps for nominal attributes [127].

$$D_{HEOM}(\vec{x}, \vec{y}) = \sqrt{\sum_{a=1}^m d_a^2(x_a, y_a)} \quad (2.19)$$

Where  $d_a(x_a, y_a)$  is:

$$d_a(x_a, y_a) = \begin{cases} 1 & \text{if } x_a \text{ or } y_a \text{ is undefined} \\ 0 & \text{if } a \text{ is nominal and } x_a = y_a \\ 1 & \text{if } a \text{ is nominal and } x_a \neq y_a \\ \frac{|x_a - y_a|}{\max(a) - \min(a)} & \text{otherwise} \end{cases} \quad (2.20)$$

Note that HEOM also supports missing data, attributing the value 1 to  $d_a(x_a, y_a)$ .

## HVDM Distance

The heterogeneous value difference metric (HVDM) from Equation 2.21 solves the limitations of HEOM, giving more weight to the nominal attributes [127].

$$D_{HVDM}(\vec{x}, \vec{y}) = \sqrt{\sum_{a=1}^m d_a^2(x_a, y_a)} \quad (2.21)$$

Where  $d_a(x_a, y_a)$  is:

$$d_a(x_a, y_a) = \begin{cases} 1 & \text{if } x_a \text{ or } y_a \text{ is undefined} \\ 0 & \text{if } x_a = y_a \\ 1 & \text{if } a \text{ is continuous and } \sigma_a = 0 \\ \frac{|x_a - y_a|}{4\sigma_a} & \text{if } a \text{ is continuous and } \sigma_a \neq 0 \\ \left[ \sum_{c=1}^C \left| \frac{N_{a,x,c}}{N_{a,x}} - \frac{N_{a,y,c}}{N_{a,y}} \right|^2 \right]^{1/2} & \text{otherwise} \end{cases} \quad (2.22)$$

Note that the standard deviation for  $a$  ( $\sigma_a$ ) has a degree of freedom equal to 1. Finally,  $N_{a,x,c}$  and  $N_{a,y,c}$  are the samples with respective values  $x$  and  $y$  for the attribute  $a$  and the class  $c$ . Consequently,  $N_{a,x}$ , for instance, is computed as in Equation 2.23.

$$N_{a,x} = \sum_{c=1}^C N_{a,x,c} \quad (2.23)$$

## 2.2.2 Types of Examples in Minority Classes

In an imbalanced data context, it is important to look at the minority class because there is the possibility of being surrounded by the noise of the majority class, decreasing classification performances. Napierala et al. [98] proposed a method for identifying minority class examples into four categories: safe (denoted as  $S$ ), rare (denoted as  $R$ ), borderline (denoted as  $B$ ), and outlier (denoted as  $O$ ). The minority class instances are labeled based on the proportion of their 5-nearest neighbors that belong to the same class using the HVDM distance from Section 2.2.1. These proportions permit to classify the instances with the following conditions:

- If there are 5/5 or 4/5 neighbors in the same class, the example is labeled as a **safe** example;
- If there are 3/5 or 2/5 neighbors in the same class, the example is labeled as a **borderline**. Their neighbors correctly classify these examples but may be close to the decision boundary between the classes;
- If 1/5 of the example's neighbors belongs to the same class, and:
  - If that neighbor also has 0 or 1/5 neighbors belonging to the same class. The example is labeled as **rare** example;
  - If the example has other neighbors from the same class in proximity, it is considered a **borderline** example instead.
- If none of the neighbors belongs to the same class, the example is labeled **outlier** (denoted as  $O$ ).

Adapting the thresholds can extend this method to more neighbors  $K$ . In the case of Figure 2.4,  $K = 5$  is chosen to classify the instances with the *ecoli* dataset from the *UCI Machine Learning Repository* [29] through the multidimensional scaling (MDS) visualization [25].

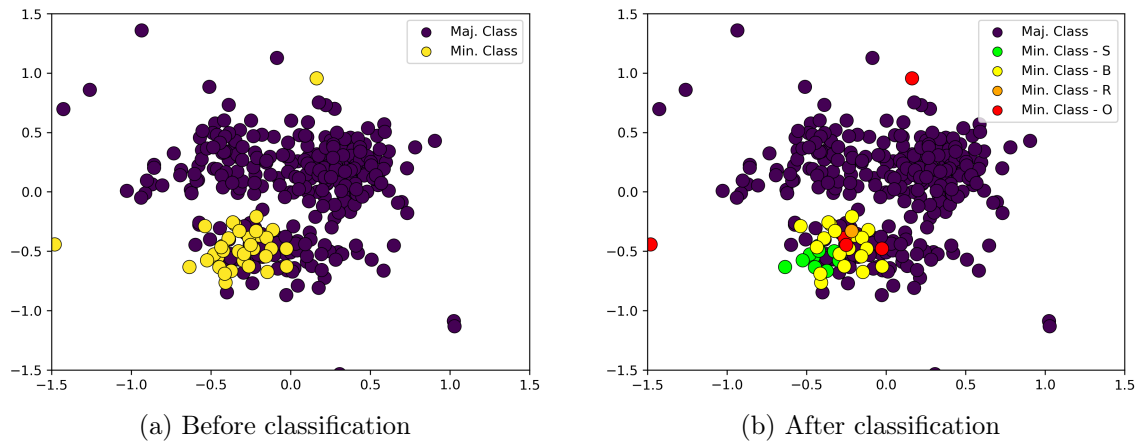


Figure 2.4: MDS visualizations of the minority classes over the *Ecoli* dataset

### 2.2.3 Imbalanced Data Processing Methods

There are different ways to deal with imbalanced datasets, such as undersampling, oversampling, or both. The methods above take into account the nearest neighbors to sample the data.

#### Neighborhood Cleaning Rule

Many works have been carried out to improve the classification of minority classes. For example, the neighborhood cleaning rule (NCR) [69] allows increasing the true positive rate (TPR) and true negative rate (TNR) up to 30% using algorithms such as KNN, with  $K = 3$ , or *C4.5*, an extension of the *ID3* algorithm. It uses the HVDM [127] to compute the KNN.

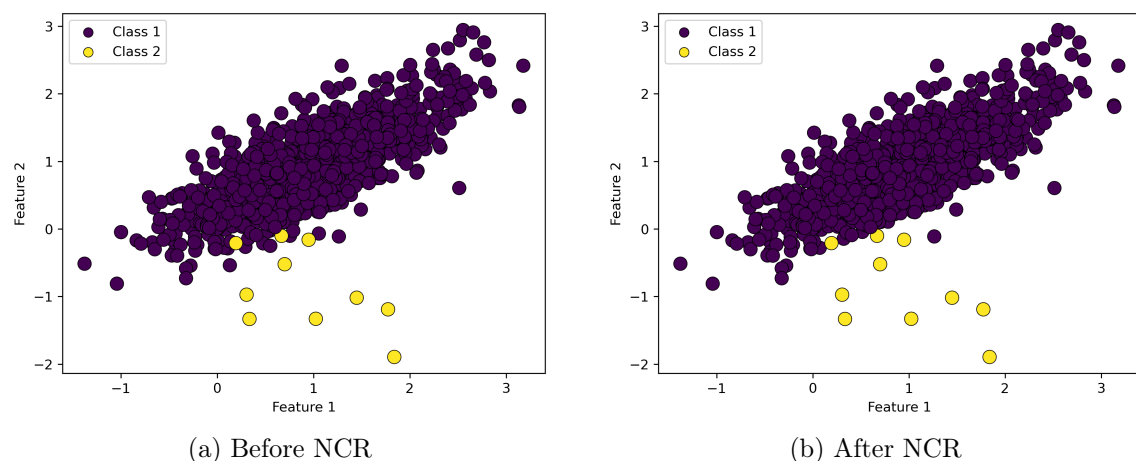


Figure 2.5: Neighborhood cleaning rule over 1000 samples with  $K = 3$

Figure 2.5 shows the difference before and after applying the NCR algorithm. We can see in Figure 2.5b that the minority class has fewer points than the majority class in its neighborhood.

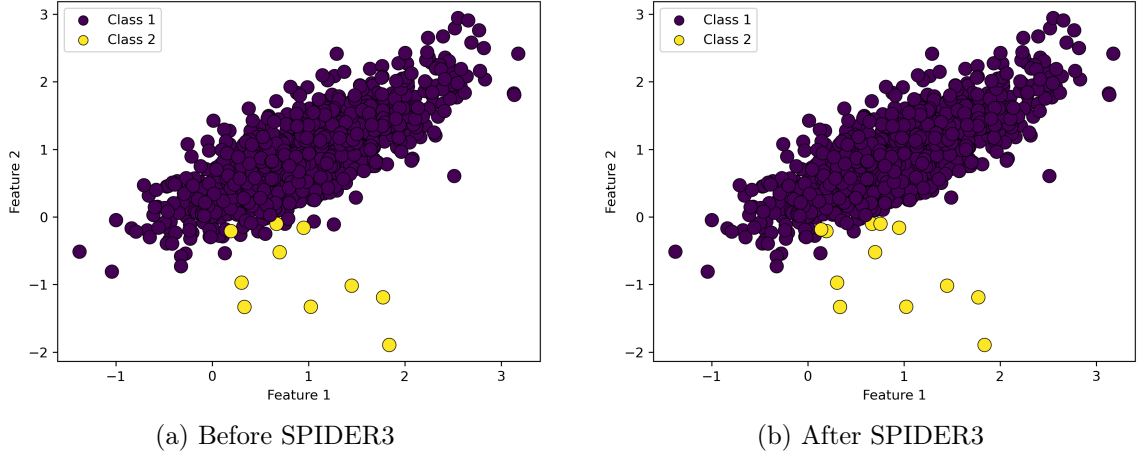
**SPIDER**

For handling imbalanced datasets, there is SPIDER [119], which stands for *Selective Preprocessing of Imbalanced Data*. The algorithm addresses this problem by selectively preprocessing the data to balance class distribution. The SPIDER algorithm involves two main phases:

- (1) In the first phase, each sample type (*safe* or *borderline*) is identified using KNN with HVDM. Safe examples are defined when the  $K$  neighbors of the same class are in their neighborhood. Otherwise, it will be a noisy example;
  
- (2) In the second phase, the data is preprocessed according to the flags assigned in the first phase. This involves modifying the minority class to distinguish it from the majority class. There are three techniques for preprocessing the data in the second phase:
  - **Weak amplification**, which involves creating copies of the noisy examples from the minority class and adding them to the data set;
  
  - **Weak amplification and relabeling**, which involve creating copies of the noisy examples from the minority class, as in weak amplification. Then, some of the noisy examples from the majority class are relabeled by changing their class from the majority class to the minority class;
  
  - **Strong amplification**, which involves creating copies of both the safe and the noisy examples from the minority class.

The second version of the algorithm, SPIDER2 [99], has a two-phase preprocessing, where the majority class is processed first, and the minority class is processed afterward. SPIDER2 includes additional options for preprocessing the data, including the ability to relabel noisy examples from the majority class as the minority class and three options for amplifying noisy examples from the minority class (*weak*, *strong*, and *no*).

The third version of the algorithm, SPIDER3 [129], is an extension of SPIDER2 that handles multiclass problems and controls the order of the preprocessing by considering the importance of specific decision classes. It reduces noise near the minority class and amplifies the examples near the border with the majority class (see Figure 2.6).

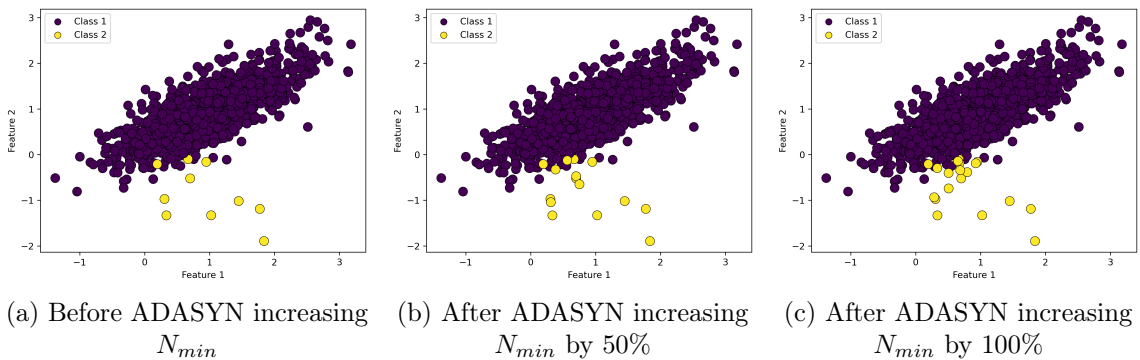
Figure 2.6: SPIDER3 over 1000 samples with  $K = 5$ 

## ADASYN

ADASYN algorithm, which stands for *Adaptive Synthetic* [46], creates minority class points near the border with the majority class (see Figure 2.7). The algorithm works as follows:

- (1) First, the algorithm finds the nearest neighbors for each minority class point, denoted as  $i$ ;
- (2) Then, it computes the proportion of majority class neighbors in its neighborhood as a weight, denoted as  $w_i$ ;
- (3) Afterward, for each point  $i$  with a weight  $w_i > 0$ , it will generate synthetic  $n$  examples between the point and its minority class neighborhood.

Note that the generated examples are proportional to the weight (the majority class examples in the neighborhood) of the currently computed neighbor.

Figure 2.7: ADASYN over 1000 samples with  $K = 5$

## SMOTE

SMOTE (*Synthetic Minority Oversampling TEchnique*) is another popular oversampling algorithm [22] capable of handling imbalanced datasets and creating synthetic minority class points near safe examples, increasing the proportion of these examples (see 2.8). The algorithm is modular with parameters like the number of samples to increase, the number of nearest neighbors to use, and the distance metric to compute the KNN. The process works as follows:

- (1) First, the algorithm randomly chooses one point of the minority class and computes its nearest neighbors;
- (2) Then, SMOTE computes the distance between the current point and one of its neighbors;
- (3) Afterward, a new point is generated between the two points;
- (4) Finally, the process is repeated until achieving the desired number of instances.

Note that SMOTE and the other algorithms should be applied only to the training data due to the generated synthetic data, which can introduce bias in classification results.

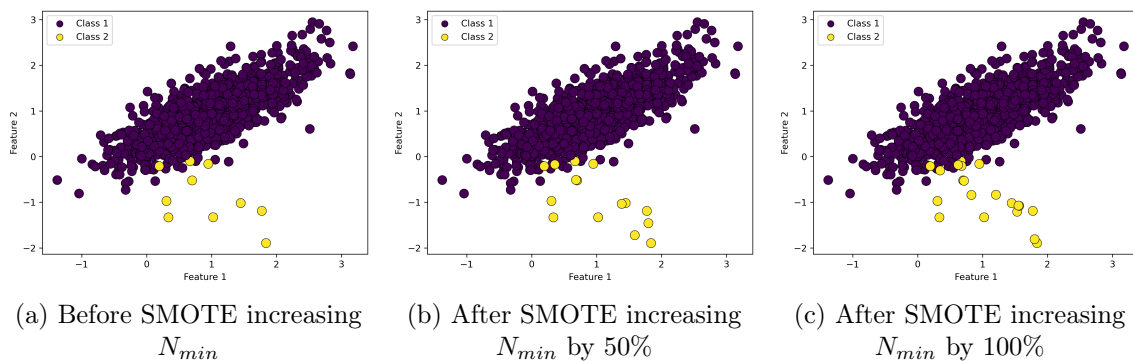


Figure 2.8: SMOTE over 1000 samples with  $K = 5$

### 2.2.4 Summary

In this section, the problem of imbalanced data was introduced. Then, different popular distance metrics were presented, such as Minkowski distance derivations and Mahalanobis distance. However, these metrics must be improved for continuous and nominal attributes simultaneously. The reason why heterogeneous metrics were presented, such as HEOM and HVDM. Afterward, a method for classifying minority class examples is presented, which uses HVDM distance. Finally, four algorithms that handle imbalanced datasets were explained: the neighborhood cleaning rule, SPIDER, ADASYN, and SMOTE.

The HVDM distance will be used in this work for computing the KNN for different datasets containing continuous and nominal attributes. This distance presents



better results than HEOM distance in Wilson et al. [127] experimentations. The SMOTE algorithm is used to handle the imbalanced data in this work due to its good performance in generating synthetic safe examples.

## 2.3 Causal Inference

Causal inference allows the understanding of complex behaviors in data, looking for information about how features in a dataset affect the prediction in a ML algorithm. SCMs defines how the data is represented in a causal context, enabling the prediction and estimation of what happens to the data in case of changes (called interventions). In addition, SCMs defines the consequences of these interventions (called counterfactuals), taking into account what happened [58]. The representation with SCMs expresses causality into different variables and connections between them, exemplified in Figure 2.9. These variables are divided into four categories:

- The **outcome**, denoted as  $y$ , is the target variable in data in which the causal effect is studied. More specifically, the causal effect of variables that cause changes in the outcome. The outcome is also called the *effect*;
- The **treatment**, denoted as  $t$ , is a feature in the data that directly impacts the target variable. The treatment is also called the *cause*;
- The **confounder** is a feature that influences both treatment and outcome, which can bias the causality effect. The link for this *common cause* to treatment and outcome is the *backdoor path*. If the confounder is an observed variable, it is denoted as  $x$ . Otherwise, the variable is an unobserved confounder denoted as  $z$ ;
- The **instrumental variable**, denoted as  $i$ , is a variable that is not directly correlated to the outcome. However, this variable is correlated to the treatment and can indirectly influence the outcome through the treatment.

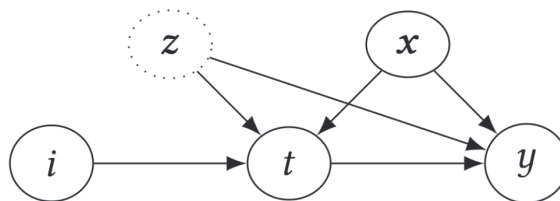


Figure 2.9: Causal graph presenting the different variables  
[Image from Brady Neal [16]]

In causal analysis, there are two types of data studies: the observational and the interventional. Observational studies, where the data is measured without intervention in the process. Interventional studies are where the data has been manipulated for a certain purpose. What happens is that observed data may have external factors, such as confounders, influencing a treatment. If the outcome under a given treatment differs from that under another treatment, then there is a causality effect [49]. In this work, data suffers interventions due to the injection of cyberattacks.

### 2.3.1 Causal Discovery

Causal discovery is a complex problem that involves discovering the structural causal models representing causality in a dataset. This SCMs is composed of directed acyclic graphs (DAGs) and structural equation models (SEMs).

- The **directed acyclic graphs** graphical representation of the connections between variables, also called causality diagrams, in which the nodes are variables, and the edges define the conditional probability between two variables.
- The **structural equation models** show the details of each edge between nodes, namely quantifying the influence that a variable has on another.

The discovery of the DAG can be made through an algorithm called *NOTEARS* that allows computing the relationship between nodes. For this, it calculates an adjacency matrix that aims to minimize a function penalizing the graph's cycles. The advantage of this algorithm is that it supports data with high dimensionality, and it does not need to make assumptions about the data's features relationship. This process is called *structure learning* since it learns the graph's structure from the data [136].

### 2.3.2 Causality Evaluation

The average treatment effect (ATE) is a measure that allows the representation of the causality in the data. It is calculated by the expected value of the difference between outcomes for each individual, also called individual treatment effect (ITE). The formula is presented in Equation 2.24.

$$\text{ATE} = \mathbb{E} [y_1 - y_0] \tag{2.24}$$

The ATE represents the change in the outcome variable if there is an intervention in treatments. The higher the value, the more the outcome is susceptible to change if the treatment has a modification.

#### Propensity Score

A propensity score is the probability that an instance receives a treatment given a set of confounders. This score serves, in particular, to compare treatments between them. The propensity score is calculated by training a logistic regression in which treatment is the dependent variable [7, 47, 54, 126].

- The **propensity matching** (Fig. 2.10a) estimates the ATE for treated individuals. This method joins pairs of instances where one has received a certain treatment and the other has not. The merge is done with instances whose propensity score is similar. A difference is made between treated and untreated. Finally, the ATE is the average of the results for each group;

- The **propensity-based stratification** (Fig. 2.10b) estimates the ATE dividing instances into subsets (usually 5) of identical size, in which the formed groups have similar propensity scores. The treated and untreated propensity scores are averaged within each group, and the difference between treated and untreated is made. Finally, the ATE is the average of the results for each group;
- The **inverse probability of treatment weighting** (Fig. 2.10c) estimates the ATE by attributing for each instance, the respective weight, which is calculated depending on its propensity score. With the weights, a weighted average is made for treated and untreated instances. Finally, the ATE is calculated by the difference between the two weighted means.

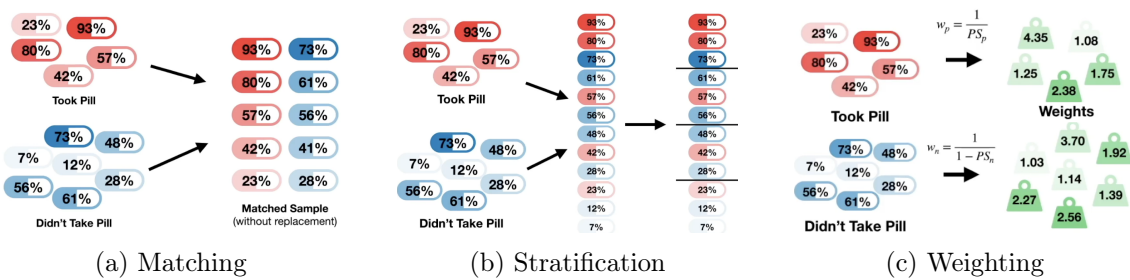


Figure 2.10: Different propensity score methods  
[Images from Shawhin Talebi [123]]

## Linear Regression

Linear regression allows us to find the line that best fits our population with the Equation 2.25.

$$y = mt + b \quad (2.25)$$

The coefficient  $b$  represents the factors that influence the treatment  $t$ , and the coefficient  $m$  is the slope of the regression, allowing the estimation of the expected value. The distance between the individual and the point in the regression is the ITE. Consequently, the ATE is the mean of the computed ITEs [16, 102].

### 2.3.3 Refutation Methods

Refutation methods are indispensable for SCMs assumptions validation. The ATE obtained with the initial model  $ATE_{init}$  is compared with the ATE obtained applying the refutation method  $ATE_{ref}$  defining a null hypothesis  $H_0$  [115].

#### Random Common Cause Refuter

The random common cause refuter generates random variables and performs the causality analysis again, comparing the results with the initial analysis. This means

that confounders are added to the data. In order not to reject the null hypothesis  $H_0$  from Equation 2.26, the  $ATE_{ref}$  cannot vary much from the initial  $ATE_{init}$ .

$$H_0 : ATE_{init} - ATE_{ref} = 0 \quad (2.26)$$

### Placebo Treatment Refuter

The placebo treatment refuter randomly defines a variable as a treatment. To not reject the null hypothesis  $H_0$  from Equation 2.27, the  $ATE_{ref}$  must approach zero.

$$H_0 : ATE_{ref} = 0 \quad (2.27)$$

### Data Subset Refuter

The data subset refuter divides the dataset into subsets  $s \in S$  as in cross-validation and measures the variation of the ATE between subsets. The null hypothesis  $H_0$  from Equation 2.28 is not rejected if the difference between subsets is minimal.

$$H_0 : \sum_{s \in S} ATE_{s,init} - ATE_{s,ref} = 0 \quad (2.28)$$

## 2.3.4 Summary

This section reviewed the basic concepts of causal inference, from causal discovery to causality evaluation. The ATE is the metric which measures causality and can be estimated through linear regression or propensity scores.

This work uses the ATE computed via linear regression since the propensity score does not support using multiple treatments simultaneously. The three presented refutation methods are used to try the model refutation, showing if the model obtained via the *NOTEARS* algorithm is viable.

## 2.4 Time-Series Classifiers

Smart Grids produce large amounts of data that can be employed for anomaly detection or optimization of the grid demand in energy. AMI allows the monitoring of the transmission line and the power system, producing time-dependent data.

Time-series classifiers are dedicated to areas using temporal data, such as speech recognition, financial analysis, or network traffic analysis [121]. Therefore, time-series classifiers (TSCs) perform better than traditional tabular algorithms since they use temporal window. Some of these time-series are presented in the following sections where Table 2.3 defines the notation. In this smart grid context, the data is composed of network traffic which uses timestamps to track packets over time.

Table 2.3: Notation for time-series classifiers

Variable	Meaning
$n$	number of time-series (attributes)
$m$	time-series length (instances)
$w$	window length
$r$	number of trees
$k$	number of intervals
$l$	word length
$c$	number of symbols
$q$	number of kernels

This section is organized as follows. The Section 2.4.1 presents shapelet-based TSCs. Then, in Section 2.4.2, the distance-based TSCs are shown. Afterward, in Section 2.4.3, the interval-based TSCs are discussed. Next, the dictionary-based are presented in Section 2.4.4 and the hybrid approaches in Section 2.4.5. Finally, an overview of this section is made in Section 2.4.6.

