Instituto Superior
de Engenharia
Politécnico de Coimbra

DEPARTMENT OF SYSTEMS AND COMPUTER ENGINEERING

Multi-Class Anomaly Detection in IoV and IIoT Networks

Project Report to fulfill the Master's degree in Informatics Engineering

Specialization in Software Engineering

Author

Ágata Ramalho Simões Palma

Supervisor

Ana Alves

Co-Supervisor

Mário Antunes

Coimbra, Dezembro, 2025

INSTITUTO POLITÉCNICO
DE COIMBRA

INSTITUTO SUPERIOR
DE ENGENHARIA
DE COIMBRA

Resumo

A crescente conectividade nos veículos e infraestruturas industriais tem aumentado a exposição de protocolos antigos, como o CAN e o MODBUS, a uma superfície de ataque mais ampla e sofisticada. A literatura tem destacado de forma consistente a necessidade de desenvolver sistemas de deteção de intrusões capazes de identificar de forma automática estes ataques, dada a ausência de mecanismos de segurança nativos nestes protocolos. Contudo, a deteção eficaz continua a ser desafiante devido ao acentuado desequilíbrio de classes, à sobreposição de assinaturas entre ataques e tráfego normal e à dificuldade em manter um desempenho estável em condições operacionais realistas.

Este trabalho aborda estas lacunas através de um estudo experimental dividido em três fases: a Fase I avalia a deteção de intrusões multi-classe para tráfego CAN, a Fase II avalia a deteção multi-classe para tráfego MODBUS e a Fase III consolida ambos os domínios num enquadramento unificado e transversal, validando o seu desempenho num cenário orientado para processamento em streaming.

Utilizando os datasets CICIoV2024 e ICS-Flow, avaliamos vários classificadores de machine learning, nomeadamente Random Forest, Extra Trees, XGBoost, AdaBoost, Logistic Regression, Artificial Neural Networks e Deep Neural Networks. Todos os classificadores são otimizados com Optuna e preservam o desequilíbrio dos dados, mantendo a distribuição original dos mesmos.

Os resultados, já publicados, das Fases I e II mostram que, para o CAN, a deteção de intrusões multi-classe está virtualmente resolvida, com ensembles baseados em árvores e DNN a atingir desempenhos quase perfeitos. Em contraste, o protocolo MODBUS continua a apresentar desafios: os ensembles mantêm a liderança e detetam quase perfeitamente o tráfego Normal e DDoS, mas classes como IP-Scan e Replay são mais difíceis de distinguir. Com base nestas conclusões, a Fase III propõe o Legacy-IDS, uma toolkit unificada para pré-processamento, otimização, treino, avaliação e inferência em streaming para ambos os protocolos, complementado por um CAN-BUS Dataset Converter que normaliza registos CAN heterogéneos para um formato compatível com o Legacy-IDS.

Palavras-chave: Sistemas de Deteção de Intrusões; Internet dos Veículos; Sistemas de Controlo Industrial; CAN bus; MODBUS; Machine Learning.

Abstract

The growing connectivity of vehicles and industrial infrastructures has increased the exposure of legacy protocols such as CAN and MODBUS to a wider and more sophisticated attack surface. Previous research consistently highlights the need for intrusion detection systems capable of automatically identifying these attacks, given the absence of native security mechanisms in these protocols. However, effective detection remains challenging due to severe class imbalance, overlapping attack signatures, and the difficulty of maintaining reliable performance under realistic operating conditions.

This work addresses these gaps through a three-phase experimental design: Phase I evaluates multi-class intrusion detection for CAN traffic, Phase II evaluates multi-class intrusion detection for MODBUS traffic, and Phase III consolidates both domains into a unified cross-domain framework and validates its performance in a streaming-ready setting.

Using CICIoV2024 and ICS-Flow, we evaluate several machine learning classifiers like Random Forest, Extra Trees, XGBoost, AdaBoost, Logistic Regression, Artificial Neural Networks and Deep Neural Networks. All classifiers are optimized using Optuna and maintain the data imbalance, the original dataset distribution.

The results, already published, for Phase I and II show that, for CAN, multi-class intrusion detection is virtually solved with tree-based ensembles and DNN achieving near-perfect performance. In contrast, MODBUS remains challenging: ensembles still lead and almost perfectly detect Normal and DDoS traffic, but classes such as IP-Scan and Replay are harder to separate from other flows. Building on these findings, Phase III proposes Legacy-IDS, a unified toolkit for preprocessing, optimization, training, evaluation and streaming inference across both protocols, complemented by a CAN-BUS Dataset Converter that standardizes heterogeneous CAN logs into a Legacy-IDS compatible format.

Keywords: Intrusion Detection Systems; Internet of Vehicles; Industrial Control Systems; CAN bus; MODBUS; Machine Learning.

Epigraph

In general, we look for a new law by the following process: First we guess it; then we compute the consequences of the guess to see what would be implied if this law that we guessed is right; then we compare the result of the computation to nature, with experiment or experience, compare it directly with observation, to see if it works. If it disagrees with experiment, it is wrong. In that simple statement is the key to science. It does not make any difference how beautiful your guess is, it does not make any difference how smart you are, who made the guess, or what his name is — if it disagrees with experiment, it is wrong.

Richard Feynman

Acknowledgements

My biggest thank you to my supervisors, my family and my wife-to-be. Thank you to Professor Ana Alves and Professor Mário Antunes, for challenging me to deliver more than I thought I would. Thank you to my family for putting up with my constant complaints and lack of time. And above all, thank you to my future wife, who supported me throughout this long process and always found ways to make my days a little easier.

Ah, and my cats. Thank you for the constant distractions whenever I tried to work.

Index

Resumo i

Abstract ii

Epigraph iii

Acknowledgements iv

List of Tables vii

List of Figures ix

List of Abbreviations x

1 Introduction 1

2 Background 5
2.1 Internet of Vehicles and CAN Protocol 5
2.2 Industrial Control Systems and MODBUS Protocol 7
2.3 Intrusion Detection Systems 12
2.4 Machine Learning 12
2.4.1 Random Forest 13
2.4.2 Extra Trees 13
2.4.3 XGBoost 14
2.4.4 AdaBoost 14
2.4.5 Logistic Regression 14
2.4.6 Artificial Neural Networks and Deep Neural Networks 14
2.4.7 Hyperparameter Optimization 15
2.4.8 Performance Metrics 16

3 Related Work 19
3.1 Intra-Vehicle Network - CAN 19
3.1.1 Binary Classification 20
3.1.2 Multi-Class Classification 24
3.2 Industrial Systems 28

3.2.1 Binary Classification . . . 29
3.2.2 Multi-Class Classification . . . 31

4 Methodology . . . 37
4.1 Literature Review . . . 37
4.2 Datasets . . . 38
4.2.1 ICS-Flow . . . 38
4.2.2 CICIoV2024 Dataset . . . 39
4.3 Pipeline Structure . . . 40
4.3.1 Legacy-IDS Evaluation . . . 42
4.3.2 CAN-BUS Dataset Converter . . . 44
4.4 Preprocessing . . . 45
4.5 Experimental Setup . . . 45

5 Results . . . 47
5.1 Phase I - Intrusion Detection on CAN Bus . . . 47
5.2 Phase II - Intrusion Detection on MODBUS . . . 51
5.3 Phase III - Intrusion Detection on CAN and MODBUS . . . 55
5.3.1 Binary Classification . . . 56
5.3.2 Multi-Class Classification . . . 59
5.4 Phase III - Legacy-IDS . . . 68
5.4.1 CAN . . . 68
5.4.2 MODBUS . . . 75

6 Conclusions . . . 83
6.1 Contributions and Limitations . . . 84
6.2 Future Work . . . 86

Bibliography . . . 87

List of Tables

2.1 Standard CAN bus message frame format. 6
2.2 Common CAN bus attacks. 7
2.3 Common MODBUS attacks. 9
2.4 Incidents in ICS 10

3.1 Summary of CAN and MODBUS most used Public Datasets. 20
3.2 Binary classification studies with CAN. 23
3.3 Multi-class classification studies with CAN. 27
3.4 Binary classification studies with MODBUS. 30
3.5 Multi-class classification studies with MODBUS. 33

4.1 Description of ICS-Flow dataset features. 38
4.2 Description of CICIoV2024 dataset features. 40
4.3 Details of the experimental setup - ASUS laptop. 46

5.1 Optimized hyperparameters for split and 10-fold training. 47
5.2 Comparison of execution time. 48
5.3 Comparison of model performance across different studies using split approach on CICIoV2024 dataset. 49
5.4 Performance comparison of different models using 10-fold training on CICIoV2024 dataset. 49
5.5 Optimized hyperparameters. 51
5.6 Comparison of multi-class results for the selected ML methods. 53
5.7 Per-class F1-score comparison between our results and the original study. 54
5.8 Optimized hyperparameters for CAN (Binary). 56
5.9 Comparison of binary results for the selected ML methods. 57
5.10 Optimized hyperparameters for MODBUS (Binary). 57
5.11 Comparison of binary results for the selected ML methods. 58
5.12 Optimized hyperparameters for CAN (Multi). 59
5.13 Comparison of multi-class results for the selected ML methods. 60
5.14 Optimized hyperparameters for CAN (Multi). 63
5.15 Comparison of multi-class results for the selected ML methods (MODBUS, Multi). 63
5.16 Details of the CAN generated dataset 69

5.17 Comparison of multi-class results for the selected ML methods with a streamed dataset (CAN, Multi). 69
5.18 Comparison of binary results for the selected ML methods with a streamed dataset (CAN, Binary). 74
5.19 Details of the MODBUS generated dataset 75
5.20 Comparison of multi-class results for the selected ML methods with a streamed dataset (MODBUS, Multi). 76
5.21 Comparison of binary results for the selected ML methods with a streamed dataset (MODBUS, Binary). 80

List of Figures

2.1 MODBUS message frame structure. 8

4.1 Study workflow. 41
4.2 Phase I and II workflow structure. 42
4.3 Phase III workflow structure 43
4.4 CAN-BUS Dataset Converter 44

5.1 Confusion matrix for XGBoost with absolute values (CAN) 60
5.2 ROC/PR Curves for Extra-Trees (CAN, Multi) 61
5.3 ROC/PR Curves for Random Forest (CAN, Multi) 62
5.4 ROC/PR Curves for XGBoost (CAN, Multi) 62
5.5 ROC/PR Curves for Extra-Trees (MODBUS, Multi) 64
5.6 Confusion matrix for Extra Trees with absolute values (MODBUS). 65
5.7 ROC/PR Curves for XGBoost multi (MODBUS, Multi) 65
5.8 Confusion matrix for XGBoost with absolute values (MODBUS). 66
5.9 ROC/PR Curves for Random Forest (MODBUS, Multi) 67
5.10 Confusion matrix for Random Forest with absolute values (MODBUS). 67
5.11 ROC/PR Curves for Random Forest (CAN, Multi) 70
5.12 Confusion matrix for Random Forest with absolute values (Stream evaluation, CAN) 71
5.13 ROC/PR Curves for Extra Trees (CAN, Multi) 71
5.14 Confusion matrix for Extra Trees with absolute values (Stream evaluation, CAN) 72
5.15 ROC/PR Curves for XGBoost (CAN, Multi) 72
5.16 Confusion matrix for XGBoost with absolute values (Stream evaluation, CAN) 73
5.17 ROC/PR Curves for Random Forest (MODBUS, Multi) 77
5.18 Confusion matrix for Random Forest with absolute values (Stream evaluation, MODBUS) 77
5.19 ROC/PR Curves for Extra Trees (MODBUS, Multi) 78
5.20 Confusion matrix for Extra Trees with absolute values (Stream evaluation, MODBUS). 78
5.21 ROC/PR Curves for XGBoost (MODBUS, Multi) 79
5.22 Confusion matrix for XGBoost with absolute values (Stream evaluation, MODBUS). 79

Lista de abreviaturas

ACK	Acknowledge
ANN	Artificial Neural Networks
CAN	Controller Area Network
CRC	Cyclic Redundancy Check
DCS	Distributed Control Systems
DNN	Deep Neural Networks
DoS	Denial-of-Service
ECU	Electronic Control Unit
ET	Extra Trees
HMI	Human-Machine Interface
ICS	Industrial Control Systems
IED	Intelligent Electronic Devices
IEEE	Institute of Electrical and Electronics Engineers
IIoT	Industrial Internet of Things
IoT	Internet of Things
IoV	Internet of Vehicles
ISEC	Instituto Superior de Engenharia de Coimbra
LIN	Local Interconnect Network
LR	Logistic Regression
ML	Machine learning
MOST	Media oriented systems transport
PLCs	Programmable Logic Controllers

RF	Random Forest
RTR	Remote Transmission Request
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
SOF	Start of Frame
V2X	Vehicle-to-everything
VANET	Vehicle Ad-hoc Network

1 Introduction

The increasing reliance on interconnected technologies has transformed industrial and vehicular systems, introducing both operational efficiencies and significant cybersecurity challenges. Industrial Control Systems (ICS) and the Internet of Vehicles (IoV) are two critical infrastructures that exemplify this duality. ICS, responsible for controlling and automating processes in sectors such as manufacturing [1], power grids [2], and water treatment facilities [3], has evolved through the integration of technologies from the Industrial Internet of Things (IIoT) [1, 2, 3, 4]. Meanwhile, IoV has become the backbone of modern intelligent transportation systems, leveraging vehicular communication technologies to enhance safety, real-time decision-making, and autonomous driving capabilities [5, 6]. However, as these environments become more interconnected, their exposure to cyber threats increases, requiring efficient security mechanisms to detect and protect against increasingly sophisticated attacks [4, 5, 7, 8, 9, 10, 11].

Both ICS and IoV rely on legacy communication protocols that were never designed with security in mind. ICS environments commonly use MODBUS, a protocol known for its simplicity and compatibility [10, 11, 12, 13] but lacking authentication and access control [14, 15], encryption [15, 16], and replay protection [17], making it highly vulnerable to cyberattacks [18]. Similarly, IoV systems rely on the Controller Area Network (CAN) protocol, which, despite its robustness in real-time communication, also lacks encryption and authentication mechanisms, making vehicles susceptible to spoofing and Denial-of-Service (DoS) attacks [5, 7, 19]. These security weaknesses have led to real-world cyber incidents, such as the 2020 ransomware attack that disrupted Honda's ICS operations and the targeted attacks on Israeli water facilities, demonstrating the urgency of improving cybersecurity in these domains [20].

Intrusion Detection Systems (IDS) have emerged as a critical security layer for both ICS [8] and IoV [21, 22, 23, 24] environments. Unlike traditional security solutions, IDS continuously monitor network traffic to detect anomalies and unauthorized activities, making them essential for real-time cyber threat detection [7, 25]. Machine learning (ML) models play a critical role in IDS by enabling behavior-based detection of cyber threats. However, a significant challenge in both domains is the severe class imbalance in network traffic datasets, where benign traffic far outweighs malicious activity, leading to biased ML models that struggle to detect minority attack classes [7, 11]. Furthermore, existing IDS approaches often fail to generalize effectively to real-world conditions, limiting their reliability in detecting evolving attack strategies.

To address these issues, we evaluated machine learning models for both ICS and IoV environments. Using the ICS-Flow dataset for industrial networks [11] and the CICIoV2024 dataset for vehicular environments [7], we assessed the performance of various ML algorithms, including Random Forest (RF), XGBoost, AdaBoost, Extra Trees (ET), Deep Neural Networks (DNN), Artificial Neural Networks (ANN) and Logistic Regression (LR). Unlike previous studies that artificially balance datasets [22, 26], we maintain the natural distribution of benign and malicious traffic to provide a realistic evaluation of model performance in highly imbalanced, multi-class scenarios.

To optimize detection capabilities, we used Optuna [27] for automated hyperparameter tuning, ensuring that the models are fine-tuned for maximum performance while mitigating overfitting. By comparing ML techniques across two critical infrastructures, ICS and IoV, we provide a cross-domain perspective on securing industrial and vehicular networks against modern cyber threats. Beyond intrusion detection, the research findings contribute to broader challenges in real-time network analysis and the development of scalable, resource-efficient security models for critical infrastructure protection.

In summary, the central problem addressed here is how to design, implement and evaluate machine-learning-based IDS that can operate effectively on both CAN and MODBUS traffic under realistic, highly imbalanced, multi-class conditions, while remaining compatible with deployment constraints such as resource usage, reproducibility, and streaming operation. Existing work tends to treat ICS and IoV in isolation, often focuses on binary discrimination between benign and attack traffic, artificially balances datasets, or evaluates models only in static settings. As a result, there is limited understanding of how different algorithms behave when confronted with realistic class distributions, difficult attack types, and protocol-specific characteristics, and there is little guidance on how to build a unified cross-domain IDS that can be reused across vehicular and industrial environments.

In response to this problem, there are four main research objectives. First, is to evaluate different ML algorithms for multi-class classification on CAN and MODBUS, using CICIoV2024 and ICS-Flow as realistic benchmarks and preserving their natural class imbalance. The second objective is to investigate the impact of automated hyperparameter optimization on model performance and computational cost, and to identify which models provide the best trade-off between detection capability and efficiency in each domain. The third objective is to design and implement a cross-domain IDS framework, Legacy-IDS, that encapsulates data loading, preprocessing, model training, evaluation, and streaming inference for both protocols within a single, reproducible software pipeline. The fourth objective is to facilitate the integration of heterogeneous CAN datasets by developing a CAN-BUS Dataset Converter that standardizes raw vehicular logs into a common format compatible with the IDS pipeline.

These objectives are articulated through a set of research questions that guide the work. **RQ1:** To what extent can classical and ensemble-based ML models, when tuned with automated hyperparameter optimization, achieve reliable multi-class intrusion detection on imbalanced CAN and MODBUS datasets without artificial resampling? **RQ2:** Which families of algorithms (tree-based ensembles, neural networks, or linear models) offer the best balance between macro-averaged detection performance, robustness to class imbalance, and computational cost in ICS and IoV settings? **RQ3:** How well do models that perform strongly under static train–test evaluation maintain their behavior when traffic is presented as a stream that mixes real and simulated traces, and what failure modes emerge under these more realistic conditions? **RQ4:** Can a unified, protocol-agnostic pipeline such as Legacy-IDS, complemented by a CAN-BUS Dataset Converter, support cross-domain experimentation and deployment without sacrificing performance on domain-specific tasks?

The scope of the research is deliberately circumscribed to allow a focused and reproducible investigation. On the data side, the study is limited to two publicly available datasets. On the methodological side, the work concentrates on supervised learning with tabular features, using Random Forest, Extra Trees, XGBoost, AdaBoost, ANN, DNN and Logistic Regression as representative models. More complex deep architectures,as well as unsupervised or self-supervised approaches, are not explored. In addition, hyperparameter optimization with Optuna is performed with a limited number of trials per model in order to reflect realistic resource constraints. These choices entail several limitations, since the conclusions drawn from CICIoV2024 and ICS-Flow may not be generalized directly to all ICS and IoV deployments, particularly those with different device mixes, topologies, or attack patterns. The focus on a specific family of algorithms leaves open whether other model classes could achieve further gains under the same constraints. The streaming evaluation, though closer to practice than static splits, still abstracts away important system-level aspects such as latency, resource contention, and operator interaction with alerts. Nevertheless, by clearly defining its scope and constraints, the aim is to provide a solid and transparent basis on which future work can build.

The remainder of this work is organized as follows. Chapter 2 introduces the necessary background on ICS, IoV, the MODBUS and CAN protocols, and fundamental concepts in machine-learning-based intrusion detection, as well as a brief description of the machine learning models used, the evaluation metrics and an introduction to hyperparameter optimization as well as the chosen optimization tool. Chapter 3 presents the related work on IDS for both domains. Chapter 4 describes the datasets used in this study, the preprocessing steps applied to CAN and MODBUS traffic, and the construction of the CAN-BUS Dataset Converter. It also presents the overall experimental design and introduces the architecture of the Legacy-IDS framework, including its streaming

component. Chapter 5 reports and discusses the experimental results in three phases. Phase I for CAN-based intrusion detection, Phase II for MODBUS-based intrusion detection, and Phase III for the unified cross-domain Legacy-IDS and its streaming evaluation. Finally, Chapter 6 synthesizes the main findings, discusses the contributions and limitations of the work, and outlines promising directions for future research on cross-domain IDS for critical infrastructures.

2 Background

This chapter provides the essential concepts needed to understand the main topics covered in this work. It presents an overview of how ICS and IoV are structured and how they communicate, with a focus on the weaknesses of legacy protocols such as MODBUS and CAN. The importance of intrusion detection systems to address the security gaps within these environments is also explained.

2.1 Internet of Vehicles and CAN Protocol

The Internet of Vehicles extends vehicular networking by joining Vehicle Ad-hoc Network (VANET) principles with the Internet of Things to enable vehicle-to-everything (V2X) exchanges among vehicles, roadside equipment, pedestrians, and cloud services [5, 28, 29, 30]. As a result, modern vehicles have shifted from isolated, mechanical machines to connected, computer-driven systems that promise faster reactions and better safety [28]. However, this great evolution comes with a trade-off: "traditional" mechanical parts were free of automation and electronics, making them immune to remote cyberattacks but with the integration of features such as lane keeping and collision avoidance, they become digitally reachable[28]. This connectivity is facilitated by the significant integration of Electronic Control Units (ECUs) that manage these modern systems through an interconnected network, the Controller Area Network (CAN) [21, 25, 31, 32]. Although there are other possible intra-vehicle protocols such as Local Interconnect Network (LIN), FlexRay and media oriented systems transport (MOST), the CAN Bus is the dominant in the vehicular domain due to its reliability and low cost [25, 28, 33, 34].

The Controller Area Network is a robust communication system designed to allow ECUs to reliably exchange data [25]. It operates on a two-wire physical bus, specifically CAN High and CAN Low [35]. These two wires carry inverse signals, a method called differential signaling [28], which makes the network highly resistant to electrical noise [28, 36]. These signals are represented by two states: a dominant state (logical '0') and a recessive state (logical '1') [28, 35]. The core principle is that the dominant state always overrides the recessive state, meaning that if even one ECU sends a dominant '0', the entire bus reflects that state, which is the key to how the protocol functions without data collisions[28].

The architecture of CAN's communication is message-centric, meaning ECUs broadcast

messages onto the network for all to hear, rather than sending data to a specific address [37]. Each message, or frame, as shown in Table 2.1, begins with a single Start of Frame (SOF) bit and is prefixed with a 12-bit arbitration field. This field, which contains an 11-bit identifier and a 1-bit Remote Transmission Request (RTR), describes the data's content and sets its priority, with a lower number signifying a higher priority [7, 19, 21]. The true elegance of CAN is its method of non-destructive bitwise arbitration. When multiple ECUs start talking at once, they begin transmitting their identifier bits. While sending, each ECU listens simultaneously to the bus[34]. If one sends a recessive '1' but detects a dominant '0' on the wire, it knows that a higher-priority message is being transmitted. It immediately stops sending and transitions to a listening state, allowing the message with the lower identifier to continue without any corruption or delay [25, 31].

Following the arbitration field, a 6-bit control field specifies the frame's particulars, including a 4-bit Data Length Code (DLC) that defines a data payload of between zero and eight bytes. To ensure reliability, every frame includes a 16-bit Cyclic Redundancy Check (CRC) field (composed of a 15-bit checksum and a 1-bit delimiter) to verify that the message content is error-free. Following this is a 2-bit Acknowledge (ACK) field where receiving ECUs can confirm a successful reception before the frame ends with a 7-bit End of Frame (EOF) sequence [28]. Although every CAN message follows the same structural rules, the physical wiring on which it travels comes in two main types: a high-speed version for critical tasks or a slower, more fault-tolerant version for convenience features [7, 5].The high-speed version, High-Speed CAN supports data rates up to 1 Mbit/s [7, 34, 5] and is used for critical powertrain and safety systems. In contrast, the slower version, Low-Speed CAN or fault-tolerant CAN, operates at slower speeds between 40 kbit/s and 250 kbit/s and is designed for less critical comfort-related systems [7, 5].

	Message Frame (Standard)													
Bus Idle	**Header Field (19 bits)**						**Protected Payload Field (80 bits max)**			**Trailer Field (12 bits)**				Bus Idle
	Start of Frame (SOF) (1 bit)	Arbitration Field (12 bits)		Control Field (6 bits)			Data Field (0 to 8 bytes)	Check Field (16 bits)		ACK Field (2 bits)		End of Frame (EOF) (7 bits)	Inter Frame Space (IFS) (3 bits)	
		Identifier (ID)	Remote Transmission Request (RTR)	Identifier Extension (IDE)	Reserved (R)	Data Length Code (DLC)	Data[0–7]	Cyclic Redundancy Check (CRC)	Delimiter (DEL)	Acknowl. (ACK)	Delimiter (DEL)			

Table 2.1: Standard CAN bus message frame format.

This design intended for a closed-loop system rather than a highly connected one introduces numerous security vulnerabilities [5, 7, 31, 37], having no provisions for network security [38]. CAN bus does not provide built-in authentication and encryption [5, 7, 19], integrity protection, confidentiality and availability [25, 28]. As IoV connectivity introduces multiple open communications surfaces with V2X, the possible attack surface also increases, leaving room to disrupt or manipulate vehicle functions[25, 28].

The vulnerabilities inherent in the CAN protocol's design allow for a range of practical

attacks, each targeting different aspects of vehicle operation, with various consequences, as described in Table 2.2.

Table 2.2: Common CAN bus attacks.

Attack	Description	Consequences
Denial-of-Service [5, 28]	Floods the bus with high-priority frames that always win arbitration.	Starves legitimate traffic; can disable safety-critical functions.
Spoofing [5, 28]	Sends frames using a trusted identifier (ID) to appear as a specific ECU.	Receivers accept forged status/commands as authentic.
Fuzzing [5]	Injects mutated or random frames and logs reactions.	Can crash ECUs, trigger bus-off, or cause unsafe behavior.
Eavesdropping [5, 28]	Passively records broadcast, unencrypted traffic.	Enables reverse-engineering and targeted follow-on attacks.
Replay [28]	Rebroadcasts captured legitimate frames.	Re-executes prior actions; no freshness checks to block it.
Masquerade [39]	Displaces or uses the credentials of a legitimate ECU (often after forcing it offline).	Sustained, trusted access to issue commands or false data.

2.2 Industrial Control Systems and MODBUS Protocol

Industrial Control Systems are the operational foundation of modern critical infrastructure, which includes the complex network of hardware and software used to monitor and automate physical processes in real time [40, 41, 42]. These systems are essential in sectors such as manufacturing [1], power generation and distribution [2], transportation systems [8, 43], water treatment [3], and oil and gas refinement [12, 44].

A standard ICS is a hierarchical system [43, 45]. Physical sensors and actuators are at the lowest level, directly controlled by field devices. These include Programmable Logic Controllers (PLCs), which are industrial-grade computers that execute logic for specific machinery or processes [42]. For geographically dispersed operations, such as a pipeline, Remote Terminal Units (RTUs) serve a similar control and monitoring role [4]. One level above PLCs and RTUs is the Supervisory Control and Data Acquisition (SCADA) system. The SCADA layer provides a centralized interface, usually a Human-Machine Interface (HMI), allowing operators to observe the entire industrial process, collect data for analysis, and issue high-level commands from a control room [40, 43]. For these distributed components to function cohesively, they must communicate through a standardized protocol. In the domain of ICS, one of the oldest and still widely used protocols on standard Ethernet networks is MODBUS, due to its simple structure, compatibility with legacy systems and easy integration [11, 12, 13, 15, 46].

The protocol works on a client-server architecture. A single client device, such as the SCADA server or a workstation, is the only one that can initiate communication. Communication is achieved through a simple request-response mechanism. It polls server devices (PLCs or RTUs) in a continuous loop, sending requests and awaiting responses [47, 48].

The MODBUS message, illustrated in Figure 2.1, is formed by a MODBUS Application Protocol (MBAP) header and the Protocol Data Unit (PDU). The PDU contains a one-byte Function Code and a variable-length Data payload. The Function Code is the command telling the server what action to perform, while the Data payload contains information regarding the action to perform. The message header consists of a 2-byte Transaction Identifier (to match requests with responses), a 2-byte Protocol Identifier (always zero for MODBUS), a 2-byte Length Field (specifying the number of remaining bytes in the frame) and a single-byte Unit Identifier (acting as a server address to route the message to the correct target device) [49].

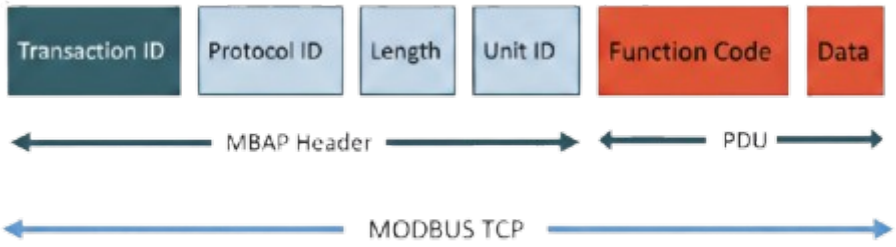


Figure 2.1: MODBUS message frame structure.

Similarly to the CAN protocol, MODBUS was designed for physically isolated, trusted industrial environments where security was not a consideration, resulting in fundamental vulnerabilities [47]. Since it does not support authentication, any device connected to the network can send valid commands to a server device, causing the wrong commands to be executed. In addition, all communications are transmitted in plaintext over packets, as there is no encryption, allowing eavesdropping attacks on the networks. Finally, there is no authorization, which means that all devices have the full authority to send any kind of command [49]. These design flaws expose ICS to several well-documented attacks, as detailed in Table 2.3.

Table 2.3: Common MODBUS attacks.

Attack	Description	Consequences
Denial-of-Service (DoS)	Floods a target device (PLC/RTU) with a high volume of requests or malformed packets to exhaust its resources.	The device becomes unresponsive to legitimate commands from the SCADA client, causing a loss of control and visibility.
Man-in-the-Middle (MitM)	Intercepts communication between the SCADA client and a server device to eavesdrop on or alter traffic in real-time.	Allows an attacker to inject malicious commands, spoof responses to hide the system's true state, and manipulate the physical process.
IP Scan	Probes the network to discover the IP addresses of active devices, such as PLCs, RTUs, and HMI workstations.	Creates a map of the operational network, identifying potential targets for more direct attacks.
Port Scan	Scans a target device's IP address to identify open communication ports, specifically the MODBUS port.	Confirms the presence and accessibility of MODBUS services on a device, enabling direct protocol-level attacks.
Replay	Captures a legitimate data packet and re-transmits it at a later, unauthorized time.	Triggers unintended and potentially hazardous actions in the physical process, disrupting stable operations.

Given that IoT is increasingly being integrated with industrial systems, resulting in the development and advancement of the IIoT [50], the attack surface for these systems is also increased [4, 51]. Traditional industrial systems are no longer isolated, and new remote access points expose these critical infrastructures to cyberattacks. Since 2020, a series of cyberattacks targeting ICS have been documented (see Table 2.4). The impact of each of these incidents is diverse, ranging from the theft of sensitive information to remote machinery manipulation, causing extensive damage to the infrastructure [52].