### 2.4.1 Shapelet-based

The shapelets are phase-independent subsequences of a time-series. A shapelet is found by sliding a window across the time-series, minimizing the Euclidean distance between each subsequence in the time-series and the shapelet.

#### Shapelet Transform

The shapelet transform classifier (STC) algorithm [50] is a pipeline classifier that searches the training data for shapelets and transforms the time-series into a vector of distances. It is a brute-force search algorithm that requires iterating through all subsequences in the dataset. This process has a training time complexity of  $\mathcal{O}(n^2m^4)$ .

#### ROCKET

ROCKET [26] stands for *RandOm Convolutional Kernel Transform*. It is a pipeline classifier using many random convolutional kernels that differ by length, weight, bias, dilation, and padding. With the convolution operation, these kernels extend the idea of shapelets applied to the time-series data. With the convolution, ROCKET extracts a diverse range of features concatenated into a feature vector. The feature vectors are then used to train a ridge classifier using cross-validation. The training operation has a time complexity of  $\mathcal{O}(qnm)$ . ROCKET has an optimized variant

called *MiniROCKET* [27], making the model more deterministic by removing random components.

## 2.4.2 Distance-based

Distance-based classifiers use specific time-series distance functions to measure the similarity between time-series as the basis of classification. A precursor in this domain is the dynamic time wrapping (DTW) algorithm [107] initially developed for speech recognition. It seeks for the temporal alignment between two time-series, which matches temporal indexes, such that euclidean distance between aligned series is minimized (Eq. 2.29).

$$DTW(\vec{x}, \vec{y}) = \min_{\pi \in \mathcal{A}(\vec{x}, \vec{y})} \sqrt{\sum_{(a,b) \in \pi} (x_a - y_b)^2} \quad (2.29)$$

Where  $x$  and  $x'$  are the two time-series to be compared.  $\pi$  is an alignment path of length  $m$ , composed by the index pairs  $\{(i_0, j_0), \dots, (i_{m-1}, j_{m-1})\}$ .  $\mathcal{A}$  is the set of all admissible paths.

### Elastic Ensemble

The elastic ensemble (EE) algorithm [72] is a weighted ensemble composed of eleven 1-nearest neighbor classifiers, each using a different distance metric. First, the following distances are computed:

- Euclidean distance;
- DTW using the full window;
- DTW with a restricted warping window;
- Weighted DTW;
- Derivative DTW using the full window;
- Derivative DTW with a restricted warping window;
- Weighted derivative DTW;
- Longest common subsequence;
- Edit distance with real penalty;
- Time warp edit distance;
- Move-split-merge.

After predicting the data with each model, the weights are updated, considering the performance of each classification. The process uses cross-validation to iterate the steps above and reach a final classification. The resulting training complexity is  $\mathcal{O}(n^2m^2)$ .

### Proximity Forest

The proximity forest (PF) algorithm [77] is an ensemble based on trees. The trees are randomly generated and use the same 11 distances functions used by EE to

compute the similarity between series. At each node, there are branches for each class in the data, where the child node receives randomly selected data from its parent, then the process is repeated recursively. If the node has only one class in its data, it becomes a leaf, and the iterative process stops. The resulting training complexity is  $\mathcal{O}(n \log nm^2)$ .

### 2.4.3 Interval-based

Interval-based classifiers divide the time-series into multiple distinct intervals. Then each interval trains an individual ML classifier.

#### Time-Series Forest

The time-series forest (TSF) algorithm [28] is an ensemble based on trees, which divides the time-series into  $\sqrt{m}$  intervals with random positions and lengths. For each interval, three metrics (mean, variance, and slope) are extracted and concatenated into a feature vector to build the tree. Then a subset of the features is selected at each node to compute the entropy gain and the margin that decides the tree's split into new nodes. This continues until the entropy gain stops increasing, defining the node as a leaf. The process has a training complexity of  $\mathcal{O}(rmn \log(n))$ .

#### Random Interval Spectral Ensemble

The random interval spectral ensemble (RISE) algorithm [73] is an interval-based tree ensemble that uses spectral features. The ensembles work similarly as TSF to build trees on random intervals from the data to construct a random forest classifier. However, RISE selects a single random interval for each base classifier. The process results in a  $\mathcal{O}(knm^2)$  training complexity.

#### Canonical Interval Forest

The canonical interval forest (CIF) algorithm [83] is another extension of TSF that improves its performance by integrating more features and by increasing diversity. Alongside the mean, standard deviation, and slope, the algorithm extracts the catch22 (*CAnonical Time-series CHaracteristics*) features [76], resulting in a total of 25 features, from which each tree selects a subset of them randomly. The process results in a training complexity of  $\mathcal{O}(rknm^{1.16})$ . The algorithm has another extension, diverse representation CIF (DrCIF) which incorporates two new series representations: the *periodograms* and *first order differences*. Phase-dependent intervals are randomly selected and concatenated into the feature vector for each of the three representations.

### 2.4.4 Dictionary-based

Classifiers use sliding windows and discretization techniques to find words in a time-series. Afterward, the distribution of these words is used to make the classification.

#### Bag-of-SFA-Symbols

The bag-of-SFA-symbols (BOSS) algorithm [109] is based on symbolic Fourier approximation (SFA) [110]. It transforms each time-series into discrete Fourier transform approximations. After that, these approximations are discretized with a technique called *multiple coefficients binning*. This preprocessing process results in a symbolic representation called *word* with the sliding window size for each time-series. Consecutive windows producing the same word are only counted as a single instance of the word. Then the nearest neighbor is computed to classify the instances. The process has a training complexity of  $\mathcal{O}(n^2m^2)$ . It has another variant called contractable BOSS (cBOSS) [82] with the possibility of defining a *time contract*. This reduces the training time. However, it decreases performance.

#### WEASEL

WEASEL stands for *Word ExtrAction for time SErie cLassification* algorithm [111]. The classifier computes the discrete Fourier transform for each series. Then statistical tests (ANOVA-F-test) are computed to separate time-series from different classes and create histograms with the words. Finally, the number of features is reduced using a chi-squared test, removing any words that score below a threshold. The logistic regression classifier is used to make the predictions. The resulting training complexity is  $\mathcal{O}(\min[nm^2, mc^{2l}])$ . A second version was developed (WEASEL v2) [112] using dilated windows which significantly improves performances compared to its predecessor.

### 2.4.5 Hybrid Approaches

Hybrid approaches are made from a combination of the previously seen algorithms. One example is HIVE-COTE, which stands for *HIerarchical Vote of COLlective Transformation-based Ensembles*. It is a highly performant algorithm that is declined in two versions (v1 and v2). Despite achieving high accuracy, HIVE-COTE is hugely computationally expensive and impractical to run in real-time, such as smart grids.

#### HIVE-COTE v1

HIVE-COTE version 1 [73] is constituted by a combination of five modules, from which EE, STC, RISE, TSF and BOSS. The algorithm works with a hierarchical voting structure, where each module has a weight and produces an estimate of the



probability of the class variable. The best-normalized probability is then chosen. It results in a training complexity bounded by the STC, resulting in  $\mathcal{O}(n^2m^4)$ .

## HIVE-COTE v2

HIVE-COTE version 2 [84] works similarly to its first version with the hierarchical voting structure. However, this version is constituted of new modules such as DrCIF, STC, temporal dictionary ensemble (TDE), and a group of ROCKET classifiers called *Arsenal*. The bottleneck is still the STC from which training time results in a complexity of  $\mathcal{O}(n^2m^4)$ .

Table 2.4: Complexity comparison between time-series classifiers

Type	Name	Year	Complexity	Ref.
Shapelet-based	STC	2014	$\mathcal{O}(n^2m^4)$	[50]
	ROCKET	2020	$\mathcal{O}(qnm)$	[26]
Distance-based	EE	2015	$\mathcal{O}(n^2m^2)$	[72]
	PF	2019	$\mathcal{O}(n \log nm^2)$	[77]
Interval-based	TSF	2013	$\mathcal{O}(rmn \log(n))$	[28]
	RISE	2018	$\mathcal{O}(knm^2)$	[73]
	CIF	2020	$\mathcal{O}(rknm^{1.16})$	[83]
Dictionary-based	BOSS	2015	$\mathcal{O}(n^2m^2)$	[109]
	WEASEL	2017	$\mathcal{O}(\min[nm^2, mc^{2l}])$	[111]
Hybrid	HIVE-COTE v1	2018	$\mathcal{O}(n^2m^4)$	[73]
	HIVE-COTE v2	2021	$\mathcal{O}(n^2m^4)$	[84]

### 2.4.6 Summary

In this section, different time-series classifiers were presented and are summarized in Table 2.4. The presented algorithms use different approaches, such as distance-based, interval-based, dictionary-based, and hybrid approaches. In this work, the fastest algorithms are compared, and the algorithm which presents better results in terms of training time, testing time, and classification performance is used for the remaining experiments.

## 2.5 Conclusions

This background knowledge chapter presents four important topics used in this work, such as federated learning, imbalanced data, causal inference, and time-series classification.

Federated learning is a decentralized machine learning approach allowing distributed data training. It consists of several phases, different data partitioning, and different aggregation strategies and can be applied at different scales. Federated learning also offers privacy benefits by allowing parties to keep their data on their devices. This work uses federated learning as a smart grid simulation through the *flower*

framework. The data is produced by local nodes aggregated to the upper nodes, enabling each node to train a different model with its data.

Imbalanced data is a challenge that affects machine learning algorithms when the data is skewed towards one class. This can lead to poor performance for the minority class. Several techniques have been proposed to address this issue, including the neighborhood cleaning rule, SPIDER, ADASYN, and SMOTE. Identifying minority class examples is essential to apply a good algorithm in the data. For this work, the methodology of minority class points classification is used to see in real-time the evolution of the minority class. Then the SMOTE algorithm is better for processing network data traffic mainly composed of noisy examples.

Causality is another challenge that affects the data. Sometimes a couple of features are correlated, but they are not always implying causation. A method for inferring the causal structure of the data is NOTEARS which constructs the DAG with few data examples. Then, it is possible to compute the average treatment effect of the data using propensity scores or linear regression. For this work, the causal effect is computed with a linear regression since propensity scores do not support multi-treatments, which is the case with network data traffic.

Time-series classification is important for anomaly detection since the data often has temporal components. It makes the classification more performant than usual machine learning algorithms. However, they have a bigger training complexity. For this work, the network traffic data is time-dependent, making using time-series classifiers relevant. The different algorithms are compared to choose the better one regarding speed and classification performance.

# Chapter 3

## Literature Review

Due to their interconnected and heterogeneous nature, some research has been conducted on using IDSs for smart grids. However, in the works conducted in this area, only a few explored the use of FL with real smart grid datasets. In most cases, the approaches use IoT or industrial internet of things (IIoT) datasets since it is more similar to the smart grid context.

This chapter will present relevant datasets in Section 3.1. They are used in most studies about intrusion detection systems applied to the smart grids in Section 3.2. Finally, Section 3.3 overviews the key points of this literature review.

### 3.1 Datasets for Intrusion Detection Systems

The datasets studied in this section are commonly used for research in IDS and anomaly detection. These datasets are categorized into three domains, such as network-based datasets in Section 3.1.1, IoT-based datasets in Section 3.1.2, and smart grid-based datasets in Section 3.1.3. These three sections are then summarized in Section 3.1.4.

#### 3.1.1 Network-based Datasets

*KDD99* was created in 1999 for a competition of intrusion detection. The dataset is simulated using transmission control protocol (TCP) packets. The dataset comprises several attack scenarios categorized into four main categories: DoS (i.e., smurf, neptune, land), user-to-root attack (i.e., buffer overflow, perl, rootkit), remote-to-local attacks (i.e., file transfer protocol (FTP) write, phf, internet message access protocol (IMAP)), and probing attacks (i.e., IP sweep, port sweep, nmap).

*NSL-KDD* was created in 2009. It aims to be an improved version of the *KDD99* dataset, removing redundant and duplicate instances. However, the dataset is in a binary scenario, distinguishing only if an attack exists.

*ISCX-IDS-2012* was created in 2012, reproducing seven days of real network activity,

where four of the seven days have attacks, such as DoS, distributed DoS (DDoS), network infiltration, and brute-force secure shell (SSH). However, the dataset is limited to one kind of attack per day. This results in a daily binary scenario.

*UNSW-NB15* was created in 2015, mixing real network traffic with synthetic data flows. The dataset comprises nine attack scenarios: fuzzers, analysis, backdoors, DoS, exploits, generic, reconnaissance, shellcode, and worms attacks.

*CIC-IDS-2017* was also created in 2017, which simulates an attack over five days into a specific network. The dataset is composed of eight attack scenarios, which are brute-force FTP, brute-force SSH, DoS, DDoS, heartbleed, Web-based, infiltration, and botnet attacks.

*CSE-CIC-IDS-2018* was created in 2018, and follows its predecessors *ISCX-IDS-2012*, and *CIC-IDS-2017*. This dataset is constructed on a large-scale testbed, hosted in *Amazon Web Services* cloud infrastructure, increasing the number of clients in networks. The dataset monitors the network traffic for ten days of different protocols such as hypertext transfer protocol (HTTP), TCP, user datagram protocol (UDP). The anomalies are divided into seven attack categories: brute-force, DoS, DDoS, botnet, port scanning, infiltration, and web attacks.

### 3.1.2 IoT-based Datasets

*Bot-IoT* was created in 2018, mixing real and simulated IoT network traffic. It aims to detect and identify botnets in IoT networks. It comprises four main attack categories: probing, DoS, DDoS, and information theft. This results in six attack scenarios.

*IoT-23* was created in 2020 and is a dataset of network traffic made from IoT devices. The dataset comprises nine attacks, including botnets, malwares, information gathering, theft, and DDoS attacks.

*TON-IoT* was created in 2021. It is a real federated dataset, constructed from IoT network traffic, using a testbed of three-level layers (edge, fog, and cloud). The dataset is composed of nine categories of attacks, such as DoS, DDoS, scanning, ransomware, backdoor, injection, cross-site scripting (XSS), password, and man-in-the-middle (MITM).

*Edge-IIoT* was created in 2022. It aims to be a dataset for centralized and federated learning, oriented to IIoT. The dataset monitors network traffic from different protocols such as such as internet control message protocol (ICMP), TCP, UDP. The anomalies are divided into five attack categories, which are DoS/DDoS, injection, MITM, malware, and information gathering attacks. It represents a total of 14 attack types.

### 3.1.3 Smart Grid-based Datasets

*IEC61850-Security* was created in 2019. This smart grid-dedicated dataset traces IEDs network communications in substations through GOOSE protocol. There are

nine scenarios divided into three categories (attack, disturbance, and normal). The four attacks are message suppression, data manipulation, DoS, and composite attack.

### 3.1.4 Summary

The data presented in Table 3.1 are datasets for anomaly detection and IDS. The table includes a summary of many datasets, each with unique characteristics. Each dataset includes the year it was created, the number of instances, features, classes (normal and attack scenarios), and the domain it pertains to. The domains include networks, IoT, IIoT, and smart grid.

Table 3.1: Comparison of datasets for intrusion detection systems

Domain	Name	Year	Instances	Features	Classes	Refs.
Networks	KDD99	1999	$\approx 5'200'000$	41	5	[120]
	NSL-KDD	2009	$\approx 5'200'000$	41	2	[37, 124]
	ISCX-IDS-2012	2012	$\approx 1'800'000$	8	5	[38, 116]
	UNSW-NB15	2015	$\approx 2'500'000$	48	10	[19, 92, 93, 94, 95, 108]
	CIC-IDS-2017	2017	$\approx 55'000'000$	79	9	[39, 113]
	CSE-CIC-IDS-2018	2018	$\approx 16'000'000$	79	7	[18, 40, 113]
Internet of Things	Bot-IoT	2018	$\approx 73'000'000$	29	7	[20, 63, 64, 65, 66, 67, 68]
	IoT-23	2020	$\approx 764'300'000$	21	10	[44]
	TON-IoT	2021	$\approx 22'000'000$	44	9	[4, 5, 14, 89, 90, 91, 96, 97]
	Edge-IIoT	2022	$\approx 21'000'000$	61	15	[35]
Smart Grids	IEC61850-Security	2019	$\approx 10'500$	28	9	[11]

## 3.2 Intrusion Detection Systems in Smart Grids

The state-of-the-art approaches can be divided into distributed and non-distributed scenarios for anomaly detection in IoT or Industrial IoT contexts like the smart grids. This section is organized as follows. Section 3.2.1 presents the non-distributed approaches for intrusion detection, while Section 3.2.2 presents distributed intrusion detection systems. Finally, an overview of this section is made in Section 3.2.3.

### 3.2.1 Non-Distributed Approaches

Taghavinejad et al. [122] used a non-distributed approach that involves a hybrid decision tree for IoT-based intrusion detection in smart grid, achieving 83.1% of accuracy with *NSL-KDD* dataset. Their approach surpasses the tested baselines, which use decision tree (DT), KNN, and support vector machine (SVM) algorithms.

Khan et al. [61] used a non-distributed configuration of four different particle swarm optimization (PSO) models. PSO was combined with DT, KNN, NN, and random

forest (RF). Their best results are PSO with NN that achieves 99.5% accuracy with *NSL-KDD* dataset and 99.2% accuracy with the *KDD99* dataset. Their approach was compared with other kinds of NN approaches.

Mohanty et al. [85] developed an intelligent intrusion detection system for smart grids with *NSL-KDD* dataset. This non-distributed approach uses a generative model with deep reinforcement learning called *CVAE-DDQN* (*Curiosity-driven Variational Auto-Encoder – Double Deep Q-Network*), which achieves 89.2% of accuracy. They also tested with a second dataset called *ISOT-CID* [3], achieving 98.2% accuracy.

Siniosoglou et al. [117] developed a model named MENSA, which stands for *anomaly dEtECTION aNd claSsificAtion*. It was proposed as a non-distributed approach for anomaly detection in a smart grid. It comprises an auto-encoder (AE) combined with a generative adversarial network (GAN) that achieves 96.5% accuracy in a *Modbus*-based network dataset [41]. Their approach was compared with logistic regression (LR), DT, SVMs, and DNNs,

### 3.2.2 Distributed Approaches

Mothukuri et al. [88] proposed a FL-based approach for anomaly detection for IoT devices using the above-mentioned *Modbus*-based network dataset. Their algorithm uses recurrent neural network (RNN) with gated recurrent units (GRUs), achieving 99.5% of accuracy in anomaly detection, a better result than their non-FL approach.

Jithish et al. [57] proposed a FL-based approach with *Flower* framework for anomaly detection in smart grids. The approach compares centralized and decentralized methods with LR, NN, AE, CNN, RNN, and GRU models. Their best model is the CNN with FL achieving 98.9% of accuracy using the *Ausgrid* dataset [6].

Li et al. [70] proposed *DeepFed* is another FL system that uses a CNN-GRU-based approach for intrusion detection in industrial cyber-physical systems. They achieve 99.2% accuracy with cyber-physical systems datasets [86]. Their approach was compared with other FL approaches.

Abdelkhalek et al. [2] proposed an anomaly detection system for DER communication in smart grids. Their testbed comprises a 2-tier architecture and reaches 98.47% of accuracy with a simple artificial NN on *CSE-CIC-IDS-2018* dataset. Other classifiers such as naive Bayes (NB), DT, RF, and SVM were tested with lower results.

Friha et al. [42] proposed a federated approach called 2DF-IDS, which stands for *Decentralized and Differentially Private Federated Learning-based Intrusion Detection System*, an intrusion detection system for industrial IoT. Their strategy is compared with centralized learning and focuses on the study of different levels of data privacy. The system uses DNN on *Edge-IIoT* dataset, achieving an accuracy of 94.3% for intrusion detection (without data privacy mechanisms).

### 3.2.3 Summary

To sum up, none of the few approaches presented for smart grids simulate the system with realistic smart grid architecture, despite using federated learning and their notable results. This is one of the motivations of the proposed smart grid simulation architecture presented in Chapter 4. On the other hand, the proposed approach in this work uses TSC with a deeper analysis of data imbalance and causality problems.

Table 3.2: Summary of research for intrusion detection systems

Approach	Year	Domain	ML Algorithm	FL	Dataset	Ref.
Non-Distributed	2020	Smart Grid	Triple-DT	✗	NSL-KDD	[122]
	2021	Smart Grid	PSO+{KNN, DT, RF, NN}	✗	KDD99 NSL-KDD	[61]
	2021	Smart Grid	DNN+AE	✗	NSL-KDD ISOT-CID	[85]
	2021	Smart Grid	DNN (AE+GAN)	✗,	Modbus	[117]
Distributed	2021	IoT	CNN+GRU	✓	CPS	[70]
	2022	Smart Grid	NB, DT, RF, SVM, NN	✗	CSE-CIC-IDS-2018	[2]
	2022	IoT	RNN+GRU	✓	Modbus	[88]
	2023	IoT	DNN	✓	Edge-IIoT	[42]
	2023	Smart Grid	LR, NN, AE, CNN, RNN, GRU	✓	Ausgrid	[57]

## 3.3 Conclusions

In summary, research on using IDS for smart grid has focused on using various ML and DL algorithms to detect and classify cyberattacks in smart grid. One study used DT classifiers. Another used a combination of an auto-encoder and generative adversarial network architecture, and another compared several classifiers combined with a PSO algorithm. In addition, traditional classifiers such as NB, DT, RF, SVM, and a simple NN approach have been tested in various datasets. These studies have shown that these IDS methods can effectively detect and classify cyberattacks in smart grid environments, particularly the ones that use a FL approach. These studies mainly used the datasets presented in Table 3.1, which describes diverse scenarios of anomalies in network traffic in distributed and non-distributed contexts.





# Chapter 4

## Experimental Setup

The primary goal of this work is to develop an efficient intrusion detection system for smart grids using federated learning to simulate the grid architecture. To do so, experiments are divided into two phases:

- The **first phase** is divided into two experiments, where the first consists in testing different time-series classifiers, and the second consists in inferring the structural causal model from datasets' data;
- The **second phase** is divided into four experiments, which test different cyberattack scenarios and compares our data processing strategy with a baseline that does not process the data.

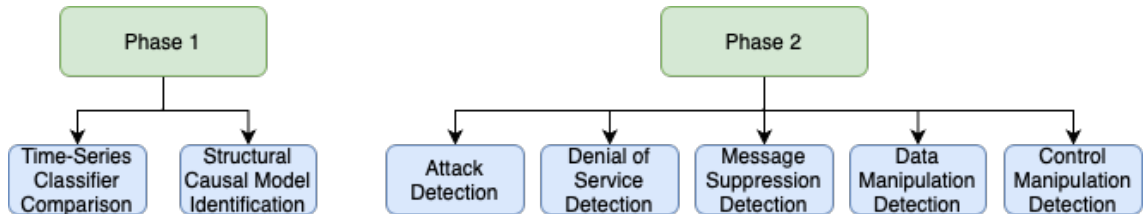


Figure 4.1: Experimental phases

In the first phase, the experimentation begins with evaluating time-series classifiers using 5-fold cross-validation. The following classifiers are pre-selected due to their low training complexity for evaluation:

- CBOSS;
- CIF;
- MiniROCKET
- RISE
- ROCKET;
- TSF;
- WEASEL version 2.

The second experiment of this first phase uses the *NOTEARS* algorithm to define which features are considered confounders or treatments. Then, the structural

causal model is inferred, defining the relationships between features. Then, the results are validated with refutation methods such as random common cause, placebo treatment, and data subset.

The second phase of experimentation consists in detecting attacks by alternating the injection of attacks every ten rounds, as shown in Figure 4.2. Rounds are communication cycles in the federated system.



Figure 4.2: Attack injection strategy

There are two scenarios of anomaly detection in this work:

- (1) **Attack detection**, which only detects if there is an attack in the system without categorizing it;
- (2) **Specific attack detection**, where only one specific attack is injected in the system (i.e., DoS, message suppression, data manipulation, or control manipulation attack).

The results are then evaluated with the metrics explained and compared with a baseline method, which does not apply our data processing strategy.

The chapter is organized as follows. Section 4.1 describes the datasets used for the experimentations and the data collection process. Then, Section 4.2 explains our data processing strategy. Afterward, Section 4.3 presents the metrics used for evaluation. Subsequently, Section 4.4 presents the smart grid testbed. Finally, Section 4.5 overviews the whole chapter.

## 4.1 Dataset Collection

In the first phase of our experimental study, several datasets commonly used in anomaly detection were used to evaluate the time-series classifiers. The datasets employed in this phase are the following:

- BOT-IoT
- CIC-IDS-2017
- CSE-CIC-IDS-2018
- Edge-IIoT
- IEC61850-Security
- IoT-23
- ISCX-IDS-2012
- KDD99
- NSL-KDD
- TON-IoT
- UNSW-NB15

These datasets provide a diverse range of attack scenarios described in Table 4.1, and their features are described in Appendix A.

Table 4.1: Description of attacks per dataset

Dataset	Instances	Attacks ( <i>number</i> )
BOT-IoT	3668522	DDoS (1926624), DoS (1650260), Reconnaissance (91082), Normal (477), Theft (79)
CIC-IDS-2017	2830743	Normal (2273097), DoS-Hulk (231073), Port Scanning (158930), DDoS (128027), DoS-GoldenEye (10293), FTP-Patator (7938), SSH-Patator (5897), DoS-Slowloris (5796), DoS SlowHTTPTest (5499), Botnet (1966), BruteForce (1507), XSS (652), Infiltration (36), SQL injection (21), Heartbleed (11)
CSE-CIC-IDS-2018	1000000	Normal (469045), DDoS-LOIC-HTTP (99918), FTP-BruteForce (99885), DDoS-HOIC (96064), DoS-SlowHTTPTestt (91434), Botnet (80182), DoS-GoldenEye (41508), DoS-Slowloris (10990), DoS-Hulk (8300), DDoS-LOIC-UDP (1730), BruteForce-Web (611), BruteForce-XSS (230), SQL Injection (87)
Edge-IIoT	157798	Normal (24301), DDoS-UDP (14498), DDoS-ICMP (14088), Ransomware (10925), DDoS-HTTP (10561), SQL injection (10311), Uploading (10269), DDoS-TCP (10247), Backdoor (10195), Vulnerability scanner (10076), Port Scanning (10071), XSS (10052), Password (9989), MITM (1214), Fingerprinting (1001)
IEC61850-Security	4558	Message Suppression (1580), Normal (1446), Data Manipulation (813), DoS (372), Control Manipulation (347)
IoT-23	1444706	Port Scanning (825931), Okiru (262687), Normal (197834), DDoS (138775), C&C (15107), Attack (3915), C&C Heartbeat (351), C&C File Download (47), C&C Torii (30), File Download (17), C&C Heartbeat File Download (11), C&C Mirai (1)
ISCX-IDS-2012	2071657	Normal (2002747), Attack (68910)
KDD99	4898431	Smurf (2807886), Neptune (1072017), Normal (972781), Satan (15892), IP sweep (12481), Port sweep (10413), Nmap (2316), Backdoor (2203), Warezclient (1020), Teardrop (979), Pod (264), Password (53), Buffer overflow (30), Land (21), Warezmaster (20), IMAP (12), Rootkit (10), Load module (9), FTP-write (8), Multihop (7), Phf (4), Perl (3), Spy (2)
NSL-KDD	148517	Normal (77054), Attack (71463)
TON-IoT	461043	Normal (300000), Port scanning (20000), DoS (20000), SQL injection (20000), DDoS (20000), Password (20000), XSS (20000), Ransomware (20000), Backdoor (20000), MITM (1043)
UNSW-NB15	257673	Normal (93000), Generic (58871), Exploits (44525), Fuzzers (24246), DoS (16353), Reconnaissance (13987), Analysis (2677), Backdoor (2329), Shellcode (1511), Worms (174)

In the second experimental phase, the focus shifted to the smart grid-oriented dataset *IEC61850-Security*, composed of DoS flooding, message suppression, data manipulation, and control manipulation attacks. Other datasets were used to test attack detection, such as *NSL-KDD*, or *BOT-IoT* and *UNSW-NB15* for DoS flooding over domain name system (DNS) service.

For all experimental phases, there is a data collection process consisting of two main steps: network packet capture extraction for raw datasets and dataset loading. These steps were crucial for acquiring the necessary data to conduct our experiments.

The step of **network packet captures extraction** is essential since two datasets comprise raw data: *IEC61850-Security* and *IoT-23* datasets. This raw data is presented into *PCAP* files, which can be analyzed with specific software like *Wireshark* [128]. Hopefully, there is a software called *Zeek* [133], which can read those

*PCAP* files and extract protocols communication variables, such as source and destination IP addresses, port numbers, the protocol used, information about the packets, and the timestamp of the communication. All these variables are used as features for this work.

```
1 $ zeek -C -r file.pcap
```

Listing 4.1: Extraction of packet captures with *Zeek*

The extraction is done via a bash command, needing only a *PCAP* file as input (*file.pcap*) as shown in Listing 4.1. On the other hand, the software outputs a *CSV* file with the extracted data. The "-r" flag tells *Zeek* where to find the trace of interest, and the "-C" flag tells *Zeek* to ignore any TCP checksum errors.

The second step, for all datasets, consists in **preparing the data for preprocessing**. First, the data is loaded into *DataFrames*, setting the data timestamp as an index and sorting them by timestamps. This is necessary since the datasets are divided into multiple *CSV* files organized by attack types. Other datasets have both raw and prepared *CSV* datasets, such as *Edge-IIoT* and *TON-IoT* datasets. To avoid the loading of unneeded data, the prepared version of these two datasets is used for the experiments. The advantage is that they have fewer instances than the original datasets presented in Table 3.1, and consequently, easier to manipulate. Note that all the datasets used in this work contain many instances, implying a folder size with more than 100GB of storage for all datasets combined. The *IoT-23* has heavy *CSV* files, which are difficult to be used with time-series classifiers due to long time processing. For this reason, the dataset is limited to 100'000 instances per *CSV* file read. Then, it is important to look at null variables, which are represented differently across the datasets, for instance, with a "-" or with a "(empty)" flag. Finally, columns that are redundant with another column are dropped. For instance, *UNSW-NB15* has a column indicating the attack type and a second indication if the instance is an attack. In this case, the most specific column is chosen. Then, if necessary, the column is converted into a binary scenario.

## 4.2 Proposed Approach

The data on each node is preprocessed to replace values like strings into numerical values through an ordinal encoder. Then, the data is normalized with values between  $-1$  and  $1$ . Since this work does not focus on missing data, the undefined values are replaced by the median value of the feature.

The experimental data processing consists of classifying the minority class examples, oversampling the minority classes, and computing the causality of the data as shown in Figure 4.3.

Then the minority classes are classified via the algorithm proposed by Napierala et al. [98]. The implementation used in this work is described in detail in Algorithm 2, which uses the HVDM distance of Algorithm 1, computed with the

distance  $d_a$  from Equation 2.22. The datasets from the *UCI Machine Learning Repository* [29] were used to validate the implementations with the paper’s results.

---

**Algorithm 1** HVDM distance

---

```

1: procedure HVDM(data)
2:    $n, m \leftarrow \text{shape}(\textit{data})$ 
3:    $\textit{matrix} \leftarrow \text{zeros}(n, n)$ 
4:   for  $x \leftarrow 0$  to  $n - 1$  do
5:     for  $y \leftarrow x + 1$  to  $n$  do
6:        $d \leftarrow 0$ 
7:       for  $a \leftarrow \text{attribute in } \textit{data}$  do
8:          $d \leftarrow d + d_a^2(\textit{data}[x, a], \textit{data}[y, a])$ 
9:          $\textit{matrix}[x, y] \leftarrow \textit{matrix}[y, x] \leftarrow \sqrt{d}$ 
10:  return  $\textit{matrix}$ 

```

---



---

**Algorithm 2** Classification of minority class examples

---

```

1: procedure TAXONOMY( $\textit{matrix}, \textit{data\_target}, K$ )
2:    $S \leftarrow B \leftarrow R \leftarrow O \leftarrow 0$ 
3:    $\textit{knn} \leftarrow \text{KNN}(\textit{matrix}, \textit{data\_target}, K)$ 
4:   for  $i \leftarrow \text{neighbour in } \textit{knn}$  do
5:      $\textit{nn}_i \leftarrow \text{nearest neighbours of the minority class for } i$ 
6:      $\textit{sum}_i \leftarrow \text{length}(\textit{nn}_i)$ 
7:     if  $\textit{sum}_i \geq \lfloor 0.8 \cdot K \rfloor$  then
8:        $S \leftarrow S + 1$ 
9:     else if  $\textit{sum}_i \geq \lfloor 0.5 \cdot K \rfloor$  then
10:       $B \leftarrow B + 1$ 
11:    else if  $\textit{sum}_i \geq \lfloor 0.2 \cdot K \rfloor$  then
12:       $j \leftarrow \text{nearest neighbour of the minority class of } i$ 
13:       $\textit{nn}_j \leftarrow \text{nearest neighbours of the minority class for } j$ 
14:       $\textit{sum}_j \leftarrow \text{length}(\textit{nn}_j)$ 
15:      if  $\textit{sum}_j = 0$  then
16:         $R \leftarrow R + 1$ 
17:      else if  $\textit{sum}_j = 1$  and  $\textit{nn}_i$  is  $\textit{nn}_j$  then
18:         $R \leftarrow R + 1$ 
19:      else
20:         $B \leftarrow B + 1$ 
21:    else
22:       $O \leftarrow O + 1$ 
23:  return  $S, B, R, O$ 

```

---

The minority classes are oversampled with the SMOTE algorithm by increasing their number by 30% each, helping the system increase the percentage of safe and borderline examples.

Finally, the causality is computed through a linear regression since propensity scores cannot deal with multiple treatments. The moving average is also computed for a sliding window of size seven. The ATE is compared between the data with 10% fewer examples and the moving average. If the data with fewer samples have an ATE smaller than the moving average, the original data continues into the system. Otherwise, these are chosen by the system.

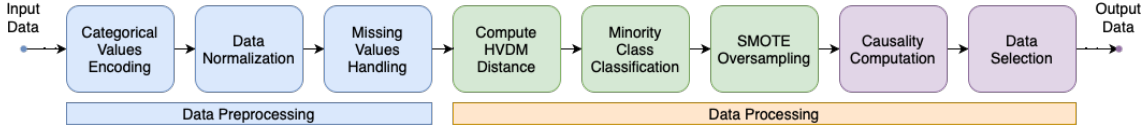


Figure 4.3: Preprocessing and processing flow

### 4.3 Evaluation Metrics

Experimental results are mainly expressed with accuracy (Eq. 4.1) for binary classification, but precision (Eq. 4.2), recall (Eq. 4.3) and F1-score (Eq. 4.4) are also computed:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \quad (4.1)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (4.2)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (4.3)$$

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4.4)$$

Where  $TP$  corresponds to the true positives (the number of positive examples correctly classified),  $FP$  denotes the false positives (the number of positive examples wrongly classified),  $TN$  corresponds to the true negatives (the number of negative examples correctly classified), and  $FN$  denotes the false negatives (the number of negatives examples wrongly classified).

Since this work mainly uses imbalanced datasets, macro-metrics are preferred instead of micro-metrics. macro-metrics give each class equal importance, while micro-metrics gives equal importance to each instance. The metrics mentioned above are computed for each class  $c$ , but instead, the Macro Average (Eq. 4.5) is computed by taking the arithmetic mean of the chosen metric  $M$ .

$$M_{macro\_avg} = \frac{1}{N} \sum_{c \in C} M_c \quad (4.5)$$

Where  $M \in \{\text{Accuracy}, \text{Precision}, \text{Recall}, \text{F1-Score}\}$ ,  $N$  is the number of classes,  $C$  is the set containing all classes, and  $M_c$  is the metric for the class  $c$ .

Receiver operating characteristic (ROC) curves compare the experimental approach with the baseline. For that, it is necessary to compute the TPR (Eq. 4.6), also known as Sensitivity (note that it is the same formula as Recall), and the false positive rate (FPR), which is the inverse of the Specificity (Eq. 4.7).

$$TPR = \text{Sensitivity} = \text{Recall} = \frac{TP}{TP + FN} \quad (4.6)$$

$$FPR = 1 - \text{Specificity} = \frac{FP}{FP + TN} \quad (4.7)$$

## 4.4 Federated System

The system for the experiments is constructed through federated learning, which allows having a distributed system simulating the smart grid substations.

The implemented system consists of a three-layer hierarchical architecture as shown in Figure 4.4. The system uses the *Flower* framework presented in Section 2.1.6 for the federated communication between the system's nodes. The framework allows the accurate distribution of the clients and servers into physical or virtual machines, defining real IP addresses from each node.

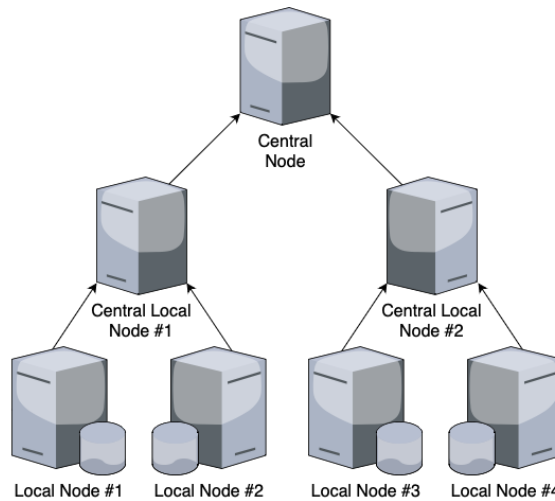


Figure 4.4: Representation of the three-level architecture

The monitoring system generates the data, which sends each second data via MQTT protocol. The data is formatted into a *JSON* to send column names as a key. Even though the data is generated externally, the local nodes begin the federation with a small amount of data to avoid problems with the system's initialization.

```
[{ "id.orig_h": "fe80::c806:7e6d:928b:8e0d", "id.orig_p": 143, "id.resp_h": "ff02::16",
↪ "id.resp_p": 0, "proto": "icmp", "service": null, "duration": 137.856511, "orig_bytes": 400.0,
↪ "resp_bytes": 0.0, "conn_state": "OTH", "local_orig": null, "local_resp": null,
↪ "missed_bytes": 0, "history": null, "orig_pkts": 16, "orig_ip_bytes": 1296, "resp_pkts": 0,
↪ "resp_ip_bytes": 0, "tunnel_parents": null, "label": 1 }]
```

Listing 4.2: Generated data sent via MQTT protocol

- The low-voltage or **LV substation** is the local node at the system's bottom, which transmits low-voltage power to our homes. The dataset's data feed

the LV substation with a stream injecting a new instance of the dataset each second into the system. In this work, the LV substations are organized in pairs for each distribution substation due to computational limitations. In a realistic case, more substations are linked to the distribution substation.

- The **distribution substation** is the central local node that simultaneously acts as a server and a client. The server part receives and aggregates the data from LV substations. On the other hand, the client part connects to the HV substation and sends its data. This substation handles high-voltage power to the LV substations.
- The high-voltage or **HV substation** provides a natural location for deploying the central node, being at the system's top level and acting as a server, receiving the distribution substation's data. The HV substation is located near the power plant, transforming extra high-voltage into high-voltage. Alternatively, the topmost layer could also be co-located with the grid dispatch center (or its computing infrastructure point of presence locations), which controls grid operations, thus constituting a potential strategic deployment point.

The monitoring system presents in real-time the evolution of the proportion of minority class examples. This data is published in real-time by nodes from the federated system through the MQTT broker, as shown in Figure 4.5. The monitoring system is subscribed to the MQTT broker to receive all information relative to the metrics of each node of the system, such as the proportion of safe, borderline, rare, and outlier examples, accuracy, precision, recall, F1-score, ATE, and the moving average of the ATE. Despite being shown in real-time, this information is also stored in a *CSV* file for each round.

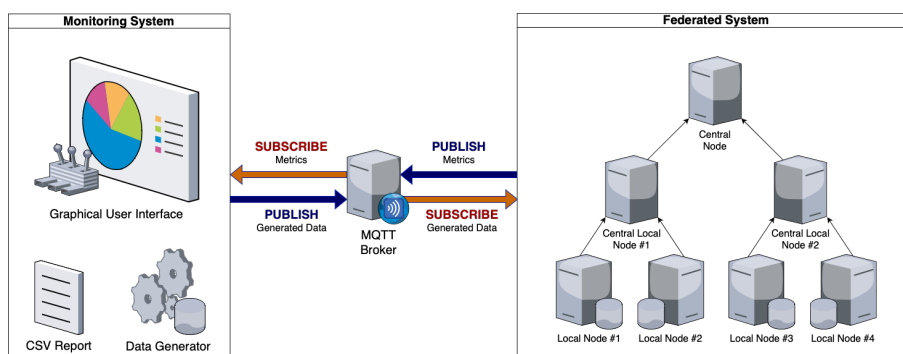


Figure 4.5: Representation of the whole system

The second role of the monitoring system is to publish data from the dataset database to the MQTT broker, which is read by the local nodes of the federated system. The data is picked from the dataset by its timestamp and published to the broker ordered.



## 4.5 Conclusions

This chapter presented the experiments in two phases, the first testing the time-series classifiers and the structural causal model for each dataset. Then, our data processing approach, which uses data imbalance and causality, was presented. This processing approach will be compared with a baseline in the second experimental phase, which only applies the classifier to the data. More concretely, the second phase detects the injected attacks with the two approaches.

Afterward, this chapter presented the data collection methods, such as raw data preparation with *Zeek* software, which extracts network traffic in *PCAP* files into features as a *CSV* file. Then, the datasets are preprocessed to work with a well-formatted *DataFrame*.

The last part of the chapter presents the smart grid testbed, with a three-level architecture through the *Flower* framework, distinguishing three types of nodes: local, central local, and central nodes. These nodes represent the LV substation, the distribution substation, and the HV substation. These nodes act as clients and servers to enable federated communication hierarchically. The federated system is controlled by a GUI, which shows the data in real-time the metrics and generates reports from them. The monitoring system also generates the data sent to the federated system. All the communications between the two systems are done by the MQTT protocol, which works in a Subscribe-Publish manner. Then, the experimental processing of the data is done by oversampling the minority class with SMOTE algorithm and looking for the causality of the data through the average treatment effect.



# Chapter 5

## Results and Discussion

The experiments performed in this work are presented and discussed in this chapter. As illustrated in the previous chapter, the experiments are separated into two phases:

- The first phase in Section 5.1 consists in gaining sensibility to the performance of time-series classifiers and structural causal models in federated learning contexts;
- The second phase in Section 5.2 tests different attack scenarios to compare our proposed data processing method with the baseline, which only classifies with the time-series classifier.

### 5.1 First Experimental Phase

The process of choosing an efficient classifier is essential for anomaly detection. This section is organized as follows. Section 5.1.1, all the datasets presented in Section 3.1 were chosen to assess a selection of classifiers. Then, Section 5.1.2 tests the structural causal model of these datasets and validates with three different refutation methods. Finally, Section 5.1.3 overviews the key findings for this first experimental phase.

#### 5.1.1 Time-Series Classifier Selection

This experiment consists in searching for a fast and efficient time-series classifier. For that, datasets were sampled with 10'000 samples, conserving class proportion distribution. This threshold was defined due to memory limitations on heavier algorithms. A few models were tested, such as *cBOSS*, *CIF*, *MiniROCKET*, *RISE*, *ROCKET*, *TSF*, and *WEASEL-v2*. Other algorithms were excluded since they take more than one day to train. The results for the binary scenario are in Table 5.1 (full tables are available in Appendix B).

Regarding training and testing time, *TSF* performs the classification process faster. Considering classification results, all classifiers present globally similar performances, where we can conclude that *TSF* is more adequate for real-time classification since

it is the best trade-off in terms of time and performance. For some datasets, *TSF* presents no difficulty in detecting attacks except on two datasets (*IEC61850-Security* and *UNW-NB15*), which seems more difficult to classify.

Table 5.1: Time-series classification in binary scenario

Dataset	Model	Train Time [s]	Test Time [s]	Accuracy	Precision	Recall	F1-Score
BOT-IoT	CIF	1011.44 ± 10.09	44.98 ± 5.49	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	MiniROCKET	83.32 ± 1.24	2.68 ± 0.07	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	RISE	93.09 ± 2.27	18.72 ± 0.5	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	ROCKET	210.4 ± 0.98	18.09 ± 0.85	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	TSF	8.48 ± 0.4	2.89 ± 0.16	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	WEASEL-v2	265.55 ± 5.41	13.83 ± 0.13	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	cBOSS	244.35 ± 1.36	207.68 ± 1.4	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	IEC61850-Security	CIF	295.58 ± 4.39	370.4 ± 17.06	0.73 ± 0.01	0.73 ± 0.01	0.95 ± 0.02
MiniROCKET		17.63 ± 0.43	0.72 ± 0.02	0.78 ± 0.01	0.8 ± 0.01	0.91 ± 0.01	0.85 ± 0.01
RISE		45.51 ± 0.84	7.97 ± 0.13	0.76 ± 0.01	0.8 ± 0.01	0.86 ± 0.01	0.83 ± 0.01
ROCKET		46.77 ± 1.01	3.44 ± 0.11	0.77 ± 0.01	0.79 ± 0.01	0.9 ± 0.01	0.84 ± 0.01
TSF		4.56 ± 0.49	1.64 ± 0.42	0.77 ± 0.01	0.81 ± 0.01	0.86 ± 0.02	0.83 ± 0.01
WEASEL-v2		75.34 ± 0.71	3.53 ± 0.13	0.72 ± 0.02	0.77 ± 0.01	0.86 ± 0.02	0.81 ± 0.01
cBOSS		21.92 ± 0.41	78.93 ± 0.11	0.75 ± 0.01	0.77 ± 0.01	0.9 ± 0.01	0.83 ± 0.01
NSL-KDD		CIF	1219.44 ± 14.56	754.62 ± 16.63	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	MiniROCKET	112.91 ± 1.64	2.87 ± 0.31	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	RISE	133.79 ± 2.9	18.5 ± 0.25	0.99 ± 0.0	0.99 ± 0.0	0.98 ± 0.0	0.99 ± 0.0
	ROCKET	245.0 ± 3.14	16.76 ± 0.92	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	TSF	11.05 ± 0.71	2.97 ± 0.31	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	WEASEL-v2	309.22 ± 6.22	13.64 ± 0.47	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	cBOSS	229.07 ± 1.09	232.35 ± 0.57	0.98 ± 0.0	0.98 ± 0.01	0.99 ± 0.0	0.98 ± 0.0
	UNSW-NB15	CIF	1255.72 ± 34.07	1966.39 ± 38.18	0.93 ± 0.01	0.94 ± 0.0	0.95 ± 0.01
MiniROCKET		103.44 ± 1.1	2.79 ± 0.16	0.92 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.01
RISE		127.81 ± 1.67	18.48 ± 0.33	0.92 ± 0.0	0.92 ± 0.0	0.96 ± 0.01	0.94 ± 0.0
ROCKET		232.56 ± 2.33	16.61 ± 0.58	0.93 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.0
TSF		12.59 ± 0.36	3.03 ± 0.23	0.93 ± 0.0	0.94 ± 0.01	0.95 ± 0.01	0.95 ± 0.0
WEASEL-v2		292.77 ± 6.08	13.08 ± 0.61	0.92 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.0
cBOSS		297.98 ± 3.84	234.07 ± 0.98	0.91 ± 0.0	0.93 ± 0.01	0.93 ± 0.0	0.93 ± 0.0

## 5.1.2 Structural Causal Model Discovery

This experiment consists in discovering the dependency between dataset variables inferred by the *NOTEARS* algorithm from *CausalNex* [9]. The DAG is constructed by removing edges with a weak relation. *NOTEARS* needs only a few examples to learn the structure of the model and is not supposed to evolve with more examples. Consequently, having the structural causal model precomputed avoids computational overhead in the federated system for the second experimental phase. The treatment variables for the dataset used in the second experimental phase are described in Table 5.2. Afterward, the structural causal models are validated by trying to refute the null hypothesis  $H_0$  for each refutation method using the *DoWhy* [114]. The results are presented in Table 5.3. All datasets present p-values greater than the significance level of 0.05, concluding that the refutation hypothesis cannot be rejected.