Table 2.4: Incidents in ICS

Attack	Year	Sector	Country	Target	Impact
Maroochy Shire sewage spill [53, 54, 55]	2000	Water, Wastewater	Australia	Sewage pumping stations (Maroochy Water Services)	800 000 l of raw sewage released; environmental damage; service disruption
StuxNet [56]	2010	Nuclear	Iran	Natanz nuclear centrifuges	1000–2000 centrifuges replaced; Environmental damage
Night Dragon [57]	2010	Oil, Energy, Petrochemical	Global	Corporate networks	Sensitive information theft
DuQu [58]	2011	Nuclear	Iran	Nuclear Plants	Information theft
Shamoom/ Disttrack [59, 60]	2012	Oil, Gas	Saudi Arabia, Qatar	Saudi Aramco, RasGas	30 000 office computers wiped in Saudi Aramco, unknown number in RasGas
Bowman Avenue Dam [55, 56, 61]	2013	Water	USA	Small sluice-gate dam SCADA	No physical impact
Havex [52, 62]	2013	Global	Global	SCADA/ICS	Espionage-grade access and mapped industrial networks
German Steel Mill [52]	2014	Steel	Germany	Production network (blast furnace)	Physical damage from improper furnace shutdown
BlackEnergy [52, 63]	2014	Energy	Ukraine	HMI workstations	Credentials theft,network discovery, KillDisk wiping
Dragonfly/Energetic Bear [64]	2014	Energy	USA, Turkey, Switzerland	Company network	Persistent access and potential to sabotage
Ukraine Power Grid [52, 64, 65]	2015	Energy	Ukraine	Distribution control systems	230 000 people without electricity for up to 6 hours

(*continues on next page*)

(*continued from previous page*)

Attack	Year	Sector	Country	Target	Impact
"Kemuri" Water Company [55]	2016	Water	USA	PLCs/SCADA	Exfiltration of 2.5 million unique records and manipulation of chemicals and flow rates
Shamoon 2 [52, 66]	2016	Aviation	Saudi Arabia	Windows workstations/servers	Thousands of systems wiped
Ukraine Power Grid [52, 67]	2016	Energy	Ukraine	Transmission substation, utility call center	Approximately 225 000 customers without electricity for around 3 hours
NotPetya [68]	2017	Global	Ukraine, Global	Supply chain	Global disruption; multi-billion losses
Dragonfly 2.0 [69, 70]	2017	Energy	USA, Europe	ICS equipment vendors	Potential to sabotage or gain control of ICS
Triton/ Trisis/ HatMan [52, 71, 72]	2017	Petrochemical	Saudi Arabia	Industrial safety systems	Network-triggered firmware backdoor enabling code execution, memory tampering, and fail-safe bypass
Colonial Pipeline Ransomware [73]	2021	Oil	USA	Pipeline operation	Fuel crisis in the Eastern United States
Oldsmar's Cyberattack [74, 75]	2021	Water	USA	Water treatment facility	Attempt to poison water by increasing levels of sodium hydroxide
Muleshoe attack [74, 76]	2024	Water	USA	Water utilities	Water tank overflow, SCADA management of hydraulic functions compromised
Arkansas City breach [77, 78]	2024	Water	USA	Water treatment facility	No disruption to service occurred but forced the facility to switch to manual operations

2.3 Intrusion Detection Systems

The increasing connectivity in ICS in IoV environments has already led to numerous threats with a real impact on safety. Given the inherent lack of security features such as authentication and encryption in legacy protocols such as CAN and MODBUS, the systems they support are left exposed to a wide array of cyber threats in ICS and IoV environments, which can be exploited. To address the gap without changing the base system, IDS supplies a critical detection layer. An IDS functions as a monitoring mechanism, similar to a digital surveillance system, that observes network traffic and system behavior in real-time [79, 80]. Instead of actively modifying the main protocols, an IDS passively analyzes data exchanges to identify patterns indicative of malicious activity or policy violations [81]. Both IoV and ICS environments are safety-critical and time-sensitive, which means that an undetected cyberattack can have catastrophic real-world consequences [82], as already demonstrated in Table 2.4. An IDS provides a safety layer of vigilance by continuously searching for anomalies, anomaly based IDS, or known attack signatures, signature-based IDS [79, 81]. For example, it can identify the high-frequency message patterns of a Denial-of-Service attack on a CAN bus or detect an unauthorized MODBUS command sent to a critical PLC. Upon identifying a potential threat, the primary role of an IDS is to generate a timely and actionable alert, notifying operators or automated systems of the intrusion [80].

The integration of machine learning has significantly advanced the capabilities of these detection systems [83]. Machine learning algorithms can analyze complex datasets from network traffic much more effectively than traditional rule-based methods, enabling them to uncover subtle correlations and hidden patterns that could be a sign of a new and different attack [79]. By establishing a sophisticated baseline of normal system operation, machine learning-based and anomaly-based IDS can detect even the slightest deviations, providing a more dynamic and adaptive defense mechanism. Techniques such as deep learning are particularly adept at processing high-velocity, high-volume data streams characteristic of IoV and ICS environments [81]. This allows for the development of more accurate and resilient IDS solutions that can evolve alongside the threat landscape, ultimately reducing false positives and improving the speed and reliability of threat detection in these critically sensitive domains.

2.4 Machine Learning

Machine Learning is now on the frontline in the development of Intrusion Detection Systems due to it's broad capabilities in pattern recognition. Here, algorithms suited to high-dimensional, imbalanced and tabular network data are addressed: bagging-based ensembles that reduce variance through randomized feature/instance sampling, such as Extra Trees and Random Forest; boosting methods that iteratively correct residual

errors and capture non-linear interactions with strong class-separation power, such as XGBoost and AdaBoost; a linear baseline that offers speed, interpretability, and well-calibrated decision boundaries, like Logistic Regression, and neural models that learn hierarchical representations directly from features, with shallow (ANN) and deeper (DNN) architectures trading off capacity and overfitting control. This toolkit allows us to balance accuracy, robustness, and interpretability across CAN and MODBUS traffic while coping with class imbalance and concept overlap. In the subsections that follow, the learning principles and practical trade-offs are outlined, as well as the metrics used to evaluate the performance of each model.

2.4.1 Random Forest

A random forest is an ensemble model that trains many decision trees and blends their outputs to improve generalization. Each tree is grown on a bootstrap replica of the training set (sampling with replacement), so every tree "sees" a slightly different dataset. While the tree is being built, every split is chosen after evaluating candidate thresholds only on a randomly selected subset of input variables; the size of this subset is controlled by the max_features setting. In other words, data rows are randomized by resampling and feature choices are randomized at each split. These two forms of randomness serve a single aim: reduce the correlation among trees so that averaging their predictions drives down variance. A lone decision tree is notorious for high variance and a tendency to memorize noise. By forcing trees to differ,through bootstrap sampling and feature subsampling, their errors become less aligned. When the forest aggregates them, through majority vote for classification, unshared errors tend to cancel, yielding a model that overfits less. This comes with a modest rise in bias, but the variance drop is usually much larger, so the overall error declines and real-world performance improves[84].

2.4.2 Extra Trees

This algorithm builds an ensemble by training many highly randomized decision trees, often called "extremely randomized trees" or "Extra Trees", on different slices of the data, then averages their outputs to boost prediction quality and curb overfitting. Compared with standard random forests, the extra-trees approach injects additional randomness into how splits are chosen. Instead of exhaustively searching for the best threshold on a randomly selected subset of features, it samples several split thresholds at random for each candidate feature and adopts the one that performs best among those samples. This extra randomness tends to lower model variance even further, typically trading a small increase in bias for a net gain in generalization [85].

2.4.3 XGBoost

"XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable"[86]. It treats learning as an optimization problem over an ensemble of classification and regression trees (CART). Instead of relying on one deep tree, it builds many small trees whose predictions are summed, with each new tree added to improve the current model's fit. Training is driven by an explicit objective that combines a data-fitting term (the loss) with a regularization term that penalizes overly complex trees, striking a balance between predictive power and simplicity to avoid overfitting. In practice, this means the model is grown additively, one tree at a time, so that the ensemble steadily refines its predictions while keeping complexity in check. At each step, XGBoost estimates how the loss would change and uses both the error (gradients) and how quickly that error changes (curvature) to choose splits. It scores candidates with these numbers, adds a penalty for model complexity, and drops any split whose benefit doesn't beat the penalty. The result is fast training, sensible pruning, and models that generalize well for regression, classification, and ranking [86].

2.4.4 AdaBoost

AdaBoost builds a model by training many weak learners, such as small decision trees, one after another. Their outputs are then blended into a single predictor using weights, so some learners count more than others in the final vote or sum. Training starts with all samples treated equally. After each round, the algorithm increases the weight of the examples the current ensemble gets wrong and reduces the weight of those it gets right. The next weak learner is fit to this reweighted dataset, effectively pushing it to pay extra attention to the hard cases. As the process continues, difficult observations gain more and more influence, and the ensemble evolves into a strong classifier formed by a weighted combination of the successive weak learners [87].

2.4.5 Logistic Regression

In scikit-learn, classification with a logistic link is provided by the LogisticRegression class. Despite the name, it's a linear classifier, not a regressor, in the library terminology. The method estimates class membership by mapping a linear combination of features through the logistic (sigmoid) function to produce probabilities for the possible outcomes of a trial [88].

2.4.6 Artificial Neural Networks and Deep Neural Networks

Artificial neural networks are computational models built from layers of simple units (neurons) that apply learned weights, biases, and nonlinear activations to transform

inputs and estimate target outputs. Depth enables hierarchical representation learning, with earlier layers capturing simple patterns while later layers encode increasingly abstract structure. Training uses loss minimization with gradient back-propagation to update parameters over large datasets, a setup that has driven major advances across vision, speech, and other domains. When these models contain many stacked layers, they are called deep neural networks [89, 90]. Deep neural networks are machine-learning models built by stacking many layers of interconnected units so that each layer transforms its input and passes richer representations to the next. This wide (or broad?) family includes fully connected, dense networks in which every neuron links to the next layer. Recurrent models that reuse hidden state so predictions depend on both current and prior inputs (RNNs) and long short-term memory variants that regulate information flow with gates to handle long-range dependencies (LSTM). Convolutional networks learn pattern detectors with sliding filters (CNN), while autoencoders (AE) compress data into latent codes and reconstruct it. Other members include extreme learning machines with randomly fixed hidden layers and analytically solved outputs (ELM), generative models such as GANs that learn to synthesize realistic samples, and transfer learning methods that repurpose knowledge from one task or domain to another (TL). Classic probabilistic stacks like restricted Boltzmann machines and deep belief networks also sit in this lineage, using layered latent variables to model data distributions. As depth and width grow, capacity and expressiveness increase but not only. Data needs, compute cost, and training time also tend to increase. So there must exist a better accuracy balance with the risk of overfitting, higher energy use, and higher costs, using techniques like regularization, and efficient models to keep training practical. [89, 90].

2.4.7 Hyperparameter Optimization

Machine learning models are controled by two distinct types of settings. The first are the parameters learned from data during training, such as split thresholds in decision trees or weights in neural networks. The second are hyperparameters, which are configuration choices defined before training begins and that control how the learning process is carried out and how complex the final model is. For example, the number of trees in an ensemble, the maximum tree depth, the learning rate in boosting, or the regularization strength in Logistic Regression, are all hyperparameters that can be tuned or, if ignored, set to the model default configuration. These choices strongly influence performance, generalization, and computational cost. Poorly selected hyperparameters can lead to underfitting, overfitting, unstable training, or unnecessarily expensive models, especially in highly imbalanced, multi-class intrusion detection tasks where some attack categories are subtle and easily overshadowed by dominant benign traffic. Consequently, hyperparameter tuning is a critical step for achieving a reliable balance

between accuracy, robustness, and efficiency.

Although the process of hyperparameter tuning can be "manual", an automated and optimization framework that systematically searches the hyperparameter space using a consistent objective and evaluation protocol is more efficient and helps standardize comparisons across algorithms and facilitates experimentation reproducibility.

There are several open source tools that allow for an automated optimization of hyperparameters, such as Optuna [27], Ray tune [91], Hyperopt [92] and Scikit-Optimize [93]. Some paid options more focused on Cloud are also available, with Google Vertex AI [94] and AWS SageMaker [95]. For the purposes of this work, we chose Optuna.

Optuna [27] is an open-source toolkit that automates hyperparameter search. Because it is framework-agnostic, it plugs into a wide range of machine-learning libraries and pipelines rather than being tied to any single ecosystem. To find the optimal values for each model, 10 Optuna trials were run, each exploring a different hyperparameter configuration. For every trial, we evaluated the model with five-fold cross-validation, averaging results across the folds to curb overfitting and obtain a more trustworthy estimate of out-of-sample performance.

2.4.8 Performance Metrics

In order to evaluate the performance of each algorithm, classification metrics are used. Let TP, FP, TN and FN denote, respectively, true positives, false positives, true negatives and false negatives for a given class. From these, the *precision* (positive predictive value), *recall* (sensitivity), *F1* score and *accuracy* are defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad \text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

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

These quantities are computed per class in the multi-class setting and then combined through averaging schemes. Macro averaging gives each class equal weight, making it informative under class imbalance; weighted averaging multiplies each per–class score by its test support, tracking the general performance while reflecting the empirical class distribution. Accuracy, is also reported but can be overly optimistic when negative/benign traffic dominates. For imbalanced data, macro-F1 is typically preferred because it penalizes poor performance on minority attacks that are operationally critical [96]. As a graphical aid to discrimination analysis, the Receiver Operating Characteristic (ROC) curve plots true-positive rate against false-positive rate and the area under this curve (ROC-AUC) summarizes ranking quality and is well established for

binary and multi-class problems [97, 96]. Nevertheless, with heavy class imbalance the ROC plot may appear beneficial while precision is poor. In such cases, precision–recall (PR) curves and their area (PR-AP) provide a more faithful view of retrieval quality for the positive (attack) class. The Average Precision (AP) computes the area under the PR curve using step-wise, interpolated, precision at each recall level. We therefore report ROC-AUC and, where relevant, PR-AP[96].

3 Related Work

The increasing integration of digital control into critical infrastructure has given rise to complex cyber-physical systems (CPS), where specialized, legacy communication protocols such as MODBUS in ICS and CAN bus in the IoV are now prime targets for cyberattacks. Despite the parallel security challenges these protocols face, a review of the literature reveals a significant gap. Existing research overwhelmingly focuses on developing domain-specific IDS. There is a clear absence of investigation into unified systems capable of processing and securing different protocols simultaneously. This work directly addresses this gap by proposing and evaluating a cross-domain IDS. The main goal is to develop a single reliable model for identifying malicious traffic across both of these critical environments, with ML to learn complex attack patterns from diverse network data. However, the transition from theoretical models to practical deployable systems is fraught with challenges that are remarkably consistent across different CPS domains. Chief among these are the persistent issues of data scarcity and realism, the inherent class imbalance of network data where benign traffic vastly outnumbers attack instances, and the methodological rigor required for model optimization. This chapter reviews the existing literature in both the ICS and IoV domains, specifically with MODBUS and CAN bus data, highlighting the parallel challenges, convergent solutions, and the critical research gaps that this work aims to address.

3.1 Intra-Vehicle Network - CAN

The public datasets created by the Hacking and Countermeasure Research Lab (HCRL) address the lack of public datasets in the IoV domain, specifically in intra-vehicular networks with CAN bus data, and lead to a large amount of research in ML-based IDS. Datasets such as CAN-Intrusion (OTIDS) [98], Survival Analysis (SAD) [99], Car-Hacking (CH) [100, 101] and the Car-Hacking Challenge (CHC) [102, 103] are widely used in the literature to evaluate the integration of different ML algorithms within IDSs. With a similar purpose, other datasets were made publicly available, such as the ROAD dataset [104], the Bus-CAN-Attack (BCA) [105], can-train-and-test (CTAT)[106], IVAD [107] and the CICIoV2024 [7] (see Table 3.1).

Table 3.1: Summary of CAN and MODBUS most used Public Datasets.

Ref.	Dataset	Year	Protocol	Articles
[7]	CICIoV2024	2024	CAN	[7, 108]
[107]	IVAD	2024	CAN	[107]
[106]	can-train-and-test	2024	CAN	[109, 110]
[105]	Bus-CAN-Attack	2024	CAN	[105]
[111, 112]	Edge-IIoTset	2023	MODBUS	[113]
[11]	ICS-Flow	2023	MODBUS	[11, 8, 2]
[114]	CICModbus	2023	MODBUS	[115]
[102, 103]	Car Hacking Challenge	2021	CAN	[116, 117, 118, 119, 30]
[120]	WDT	2021	MODBUS	[121]
[122, 123]	X-IIoTID	2021	MODBUS	[124]
[125, 126]	WUSTL-IIoT	2021	MODBUS	[124]
[104]	ROAD	2020	CAN	[127, 128, 129]
[130, 131]	TON_IoT	2020	MODBUS	[132, 133, 134]
[135, 136]	Cyber-security ModbusICS	2019	MODBUS	[137]
[138]	Electra	2019	MODBUS	[139]
[100, 101]	Car-Hacking	2018	CAN	[116, 107, 140, 141, 127, 142, 143, 144, 117, 109, 145, 30, 128, 36, 146, 129, 147, 105, 148]
[99]	SAD	2018	CAN	[107, 141, 127, 117]
[98]	CAN-Intrusion/OTIDS	2017	CAN	[141, 149, 117, 150, 105]

3.1.1 Binary Classification

Binary classification in CAN-bus IDS frame detection is a two-way decision, malicious versus benign, prioritizing rapid, low-complexity screening over fine-grained attack attribution. Because many public benchmarks are heavily imbalanced, models often report near-perfect metrics (accuracy/F1/AUC) but risk optimistic bias and limited transferability across datasets or vehicles. Recent studies span lightweight distance/similarity rules and linear models to tree ensembles and deep encoders. Results consistently show that feature design, reliable validation (such as stratified splits or cross-validation), and principled hyperparameter tuning are more important than the model family. We contextualize these findings below, focusing on what truly improves generalization to unseen traces rather than inflating in-sample scores.

Different studies [117, 141, 149] used the CAN-Intrusion dataset [98] for that purpose. Bonomo et al. [149], besides using the CAN-Intrusion dataset, but also created a data-

set of their own for the experiment. They concluded that for DoS attacks, the majority of the tested algorithms got an F1-Score of 1.0, specifically DT, RF and XGBoost, with KNN achieving 0.9997. When evaluating impersonation attacks, KNN maintained its score of 0.9997 while DT, RF and XGBoost dropped to 0.9951. However,in this study, hyperparameter optimization was not used to improve the performance of the algorithms. In another study, Kousar et al. [117] took advantage of more public datasets, such as the Car-Hacking, SAD and the Car Hacking Challenge, to develop a different and lightweight classifier for anomaly detection, based on the Pythagorean distance and achieving up to 99.50% accuracy for spoofing (Gear) attacks in the Car-Hacking dataset, 99.71% for spoofing in the Car Hacking Challenge, 98.722% for malfunction in SAD and 99.691% for fuzzy and impersonation attacks in the CAN-Intrusion dataset. The authors also opted for a 70/30 split in order to overcome the model overfitting problem, and the resulting ROC curves show that the area under the curve (AUC) value is above 0.95, demonstrating its reliability. With a different approach, Rai and Grover [141] also attempted to maintain a larger training sample by adding their own dataset, the Car-Hacking and SAD datasets, to the experiments. The chosen approach used LR for the Jaccard and cosine similarity-based binary classification, resulting in an outperformance of cosine across all datasets (in accuracy and F1-scores), achieving an F1-score up to 0.98 in their own dataset and 0.91 in the Car-Hacking dataset.

A recent study led by Fatahi et al. [116], leveraging one of the most used datasets in literature, the Car-Hacking [100, 101], and the Car Hacking Challenge dataset [102, 103], approached the CAN network vulnerabilities with the coupling of entropy-based genetic feature engineering with multi-classifier fusion. The authors considered the temporal and spatial aspects of the attacks, extracting new features based on a two-parameter genetic algorithm (2P-GA) and Shannon entropy. It then uses a classifier fusion through Ordered Weighted Averaging (OWA) with DT, RF and XGBoost, resulting in an accuracy of 99.92% and 99.50%, and an F1-score of 0.9996 and 0.9692 for anomaly-sensitive mode (with a sequence-level cut-off to increase model sensitivity to attacks) and normal mode (without the cut-off), respectively. However, this is a binary classification, leaving room for further improvements. Another study [118], based on the Car Hacking Challenge dataset [102, 103], achieved perfect F1-scores for RF and up to 0.99 with SVM.

Using the BERT model for in-vehicle binary classification, Li et al. [107] created the Cloud Collaborative-based Intrusion Detection and Prevention System (CC-IDPS). Additionally, the authors addressed the lack of datasets and attack diversity in publicly available ones by creating their own dataset, the In-Vehicle Attack Dataset (IVAD). This dataset is a compilation of real and simulated data with the Car-Hacking [100, 101] and SAD [99] datasets. The results of the final model show an F1-score of 0.9883 and an accuracy of 99.27%.

Recently, Oladimeji et al.[32] generated and tested their own dataset with the intention of identifying the responsible node for the attack. Using LR, RF, Gradient Boosting and Multilayer Perception, the study evaluates the model performance for binary classification to identify the occurrence of attacks. The results show that all models achieved perfect scores in all metrics (accuracy, precision, recall, and F1 score), using feature selection and fixed fine-tuned hyperparameters, highlighting the high performance of these algorithms in binary classification. On the other hand, Jha and Jaiswal [140] disregarded hyperparameter optimization and obtained an accuracy for DoS, fuzzy, gear, and RPM attacks of 93%, 0.99, 100%, and 100%, respectively, using only the RF classifier and a fixed selection for the hyperparameters.

Im et al. [143] propose a different architecture, using RF, kNN, MLP, and SVM as base classifiers for binary classification and a cross-check filter. After each prediction, the cross-check filter is applied, resulting in an improvement of the results. Nonetheless, unknown attacks remain difficult to detect, and the use of only one dataset increases this limitation. Kidmose and Meng [110] also tested multiple algorithms (LR, DT, RF, Gradient Boosting, Isolation Forest, MLP, BIRCH) in order to reduce the false positive rate (FPR). The authors concluded that features such as timestamp with a time delta, together, tend to work best, while using time delta alone often degrades FPR. In terms of models, RF can outperform F1 in some configurations, but Logistic Regression is the most consistently strong across scenarios in a binary analysis, using the can-train-and-test dataset. Another study [109], based on Federated Learning, also used the can-train-and-test dataset, achieving an accuracy of 99%, but without employing any hyperparameter optimization technique.

In [142], the authors designed an explainable transfer-learning ensemble IDS that should detect known and zero-day botnet attacks on in-vehicle CAN traffic. They also aim to open the "black box" by adding SHAP explanations. The reported results are 100% for all metrics, and no multi-class analysis in intra-vehicular networks. Althunayyan et al. [144] propose a multi-stage approach, with a multi-class classification first step and the use of an LSTM autoencoder as the second step, for detecting new and unseen attacks. The highlighted results show LSTM autoencoder F1-scores varied from approximately 0.81 to 0.95 against unseen attacks.

Neto et al., [7] created and tested the CICIoV2024 dataset, a recent and flexible labeled dataset, available in different formats, such as decimal, hexadecimal and binary. The authors used LR, AdaBoost, DNN and RF for a binary and multi-class classification, achieving perfect scores in the binary classification for AdaBoost and RF.

Table 3.2: Binary classification studies with CAN.

Art.	Dataset	Method/ Classifier	Data Balanc.	Feat. Eng.	Hyperp. Optim.	Key Res.
[32] 2025	Self-generated	LR, RF, Gradient Boosting, MLP	✗	✓	✗	Perfect scores in all models
[116] 2025	CH, CHC	KNN, DT, XGBoost, CatBoost, RF, LR, LightGBM	✓	✓	✓	Acc. 99.92%; F1. 0.9996
[7] 2024	CICIoV 2024	LR, AdaBoost, DNN, RF	✗	✓	✗	AdaBoost, RF: F1. 1.00
[107] 2024	IVAD, CH, SAD	BERT, K-means, LR, SVM, DT, CNN-LSTM	✗	✗	✗	BERT: Acc. 99.27%, F1. 0.9886
[140] 2024	CH	RF	✗	✓	✗	Acc. 93% (DoS), 99% (Fuzzy), 100% (Gear), 100% (RPM)
[141] 2024	Self-gen., OTIDS, CH, SAD	Jaccard, cosine similarity	✗	✗	✗	F1. 0.98 (own), 0.91 (CH)
[127] 2024	ROAD, CH, SAD	RF, LightGBM, LCCDE, DCNN, TAN, LSTM	✗	✗	✓	Near-perfect F1 for ROAD
[149] 2024	OTIDS, Self-gen.	KNN, DT, RF, XGBoost, K-Means	✗	✓	✗	DT/RF/XGB: F1. 1.0 (DoS); KNN: 0.9997 (impersonation)
[142] 2024	CH	XAI Ensemble TL	✗	✓	✓	Perfect scores in all metrics
[143] 2024	CH	RF, KNN, MLP, SVM	✗	✗	✗	kNN: 99.995% (DoS); SVM: 99.979% (Gear); MLP: 99.998% (RPM), 99.835% (Fuzzy)

(*continues on next page*)

(*continued from previous page*)

Art.	Dataset	Method/ Classifier	Data Balanc.	Feat. Eng.	Hyperp. Optim.	Key Res.
[110] 2024	CTAT	LR, DT, RF, GB, iForest, MLP, BIRCH	✗	✓	✗	Time features cut FPR; RF best
[144] 2024	CH	ANN, LSTM	✓	✓	✓	LSTM-AE: F1 0.81–0.95 (unseen)
[117] 2024	CH, OTIDS, SAD, CHC	Pythagorean distance ML	✗	✗	✗	Acc. 99.50% (Gear), 99.71% (Spoofing), 98.722% (Malfunction), 99.691% (Fuzzy/ Impersonation)
[109] 2024	CH, CTAT	Federated learning	✗	✗	✗	Acc.99% for CTAT
[118] 2024	CHC	SVM, RF	✗	✓	✗	RF: F1. 1.00; SVM: F1. 0.99

In sum, binary CAN-bus detection is valuable as a fast frontline, but its headline-perfect scores often reflect dataset bias and narrow operating points. Stronger practice pairs calibrated probability outputs with threshold analysis (ROC/PR). Lightweight feature sets with tuned tree ensembles or regularized linear models tend to offer the best accuracy–latency trade-off, while one-class or reconstruction methods help surface zero-day behavior. Still, coarse "attack vs. benign" decisions limit response planning. This motivates the shift to multi-class labeling and, ultimately, cross-domain models that preserve speed while improving specificity and transferability.

3.1.2 Multi-Class Classification

Multi-class classification on CAN traffic goes beyond simply flagging "attack vs. normal" and aims to discriminate among specific attack types, such as DoS, fuzzy injection, gear/RPM spoofing, replay, etc. This setting is significantly more challenging due to severe class imbalance and dataset heterogeneity across benchmarks. Recent work spans traditional ensembles and margin-based learners to sequence and representation models (CNN/ LSTM and Transformer variants), with careful preprocessing and hyperparameter tuning proving decisive. Below, in Table 3.3, a review of representative studies is presented, highlighting methodological choices and where performance gains translate—or fail to translate—across datasets and attack classes.

Samir et al. [105] tested algorithms such as Long Short-Term Memory(LSTM) and Convolutional Neural Network (CNN) with hyperparameter optimization for multiple datasets. Different datasets were used (CAN-Intrusion, Car-Hacking), including their own (Bus-CAN-Attack), obtaining with LSTM an F1-Score of 1.0 for fuzzy, spoofing (gear) and spoofing (rpm) attacks on the Car-hacking dataset, 1.0 for normal and DoS attacks on the Bus-CAN-Attack, and 0.98 on the fuzzy attacks on the CAN-Intrusion dataset. However, they were able to improve the results for the replay attack on the Bus-CAN-Attack to an F1-score of 1.0, while decreasing their best result on the CAN-Intrusion to 0.96 for Fuzzy attacks. Barletta et al. [150] achieved an accuracy and F1-scores of 0.99 for DT and RF under evasion attacks with the CAN-Intrusion dataset while maintaining the original class distribution. Still, these results dropped significantly (50%) when faced with inference attacks. However, no hyperparameter optimization was used to improve the performance of the algorithms in [150].

Alalwany and Mahgoub [119], with the Car Hacking Challenge Dataset[102, 103], used a stacked ensemble of RF, DT, and XGBoost. The preprocessing steps included feature selection and class imbalance strategies such as NearMiss undersampling, and the final model achieved an accuracy of 98.5% and an F1-score of 0.985. Along the same line of sequence modeling, Fu et al. [30], converted the CAN messages into Can Byte Sentence (CBS) to be used as a language-like sequence for the BERT model. The final model (IoV-BERT-IDS) shows an accuracy of 99.96% and an F1-score of 0.9996 on one of the datasets used [102, 103] and, an accuracy of 99.97% and F1-score of 0.9985 on the other [100, 101].

Extending beyond vehicles, a satellite-focused study [148] used the Car-Hacking dataset [100, 101] and the UAVCAN dataset [151] to train and test their IDS for satellites that use CAN, the CANSat-IDS. The study uses ANN and K-SVM, and they were able to obtain in their final model an F1-score of 0.9112 for the replay attacks, 0.9818 for the fuzzy injection and, 0.9986 for the DoS message flooding. However, the results are for a unified dataset and the UAVCAN is a CAN-variant dataset, with an additional application-layer protocol on top of the standard CAN.

Wu and Tao [147] address the importance of hyperparameter optimization and confirm the differences in detection performance in different attacks when testing isolated models with DT, RF, ET, LightGBM, and Adaboost with their default hyperparameters against their proposed model, a stacking ensemble model with DT, ET and XGBoost and an optimized hyperparameter approach, resulting in an accuracy of 99% for DoS attacks. Another study [146] chose to use the DistilBERT model to identify complex patterns for the vehicle networks, resulting in perfect F1-scores for all types of attacks, including DoS, Fuzzy, Gear Spoofing, and RPM Spoofing. However, the dataset was balanced, which does not accurately represent a real-world scenario. Nazeer et al. [145] proposed the DeepXG, a model composed of XGBoost and deep learning models. It

initially uses XGBoost to train and extract features that are later used to feed the deep learning model. The final model is then compared against RF, AdaBoost, XGBoost alone, and (reported) CNN-LSTM and achieved the best F1-score of 0.9760. Another study [144] proposes a multi-stage approach, using ANN for the first identification of seen attacks and a second phase resourcing to a binary classification for anomaly detection. The results show F1-scores that exceed 0.99 for the multi-class classification with ANN.

In [36], the authors used clustering techniques. First, they use a rule-based detector for detecting and alerting locally based on the rules that match. If they do not match, the open-set model (CLUSTER) is applied, and if no class is close enough to the centroids, it is considered as unknown and sent to the cloud to be forwarded for expert triage, incremental learning of the model, and rule generation, which are then pushed back to vehicles. This model achieves F1-scores from 0.9808, in RPM attacks up to 0.9937 in DoS attacks. Integrating an autoencoder (AE) at the packet level with a time-embedded transformer at the sequence level, Le et al. [129] achieved perfect scores for the Car-Hacking dataset and masquerade attacks in the ROAD dataset. Nguyen et al. [128] however, tackled the issue on the same datasets with a variational autoencoder (VAE) and adversarial reinforcement learning in stages. A VAE learns a compact latent space from mostly unlabeled CAN traffic, and a semi-supervised AERL classifier focuses training on hard and rare cases. The teacher model is trained with a small labeled subset and generates pseudo-labels for the unlabeled data, and the student model is then retrained on the combined real with the pseudo-labels for deployment. The results highlight a better performance for known attacks for the teacher and student models in the Car-Haking dataset, achieving F1-scores of 0.9999, while in the ROAD dataset, the results show a small decrease to 0.9852 in the teacher model and 0.9921 in the student model. In unknown attacks, F1-scores show a high performance for Car-Hacking with 0.9979 in spoofing attacks, while in the ROAD dataset, the results were lower, with values ranging from 0.8405 to 0.9156.

Guerra et al. [127], evaluated algorithms such as RF, LightGBM, LCCDDE, DCNN,TAN and LSTM on the Car-Hacking, ROAD and SAD datasets, and applied SMOTE to the ROAD training set. On the Car-Hacking and SAD datasets, the majority of the models achieved near-perfect scores with F1-scores from 0.99 to 1.00, though the results for the ROAD dataset demonstrate a struggle in a multiclass scenario with scores ranging from 0.3144 in "reverse light on masquerade" attack with LCCDE, to 1.0 in fuzzing attack with RF. Huang et al. [109] proposed an intrusion detection method named FED-IoV. On the Car-Hacking dataset, the best local model reached 0.9917, while the federated model reached up to 0.9907 (K = 5) and averaged 0.9851. Neto et al., [7], as already mentioned, tested the binary and multi-class classification on their dataset. Using LR, AdaBoost, DNN and RF, the authors found decreased values from the binary results.

The algorithms that had the best performance were RF and and DNN, with F1-scores of 0.76 and 0.78, respectively. These results were improved in another study [108], using the same dataset and a GA-based Hyperparameter Optimization. The results showed a significant improvement, with F1-scores up to 0.9679 with DNN and 0.9745 with RF.

Table 3.3: Multi-class classification studies with CAN.

Art.	Dataset	Method/ Classifier	Data Balanc.	Feat. Eng.	Hyperp. Optim.	Key Res.
[127] 2024	ROAD, CH, SAD	RF, LighGBM, LCCDE, DCNN, TAN, LSTM	✗	✗	✓	Near perfect F1. for CH and SAD; F1. ranging from 0.3144 to 1.0 for ROAD
[119] 2024	CHC	RF, DT, XGBoost	✓	✓	✓	Acc. 98.5%, F1. 0.985
[144] 2024	CH	ANN, LSTM	✓	✓	✓	ANN: F1. >0.99
[150] 2024	OTIDS	DT, RF, GB, LiR, KNN	✗	✓	✓	DT, RF: F1. 0.99 for evasion attacks
[145] 2024	CH	XGBoost, DL	✓	✓	✗	F1. 0.9760
[30] 2024	CH, CHC	LLM	✗	✗	✗	Acc. 99.96%, F1. 0.9996
[128] 2024	CH, ROAD	VAE and AERL	✗	✗	✗	Known attacks: F1. 0.9999 (student and teacher) in CH
[36] 2024	CH	Clustering-based	✗	✗	✗	F1. 0.9808 in RPM; F1. 0.9937 in DoS
[109] 2024	CH, CTAT	FL-based	✗	✗	✗	Acc. 99.17% with non-federated model and 0.9851 with federated model
[146] 2024	CH	DistilBERT	✓	✓	✗	Perfect F1-scores

(*continues on next page*)

(*continued from previous page*)

Art.	Dataset	Method/ Classifier	Data Balanc.	Feat. Eng.	Hyperp. Optim.	Key Res.
[129] **2024**	ROAD, CH	time-embedded transforme and AE	✗	✓	✗	Perfect F1 in CH
[147] **2024**	CH	DT, ET, XGBoost	✗	✗	✓	F1. 0.9698, Acc. 99.21%
[108] **2024**	CICIoV 2024	LR, Adaboost, DNN, RF	✗	✗	✓	RF:Acc. 99.64%, F1. 0.9745; DNN:Acc. 0.99.29%,F1. 0.9679
[105] **2024**	BCA, CH, OTIDS	LSTM, CNN	✗	✗	✓	LSTM: F1. 1.0 for fuzzy, spoofing(gear), spoofing(rpm), DoS, Normal
[148] **2024**	CH	ANN, K-SVM	✗	✓	✓	F1. 0.9986 for DoS
[7] **2024**	CICIoV 2024	LR, AdaBoost, DNN, RF	✗	✓	✗	RF, DNN: F1. 0.76 and 0.78

Overall, multi-class CAN intrusion detection is maturing. While stacked ensembles and Transformer-style encoders have lifted headline scores on popular benchmarks, transfer across datasets, vehicles, and unseen variants remains the bottleneck. Pragmatically, hierarchical labeling (benign, attack type, subtype) and disciplined hyperparameter tuning give the best path for a deployable specificity, setting the stage for cross-domain IDS that can maintain precision without sacrificing coverage.

3.2 Industrial Systems

A natural twin to the CAN-centric IoV line is MODBUS in ICS: two legacy, lightweight field protocols that move safety-critical signals without built-in security. In addition, the scarcity of available datasets [11, 13, 8, 152] is a similar constraint, as shown in Table 3.1, with the majority of recent research being focused on the same datasets. Since both protocols share similar issues, the approaches for intrusion detection tend to converge.

3.2.1 Binary Classification

A recent study [115] used a compilation of datasets in order to improve their evaluation. The authors used the CICModbus [114], containing attack and normal traffic, and a second dataset, obtained from the Center SAU Water system [153], containing only normal traffic. The results show that XGBoost and an ensemble model (RF and Naive Bayes (NB)) achieved the best results in the binary classification problem, with an accuracy of 98.41% and 97.2%, respectively. However, the hybrid dataset is highly balanced, with 50% of attacks and 50% of normal traffic, which does not correspond to a realistic scenario.

Jadidi et al. [132], tested the vulnerability of deep-learning approaches to gradient-based adversarial attacks (FGSM), retraining the model with generated attack samples. The authors used the TON_IoT dataset and chose ANN to act as the detector model, achieving an F1-score of 0.9676 in the first phase, without FGSM. Under FGSM, with the epsilon ranging from 0 to 1 (noise), the results drop significantly to 0.8005, 0.1612 and 0. After retraining the model with 10% and 20% adversarial samples, the scores changed drastically to 0.0004 and 0.9053, respectively.

From an evasion-resilience angle, Mumrez et al. [152] worked the problem from a different point. In addition to creating their own dataset, they assumed a threat model in which the attacker has perfect knowledge of the target's IDS pipeline and makes changes to the most influential features for the models, trying to evade the IDS. The results show that SVM was capable of detecting accurately some of the attacks, with an accuracy and F1-score of 94.8% and 0.974 for DoS, 88.9% and 0.941 for Data modification and 0, in all metrics for MiTM attacks. Nonetheless, all these results dropped significantly, to 0, after the evasion techniques were applied. Contrary to the SVM performance, RF demonstrated a superior response, achieving an accuracy and F1-score of 94.8% and 0.974 for Dos, 88.9% and 0.941 for data modification and, 5.6% and 0.011 for the MiTM. After the modification of features, the results surprised for the data modification attacks, where RF achieved an accuracy and F1-Score of 100% and 1.0. With the goal of addressing the shortage of public datasets, Zhang et al. [15] generated data with the SMOD tool and introduced MODLSTM, a model tailored to detect DoS activity, achieving an accuracy of 90.037%.

Saba et al. [133] aim to build a practical intrusion-detection approach for smart-city IoT by training classic ML and simple DL models on TON-IoT dataset using binary classification with SMOTE. The study evaluates RF, a voting ensemble (LR, RF, GNB), ANN and 1D-CNN. Regarding the MODBUS subset of the dataset, the voting classifier has achieved the highest accuracy of 99.7% and an F1-score of 0.989, with RF in second place with an accuracy of 99.3%. The models 1D-CNN and ANN had the lowest results, with an accuracy of 67.03% each. Another approach using the same dataset, introduced by Gueye et al. [134], focuses on a compact deep-learning pipeline that embeds discre-

te protocol fields into dense vectors before classification with simple backbones (fully connected "Linear," CNN, or LSTM). On the TON_IoT MODBUS subset, using only a handful of MODBUS function code features for binary and multi-class detection, the embedding-enhanced models consistently outperform their non-embedding counterparts. Results show that CNN with embedding attains 98.91% accuracy and an F1-score of 0.99 for binary detection.

Recently [137] used the Cyber-securityModbusICS to test two unsupervised machine learning algorithms, One-Class SVM with PCA and Isolation Forest (iForest) for intrusion detection. Although the authors optimized the models hyperparameters through grid-search, the results in the test set demonstrated a poor performance with and F1-score of 0.573 in One-Class SVM and, 0.179 for iForest.

Table 3.4: Binary classification studies with MODBUS.

Art.	Dataset	Method/ Classifier	Data Balanc.	Feat. Eng.	Hyperp. Optim.	Key Res.
[115] 2025	CICModbus	DT, RF, SVM, KNN, NB, XGBoost, DNN	✓	✓	✗	XGBoost: Acc. 98.41%; Ensemble (RF,NB): Acc. 97.20%
[137] 2025	Cyber-security ModbusICS	One-Class SVM with PCA, Isolation Forest	✗	✓	✓	One-Class SVM: F1. 0.573; iForest: F1. 0.179
[2] 2024	ICS-Flow	DT, RF, ET, Gradient Boosting, CatBoost, LightGBM, XGBoost	✗	✓	✗	Near perfect results for al models
[8] 2024	ICS-Flow	LR, DT, XGBoost, RF, ANN, LightGBM, SVM	✗	✓	✗	Near perfect performance for all methods. XGBoost got perfect results.
[154] 2023	Self-generated	PCA, SVM	✗	✓	✓	SVM-HNIDS has a detection rate of 98.6%, PCA-HNIDS achieves 99.3%

(*continues on next page*)

(*continued from previous page*)

Art.	Dataset	Method/ Classifier	Data Balanc.	Feat. Eng.	Hyperp. Optim.	Key Res.
[134] 2023	TON_IoT	Linear, CNN, LSTM	✗	✓	✗	CNN(embedding): Acc. 98.91%, F1. 0.99
[152] 2023	Self-generated	RF, SVM	✗	✗	✓	RF: F1.(before feature mod.) 0.941, F1.(after feat. mod.) 1.0
[15] 2022	Self-generated	MODLSTM	✗	✗	✗	DoS-focused model; Acc. 90.037%
[133] 2022	TON_IoT	Voting Classifier(LR RF GNB), RF, ANN, 1D-CNN	✓	✗	✗	Voting Classifier(LR RF GNB): Acc. 99.7%, F1. 0.989
[132] 2022	TON_IoT	ANN, FGSM	✗	✗	✗	F1. 0.9676
[155] 2021	Self-generated	FNN, LSTM, FNN-LSTM, NDAE RF	✗	✓	✓	FNN-LSTM: F1 of 0.9968

3.2.2 Multi-Class Classification

To handle imbalance explicitly, Popoola et al. [124] designed a multi-stage deep learning (MSDL) method, using DNN to use against highly imbalanced datasets, such as X-IIoTID and WUSTL-IIoT datasets. Initially, a multi-class model is used on all data, and if any class misses a set F1-score threshold, a split to Normal vs Attack is made, and then, if needed, repeatedly "peel off" the hardest minority attack class with binary classifiers until every class meets the threshold. At runtime, traffic is gated by Normal/Attack and, if Attack, routed to the attack multi-class or the learned per-class binary splits. The results of the base approach are then compared to different data-sampling techniques, with the conclusion that the proposed model achieves better results with near-perfect scores, such as 0.9971 with WUSTL-IIoT and 0.9978 with X-IIoTID.

Trivedi et al. [113] propose a stacking-based IDS using the Edge-IIoTset dataset. The model takes advantage of multiple classifiers, such as NB, kNN, LR and MLP, in an ensemble learning approach and employs a meta-classifier to build up their predictions. The performance of the proposed IDS has an overall accuracy and F1-score of 99.88%

and 0.9988 in a multi-class setting with LR as a meta-classifier and an accuracy and F1-score of 99.73% and 0.9973 with MLP as the meta-classifier.

Abdelkhalek et al. [156] address a different weak point, distributed energy resources (DER), since communication from DER to the smart grid is made using the MODBUS protocol. The research used their own dataset and tested NB, DT, RF, SVM and ANN with 10-fold cross-validation (CV), to find the best hyperparameters. In pre-deployment evaluation with live-attack testing, the ANN achieved the highest detection accuracy, 98.47%, followed by RF , 93.44%, being the first selected for the testbed implementation. After the deployment, another performance evaluation is made with results demonstrating a 98.4% detection accuracy. Also in the smart-grid context, Elrawy et al. [154] design an Hybrid Network IDS (HNIDS), combining machine learning-based anomaly and signature based approaches to detect and classify MiTM attacks. For the detection step, the authors used two methods: a PCA outlier detector trained on normal traffic only and a soft-margin SVM trained on normal plus ARP-poisoning cases to improve robustness to load and/or fault fluctuations. After detection, the study classifies MITM into four end classes: attacking the router, two-way, attacking the controller, and unknown technique. It does this hierarchically, using first a binary split (poisoning vs non-poisoning via an ARP-poisoning indicator), then a binary SVM for poisoning (router vs two-way) and a MAC-check for non-poisoning (controller vs unknown). In summary, it applies a multi-class analysis through the implementation of binary steps. The results show that SVM-HNIDS has a detection rate of 98.6%, while PCA-HNIDS achieves 99.3%.

Gao et al [155], at the temporal granularity level, emphasize the need of an omni attack detector, an IDS that is capable of detecting temporally uncorrelated attacks (single-packet anomalies) and correlated attacks (time-dependent patterns). In order to evaluate this, the authors tested four IDSs on a self-generated dataset with the two types of attacks (correlated and uncorrelated): FNN, LSTM, FNN-LSTM and, nonsymmetric deep autoencoder with a random forest (NDAE RF). The results show that the ensemble model (FNN-LSTM) got the highest score with an F1 of 0.9968, and the FNN the lowest, with an F1 of 0.874.

Within critical energy contexts, Siniosoglou et al. [10] recognized the vulnerability around smart grids that use legacy protocols such as MODBUS. The authors present MENSA, a DL-based anomaly detection and classification model with a capability of detecting 13 MODBUS cyberattacks. Results highlight the performance of the model within different environments, such as an F1-score of 0.767 in the hydropower plant, 0.734 in a power plant, 0.759 in a substation and, 0.730 in the smart-grid lab.

Dehlaghi-Ghadim et al.[11] introduced the ICS-Flow dataset and benchmarked several machine-learning models such as DT, RF and ANN, using predefined hyperparameters and a selected feature set, while preserving the dataset's original class imbalance.

RF delivered the strongest performance, and its confusion matrix highlighted where certain flows were mistakenly labeled. A separate study [2] also leveraged ICS-Flow to compare DT, RF, Extra Trees, Gradient Boosting, CatBoost, LightGBM, and XGBoost with an emphasis on feature selection and engineering. In that work, the chosen pipeline, with RF, achieved near-perfect F1-scores in the multi-class setting.

Adopting a feature-centric strategy, Amer and Elboghdadly [8] evaluated LR, DT, XGBoost, RF, ANN, LightGBM, and SVM in a multiclass setting and reported strong performance across all models. Notably, they did not disclose results for the IP-Scan category—the class that typically proves most difficult on ICS-Flow [8].

Müller et al. [121] contrasted intrusion detection and classification using cyber-physical features versus network-only features on the dataset from [120]. Incorporating physical variables raised macro-averaged F1-scores by roughly 15 points on average across RF, KNN, SVM, and ANN. However, their pipeline employed both undersampling and oversampling, altering the dataset's original imbalance. In a different research, Calvino et al. [139] ran a multi-class IDS and tackled imbalance with targeted oversampling and GANs, then transfer-learn a CNN from Electra-Modbus to Electra-S7Comm. Balancing lifts overall accuracy from 46% to 92%, while adding transfer learning reaches 99.6% with strong per-class F1. Beyond accuracy, the transferred model recognizes specific attacks, including improved detection of a previously unseen S7Comm "forced-error in response", indicating cross-protocol generalization to new attack variants.

Table 3.5: Multi-class classification studies with MODBUS.

Art.	Dataset	Method/ Classifier	Data Balanc.	Feat. Eng.	Hyperp. Optim.	Key Res.
[139] 2025	Electra	CNN with Transfer Learning	✓	✓	✓	Acc. 99.6%
[113] 2025	Edge-IIoTset	NB, kNN, LR, MLP	✗	✗	✗	LR: Acc. 99.88%, F1. 0.9988; MLP: Acc. 99.11%, F1. 0.9973
[124] 2025	X-IIoTID, WUSTL-IIoT	DNN	✗	✓	✗	F1. 0.9971 with WUSTL-IIoT, F1. 0.9978 with X-IIoTID

(*continues on next page*)

(*continued from previous page*)

Art.	Dataset	Method/ Classifier	Data Balanc.	Feat. Eng.	Hyperp. Optim.	Key Res.
[2] **2024**	ICS-Flow	DT, RF, ET, Gradient Boosting, CatBoost, LightGBM, XGBoost	✗	✓	✗	RF , ET, and CatBoost: F1. 0.999
[8] **2024**	ICS-Flow	LR, DT, XGBoost, RF, ANN, LightGBM, SVM	✗	✓	✗	Strong results across all models; IP-Scan class not reported
[154] **2023**	Self-generated	PCA, SVM	✗	✓	✓	SVM-HNIDS has a detection rate of 98.6%, PCA-HNIDS achieves 99.3%
[134] **2023**	TON_IoT	Linear, CNN, LSTM	✗	✓	✗	LSTM(embedding): Acc. 98.06%, F1. 0.99
[11] **2023**	ICS-Flow	DT, RF, ANN	✗	✓	✗	RF: Avg Acc. 98.4%, F1. 0.9995 for DDoS and 0.9970 for Normal; DT: F1. 0.9995 for DDoS and 0.9965 for Normal; ANN: F1. 0.9980 for DDoS and 0.9964 for Normal
[121] **2022**	WDT	RF, KNN, SVM, ANN	✓	✓	✗	Adding physical features raised macro F1; uses
[156] **2022**	Self-generated	NB, DT, RF, SVM, ANN	✓	✓	✓	ANN: Acc.98.4%
[155] **2021**	Self-generated	FNN, LSTM, FNN-LSTM, NDAE RF	✗	✓	✓	FNN-LSTM: F1 of 0.9968

(*continues on next page*)

(*continued from previous page*)

Art.	Dataset	Method/ Classifier	Data Balanc.	Feat. Eng.	Hyperp. Optim.	Key Res.
[10] 2021	Self-generated	DNN, AE–GAN	✗	✓	✓	F1: 0.730 - 0.767

Taken together, the IoV and ICS lines reveal parallel constraints and convergent methodological responses, such as feature engineering, sequence models, ensembles, hierarchical or staged classification, and hyperparameter optimization. Both domains have persistent open issues with binary and multi-class performance gaps and generalization across datasets and protocols. These patterns, across MODBUS and CAN, motivate a cross-domain IDS that unifies detection while being robust to a generally imbalanced, realistic traffic and adversarial behaviors. Legacy-IDS fills the literature gap and advances state-of-the-art CAN and MODBUS intrusion detection by coupling automated Optuna hyperparameter sweeps with ensemble models, enabling a single system to target multiple tasks, such as binary, multi-class, or both. By focusing on two legacy protocols that share comparable weaknesses, Legacy-IDS delivers true cross-domain coverage within one streamlined workflow, simplifying model design while improving adaptability and performance.

4 Methodology

In this chapter, we'll outline the datasets used in this work, the preprocessing pipeline applied to both CAN and MODBUS traffic, and the development of the CAN-BUS Dataset Converter, finalizing with the introduction of the Legacy-IDS framework architecture.

4.1 Literature Review

A systematic literature review was conducted to map current approaches to intrusion detection in CAN/IoV and MODBUS/ICS, guided by predefined research questions. The review focused on peer-reviewed publications and high-quality conference proceedings within a defined time window. The search focused on Scopus, Google Scholar and IEEEXplore, for works published between 2021 and 2025. Search strings combined protocol terms, IDS terminology, and evaluation terms using Boolean operators and wildcards. Some of these strings were:

- (''Internet of Vehicles'' OR ''IoV'' OR ''CAN Bus'' OR ''CAN protocol'') AND (''intrusion detection'' OR ''anomaly detection'') AND (''machine learning'')
- (''Industrial Internet of Things'' OR ''IIoT'' OR ''Industrial Control Systems'' OR ''ICS'') AND (''Modbus/TCP'' OR ''Modbus'') AND (''intrusion detection'' OR ''anomaly detection'') AND (''machine learning'')
- (''cross-domain'' OR ''multi-domain'') AND (''Intrusion detection'' OR ''IDS'') AND (''machine learning'' OR ''adaptive learning'') AND (''IoV'' OR ''ICS'' OR ''IIoT'')

The eligibility criteria included peer-reviewed journal articles and conference papers that addressed intrusion detection for CAN, MODBUS, or cross-domain IDS and reported methods with quantitative results. We excluded non-technical opinion pieces, studies lacking evaluative results, works outside the defined time window, duplicates, and inaccessible full texts. Screening and selection proceeded in two stages: first, titles and abstracts were screened against the inclusion and exclusion criteria; second, the remaining records underwent full-text assessment. Duplicates were removed before screening, and any uncertainties were resolved by revisiting the research problem and eligibility criteria to reach consensus.

4.2 Datasets

Two public benchmarks were used to represent each domain under realistic class imbalance. For MODBUS data, we used the ICS-Flow dataset [11] and for CAN, we used the CICIoV2024 [7].

4.2.1 ICS-Flow

The ICS-Flow dataset [11] is a recent benchmark for intrusion detection in Industrial Control Systems, built on a virtual bottle-filling factory testbed (ICSSIM) with two PLCs and multiple HMIs communicating over MODBUS (default port 502). The testbed generates network traffic and process logs under normal operation and under four common attack classes, Reconnaissance (IP/Port scan), Distributed Denial of Service, Man-in-the-Middle (false-data injection), and Replay, implemented with standard tools, such as Nmap and Scapy, and scripted to allow system recovery between runs, giving realistic traces. Packet capture occurred at a switch (TCPdump), then flows were aggregated every 500 ms with the authors ICSFlowGenerator, producing a flow dataset with 54 columns: 50 features (flow, general, TCP) and 4 label fields.

Table 4.1: Description of ICS-Flow dataset features.

#	Feature	Description	Type
1,2	(s/r)Address	Sender/Receiver IP address or MAC address of flow	str
3	Protocol	Packet type and network protocol	str
4,5	start/end	Timestamp of the first/last packet in the flow	float
6,7	startOffset/ endOffset	Seconds (from capture start) to first/last packet of the flow	float
8	duration	Flow interval, the difference between end and start timestamps	float
9,10	(s/r)Packets	Sent or received packets count within the flow	int
11–14	(s/r)BytesMax/ BytesMin	Maximum/Minimum bytes sent or received in one packet within the flow	int
15,16	(s/r)BytesAvg	Average bytes sent or received per packet within the flow	float
17,18	(s/r)Load	Sent or received bits per second	float
19–22	(s/r)PayloadMax/ PayloadMin	Maximum/Minimum bytes sent or received as payload in one packet within the flow	int
23,24	(s/r)PayloadAvg	Average payload bytes of sent or received packets	float
25,26	(s/r)InterPacket	Average send or receive inter-packet arrival time	float
27,28	(s/r)ttl	Average sending or receiving time-to-live	float

(*continues on next page*)

(*continued from previous page*)

#	Feature	Description	Type
29–32	(s/r)AckDelayMax/ AckDelayMin	Maximum/Minimum interval between packets and their acknowledgement at sender/receiver node	float
33,34	(s/r)AckDelayAvg	Average interval between packets and their acknowledgement at sender/receiver node	float
35–46	(s/r)AckRate/ FinRate/ PshRate/ RstRate/ UrgRate	Rate of packets sent or received containing the ACK/FIN/PSH/RST/URG flags	float
47,48	(s/r)WinTCP	Sender or receiver average TCP window advertisement	float
49,50	(s/r)FragmentRate	Sender or receiver fragmentation rate of TCP packets	float
51	IT-B-Label	0 for normal, 1 for attack	int
52	IT-M-Label	`normal` for normal and `attack-name` for malicious	str
53	NST-B-Label	0 for normal, 1 for attack	int
54	NST-M-Label	`normal` for normal and `attack-name` for malicious	str

Overall, 25M raw packets were collected and condensed into 45719 flows. The authors also provide the raw PCAP, attack log, and process state snapshots to support supervised and semi-unsupervised studies. Class imbalance is intentional, providing a realistic scenario and both binary (normal/attack) and multi-class labels are included [11].

4.2.2 CICIoV2024 Dataset

The first phase of the experiments used the CICIoV2024 dataset [7], a recent, wide-scope benchmark for IoV cybersecurity focused on CAN traffic. It was built from the intact internal network of a 2019 Ford vehicle and relies on the standard CAN protocol for ECU-to-ECU messaging. CAN is fast and simple, but it lacks native security, leaving room for abuse, hence the need for intrusion detection in modern vehicles. The dataset has both normal and attack activity and spans six labels: benign plus five attacks, such as Denial-of-Service, Gas Spoofing, Speed Spoofing, Steering Wheel Spoofing, and RPM Spoofing. These attacks mirror practical methods seen on the CAN bus, from flooding messages to altering critical control values like speed, steering angle, and throttle position.

A notable trait of CICIoV2024 [7] is realism, not only in how the traffic looks but also in its class skew, with the benign samples dominating, and the rarer attack classes being harder to catch, which pressures IDS design. The feature set (see Table 4.2) is provided in three encodings, a binary, decimal, and hexadecimal, offering multiple ways to parse the data. In this study, we worked exclusively with the decimal view, treating all fields

as numeric inputs for machine-learning models.

Table 4.2: Description of CICIoV2024 dataset features.

Feature	Description
ID	Represents the message priority and specifies the type of data being transmitted—not unique
DATA_0	Byte 0 of the transmitted data
DATA_1	Byte 1 of the transmitted data
DATA_2	Byte 2 of the transmitted data
DATA_3	Byte 3 of the transmitted data
DATA_4	Byte 4 of the transmitted data
DATA_5	Byte 5 of the transmitted data
DATA_6	Byte 6 of the transmitted data
DATA_7	Byte 7 of the transmitted data
label	Indicates whether the traffic is benign or an attack
category	Specifies the traffic classification as benign, DoS, or spoofing
specific_class	Specifies the precise class of the traffic as benign, DoS, spoofing-GAS, spoofing-RPM, spoofing-SPEED, and spoofing-STEERING_WHEEL

4.3 Pipeline Structure

This work is divided into three phases as depicted in Figure 4.1. In the first phase, we evaluate our problem using a CAN dataset with algorithms such as Random Forest, Extra Trees, XGBoost, AdaBoost, Logistic Regression and Deep Neural Networks. In the second phase, we evaluate the problem using a MODBUS dataset with the same algorithms, supplement by Artificial Neural Networks. After testing all algorithms for both datasets, we selected the top three to use as possible choices for our IDS, as described in Figure 4.1. The combination of the first two steps is the foundation of the proof of concept for this work and and serves as the basis for the cross-domain IDS.

In the first evaluation, using the CICIoV2024 dataset, we ran a combination of preprocessing steps for each algorithm and evaluated two different approaches. One uses a split of 80/20 train/test, and another uses the entire dataset on a 10-fold CV. For both approaches, we conducted 10 trials in Optuna to identify the optimal hyperparameters and evaluated each algorithm using the best values obtained. In the split approach, we finalize with the evaluation of the model on the test set. The second phase shared the main preprocessing steps with the phase I and was tested using an 80/20 split. The 80% training set was also used to find the optimal hyperparameters using Optuna, and the

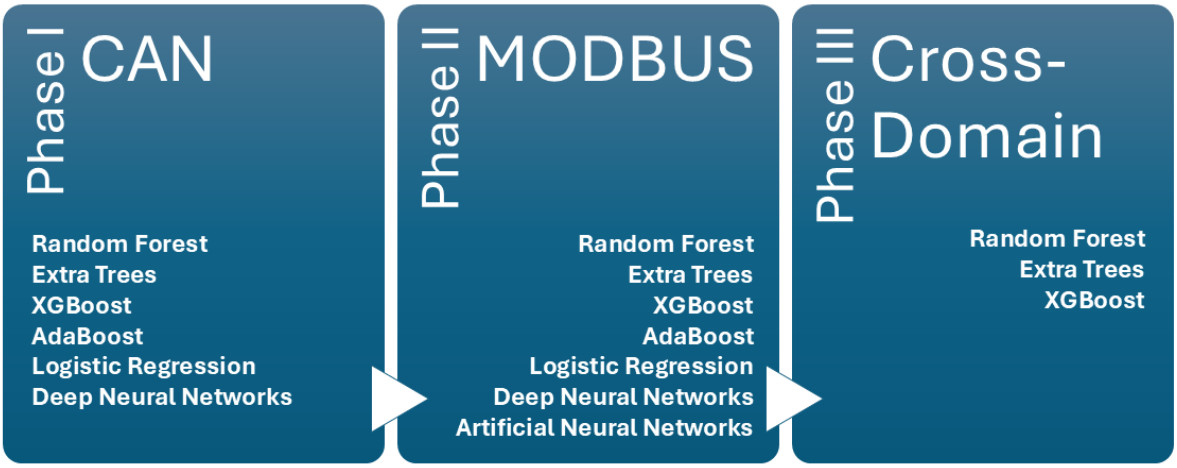


Figure 4.1: Study workflow.

validation of each model was performed on the 20% test set. The key stages of both phases are described in Figure 4.2 and the findings for both phases are already published [157, 158].

The final phase of Legacy-IDS is delivered as a Python command-line toolkit that consolidates training, batched evaluation, and streaming detection so the full research methodology can be rerun without custom notebooks, as depicted in the schematic workflow in Figure 4.3. The orchestrator accepts CAN traces that follow the CICIoV2024 layout or MODBUS flows organized like ICS-Flow, inspects the incoming columns to detect the protocol, applies the appropriate cleaning and feature engineering path, and then hands the resulting feature matrix to the modeling stack. From this single entry point, it is easy to switch among Random Forest, Extra Trees, or XGBoost, decide between binary, multi-class, or dual-task runs, and toggle options such as protocol-aware stratified sampling, CAN group-based splitting, standard or min–max scaling, and Optuna search budgets. Every optimization trial evaluates macro-F1 under stratified or group folds, and the best configuration persists as a scikit-learn pipeline, exporting models artifacts with matching JSON and text summaries.

The workflow continues and rebuilds the protocol-specific feature frame for a given dataset, loads the saved pipeline, and either reuses the original hold-out split or scores the entire dataset depending on the selected flags. The reporter then writes accuracy, macro/micro F1, precision, recall, confusion matrices, and (when probabilistic outputs are available) ROC and PR curves, providing consistent comparisons across CAN and MODBUS experiments. For live or batched inference, it reads arbitrary CAN or MODBUS captures, regenerates the engineered features, reloads the correctly tagged model, and writes predictions, optionally running the same evaluation routine when ground-truth labels are supplied. Because data acquisition, feature extraction, optimization and learner factories are implemented as modular components, extending Legacy-IDS to

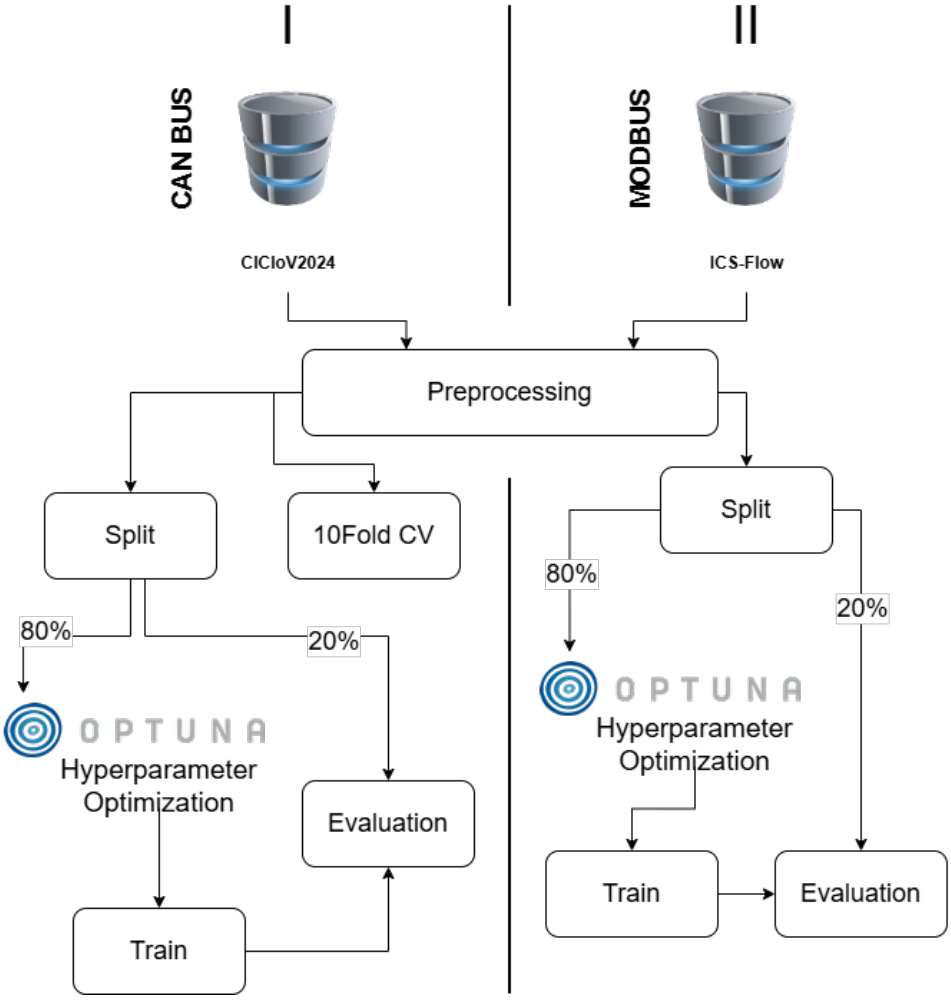


Figure 4.2: Phase I and II workflow structure.

new algorithms or additional industrial protocols primarily involves registering an optimizer and trainer pair and wiring a new featurizer into the protocol router rather than rebuilding the pipeline from scratch.