Table 5.2: Treatment variables inferred from *NOTEARS*

Dataset	Treatment Variables
BOT-IoT	dbytes, stime, pkts, ltime, tnp-pdstip, tnp-per-dport, ar-p-prot-p-dstip, drate, daddr, spkts, ar-p-prot-p-dport, ar-p-prot-p-srcip, dpkts, proto, mean, flgs-number, pkts-p-state-p-prot-p-destip, stddev, min, state, sport, dport, tnp-perproto, max, ar-p-prot-p-sport, n-in-conn-p-srcip, n-in-conn-p-dstip, pkts-p-state-p-prot-p-srcip, rate, flgs, dur, tnp-pscip, saddr, state-number, srate, proto-number, sum
IEC61850-Security	orig-bytes
NSL-KDD	land, srv-error-rate, dst-host-diff-srv-rate, dst-host-srv-diff-host-rate, dst-host-srv-error-rate, diff-srv-rate, src-bytes, dst-host-srv-error-rate, urgent, num-failed-logins, dst-host-error-rate, dst-bytes, is-guest-login, error-rate, num-shells, srv-error-rate, su-attempted, root-shell, srv-diff-host-rate, num-access-files, num-file-creations, wrong-fragment, error-rate, dst-host-serror-rate
UNSW-NB15	ct-src-ltm, dttl, service, trans-depth, ct-dst-src-ltm, state, dinpkt, spkts, djit, smean, response-body-len, ct-ftp-cmd, ct-srv-dst, synack, dloss, sload, tcprrt, ct-src-dport-ltm, is-sm-ips-ports, sjit, swin, dpkts, ct-flw-http-mthd, sloss, ct-srv-src, dtcpb, dwin, ct-dst-ltm, stepb, proto, ackdat, dmean, sttl, sinpkt, dur, ct-state-ttl, ct-dst-sport-ltm, is-ftp-login

Table 5.3: Refutation of structural causal models

Dataset	Refuter	$ATE_{mit}$	$ATE_{ref}$	Result
BOT-IoT	Random Common Cause	-0.8058	-0.8058	No Significant Difference
	Placebo Treatment	-0.8058	0	No Significant Difference
	Data Subset	-0.8058	-0.9378	No Significant Difference
IEC61850-Security	Random Common Cause	-0.0135	-0.0137	No Significant Difference
	Placebo Treatment	-0.0135	0	No Significant Difference
	Data Subset	-0.0135	-0.0053	No Significant Difference
NSL-KDD	Random Common Cause	3.2465	3.2465	No Significant Difference
	Placebo Treatment	3.2465	0	No Significant Difference
	Data Subset	3.2465	3.0592	No Significant Difference
UNSW-NB15	Random Common Cause	0.2823	0.2812	No Significant Difference
	Placebo Treatment	0.2823	0	No Significant Difference
	Data Subset	0.2823	0.2323	No Significant Difference

### 5.1.3 Summary

This section discussed the results from the preliminary experimentations. The best classifier found is the TSF which offers good results compared to the other classifiers but is extremely fast to train and test. The three refutation methods do not reject the structural causal models, which means they are acceptable.

## 5.2 Second Experimental Phase

This section presents a different evaluation of attacks in the proposed federated system. Section 5.2.1 presents the obtained results in attack detection attacks. Section 5.2.2 discusses the results relative to DoS attack detection. Then, the other smart grid attacks are tested and discussed in Section 5.2.3. Finally, key findings are summarized in Section 5.2.4.

## 5.2.1 Attack Detection

The results for attack detection were assessed for *IEC61850-Security* and *NSL-KDD* datasets. *IEC61850-Security* is composed by 68/32 of attacks and normal instances. On the other hand, *NSL-KDD* is more balanced with 52/48 of attacks and normal instances. As expected, in Table 5.4, *IEC61850-Security* has more difficulties in detecting attacks than *NSL-KDD* due to the different proportion of classes. This table also shows that our approach surpasses the baseline approach for each metric.

Table 5.4: Performance in detecting attacks

Dataset	Round	Strategy	Safe [%]	Borderline [%]	Accuracy	Recall	Precision	F1-Score	AUC
IEC61850-Security	1-25	baseline	27.55 ± 10.33	53.13 ± 7.67	0.64 ± 0.08	0.63 ± 0.09	0.63 ± 0.09	0.62 ± 0.09	0.63 ± 0.09
		ours	31.25 ± 8.53	52.58 ± 8.36	0.67 ± 0.09	0.67 ± 0.09	0.67 ± 0.09	0.66 ± 0.09	0.67 ± 0.09
	1-50	baseline	30.12 ± 8.68	52.69 ± 6.39	0.65 ± 0.1	0.64 ± 0.1	0.64 ± 0.1	0.63 ± 0.1	0.64 ± 0.1
		ours	30.36 ± 7.24	55.88 ± 8.69	0.66 ± 0.09	0.66 ± 0.1	0.66 ± 0.1	0.66 ± 0.1	0.66 ± 0.1
	1-75	baseline	29.25 ± 8.0	54.34 ± 7.02	0.64 ± 0.11	0.63 ± 0.11	0.64 ± 0.11	0.63 ± 0.11	0.63 ± 0.11
		ours	27.42 ± 8.05	59.4 ± 9.41	0.65 ± 0.11	0.65 ± 0.11	0.65 ± 0.11	0.64 ± 0.11	0.65 ± 0.11
	1-100	baseline	26.65 ± 8.47	57.66 ± 8.57	0.63 ± 0.12	0.63 ± 0.12	0.63 ± 0.12	0.63 ± 0.12	0.63 ± 0.12
		ours	26.58 ± 7.7	59.55 ± 8.58	0.65 ± 0.11	0.65 ± 0.12	0.65 ± 0.12	0.64 ± 0.12	0.65 ± 0.12
NSL-KDD	1-25	baseline	80.21 ± 4.78	7.45 ± 3.5	0.93 ± 0.05	0.92 ± 0.06	0.93 ± 0.05	0.92 ± 0.05	0.92 ± 0.06
		ours	86.28 ± 6.23	8.06 ± 4.18	0.96 ± 0.04	0.95 ± 0.05	0.96 ± 0.04	0.95 ± 0.04	0.95 ± 0.05
	1-50	baseline	79.74 ± 4.59	9.02 ± 3.6	0.94 ± 0.04	0.93 ± 0.05	0.94 ± 0.04	0.93 ± 0.05	0.93 ± 0.05
		ours	88.2 ± 5.51	7.55 ± 3.55	0.97 ± 0.03	0.96 ± 0.04	0.97 ± 0.03	0.96 ± 0.04	0.96 ± 0.04
	1-75	baseline	80.83 ± 4.79	9.91 ± 3.66	0.94 ± 0.04	0.94 ± 0.05	0.95 ± 0.04	0.94 ± 0.04	0.94 ± 0.05
		ours	89.71 ± 5.04	6.89 ± 3.1	0.97 ± 0.03	0.97 ± 0.03	0.97 ± 0.03	0.97 ± 0.03	0.97 ± 0.03
	1-100	baseline	82.63 ± 5.34	9.31 ± 3.53	0.95 ± 0.04	0.95 ± 0.04	0.95 ± 0.04	0.95 ± 0.04	0.95 ± 0.04
		ours	90.52 ± 4.62	6.52 ± 2.79	0.97 ± 0.03	0.97 ± 0.03	0.97 ± 0.03	0.97 ± 0.03	0.97 ± 0.03

Statistical tests are performed to confirm if there is a significant difference between the two approaches. In Table 5.5, the normal distribution is checked with the Kolmogorov-Smirnov test for both datasets. The table shows no scenario with data following a normal distribution, so a non-parametrical test is chosen, such as the Wilcoxon signed-rank test.

Table 5.5: Kolmogorov-Smirnov normality test for attack scenarios

Dataset	Metric	Strategy	Statistic	p-value	Result
IEC61850-Security	Accuracy	baseline	0.475	2.80e-21	Not Normally Distributed
		ours	0.459	9.49e-20	Not Normally Distributed
	Precision	baseline	0.434	1.22e-17	Not Normally Distributed
		ours	0.451	4.51e-19	Not Normally Distributed
	Recall	baseline	0.420	1.70e-16	Not Normally Distributed
		ours	0.455	1.84e-19	Not Normally Distributed
	F1-Score	baseline	0.419	2.01e-16	Not Normally Distributed
		ours	0.452	3.36e-19	Not Normally Distributed
	AUC	baseline	0.420	1.70e-16	Not Normally Distributed
		ours	0.455	1.84e-19	Not Normally Distributed
NSL-KDD	Accuracy	baseline	0.492	7.07e-23	Not Normally Distributed
		ours	0.569	3.51e-31	Not Normally Distributed
	Precision	baseline	0.484	4.82e-22	Not Normally Distributed
		ours	0.581	1.24e-32	Not Normally Distributed
	Recall	baseline	0.492	8.18e-23	Not Normally Distributed
		ours	0.621	9.70e-38	Not Normally Distributed
	F1-Score	baseline	0.490	1.08e-22	Not Normally Distributed
		ours	0.614	8.93e-37	Not Normally Distributed
	AUC	baseline	0.492	8.18e-23	Not Normally Distributed
		ours	0.621	9.70e-38	Not Normally Distributed

The results of the Wilcoxon test in Table 5.6 show that in all cases, the hypothesis that there is no significant difference is rejected ( $p < 0.05$ ). Therefore, it is possible to conclude that our approach performs better than the baseline for both datasets.

Table 5.6: Wilcoxon signed-rank test for attack scenarios

Dataset	Metric	Statistic	p-value	Result	Effect Size
IEC61850-Security	Accuracy	1240.0	9.95e-06	Significant Difference	0.442
	Precision	1191.0	4.50e-06	Significant Difference	0.459
	Recall	1150.0	2.27e-06	Significant Difference	0.473
	F1-Score	1128.0	1.56e-06	Significant Difference	0.480
NSL-KDD	AUC	1150.0	2.27e-06	Significant Difference	0.473
	Accuracy	0.0	3.90e-18	Significant Difference	0.868
	Precision	18.0	6.70e-18	Significant Difference	0.862
	Recall	8.0	4.96e-18	Significant Difference	0.865
NSL-KDD	F1-Score	0.0	3.90e-18	Significant Difference	0.868
	AUC	8.0	4.96e-18	Significant Difference	0.865

## 5.2.2 Denial of Service Detection

The results for DoS detection were assessed for *BOT-IoT*, *IEC61850-Security* and *UNSW-NB15* datasets, which use DNS flooding. *BOT-IoT* is composed by 95/5 of DoS attacks and normal instances, *IEC61850-Security* is composed by 80/20 of DoS attacks and normal instances, *UNSW-NB15* is more unbalanced with 15/85 of attacks and normal instances. In Table 5.7, *BOT-IoT* easily detects DoS due to its large number of attack examples, and we cannot compare the two approaches on this dataset. Overall, the reminder datasets present better results with our approach, except the accuracy in *IEC61850-Security* dataset, which is better with the baseline.

Table 5.7: Performance in detecting DoS attacks

Dataset	Round	Strategy	Safe [%]	Borderline [%]	Accuracy	Recall	Precision	F1-Score	AUC
BOT-IoT	1-25	baseline	99.43 ± 3.59	0.57 ± 3.59	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
		ours	99.17 ± 3.83	0.61 ± 3.79	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	1-50	baseline	99.67 ± 2.56	0.33 ± 2.56	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
		ours	99.19 ± 2.73	0.41 ± 2.7	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	1-75	baseline	99.78 ± 2.09	0.22 ± 2.09	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
		ours	99.16 ± 2.26	0.41 ± 2.22	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	1-100	baseline	99.83 ± 1.81	0.17 ± 1.81	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
		ours	99.23 ± 1.97	0.34 ± 1.93	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
IEC61850-Security	1-25	baseline	1.48 ± 2.58	29.23 ± 13.51	0.77 ± 0.09	0.58 ± 0.1	0.61 ± 0.15	0.57 ± 0.11	0.58 ± 0.1
		ours	5.97 ± 6.39	49.05 ± 12.97	0.77 ± 0.09	0.63 ± 0.1	0.66 ± 0.13	0.63 ± 0.1	0.63 ± 0.1
	1-50	baseline	2.57 ± 3.62	42.33 ± 16.55	0.74 ± 0.12	0.61 ± 0.13	0.63 ± 0.15	0.61 ± 0.14	0.61 ± 0.13
		ours	8.13 ± 6.68	55.46 ± 11.91	0.73 ± 0.11	0.64 ± 0.11	0.66 ± 0.13	0.64 ± 0.11	0.64 ± 0.11
	1-75	baseline	5.08 ± 5.72	47.32 ± 15.61	0.72 ± 0.13	0.63 ± 0.14	0.64 ± 0.16	0.63 ± 0.15	0.63 ± 0.14
		ours	11.23 ± 7.25	58.72 ± 11.15	0.71 ± 0.12	0.65 ± 0.12	0.66 ± 0.13	0.64 ± 0.12	0.65 ± 0.12
	1-100	baseline	7.18 ± 6.43	51.55 ± 15.55	0.71 ± 0.14	0.64 ± 0.14	0.65 ± 0.16	0.63 ± 0.15	0.64 ± 0.14
		ours	14.06 ± 8.09	60.84 ± 10.42	0.69 ± 0.12	0.64 ± 0.12	0.66 ± 0.13	0.64 ± 0.12	0.64 ± 0.12
UNSW-NB15	1-25	baseline	59.83 ± 20.37	20.5 ± 13.11	0.95 ± 0.04	0.85 ± 0.13	0.91 ± 0.12	0.87 ± 0.12	0.85 ± 0.13
		ours	66.18 ± 18.8	19.04 ± 14.58	0.96 ± 0.03	0.88 ± 0.11	0.94 ± 0.11	0.9 ± 0.11	0.88 ± 0.11
	1-50	baseline	70.76 ± 18.25	15.15 ± 11.09	0.96 ± 0.03	0.89 ± 0.1	0.94 ± 0.09	0.91 ± 0.1	0.89 ± 0.1
		ours	75.85 ± 16.68	14.55 ± 11.49	0.96 ± 0.03	0.92 ± 0.09	0.95 ± 0.08	0.93 ± 0.08	0.92 ± 0.09
	1-75	baseline	75.59 ± 16.55	13.21 ± 9.56	0.96 ± 0.03	0.91 ± 0.09	0.95 ± 0.07	0.93 ± 0.08	0.91 ± 0.09
		ours	80.76 ± 15.38	12.18 ± 10.07	0.97 ± 0.03	0.94 ± 0.08	0.96 ± 0.07	0.94 ± 0.07	0.94 ± 0.08
	1-100	baseline	79.02 ± 15.52	11.81 ± 8.66	0.96 ± 0.03	0.93 ± 0.08	0.96 ± 0.07	0.94 ± 0.08	0.93 ± 0.08
		ours	83.8 ± 14.36	10.6 ± 9.19	0.97 ± 0.02	0.95 ± 0.07	0.97 ± 0.06	0.95 ± 0.06	0.95 ± 0.07

Statistical tests are performed to confirm if there is a significant difference between the two approaches. In Table 5.8, the normal distribution is checked with the

Kolmogorov-Smirnov test for the two datasets presenting differences between the two approaches. The table shows no scenario with data following a normal distribution, so a non-parametrical test is chosen, such as the Wilcoxon signed-rank test.

Table 5.8: Kolmogorov-Smirnov normality test for DoS attack scenarios

Dataset	Metric	Strategy	Statistic	p-value	Result
IEC61850-Security	Accuracy	baseline	0.548	9.30e-29	Not Normally Distributed
		ours	0.535	2.85e-27	Not Normally Distributed
	Precision	baseline	0.453	3.06e-19	Not Normally Distributed
		ours	0.488	1.84e-22	Not Normally Distributed
	Recall	baseline	0.505	3.75e-24	Not Normally Distributed
		ours	0.424	7.52e-17	Not Normally Distributed
	F1-Score	baseline	0.534	3.45e-27	Not Normally Distributed
		ours	0.435	1.05e-17	Not Normally Distributed
	AUC	baseline	0.505	3.75e-24	Not Normally Distributed
		ours	0.424	7.52e-17	Not Normally Distributed
UNSW-NB15	Accuracy	baseline	0.434	1.11e-17	Not Normally Distributed
		ours	0.527	1.88e-26	Not Normally Distributed
	Precision	baseline	0.693	2.05e-48	Not Normally Distributed
		ours	0.613	1.16e-36	Not Normally Distributed
	Recall	baseline	0.606	8.85e-36	Not Normally Distributed
		ours	0.597	1.33e-34	Not Normally Distributed
	F1-Score	baseline	0.618	2.29e-37	Not Normally Distributed
		ours	0.596	2.19e-34	Not Normally Distributed
	AUC	baseline	0.606	8.85e-36	Not Normally Distributed
		ours	0.597	1.33e-34	Not Normally Distributed

The results of the Wilcoxon test in Table 5.9 show no significant difference for *IEC61850-Security* dataset metrics except accuracy. However, for *UNSW-NB15*, there is a significant difference between the baseline and our approach.

Table 5.9: Wilcoxon signed-rank test for DoS attack scenarios

Dataset	Metric	Statistic	p-value	Result	Effect Size
IEC61850-Security	Accuracy	986.0	1.21e-07	Significant Difference	0.529
	Precision	2379.0	6.16e-01	No Significant Difference	0.050
	Recall	2285.0	4.09e-01	No Significant Difference	0.083
	F1-Score	2258.0	3.59e-01	No Significant Difference	0.092
	AUC	2285.0	4.09e-01	No Significant Difference	0.083
UNSW-NB15	Accuracy	763.0	1.38e-09	Significant Difference	0.606
	Precision	866.0	1.17e-08	Significant Difference	0.570
	Recall	522.0	5.70e-12	Significant Difference	0.689
	F1-Score	548.0	1.06e-11	Significant Difference	0.680
	AUC	522.0	5.70e-12	Significant Difference	0.689

### 5.2.3 Detection of Other Attacks

The results for other *IEC61850-Security* dataset attacks were presented in Table 5.10. Control manipulation scenario has a proportion of 19/81 attacks and normal examples, message suppression is composed of 52/48 attacks and normal instances, and finally, data manipulation is composed of 36/64 attacks and normal instances. The table shows that overall our approach is better, except for accuracy in control and data manipulation attacks.



Table 5.10: Performance in detecting other *IEC61850-Security* attacks

Attack	Round	Strategy	Safe [%]	Borderline [%]	Accuracy	Recall	Precision	F1-Score	AUC
Control Manipulation	1-25	baseline	2.63 ± 5.06	33.61 ± 12.6	0.77 ± 0.11	0.56 ± 0.12	0.56 ± 0.14	0.55 ± 0.11	0.56 ± 0.12
		ours	7.82 ± 7.82	46.52 ± 11.77	0.77 ± 0.09	0.63 ± 0.12	0.65 ± 0.13	0.62 ± 0.11	0.63 ± 0.12
	1-50	baseline	5.0 ± 5.08	45.75 ± 16.52	0.73 ± 0.14	0.59 ± 0.15	0.59 ± 0.17	0.58 ± 0.15	0.59 ± 0.15
		ours	10.58 ± 7.26	55.95 ± 14.17	0.71 ± 0.14	0.62 ± 0.14	0.64 ± 0.15	0.62 ± 0.14	0.62 ± 0.14
	1-75	baseline	4.91 ± 4.37	54.36 ± 18.99	0.69 ± 0.17	0.58 ± 0.17	0.59 ± 0.18	0.58 ± 0.17	0.58 ± 0.17
		ours	11.36 ± 6.37	62.88 ± 15.44	0.66 ± 0.16	0.6 ± 0.15	0.61 ± 0.16	0.6 ± 0.15	0.6 ± 0.15
	1-100	baseline	7.02 ± 5.82	58.76 ± 18.35	0.66 ± 0.18	0.58 ± 0.18	0.58 ± 0.18	0.57 ± 0.18	0.58 ± 0.18
		ours	12.97 ± 6.58	65.08 ± 14.19	0.64 ± 0.17	0.6 ± 0.15	0.6 ± 0.16	0.59 ± 0.15	0.6 ± 0.15
Data Manipulation	1-25	baseline	11.93 ± 7.89	43.68 ± 11.39	0.72 ± 0.1	0.63 ± 0.1	0.65 ± 0.11	0.63 ± 0.11	0.63 ± 0.1
		ours	15.84 ± 7.44	55.21 ± 10.43	0.71 ± 0.1	0.65 ± 0.1	0.67 ± 0.11	0.65 ± 0.1	0.65 ± 0.1
	1-50	baseline	12.79 ± 6.33	51.14 ± 12.22	0.7 ± 0.13	0.64 ± 0.13	0.65 ± 0.14	0.64 ± 0.13	0.64 ± 0.13
		ours	18.88 ± 7.04	59.49 ± 9.3	0.68 ± 0.12	0.65 ± 0.12	0.66 ± 0.12	0.65 ± 0.12	0.65 ± 0.12
	1-75	baseline	15.72 ± 7.04	54.39 ± 11.29	0.69 ± 0.13	0.64 ± 0.13	0.65 ± 0.14	0.64 ± 0.14	0.64 ± 0.13
		ours	22.86 ± 8.57	59.56 ± 8.16	0.67 ± 0.12	0.65 ± 0.12	0.66 ± 0.12	0.65 ± 0.12	0.65 ± 0.12
	1-100	baseline	18.5 ± 7.88	54.29 ± 9.97	0.68 ± 0.14	0.64 ± 0.14	0.65 ± 0.14	0.64 ± 0.14	0.64 ± 0.14
		ours	23.94 ± 8.47	60.45 ± 8.38	0.67 ± 0.12	0.65 ± 0.12	0.66 ± 0.12	0.65 ± 0.12	0.65 ± 0.12
Message Suppression	1-25	baseline	9.03 ± 7.69	62.55 ± 10.01	0.64 ± 0.1	0.6 ± 0.1	0.6 ± 0.1	0.59 ± 0.1	0.6 ± 0.1
		ours	17.96 ± 7.99	61.15 ± 10.23	0.67 ± 0.1	0.65 ± 0.1	0.66 ± 0.1	0.65 ± 0.1	0.65 ± 0.1
	1-50	baseline	6.81 ± 6.1	67.77 ± 10.37	0.62 ± 0.15	0.59 ± 0.15	0.59 ± 0.15	0.58 ± 0.15	0.59 ± 0.15
		ours	14.75 ± 6.95	67.9 ± 10.42	0.63 ± 0.14	0.62 ± 0.13	0.63 ± 0.14	0.62 ± 0.13	0.62 ± 0.13
	1-75	baseline	6.96 ± 5.49	71.88 ± 10.57	0.59 ± 0.17	0.57 ± 0.16	0.57 ± 0.17	0.57 ± 0.17	0.57 ± 0.16
		ours	15.57 ± 6.35	69.25 ± 9.16	0.62 ± 0.15	0.61 ± 0.14	0.61 ± 0.15	0.61 ± 0.14	0.61 ± 0.14
	1-100	baseline	8.43 ± 5.46	72.45 ± 9.43	0.59 ± 0.18	0.57 ± 0.17	0.57 ± 0.18	0.57 ± 0.17	0.57 ± 0.17
		ours	15.59 ± 6.14	70.2 ± 8.54	0.61 ± 0.15	0.61 ± 0.15	0.61 ± 0.15	0.61 ± 0.15	0.61 ± 0.15

Table 5.11: Kolmogorov-Smirnov normality test for other *IEC61850-Security* attack scenarios

Attack	Metric	Strategy	Statistic	p-value	Result
Control Manipulation	Accuracy	baseline	0.482	6.95e-22	Not Normally Distributed
		ours	0.580	2.05e-32	Not Normally Distributed
	Precision	baseline	0.463	3.85e-20	Not Normally Distributed
		ours	0.540	8.59e-28	Not Normally Distributed
	Recall	baseline	0.438	5.59e-18	Not Normally Distributed
		ours	0.507	2.40e-24	Not Normally Distributed
	F1-Score	baseline	0.441	3.41e-18	Not Normally Distributed
		ours	0.496	3.35e-23	Not Normally Distributed
	AUC	baseline	0.438	5.59e-18	Not Normally Distributed
		ours	0.507	2.40e-24	Not Normally Distributed
Data Manipulation	Accuracy	baseline	0.461	6.05e-20	Not Normally Distributed
		ours	0.540	8.38e-28	Not Normally Distributed
	Precision	baseline	0.450	5.04e-19	Not Normally Distributed
		ours	0.429	3.16e-17	Not Normally Distributed
	Recall	baseline	0.418	2.39e-16	Not Normally Distributed
		ours	0.419	2.04e-16	Not Normally Distributed
	F1-Score	baseline	0.405	2.40e-15	Not Normally Distributed
		ours	0.419	2.08e-16	Not Normally Distributed
	AUC	baseline	0.418	2.39e-16	Not Normally Distributed
		ours	0.419	2.04e-16	Not Normally Distributed
Message Suppression	Accuracy	baseline	0.519	1.58e-25	Not Normally Distributed
		ours	0.491	9.37e-23	Not Normally Distributed
	Precision	baseline	0.459	9.03e-20	Not Normally Distributed
		ours	0.502	7.47e-24	Not Normally Distributed
	Recall	baseline	0.461	6.34e-20	Not Normally Distributed
		ours	0.502	7.41e-24	Not Normally Distributed
	F1-Score	baseline	0.467	1.80e-20	Not Normally Distributed
		ours	0.503	5.64e-24	Not Normally Distributed
	AUC	baseline	0.461	6.34e-20	Not Normally Distributed
		ours	0.502	7.41e-24	Not Normally Distributed

Statistical tests are performed to confirm if there is a significant difference between the two approaches. Table 5.11 checks the normal distribution with the Kolmogorov-Smirnov test for the three attack scenarios. The table shows no scenario with data following a normal distribution, so a non-parametrical test is chosen, such as the Wilcoxon signed-rank test.

The results of the Wilcoxon test in Table 5.12 show a significant difference between the two approaches for all scenarios. There is an exception for the precision of the data manipulation attack.