In order to facilitate the use of the toolkit, since it is restricted to the CICIoV2024 format for CAN, a simple application was implemented, the CAN-BUS Dataset Converter. For ICS-Flow, there is already a tool, the ICSFlowGenerator provided by [11] that allows you to capture offline PCAPs and export the data.

4.3.1 Legacy-IDS Evaluation

All trained models are validated in a streaming-oriented mode that mirrors the operational deployment of Legacy-IDS. To prove generalization beyond the training scenario, new hybrid benchmarks are synthesized and evaluated in two tracks. In the IoV track, attack CAN messages extracted from CICIoV2024 are replayed through our CAN simulator, which emits temporally coherent traffic bursts that mix benign background

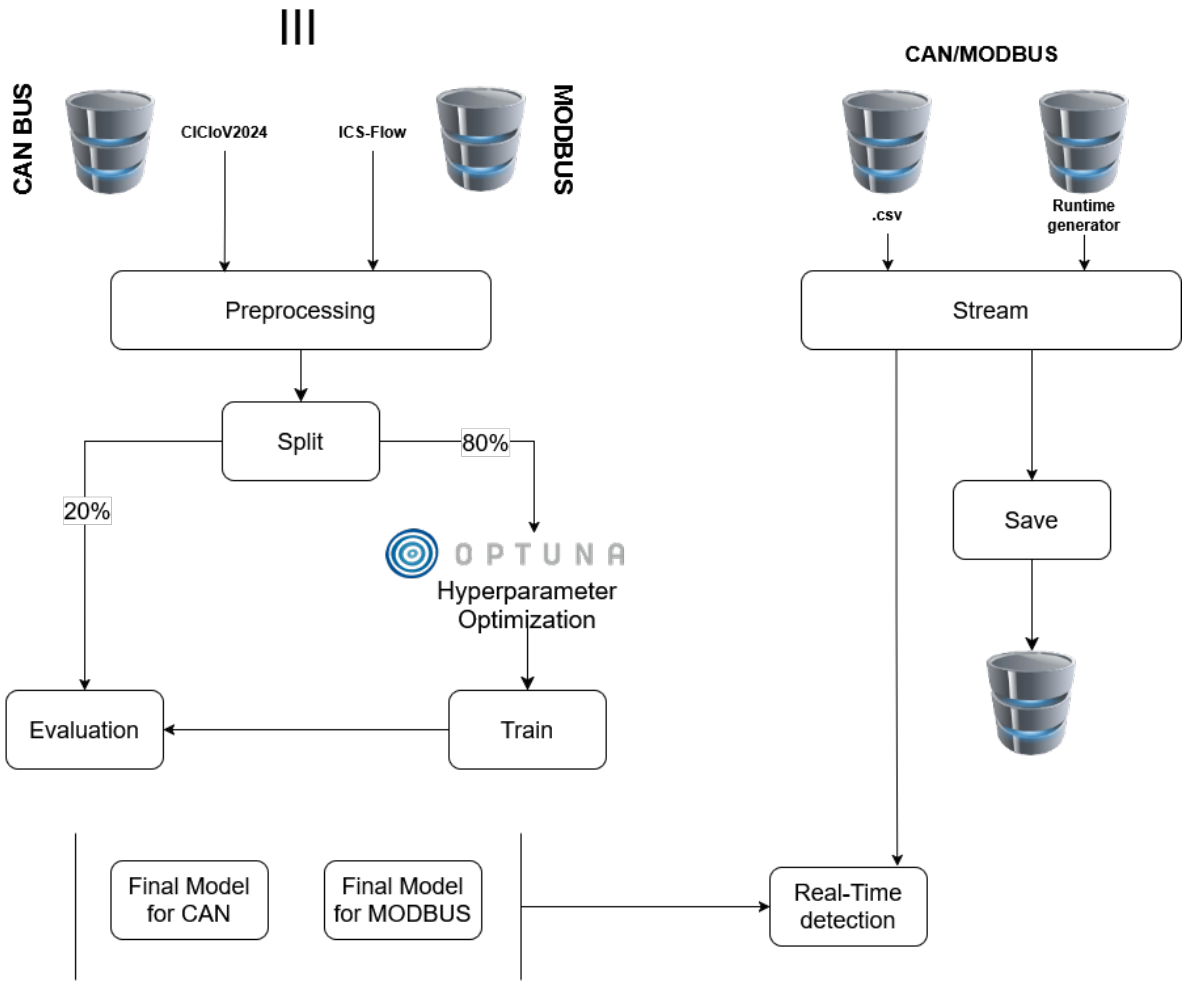


Figure 4.3: Phase III workflow structure

sequences with labeled attack injections. For the evaluation, 30% of the dataset is populated with attack messages and the remaining 70% with benign traffic. The aggregated stream is saved to a csv file and then loaded to the Legacy-IDS streaming engine and scored against all six CAN models, producing a cross-dataset comparison that highlights each detector's resilience to unseen attack payloads. In the industrial control track, a complementary dataset is generated with the MODBUS simulator. It fuses 50% of ICS-Flow attack traffic and 50% simulated perturbations that emulate command tampering, register flooding, and crafted replay scenarios, creating an attack sample of 30%, and the remaining 70% with normal data. This composite stream is evaluated against the six MODBUS models in the same streaming harness. Together, these validation steps ensure that the twelve archived models remain trustworthy when exposed to both their native datasets and heterogeneous simulator-driven traffic sourced from multiple attack repositories.

4.3.2 CAN-BUS Dataset Converter

The CAN-BUS Dataset Converter is a PyQt-based desktop application that standardizes CAN-BUS csv and log files so they can be analyzed with a consistent schema. The GUI guides users through loading a dataset, optionally running a preprocessing script, configuring column mappings, previewing the results, and exporting a converted CSV. The workflow is launched from a simple entry point that creates a QApplication, instantiates the main window, and starts the event loop, making the project easy to run. Inside the main window, the interface offers toggles for enforcing the built-in CAN schema, buttons for uploading datasets and preprocessing scripts, controls for selecting hexadecimal and binary conversion options, and a preview table showing up to the first 100 rows of the current DataFrame so users can validate their mappings before exporting, as shown in Figure 4.4. When a file is uploaded, the application detects missing headers, lets the user supply column names, and reconstructs the mapping UI accordingly. During export, the application builds a DataFrame that respects the user target-to-source selections, and it can convert hexadecimal or binary strings into decimal integers before writing the final csv.

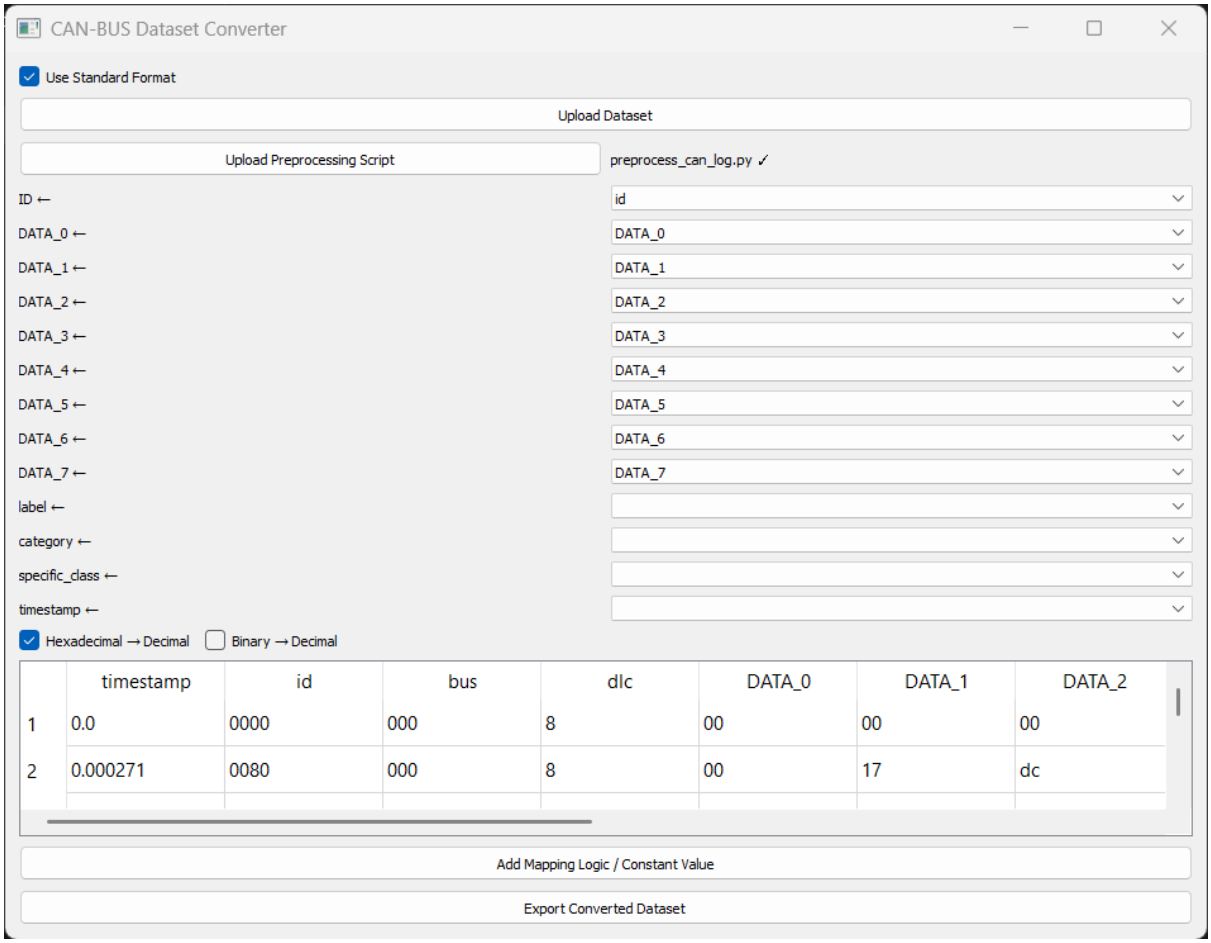


Figure 4.4: CAN-BUS Dataset Converter

An example preprocessing module script is made available with the code to demonstra-

te how to parse textual CAN logs into structured rows with timestamps, identifiers, bus values, DLC, and up to eight data bytes, illustrating how users can bridge non-tabular sources into the converters pipeline.

4.4 Preprocessing

The preprocessing steps are similar for all phases, such as data cleaning and data transformation. In the cleaning step, no changes were done to the CICIoV2024 dataset, since it was already clean and ready for ML applications. However, the ICS-Flow dataset had some "NaN" values, which were replaced with zero (0), maintaining the original dataset structure and consistency across features. The target variables specific_class (Can) and NST_M_Label (MODBUS) were encoded using label encoding, transforming categorical classes into numerical values to ensure compatibility with machine learning algorithms. To address the differences in feature size, standard scaling was applied for both datasets. Regarding feature selection, columns in ICS-Flow with source and destination addresses (IP and MAC), timestamps and offset-based features were removed, since they did not include important information for the learning process. For the CICIoV2024 dataset, all features were kept. All numeric variables were standardized to a mean of zero (0) and standard deviation of 1 so that features share a common scale. This is especially beneficial for scale-sensitive algorithms like Logistic Regression, while leaving the behavior of tree-based models essentially unchanged.

Finally, the original class ratios are preserved to keep the task faithful to operational conditions. This shared recipe produces clean, numerically stable inputs for both MODBUS and CAN, avoids dataset rebalancing and heavy feature engineering, and allows a fair comparison of models under a consistent, realistic preprocessing setup.

4.5 Experimental Setup

The experimental setup described in Table 4.3 supplied the compute and tooling needed for hyperparameter tuning, model evaluation, and the development of the base IDS, ensuring all computational tasks were carried out effectively. For Phases I and II, we took advantage of Jupiter Notebooks. However, for the final phase, a Python project was created.

Table 4.3: Details of the experimental setup - ASUS laptop.

Hardware	**Software**
Processor: Intel® Core™ i7-8750H	Windows 11, 64-bit
CPU @ 2.20 GHz, 2.21 GHz	Jupyter Lab 4.2.5
RAM: 16.00 GB, SSD: 1TB + 256 GB	Python 3.12.7
Graphics card: Nvidia® GeForce™ GTX 1050 Ti 4GB VRAM	Optuna 4.0.0

5 Results

In this chapter we report and discuss the experimental results across three phases: Phase I addressing CAN-based intrusion detection, Phase II focusing on MODBUS-based intrusion detection, and Phase III evaluating the unified cross-domain Legacy-IDS.

5.1 Phase I - Intrusion Detection on CAN Bus

In Phase I, the analysis of CICIoV2024 showed that several models are able to learn the structure of multi-class IoV traffic extremely well, even when the original class imbalance is preserved.

Table 5.1: Optimized hyperparameters for split and 10-fold training.

Model	Hyperparameter	Split	10-Fold
RF	n_estimators	112	57
	max_depth	25	8
	min_samples_split	2	6
AdaBoost	n_estimators	169	91
	learning_rate	0.22031826639728547	0.868649991657305
XGBoost	n_estimators	346	338
	max_depth	4	6
	learning_rate	0.08257351184613305	0.12486607179583395
	subsample	0.820421224719867	0.8398820455022838
	colsample_bytree	0.6442582375605579	0.8885000829639664
LR	C	0.43873275517235	3.2376253329526534
	penalty	l1	l1
	solver	saga	saga
ET	n_estimators	94	50
	max_depth	16	34
DNN	layers	4	4
	units	128	96
	dropout_rate	0.2774088047541644	0.24552040101739658
	learning_rate	0.0009501230980605523	0.00010682901273334459
	batch_size	112	128
	epochs	45	34

We tuned each model with Optuna across 10 trials, where each trial tested a different

hyperparameter configuration. The selected hyperparameters for all models are reported in Table 5.1. In total, this setup required 50 train/validation runs per model, balancing computation time with a thorough evaluation.

Table 5.2: Comparison of execution time.

Models	**Train/Test Split (s)** [1]		**10-Fold CV (s)** [1]	
	Optuna	**Training**	**Optuna**	**Training**
Random Forest	1830	120	1971	238
AdaBoost	5620	191	5935	496
XGBoost	1820	106	3512	1532
Logistic Regression	1745	1116	3997	3647
Extra Trees	1819	47	3271	221
Deep Neural Networks	11,010	1022	40,387	5104

[1] All times have been rounded to the nearest integer for simplicity. Time in seconds.

As outlined in Chapter 4, model performance was assessed using two schemes: a standard train–test split and 10-fold cross-validation. The evaluation targets the specific difficulties of imbalanced, multi-class intrusion detection. We report precision, recall, accuracy, and F1-score as the principal metrics. Table 5.2 summarizes, in seconds, the time spent on hyperparameter tuning and training for each model under both evaluation procedures.

After hyperparameter optimization, Random Forest, Deep Neural Networks, XGBoost and Extra Trees achieved perfect or near-perfect performance in the split setting. Table 5.3 reports the outcomes of the train–test split procedure and compares the split-based results with those reported by Neto et al. [7] and Gul and Bakir [108]. The strongest results are in bold. Each model was trained on 80% of the data using Optuna-selected hyperparameters and evaluated on the remaining 20%. Logistic Regression achieved 97% accuracy and a 0.97 F1-score, indicating solid generalization across classes. AdaBoost reached 98% accuracy but exhibited lower recall (81%) and precision (80%), likely reflecting the dataset's class imbalance. Random Forest, Extra Trees, XGBoost, and the DNN all attained perfect scores (1.0) on every metric, clearly surpassing the values reported in [7, 108], where the best F1-scores for DNN and RF were 0.78 and 0.76 in [7] and 0.9679 and 0.9745 in [108]. These results highlight their effectiveness in detecting both majority and minority classes.

Table 5.3: Comparison of model performance across different studies using split approach on CICIoV2024 dataset.

Models	Neto et al. [7]				Gul and Bakir [108]				Our Study			
	Acc.	Rec.	Prec.	F1	Acc.	Rec.	Prec.	F1	Acc.	Rec.	Prec.	F1
LR	0.89	0.50	0.48	0.49	0.90	0.60	0.72	0.62	0.97	0.86	0.98	0.88
AdaBoost	0.92	0.66	0.48	0.51	0.97	0.73	0.75	0.72	0.98	0.82	0.80	0.81
RF	0.96	0.76	0.76	**0.76**	1.00	0.97	0.99	**0.97**	1.00	1.00	1.00	**1.00**
DNN	0.96	0.76	0.83	**0.78**	0.99	0.97	0.97	**0.97**	1.00	1.00	1.00	**1.00**
XGBoost	–	–	–	–	–	–	–	–	1.00	1.00	1.00	**1.00**
ET	–	–	–	–	–	–	–	–	1.00	1.00	1.00	**1.00**

When the training strategy changed to 10-fold cross-validation, in Table 5.4, AdaBoost degraded noticeably and became weaker than in both the original works [7] and our own split configuration, with an F1-score of 0.32 while still posting 92% accuracy. Nonetheless, RF, DNN, ET, XGBoost and LR retained outstanding performance. Random Forest, Extra Trees, XGBoost and the deep neural network again reached perfect scores across the metrics considered(precision, recall, accuracy, and F1-score), and even Logistic Regression, which did not achieve perfection, reflecting some sensitivity to class imbalance despite solid overall performance with an accuracy of 97% and an F1-Score of 0.88, confirming its suitability for this type of problem.

Table 5.4: Performance comparison of different models using 10-fold training on CICIoV2024 dataset.

Models	Metrics			
	Acc.	Rec.	Prec.	F1
Random Forest	1.00	1.00	1.00	1.00
AdaBoost	0.92	0.33	0.30	0.32
XGBoost	1.00	1.00	1.00	1.00
Logistic Regression	0.97	0.86	0.98	0.88
Extra Trees	1.00	1.00	1.00	1.00
Deep Neural Networks	1.00	1.00	1.00	1.00

The stability of RF, XGBoost, ET and DNN across the two evaluation strategies indicates that these methods are particularly well-suited to the multi-class CAN intrusion detection task, despite the pronounced skew between benign and attack classes. This robustness can be connected to two main aspects. First, hyperparameter optimization

with Optuna [27] allowed a careful adjustment of parameters such as tree depth, learning rate and number of estimators, so that each model operated close to its best configuration for CICIoV2024. Second, the structures of these algorithms are inherently favorable to imbalanced data: tree-based ensembles such as RF and ET aggregate many diverse decision trees and naturally capture complex boundaries between majority and minority classes, while gradient boosting in XGBoost refines the model iteratively to correct the most difficult cases. The deep neural network, in turn, takes advantage of its layered representation to learn highly expressive patterns, and, when regularized with mechanisms such as early stopping, can do so without collapsing into overfitting. This combination helps explain the perfect classification of all attack types obtained in Phase I.

However, Adaboost proved to be the most fragile in this setting, both in its ability to detect rare attacks and in terms of computational cost. While its overall accuracy remained reasonable, the recall values for minority classes dropped significantly, suggesting that many attacks were being missed. This behavior reflects the intrinsic sensitivity of AdaBoost to imbalanced data, by repeatedly increasing the weight of misclassified instances, the algorithm can end up focusing disproportionately on majority-class examples when those dominate the dataset, which reduces its capacity to correct errors on infrequent classes. This phenomenon has been described in the literature as a limitation of AdaBoost under imbalance [159], and the Phase I results reinforce that view. Logistic Regression occupied an intermediate position. It did not reach the perfect scores of the ensembles and DNN and required non-trivial computational effort when fully tuned, but it systematically outperformed AdaBoost and remained competitive, in line with previous indications that logistic models can behave well under imbalance when properly calibrated [160]. Its linear decision boundaries cannot express the same level of non-linearity as tree-based and deep models, yet its high recall and solid F1-scores show that it can still be a practical option, especially in scenarios where interpretability remains relevant.

A critical aspect of Phase I is that the original class distribution of CICIoV2024 [7] was preserved, rather than artificially balanced, aligning the evaluation with realistic IoV deployments in which benign traffic clearly outnumbers malicious events. Under these conditions, Random Forest, XGBoost, Extra Trees and DNN were able to mitigate the impact of imbalance mainly through their structure and careful tuning, whereas AdaBoost and Logistic Regression were less successful in fully compensating for the skew, underscoring that in real-world scenarios model choice and hyperparameter optimization must explicitly take imbalance into account rather than assuming that a generic tuning procedure is sufficient. At the same time, the findings reveal a clear tension between computational cost and predictive quality: Random Forest, Extra Trees and XGBoost achieved flawless classification with relatively modest compute time, while

the DNN was substantially more time-consuming, especially during hyperparameter search and training; AdaBoost and Logistic Regression also showed limitations on the efficiency side, whereas Random Forest and Extra Trees combined top-tier accuracy with strong efficiency.

5.2 Phase II - Intrusion Detection on MODBUS

Phase II extends the analysis to the ICS domain, using the ICS-Flow dataset and moving from IoV CAN frames to MODBUS flows. Here, the problem becomes more challenging, particularly in the multi-class configuration that separates Normal, DDoS, IP-Scan, MitM, Port-Scan and Replay. The hyperparameters were tuned with Optuna over 10 trials, each exploring a different configuration. This setup evaluates performance across multiple train/validation splits, helping curb overfitting and providing a more trustworthy estimate of generalization on an imbalanced dataset. The best resulting settings for each model are reported in Table 5.5.

Table 5.5: Optimized hyperparameters.

Model	Hyperparameter	Value
RF	n_estimators	300
	max_depth	13
	min_samples_split	2
	min_samples_leaf	4
	max_features	None
	bootstrap	True
AdaBoost	n_estimators	455
	learning_rate	0.739409912906198
XGBoost	n_estimators	168
	max_depth	7
	learning_rate	0.07941533166492101
	subsample	0.8008294157501825
	colsample_bytree	0.9331479180202632
LR	C	2.172251391585297
	penalty	l1
	solver	saga
ET	n_estimators	100
	max_depth	19

(*continues on next page*)

(*continued from previous page*)

Model	Hyperparameter	Value
	min_samples_split	6
	min_samples_leaf	1
	max_features	sqrt
	bootstrap	False
DNN	layers	4
	units	256
	dropout_rate	0.35073545700562325
	learning_rate	0.000256475313888533
	batch_size	80
	epochs	42
ANN	units	48
	dropout_rate	0.42178712219684256
	learning_rate	0.0017966927951897516
	batch_size	112
	epochs	50

Table 5.6 reports the performance of seven classifiers Random Forest, AdaBoost, XGBoost, Logistic Regression, Extra Trees, a deep neural network, and a shallow artificial neural network, across six categories: Normal, DDoS, IP-Scan, MitM, Port-Scan, and Replay. For each model, Accuracy, Precision, Recall, and F1-score are presented by class, along with the overall macro-average F1. Hyperparameter optimization again led to marked gains at the global level, with macro-averaged F1-scores clearly higher than those obtained with untuned baselines as shown in Table 5.7. RF attains the best macro-average F1 of 0.89, with XGBoost and ET close behind at 0.88. The DNN reaches 0.87, the ANN 0.86, while LR trails slightly at 0.81. AdaBoost records the lowest macro-average F1 (0.71), suggesting greater difficulty with the dataset's imbalanced multi-class structure.

Across individual classes, most methods perform strongly on normal and DDoS traffic. For DDoS in particular, the F1-scores are near-perfect (often 1.00). However, the results also show that tuning alone does not resolve all the complexities of multi-class ICS traffic. Several studies [11, 8, 2] point to the role of explicit feature selection and engineering in further improving performance, by isolating the most informative indicators of malicious behavior. This perspective matches our own observations, with specific categories such as IP-Scan consistently remaining the hardest to correctly classify, where the F1-scores fall in the 0.48–0.52 range, visibly lower than the other attacks. The signatures of IP-Scan flows overlap considerably with benign traffic and other

scanning patterns, making them difficult to separate by parameter adjustment alone. Dehlaghi-Ghadim et al. [11] reached a similar conclusion, noting that similarities in how different attacks affect the underlying process can lead to systematic misclassifications in some flows. Even so, tree-based ensembles (RF, ET, XGBoost) show a modest edge over AdaBoost and the neural models in this class.

MitM, Port-Scan, and Replay are generally detected with high accuracy and F1-scores across methods. RF, XGBoost, and ET consistently excel on MitM and Replay, underscoring the strength of ensemble techniques for modeling complex ICS traffic patterns. While both neural approaches (DNN, ANN) are competitive on these classes, they do not surpass the top ensemble methods on the macro-average F1.

Table 5.6: Comparison of multi-class results for the selected ML methods.

Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
RF	Normal	0.98	1.00	1.00	1.00	0.89
	DDoS		1.00	0.99	1.00	
	IP-Scan		0.36	0.92	0.52	
	MitM		0.91	0.94	0.93	
	Port-Scan		0.98	0.90	0.94	
	Replay		0.98	0.89	0.94	
AdaBoost	Normal	0.92	0.93	0.99	0.96	0.71
	DDoS		1.00	0.42	0.60	
	IP-Scan		0.35	0.84	0.49	
	MitM		0.84	0.89	0.86	
	Port-Scan		0.93	0.90	0.91	
	Replay		0.84	0.30	0.44	
XGBoost	Normal	0.98	1.00	1.00	1.00	0.88
	DDoS		1.00	1.00	1.00	
	IP-Scan		0.34	0.84	0.48	
	MitM		0.91	0.94	0.92	
	Port-Scan		0.97	0.91	0.94	
	Replay		0.97	0.89	0.93	
LR	Normal	0.95	0.97	1.00	0.98	0.81
	DDoS		1.00	0.99	1.00	
	IP-Scan		0.34	0.84	0.49	
	MitM		0.89	0.83	0.86	
	Port-Scan		0.97	0.83	0.90	
	Replay		0.84	0.54	0.66	

(*continues on next page*)

(*continued from previous page*)

Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
ET	Normal	0.98	1.00	1.00	1.00	0.88
	DDoS		1.00	1.00	1.00	
	IP-Scan		0.36	0.92	0.52	
	MitM		0.92	0.93	0.92	
	Port-Scan		0.98	0.90	0.94	
	Replay		0.96	0.89	0.92	
DNN	Normal	0.98	0.99	1.00	0.99	0.87
	DDoS		1.00	1.00	1.00	
	IP-Scan		0.35	0.84	0.49	
	MitM		0.91	0.93	0.92	
	Port-Scan		0.93	0.87	0.90	
	Replay		1.00	0.82	0.90	
ANN	Normal	0.98	0.99	1.00	0.99	0.86
	DDoS		1.00	1.00	1.00	
	IP-Scan		0.35	0.84	0.49	
	MitM		0.90	0.91	0.91	
	Port-Scan		0.97	0.85	0.91	
	Replay		0.99	0.80	0.88	

Table 5.7: Per-class F1-score comparison between our results and the original study.

Model	Normal		DDoS		IP-Scan		MitM		Port-Scan		Replay	
	Ours	[11]	Ours	[11]	Ours	[11]	Ours	[11]	Ours	[11]	Ours	[11]
RF	1.00	0.9970	1.00	0.9995	0.52	0.5357	0.93	0.9075	0.94	0.9419	0.94	0.9257
AdaBoost	0.96	–	0.60	–	0.49	–	0.86	–	0.91	–	0.44	–
XGBoost	1.00	–	1.00	–	0.48	–	0.92	–	0.94	–	0.93	–
LR	0.98	–	1.00	–	0.49	–	0.86	–	0.90	–	0.66	–
ET	1.00	–	1.00	–	0.52	–	0.92	–	0.94	–	0.92	–
DT	–	0.9965	–	0.9995	–	0.5208	–	0.8953	–	0.9404	–	0.9240
ANN	0.99	0.9964	1.00	0.9980	0.49	0.4690	0.91	0.9149	0.91	0.9141	0.88	0.9347
DNN	0.99	–	1.00	–	0.49	–	0.92	–	0.90	–	0.90	–

– Models not tested.

A second point emerging from Phase II is the diversity in how algorithms respond to the pronounced imbalance present in ICS-Flow. Industrial traffic rarely follows uniform distributions of events, and the dataset reflects this reality through strong asymmetries

between classes. Tools such as Optuna are very effective at finding strong hyperparameter settings, but they do not, by themselves, resolve the structural difficulties posed by minority classes that correspond to stealthy or specialized attacks. The behavior of the models on IP-Scan and, to a lesser extent, on Replay makes clear that additional mechanisms are needed if one aims to raise recall on these categories without degrading precision for the majority ones. This is precisely why, in Phase II, the discussion emphasizes macro-averaged F1 rather than accuracy: by giving the same weight to each class, the macro-F1 offers a more honest view of performance under imbalance than a global accuracy dominated by the Normal class. Within this macro-F1 perspective, the results obtained here improve on those of the original study by Dehlaghi-Ghadim et al. [11] for Normal and DDoS flows. For these classes, we reach perfect or near-perfect F1-scores, comparable to those typically seen in binary classification work on ICS-Flow [8, 2], where the task is restricted to separating benign from attack traffic. At the same time, the remaining classes remain competitive, as summarized in Table 5.7, even though they are inherently more difficult due to their lower frequency and overlapping patterns. Overall, ensemble-based models lead multi-class detection, yet IP-Scan remains challenging for all, and class imbalance continues to impact minority classes despite automated hyperparameter tuning.

In summary, the MODBUS results indicate that ensemble-based methods, particularly Random Forest, Extra Trees, and XGBoost, are the most dependable default choice for multi-class IDS under class imbalance, combining near-ceiling detection of Normal and DDoS with consistently strong performance on MitM, Port-Scan and Replay, while IP-Scan remains challenging for all models and minority classes continue to reflect the effects of imbalance despite automated hyperparameter tuning. In this six-class ICS scenario, well-tuned tree-based ensembles thus deliver the best macro-averaged performance for the current feature set, with Normal and DDoS essentially solved, MitM, Port-Scan and Replay reliably detected, and IP-Scan standing out as the main source of misclassification, largely due to precision, pointing to the need for class-aware optimization and domain-informed feature engineering.

5.3 Phase III - Intrusion Detection on CAN and MODBUS

The final phase marks a change not only in evaluation but also in how the whole system is organized. The work moves from separate Jupyter notebook experiments to a unified cross-domain Legacy-IDS project. Before any streaming experiments are run, the best models from Phases I and II, ET, RF and XGBoost, are retrained inside this unified pipeline, for binary and multi-class classifications. This retraining step is essential for two reasons. First, it ensures that results for both domains are obtained under the same

preprocessing logic, evaluation procedures and metric calculations, making comparisons more consistent. Second, it brings the models closer to a deployable state, because all components—protocol detection, feature extraction, model inference and metric reporting—are integrated into one project instead of being scattered across exploratory notebooks.