Table 5.12: Wilcoxon signed-rank test for other *IEC61850-Security* attack scenarios

Attack	Metric	Statistic	p-value	Result	Effect Size
Control Manipulation	Accuracy	1296.0	2.38e-05	Significant Difference	0.423
	Precision	1419.0	1.43e-04	Significant Difference	0.380
	Recall	1449.0	2.16e-04	Significant Difference	0.370
	F1-Score	1347.0	5.11e-05	Significant Difference	0.405
	AUC	1449.0	2.16e-04	Significant Difference	0.370
Data Manipulation	Accuracy	1305.0	2.73e-05	Significant Difference	0.419
	Precision	2062.0	1.11e-01	No Significant Difference	0.159
	Recall	1713.0	5.24e-03	Significant Difference	0.279
	F1-Score	1891.0	2.93e-02	Significant Difference	0.218
	AUC	1713.0	5.24e-03	Significant Difference	0.279
Message Suppression	Accuracy	627.0	6.76e-11	Significant Difference	0.653
	Precision	198.0	1.23e-15	Significant Difference	0.800
	Recall	139.0	2.33e-16	Significant Difference	0.820
	F1-Score	139.0	2.33e-16	Significant Difference	0.820
	AUC	139.0	2.33e-16	Significant Difference	0.820

## 5.2.4 Summary

In this section, different attack scenarios were tested using different datasets. These scenarios are attack, denial of service, message suppression, data manipulation, and control manipulation detection. Our approach shows better results than the baseline for all metrics, except for the DoS attack in the *IEC61850-Security* dataset, which presents no significant difference from the baseline. It can be explained by the *No Free Lunch Theorem* [130], where finding an algorithm that fits all situations is difficult.

## 5.3 Conclusions

This chapter begins with analyzing the time-series classifier, which fits better into the smart grid context and our need for a fast and efficient classifier. From all tested classifiers, only the TSF is the fastest for training in all datasets and present good classification results.

After that, structure learning was applied to each dataset to discover the structural causal model and, more precisely, the treatment and confounder features. Then, this SCM was validated with three refutation methods: random common cause, placebo treatment, and data subset.

Afterward, the second experimental phase involves testing our data processing ap-

proach in different scenarios, such as attack detection (without categorizing them). But also the denial of service, message suppression, data manipulation, and control manipulation detection. Overall, in each attack scenario studied, our approach is always better than the baseline scenario due to the augmentation of safe examples.



# Chapter 6

## Conclusion

This work examined the potential for using FL to simulate an intrusion detection system for smart grids, looking for imbalanced data and causality problems. While there have been previous efforts in this direction, only some have explored using data processing techniques in a federated context as a key component, which solves these two issues.

To address this gap in the literature, a three-level hierarchical system was proposed using the *Flower* framework. The system is divided into three kinds of nodes representing the different types of substations in a smart grid: the local node at the bottom level, the central local node at the second level, and the central node at the top level. The system is then controlled by a monitoring system, which communicates with the system through the MQTT protocol. This monitoring system generates the data sent to the federated system while the system retrieves metrics about its performance. The monitoring system displays real-time metrics information through a GUI and generates simulation reports.

The proposed system evaluates different kinds of cyberattacks from well-known network traffic datasets. In this work, two approaches were compared: the first, as a baseline, only apply time-series classifiers, and the second considers the proportion of minority class examples and the causality of the data.

The experiments in this work were separated into two phases. The first phase chooses the best time-series classifier which best fits this smart grid context and the construction of the structural causal model. The second phase tested different datasets in detecting anomalies without categorizing them. Then, other experiments were conducted to detect specific categories of attacks, such as DoS, message suppression, data manipulation, and control manipulation.

The results show that the best time-series classifier for this smart grid problem is the TSF. Overall in each experiment, the proposed approach significantly improves the classification result, enabling this work to compete with other works in the literature.

In future work, this federated system can conduct experiments simulating the smart grid with large-scale cloud infrastructure such as *Azure* or *Amazon Web Services* for more realistic scenarios. This deportation to cloud infrastructures solves hardware limitations, namely in processing large datasets over long periods, which was not

possible in the context of this work. However, there is also a limitation to public datasets, which may not represent the reality of a large-scale smart grid network traffic communication. This work approaches this reality with network traffic from small infrastructures. One of the key directions passes to develop real large-scale smart grid network communication datasets to evaluate with more confidence approaches developed by the scientific community.

In summary, this research has provided an overview of a federated-based simulation and its potential application in smart grids for anomaly detection. A novel data processing approach that considers data imbalance and causality problems, which performs better than using only a time-series classifier. The foundations for future work are laid for more realistic large-scale testbeds.

# References

- [1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv preprint arXiv:1603.04467*, 3 2016.
- [2] Moataz Abdelkhalek, Gelli Ravikumar, and Manimaran Govindarasu. ML-based Anomaly Detection System for DER Communication in Smart Grid. *2022 IEEE Power and Energy Society Innovative Smart Grid Technologies Conference, ISGT 2022*, pages 1–5, 2022. doi: 10.1109/ISGT50606.2022.9817481.
- [3] Abdulaziz Aldribi, Issa Traore, Paulo Gustavo Quinan, and Onyekachi Nwamu. Documentation for the ISOT Cloud Intrusion Detection Benchmark Dataset (ISOT-CID). Technical report, University of Victoria, 2020.
- [4] Abdullah Alsaedi, Nour Moustafa, Zahir Tari, Abdun Mahmood, and Adna N Anwar. TON-IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems. *IEEE Access*, 8:165130–165150, 2020. ISSN 21693536. doi: 10.1109/ACCESS.2020.3022862.
- [5] Javed Ashraf, Marwa Keshk, Nour Moustafa, Mohamed Abdel-Basset, Hasnat Khurshid, Asim D. Bakhshi, and Reham R. Mostafa. IoTBoT-IDS: A novel statistical learning-enabled botnet detection framework for protecting networks of smart cities. *Sustainable Cities and Society*, 72:103041, 9 2021. ISSN 2210-6707. doi: 10.1016/J.SCS.2021.103041.
- [6] Ausgrid. Solar home electricity data, 2017. URL <https://www.ausgrid.com.au/Industry/Our-Research/Data-to-share/Solar-home-electricity-data>. (accessed on 15/01/2023).
- [7] Peter C. Austin. An Introduction to Propensity Score Methods for Reducing the Effects of Confounding in Observational Studies. <https://doi.org/10.1080/00273171.2011.568786>, 46(3):399–424, 5 2011. ISSN 00273171. doi: 10.1080/00273171.2011.568786.

- [8] Andrew Banks, Ed Briggs, Ken Borgendale, and Rahul Gupta. MQTT Version 5.0, 2019. URL <https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.pdf>. (accessed on 15/01/2023).
- [9] Paul Beaumont, Ben Horsburgh, Philip Pilgerstorfer, Angel Droth, Richard Oentaryo, Steven Ler, Hiep Nguyen, Gabriel Azevedo Ferreira, Zain Patel, and Wesley Leong. CausalNex, 2021. URL <https://github.com/quantumblacklabs/causalnex>. (accessed on 30/06/2023).
- [10] Daniel J. Beutel, Taner Topal, Akhil Mathur, Xinchu Qiu, Javier Fernandez-Marques, Yan Gao, Lorenzo Sani, Kwing Hei Li, Titouan Parcollet, Pedro Porto Buarque de Gusmão, and Nicholas D. Lane. Flower: A Friendly Federated Learning Research Framework. *arXiv preprint arXiv:2007.14390*, 7 2020.
- [11] Partha P. Biswas, Heng Chuan Tan, Qingbo Zhu, Yuan Li, Daisuke Mashima, and Binbin Chen. A Synthesized Dataset for Cybersecurity Study of IEC 61850 based Substation. In *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (Smart-GridComm)*, pages 1–7. IEEE, 10 2019. ISBN 978-1-5386-8099-5. doi: 10.1109/SmartGridComm.2019.8909783.
- [12] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical Secure Aggregation for Federated Learning on User-Held Data. *arXiv preprint arXiv:1611.04482*, 11 2016. doi: 10.48550/arxiv.1611.04482.
- [13] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečný, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards Federated Learning at Scale: System Design. *Proceedings of Machine Learning and Systems*, 1:374–388, 2 2019.
- [14] Tim M. Booij, Irina Chiscop, Erik Meeuwissen, Nour Moustafa, and Frank T.H. Den Hartog. ToN-IoT: The Role of Heterogeneity and the Need for Standardization of Features and Attack Types in IoT Network Intrusion Data Sets. *IEEE Internet of Things Journal*, 9(1):485–496, 1 2022. ISSN 23274662. doi: 10.1109/JIOT.2021.3085194.
- [15] Raouf Boutaba, Mohammad A. Salahuddin, Noura Limam, Sara Ayoubi, Nashid Shahriar, Felipe Estrada-Solano, and Oscar M. Caicedo. A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. *Journal of Internet Services and Applications*, 9 (1):16, 12 2018. ISSN 1867-4828. doi: 10.1186/s13174-018-0087-2.
- [16] Brady Neal. *Introduction to Causal Inference from a Machine Learning Perspective*. Course Lecture Notes (draft), 2020.
- [17] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. LEAF: A Benchmark for Federated Settings. *arXiv preprint arXiv:1812.01097*, 12 2018.



- 
- [18] Canadian Institute for Cybersecurity. A Realistic Cyber Defense Dataset (CSE-CIC-IDS-2018) - Registry of Open Data on AWS, 2018. URL <https://registry.opendata.aws/cse-cic-ids2018/>. (accessed on 15/01/2023).
- [19] UNSW Canberra. UNSW-NB15 Dataset, 2015. URL <https://research.unsw.edu.au/projects/unsw-nb15-dataset>. (accessed on 15/01/2023).
- [20] UNSW Canberra. Bot-IoT Dataset, 2018. URL <https://research.unsw.edu.au/projects/bot-iot-dataset>. (accessed on 15/01/2023).
- [21] Ran Canetti, Uri Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96*, pages 639–648, New York, New York, USA, 1996. ACM Press. ISBN 0897917855. doi: 10.1145/237814.238015.
- [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, 6 2002. ISSN 1076-9757. doi: 10.1613/JAIR.953.
- [23] Dengsheng Chen, Vince Tan, Zhilin Lu, and Jie Hu. OpenFed: A Comprehensive and Versatile Open-Source Federated Learning Framework. *arXiv preprint arXiv:2109.07852*, 9 2021.
- [24] Olivia Choudhury, Aris Gkoulalas-Divanis, Theodoros Salonidis, Issa Sylla, Yoonyoung Park, Grace Hsu, and Amar Das. Differential Privacy-enabled Federated Learning for Sensitive Health Data. *arXiv preprint arXiv:1910.02578*, 10 2019. doi: 10.48550/arxiv.1910.02578.
- [25] Michael A. A. Cox and Trevor F. Cox. Multidimensional Scaling. In *Handbook of Data Visualization*, pages 315–347. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. doi: 10.1007/978-3-540-33037-0\_14.
- [26] Angus Dempster, François Petitjean, and Geoffrey I Webb. ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34:1454–1495, 2020. doi: 10.1007/s10618-020-00701-z.
- [27] Angus Dempster, Daniel F. Schmidt, and Geoffrey I. Webb. MiniRocket: A Very Fast (Almost) Deterministic Transform for Time Series Classification. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 248–257, 8 2021. doi: 10.1145/3447548.3467231.
- [28] Houtao Deng, George Runger, Eugene Tuv, and Martyanov Vladimir. A time series forest for classification and feature extraction. *Information Sciences*, 239:142–153, 8 2013. ISSN 0020-0255. doi: 10.1016/J.INS.2013.02.030.
- [29] Dheeru Dua and Casey Graff. UCI Machine Learning Repository, 2017. URL <http://archive.ics.uci.edu/ml>. (accessed on 15/01/2023).

- [30] Cynthia Dwork. Differential Privacy: A Survey of Results. In *Theory and Applications of Models of Computation*, volume 4978 LNCS, pages 1–19. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. ISBN 3540792279. doi: 10.1007/978-3-540-79228-4\_1.
- [31] Ebenezer Esenogho, Karim Djouani, and Anish M. Kurien. Integrating Artificial Intelligence Internet of Things and 5G for Next-Generation Smartgrid: A Survey of Trends Challenges and Prospect, 2022. ISSN 21693536.
- [32] Haokun Fang and Quan Qian. Privacy Preserving Machine Learning with Homomorphic Encryption and Federated Learning. *Future Internet 2021, Vol. 13, Page 94*, 13(4):94, 4 2021. ISSN 1999-5903. doi: 10.3390/FI13040094.
- [33] Xi Fang, Satyajayant Misra, Guoliang Xue, and Dejun Yang. Smart grid - The new and improved power grid: A survey, 2012. ISSN 1553877X.
- [34] Hassan Farhangi. The Path of the Smart Grid. *IEEE Power and Energy Magazine*, 8(1):18–28, 1 2010. ISSN 15407977. doi: 10.1109/MPE.2009.934876.
- [35] Mohamed Amine Ferrag, Othmane Friha, Djallel Hamouda, Leandros Maglaras, and Helge Janicke. Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning. *IEEE Access*, 10:40281–40306, 2022. ISSN 21693536. doi: 10.1109/ACCESS.2022.3165809.
- [36] Patrick Foley, Micah J Sheller, Brandon Edwards, Sarthak Pati, Walter Riviera, Mansi Sharma, Prakash Narayana Moorthy, Shih-han Wang, Jason Martin, Parsa Mirhaji, Prashant Shah, and Spyridon Bakas. OpenFL: the open federated learning library. *Physics in Medicine & Biology*, 67(21):214001, 11 2022. ISSN 0031-9155. doi: 10.1088/1361-6560/ac97d9.
- [37] Canadian Institute for Cybersecurity. NSL-KDD Dataset, 2009. URL <https://www.unb.ca/cic/datasets/nsl.html>. (accessed on 15/01/2023).
- [38] Canadian Institute for Cybersecurity. ISCX-IDS-2012 Dataset, 2012. URL <https://www.unb.ca/cic/datasets/ids.html>. (accessed on 15/01/2023).
- [39] Canadian Institute for Cybersecurity. CIC-IDS-2017 Dataset, 2017. URL <https://www.unb.ca/cic/datasets/ids-2017.html>. (accessed on 15/01/2023).
- [40] Canadian Institute for Cybersecurity. CSE-CIC-IDS-2018 Dataset, 2018. URL <https://www.unb.ca/cic/datasets/ids-2018.html>. (accessed on 15/01/2023).
- [41] Ivo Frazão, Pedro Henriques Abreu, Tiago Cruz, Hélder Araújo, and Paulo Simões. Denial of service attacks: Detecting the frailties of machine learning algorithms in the classification process. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11260 LNCS:230–235, 2019. ISSN 16113349. doi: 10.1007/978-3-030-05849-4\_19.

- 
- [42] Othmane Friha, Mohamed Amine Ferrag, Mohamed Benbouzid, Tarek Berghout, Burak Kantarci, and Kim-Kwang Raymond Choo. 2DF-IDS: Decentralized and Differentially Private Federated Learning-based Intrusion Detection System for Industrial IoT. *Computers & Security*, page 103097, 1 2023. ISSN 01674048. doi: 10.1016/j.cose.2023.103097.
- [43] Mirian Hipolito Garcia, Andre Manoel, Daniel Madrigal Diaz, Fatemehsadat Miresghallah, Robert Sim, and Dimitrios Dimitriadis. FLUTE: A Scalable, Extensible Framework for High-Performance Federated Learning Simulations. *arXiv preprint arXiv:2203.13789*, 3 2022.
- [44] Sebastian Garcia, Agustin Parmisano, and Maria Jose Erquiaga. IoT-23: A labeled dataset with malicious and benign IoT network traffic. *Zenodo*, 1 2020. doi: 10.5281/ZENODO.4743746.
- [45] Robin C Geyer, Tassilo Klein, Moin Nabi, Sap Se, and Eth Zurich. Differentially Private Federated Learning: A Client Level Perspective. *arXiv preprint arXiv:1712.07557*, 12 2017. doi: 10.48550/arxiv.1712.07557.
- [46] 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, 6 2008. ISBN 978-1-4244-1820-6. doi: 10.1109/IJCNN.2008.4633969.
- [47] Jason S. Haukoos and Roger J. Lewis. The Propensity Score. *JAMA*, 314(15): 1637, 10 2015. ISSN 15383598. doi: 10.1001/JAMA.2015.13480.
- [48] Chaoyang He, Songze Li, Jinhyun So, Xiao Zeng, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, Xinghua Zhu, Jianzong Wang, Li Shen, Peilin Zhao, Yan Kang, Yang Liu, Ramesh Raskar, Qiang Yang, Murali Annavaram, and Salman Avestimehr. FedML: A Research Library and Benchmark for Federated Machine Learning. *arXiv preprint arXiv:2007.13518*, 7 2020.
- [49] MA Hernán and JM Robins. *Causal Inference: What If*. Chapman & Hall/CRC, Boca Raton, 2020.
- [50] Jon Hills, Jason Lines, Edgaras Baranauskas, James Mapp, and Anthony Bagnall. Classification of time series by shapelet transformation. *Data Mining and Knowledge Discovery*, 28(4):851–881, 5 2014. ISSN 13845810. doi: 10.1007/S10618-013-0322-1.
- [51] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the Effects of Non-Identical Data Distribution for Federated Visual Classification. *arXiv preprint arXiv:1909.06335*, 9 2019.
- [52] Yao Hu, Xiaoyan Sun, Ye Tian, Linqi Song, and Kay Chen Tan. Communication Efficient Federated Learning With Heterogeneous Structured Client Models. *IEEE Transactions on Emerging Topics in Computational Intelligence*, pages 1–15, 2022. ISSN 2471-285X. doi: 10.1109/TETCI.2022.3209345.

- [53] Zeou Hu, Kiarash Shaloudegi, Guojun Zhang, and Yaoliang Yu. Federated Learning Meets Multi-Objective Optimization. *IEEE Transactions on Network Science and Engineering*, 9(4):2039–2051, 2022. ISSN 23274697. doi: 10.1109/TNSE.2022.3169117.
- [54] Maria C S Inacio, Chen Bs, Elizabeth W Paxton, Robert S Namba, Steven M Kurtz, and Guy Cafri. Statistics in Brief: An Introduction to the Use of Propensity Scores. *Clinical Orthopaedics and Related Research®*, 2015. doi: 10.1007/s11999-015-4239-4.
- [55] Academic Conferences International. ECCWS future and past conferences, 2023. URL <https://www.academic-conferences.org/conferences/eccws/eccws-future-and-past/>. (accessed on 30/06/2023).
- [56] Jing Jiang, Shaoxiong Ji, and Guodong Long. Decentralized Knowledge Acquisition for Mobile Internet Applications. *World Wide Web*, 23(5):2653–2669, 9 2020. ISSN 15731413. doi: 10.1007/s11280-019-00775-w.
- [57] J. Jithish, Bithin Alangot, Nagarajan Mahalingam, and Kiat Seng Yeo. Distributed Anomaly Detection in Smart Grids: A Federated Learning-Based Approach. *IEEE Access*, 11:7157–7179, 2023. ISSN 21693536. doi: 10.1109/ACCESS.2023.3237554.
- [58] Jean Kaddour, Aengus Lynch, Qi Liu, Matt J. Kusner, and Ricardo Silva. Causal Machine Learning: A Survey and Open Problems. *arXiv preprint arXiv:2206.15475*, 6 2022.
- [59] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G.L. D’Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecni, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrede Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Hang Qi, Daniel Ramage, Ramesh Raskar, Mariana Raykova, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and Open Problems in Federated Learning. *Foundations and Trends® in Machine Learning*, 14 (1–2):1–210, 6 2021. ISSN 1935-8237. doi: 10.1561/22000000083.
- [60] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. In *International Conference on Machine Learning*, pages 5132–5143, 2020.
- [61] Suleman Khan, Kashif Kifayat, Ali Kashif Bashir, Andrei Gurtov, and Mehdi Hassan. Intelligent intrusion detection system in smart grid using computational intelligence and machine learning. *Transactions on Emerging*

- 
- Telecommunications Technologies*, 32(6):e4062, 6 2021. ISSN 2161-3915. doi: 10.1002/ETT.4062.
- [62] Diederik P. Kingma and Jimmy Lei Ba. Adam: A Method for Stochastic Optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 12 2014. doi: 10.48550/arxiv.1412.6980.
- [63] Nickolaos Koroniotis. *Designing an effective network forensic framework for the investigation of botnets in the Internet of Things*. PhD thesis, University of New South Wales, 2020.
- [64] Nickolaos Koroniotis and Nour Moustafa. Enhancing Network Forensics with Particle Swarm and Deep Learning: The Particle Deep Framework. In *7th International Conference on Artificial Intelligence and Applications*, pages 41–60. Aircc Publishing Corporation, 3 2020. ISBN 9781925953176. doi: 10.5121/csit.2020.100304.
- [65] Nickolaos Koroniotis, Nour Moustafa, Elena Sitnikova, and Jill Slay. Towards developing network forensic mechanism for botnet activities in the IoT based on machine learning techniques. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, 235:30–44, 2018. ISSN 18678211. doi: 10.1007/978-3-319-90775-8\_3.
- [66] 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, 11 2019. ISSN 0167-739X. doi: 10.1016/J.FUTURE.2019.05.041.
- [67] Nickolaos Koroniotis, Nour Moustafa, Francesco Schiliro, Praveen Gauravaram, and Helge Janicke. A Holistic Review of Cybersecurity and Reliability Perspectives in Smart Airports. *IEEE Access*, 8:209802–209834, 2020. ISSN 21693536. doi: 10.1109/ACCESS.2020.3036728.
- [68] Nickolaos Koroniotis, Nour Moustafa, and Elena Sitnikova. A new network forensic framework based on deep learning for Internet of Things networks: A particle deep framework. *Future Generation Computer Systems*, 110:91–106, 9 2020. ISSN 0167-739X. doi: 10.1016/J.FUTURE.2020.03.042.
- [69] Jorma Laurikkala. Improving identification of difficult small classes by balancing class distribution. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 2101, pages 63–66. Springer Verlag, 2001. ISBN 3540422943. doi: 10.1007/3-540-48229-6\_9.
- [70] Beibei Li, Yuhao Wu, Jiarui Song, Rongxing Lu, Tao Li, and Liang Zhao. DeepFed: Federated Deep Learning for Intrusion Detection in Industrial Cyber-Physical Systems. *IEEE Transactions on Industrial Informatics*, 17(8):5615–5624, 8 2021. ISSN 19410050. doi: 10.1109/TII.2020.3023430.

- [71] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated Optimization In Heterogeneous Networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.
- [72] Jason Lines and Anthony Bagnall. Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 29(3): 565–592, 4 2015. ISSN 13845810. doi: 10.1007/S10618-014-0361-2.
- [73] Jason Lines, Sarah Taylor, and Anthony Bagnall. Time Series Classification with HIVE-COTE. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(5):52, 7 2018. ISSN 1556472X. doi: 10.1145/3182382.
- [74] Yang Liu, Tao Fan, Tianjian Chen, Qian Xu, and Qiang Yang. FATE: An Industrial Grade Platform for Collaborative Learning With Data Protection. *Journal of Machine Learning Research*, 22(226):1–6, 2021. ISSN 1533-7928.
- [75] Yang Liu, Xinwei Zhang, Yan Kang, Liping Li, Tianjian Chen, Mingyi Hong, and Qiang Yang. FedBCD: A Communication-Efficient Collaborative Learning Framework for Distributed Features. *IEEE Transactions on Signal Processing*, 70:4277–4290, 2022. ISSN 19410476. doi: 10.1109/TSP.2022.3198176.
- [76] Carl H. Lubba, Sarab S. Sethi, Philip Knaute, Simon R. Schultz, Ben D. Fulcher, and Nick S. Jones. catch22: CAnonical Time-series CHaracteristics: Selected through highly comparative time-series analysis. *Data Mining and Knowledge Discovery*, 33(6):1821–1852, 11 2019. ISSN 1573756X. doi: 10.1007/S10618-019-00647-X.
- [77] Benjamin Lucas, Ahmed Shifaz, Charlotte Pelletier, Lachlan O’Neill, Nayyar Zaidi, Bart Goethals, François Petitjean, and Geoffrey I. Webb. Proximity Forest: an effective and scalable distance-based classifier for time series. *Data Mining and Knowledge Discovery*, 33(3):607–635, 5 2019. ISSN 1573756X. doi: 10.1007/S10618-019-00617-3.
- [78] Yanjun Ma, Dianhai Yu, Tian Wu, Haifeng Wang, Yanjun Ma, Dianhai Yu, Tian Wu, and Haifeng Wang. PaddlePaddle: An Open-Source Deep Learning Platform from Industrial Practice. *Frontiers of Data and Computing*, 1(1): 105–115, 10 2019. ISSN 1674-9480. doi: 10.11871/JFDC.ISSN.2096.742X. 2019.01.011.
- [79] Macrovector. Macrovector on Feepik, 2019. URL <https://www.freepik.com/author/macrovector>. (accessed on 30/06/2023).
- [80] Joaquim P. Marques de Sá. *Pattern Recognition*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. ISBN 978-3-642-62677-7. doi: 10.1007/978-3-642-56651-6.
- [81] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Agüera Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Artificial intelligence and statistics*, pages 1273–1282, 2 2016.