5.3.1 Binary Classification

With the intention of expanding the research, we also added binary classification, using the three algorithm leaders from earlier phases (RF, ET, XGBoost) for CAN and MODBUS. The Optuna-selected settings are listed in Tables 5.8 and 5.10 and the corresponding results are reported in Tables 5.9 and 5.11.

CAN

For CAN, all three models achieved perfect performance, with accuracy, precision, recall and F1 scoring 1.00 for both Benign and Attack across RF, ET and XGBoost 5.9. This indicates that under the current feature set and data split, the learned decision surfaces separate normal from malicious traffic without ambiguity. These results reinforce the pattern already observed in Phase I, where the Normal and Attack boundaries become almost trivial once the hyperparameters are tuned with Optuna.

Table 5.8: Optimized hyperparameters for CAN (Binary).

Model	Hyperparameter	Value
RF	n_estimators	194
	max_depth	27
	min_samples_split	3
	min_samples_leaf	2
	max_features	0.301631259882531
	bootstrap	False
ET	n_estimators	335
	max_depth	20
	min_samples_split	2
	min_samples_leaf	1
	max_features	0.4323581943609807
XGBoost	n_estimators	222
	max_depth	4
	learning_rate	0.1629758639898246

(*continues on next page*)

(*continued from previous page*)

Model	Hyperparameter	Value
	subsample	0.9772085305809728
	colsample_bytree	0.6703185876737905
	reg_lambda	0.028447678074086786
	reg_alpha	0.007865550026404815
	min_child_weight	6.527225734975923

The consistency across methods further indicates that the CAN task is easily separable with low model complexity, since XGBoost achieved perfect results with a shallow depth as presented in 5.8. The results corroborate with the literature, where binary IDS models on CICIoV2024 and other datasets, such as OTIDS, Car-Hacking and so on, often report perfect or near perfect metrics when trained and tested on similar distributions [7, 149, 116].

Table 5.9: Comparison of binary results for the selected ML methods.

Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
RF	Benign	1.00	1.00	1.00	1.00	1.00
	Attack		1.00	1.00	1.00	
ET	Benign	1.00	1.00	1.00	1.00	1.00
	Attack		1.00	1.00	1.00	
XGBoost	Benign	1.00	1.00	1.00	1.00	1.00
	Attack		1.00	1.00	1.00	

MODBUS

Regarding MODBUS, performance remains near-perfect but no longer identical across models. RF and ET obtain the highest overall accuracy, with 0.997266, and essentially tie on macro-F1, 0.995687 vs. 0.995685. Nonetheless, XGBoost is not far behind, achieving an accuracy of 0.997157 and a macro-F1 of 0.995517, as shown in Table 5.11.

Table 5.10: Optimized hyperparameters for MODBUS (Binary).

Model	Hyperparameter	Value
RF	n_estimators	130
	max_depth	18

(*continues on next page*)

(*continued from previous page*)

Model	Hyperparameter	Value
	min_samples_split	2
	min_samples_leaf	3
	max_features	0.5056292919597651
	bootstrap	False
ET	n_estimators	251
	max_depth	22
	min_samples_split	7
	min_samples_leaf	1
	max_features	0.5724472788026354
XGBoost	n_estimators	288
	max_depth	10
	learning_rate	0.21768566395155792
	subsample	0.9889683232468998
	colsample_bytree	0.6490522704140281
	reg_lambda	0.002424162560409023
	reg_alpha	0.02903076156028365
	min_child_weight	0.7348522218999524

The per-class breakdown highlights small, compensating trade-offs. XGBoost has the highest Attack recall with 0.9956 but slightly lower Attack precision, 0.9901. RF attains the best Benign precision, 0.9988, while ET has the highest Benign recall, and both models balance precision/recall well for both classes.

Table 5.11: Comparison of binary results for the selected ML methods.

Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
RF	Benign	0.997266	0.9988	0.9978	0.9983	0.995687
	Attack		0.9912	0.9950	0.9931	
ET	Benign	0.997266	0.9986	0.9980	0.9983	0.995685
	Attack		0.9917	0.9945	0.9931	
XGBoost	Benign	0.997157	0.9989	0.9975	0.9982	0.995517
	Attack		0.9901	0.9956	0.9928	

Although performance remains very high, the small differences in macro-F1 and the minor asymmetries between Benign and Attack classes already anticipate the difficul-

ties that emerge more clearly in the multi-class setting. Compared with studies that use ICS-Flow purely as a binary benchmark and also obtain near-perfect scores [8, 2], the results here show that carefully tuned RF, ET and XGBoost are able to match or exceed those baselines while being trained inside a more general-purpose pipeline that must also support multi-class predictions and a second protocol. At the same time, the fact that MODBUS binary performance is marginally below the CAN case under the same optimization procedure hints that ICS-Flow benign/attack separation is less pronounced than in CICIoV2024. This aligns with reports that industrial datasets often contain more subtle overlaps between legitimate and malicious flows, especially when multiple attack families coexist on similar topologies [11].

5.3.2 Multi-Class Classification

CAN

For CAN multi-class detection, the three shortlisted models deliver essentially perfect performance. RF and ET achieved 1.00 across accuracy, precision, recall and F1 in every class, as shown in Table 5.13, which was expected from the previous phase results listed in Table 5.3, since the pipeline did not suffer changes.

Table 5.12: Optimized hyperparameters for CAN (Multi).

Model	Hyperparameter	Value
RF	n_estimators	195
	max_depth	12
	min_samples_split	3
	min_samples_leaf	1
	max_features	0.6361227004708269
	bootstrap	True
ET	n_estimators	244
	max_depth	48
	min_samples_split	2
	min_samples_leaf	2
	max_features	0.6895260963860841
XGBoost	n_estimators	123
	max_depth	11
	learning_rate	0.10010690163123717
	subsample	0.6138203293838544
	colsample_bytree	0.7174269669525961

(*continues on next page*)

(*continued from previous page*)

Model	Hyperparameter	Value
	reg_lambda	6.762295425580662
	reg_alpha	0.9027298786465361
	min_child_weight	4.6172116398240535

Although XGBoost did not scored perfect results, it did achieve a near-perfect performance with an overall accuracy of 0.999989 and a macro-F1 of 0.999950, showing only minimal deviations on RPM and Steering Wheel, easily seen in confusion matrix in Figure 5.1.

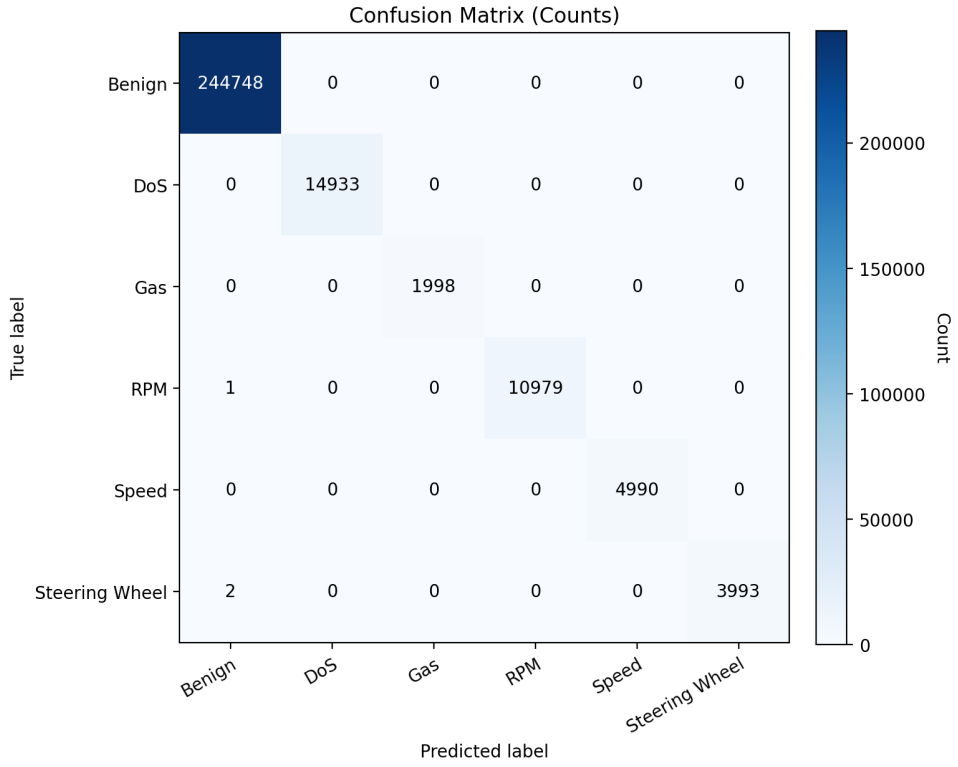


Figure 5.1: Confusion matrix for XGBoost with absolute values (CAN)

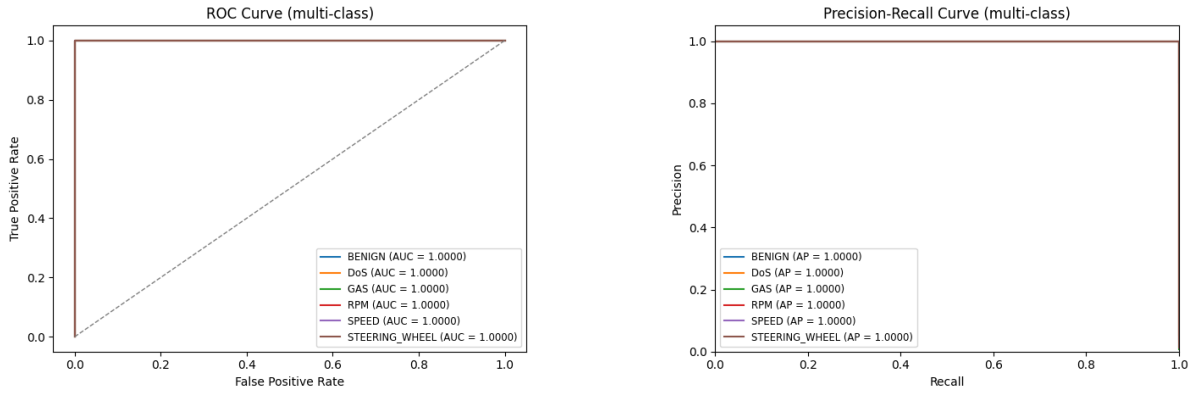
Table 5.13: Comparison of multi-class results for the selected ML methods.

Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
RF	Benign	1.00	1.00	1.00	1.00	1.00
	DoS		1.00	1.00	1.00	
	Gas		1.00	1.00	1.00	
	RPM		1.00	1.00	1.00	

(*continues on next page*)

(*continued from previous page*)

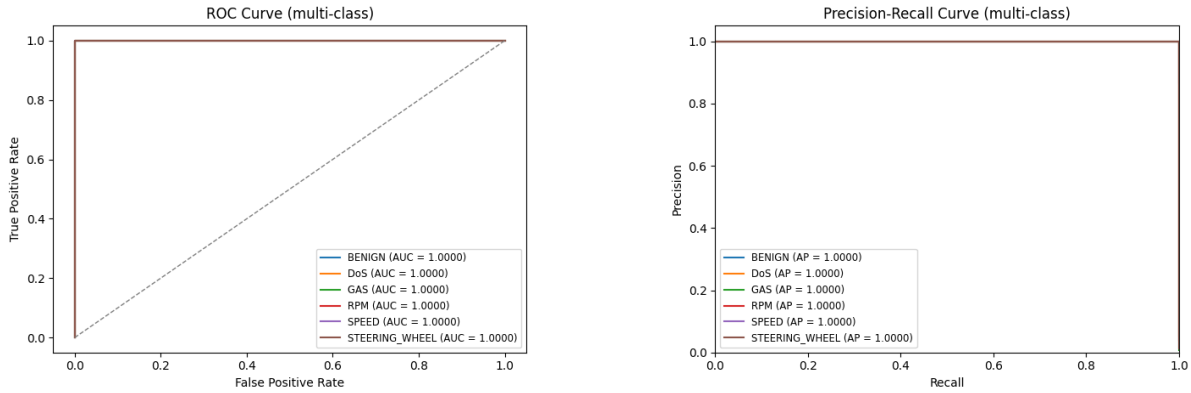
Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
	Speed		1.00	1.00	1.00	
	Steering Wheel		1.00	1.00	1.00	
ET	Benign	1.00	1.00	1.00	1.00	1.00
	DoS		1.00	1.00	1.00	
	Gas		1.00	1.00	1.00	
	RPM		1.00	1.00	1.00	
	Speed		1.00	1.00	1.00	
	Steering Wheel		1.00	1.00	1.00	
XGBoost	Benign	0.999989	1.00	1.00	1.00	0.999950
	DoS		1.00	1.00	1.00	
	Gas		1.00	1.00	1.00	
	RPM		1.00	0.9999	1.00	
	Speed		1.00	1.00	1.00	
	Steering Wheel		1.00	0.9995	0.9997	



(a) ROC Curve (b) PR Curve

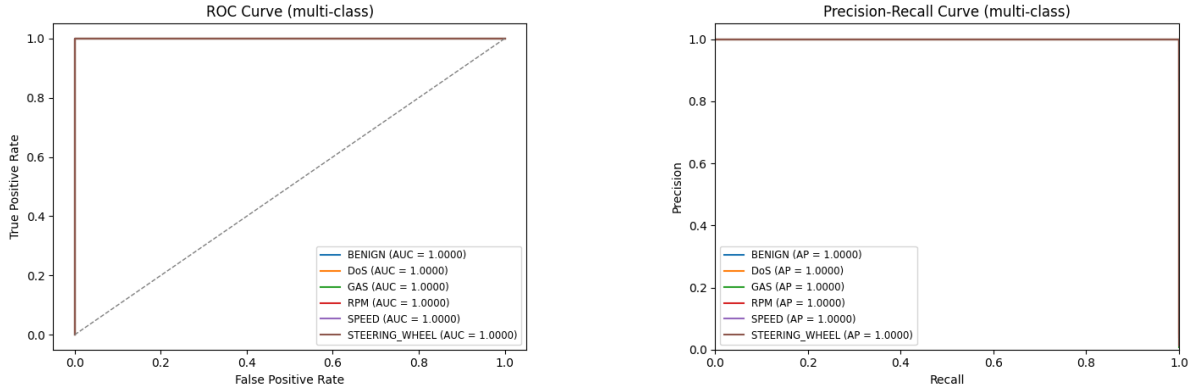
Figure 5.2: ROC/PR Curves for Extra-Trees (CAN, Multi)

The ROC plots saturate at the upper envelope for all classes and all models, as seen in Figures 5.2a 5.3a 5.4a, with AUC = 1.0000, meaning the true-positive rate remains 1 while the false-positive rate is effectively 0 across the scoring range. The PR curves mirror this, AP = 1.0000 for every class indicates precision stays at 1 for the entire recall sweep, meaning that no false positives are introduced as recall increases to 1.



(a) ROC Curve (b) PR Curve

Figure 5.3: ROC/PR Curves for Random Forest (CAN, Multi)



(a) ROC Curve (b) PR Curve

Figure 5.4: ROC/PR Curves for XGBoost (CAN, Multi)

In conclusion, the curves confirm what the tables already show, that class boundaries in the CAN feature space are cleanly separated, with wide margins, and this holds consistently for RF, ET, and XGBoost. The achieved results for CAN multi-class confirm that under the present feature set and splitting strategy, the CICIoV2024 dataset is almost entirely linearly separable for the chosen ensembles. RF, ET and XGBoost not only achieve 1.00 across accuracy, precision, recall and F1 for every class, but do so with ROC and PR curves that saturate at the upper envelope. This confirms the findings from previous CAN-bus IDS studies in the literature, where tree-based ensembles, MLPs and hybrid deep models routinely report near-perfect scores on CAN datasets [119, 108, 146, 147].

MODBUS

The three tested algorithms delivered a high overall accuracy of above 98% with macro-F1 scores between 0.885 and 0.894, indicating consistent detection across the six MODBUS traffic categories (Normal, DDoS, IP-scan, MITM, Port-scan, Replay). Extra Trees

achieved the best aggregate metrics, closely followed by Random Forest and XGBoost. The tight spread suggests that preprocessing and feature selection are well aligned with the detection task, and each ensemble captures similar decision boundaries.

Table 5.14: Optimized hyperparameters for CAN (Multi).

Model	Hyperparameter	Value
RF	n_estimators	159
	max_depth	18
	min_samples_split	10
	min_samples_leaf	4
	max_features	0.4662213472738182
	bootstrap	True
ET	n_estimators	264
	max_depth	45
	min_samples_split	3
	min_samples_leaf	4
	max_features	0.5080011164104428
XGBoost	n_estimators	144
	max_depth	12
	learning_rate	0.06979075453168315
	subsample	0.947107373085814
	colsample_bytree	0.9040571983290013
	reg_lambda	5.500019724397243
	reg_alpha	1.7677522560077903
	min_child_weight	0.17658781489563607

Table 5.15: Comparison of multi-class results for the selected ML methods (MODBUS, Multi).

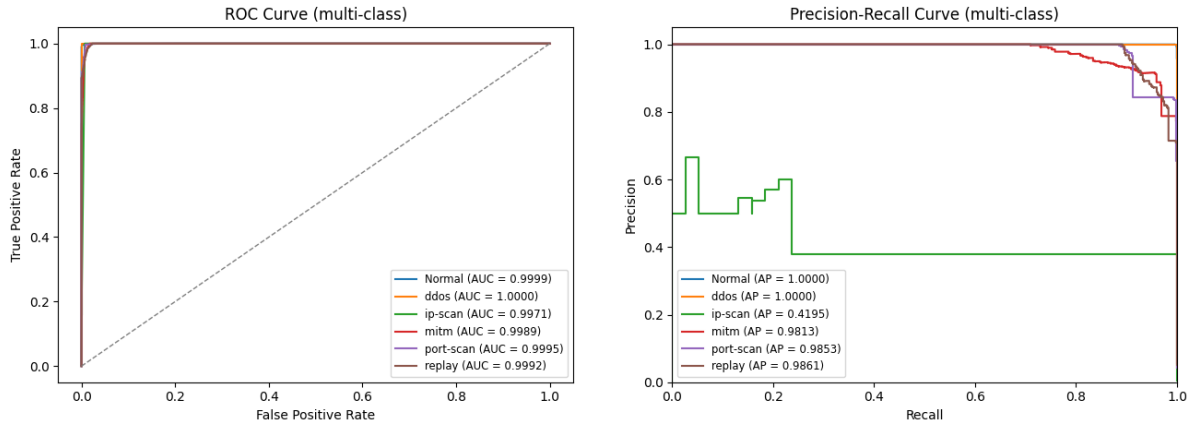
Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
RF	Normal	0.984471	0.9981	0.9978	0.9980	0.884711
	DDoS		1.00	0.9948	0.9974	
	IP-Scan		0.3608	0.9211	0.5185	
	MitM		0.9163	0.9323	0.9243	
	Port-Scan		0.9777	0.8997	0.9371	
	Replay		0.9681	0.9004	0.9330	

(*continues on next page*)

(*continued from previous page*)

Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
ET	Normal	0.986002	0.9985	0.9978	0.9982	0.893992
	DDoS		1.00	1.00	1.00	
	IP-Scan		0.38	1.00	0.5507	
	MitM		0.9164	0.9536	0.9346	
	Port-Scan		0.9777	0.9023	0.9385	
	Replay		0.9953	0.8941	0.9420	
XGBoost	Normal	0.984908	0.9988	0.9975	0.9982	0.885801
	DDoS		1.00	0.9974	0.9987	
	IP-Scan		0.3608	0.9211	0.5185	
	MitM		0.9120	0.9420	0.9267	
	Port-Scan		0.9725	0.9100	0.9402	
	Replay		0.9768	0.8919	0.9324	

Extra Trees delivered the top macro-F1 of 0.8940 and accuracy of 0.9860. It perfectly classified DDoS samples and fully captured the scarce IP-scan class with a recall of 1.0 and F1 of 0.5507, while keeping false positives low in other categories, highlighted in the confusion matrix in Figure 5.6. Replay traffic showed slightly lower recall, 0.8941, but the model compensated with very high precision, 0.9953. These metrics line up with the curve plots. The ROC plot in Figure 5.5a indicates an optimal separation of classes with AUC ranging from 0.9971 to 1.0. The PR curve in Figure 5.5b, however, has AP values from 0.4195 in IP-Scan up to 1.0 in Normal and DDoS traffic.



(a) ROC Curve

(b) PR Curve

Figure 5.5: ROC/PR Curves for Extra-Trees (MODBUS, Multi)

XGBoost balanced a strong accuracy of 0.9849 with a macro-F1 of 0.8858. Precision and recall were near perfect for Normal and DDoS traffic, being above 0.99, while IP-

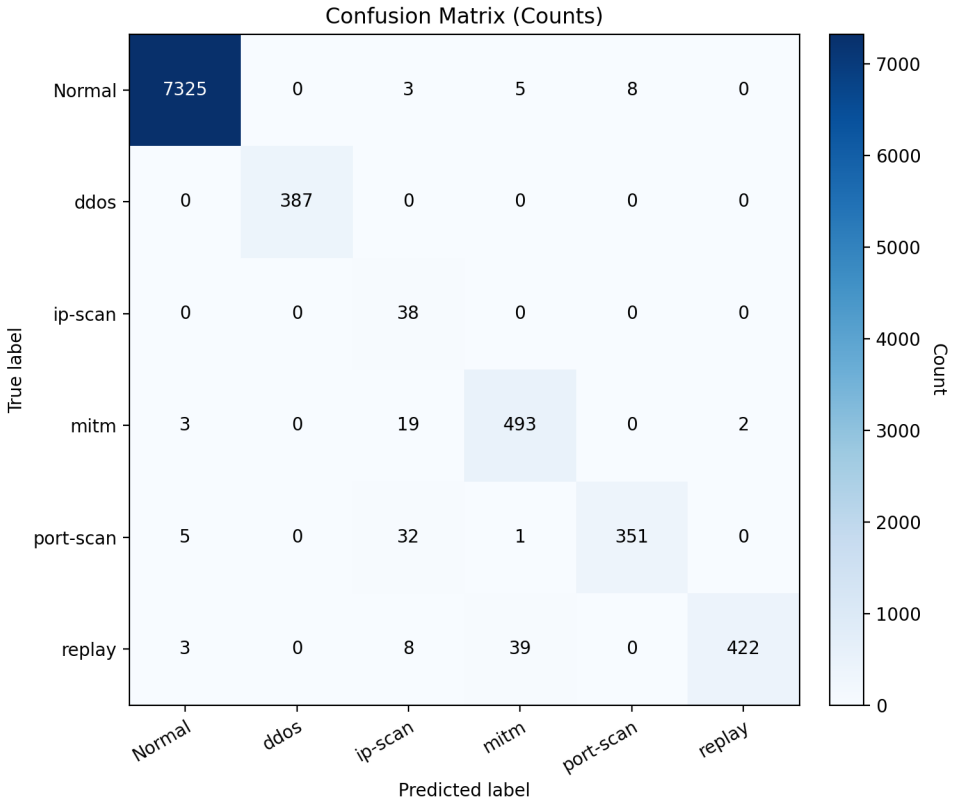
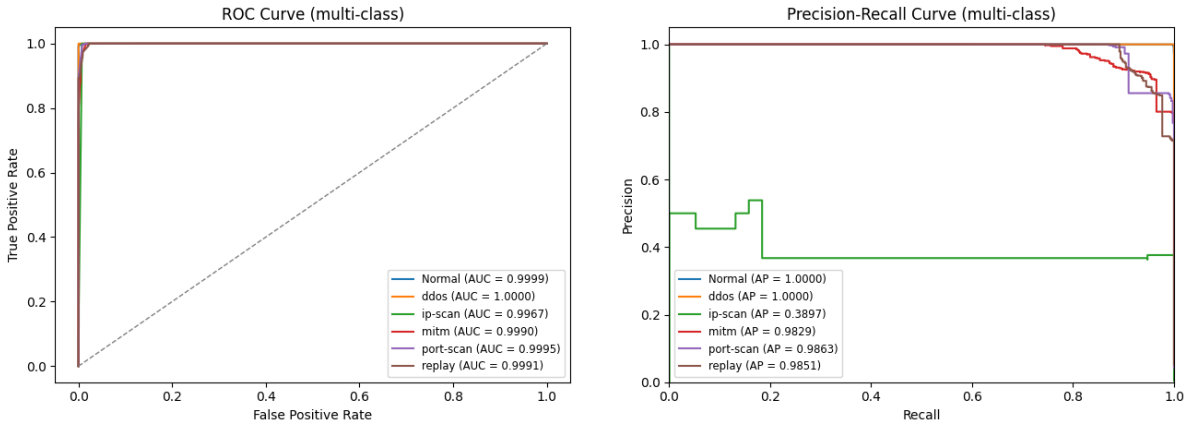


Figure 5.6: Confusion matrix for Extra Trees with absolute values (MODBUS).

scan remained the most challenging minority class with a precision of 0.3608, recall of 0.9211, and F1 of 0.5185). MITM and Port-scan showed high precision/recall pairs (0.91–0.97), and Replay traffic reached 0.9324 F1.



(a) ROC Curve (b) PR Curve

Figure 5.7: ROC/PR Curves for XGBoost multi (MODBUS, Multi)

Misclassifications concentrated on cross-confusions among minority attacks, particularly IP-scan being mistaken for MITM or Replay as presented in Figure 5.8. The ROC plot in Figure 5.7a has its curves pinned to the top, with AUC varying from 0.9967

to 1.0, meaning that discrimination is not the bottleneck. The PR plot in Figure 5.7b on the other hand, reveals the class-imbalance story. Lines for Normal and DDoS are completely horizontal, with an AP of 1.0, and MitM, Port-scan and Replay scores between 0.9829 and 0.9863. IP-scan again exhibits a low AP of 0.3897, consistent with few positive examples and overlap with other attack behaviors.

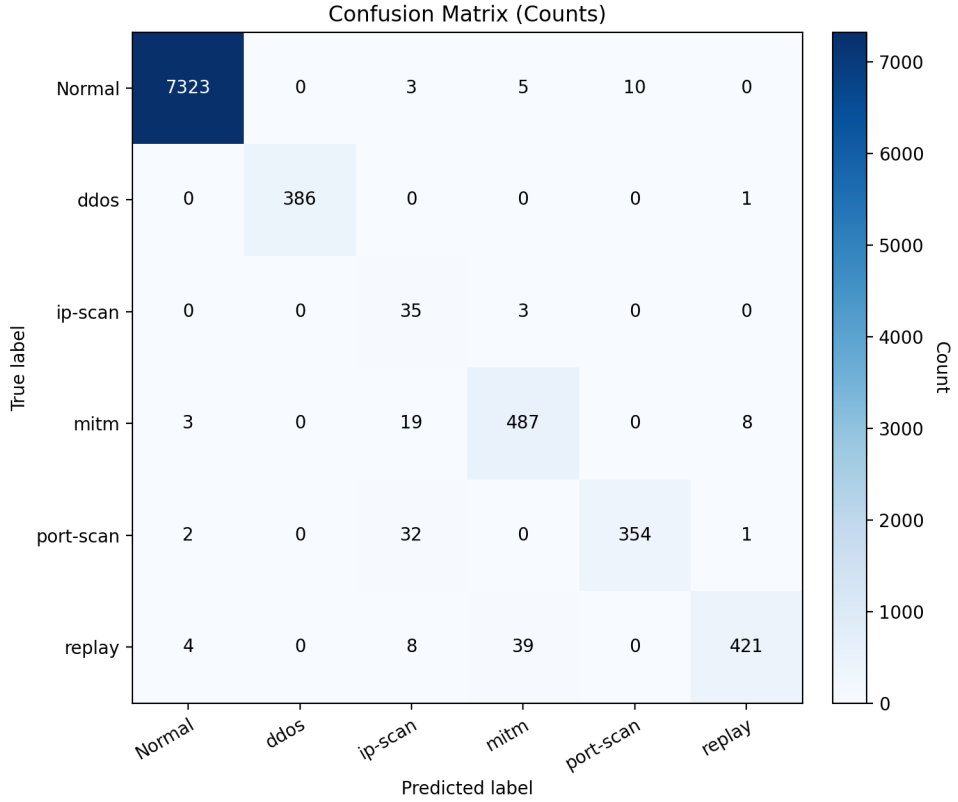
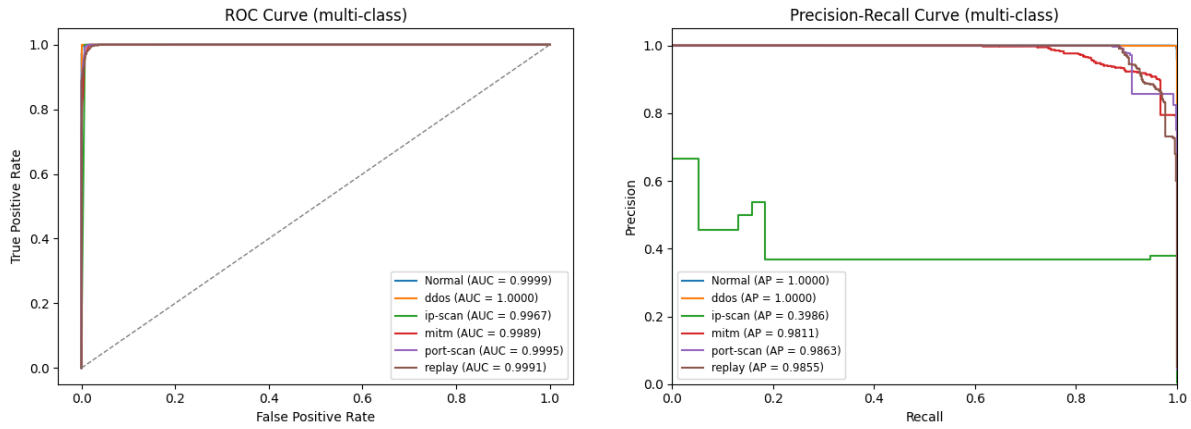


Figure 5.8: Confusion matrix for XGBoost with absolute values (MODBUS).

Random Forest mirrored XGBoost performance, with an accuracy of 0.9845 and macro-F1=0.8847. It preserved excellent detection for Normal and DDoS traffic, but, like XGBoost, struggled to raise IP-scan precision beyond 0.3608 despite high recall, 0.9211. Replay recall, 0.9004, slightly undercut Extra Trees, although the overall confusion remained low. These results corroborate the graphical validations, such as ROC and PR curves in Figures 5.9a and 5.9b. The ROC lines in Figure 5.9a reach the upper-left boundary for every class, with AUCs essentially ranging from 0.9967 to 1.0, which means the model makes a clean separation of positives and negatives, even at very low false-positive rates. The corresponding PR plot in Figure 5.9b is more revealing under imbalance, having Normal and DDoS with flat precisions of 1.0 across the whole recall range, while MitM, Port-Scan and Replay also stay high with AP ranging from 0.9811 to 0.9863. However, IP-Scan has a visible drop with AP at 0.3986. The models show comparable mistake profiles, with most errors coming from the smallest attack classes (IP-scan and Replay), often being swapped with MITM or the reverse. Extra Trees reaches the best macro-F1 largely because it excelled DDoS/IP-scan recall and edges out others

on precision for replayed traffic. Random Forest and XGBoost are close—within about a single F1 point but lean on different signals (boosting tends to elevate payload-sum and TTL cues, while bagging gives more weight to acknowledgment delay patterns). In short, ensembles fit MODBUS telemetry well, with Extra Trees delivering the most even performance on minority classes.



(a) ROC Curve (b) PR Curve

Figure 5.9: ROC/PR Curves for Random Forest (MODBUS, Multi)

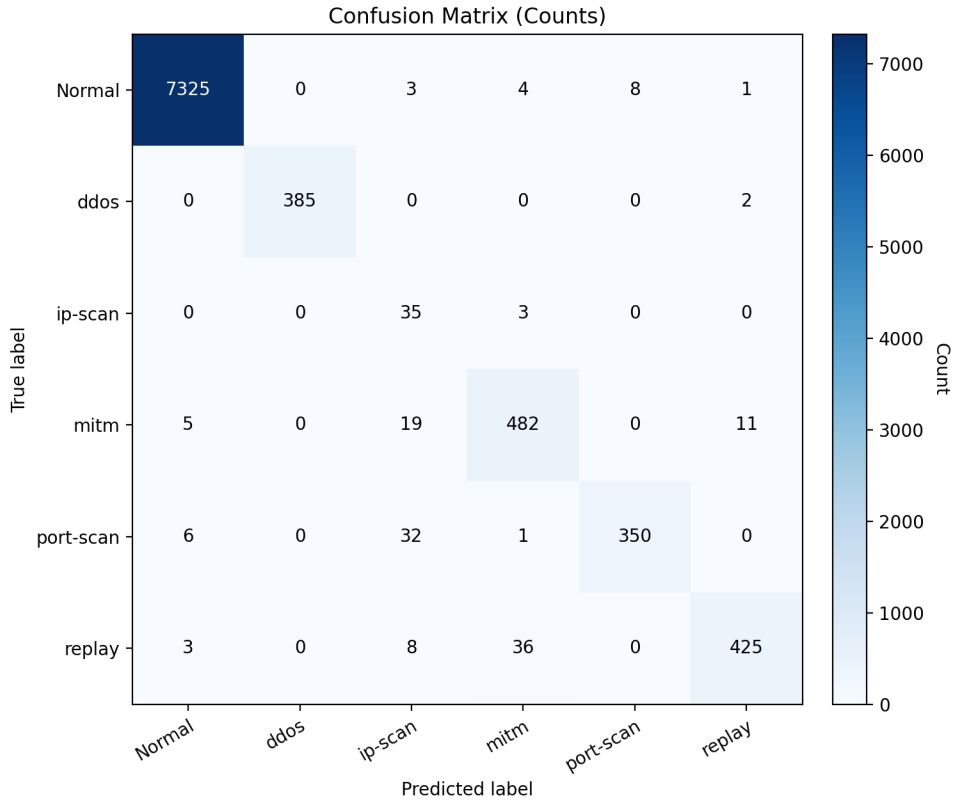


Figure 5.10: Confusion matrix for Random Forest with absolute values (MODBUS).

In contrast with the CAN results, the MODBUS multi-class experiments underline

how challenging ICS-Flow becomes once more than two classes are considered. Even though overall accuracies exceed 98% and macro-F1 scores approach 0.89, the per-class analysis shows that the minority attacks, especially IP-Scan, remain difficult to pin down. This echoes observations in the original ICS-Flow study and subsequent work [11], where Normal and DDoS flows are often close to solved while scan attacks suffer from overlapping signatures and severe imbalance. The behavior of IP-Scan is particularly illustrative. All three models attain very high recall on this class, yet precision remains low and the PR curves for IP-Scan sit well below those of the majority categories. In practice, this means that the classifiers learn to "catch" virtually all IP-Scan flows, but at the cost of misclassifying some MITM or Replay traffic. This pattern is consistent with reports that flow-level features in ICS-Flow do not fully disentangle different scan and replay behaviors when they share similar payload and timing characteristics [11].

The misclassifications traces back to three intertwined effects. First, the very limited IP-scan support (only 38 examples) induces imbalance that nudges models toward high recall, capturing nearly all IP-scan instances, at the cost of precision when a few MITM or Replay flows exhibit scan-like traits. Secondly, Replay and MITM occasionally become entangled because their payload characteristics are highly alike, and, when timing features align, the boundary between them blurs. Third, a small amount of Port-scan traffic is mislabeled as Normal or Replay, possibly due to overlaps in packet volume and TTL patterns.

5.4 Phase III - Legacy-IDS

After training the 12 models and achieving high results in the test sets, as highlighted in the previous sections, the models were saved and tested as core models for the IDS.

5.4.1 CAN

Initially, a CAN dataset was created with the developed simulator. The dataset is the result of a combination of 50% CICIoV2024 real attacks, with 50% of synthethic attacks, having a total number of attack records comprising 30% of the dataset. The benign traffic is itself all simulated. The description of the dataset is presented in Table 5.16. The dataset was then streamed and frame by frame, the chosen model made the predictions. Below are the results of the stream for each model.

Multi-Class

From the streaming evaluation on the generated dataset, the multiclass results in Table 5.17 show that the three ensemble models remain effective but no longer operate in the near-perfect scoring observed in the test sets from the previous section. Extra

Table 5.16: Details of the CAN generated dataset .

Classes	Total
Benign	158113
DoS	13862
Gas	1788
RPM	44066
Speed	4675
Steering Wheel	3617

trees achieves the highest overall accuracy (0.902548) with a macro-F1 of 0.775434, closely followed by XGBoost, which delivers the best macro-F1 (795090) despite a lower accuracy of 0.813503. Random Forest is slightly behind with 0.831882 accuracy and a macro-F1 of 680528. Across the three models, Benign, DoS and Gas remain the easiest classes to discriminate. Benign traffic achieves F1 scores ranging from 0.8858 to 9885, while DoS is detected with recalls close to 1.0 and Gas spoofing is classified almost perfectly (F1 of 1.00 for RF and ET, and 0.9878 for XGBoost). Speed attacks also achieve high values, between 0.6971 and 0.7816, whereas RPM and Steering Wheel are consistently the most demanding classes. For RPM, recall drops to 0.1878 with Random Forest and 0.0964 with XGBoost, but rises to 0.5027 under Extra Trees, which explains the higher macro-F1 of this model. Steering wheel, on the other hand, shows the opposite pattern with XGBoost classifying this class perfectly (precision, recall and F1-score of 1.00), while the bagging models reach very high recall but low precision (F1 ranges between 0.36477 and 0.3751), indicating a tendency to over-assign Steering Wheel labels when other attacks are present, as shown in the confusion matrices for Random Forest and Extra Trees in Figures 5.12 and 5.14.

Table 5.17: Comparison of multi-class results for the selected ML methods with a streamed dataset (CAN, Multi).

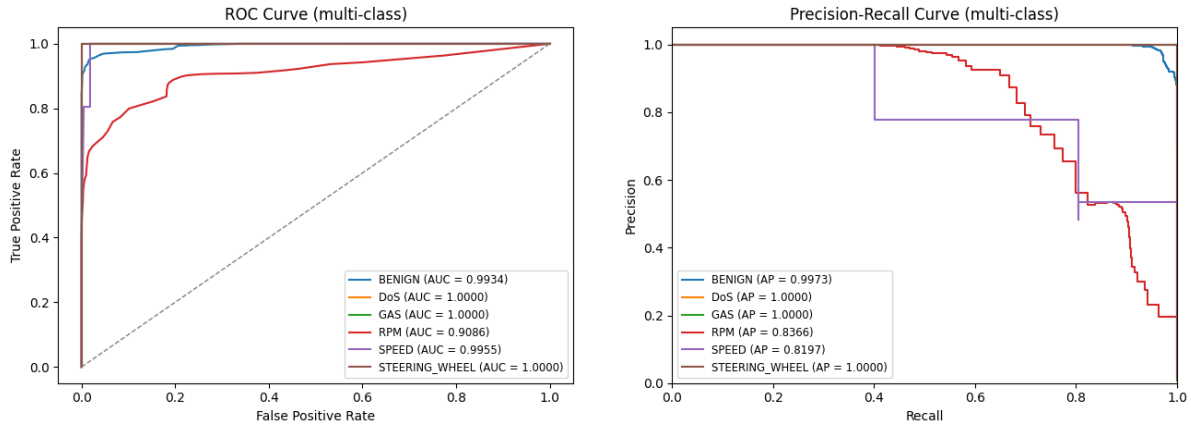
Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
RF	Benign	0.831882	0.9282	0.9859	0.9562	0.680528
	DoS		0.5986	1.00	0.7489	
	Gas		1.00	1.00	1.00	
	RPM		1.00	0.1878	0.3162	
	Speed		0.5350	1.00	0.6971	
	Steering Wheel		0.2230	1.00	0.3647	

(*continues on next page*)

(*continued from previous page*)

Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
ET	Benign	0.902548	0.9780	0.9992	0.9885	0.775434
	DoS		0.8498	1.00	0.9188	
	Gas		1.00	1.00	1.00	
	RPM		1.00	0.5027	0.6691	
	Speed		0.5397	1.00	0.7011	
	Steering Wheel		0.2309	1.0000	0.3751	
XGBoost	Benign	0.813503	0.7951	0.9997	0.8858	0.795090
	DoS		0.9834	0.8991	0.9394	
	Gas		0.9760	1.00	0.9878	
	RPM		1.00	0.0964	0.1759	
	Speed		0.7598	0.8047	0.7816	
	Steering Wheel		1.00	1.00	1.00	

The precision-recall curves support this reading in the different models. For Extra Trees, in Figure 5.13b, Benign, DoS, Gas and Steering Wheel all exhibit almost flat traces near precision 1.0, with AP values of 1.0, while RPM still achieves an AP of 0.9265 and Speed of 0.6657. Random Forest, in Figure 5.11b, and XGBoost, in Figure 5.15b show a similar picture, with AP values close to 1.0 for Benign, DoS,Gas and Steering Wheel, and slightly lower but still high AP for RPM, with values between 0.8366 and 0.9265, and Speed between 0.6657 and 0.8197. As recall increases, the RPM and Speed curves are the first to lose precision, which matches the confusion matrices, in Figures 5.12, 5.14 and 5.16 , where these classes are more often confused with other attacks.



(a) ROC Curve (b) PR Curve

Figure 5.11: ROC/PR Curves for Random Forest (CAN, Multi)

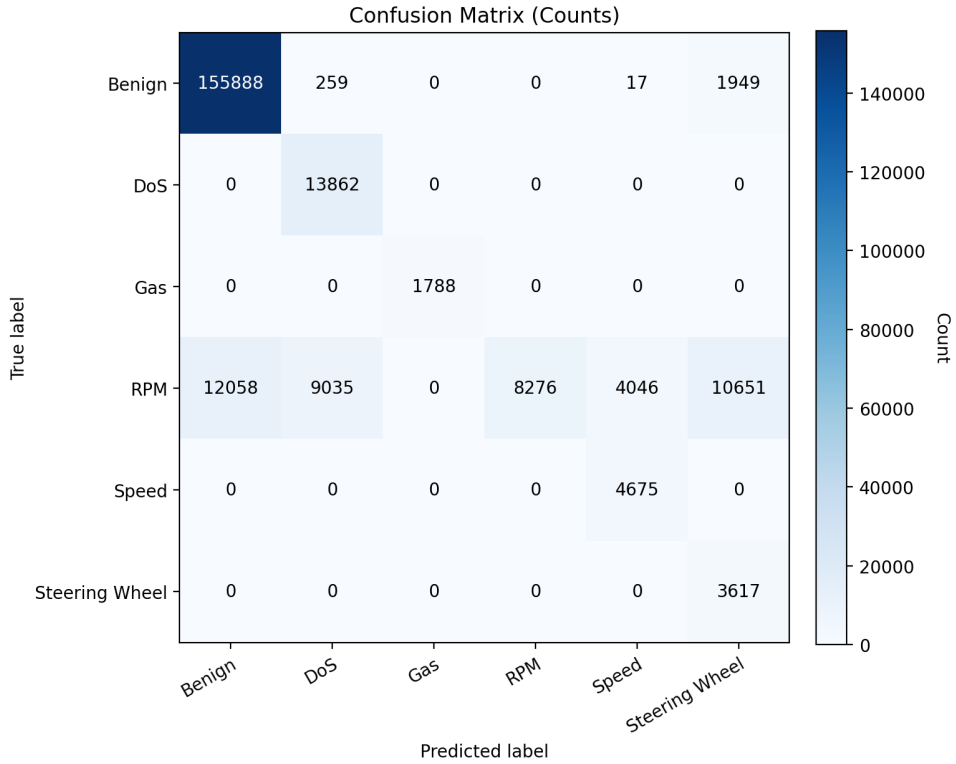
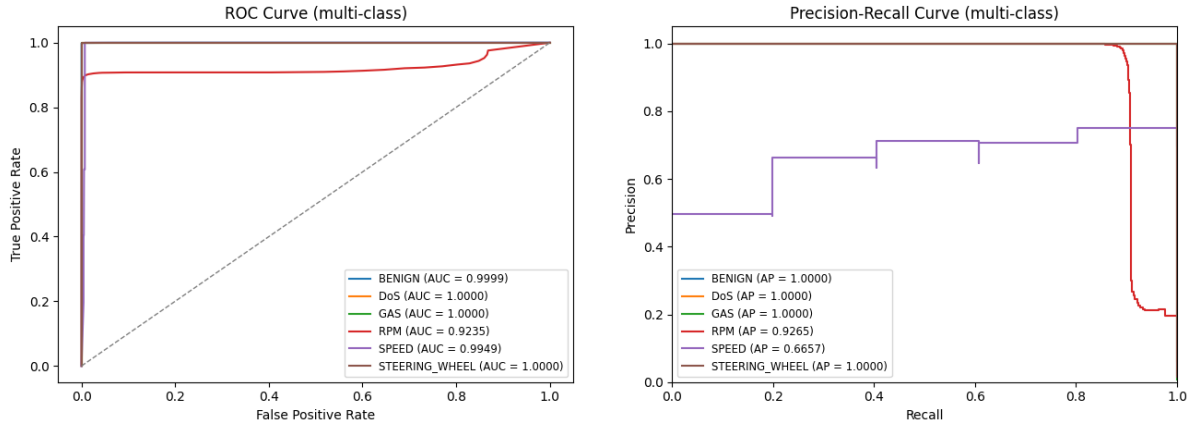


Figure 5.12: Confusion matrix for Random Forest with absolute values (Stream evaluation, CAN)



(a) ROC Curve (b) PR Curve

Figure 5.13: ROC/PR Curves for Extra Trees (CAN, Multi)

The ROC curves, in Figures 5.11a, 5.13a and 5.15a further indicate that for all three models, each class lies very close to the top, with AUC values between 0.9086 and 1.0 for RF, 0.9235 and 1.0 for ET, and 0.9409 to 1.0 with XGBoost. In the three models, Speed and RPM are the ones with the lowest AUC, but always above 0.9, meaning that the models still assign clearly different scores to each class.

In summary, the general drop from perfect or near perfect scores from the static test-set,

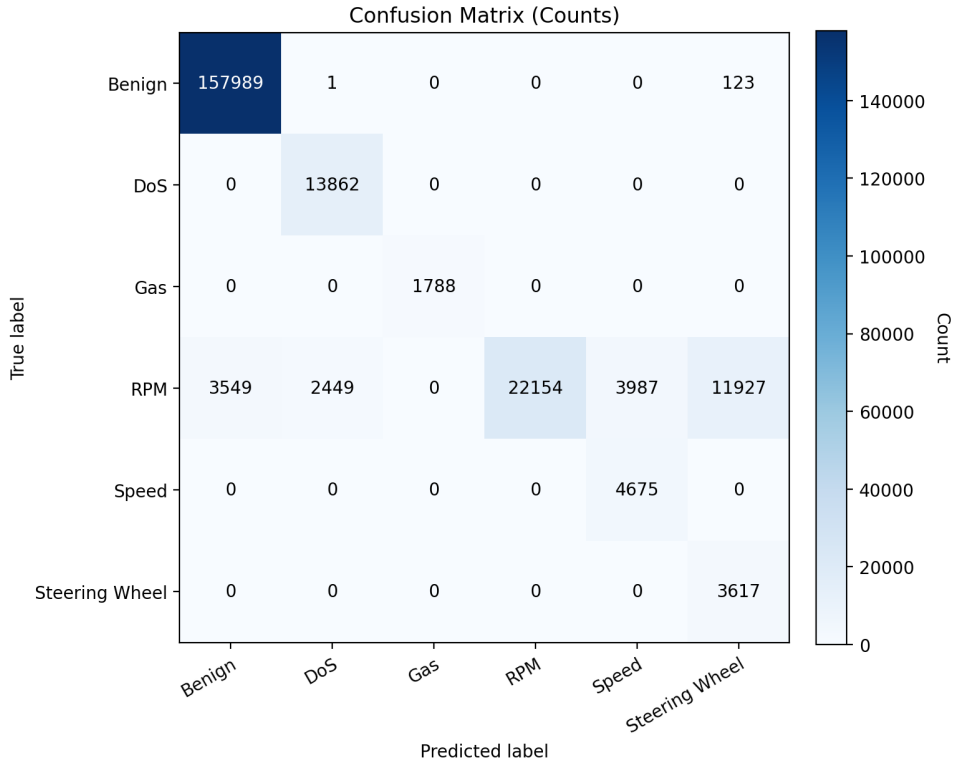
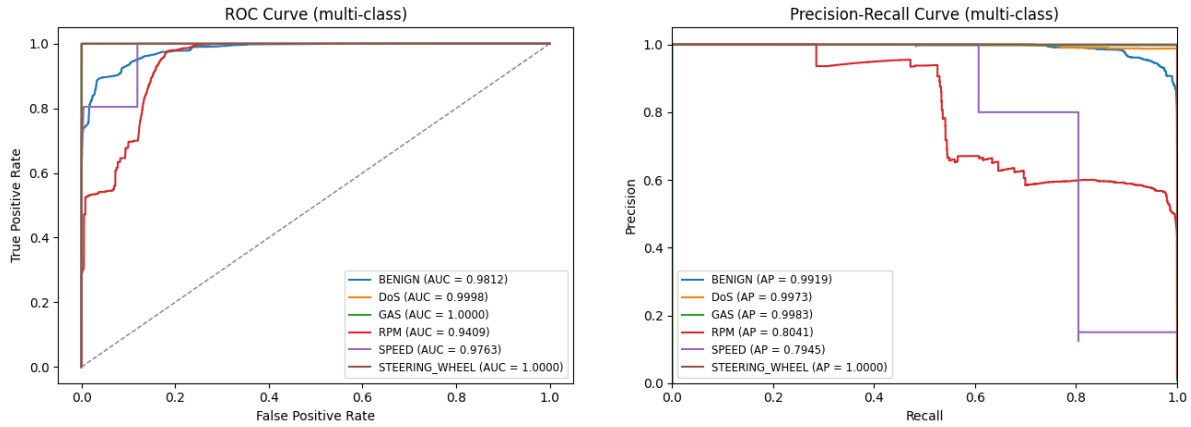


Figure 5.14: Confusion matrix for Extra Trees with absolute values (Stream evaluation, CAN)



(a) ROC Curve (b) PR Curve

Figure 5.15: ROC/PR Curves for XGBoost (CAN, Multi)

provides a more realistic picture of how Legacy-IDS behaves, reflecting the impact of mixing simulated benign traffic with a blend of real and synthetic attacks. Nonetheless, the degradation is selective, since classes such as Benign, DoS and GaS, remain extremely well detected while RPM and Speed are the first classes to lose precision and recall as the stream unfolds. This suggests that the decision surfaces learned on CICIoV2024 generalize reasonably well to synthetic perturbations that preserve the main structural patterns of attacks, but become more vulnerable for behaviors whose distinguishing cues are more sensitive to how the simulator generates the messages.

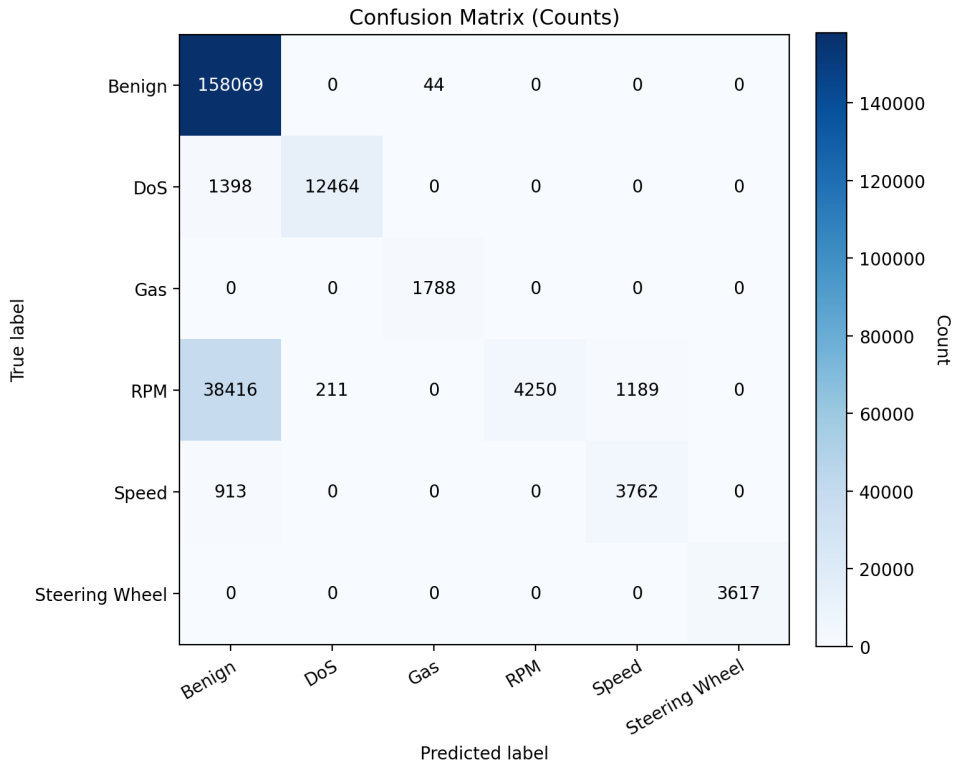


Figure 5.16: Confusion matrix for XGBoost with absolute values (Stream evaluation, CAN)

Binary

In the binary classification for CAN, the three ensemble models retain strong performance when exposed to the streamed and mixed generated dataset, described in Table 5.16. As shown in Table 5.18, Random Forest and Extra Trees remain close to their near-perfect behavior from the evaluations on the test set from the phase III training step, in Table 5.9, while XGBoost shows a small decrease.

Random Forest achieves and overall accuracy of 0.996979 and a macro-F1 of 0.996411. Both classes are treated symmetrically, with Benign reaching an F1-score of 0.9978 (precision 0.9982, recall 0.9975) and Attack obtaining an F1-score of 0.9950 (precision 0.9942, recall 0.9958). This balance indicates that, under streaming conditions, RF preserves a very low false-positive rate on benign frames without sacrificing the ability to flag malicious traffic. Regarding Extra Trees, the model stands out as the best binary detector on the CAN stream, obtaining an accuracy of 0.998974 and a macro-F1 of 0.998782, essentially matching the test set results. Benign traffic is almost perfectly preserved (precision 1.00, recall 0.9985, F1-score 0.9993), and the Attack class benefits from both high precision and perfect recall (0.9966 and 1.00, respectively), leading to an F1-score of 0.9983. In practice, this means that ET rarely mislabels benign frames as attacks while still recovering virtually every malicious frame.

XGBoost, with the lowest performance between the three models, still delivers strong results for a streaming scenario. It reaches an accuracy of 0.968773 and a macro-F1 of

0.962239. Benign traffic is detected with an F1-score of 0.9779 (precision 0.9660, recall 0.9902), while the Attack class attains an F1-score of 0.9465, with precision 0.9758 and recall 0.9190. Compared with RF and ET, XGBoost is slightly more conservative on the Attack class, allowing a small fraction of malicious frames to pass as benign, which explains the lower recall and macro-F1.

Table 5.18: Comparison of binary results for the selected ML methods with a streamed dataset (CAN, Binary).

Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
RF	Benign	0.996979	0.9982	0.9975	0.9978	0.996411
	Attack		0.9942	0.9958	0.9950	
ET	Benign	0.998974	1.00	0.9985	0.9993	0.998782
	Attack		0.9966	1.00	0.9983	
XGBoost	Benign	0.968773	0.9660	0.9902	0.9779	0.962239
	Attack		0.9758	0.9190	0.9465	

In conclusion, the binary streaming evaluation on CAN confirms that all three ensembles are suitable for real-time separation of benign and malicious traffic, with error rates that remain very low even under the mixed real/synthetic workload. Extra Trees provides the most favorable trade-off, combining almost perfect recall on attacks with minimal contamination of benign traffic. From a practical point of view, this shows that the features and decision boundaries learned from CICIoV2024 are stable enough to support real-time separation between benign and malicious CAN traffic under mixed conditions. This result resonates with earlier work that promotes tree-based ensembles as effective front-line detectors in automotive IDS architectures [32], where a simple Normal/Attack gate is often used to filter flows before any deeper multi-class analysis is attempted. For Legacy-IDS, the CAN binary streaming results therefore play a dual role. First, they validate that the unified pipeline and simulator do not introduce hidden implementation flaws that would silently degrade detection quality once the models are deployed in a streaming context. Second, they suggest a pragmatic deployment strategy: even if multi-class performance were to deteriorate further under harsher conditions or unseen attack types, the binary detectors already provide a reliable safety net that flags suspicious traffic with very low false-positive rates. In other words, for the CAN side of the system, the main research challenge is not to secure basic anomaly detection, but to extend and stress-test the multi-class component and the handling of novel behaviors.

5.4.2 MODBUS

Similar to CAN, a MODBUS dataset with the ICS-Flow structure was generated using the simulator. This dataset combines 50% of synthetic attacks with 50% of real attacks (from the ICS-Flow dataset) summing up to 30% of the dataset. The benign traffic is all simulated. The description of the dataset is presented in Table 5.19. The dataset was then streamed and frame by frame, the chosen model made the predictions. Below are the results of the stream for each model.

Table 5.19: Details of the MODBUS generated dataset .

Classes	**Total**
Benign	37250
DDoS	5121
IP-Scan	1200
MitM	4083
Port-Scan	2770
Replay	2549

Multi-Class

In the MODBUS stream, the multi-class performance of all models drops when compared to the near-perfect results obtained on the evaluation of the test set in the previous steps, as shown in Table 5.15. As described in Table 5.20 the results suffer a significant fall with accuracies and macro-F1 of 0.183207 and 0.343465 for Random Forest, 0.214411 and 0.415785 for Extra Trees and, 0.238555 and 0.397275 for XGBoost. These values indicate that, under the mixed real/synthetic MODBUS workload, the models still manage to recognize some attack patterns but struggle to preserve a clean separation across all six classes.

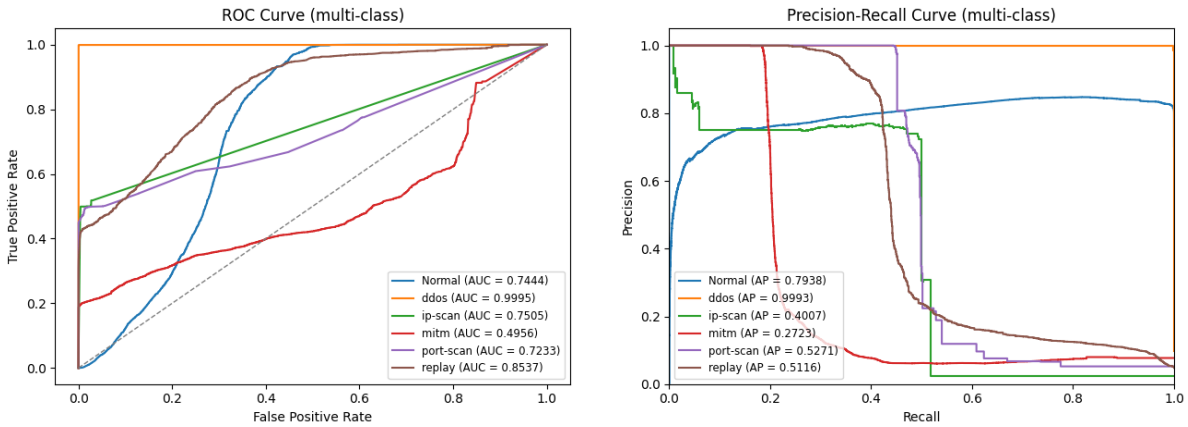
Breaking the results by class, a very asymmetric behavior is visible. DDoS remains the easiest class, with all models reaching an almost perfect recall, from 0.9869 to 0.9994. Extra Trees manages to attain an F1-score of 0.9803, with XGBoost in second place with 0.8638 and Random forest in last with 0.4367. IP-Scan, Port-Scan and Replay are detected with moderate success, with F1-scores for IP-Scan and Port-Scan going from 0.3652 to 0.3821 and, 0.3821 to 0.6134 respectively, while Replay fluctuates between 0.2982 and 0.5275, depending on the model.

Table 5.20: Comparison of multi-class results for the selected ML methods with a streamed dataset (MODBUS, Multi).

Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
RF	Benign	0.183207	0.4916	0.0055	0.0109	0.343465
	DDoS		0.2794	0.9992	0.4367	
	IP-Scan		0.7474	0.2417	0.3652	
	MitM		0.0605	0.4673	0.1071	
	Port-Scan		0.9741	0.4477	0.6134	
	Replay		0.9139	0.3707	0.5275	
ET	Benign	0.214411	0.0000	0.0000	0.0000	0.415785
	DDoS		0.9738	0.9869	0.9803	
	IP-Scan		0.7543	0.2558	0.3821	
	MitM		0.0856	0.9003	0.1563	
	Port-Scan		0.7831	0.4783	0.5939	
	Replay		0.3739	0.3907	0.3821	
XGBoost	Benign	0.238555	0.8240	0.0634	0.1177	0.397275
	DDoS		0.7606	0.9994	0.8638	
	IP-Scan		0.7458	0.2567	0.3819	
	MitM		0.0665	0.5954	0.1197	
	Port-Scan		0.8134	0.4783	0.6024	
	Replay		0.2284	0.4296	0.2982	

MitM proves to be more problematic. Although Extra Trees reaches a very high recall of 0.9003 in this class, its precision drops significantly to 0.0856 and the F1-score to 0.1563, and Random Forest and XGBoost show similarly low F1 values, ranging from 0.1071 to 0.1197. However, the most critical weakness is the handling of Benign traffic. Random Forest and XGBoost achieve recalls of 0.0055 and 0.0634, and Extra Trees effectively fails to recognize benign samples (zero recall), meaning that a large proportion of normal flows are mislabeled as attacks. For a practical IDS, this results in an unsustainable false-alarm rate in MODBUS streams.

The confusion matrices in Figures 5.18, 5.20 and 5.22 illustrate these patterns. Most benign flows are absorbed by attack classes, particularly DDoS and Port-Scan, while minority attacks such as IP-Scan and Replay are often swapped with MITM or each other. Extra Trees allocates almost all benign frames to some attack label, while Random Forest and XGBoost leave only a small fraction correctly classified as normal. In contrast, DDoS flows are consistently mapped to the correct class, and a substantial portion of Port-Scan and Replay traffic is also correctly identified.