- 
- [82] Matthew Middlehurst, William Vickers, and Anthony Bagnall. Scalable Dictionary Classifiers for Time Series Classification. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11871 LNCS:11–19, 7 2019. doi: 10.1007/978-3-030-33607-3\_2.
- [83] Matthew Middlehurst, James Large, and Anthony Bagnall. The Canonical Interval Forest (CIF) Classifier for Time Series Classification. *Proceedings - 2020 IEEE International Conference on Big Data, Big Data 2020*, pages 188–195, 12 2020. doi: 10.1109/BIGDATA50022.2020.9378424.
- [84] Matthew Middlehurst, James Large, Michael Flynn, Jason Lines, Aaron Bostrom, and Anthony Bagnall. HIVE-COTE 2.0: a new meta ensemble for time series classification. *Machine Learning*, 110(11-12):3211–3243, 12 2021. ISSN 15730565. doi: 10.1007/S10994-021-06057-9.
- [85] Dinesh Mohanty, Kamalakanta Sethi, Sai Prasath, Rashmi Ranjan Rout, and Padmalochan Bera. Intelligent Intrusion Detection System for Smart Grid Applications. *2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment, CyberSA 2021*, 6 2021. doi: 10.1109/CYBERSA52016.2021.9478200.
- [86] Thomas Morris and Wei Gao. Industrial control system traffic data sets for intrusion detection research. *IFIP Advances in Information and Communication Technology*, 441:65–78, 2014. ISSN 18684238. doi: 10.1007/978-3-662-45355-1\_5.
- [87] Viraaaji Mothukuri, Reza M. Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 115:619–640, 2 2021. ISSN 0167-739X. doi: 10.1016/J.FUTURE.2020.10.007.
- [88] Viraaaji Mothukuri, Prachi Khare, Reza M. Parizi, Seyedamin Pouriyeh, Ali Dehghantanha, and Gautam Srivastava. Federated-Learning-Based Anomaly Detection for IoT Security Attacks. *IEEE Internet of Things Journal*, 9(4): 2545–2554, 2 2022. ISSN 23274662. doi: 10.1109/JIOT.2021.3077803.
- [89] Nour Moustafa. New Generations of Internet of Things Datasets for Cybersecurity Applications based Machine Learning: TON-IoT Datasets. In *Proceedings of the eResearch Australasia Conference*, Brisbane, Australia., 2019. doi: <https://doi.org/10.26190/5d7ac9bfe8487>.
- [90] Nour Moustafa. A Systemic IoT-Fog-Cloud Architecture for Big-Data Analytics and Cyber Security Systems: A Review of Fog Computing. *Secure Edge Computing*, pages 41–50, 5 2019.
- [91] Nour Moustafa. A new distributed architecture for evaluating AI-based security systems at the edge: Network TON-IoT datasets. *Sustainable Cities and Society*, 72:102994, 9 2021. ISSN 2210-6707. doi: 10.1016/J.SCS.2021.102994.

- [92] Nour Moustafa and Jill Slay. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). *2015 Military Communications and Information Systems Conference, MilCIS 2015 - Proceedings*, 12 2015. doi: 10.1109/MILCIS.2015.7348942.
- [93] Nour Moustafa and Jill Slay. The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Information Security Journal: A Global Perspective*, 25(1-3):18–31, 4 2016. ISSN 1939-3555. doi: 10.1080/19393555.2015.1125974.
- [94] Nour Moustafa, Gideon Creech, and Jill Slay. Big Data Analytics for Intrusion Detection System: Statistical Decision-Making Using Finite Dirichlet Mixture Models. In *Data Analytics and Decision Support for Cybersecurity*, pages 127–156. Springer, Cham, 2017. doi: 10.1007/978-3-319-59439-2\_5.
- [95] Nour Moustafa, Jill Slay, and Gideon Creech. Novel Geometric Area Analysis Technique for Anomaly Detection Using Trapezoidal Area Estimation on Large-Scale Networks. *IEEE Transactions on Big Data*, 5(4):481–494, 12 2019. ISSN 23327790. doi: 10.1109/TBDDATA.2017.2715166.
- [96] Nour Moustafa, Mohiuddin Ahmed, and Sherif Ahmed. Data Analytics-enabled Intrusion Detection: Evaluations of ToN IoT Linux Datasets. *Proceedings - 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2020*, pages 727–735, 12 2020. doi: 10.1109/TRUSTCOM50675.2020.00100.
- [97] Nour Moustafa, Marwa Keshk, Essam Debie, and Helge Janicke. Federated TON-IoT windows datasets for evaluating AI-based security applications. *Proceedings - 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2020*, pages 848–855, 12 2020. doi: 10.1109/TRUSTCOM50675.2020.00114.
- [98] Krystyna Napierala and Jerzy Stefanowski. Types of minority class examples and their influence on learning classifiers from imbalanced data. *Journal of Intelligent Information Systems*, 46(3):563–597, 6 2016. ISSN 15737675. doi: 10.1007/s10844-015-0368-1.
- [99] Krystyna Napierała, Jerzy Stefanowski, and Szymon Wilk. Learning from Imbalanced Data in Presence of Noisy and Borderline Examples. In Marcin Szczuka, Marzena Kryszkiewicz, Sheela Ramanna, Richard Jensen, and Qinghua Hu, editors, *International conference on rough sets and current trends in computing*, pages 158–167. Springer Berlin Heidelberg, 2010. ISBN 3642135285. doi: 10.1007/978-3-642-13529-3\_18.
- [100] John Nguyen, Kshitiz Malik, Hongyuan Zhan, Ashkan Yousefpour, Mike Rabbat, Mani Malek, Dzmitry Huba, and Meta Ai. Federated Learning with Buffered Asynchronous Aggregation. In *International Conference on Artificial Intelligence and Statistics*, pages 3581–3607, 2022.
- [101] Luong Trung Nguyen, Junhan Kim, and Byonghyo Shim. Gradual Federated Learning with Simulated Annealing. *IEEE Transactions on Signal Processing*, 69:6299–6313, 2021. ISSN 19410476. doi: 10.1109/TSP.2021.3125137.



- 
- [102] Judea Pearl. An introduction to causal inference. *International Journal of Biostatistics*, 6(2), 2 2010. ISSN 15574679. doi: 10.2202/1557-4679.1203/MACHINEREAABLECITATION/RIS.
- [103] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and H. Brendan McMahan. Adaptive Federated Optimization. *arXiv preprint arXiv:2003.00295*, 2 2020.
- [104] Nuria Rodríguez-B., Goran Stipcich, Daniel Jiménez-L., José Antonio Ruiz-M., Eugenio Martínez-C., Gerardo González-S., M. Victoria Luzón, Miguel Angel Veganzones, and Francisco Herrera. Federated Learning and Differential Privacy: Software tools analysis, the Sherpa.ai FL framework and methodological guidelines for preserving data privacy. *Information Fusion*, 64: 270–292, 12 2020. ISSN 15662535. doi: 10.1016/j.inffus.2020.07.009.
- [105] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 9 2016.
- [106] Stuart Russell and Peter Norvig. *Artificial Intelligence, Global Edition A Modern Approach*. Pearson Deutschland, 2021. ISBN 9781292401133.
- [107] Hiroaki Sakoe and Seibi Chiba. Dynamic Programming Algorithm Optimization for Spoken Word Recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978. ISSN 00963518. doi: 10.1109/TASSP.1978.1163055.
- [108] Mohanad Sarhan, Siamak Layeghy, Nour Moustafa, and Marius Portmann. NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, 371 LNICST: 117–135, 2021. ISSN 1867822X. doi: 10.1007/978-3-030-72802-1\_9.
- [109] Patrick Schäfer. The BOSS is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, 29(6):1505–1530, 11 2015. ISSN 13845810. doi: 10.1007/S10618-014-0377-7.
- [110] Patrick Schäfer and Mikael Höggqvist. SFA: A symbolic fourier approximation and index for similarity search in high dimensional datasets. *ACM International Conference Proceeding Series*, pages 516–527, 2012. doi: 10.1145/2247596.2247656.
- [111] Patrick Schäfer and Ulf Leser. Fast and accurate time series classification with WEASEL. *International Conference on Information and Knowledge Management, Proceedings*, Part F131841:637–646, 11 2017. doi: 10.1145/3132847.3132980.
- [112] Patrick Schäfer and Ulf Leser. WEASEL 2.0 – A Random Dilated Dictionary Transform for Fast, Accurate and Memory Constrained Time Series Classification. *arXiv preprint arXiv:2301.10194*, 1 2023. doi: 10.48550/arXiv.2301.10194.

- [113] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, pages 108–116. SCITEPRESS - Science and Technology Publications, 2018. ISBN 978-989-758-282-0. doi: 10.5220/0006639801080116.
- [114] Amit Sharma and Emre Kiciman. Dowhy: An end-to-end library for causal inference. *arXiv preprint arXiv:2011.04216*, 2020.
- [115] Amit Sharma, Vasilis Syrgkanis, Cheng Zhang, and Emre Kiciman. DoWhy: Addressing Challenges in Expressing and Validating Causal Assumptions. *arXiv preprint arXiv:2108.13518*, 8 2021.
- [116] Ali Shiravi, Hadi Shiravi, Mahbod Tavallaee, and Ali A. Ghorbani. Toward Developing a Systematic Approach to Generate Benchmark Datasets for Intrusion Detection. *Computers & Security*, 31(3):357–374, 5 2012. ISSN 0167-4048. doi: 10.1016/J.COSE.2011.12.012.
- [117] Ilias Siniosoglou, Panagiotis Radoglou-Grammatikis, Georgios Efstathopoulos, Panagiotis Fouliras, and Panagiotis Sarigiannidis. A Unified Deep Learning Anomaly Detection and Classification Approach for Smart Grid Environments. *IEEE Transactions on Network and Service Management*, 18(2):1137–1151, 6 2021. ISSN 19324537. doi: 10.1109/TNSM.2021.3078381.
- [118] S. Sofana Reka, Tomislav Dragičević, Pierluigi Siano, and S. R. Sahaya Prabaharan. Future generation 5G wireless networks for smart grid: A comprehensive review, 6 2019. ISSN 19961073.
- [119] Jerzy Stefanowski and Szymon Wilk. Selective Pre-processing of Imbalanced Data for Improving Classification Performance. In *Data Warehousing and Knowledge Discovery*, pages 283–292. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. ISBN 3540858350. doi: 10.1007/978-3-540-85836-2\_27.
- [120] Salvatore Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, and Philip Chan. KDD Cup 1999 Data. UCI Machine Learning Repository, 1999.
- [121] Gian Antonio Susto, Angelo Cenedese, and Matteo Terzi. Time-Series Classification Methods: Review and Applications to Power Systems Data. *Big Data Application in Power Systems*, pages 179–220, 1 2018. doi: 10.1016/B978-0-12-811968-6.00009-7.
- [122] Seyedeh Mahsan Taghavinejad, Mehran Taghavinejad, Lida Shahmiri, Mohammad Zavvar, and Mohammad Hossein Zavvar. Intrusion Detection in IoT-Based Smart Grid Using Hybrid Decision Tree. *2020 6th International Conference on Web Research, ICWR 2020*, pages 152–156, 4 2020. doi: 10.1109/ICWR49608.2020.9122320.
- [123] Shawhin Talebi. Causal effects via propensity scores, 2023. URL <https://towardsdatascience.com/propensity-score-5c29c480130c>. (accessed on 30/06/2023).

- 
- [124] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the KDD CUP 99 data set. *IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009*, 12 2009. doi: 10.1109/CISDA.2009.5356528.
- [125] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated Learning with Matched Averaging. *arXiv preprint arXiv:2002.06440*, 2 2020.
- [126] Daniel Westreich, Justin Lessler, and Michele Jonsson Funk. Propensity score estimation: neural networks, support vector machines, decision trees (CART), and meta-classifiers as alternatives to logistic regression. *Journal of Clinical Epidemiology*, 63(8):826–833, 8 2010. ISSN 0895-4356. doi: 10.1016/J.JCLINEPI.2009.11.020.
- [127] D. R. Wilson and T. R. Martinez. Improved Heterogeneous Distance Functions. *Journal of Artificial Intelligence Research*, 6:1–34, 12 1996.
- [128] Wireshark. The Wireshark Network Protocol Analyzer, 1998. URL <https://www.wireshark.org/>. (accessed on 30/06/2023).
- [129] Szymon Wojciechowski, Szymon Wilk, and Jerzy Stefanowski. An algorithm for selective preprocessing of multi-class imbalanced data. *Advances in Intelligent Systems and Computing*, 578:238–247, 2018. ISSN 21945357. doi: 10.1007/978-3-319-59162-9\_25.
- [130] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997. doi: 10.1109/4235.585893.
- [131] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Asynchronous Federated Optimization. *arXiv preprint arXiv:1903.03934*, 3 2019.
- [132] Manzil Zaheer, Sashank J Reddi, Devendra Sachan, Satyen Kale, Google Research, and Sanjiv Kumar. Adaptive Methods for Nonconvex Optimization. *Advances in neural information processing systems*, 2018. doi: 10.5555/3327546.3327647.
- [133] Zeek. The Zeek Network Security Monitor, 1998. URL <https://www.zeek.org/>. (accessed on 30/06/2023).
- [134] Dun Zeng, Siqi Liang, Xiangjing Hu, Hui Wang, and Zenglin Xu. FedLab: A Flexible Federated Learning Framework. *arXiv preprint arXiv:2107.11621*, 7 2021.
- [135] Chengliang Zhang, Suyi Li, Junzhe Xia, Wei Wang, Hong Kong, Feng Yan, and Yang Liu. BatchCrypt: Efficient Homomorphic Encryption for Cross-Silo Federated Learning. *USENIX annual technical conference (USENIX ATC 20)*, pages 493–506, 2020. doi: 10.5555/3489146.3489179.
- [136] Xun Zheng, Bryon Aragam, Pradeep Ravikumar, and Eric P. Xing. DAGs with NO TEARS: Continuous Optimization for Structure Learning. *Advances*

*in Neural Information Processing Systems*, 31, 3 2018. doi: <https://doi.org/10.48550/arXiv.1803.01422>.

- [137] Alexander Ziller, Andrew Trask, Antonio Lopardo, Benjamin Szymkow, Bobby Wagner, Emma Bluemke, Jean Mickael Nounahon, Jonathan Passerat-Palmbach, Kritika Prakash, Nick Rose, Théo Ryffel, Zarreen Naowal Reza, and Georgios Kaissis. PySyft: A Library for Easy Federated Learning. In *Studies in Computational Intelligence*, volume 965, pages 111–139. Springer Science and Business Media Deutschland GmbH, 2021. doi: 10.1007/978-3-030-70604-3\_5.

# Appendices



# Appendix A

## Datasets

Table A.1: BOT-IoT Dataset Description

Name	Type	Unique Values	Missing Values	Description
stime	Float	392259	0	Record start time
flgs	Categorical	9	0	Flow state flags seen in transactions
flgs-number	Integer	9	0	Numerical representation of feature flags
proto	Categorical	5	0	Protocols present in network flow (TCP, UDP, ICMP, ICMP/IPv6)
proto-number	Integer	5	0	Numerical representation of the protocol feature
saddr	Categorical	21	0	Source IP address
sport	Integer	65541	0	Source port number
daddr	Categorical	84	0	Destination IP address
dport	Integer	7698	0	Destination port number
pkts	Integer	123	0	Number of packets in transaction
bytes	Integer	1633	0	Number of bytes in transaction
state	Categorical	11	0	Transaction state
state-number	Integer	11	0	Numerical representation of feature state
ltime	Float	383624	0	Record last time
seq	Integer	262212	0	Argus sequence number
dur	Float	612509	0	Record total duration
mean	Float	507089	0	Average duration of aggregated records
stddev	Float	421379	0	Standard deviation of aggregated records
sum	Float	934972	0	Total duration of aggregated records
min	Float	271147	0	Minimum duration of aggregated records
max	Float	594525	0	Maximum duration of aggregated records
spkts	Integer	91	0	Source-to-destination packet count
dpkts	Integer	62	0	Destination-to-source packet count
sbytes	Integer	1052	0	Source-to-destination byte count
dbytes	Integer	472	0	Destination-to-source byte count
rate	Float	139677	0	Total packets per second in transaction
srate	Float	119709	0	Source-to-destination packets per second
drate	Float	20714	0	Destination-to-source packets per second
tnbpsrcip	Integer	8639	0	Total number of bytes per source IP
tnbpdstip	Integer	7631	0	Total number of bytes per destination IP
tnp-psrcip	Integer	1522	0	Total number of packets per source IP
tnp-pdstip	Integer	1587	0	Total number of packets per destination IP
tnp-perproto	Integer	1560	0	Total number of packets per protocol
tnp-per-dport	Integer	1582	0	Total number of packets per dport
ar-p-proto-p-srcip	Float	46289	0	Average rate per protocol per source IP
ar-p-proto-p-dstip	Float	39186	0	Average rate per protocol per destination IP
n-in-conn-p-dstip	Integer	100	0	Number of inbound connections per destination IP
n-in-conn-p-srcip	Integer	100	0	Number of inbound connections per source IP
ar-p-proto-p-sport	Float	136207	0	Average rate per protocol per sport
ar-p-proto-p-dport	Float	42237	0	Average rate per protocol per dport
pkts-p-state-p-protocol-p-destip	Integer	1595	0	Number of packets grouped by the state of flows and protocols per destination IP
pkts-p-state-p-protocol-p-srcip	Integer	1526	0	Number of packets grouped by the state of flows and protocols per source IP
label	Numeric	5	0	Attack class label



Table A.2: CIC-IDS-2017 Dataset Description

Name	Type	Unique Values	Missing Values	Description
destination-port	Integer	53805	0	Destination port
flow-duration	Integer	1050899	0	Flow duration [ms]
total-fwd-packets	Integer	1432	0	Total forward packets
total-backward-packets	Integer	1747	0	Total backward packets
total-length-of-fwd-packets	Integer	17928	0	Total length of forward packets
total-length-of-bwd-packets	Integer	64698	0	Total length of backward packets
fwd-packet-length-max	Integer	5279	0	Forward packet maximum length
fwd-packet-length-min	Integer	384	0	Forward packet minimum length
fwd-packet-length-mean	Float	99716	0	Average length of forward packet
fwd-packet-length-std	Float	253909	0	Standard deviation of forward packet length
bwd-packet-length-max	Integer	4838	0	Backward packet maximum length
bwd-packet-length-min	Integer	583	0	Backward packet minimum length
bwd-packet-length-mean	Float	147614	0	Average length of backward packet
bwd-packet-length-std	Float	248869	0	Standard deviation of backward packet length
flow-bytes/s	Float	1593908	1358	Number of flow bytes per second
flow-packets/s	Float	1240164	0	Number of flow packet per second
flow-iat-mean	Float	1166311	0	Average time between two packets sent in the flow
flow-iat-std	Float	1056642	0	Standard deviation of time between two packets sent in the flow
flow-iat-max	Integer	580289	0	Maximum time between two packets sent in the flow
flow-iat-min	Integer	136316	0	Minimum time between two packets sent in the flow
fwd-iat-total	Integer	493098	0	Total time between two forward packets
fwd-iat-mean	Float	737737	0	Average time between two forward packets
fwd-iat-std	Float	700313	0	Standard deviation of time between two forward packets
fwd-iat-max	Integer	437316	0	Maximum time between two forward packets
fwd-iat-min	Integer	110631	0	Minimum time between two forward packets
bwd-iat-total	Integer	414928	0	Total time between two backward packets
bwd-iat-mean	Float	670824	0	Average time between two backward packets
bwd-iat-std	Float	709042	0	Standard deviation of time between two backward packets
bwd-iat-max	Integer	368285	0	Maximum time between two backward packets
bwd-iat-min	Integer	66074	0	Minimum time between two backward packets
fwd-psh-flags	Integer	2	0	Number of times the <i>PSH</i> flag was set in forward packets
bwd-psh-flags	Integer	1	0	Number of times the <i>PSH</i> flag was set in backward packets
fwd-urg-flags	Integer	2	0	Number of times the <i>URG</i> flag was set in forward packets
bwd-urg-flags	Integer	1	0	Number of times the <i>URG</i> flag was set in backward packets
fwd-header-length	Integer	3771	0	Forward header length
bwd-header-length	Integer	3945	0	Backward header length
fwd-packets/s	Float	1220423	0	Forward packets per second
bwd-packets/s	Float	1107886	0	Backward packets per second
min-packet-length	Integer	215	0	Minimum packet length
max-packet-length	Integer	5708	0	Maximum packet length
packet-length-mean	Float	215826	0	Average packet length
packet-length-std	Float	412246	0	Standard deviation of packet length
packet-length-variance	Float	405565	0	Variance of packet length
fin-flag-count	Integer	2	0	Number of packets with <i>FIN</i> flag
syn-flag-count	Integer	2	0	Number of packets with <i>SYN</i> flag
rst-flag-count	Integer	2	0	Number of packets with <i>RST</i> flag
psh-flag-count	Integer	2	0	Number of packets with <i>PSH</i> flag
ack-flag-count	Integer	2	0	Number of packets with <i>ACK</i> flag
urg-flag-count	Integer	2	0	Number of packets with <i>URG</i> flag
cwe-flag-count	Integer	2	0	Number of packets with <i>CWE</i> flag
ece-flag-count	Integer	2	0	Number of packets with <i>ECE</i> flag
down/up-ratio	Float	31	0	Download and upload ratio
average-packet-size	Float	212207	0	Average packet size
avg-fwd-segment-size	Float	99716	0	Average forward segment size observed
avg-bwd-segment-size	Float	147611	0	Average backward segment size observed
fwd-avg-bytes/bulk	Float	1	0	Average forward number of bytes bulk rate
fwd-avg-packets/bulk	Float	1	0	Average number of forward packets bulk rate
fwd-avg-bulk-rate	Float	1	0	Average number of forward bulk rate
bwd-avg-bytes/bulk	Float	1	0	Average number of backward bytes bulk rate
bwd-avg-packets/bulk	Float	1	0	Average number of backward packets bulk rate
bwd-avg-bulk-rate	Float	1	0	Average number of backward bulk rate
subflow-fwd-packets	Integer	1432	0	Total number of forward packets in a subflow
subflow-fwd-bytes	Integer	17928	0	Total number of forward bytes in a subflow
subflow-bwd-packets	Integer	1747	0	Total number of backward packets in a subflow
subflow-bwd-bytes	Integer	64738	0	Total number of backward bytes in a subflow
init-win-bytes-forward	Integer	12151	0	Total number of forward bytes sent in the initial window
init-win-bytes-backward	Integer	13112	0	Total number of backward bytes sent in the initial window
act-data-pkt-fwd	Integer	1093	0	Count of forward packets with at least one byte of TCP data payload
min-seg-size-forward	Integer	28	0	Minimum forward segment size observed
active-mean	Float	326325	0	Average time a flow was active before becoming idle
active-std	Float	202826	0	Standard deviation time a flow was active before becoming idle
active-max	Integer	299565	0	Maximum time a flow was active before becoming idle
active-min	Integer	175670	0	Minimum time a flow was active before becoming idle
idle-mean	Float	222016	0	Average time a flow was idle before becoming active
idle-std	Float	197616	0	Standard deviation time a flow was idle before becoming active
idle-max	Integer	149737	0	Maximum time a flow was idle before becoming active
idle-min	Integer	223888	0	Minimum time a flow was idle before becoming active
label	Categorical	15	0	Attack class label