(a) ROC Curve (b) PR Curve

Figure 5.17: ROC/PR Curves for Random Forest (MODBUS, Multi)

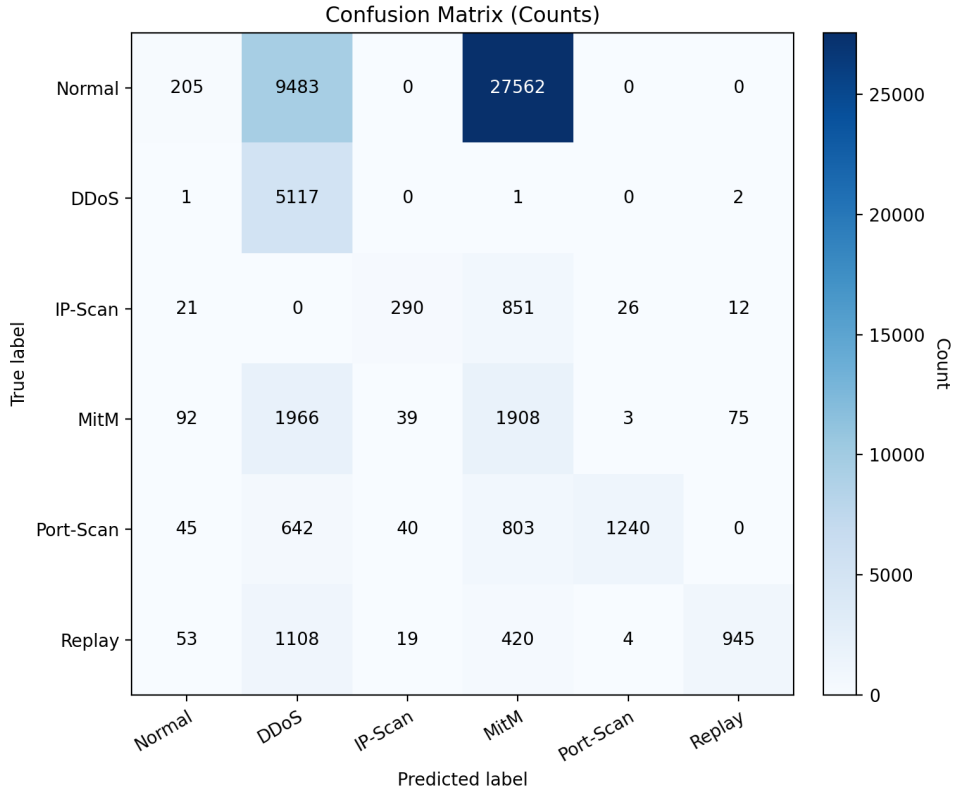
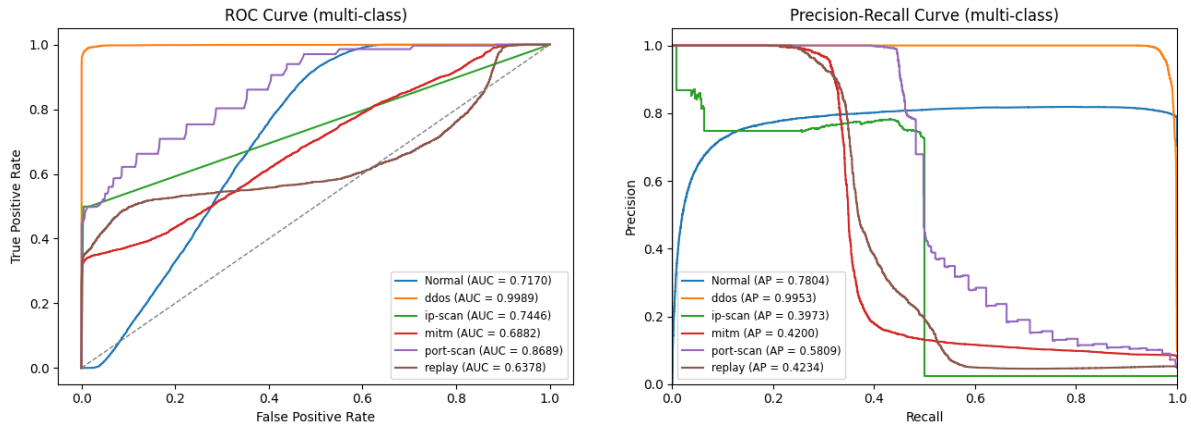


Figure 5.18: Confusion matrix for Random Forest with absolute values (Stream evaluation, MODBUS)

The ROC and PR curves for the streaming evaluation (Figures 5.17a–5.21b) contribute to the understanding of the discrepancy found between class-wise and global performance. For all three models, the DDoS trace still lies close to the upper-left corner in the ROC space and maintains high precision throughout most of the recall range in the PR space, confirming that the decision surfaces learned in the previous steps continue to separate DDoS traffic well from the remaining classes. IP-Scan, Port-Scan and



(a) ROC Curve (b) PR Curve

Figure 5.19: ROC/PR Curves for Extra Trees (MODBUS, Multi)

Replay present intermediate curves: their ROC traces stay above the diagonal, and the corresponding PR curves show reasonable precision at moderate recall, but precision decays quickly as the classifier attempts to recover more positives. Benign and MITM show the weakest behavior, with ROC curves closer to the diagonal and PR curves that start with modest precision and then collapse as recall increases, reflecting the strong confusion seen in the matrices.

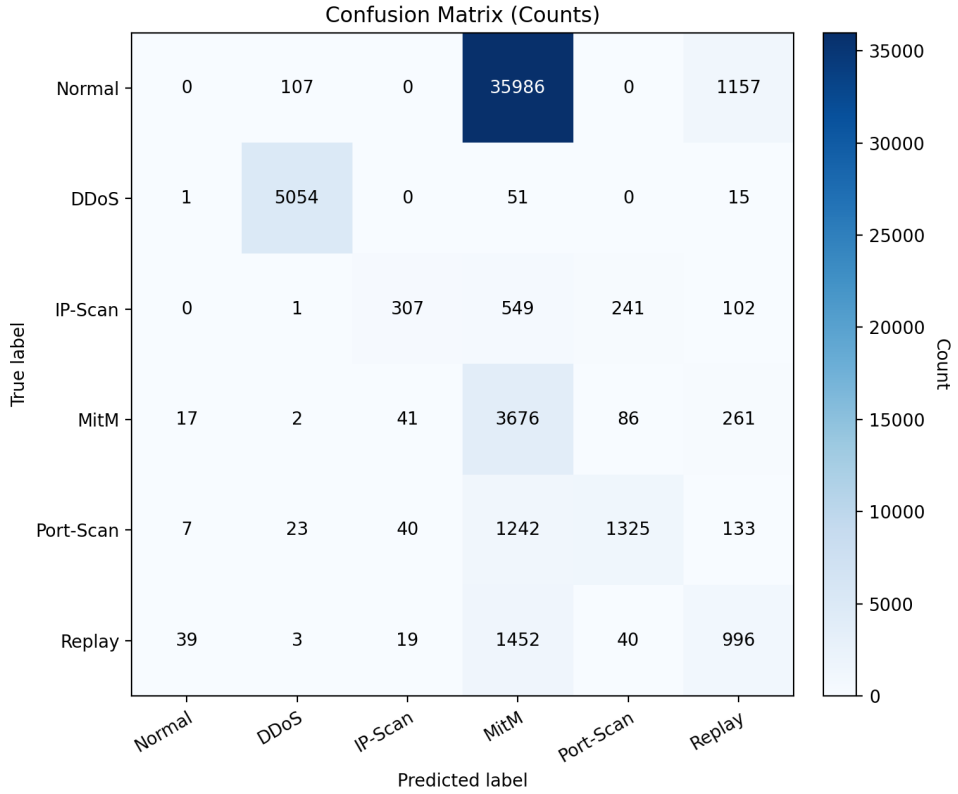
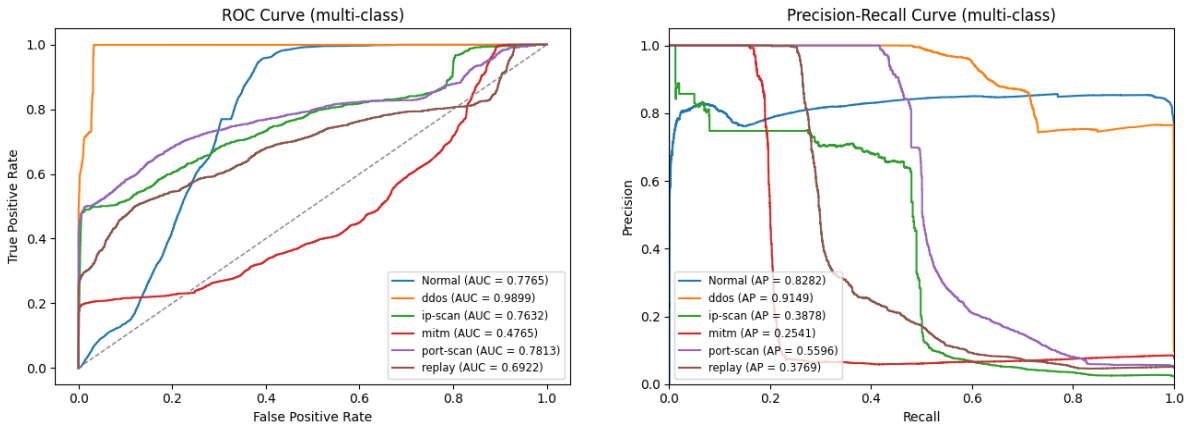


Figure 5.20: Confusion matrix for Extra Trees with absolute values (Stream evaluation, MODBUS).



(a) ROC Curve (b) PR Curve

Figure 5.21: ROC/PR Curves for XGBoost (MODBUS, Multi)

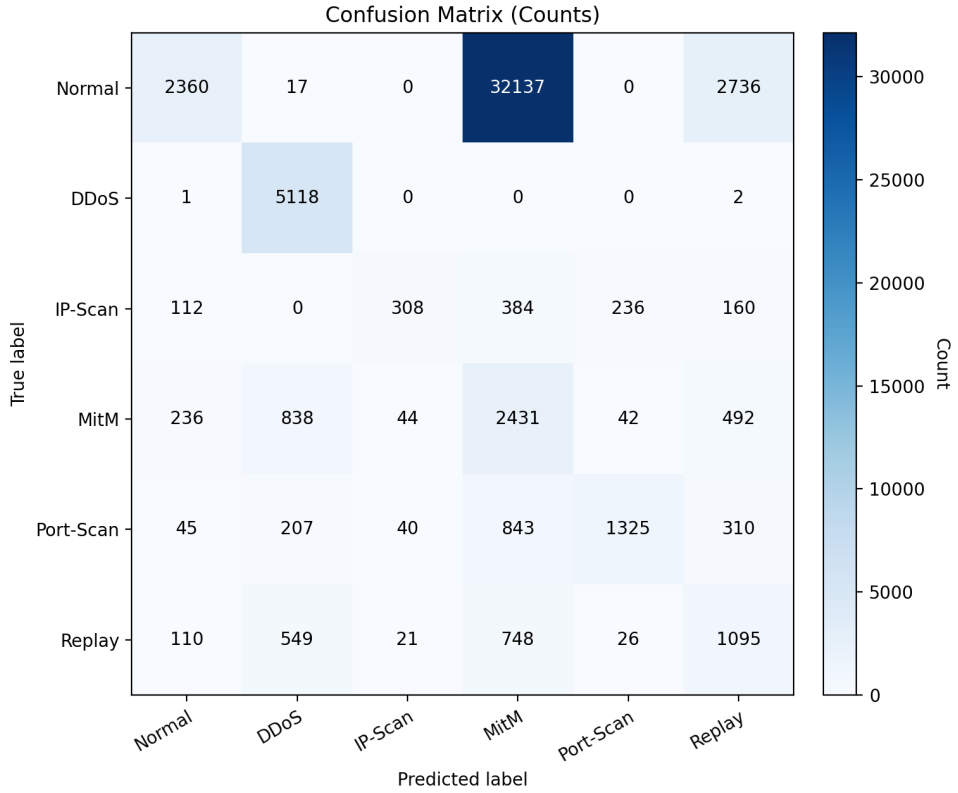


Figure 5.22: Confusion matrix for XGBoost with absolute values (Stream evaluation, MODBUS).

The multi-class streaming evaluation on MODBUS reveals a much more severe impact of combining real ICS-Flow traces with synthetic traffic. While DDoS remains well detected, with high recall and solid F1-scores across all models, the overall accuracies and macro-F1 scores collapse compared with the static test-set results reported in Phase II and III. Minority attacks such as IP-Scan, Port-Scan and Replay retain only moderate F1 values, and the benign class suffers dramatic losses in recall, particularly under

Extra Trees, which essentially fails to recognize normal flows. In effect, the decision surfaces that were neatly aligned with the original ICS-Flow distribution are no longer adequate once new payload patterns from the simulator are introduced. This behavior highlights a limitation in the current studies, where strong performance on a single, carefully curated benchmark, in a static lab dataset, does not guarantee that a model will tolerate changes in network topology, device mix or attack generation processes. The Phase III streaming experiments explicitly expose this gap for ICS-Flow. For MODBUS, the macro-F1 gains obtained through hyperparameter optimization and ensemble selection cannot, by themselves, ensure that the model will maintain its behavior under a mixed real/synthetic workload.

Binary

In contrast with the CAN stream, the binary MODBUS evaluation exposes a marked degradation when the models are confronted with the mixed real/synthetic workload. As summarized in Table 5.21, all three ensembles achieve high sensitivity to attacks but largely lose the ability to recognize normal traffic. The overall accuracies drop to 0.336568 for Random Forest, 0.296340 for Extra Trees and 0.305099 for XGBoost, with macro-F1 scores between 0.228597 and 0.294958. This significant decline compared with the near-perfect results from previous steps highlight a shift between the training distribution and the streamed MODBUS data. Random Forest results illustrate this asymmetry, achieving an attack recall of 0.9762 and an F1-score of 0.4662 while the Normal class reaches only 0.0666 recall and 0.1237 F1. In practice, the classifier tends to label most flows as attacks, capturing almost all malicious instances but at the expense of misclassifying the majority of normal MODBUS traffic. The high precision for Normal (0.8690) is therefore misleading: only a very small fraction of normal flows are ever predicted as such, and almost all normal behavior is absorbed into the attack label. Extra Trees follow the same asymmetry, failing to predict all the Normal instances (precision, recall and F1 all equal to 0.0000), while the Attack class achieves a recall of 0.9984 and an F1-score of 0.4572.

Table 5.21: Comparison of binary results for the selected ML methods with a streamed dataset (MODBUS, Binary).

Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
RF	Normal	0.336568	0.8690	0.0666	0.1237	0.294958
	Attack		0.3063	0.9762	0.4662	
ET	Normal	0.296340	0.0000	0.0000	0.0000	0.228597
	Attack		0.2965	0.9984	0.4572	

(*continues on next page*)

(*continued from previous page*)

Model	Class	Overall Acc.	Prec.	Rec.	F1	Macro F1
XGBoost	Normal	0.305099	0.7197	0.0193	0.0376	0.246917
	Attack		0.2971	0.9822	0.4562	

XGBoost occupies an intermediate position between Random Forest and Extra Trees, but with the same underlying bias. The Attack class is again recovered with very high recall (0.9822) and an F1-score of 0.4562, while the Normal class suffers from a recall of only 0.0193 and an F1-score of 0.0376. This means that under streaming conditions, the classifier behaves almost as a one-sided attack detector that rarely accepts traffic as normal.

Concluding, all three ensembles behave almost like "attack detectors" that rarely classify any traffic as normal. From a purely security-oriented standpoint, high attack recall might appear attractive, but the associated collapse in Normal recall and F1-score would translate into an unsustainable volume of false alarms in an operational setting. In the current configuration, Legacy-IDS can still identify most malicious MODBUS flows in the stream, but it does so at the cost of almost losing the notion of baseline behavior. For a cross-domain IDS, this imbalance has important design implications. On the CAN side, the system can safely rely on binary detectors as reliable sentinels. However, in the MODBUS side, the same strategy would inundate operators with alerts and make it difficult to distinguish genuine attacks from routine traffic fluctuations.

6 Conclusions

This work set out to examine how machine-learning-based intrusion detection can be used to protect two distinct but structurally similar domains, such as Intra-vehicular CAN networks in the Internet of Vehicles and MODBUS-based Industrial Control Systems. Both environments rely on legacy protocols with no built-in security mechanisms, operate under severe class imbalance between benign and attack traffic, and are increasingly exposed through their integration into larger IoT and IIoT ecosystems. Against these current limitations, this work had the objective of systematically evaluating a set of machine learning models for multi-class anomaly detection in realistic CAN and MODBUS scenarios, and consolidating these findings into a cross-domain IDS, Legacy-IDS, capable of handling both protocols within a single, reproducible pipeline.

The experimental design was structured in three phases. Phase I focused on the CICIoV2024 dataset, treating CAN traffic as a multi-class problem with naturally imbalanced classes, while phase II extended the analysis to ICS-Flow. In both phases, the same set of algorithms was studied: Random Forest, Extra Trees, XGBoost, AdaBoost, Logistic Regression, and neural models (ANN and DNN in the ICS case). Optuna was used to conduct a systematic hyperparameter search with cross-validation, providing a controlled way to explore the trade-off between predictive performance and computational cost. In Phase III, the work departed from notebook-based experimentation and reimplemented the best-performing models inside a unified Python project, Legacy-IDS, which integrates protocol detection, preprocessing, model training, evaluation, and streaming inference for both CAN and MODBUS.

The results of Phase I show that multi-class intrusion detection on CAN traffic is now an easy task when following the used configuration. While preserving class imbalance, tree-based ensembles and deep neural networks achieve near-perfect macro-F1 and class-wise F1-scores. Logistic Regression remains competitive but systematically trails the ensembles, while AdaBoost struggles more clearly with the skewed distribution. Hyperparameter optimization via Optuna improves all models, but the efficiency–performance balance clearly favors ensembles, suggesting that, for CAN-based IoV IDS in settings similar to CICIoV2024, well-tuned ensembles are an excellent default choice, combining high accuracy, good performance under imbalance, and reasonable runtime. In Phase II, the problem becomes more demanding. ICS-Flow brings a six-class MODBUS scenario with realistic asymmetries between categories such as Normal, DDoS, IP-Scan, MitM, Port-Scan, and Replay. Although Optuna again improves performance across all algorithms, the results reveal that multi-class ICS detection is

considerably harder than the CAN case. Random Forest achieves the best macro-F1, closely followed by XGBoost and Extra Trees, with DNN and ANN only slightly behind and Logistic Regression and AdaBoost clearly weaker. The dominant classes, particularly Normal and DDoS, reach perfect or near-perfect F1, but minority and overlapping classes, especially IP-Scan and, to a lesser extent, Replay, remain problematic. Even with tuned hyperparameters, the models struggle to differentiate these attacks from each other and from other traffic, and performance on them is consistently lower. These results justify the choice of macro-F1 as a primary evaluation metric: unlike global accuracy, which is dominated by the Normal class and can hide failures on rare but critical attacks, macro-averaging exposes precisely where the models still fall short. Phase III initially evaluates binary and multi-class detection in a more operational setting and, then proceeds to a streaming evaluation. In the unified Legacy-IDS pipeline, the best models from Phases I and II (Random Forest, Extra Trees, XGBoost) are retrained for both CAN and MODBUS under consistent preprocessing and metric computation. In the binary evaluation, the three ensemble models achieved perfect results in accurately separating attacks from normal traffic in the CAN environment. In contrast, in MODBUS scenario, the results remained near-perfect. Although multi-class CAN detection stays strong, with macro-F1 staying very close to those reported in Phase I, static multi-class results in MODBUS reproduce the Phase II pattern, with ensembles leading while minority classes struggle. Nonetheless, the most distinctive part of Phase III is the streaming-oriented evaluation of Legacy-IDS on synthesized CAN and MODBUS streams that combine real traces and simulator-generated traffic. On the CAN side, the models maintain high discrimination ability. Attack categories are reliably detected, and benign traffic retains acceptable recall, meaning that in an IoV context, Legacy-IDS can act as a practical, real-time IDS for both binary and multi-class detection with manageable false-alarm rates. On the MODBUS side, however, the behavior is markedly different. Under streaming conditions with hybrid ICS-Flow and simulated traffic, the ensembles tend to behave as almost pure attack detectors, meaning that the models recover most malicious flows with very high recall but misclassify a large share of benign traffic as attacks, driving the recall and F1 for the Normal class down to low values. This imbalance between high attack recall and poor benign recognition would generate an unacceptably high alert volume in practice and illustrates how distribution shifts and class imbalance can degrade a model that performs well in static evaluation.

6.1 Contributions and Limitations

Beyond the experimental results, this work makes a set of concrete contributions. First, it provides a comparative, cross-domain study of multi-class intrusion detection across two widely used, realistic datasets, CICIoV2024 and ICS-Flow, deliberately preserving their natural imbalance rather than relying on artificial resampling. The CAN-focused

experimental evidence developed in Phase I was published in a peer-reviewed article [157], and the MODBUS results reported in Phase II were subsequently extended and validated in a second publication [158]. Together, these peer-reviewed outputs strengthen the empirical foundation of this work and support a more faithful assessment of how models behave in conditions that resemble real deployments in IoV and ICS. Second, it systematically shows that tree-based ensembles tuned with Optuna form a strong baseline for both CAN and MODBUS, outperforming or matching more complex neural architectures while being computationally cheaper to train, and that Logistic Regression can still be a viable option when interpretability is important and perfect performance is not required. Third, it introduces Legacy-IDS, a unified command-line toolkit that encapsulates data loading, preprocessing, hyperparameter optimization, model training and evaluation, and streaming inference for both CAN and MODBUS, which has not been done yet as far as we know, avoiding the fragmentation common in notebook-centric research and providing a reproducible base for future work. Fourth, it delivers a CAN-BUS Dataset Converter that standardizes disparate CAN logs and CSV files into a consistent schema compatible with CICIoV2024, lowering the barrier to integrating new CAN datasets into the same IDS workflow.

Despite these contributions, the work also has clear limitations that frame its scope. The empirical evaluation is restricted to two datasets and two protocols. While CICIoV2024 and ICS-Flow are representative and widely cited, they cannot capture the full diversity of real-world IoV and ICS deployments, other protocols (such as FlexRay, LIN, Profinet, DNP3 or MQTT), or the variety of device configurations and traffic patterns in the field. The number of Optuna trials per model was kept deliberately modest to respect the available computational resources, which may leave some performance on the table, particularly for high-capacity models like DNNs. Feature engineering was intentionally lightweight to preserve comparability and avoid overfitting to specific datasets, but this also means that protocol-level and temporal information that could help distinguish difficult classes (such as IP-Scan and replay-like behavior) was only partially exploited. Finally, the streaming evaluation, while closer to deployment than static train/test splits, still relies on offline-synthesized streams and does not include measurements of latency, resource consumption, or integration with actual ICS or vehicular platforms.

Summing up, the main contributions are as follows:

- **Multi-Class Intrusion Detection in Internet of Vehicles: Optimizing Machine Learning Models on Imbalanced Data**: Peer-reviewed multi-class CAN intrusion detection results on CICIoV2024 under natural class imbalance, establishing strong evidence for ensemble-based IDS in IoV settings [157].
- **An Optimized Multi-class Classification for Industrial Control Systems**: Extended and validated multi-class MODBUS intrusion detection results on ICS-Flow,

confirming the strengths and limits of the same model families under a more challenging ICS scenario [158].

- **GitHub Repositories**:
  - **Legacy-IDS**
    `https://github.com/AgataPalma/Legacy-IDS`
  - **CAN-BUS Dataset Converter**
    `https://github.com/AgataPalma/CANBUS_DatasetConverter`
  - **Multi-Class Intrusion Detection on CICIoV2024**
    `https://github.com/AgataPalma/Multi-ClassIntDetectCICIoV`
  - **Multi-Class Intrusion Detection on ICS-Flow**
    `https://github.com/AgataPalma/Multi-ClassIntDetectICSFlow`

6.2 Future Work

The previously mentioned limitations suggest several directions for future work. First is to broaden the data basis of Legacy-IDS by incorporating additional CAN and MODBUS datasets and, progressively, other industrial and vehicular protocols, testing whether the same pipeline and model choices generalize across different environments. Since we observed degradation in the results when adding synthetic data, we should expect a similar response when the model is confronted with other datasets. Another is to invest in richer feature design, including temporal aggregation windows and a different set of feature combinations. At the system level, Legacy-IDS can be extended in several ways. The current toolkit already structures training, evaluation, and streaming inference, but it can be enriched with mechanisms for incremental learning to adapt to evolving traffic patterns without full retraining. Another possibility for future work is to evaluate the use of a single ensemble model as a first line of defense in a binary task, flagging potential attacks to a multi-class classification problem, capable of also identifying novel attacks. Finally, the CAN-BUS Dataset Converter could be expanded to cover more input formats, more expressive mapping logic, and automated protocol fingerprinting, further simplifying the ingestion of legacy logs.

In conclusion, this work shows that, under realistic imbalance and with careful design, multi-class anomaly detection on CAN and MODBUS is feasible and that a unified cross-domain IDS is not only technically feasible but also advantageous. It also makes clear that good static metrics do not automatically translate into operationally acceptable behavior, especially in industrial settings where benign traffic must be reliably recognized to avoid overwhelming operators. The contributions of this work lie in mapping these strengths and weaknesses with precision, offering a practical toolkit with Legacy-IDS, and clarifying where future research needs to concentrate.

Bibliography

[1] S. L. Sreerama, H. R. Shashank, R. B. Shashank, T. Shankar, R. Hemavathy, and P. R. Kumar, "Anomaly detection model for bottles in a manufacturing unit," in *3rd International Conference on Innovative Mechanisms for Industry Applications, ICIMIA 2023 - Proceedings*. Institute of Electrical and Electronics Engineers Inc., 2023, pp. 1488–1493.

[2] Z. Allal, H. Noura, O. Salman, and A. Chehab, "Advanced anomaly detection in energy control systems using machine learning and feature engineering," in *Proceedings of the 8th Cyber Security in Networking Conference: AI for Cybersecurity, CSNet 2024*. Institute of Electrical and Electronics Engineers Inc., 2024, pp. 15–21.

[3] I. Almazyad, S. Shao, S. Hariri, and H. A. Kholidy, "Anomaly behavior analysis of smart water treatment facility service: Design, analysis, and evaluation," in *2023 20th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA)*, 2023, pp. 1–7.

[4] N. Yalcin, S. Cakir, and S. Ualdi, "Attack detection using artificial intelligence methods for scada security," *IEEE Internet of Things Journal*, 2024.

[5] H. Taslimasa, S. Dadkhah, E. C. P. Neto, P. Xiong, S. Ray, and A. A. Ghorbani, "Security issues in internet of vehicles (iov): A comprehensive survey," *Internet of Things (Netherlands)*, vol. 22, 7 2023.

[6] W. Gong, S. Yang, H. Guang, B. Ma, B. Zheng, Y. Shi, B. Li, and Y. Cao, "Multi-order feature interaction-aware intrusion detection scheme for ensuring cyber security of intelligent connected vehicles," *Engineering Applications of Artificial Intelligence*, vol. 135, 9 2024.

[7] E. C. P. Neto, H. Taslimasa, S. Dadkhah, S. Iqbal, P. Xiong, T. Rahman, and A. A. Ghorbani, "Ciciov2024: Advancing realistic ids approaches against dos and spoofing attack in iov can bus," *Internet of Things*, vol. 26, p. 101209, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2542660524001501

[8] E. Amer and T. Elboghdadly, "Evaluating machine learning techniques for ics security: Insights from dataset limitations and classifier performance," in *4th International Mobile, Intelligent, and Ubiquitous Computing Conference, MIUCC 2024*. Institute of Electrical and Electronics Engineers Inc., 2024, pp. 368–373.

[9] A. Dehlaghi-Ghadim, N. Ericsson, L.-G. Magnusson, M. Eriksson, M. H. Moghadam, A. Balador, and H. Hansson, "Using decision support to fortify industrial control system against cyberattacks," in *2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 9 2024, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/document/10710892/

[10] I. Siniosoglou, P. Radoglou-Grammatikis, G. Efstathopoulos, P. Fouliras, and P. Sarigiannidis, "A unified deep learning anomaly detection and classification approach for smart grid environments," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1137–1151, 2021.

[11] A. Dehlaghi-Ghadim, M. H. Moghadam, A. Balador, and H. Hansson, "Anomaly detection dataset for industrial control systems," *IEEE Access*, vol. 11, pp. 107 982–107 996, 2023.

[12] H. Ochiai, M. D. Hossain, P. Chirupphapa, Y. Kadobayashi, and H. Esaki, "Modbus/rs-485 attack detection on communication signals with machine learning," *IEEE Communications Magazine*, vol. 61, pp. 43–49, 6 2023.

[13] M. Muhammad, A. S. Alshra'a, and R. German, "Survey of cybersecurity in smart grids protocols and datasets," *Procedia Computer Science*, vol. 241, pp. 365–372, 2024, 19th International Conference on Future Networks and Communications/ 21th International Conference on Mobile Systems and Pervasive Computing/14th International Conference on Sustainable Energy Information Technology. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050924017605

[14] S. Banik, T. Banik, S. M. M. Hossain, and S. K. Saha, "Implementing man-in-the-middle attack to investigate network vulnerabilities in smart grid test-bed," in *2023 IEEE World AI IoT Congress, AIIoT 2023*. Institute of Electrical and Electronics Engineers Inc., 2023, pp. 345–351.

[15] H. Zhang, Y. Min, S. Liu, H. Tong, and Y. Li, "Modlstm: A method to recognize dos attacks on modbus/tcp," in *Conference Proceedings of the IEEE International Performance, Computing, and Communications Conference*, vol. 2022-November. Institute of Electrical and Electronics Engineers Inc., 2022, pp. 319–324.

[16] U. J. Otokwala and A. Petrovski, "Ensemble common features technique for lightweight intrusion detection in industrial control system," in *Proceedings - 2023 IEEE 6th International Conference on Industrial Cyber-Physical Systems, ICPS 2023*. Institute of Electrical and Electronics Engineers Inc., 2023.

[17] D. Pliatsios, P. Sarigiannidis, T. Lagkas, and A. G. Sarigiannidis, "A survey on scada systems: Secure protocols, incidents, threats and tactics," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1942–1976, 2020.

[18] R. O. Ogundokun, P. A. Owolawi, and E. Van Wyk, "Litert-idsnet: A lightweight hybrid deep learning framework for real-time intrusion detection in industrial iot using the rt-iot 2022 dataset," in *2025 60th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST)*, 2025, pp. 1–4.

[19] S. Wang, B. Zheng, Z. Liu, Z. Fan, Y. Liu, and Y. Dai, "A lightweight intrusion detection system for vehicular networks based on an improved vit model," *IEEE Access*, vol. 12, pp. 118 842–118 856, 2024.

[20] T. Miller, A. Staves, S. Maesschalck, M. Sturdee, and B. Green, "Looking back to look forward: Lessons learnt from cyber-attacks on industrial control systems," *International Journal of Critical Infrastructure Protection*, vol. 35, p. 100464, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1874548221000524

[21] P. Cheng, K. Xu, S. Li, and M. Han, "Tcan-ids: Intrusion detection system for internet of vehicle using temporal convolutional attention network," *Symmetry*, vol. 14, 2 2022.

[22] M. M. El-Gayar, F. A. Alrslani, and S. El-Sappagh, "Smart collaborative intrusion detection system for securing vehicular networks using ensemble machine learning model," *Information (Switzerland)*, vol. 15, 10 2024.

[23] L. Yang, A. Moubayed, and A. Shami, "Mth-ids: A multitiered hybrid intrusion detection system for internet of vehicles," *IEEE Internet of Things Journal*, vol. 9, pp. 616–632, 1 2022.

[24] S. Wang, Y. Wang, B. Zheng, J. Cheng, Y. Su, and Y. Dai, "Intrusion detection system for vehicular networks based on mobilenetv3," *IEEE Access*, vol. 12, pp. 106 285–106 302, 2024.

[25] M. Almehdhar, A. Albaseer, M. A. Khan, M. Abdallah, H. Menouar, S. Al-Kuwari, and A. Al-Fuqaha, "Deep learning in the fast lane: A survey on advanced intrusion detection systems for intelligent vehicle networks," *IEEE Open Journal of Vehicular Technology*, vol. 5, pp. 869–906, 2024.

[26] F. Jin, M. Chen, W. Zhang, Y. Yuan, and S. Wang, "Intrusion detection on internet of vehicles via combining log-ratio oversampling, outlier detection and metric learning," *Information Sciences*, vol. 579, pp. 814–831, 11 2021.