Table A.3: CSE-CIC-IDS-2018 Dataset Description

Name	Type	Unique Values	Missing Values	Description
src-port	Integer	10879	900000	Source port number
dst-port	Integer	32939	0	Destination port number
protocol	Integer	7	0	Numerical representation of the protocol name
flow-duration	Integer	423968	0	Flow duration [ms]
tot-fwd-pkts	Integer	2429	0	Total forward packets
tot-bwd-pkts	Integer	1046	0	Total backward packets
totlen-fwd-pkts	Integer	9916	0	Total length of forward packets
totlen-bwd-pkts	Integer	18003	0	Total length of backward packets
fwd-pkt-len-max	Integer	2633	0	Forward packet maximum length
fwd-pkt-len-min	Integer	248	0	Forward packet minimum length
fwd-pkt-len-mean	Float	22475	0	Average length of forward packet
fwd-pkt-len-std	Float	31956	0	Standard deviation of forward packet length
bwd-pkt-len-max	Integer	1682	0	Backward packet maximum length
bwd-pkt-len-min	Integer	535	0	Backward packet minimum length
bwd-pkt-len-mean	Float	26774	0	Average length of backward packet
bwd-pkt-len-std	Float	30207	0	Standard deviation of backward packet length
flow-byts/s	Float	472359	2940	Number of flow bytes per second
flow-pkts/s	Float	461702	1551	Number of flow packet per second
flow-iat-mean	Float	441698	0	Average time between two packets sent in the flow
flow-iat-std	Float	453905	0	Standard deviation of time between two packets sent in the flow
flow-iat-max	Integer	351520	0	Maximum time between two packets sent in the flow
flow-iat-min	Integer	95069	0	Minimum time between two packets sent in the flow
fwd-iat-tot	Integer	278728	0	Total time between two forward packets
fwd-iat-mean	Float	303571	0	Average time between two forward packets
fwd-iat-std	Float	228918	0	Standard deviation of time between two forward packets
fwd-iat-max	Integer	261649	0	Maximum time between two forward packets
fwd-iat-min	Integer	120554	0	Minimum time between two forward packets
bwd-iat-tot	Integer	302173	0	Total time between two backward packets
bwd-iat-mean	Float	318048	0	Average time between two backward packets
bwd-iat-std	Float	376760	0	Standard deviation of time between two backward packets
bwd-iat-max	Integer	260687	0	Maximum time between two backward packets
bwd-iat-min	Integer	86938	0	Minimum time between two backward packets
fwd-psh-flags	Integer	5	0	Number of times the <i>PSH</i> flag was set in forward packets
bwd-psh-flags	Integer	3	0	Number of times the <i>PSH</i> flag was set in backward packets
fwd-urg-flags	Integer	4	0	Number of times the <i>URG</i> flag was set in forward packets
bwd-urg-flags	Integer	3	0	Number of times the <i>URG</i> flag was set in backward packets
fwd-header-len	Integer	3041	0	Forward header length
bwd-header-len	Integer	2031	0	Backward header length
fwd-pkts/s	Float	458364	0	Forward packets per second
bwd-pkts/s	Float	381144	0	Backward packets per second
pkt-len-min	Integer	174	0	Minimum packet length
pkt-len-max	Integer	2113	0	Maximum packet length
pkt-len-mean	Float	39719	0	Average packet length
pkt-len-std	Float	45332	0	Standard deviation of packet length
pkt-len-var	Float	44982	0	Variance of packet length
fin-flag-cnt	Integer	5	0	Number of packets with <i>FIN</i> flag
syn-flag-cnt	Integer	5	0	Number of packets with <i>SYN</i> flag
rst-flag-cnt	Integer	5	0	Number of packets with <i>RST</i> flag
psh-flag-cnt	Integer	5	0	Number of packets with <i>PSH</i> flag
ack-flag-cnt	Integer	5	0	Number of packets with <i>ACK</i> flag
urg-flag-cnt	Integer	5	0	Number of packets with <i>URG</i> flag
cwe-flag-count	Integer	4	0	Number of packets with <i>CWE</i> flag
ece-flag-cnt	Integer	5	0	Number of packets with <i>ECE</i> flag
down/up-ratio	Float	75	0	Download and upload ratio
pkt-size-avg	Float	40410	0	Average packet size
fwd-seg-size-avg	Float	22475	0	Average forward segment size observed
bwd-seg-size-avg	Float	26774	0	Average backward segment size observed
fwd-byts/b-avg	Float	3	0	Average forward number of bytes bulk rate
fwd-pkts/b-avg	Float	3	0	Average number of forward packets bulk rate
fwd-blk-rate-avg	Float	3	0	Average number of forward bulk rate
bwd-byts/b-avg	Float	3	0	Average number of backward bytes bulk rate
bwd-pkts/b-avg	Float	3	0	Average number of backward packets bulk rate
bwd-blk-rate-avg	Float	3	0	Average number of backward bulk rate
subflow-fwd-pkts	Integer	2429	0	Total number of forward packets in a subflow
subflow-fwd-byts	Integer	9916	0	Total number of forward bytes in a subflow
subflow-bwd-pkts	Integer	1046	0	Total number of backward packets in a subflow
subflow-bwd-byts	Integer	18003	0	Total number of backward bytes in a subflow
init-fwd-win-byts	Integer	4638	0	Total number of forward bytes sent in the initial window
init-bwd-win-byts	Integer	4547	0	Total number of backward bytes sent in the initial window
fwd-act-data-pkts	Integer	1943	0	Count of forward packets with at least one byte of TCP data payload
fwd-seg-size-min	Integer	20	0	Minimum forward segment size observed
active-mean	Float	68459	0	Average time a flow was active before becoming idle
active-std	Float	47568	0	Standard deviation of the time a flow was active before becoming idle
active-max	Integer	67548	0	Maximum time a flow was active before becoming idle
active-min	Integer	51339	0	Minimum time a flow was active before becoming idle
idle-mean	Float	93401	0	Average time a flow was idle before becoming active
idle-std	Float	53341	0	Standard deviation of the time a flow was idle before becoming active
idle-max	Integer	82800	0	Maximum time a flow was idle before becoming active
idle-min	Integer	89464	0	Minimum time a flow was idle before becoming active
label	Categorical	14	0	Attack class label

Table A.4: Edge-IIoT Dataset Description

Name	Type	Unique Values	Missing Values	Description
ip.src-host	Categorical	19088	0	Source IP address
ip.dst-host	Categorical	8084	0	Destination IP address
arp.dst.proto-ipv4	Categorical	8	0	Target IP address
arp.opcode	Integer	3	0	Opcode
arp.hw.size	Integer	2	0	Hardware size
arp.src.proto-ipv4	Categorical	8	0	Sender IP address
icmp.checksum	Integer	13186	0	ICMP checksum
icmp.seq-le	Integer	13823	0	ICMP sequence number
icmp.transmit-timestamp	Integer	84	0	Transmit timestamp
icmp.unused	Categorical	1	0	Unused
http.file-data	Categorical	496	0	File data
http.content-length	Integer	33	0	Content length
http.request.uri.query	Categorical	1665	0	Request uniform resource identifier (URI) query
http.request.method	Categorical	6	0	Request method
http.referer	Categorical	4	0	Referrer
http.request.full-uri	Categorical	4073	0	Full request URI
http.request.version	Categorical	8	0	HTTP request version
http.response	Boolean	2	0	If response, it will be set as 1; otherwise, 0
http.tls-port	Integer	1	0	Unencrypted HTTP protocol
tcp.ack	Integer	27929	0	Acknowledgment number
tcp.ack-raw	Integer	94716	0	Raw acknowledgment number
tcp.checksum	Integer	55513	0	TCP checksum
tcp.connection.fin	Boolean	2	0	If connection finish ( <i>FIN</i> ), it will be set as 1; otherwise, 0
tcp.connection.rst	Boolean	2	0	If connection reset ( <i>RST</i> ), it will be set as 1; otherwise, 0
tcp.connection.syn	Boolean	2	0	If connection establish request ( <i>SYN</i> ), it will be set as 1; otherwise, 0
tcp.connection.synack	Boolean	2	0	If connection establish acknowledge ( <i>SYN-ACK</i> ), it will be set as 1; otherwise, 0
tcp.dstport	Integer	23188	0	Destination port
tcp.flags	Integer	9	0	Flags
tcp.flags.ack	Boolean	2	0	If TCP acknowledgment, it will be set as 1; otherwise, 0
tcp.len	Integer	786	0	TCP segment length
tcp.options	Categorical	73139	0	TCP options
tcp.payload	Categorical	27369	0	TCP payload
tcp.seq	Integer	18199	0	TCP sequence number
tcp.srport	Integer	32186	0	Source port
udp.port	Integer	32	0	UDP source or destination port
udp.stream	Integer	14492	0	Stream index
udp.time-delta	Integer	39	0	Time since previous frame
dns.qry.name	Categorical	8	0	Query name
dns.qry.name.len	Integer	1430	0	Query name length
dns.qry.qu	Integer	66	0	Query <i>QU</i> question
dns.qry.type	Integer	1	0	Query type
dns.retransmission	Integer	4	0	Retransmission
dns.retransmit-request	Boolean	2	0	If DNS query retransmission, it will be set as 1; otherwise, 0
dns.retransmit-request-in	Integer	1	0	DNS query retransmission frame number
mqtt.conack.flags	Integer	3	0	MQTT acknowledgment flags
mqtt.conflog.cleansess	Boolean	2	0	If clean session flag, it will be set as 1; otherwise, 0
mqtt.conflogs	Integer	2	0	Connection flags
mqtt.hdrflags	Integer	5	0	Header flags
mqtt.len	Integer	4	0	MQTT message length
mqtt.msg-decoded-as	Integer	1	0	MQTT message decoded
mqtt.msg	Categorical	117	0	MQTT message
mqtt.msgtype	Integer	5	0	MQTT message type
mqtt.proto-len	Integer	2	0	MQTT protocol name length
mqtt.protoname	Categorical	3	0	MQTT protocol name
mqtt.topic	Categorical	3	0	MQTT topic
mqtt.topic-len	Integer	2	0	MQTT topic length
mqtt.ver	Integer	2	0	MQTT version
mbtcp.len	Integer	1	0	Modbus/TCP length
mbtcp.trans-id	Integer	1	0	Modbus/TCP transaction identifier
mbtcp.unit-id	Integer	1	0	Modbus/TCP unit identifier
label	Categorical	15	0	Attack class label

Table A.5: IEC61850-Security Dataset Description

Name	Type	Unique Values	Missing Values	Description
id.orig-h	Categorical	57	0	Source IP address
id.orig-p	Integer	8	0	Source port number
id.resp-h	Categorical	41	0	Destination IP address
id.resp-p	Integer	9	0	Destination port number
proto	Categorical	2	0	Protocol name (UDP, ICMP)
service	Categorical	2	1931	Service name (DHCP, DNS)
duration	Float	3604	909	Duration of the connection [s]
orig-bytes	Integer	654	909	Number of payload bytes originator sent
resp-bytes	Integer	1	909	Number of payload bytes responder sent
conn-state	Categorical	2	0	Connection state
local-orig	Boolean	0	4558	If the connection originated locally, it will be set as 1;
local-resp	Boolean	0	4558	If the connection responded locally, it will be set as 1;
missed-bytes	Integer	1	0	Number of bytes missed (packet loss)
history	Categorical	1	1889	Connection state history
orig-pkts	Integer	98	0	Number of packets originator sent
orig-ip-bytes	Integer	696	0	Number of originator IP bytes (via IP <i>total-length</i> header field)
resp-pkts	Integer	1	0	Number of packets responder sent
resp-ip-bytes	Integer	1	0	Number of responder IP bytes (via IP <i>total-length</i> header field)
tunnel-parents	Categorical	0	4558	If tunneled, connection UID value of encapsulating parent(s)
label	Categorical	5	0	Attack class label

Table A.6: IoT-23 Dataset Description

Name	Type	Unique Values	Missing Values	Description
id.orig-h	Categorical	3266	0	Source IP address
id.orig-p	Integer	62278	0	Source port number
id.resp-p	Integer	29263	0	Destination port number
proto	Categorical	3	0	Protocol name (TCP, UDP, ICMP)
service	Categorical	6	1434046	Service name (dynamic host configuration protocol (DHCP), DNS, secure sockets layer (SSL), HTTP, SSH)
duration	Float	46584	900109	Duration of the connection [s]
orig-bytes	Integer	279	900109	Number of payload bytes originator sent
resp-bytes	Integer	509	900109	Number of payload bytes responder sent
conn-state	Categorical	13	0	Connection state
local-orig	Boolean	0	1444706	If the connection originated locally, it will be set as 1;
local-resp	Boolean	0	1444706	If the connection responded locally, it will be set as 1;
missed-bytes	Integer	21	0	Number of bytes missed (packet loss)
history	Categorical	144	3651	Connection state history
orig-pkts	Integer	176	0	Number of packets originator sent
orig-ip-bytes	Integer	928	0	Number of originator IP bytes (via IP <i>total-length</i> header field)
resp-pkts	Integer	134	0	Number of packets responder sent
resp-ip-bytes	Integer	932	0	Number of responder IP bytes (via IP <i>total-length</i> header field)
label	Categorical	12	0	Attack class label

Table A.7: ISCX-IDS-2012 Dataset Description

Name	Type	Unique Values	Missing Values	Description
appname	Categorical	107	0	Application name
totalsourcebytes	Integer	28029	0	Number of bytes at source
totaldestinationbytes	Integer	146732	0	Number of bytes at destination
totaldestinationpackets	Integer	3317	0	Number of packets at destination
totalsourcepackets	Integer	2430	0	Number of packets at source
sourcepayloadasbase64	Categorical	528277	1098051	Source payload in base 64
sourcepayloadasutf	Categorical	356549	1182855	Source payload
destinationpayloadasbase64	Categorical	688095	1188626	Destination payload in base 64
destinationpayloadasutf	Categorical	671783	1188687	Destination payload
direction	Categorical	4	0	Flow Direction ( <i>L2L</i> , <i>L2R</i> , <i>R2R</i> , <i>R2L</i> )
sourcetcpflagsdescription	Categorical	23	430943	TCP flags at source
destinationtcpflagsdescription	Categorical	27	493422	TCP flags at destination
source	Categorical	2478	0	Source IP address
protocolname	Categorical	6	0	Protocol name (UDP/IP, TCP/IP, ICMP/IP, ICMP/IPv6, IP, IGMP)
sourceport	Integer	64482	0	Source port number
destination	Categorical	34555	0	Destination IP address
destinationport	Integer	24238	0	Destination port number
startdatetime	Timestamp	314360	0	Connection start (date and time)
stopdatetime	Timestamp	339485	0	Connection end (date and time)
starttime	Float	171970	1888689	Connection start
sensorinterfaceid	Integer	1	1879616	Sensor Interface ID
label	Boolean	2	0	If is an attack, it will be set as 1; otherwise, 0

Table A.8: KDD99 Dataset Description

Name	Type	Unique Values	Missing Values	Description
duration	Integer	9883	0	Time's duration of the connection [s]
protocol-type	Categorical	3	0	Protocol used (TCP, UDP, ...)
service	Categorical	70	0	Network service on the destination (HTTP, TELNET, ...)
flag	Categorical	11	0	Status of the connection (Error or Normal)
src-bytes	Integer	7195	0	Number of data bytes transferred from source to destination
dst-bytes	Integer	21493	0	Number of data bytes transferred from destination to source
land	Boolean	2	0	If the source and destination are the same, it will be set as 1; otherwise, 0
wrong-fragment	Integer	3	0	Number of wrong fragments in a connection
urgent	Integer	6	0	Number of urgent packets, which means packets with urgent bit activated
hot	Integer	30	0	Number of hot indicators, which means entering in a system directory
num-failed-logins	Integer	6	0	Number of failed login attempts
logged-in	Boolean	2	0	If successful login, it will be set as 1; otherwise, 0
num-compromised	Integer	98	0	Number of compromised conditions
root-shell	Boolean	2	0	If the root shell is obtained, it will be set as 1; otherwise, 0
su-attempted	Boolean	2	0	If <i>su root</i> command is attempted, it will be set as 1; otherwise, 0
num-root	Integer	93	0	Number of operations performed as root
num-file-creations	Integer	42	0	Number of file creation operations
num-shells	Integer	3	0	Number of shell prompts in a connection
num-access-files	Integer	10	0	Number of operations on access control files
num-outbound-cmds	Integer	1	0	Number of outbound commands in a FTP session
is-host-login	Boolean	2	0	If login as root or admin, it will be set as 1; otherwise, 0
is-guest-login	Boolean	2	0	If login as guest, it will be set as 1; otherwise, 0
count	Integer	512	0	Number of connections to the same destination host
srv-count	Integer	512	0	Number of connection to the same service
error-rate	Float	96	0	Percentage of connections that have activated flag <i>s0</i> , <i>s1</i> , <i>s2</i> or <i>s3</i> , among the connections aggregated in count
srv-error-rate	Float	87	0	Percentage of connection that have activated flag <i>s0</i> , <i>s1</i> , <i>s2</i> or <i>s3</i> , among the connections aggregated in <i>srv</i> count
reror-rate	Float	89	0	Percentage of connections that have activated flag <i>REJ</i> , among the connections, aggregated in count
srv-reror-rate	Float	76	0	Percentage of connections that have activated flag <i>REJ</i> , among the connections aggregated in <i>srv</i> count
same-srv-rate	Float	101	0	Percentage of connections that were to the same services, among the connections aggregated in count
diff-srv-rate	Float	95	0	Percentage of connections that were to the different services, among the connections aggregated in count
srv-diff-host-rate	Float	72	0	Percentage of connections that were different destination machines among the connections aggregated in <i>srv</i> count
dst-host-count	Integer	256	0	Number of connections having the same destination host IP address
dst-host-srv-count	Integer	256	0	Number of connections having same port number
dst-host-same-srv-rate	Float	101	0	Percentage of connections that were to the same service among the connections aggregated in <i>dst</i> host count
dst-host-diff-srv-rate	Float	101	0	Percentage of connections that were different services among the connections aggregated in <i>dst</i> host count
dst-host-same-src-port-rate	Float	101	0	Percentage of connections that were to the same source port among the connections aggregated in <i>dst</i> host <i>srv</i> count
dst-host-srv-diff-host-rate	Float	76	0	Percentage of connections that were to the different destination machines among the connections aggregated in <i>dst</i> host <i>srv</i> count
dst-host-error-rate	Float	101	0	Percentage of connections that have activated flag <i>s0</i> , <i>s1</i> , <i>s2</i> or <i>s3</i> , among the connections aggregated in <i>dst</i> host count
dst-host-srv-error-rate	Float	100	0	Percentage of connections that have activated flag <i>s0</i> , <i>s1</i> , <i>s2</i> or <i>s3</i> , among the connections aggregated in <i>dst</i> host <i>srv</i> count
dst-host-reror-rate	Float	101	0	Percentage of connections that have activated flag <i>REJ</i> , among the connections, aggregated in <i>dst</i> host count
dst-host-srv-reror-rate	Float	101	0	Percentage of connections that have activated flag <i>REJ</i> , among the connections, aggregated in <i>dst</i> host <i>srv</i> count
label	Categorical	23	0	Attack class label

Table A.9: NSL-KDD Dataset Description

Name	Type	Unique Values	Missing Values	Description
duration	Integer	3424	0	Time's duration of the connection [s]
protocol-type	Categorical	3	0	Protocol used (TCP, UDP, ...)
service	Categorical	70	0	Network service on the destination (HTTP, TELNET, ...)
flag	Categorical	11	0	Status of the connection (Error or Normal)
src-bytes	Integer	3601	0	Number of data bytes transferred from source to destination
dst-bytes	Integer	10401	0	Number of data bytes transferred from destination to source
land	Boolean	2	0	If the source and destination are the same, it will be set as 1; otherwise, 0
wrong-fragment	Integer	3	0	Number of wrong fragments in a connection
urgent	Integer	4	0	Number of urgent packets, which means packets with urgent bit activated
hot	Integer	29	0	Number of hot indicators, which means entering in a system directory
num-failed-logins	Integer	6	0	Number of failed login attempts
logged-in	Boolean	2	0	If successful login, it will be set as 1; otherwise, 0
num-compromised	Integer	96	0	Number of compromised conditions
root-shell	Boolean	2	0	If the root shell is obtained, it will be set as 1; otherwise, 0
su-attempted	Boolean	2	0	If <i>su root</i> command is attempted, it will be set as 1; otherwise, 0
num-root	Integer	91	0	Number of operations performed as root
num-file-creations	Integer	36	0	Number of file creation operations
num-shells	Integer	4	0	Number of shell prompts in a connection
num-access-files	Integer	10	0	Number of operations on access control files
num-outbound-cmds	Integer	1	0	Number of outbound commands in a FTP session
is-host-login	Boolean	2	0	If login as root or admin, it will be set as 1; otherwise, 0
is-guest-login	Boolean	2	0	If login as guest, it will be set as 1; otherwise, 0
count	Integer	512	0	Number of connections to the same destination host
srv-count	Integer	512	0	Number of connection to the same service
error-rate	Float	99	0	Percentage of connections that have activated flag <i>s0</i> , <i>s1</i> , <i>s2</i> or <i>s3</i> , among the connections aggregated in count
srv-error-rate	Float	94	0	Percentage of connection that have activated flag <i>s0</i> , <i>s1</i> , <i>s2</i> or <i>s3</i> , among the connections aggregated in <i>srv</i> count
reror-rate	Float	98	0	Percentage of connections that have activated flag <i>REJ</i> , among the connections, aggregated in count
srv-reror-rate	Float	95	0	Percentage of connections that have activated flag <i>REJ</i> , among the connections aggregated in <i>srv</i> count
same-srv-rate	Float	101	0	Percentage of connections that were to the same services, among the connections aggregated in count
diff-srv-rate	Float	101	0	Percentage of connections that were to the different services, among the connections aggregated in count
srv-diff-host-rate	Float	87	0	Percentage of connections that were different destination machines among the connections aggregated in <i>srv</i> count
dst-host-count	Integer	256	0	Number of connections having the same destination host IP address
dst-host-srv-count	Integer	256	0	Number of connections having same port number
dst-host-same-srv-rate	Float	101	0	Percentage of connections that were to the same service among the connections aggregated in <i>dst</i> host count
dst-host-diff-srv-rate	Float	101	0	Percentage of connections that were different services among the connections aggregated in <i>dst</i> host count
dst-host-same-src-port-rate	Float	101	0	Percentage of connections that were to the same source port among the connections aggregated in <i>dst</i> host <i>srv</i> count
dst-host-srv-diff-host-rate	Float	75	0	Percentage of connections that were to the different destination machines among the connections aggregated in <i>dst</i> host <i>srv</i> count
dst-host-error-rate	Float	101	0	Percentage of connections that have activated flag <i>s0</i> , <i>s1</i> , <i>s2</i> or <i>s3</i> , among the connections aggregated in <i>dst</i> host count
dst-host-srv-error-rate	Float	101	0	Percentage of connections that have activated flag <i>s0</i> , <i>s1</i> , <i>s2</i> or <i>s3</i> , among the connections aggregated in <i>dst</i> host <i>srv</i> count
dst-host-reror-rate	Float	101	0	Percentage of connections that have activated flag <i>REJ</i> , among the connections, aggregated in <i>dst</i> host count
dst-host-srv-reror-rate	Float	101	0	Percentage of connections that have activated flag <i>REJ</i> , among the connections, aggregated in <i>dst</i> host <i>srv</i> count
label	Categorical	2	0	Attack class label

Table A.10: TON-IoT Dataset Description

Name	Type	Unique Values	Missing Values	Description
src-ip	Categorical	11536	0	Source IP addresses which originate endpoints' IP addresses
src-port	Integer	53671	0	Source port which originate endpoint's TCP/UDP ports
dst-ip	Categorical	5268	0	Destination IP addresses which respond to endpoint's IP addresses
dst-port	Integer	2666	0	Destination ports which respond to endpoint's TCP/UDP ports
proto	Categorical	3	0	Transport layer protocols of flow connections (TCP, UDP, ICMP)
service	Categorical	9	280216	Dynamically detected protocols (DNS, HTTP, SSL, ...)
duration	Float	124727	0	The time of the packet connections, which is estimated by subtracting <i>time of last packet seen</i> and <i>time of first packet seen</i>
src-bytes	Integer	3056	0	Source bytes which are originated payload bytes of TCP sequence numbers
dst-bytes	Integer	3511	0	Destination bytes which are responded payload bytes from TCP sequence numbers
conn-state	Categorical	13	0	Various connection states, such as SO (connection without replay), SI (connection established), REJ (connection rejected), and OTH (other)
missed-bytes	Integer	929	0	Number of missing bytes in content gaps
src-pkts	Integer	414	0	Number of original packets which is estimated from source systems
src-ip-bytes	Integer	4549	0	Number of original IP bytes which is the total length of IP header field of source systems
dst-pkts	Integer	332	0	Number of destination packets which is estimated from destination systems
dst-ip-bytes	Integer	4577	0	Number of destination IP bytes which is the total length of IP header field of destination systems
dns-query	Categorical	14148	366019	Domain name subjects of the DNS queries
dns-qclass	Integer	3	0	Value which specifies the DNS query classes
dns-qtype	Integer	12	0	Value which specifies the DNS query types
dns-rcode	Integer	5	0	Response code values in the DNS response
dns-aa	Boolean	2	365158	If authoritative answers of DNS, it will be set as True; otherwise, False
dns-rd	Boolean	2	365158	If recursion desired of DNS, it will be set as True; otherwise, False
dns-ra	Boolean	2	365158	If recursion available of DNS, it will be set as True; otherwise, False
dns-rejected	Boolean	2	365158	If there is a DNS rejection, it will be set as True; otherwise, False
ssl-version	Categorical	3	460737	SSL version which is offered by the server
ssl-cipher	Categorical	5	460737	SSL cipher suite which the server chose
ssl-resumed	Boolean	2	460352	If session that can be used to initiate new connections, it will be set as True; otherwise, False
ssl-established	Boolean	2	460352	If connection between two parties is established, it will be set as True; otherwise, False
ssl-subject	Categorical	5	461034	Subject of the X.509 cert offered by the server
ssl-issuer	Categorical	4	461034	Trusted owner/originator of the SSL and digital certificate (certificate authority)
http-trans-depth	Integer	10	460796	Pipelined depth into the HTTP connected
http-method	Categorical	3	460809	HTTP request methods (GET, POST, HEAD)
http-uri	Categorical	73	460809	URLs used in the HTTP request
http-version	Float	1	460801	The HTTP version utilized
http-request-body-len	Integer	6	0	Actual uncompressed content sizes of the data transferred from the HTTP client
http-response-body-len	Integer	68	0	Actual uncompressed content sizes of the data transferred from the HTTP server
http-status-code	Integer	8	0	Status codes returned by the HTTP server
http-user-agent	Categorical	35	460809	Values of the User-Agent header in the HTTP protocol
http-orig-mime-types	Categorical	2	461029	Ordered vectors of mime types from source system in the HTTP protocol
http-resp-mime-types	Categorical	9	460883	Ordered version of mime types from destination system in the HTTP protocol
weird-name	Categorical	11	459749	Names of anomalies/violations related to protocols that happened
weird-addl	Categorical	3	460290	Additional information is associated to protocol anomalies/violations
weird-notice	Boolean	1	459749	If the violation/anomaly was turned into a notice, it will be set as True; otherwise, False
label	Categorical	10	0	Attack class label



Table A.11: UNSW-NB15 Dataset Description

Name	Type	Unique Values	Missing Values	Description
dur	Float	109945	0	Connection duration [ms]
proto	Categorical	133	0	Protocol name (TCP, UDP, ...)
service	Categorical	13	0	Service name (HTTP, FTP, ...)
state	Categorical	11	0	Connection state (CON, CLO, ...)
spkts	Integer	646	0	Source to destination packet count
dpkts	Integer	627	0	Destination to source packet count
sbytes	Integer	9382	0	Source to destination bytes
dbytes	Integer	8653	0	Destination to source bytes
rate	Float	115763	0	Packets transmission flow rate
sttl	Integer	13	0	Source to destination time to live
dttl	Integer	9	0	Destination to source time to live
sload	Float	121356	0	Source bits per second
dload	Float	116380	0	Destination bits per second
sloss	Integer	490	0	Source packets retransmitted or dropped
dloss	Integer	476	0	Destination packets retransmitted or dropped
sinpkt	Float	114318	0	Source inter-packet arrival time [ms]
dinpkt	Float	110270	0	Destination inter-packet arrival time [ms]
sjit	Float	117101	0	Source jitter [ms]
djit	Float	114861	0	Destination jitter [ms]
swin	Integer	22	0	Source TCP window advertisement
stcpb	Integer	114473	0	Source TCP sequence number
dtcpb	Integer	114187	0	Destination TCP sequence number
dwin	Integer	19	0	Destination TCP window advertisement
tcprrt	Float	63878	0	The sum of <i>synack</i> and <i>ackdat</i> of the TCP
synack	Float	57366	0	The time between the SYN and the SYN_ACK packets of the TCP
ackdat	Float	53248	0	The time between the SYN_ACK and the ACK packets of the TCP
smean	Integer	1377	0	Mean of the flow packet size transmitted by the source
dmean	Integer	1362	0	Mean of the flow packet size transmitted by the destination
trans-depth	Integer	14	0	The depth into the connection of HTTP request/response transaction
response-body-len	Integer	2819	0	The content size of the data transferred from the server's HTTP service
ct-srv-src	Integer	57	0	Number of connections that contain the same service and source address in 100 connections according to the last time
ct-state-ttl	Integer	7	0	Number for each state according to the specific range of values for source/destination time to live
ct-dst-ltm	Integer	52	0	Number of connections of the same destination address in 100 connections according to the last time
ct-src-dport-ltm	Integer	52	0	Number of records of the same source IP and the destination port in 100 records according to the last time
ct-dst-sport-ltm	Integer	35	0	Number of records of the same destination IP and the source port in 100 records according to the last time
ct-dst-src-ltm	Integer	58	0	Number of records of the same source IP and the destination IP in 100 records according to the last time
is-ftp-login	Boolean	2	0	If user and password access the FTP session, it will be set as 1; otherwise, 0
ct-ftp-cmd	Integer	4	0	Number of flows with a command in FTP session
ct-flw-http-mthd	Integer	11	0	Number of flows with methods like GET and POST in HTTP service
ct-src-ltm	Integer	52	0	Number of connections of the same source address in 100 connections according to the last time
ct-srv-dst	Integer	57	0	Number of connections that contain the same service and destination address in 100 connections according to the last time
is-sm-ips-ports	Boolean	2	0	If source equals to destination IP addresses and port numbers are equal, it will be set as 1; otherwise, 0
label	Categorical	10	0	Attack class label



# Appendix B

## Results Tables

Table B.1: Time-Series Classification in Binary Scenario (Full Table)

Dataset	Model	Train Time [s]	Test Time [s]	Accuracy	Precision	Recall	F1-Score
BOT-IoT	CIF	1011.44 ± 10.09	44.98 ± 5.49	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	MiniROCKET	83.32 ± 1.24	2.68 ± 0.07	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	RISE	93.09 ± 2.27	18.72 ± 0.5	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	ROCKET	210.4 ± 0.98	18.09 ± 0.85	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	TSF	8.48 ± 0.4	2.89 ± 0.16	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	WEASEL-v2	265.55 ± 5.41	13.83 ± 0.13	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	cBOSS	244.35 ± 1.36	207.68 ± 1.4	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
CIC-IDS-2017	CIF	1517.43 ± 18.25	445.15 ± 20.86	0.99 ± 0.0	0.97 ± 0.01	0.97 ± 0.01	0.97 ± 0.01
	MiniROCKET	103.92 ± 0.77	4.79 ± 0.36	0.99 ± 0.0	0.98 ± 0.0	0.98 ± 0.01	0.98 ± 0.0
	RISE	161.51 ± 4.86	21.71 ± 0.55	0.99 ± 0.0	0.98 ± 0.01	0.97 ± 0.01	0.97 ± 0.01
	ROCKET	322.14 ± 4.25	28.23 ± 0.7	0.99 ± 0.0	0.98 ± 0.0	0.98 ± 0.01	0.98 ± 0.0
	TSF	16.2 ± 0.41	3.8 ± 0.33	0.99 ± 0.0	0.99 ± 0.0	0.98 ± 0.01	0.99 ± 0.01
	WEASEL-v2	393.05 ± 5.65	29.89 ± 1.5	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.01	0.99 ± 0.01
	cBOSS	635.86 ± 2.3	255.31 ± 1.93	0.99 ± 0.0	0.97 ± 0.01	0.98 ± 0.01	0.98 ± 0.01
CSE-CIC-IDS-2018	CIF	1821.9 ± 32.73	369.25 ± 16.55	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	MiniROCKET	118.57 ± 0.92	4.26 ± 0.12	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	RISE	163.95 ± 2.41	22.18 ± 0.5	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	ROCKET	360.21 ± 4.11	29.97 ± 0.81	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	TSF	18.17 ± 0.22	3.94 ± 0.09	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	WEASEL-v2	425.71 ± 2.26	28.35 ± 0.94	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	cBOSS	1079.63 ± 5.91	278.11 ± 2.89	0.99 ± 0.0	0.99 ± 0.0	1.0 ± 0.0	0.99 ± 0.0
Edge-IoT	CIF	1227.35 ± 9.49	38.77 ± 2.3	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	MiniROCKET	98.77 ± 0.48	3.66 ± 0.2	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	RISE	105.04 ± 0.87	18.87 ± 0.34	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	ROCKET	265.91 ± 2.24	22.26 ± 0.94	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	TSF	9.48 ± 0.62	3.48 ± 0.18	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	WEASEL-v2	330.02 ± 2.26	19.47 ± 0.8	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	cBOSS	422.55 ± 17.64	244.51 ± 6.46	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
IEC61850-Security	CIF	295.58 ± 4.39	370.4 ± 17.06	0.73 ± 0.01	0.73 ± 0.0	0.95 ± 0.02	0.83 ± 0.01
	MiniROCKET	17.63 ± 0.43	0.72 ± 0.02	0.78 ± 0.01	0.8 ± 0.01	0.91 ± 0.01	0.85 ± 0.01
	RISE	45.51 ± 0.84	7.97 ± 0.13	0.76 ± 0.01	0.8 ± 0.01	0.86 ± 0.01	0.83 ± 0.01
	ROCKET	46.77 ± 1.01	3.44 ± 0.11	0.77 ± 0.01	0.79 ± 0.01	0.9 ± 0.01	0.84 ± 0.01
	TSF	4.56 ± 0.49	1.64 ± 0.42	0.77 ± 0.01	0.81 ± 0.01	0.86 ± 0.02	0.83 ± 0.01
	WEASEL-v2	75.34 ± 0.71	3.53 ± 0.13	0.72 ± 0.02	0.77 ± 0.01	0.86 ± 0.02	0.81 ± 0.01
	cBOSS	21.92 ± 0.41	78.93 ± 0.11	0.75 ± 0.01	0.77 ± 0.01	0.9 ± 0.01	0.83 ± 0.01
ISCX-IDS-2012	CIF	679.57 ± 7.77	162.68 ± 9.05	1.0 ± 0.0	1.0 ± 0.01	0.97 ± 0.02	0.98 ± 0.01
	MiniROCKET	80.64 ± 0.89	1.84 ± 0.32	1.0 ± 0.0	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.01
	RISE	102.57 ± 2.56	17.57 ± 0.44	1.0 ± 0.0	1.0 ± 0.01	0.95 ± 0.02	0.98 ± 0.01
	ROCKET	153.89 ± 1.99	9.02 ± 0.52	1.0 ± 0.0	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.01
	TSF	7.01 ± 0.66	2.11 ± 0.08	1.0 ± 0.0	1.0 ± 0.0	0.98 ± 0.02	0.99 ± 0.01
	WEASEL-v2	205.36 ± 5.72	9.23 ± 0.18	1.0 ± 0.0	0.99 ± 0.01	0.98 ± 0.02	0.98 ± 0.01
	cBOSS	104.99 ± 10.16	187.85 ± 2.33	1.0 ± 0.0	0.99 ± 0.01	0.98 ± 0.02	0.98 ± 0.01
IoT-23	CIF	719.16 ± 9.11	447.6 ± 51.05	0.95 ± 0.01	0.94 ± 0.01	1.0 ± 0.0	0.97 ± 0.0
	MiniROCKET	82.01 ± 2.01	1.62 ± 0.24	0.99 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	RISE	98.58 ± 0.83	17.21 ± 0.45	0.95 ± 0.0	0.97 ± 0.0	0.97 ± 0.0	0.97 ± 0.0
	ROCKET	145.03 ± 0.58	8.0 ± 0.59	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	TSF	5.28 ± 0.15	2.23 ± 0.2	0.98 ± 0.0	0.98 ± 0.0	1.0 ± 0.0	0.99 ± 0.0
	WEASEL-v2	197.36 ± 2.15	8.25 ± 0.42	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	cBOSS	105.37 ± 11.74	186.05 ± 0.53	0.99 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
KDD99	CIF	1104.15 ± 13.03	118.08 ± 8.11	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	MiniROCKET	93.45 ± 0.89	2.81 ± 0.26	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	RISE	102.88 ± 0.96	18.28 ± 0.2	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	ROCKET	217.18 ± 2.88	15.79 ± 1.19	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	TSF	8.76 ± 0.66	2.87 ± 0.29	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	WEASEL-v2	277.8 ± 4.68	12.97 ± 0.32	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	cBOSS	180.16 ± 1.82	228.96 ± 1.73	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
NSL-KDD	CIF	1219.44 ± 14.56	754.62 ± 16.63	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	MiniROCKET	112.91 ± 1.64	2.87 ± 0.31	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	RISE	133.79 ± 2.9	18.5 ± 0.25	0.99 ± 0.0	0.99 ± 0.0	0.98 ± 0.0	0.99 ± 0.0
	ROCKET	245.0 ± 3.14	16.76 ± 0.92	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	TSF	11.05 ± 0.71	2.97 ± 0.31	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	WEASEL-v2	309.22 ± 6.22	13.64 ± 0.47	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	cBOSS	229.07 ± 1.09	232.35 ± 0.57	0.98 ± 0.0	0.98 ± 0.01	0.99 ± 0.0	0.98 ± 0.0
TON-IoT	CIF	1175.99 ± 11.89	405.78 ± 15.59	0.99 ± 0.0	0.97 ± 0.01	0.99 ± 0.0	0.98 ± 0.01
	MiniROCKET	97.52 ± 1.97	2.95 ± 0.34	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	RISE	106.22 ± 0.77	18.37 ± 0.32	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0
	ROCKET	227.18 ± 2.55	16.07 ± 0.87	1.0 ± 0.0	0.99 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	TSF	8.17 ± 0.1	2.84 ± 0.16	0.99 ± 0.0	0.99 ± 0.01	0.99 ± 0.0	0.99 ± 0.0
	WEASEL-v2	274.38 ± 2.67	12.58 ± 0.45	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
	cBOSS	171.55 ± 0.65	225.36 ± 2.89	0.99 ± 0.0	0.98 ± 0.0	0.99 ± 0.0	0.98 ± 0.0
UNSW-NB15	CIF	1255.72 ± 34.07	1966.39 ± 38.18	0.93 ± 0.01	0.94 ± 0.0	0.95 ± 0.01	0.95 ± 0.0
	MiniROCKET	103.44 ± 1.1	2.79 ± 0.16	0.92 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.01
	RISE	127.81 ± 1.67	18.48 ± 0.33	0.92 ± 0.0	0.92 ± 0.0	0.96 ± 0.01	0.94 ± 0.0
	ROCKET	232.56 ± 2.33	16.61 ± 0.58	0.93 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.0
	TSF	12.59 ± 0.36	3.03 ± 0.23	0.93 ± 0.0	0.94 ± 0.01	0.95 ± 0.01	0.95 ± 0.0
	WEASEL-v2	292.77 ± 6.08	13.08 ± 0.61	0.92 ± 0.01	0.94 ± 0.01	0.94 ± 0.01	0.94 ± 0.0
	cBOSS	297.98 ± 3.84	234.07 ± 0.98	0.91 ± 0.0	0.93 ± 0.01	0.93 ± 0.0	0.93 ± 0.0

Table B.2: Time-Series Classification in Multiclass Scenario (Full Table)

Dataset	Model	Train Time [s]	Test Time [s]	Accuracy	Precision	Recall	F1-Score
BOT-IoT	CIF	1209.81 ± 13.77	517.18 ± 25.96	1.0 ± 0.0	1.0 ± 0.0	0.94 ± 0.05	0.96 ± 0.04
	MiniROCKET	114.43 ± 2.28	2.71 ± 0.46	1.0 ± 0.0	0.96 ± 0.09	0.92 ± 0.08	0.93 ± 0.08
	RISE	137.08 ± 0.18	18.68 ± 0.33	1.0 ± 0.0	0.96 ± 0.09	0.92 ± 0.08	0.93 ± 0.08
	ROCKET	248.0 ± 4.92	15.81 ± 0.79	1.0 ± 0.0	0.92 ± 0.11	0.9 ± 0.1	0.91 ± 0.1
	TSF	14.68 ± 0.89	2.86 ± 0.33	1.0 ± 0.0	1.0 ± 0.0	0.94 ± 0.05	0.96 ± 0.04
	WEASEL-v2	313.04 ± 3.82	13.42 ± 0.56	1.0 ± 0.0	0.96 ± 0.09	0.92 ± 0.08	0.93 ± 0.08
	cBOSS	272.37 ± 3.36	223.1 ± 0.86	1.0 ± 0.0	0.96 ± 0.09	0.92 ± 0.08	0.93 ± 0.08
	CIC-IDS-2017	CIF	1559.61 ± 28.39	621.43 ± 13.92	0.98 ± 0.0	0.72 ± 0.03	0.65 ± 0.05
MiniROCKET		107.35 ± 1.88	4.69 ± 0.57	0.99 ± 0.0	0.74 ± 0.08	0.67 ± 0.04	0.69 ± 0.05
RISE		206.45 ± 3.16	22.61 ± 0.19	0.99 ± 0.0	0.76 ± 0.09	0.67 ± 0.09	0.7 ± 0.08
ROCKET		322.1 ± 1.81	28.2 ± 0.58	0.99 ± 0.0	0.76 ± 0.07	0.73 ± 0.06	0.74 ± 0.06
TSF		21.79 ± 0.5	4.22 ± 0.29	0.99 ± 0.0	0.76 ± 0.04	0.7 ± 0.06	0.72 ± 0.05
WEASEL-v2		391.83 ± 4.73	26.6 ± 0.94	0.99 ± 0.0	0.77 ± 0.05	0.72 ± 0.06	0.73 ± 0.05
cBOSS		615.52 ± 1.13	243.92 ± 2.08	0.99 ± 0.0	0.75 ± 0.05	0.75 ± 0.03	0.74 ± 0.02
CSE-CIC-IDS-2018		CIF	1884.56 ± 21.53	540.22 ± 8.61	0.98 ± 0.0	0.92 ± 0.07	0.86 ± 0.04
	MiniROCKET	125.37 ± 2.35	5.23 ± 0.77	1.0 ± 0.0	0.91 ± 0.06	0.86 ± 0.03	0.88 ± 0.04
	RISE	165.32 ± 1.52	22.26 ± 0.38	0.99 ± 0.0	0.91 ± 0.06	0.85 ± 0.03	0.87 ± 0.04
	ROCKET	371.63 ± 5.88	29.8 ± 0.44	1.0 ± 0.0	0.91 ± 0.06	0.87 ± 0.03	0.88 ± 0.04
	TSF	23.41 ± 0.66	4.15 ± 0.28	1.0 ± 0.0	0.91 ± 0.05	0.89 ± 0.04	0.9 ± 0.04
	WEASEL-v2	435.14 ± 5.14	27.33 ± 0.74	1.0 ± 0.0	0.92 ± 0.06	0.88 ± 0.05	0.89 ± 0.05
	cBOSS	1064.73 ± 6.55	282.97 ± 2.19	0.99 ± 0.0	0.9 ± 0.07	0.88 ± 0.05	0.89 ± 0.05
	Edge-IIoT	CIF	1615.61 ± 26.54	1099.32 ± 58.52	0.99 ± 0.0	0.98 ± 0.01	0.96 ± 0.01
MiniROCKET		264.62 ± 2.68	3.73 ± 0.41	1.0 ± 0.0	0.99 ± 0.0	0.99 ± 0.01	0.99 ± 0.0
RISE		143.8 ± 1.21	19.3 ± 0.25	0.99 ± 0.0	0.99 ± 0.0	0.94 ± 0.01	0.95 ± 0.01
ROCKET		348.58 ± 2.35	21.78 ± 1.18	0.99 ± 0.0	0.99 ± 0.0	0.98 ± 0.01	0.98 ± 0.0
TSF		18.49 ± 0.48	3.57 ± 0.13	0.99 ± 0.0	0.99 ± 0.0	0.96 ± 0.01	0.97 ± 0.01
WEASEL-v2		433.37 ± 4.71	18.47 ± 0.98	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.01	0.99 ± 0.0
cBOSS		538.06 ± 5.83	253.78 ± 1.64	0.98 ± 0.0	0.97 ± 0.01	0.96 ± 0.01	0.96 ± 0.01
IEC61850-Security		CIF	327.87 ± 5.18	487.33 ± 17.35	0.44 ± 0.01	0.48 ± 0.09	0.3 ± 0.02
	MiniROCKET	20.09 ± 0.41	0.71 ± 0.02	0.44 ± 0.01	0.46 ± 0.04	0.31 ± 0.01	0.31 ± 0.02
	RISE	52.17 ± 0.57	8.13 ± 0.25	0.44 ± 0.01	0.35 ± 0.01	0.34 ± 0.01	0.34 ± 0.01
	ROCKET	52.68 ± 1.51	3.37 ± 0.07	0.43 ± 0.01	0.42 ± 0.03	0.31 ± 0.01	0.3 ± 0.02
	TSF	5.51 ± 0.12	1.56 ± 0.15	0.45 ± 0.01	0.35 ± 0.01	0.34 ± 0.01	0.34 ± 0.01
	WEASEL-v2	84.51 ± 1.31	3.71 ± 0.11	0.42 ± 0.02	0.44 ± 0.05	0.29 ± 0.01	0.28 ± 0.02
	cBOSS	26.51 ± 0.45	79.5 ± 0.36	0.42 ± 0.01	0.39 ± 0.05	0.3 ± 0.01	0.3 ± 0.01
	IoT-23	CIF	798.55 ± 18.4	558.06 ± 34.92	0.93 ± 0.01	0.94 ± 0.05	0.82 ± 0.04
MiniROCKET		104.12 ± 1.48	1.61 ± 0.12	1.0 ± 0.0	0.98 ± 0.03	0.93 ± 0.05	0.94 ± 0.04
RISE		109.85 ± 1.71	17.74 ± 0.15	0.92 ± 0.01	0.92 ± 0.02	0.89 ± 0.02	0.89 ± 0.01
ROCKET		170.96 ± 2.1	8.24 ± 0.51	1.0 ± 0.0	0.96 ± 0.03	0.92 ± 0.03	0.93 ± 0.03
TSF		6.51 ± 0.26	2.2 ± 0.15	0.99 ± 0.0	0.96 ± 0.03	0.92 ± 0.04	0.93 ± 0.04
WEASEL-v2		230.43 ± 2.97	8.43 ± 0.22	1.0 ± 0.0	0.98 ± 0.02	0.93 ± 0.05	0.95 ± 0.04
cBOSS		117.08 ± 1.22	202.2 ± 0.92	0.99 ± 0.0	0.95 ± 0.04	0.9 ± 0.04	0.91 ± 0.04
KDD99		CIF	1221.2 ± 21.68	277.62 ± 9.7	1.0 ± 0.0	0.97 ± 0.03	0.92 ± 0.02
	MiniROCKET	113.13 ± 2.4	2.76 ± 0.08	1.0 ± 0.0	0.96 ± 0.02	0.93 ± 0.02	0.94 ± 0.02
	RISE	126.25 ± 1.97	19.74 ± 0.45	1.0 ± 0.0	0.94 ± 0.02	0.84 ± 0.05	0.87 ± 0.03
	ROCKET	254.05 ± 4.95	16.61 ± 1.05	1.0 ± 0.0	0.95 ± 0.03	0.93 ± 0.03	0.93 ± 0.02
	TSF	12.34 ± 0.78	3.08 ± 0.18	1.0 ± 0.0	0.96 ± 0.04	0.91 ± 0.03	0.92 ± 0.04
	WEASEL-v2	324.65 ± 11.53	13.32 ± 0.56	1.0 ± 0.0	0.96 ± 0.03	0.93 ± 0.03	0.93 ± 0.02
	cBOSS	208.35 ± 29.39	237.06 ± 6.41	1.0 ± 0.0	0.95 ± 0.03	0.92 ± 0.04	0.93 ± 0.04
	TON-IoT	CIF	1177.48 ± 28.05	603.26 ± 36.37	0.96 ± 0.01	0.89 ± 0.04	0.84 ± 0.02
MiniROCKET		100.23 ± 0.62	2.92 ± 0.29	0.99 ± 0.0	0.96 ± 0.03	0.94 ± 0.0	0.94 ± 0.01
RISE		107.71 ± 2.14	18.53 ± 0.63	0.98 ± 0.0	0.93 ± 0.03	0.88 ± 0.01	0.89 ± 0.01
ROCKET		229.52 ± 2.97	16.41 ± 0.43	0.98 ± 0.0	0.94 ± 0.03	0.9 ± 0.02	0.91 ± 0.02
TSF		9.38 ± 0.33	3.03 ± 0.19	0.98 ± 0.0	0.91 ± 0.04	0.89 ± 0.01	0.89 ± 0.02
WEASEL-v2		282.56 ± 6.82	13.07 ± 0.25	0.98 ± 0.0	0.93 ± 0.04	0.9 ± 0.02	0.91 ± 0.03
cBOSS		177.88 ± 0.77	220.39 ± 2.26	0.97 ± 0.0	0.85 ± 0.01	0.86 ± 0.02	0.85 ± 0.02
UNSW-NB15		CIF	1566.41 ± 22.12	4522.62 ± 49.73	0.78 ± 0.01	0.5 ± 0.07	0.46 ± 0.07
	MiniROCKET	122.67 ± 1.52	3.2 ± 0.28	0.78 ± 0.0	0.52 ± 0.07	0.43 ± 0.02	0.44 ± 0.03
	RISE	136.93 ± 2.11	18.84 ± 0.22	0.78 ± 0.01	0.49 ± 0.07	0.44 ± 0.03	0.45 ± 0.04
	ROCKET	261.55 ± 1.97	15.65 ± 0.59	0.78 ± 0.01	0.52 ± 0.04	0.47 ± 0.06	0.48 ± 0.06
	TSF	18.95 ± 0.74	3.39 ± 0.13	0.79 ± 0.0	0.53 ± 0.04	0.47 ± 0.03	0.49 ± 0.03
	WEASEL-v2	335.34 ± 4.91	13.92 ± 0.96	0.78 ± 0.01	0.48 ± 0.07	0.43 ± 0.03	0.44 ± 0.04
	cBOSS	330.51 ± 2.01	236.65 ± 2.01	0.74 ± 0.01	0.45 ± 0.05	0.44 ± 0.05	0.44 ± 0.05