[27] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, A. Teredesai, V. Kumar, Y. Li, R. Rosales, E. Terzi, and G. Karypis, Eds. ACM, 2019, pp. 2623–2631. [Online].

Available: https://doi.org/10.1145/3292500.3330701

[28] B. Lampe and W. Meng, "Intrusion detection in the automotive domain: A comprehensive review," *IEEE Communications Surveys and Tutorials*, vol. 25, pp. 2356–2426, 2023.

[29] W. Wu, J. H. Joloudari, S. K. Jagatheesaperumal, K. N. Rajesh, S. Gaftandzhieva, S. Hussain, R. Rabih, N. Haqjoo, M. Nazar, H. Vahdat-Nejad, and R. Doneva, "Deep transfer learning techniques in intrusion detection system-internet of vehicles: A state-of-the-art review," *Computers, Materials and Continua*, vol. 80, pp. 2785–2813, 2024.

[30] M. Fu, P. Wang, M. Liu, Z. Zhang, and X. Zhou, "Iov-bert-ids: Hybrid network intrusion detection system in iov using large language models," *IEEE Transactions on Vehicular Technology*, 2024.

[31] S. T. Mehedi, A. Anwar, Z. Rahman, and K. Ahmed, "Deep transfer learning based intrusion detection system for electric vehicular networks," *Sensors*, vol. 21, 7 2021.

[32] D. Oladimeji, R. Jinad, A. Rasheed, and M. Baza, "Canguard: An enhanced approach to the detection of anomalies in can-enabled vehicles," *Sensors*, vol. 25, 1 2025.

[33] T. Moulahi, S. Zidi, A. Alabdulatif, and M. Atiquzzaman, "Comparative performance evaluation of intrusion detection based on machine learning in in-vehicle controller area network bus," *IEEE Access*, vol. 9, pp. 99 595–99 605, 2021.

[34] J. Nagarajan, P. Mansourian, M. A. Shahid, A. Jaekel, I. Saini, N. Zhang, and M. Kneppers, "Machine learning based intrusion detection systems for connected autonomous vehicles: A survey," *Peer-to-Peer Networking and Applications*, vol. 16, pp. 2153–2185, 9 2023.

[35] J. Qin, Y. Xun, and J. Liu, "Cvmids: Cloud-vehicle collaborative intrusion detection system for internet of vehicles," *IEEE Internet of Things Journal*, vol. 11, pp. 321–332, 1 2024.

[36] L. Du, Z. Gu, Y. Wang, and C. Gao, "Open world intrusion detection: An open set recognition method for can bus in intelligent connected vehicles," *IEEE Network*, vol. 38, pp. 76–82, 5 2024.

[37] F. Aloraini, A. Javed, and O. Rana, "Adversarial attacks on intrusion detection systems in in-vehicle networks of connected and autonomous vehicles," *Sensors*, vol. 24, 6 2024.

[38] Robert Bosch GmbH, "Can specification version 2.0," Robert Bosch GmbH, Gerlingen, Germany, Technical Report, 1991, version 2.0.

[39] H. Kang, T. Vo, H. K. Kim, and J. B. Hong, "Canival: A multimodal approach to intrusion detection on the vehicle can bus," *Vehicular Communications*, vol. 50, 12 2024.

[40] R. Sangkhro and A. K. Agrawal, "Cybersecurity in industrial control systems: A review of the current trends and challenges," in *2023 10th International Conference on Computing for Sustainable Global Development (INDIACom)*, 2023, pp. 355–359.

[41] H. Kheddar, Y. Himeur, and A. I. Awad, "Deep transfer learning for intrusion detection in industrial control networks: A comprehensive review," *Journal of Network and Computer Applications*, vol. 220, 2023, cited by: 65; All Open Access, Green Open Access.

[42] R. Eltomy and W. Lalouani, "Explainable intrusion detection in industrial control systems," in *2024 IEEE 7th International Conference on Industrial Cyber-Physical Systems, ICPS 2024*. IEEE Inc., 2024.

[43] S. Rakas, M. D. Stojanović, and J. Marković-Petrović, "A review of research work on network-based scada intrusion detection systems," *IEEE Access*, vol. 8, pp. 93 083–93 108, 2020.

[44] M. Asiri, A. Arunasalam, N. Saxena, and Z. B. Celik, "Frontline responders: Rethinking indicators of compromise for industrial control system security," *Computers and Security*, vol. 154, 2025, cited by: 0; All Open Access.

[45] A. Farraj, "On using zero trust to securing industrial control systems in the power systems industry," in *2025 IEEE Texas Power and Energy Conference (TPEC)*, 2025, pp. 1–5.

[46] A. Plager, E. Olexa, D. Gardner, B. Torres, A. Hays, and A. Farraj, "A study of scada system vulnerabilities and man-in-the-middle threats in substation operations," in *2025 IEEE Texas Power and Energy Conference (TPEC)*, 2025, pp. 1–5.

[47] G. Lazaridis, A. Drosou, P. Chatzimisios, and D. Tzovaras, "Unraveling the threat landscape of cps: Modbus tcp vulnerabilities in the era of i4.0," in *Proceedings of the 2024 IEEE International Conference on Cyber Security and Resilience, CSR 2024*. Institute of Electrical and Electronics Engineers Inc., 2024, pp. 593–598.

[48] W. Alsabbagh, S. Amogbonjaye, D. Urrego, and P. Langendörfer, "A stealthy false command injection attack on modbus based scada systems," in *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*, 2023, pp. 1–9.

[49] F. Katulić, D. Sumina, S. Groš, and I. Erceg, "Protecting modbus/tcp-based industrial automation and control systems using message authentication codes," *IEEE Access*, vol. 11, pp. 47 007–47 023, 2023.

[50] G. Yadav and K. Paul, "Architecture and security of scada systems: A review," *International Journal of Critical Infrastructure Protection*, vol. 34, 9 2021.

[51] S. S. Gujar, "Securing industrial control systems against cyber-physical attacks," in *2025 7th International Conference on Intelligent Sustainable Systems (ICISS)*, 2025, pp. 465–470.

[52] K. E. Hemsley and R. E. Fisher, "History of industrial control system cyber incidents," Idaho National Laboratoy, Tech. Rep., 12 2018. [Online]. Available: https://www.osti.gov/servlets/purl/1505628

[53] J. Slay and M. Miller, "Lessons learned from the maroochy water breach," in *Critical Infrastructure Protection*, E. Goetz and S. Shenoi, Eds. Boston, MA: Springer US, 2008, pp. 73–82.

[54] M. D. Abrams and J. Weiss, "Malicious control system cyber security attack case study-maroochy water services, australia," MITRE, Tech. Rep., 2008. [Online]. Available: https://www.mitre.org/sites/default/files/pdf/08_1145.pdf

[55] A. Hassanzadeh, A. Rasekh, S. Galelli, M. Aghashahi, R. Taormina, A. Ostfeld, and M. K. Banks, "A review of cybersecurity incidents in the water sector," *Journal of Environmental Engineering*, vol. 146, no. 5, p. 03120003, 2020. [Online]. Available: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29EE.1943-7870.0001686

[56] S. K. Venkatachary, J. Prasad, A. Alagappan, L. J. B. Andrews, R. A. Raj, and S. Duraisamy, "Cybersecurity and cyber-terrorism challenges to energy-related infrastructures – cybersecurity frameworks and economics – comprehensive review," *International Journal of Critical Infrastructure Protection*, vol. 45, 7 2024.

[57] M. Foundstone, P. Services, and M. Labs, "Global energy cyberattacks: "night dragon"," McAfee, Tech. Rep., 2011. [Online]. Available: https://www.mcafee.com/blogs/wp-content/uploads/2011/02/McAfee_NightDragon_wp_draft_to_customersv1-1.pdf

[58] K. Lab, "The duqu 2.0 technical details," Kaspersky Lab, Tech. Rep., 6 2015. [Online]. Available: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/07205202/The_Mystery_of_Duqu_2_0_a_sophisticated_cyberespionage_actor_returns.pdf

[59] Z. Dehlawi and N. Abokhodair, "Saudi arabia's response to cyber conflict: A case study of the shamoon malware incident," in *2013 IEEE International Conference on Intelligence and Security Informatics*, 2013, pp. 73–75.

[60] C. Wueest, "Security response - targeted attacks against the energy sector," Symantec, Tech. Rep., 1 2014. [Online]. Available: https://community.broadcom.com/HigherLogic/System/DownloadDocumentFile.ashx?DocumentFileKey=177c15b0-66ca-4319-bf31-8fa0089fbbfd&forceDialog=1

[61] U.S. Department of Justice, "Seven iranians working for IRGC-

affiliated entities charged for conducting coordinated cyber attacks against U.S. financial sector," https://www.justice.gov/archives/opa/pr/seven-iranians-working-islamic-revolutionary-guard-corps-affiliated-entities-charged, Mar. 2016, press release, Mar 24, 2016.

[62] NCCIC/ICS-CERT, "Ics focused malware (Havex)," https://www.cisa.gov/news-events/ics-advisories/icsa-14-178-01, U.S. Department of Homeland Security, Tech. Rep., 2014, iCS-ALERT-14-176-02A (superseded; see advisory ICSA-14-178-01 for details).

[63] ——, "Ongoing sophisticated malware campaign compromising ICS (blackenergy/killdisk updates)," https://www.cisa.gov/news-events/ics-alerts/ics-alert-14-281-01e, U.S. Department of Homeland Security, Tech. Rep., 2014, iCS-ALERT-14-281-01 (and updates through 2016).

[64] Symantec Threat Intelligence, "Western energy sector targeted by sophisticated attack group (dragonfly 2.0)," https://www.security.com/threat-intelligence/dragonfly-energy-sector-cyber-attacks, Oct. 2017.

[65] R. M. Lee, M. J. Assante, and T. Conway, "Analysis of the cyber attack on the ukrainian power grid," https://nsarchive.gwu.edu/sites/default/files/documents/3891751/SANS-and-Electricity-Information-Sharing-and.pdf, E-ISAC and SANS Industrial Control Systems, Tech. Rep., Mar. 2016.

[66] R. Falcone, "Second wave of shamoon 2 attacks identified," UNIT42, Palo Alto Networks, Tech. Rep., 2017. [Online]. Available: https://unit42.paloaltonetworks.com/unit42-second-wave-shamoon-2-attacks-identified/

[67] D. Inc., "Analysis of the threat to electric grid operations," Dragos, Tech. Rep., 6 2017. [Online]. Available: https://www.dragos.com/wp-content/uploads/CrashOverride-01.pdf

[68] M. Crosignani, M. Macchiavelli, and A. F. Silva, "Pirates without borders: The propagation of cyberattacks through firms' supply chains," *Journal of Financial Economics*, vol. 147, pp. 432–448, 2 2023.

[69] F. B. Khan, A. Asad, H. Durad, S. M. Mohsin, S. N. Kazmi, and F. B. Khan, "Dragonfly cyber threats: A case study of malware attacks targeting power grids," *Journal of Computing & Biomedical Informatics*, vol. 4, 2023. [Online]. Available: https://jcbi.org/index.php/Main/article/view/137

[70] Symantec Threat Intelligence, "Western energy sector targeted by sophisticated attack group (dragonfly 2.0)," https://www.security.com/threat-intelligence/dragonfly-energy-sector-cyber-attacks, Oct. 2017.

[71] CISA/ICS-CERT, "Mar-17-352-01: Hatman—safety system targeted malware," https://www.cisa.gov/sites/default/files/documents/MAR-17-352-01%

20HatMan%E2%80%94Safety%20System%20Targeted%20Malware_S508C.pdf, CISA, Tech. Rep., Dec. 2017.

[72] S. Larson and S. Caltagirone, "Industrial cyberattacks in the middle east and international consequences," in *Cyber War and Cyber Peace: Digital Conflict in the Middle East*, M. Sexton and E. Campbell, Eds. London: I.B. Tauris, 2022, pp. 43–62. [Online]. Available: http://dx.doi.org/10.5040/9780755646036.0007

[73] M. Musluoglu, N. Kunicina, and J. Caiko, "Vulnerability assessment of industrial control systems for colonial pipeline and wannacry ransomware," in *2024 IEEE 65th International Scientific Conference on Power and Electrical Engineering of Riga Technical University (RTUCON)*, 2024, pp. 1–7.

[74] A. A. Albustami and A. F. Taha, "Breaking the flow and the bank: Stealthy cyberattacks on water network hydraulics," *Water Research*, vol. 283, p. 123719, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0043135425006281

[75] J. Cervini, A. Rubin, and L. Watkins, "Don't drink the cyber: Extrapolating the possibilities of oldsmar's water treatment cyberattack," *International Conference on Cyber Warfare and Security*, vol. 17, pp. 19–25, 03 2022.

[76] S. Lyngaas, "Russia-linked hacking group suspected of carrying out cyberattack on texas water facility, cybersecurity firm says," 4 2024. [Online]. Available: https://edition.cnn.com/2024/04/17/politics/russia-hacking-group-suspected-texas-water-cyberattack/index.html

[77] J. Reed, "Is the water safe? the state of critical infrastructure cybersecurity," 1 2025. [Online]. Available: https://www.ibm.com/think/insights/is-the-water-safe-state-of-critical-infrastructure-cybersecurity

[78] A. C. Kansas, "City of arkansas city faces cybersecurity incident," 9 2024. [Online]. Available: https://www.arkcity.org/environmental-services/page/city-arkansas-city-faces-cybersecurity-incident

[79] M. M. Rahman, S. A. Shakil, and M. R. Mustakim, "A survey on intrusion detection system in iot networks," *Cyber Security and Applications*, vol. 3, 12 2025.

[80] H. Nandanwar and R. Katarya, "Securing industry 5.0: An explainable deep learning model for intrusion detection in cyber-physical systems," *Computers and Electrical Engineering*, vol. 123, 4 2025.

[81] A. Alharthi, M. Alaryani, and S. Kaddoura, "A comparative study of machine learning and deep learning models in binary and multiclass classification for intrusion detection systems," *Array*, vol. 26, 7 2025.

[82] J. Ali, "Intrusion detection systems trends to counteract growing cyber-attacks on cyber-physical systems," in *2021 22nd International Arab Conference on Infor-*

mation Technology, ACIT 2021. Institute of Electrical and Electronics Engineers Inc., 2021.

[83] N. Canino, P. Dini, S. Mazzetti, D. Rossi, S. Saponara, and E. Soldaini, "Cybersecurity of automotive wired networking systems: Evolution, challenges, and countermeasures," *Electronics (Switzerland)*, vol. 14, 2 2025.

[84] "Randomforestclassifier — scikit-learn 1.7.2 documentation." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[85] "Extratreesclassifier — scikit-learn 1.7.2 documentation." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html

[86] "Xgboost documentation — xgboost 3.1.1 documentation." [Online]. Available: https://xgboost.readthedocs.io/en/stable/index.html

[87] "Adaboostclassifier — scikit-learn 1.7.2 documentation." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html#r33e4ec8c4ad5-2

[88] "Logisticregression — scikit-learn 1.7.2 documentation." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[89] D. Soni and N. Kumar, "Machine learning techniques in emerging cloud computing integrated paradigms: A survey and taxonomy," *Journal of Network and Computer Applications*, vol. 205, p. 103419, 9 2022.

[90] A. Subasi, "Machine learning techniques," *Practical Machine Learning for Data Analysis Using Python*, pp. 91–202, 2020.

[91] "Ray tune: Hyperparameter tuning — ray 3.0.0.dev0." [Online]. Available: https://docs.ray.io/en/master/tune/index.html

[92] "Github - hyperopt/hyperopt: Distributed asynchronous hyperparameter optimization in python." [Online]. Available: https://github.com/hyperopt/hyperopt

[93] "Tuning a scikit-learn estimator with skopt — scikit-optimize 0.8.1 documentation." [Online]. Available: https://scikit-optimize.github.io/stable/auto_examples/hyperparameter-optimization.html

[94] "Vertex ai|google cloud documentation." [Online]. Available: https://docs.cloud.google.com/vertex-ai/docs

[95] "The center for all your data, analytics, and ai – amazon sagemaker – aws." [Online]. Available: https://aws.amazon.com/sagemaker

[96] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, pp. 1263–1284, 9 2009.

[97] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, pp. 861–874, 6 2006.

[98] H. Lee, S. H. Jeong, and H. K. Kim, "Otids: A novel intrusion detection system for in-vehicle network by using remote frame," in *2017 15th Annual Conference on Privacy, Security and Trust (PST)*, 2017, pp. 57–5709.

[99] M. L. Han, B. I. Kwak, and H. K. Kim, "Anomaly intrusion detection method for vehicular networks based on survival analysis," *Vehicular Communications*, vol. 14, pp. 52–63, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214209618301189

[100] E. Seo, H. M. Song, and H. K. Kim, "Gids: Gan based intrusion detection system for in-vehicle network," in *2018 16th Annual Conference on Privacy, Security and Trust (PST)*, Aug 2018, pp. 1–6.

[101] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Vehicular Communications*, vol. 21, p. 100198, 2020.

[102] H. Kang, B. I. Kwak, Y. H. Lee, H. Lee, H. Lee, and H. K. Kim, "Car hacking and defense competition on in-vehicle network," 8 2021.

[103] ——, "Car hacking: Attack & defense challenge 2020 dataset," 2021. [Online]. Available: https://dx.doi.org/10.21227/qvr7-n418

[104] M. E. Verma, R. A. Bridges, M. D. Iannacone, S. C. Hollifield, P. Moriano, S. C. Hespeler, B. Kay, and F. L. Combs, "A comprehensive guide to can ids data & introduction of the road dataset," 2024. [Online]. Available: https://arxiv.org/abs/2012.14600

[105] S. B. H. Samir, M. Raissa, H. Touati, M. Hadded, and H. Ghazzai, "Machine learning-based intrusion detection for securing in-vehicle can bus communication," *SN Computer Science*, vol. 5, 12 2024.

[106] B. Lampe and W. Meng, "can-train-and-test: A curated can dataset for automotive intrusion detection," *Computers and Security*, vol. 140, 5 2024.

[107] S. Li, Y. Cao, Y. Zhang, T. Liao, F. Yan, and H. Lin, "A cloud collaborative-based intrusion detection and prevention system for ivn," *IEEE Transactions on Cognitive Communications and Networking*, 2024.

[108] M. F. Gul and H. Bakir, "Improving attack detection in iov systems using ga-based hyperparameter optimization," in *8th International Artificial Intelligence and Data Processing Symposium, IDAP 2024*. Institute of Electrical and Electronics Engineers Inc., 2024.

[109] K. Huang, R. Xian, M. Xian, H. Wang, and L. Ni, "A comprehensive intrusion detection method for the internet of vehicles based on federated learning architecture," *Computers and Security*, vol. 147, 12 2024.

[110] B. Kidmose and W. Meng, "can-fp: An attack-aware analysis of false alarms in automotive intrusion detection models," in *2024 21st Annual International Conference on Privacy, Security and Trust, PST 2024*. Institute of Electrical and Electronics Engineers Inc., 2024.

[111] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-iiotset: A new comprehensive realistic cyber security dataset of iot and iiot applications for centralized and federated learning," *IEEE Access*, vol. 10, pp. 40 281–40 306, 2022.

[112] ——, "Edge-iiotset: A new comprehensive realistic cyber security dataset of iot and iiot applications: Centralized and federated learning," 2022. [Online]. Available: https://dx.doi.org/10.21227/mbc1-1h68

[113] D. Trivedi, Shivani, T. Pandey, and Amrita, "Enhancing iiot security: A stacked intelligent ensemble model for intrusion detection," in *2025 International Conference on Networks and Cryptology, NETCRYPT 2025*. Institute of Electrical and Electronics Engineers Inc., 2025, pp. 1874–1879.

[114] K. Boakye-Boateng, A. A. Ghorbani, and A. H. Lashkari, "Securing substations with trust, risk posture, and multi-agent systems: A comprehensive approach," in *2023 20th Annual International Conference on Privacy, Security and Trust, PST 2023*. Institute of Electrical and Electronics Engineers Inc., 2023.

[115] M. Varol and M. İskefiyeli, "An intrusion detection system for critical infrastructures: Modbus approach," *Engineering Applications of Artificial Intelligence*, vol. 162, 12 2025.

[116] M. Fatahi, D. S. Zadeh, B. Moshiri, and O. Basir, "Entropy-based genetic feature engineering and multi-classifier fusion for anomaly detection in vehicle controller area networks," *Future Generation Computer Systems*, vol. 169, 8 2025.

[117] A. Kousar, S. Ahmed, A. Altamimi, and Z. A. Khan, "A novel light-weight machine learning classifier for intrusion detection in controller area network in smart cars," *Smart Cities*, vol. 7, pp. 3289–3314, 12 2024.

[118] M. Hanifa and H. T. Zubair, "Intrusion detection system for cyber-attacks in the internet of vehicles (iov) environment," in *1st International Conference on Cyber Security and Computing 2024, CyberComp 2024*. Institute of Electrical and Electronics Engineers Inc., 2024, pp. 68–73.

[119] E. Alalwany and I. Mahgoub, "An effective ensemble learning-based real-time intrusion detection scheme for an in-vehicle network," *Electronics (Switzerland)*,

vol. 13, 3 2024.

[120] L. Faramondi, F. Flammini, S. Guarino, and R. Setola, "A hardware-in-the-loop water distribution testbed dataset for cyber-physical security testing," *IEEE Access*, vol. 9, pp. 122 385–122 396, 2021.

[121] N. Müller, C. Ziras, and K. Heussen, "Assessment of cyber-physical intrusion detection and classification for industrial control systems," in *2022 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, 2022, pp. 432–438.

[122] M. Al-Hawawreh, E. Sitnikova, and N. Aboutorab, "X-iiotid: A connectivity- and device-agnostic intrusion dataset for industrial internet of things," 2021. [Online]. Available: https://dx.doi.org/10.21227/mpb6-py55

[123] ——, "X-iiotid: A connectivity-agnostic and device-agnostic intrusion data set for industrial internet of things," *IEEE Internet of Things Journal*, vol. 9, pp. 3962–3977, 3 2022.

[124] S. I. Popoola, Y. Tsado, A. A. Ogunjinmi, E. Sanchez-Velazquez, Y. Peng, and D. B. Rawat, "Multi-stage deep learning for intrusion detection in industrial internet of things," *IEEE Access*, vol. 13, pp. 60 532–60 555, 2025.

[125] M. Zolanvari, M. A. Teixeira, L. Gupta, K. M. Khan, and R. Jain, "Machine learning-based network vulnerability analysis of industrial internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6822–6834, 2019.

[126] ——, "WUSTL-IIoT-2021 Dataset for IIoT Cybersecurity Research," http://www.cse.wustl.edu/~jain/iiot2/index.html, St. Louis, MO, USA, Oct. 2021, dataset.

[127] L. Guerra, L. Xu, P. Bellavista, T. Chapuis, G. Duc, P. Mozharovskyi, and V. T. Nguyen, "Ai-driven intrusion detection systems (ids) on the road dataset: A comparative analysis for automotive controller area network (can)," in *CSCS 2024 - Proceedings of the 2024 Cyber Security in CarS Workshop, Co-Located with: CCS 2024*. Association for Computing Machinery, Inc, 11 2024, pp. 39–49.

[128] T. P. Nguyen, J. Cho, and D. Kim, "Semi-supervised intrusion detection system for in-vehicle networks based on variational autoencoder and adversarial reinforcement learning," *Knowledge-Based Systems*, vol. 304, 11 2024.

[129] T. D. Le, H. B. H. Truong, V. P. Pham, and D. Kim, "Multi-classification in-vehicle intrusion detection system using packet- and sequence-level characteristics from time-embedded transformer with autoencoder," *Knowledge-Based Systems*, vol. 299, 9 2024.

[130] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, and A. N. Anwar, "Ton-iot telemetry dataset: A new generation dataset of iot and iiot for data-driven intrusion detection systems," *IEEE Access*, vol. 8, pp. 165 130–165 150, 2020.

[131] U. Research, "The ton_iot datasets." [Online]. Available: https://research.unsw.edu.au/projects/toniot-datasets

[132] Z. Jadidi, S. Pal, K. N. Nayak, A. Selvakkumar, C. C. Chang, M. Beheshti, and A. Jolfaei, "Security of machine learning-based anomaly detection in cyber physical systems," in *Proceedings - International Conference on Computer Communications and Networks, ICCCN*, vol. 2022-July. Institute of Electrical and Electronics Engineers Inc., 2022.

[133] T. Saba, A. R. Khan, T. Sadad, and S. P. Hong, "Securing the iot system of smart city against cyber threats using deep learning," *Discrete Dynamics in Nature and Society*, vol. 2022, 2022.

[134] T. Gueye, Y. Wang, M. Rehman, R. T. Mushtaq, and S. Zahoor, "A novel method to detect cyber-attacks in iot/iiot devices on the modbus protocol using deep learning," *Cluster Computing*, vol. 26, pp. 2947–2973, 10 2023.

[135] I. Frazão, P. H. Abreu, T. Cruz, H. Araújo, and P. 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*), vol. 11260 LNCS, pp. 230–235, 2019. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-05849-4_19

[136] I. Frazão, P. Abreu, T. Cruz, H. Araújo, and P. Simões, "Cyber-security modbus ics dataset," 2019. [Online]. Available: https://dx.doi.org/10.21227/pjff-1a03

[137] K. Roopraj, N. Pillay, and N. Singh, "Intrusion detection in an industrial control system using unsupervised machine learning algorithms," in *2025 Annual IEEE Conference on Information Communication Technology and Society, ICTAS 2025 - Proceedings*. Institute of Electrical and Electronics Engineers Inc., 2025.

[138] Ángel Luis Perales Gómez, L. F. Maimó, A. H. Celdrán, F. lix J.Garc a. Clemente, C. C. Sarmiento, C. J. D. C. Masa, and R. n.M.ndez Nistal, "On the generation of anomaly detection datasets in industrial control systems," *IEEE Access*, vol. 7, pp. 177 460–177 473, 2019.

[139] B. O. Calviño, E. Rodriguez, J. J. Costa, and M. Oriol, "Enhancing cybersecurity in railways: Machine learning approaches for attack detection," *International Journal of Critical Infrastructure Protection*, vol. 50, 9 2025.

[140] M. K. Jha and R. Jaiswal, "A machine learning model to predict cyberattacks in connected and autonomous vehicles," *Journal of Computational and Cognitive Engineering*, vol. 3, pp. 307–315, 8 2024.

[141] R. Rai and J. Grover, "Comparative analysis of cosine and jaccard similarity-based classification for detecting can bus attacks," in *2024 IEEE Region 10 Sym-*

posium, TENSYMP 2024. Institute of Electrical and Electronics Engineers Inc., 2024.

[142] Y. K. Saheed and J. E. Chukwuere, "Xaiensembletl-iov: A new explainable artificial intelligence ensemble transfer learning for zero-day botnet attack detection in the internet of vehicles," *Results in Engineering*, vol. 24, 12 2024.

[143] H. Im, D. Lee, and S. Lee, "A novel architecture for an intrusion detection system utilizing cross-check filters for in-vehicle networks," *Sensors*, vol. 24, 5 2024.

[144] M. Althunayyan, A. Javed, and O. Rana, "A robust multi-stage intrusion detection system for in-vehicle network security using hierarchical federated learning," *Vehicular Communications*, vol. 49, 10 2024.

[145] M. Nazeer, A. Alasiry, M. Qayyum, V. K. Madhan, G. Patil, and P. Srilatha, "Enhancing cyber security in autonomous vehicles: A hybrid xg boost-deep learning approach for intrusion detection in the can bus," *Journal Europeen des Systemes Automatises*, vol. 57, pp. 1295–1304, 10 2024.

[146] Y. Zhang, C. Liu, Z. Xue, J. Pan, C. Lane, G. Han, and Y. Zeng, "Cdib-ids: Contextual distilbert-based intrusion detection system for iov," in *2024 IEEE 10th International Symposium on Microwave, Antenna, Propagation and EMC Technologies for Wireless Communications, MAPE 2024*. Institute of Electrical and Electronics Engineers Inc., 2024.

[147] Y. Wu and X. Tao, "Network traffic anomaly detection in can bus based on ensemble learning," in *2024 4th International Conference on Machine Learning and Intelligent Systems Engineering, MLISE 2024*. Institute of Electrical and Electronics Engineers Inc., 2024, pp. 240–245.

[148] O. Driouch, S. Bah, and Z. Guennoun, "Cansat-ids: An adaptive distributed intrusion detection system for satellites, based on combined classification of can traffic," *Computers and Security*, vol. 146, 11 2024.

[149] J. P. A. Bonomo, J. V. Volpato, R. S. D. Carvalho, and G. Gracioli, "Machine learning-based intrusion detection for automotive can networks on embedded platforms," in *Brazilian Symposium on Computing System Engineering, SBESC*. IEEE Computer Society, 2024.

[150] V. S. Barletta, D. Caivano, C. Catalano, and S. D. Vescovo, "Black-box adversarial ml attacks on ids and multi-domain impact analysis for threat intelligence in automotive scenarios," in *2024 IEEE International Workshop on Technologies for Defense and Security, TechDefense 2024 - Proceedings*. Institute of Electrical and Electronics Engineers Inc., 2024, pp. 132–137.

[151] D. Kim, Y. Song, S. Kwon, H. Kim, J. D. Yoo, and H. K. Kim, "Uavcan dataset description," 2024. [Online]. Available: https://arxiv.org/abs/2212.09268

[152] A. Mumrez, G. Sanchez, G. Elbez, and V. Hagenmeyer, "On evasion of machine learning-based intrusion detection in smart grids," in *2023 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, SmartGridComm 2023 - Proceedings*. IEEE Inc., 2023.

[153] I. Ozcelik, M. Iskefiyeli, M. Balta, K. O. Akpinar, and F. S. Toker, "Center water: A secure testbed infrastructure proposal for waste and potable water management," *9th International Symposium on Digital Forensics and Security, ISDFS 2021*, 6 2021.

[154] M. F. Elrawy, L. Hadjidemetriou, C. Laoudias, and M. K. Michael, "Detecting and classifying man-in-the-middle attacks in the private area network of smart grids," *Sustainable Energy, Grids and Networks*, vol. 36, 12 2023.

[155] J. Gao, L. Gan, F. Buschendorf, L. Zhang, H. Liu, P. Li, X. Dong, and T. Lu, "Omni scada intrusion detection using deep learning algorithms," *IEEE Internet of Things Journal*, vol. 8, pp. 951–961, 1 2021.

[156] M. Abdelkhalek, G. Ravikumar, and M. Govindarasu, "Ml-based anomaly detection system for der communication in smart grid," in *2022 IEEE Power and Energy Society Innovative Smart Grid Technologies Conference, ISGT 2022*. Institute of Electrical and Electronics Engineers Inc., 2022.

[157] Ágata Palma, M. Antunes, J. Bernardino, and A. Alves, "Multi-class intrusion detection in internet of vehicles: Optimizing machine learning models on imbalanced data," *Future Internet*, vol. 17, no. 4, 2025. [Online]. Available: https://www.mdpi.com/1999-5903/17/4/162

[158] Ágata Palma, M. Antunes, and A. Alves, "An optimized multi-class classification for industrial control systems," *Lecture Notes in Computer Science*, vol. 15937 LNCS, pp. 28–40, 2026. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-99565-1_3

[159] W. Wang and D. Sun, "The improved adaboost algorithms for imbalanced data classification," *Information Sciences*, vol. 563, pp. 358–374, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025521002875

[160] H. A. A. Rahman, Y. B. Wah, H. He, and A. Bulgiba, "Comparisons of adaboost, knn, svm and logistic regression in classification of imbalanced dataset," in *Soft Computing in Data Science*, M. W. Berry, A. Mohamed, and B. W. Yap, Eds. Singapore: Springer Singapore, 2015, pp. 54–64.

Instituto Superior de Engenharia
Politécnico de Coimbra