electronics

MDPI

Article

DataSense: A Real-Time Sensor-Based Benchmark Dataset for Attack Analysis in IIoT with Multi-Objective Feature Selection

Amir Firouzi *, Sajjad Dadkhah, Sebin Abraham Maret and Ali A. Ghorbani

Faculty of Computer Science, University of New Brunswick (UNB), Fredericton, NB E3B-5A3, Canada; sdadkhah@unb.ca (S.D.); sebin.maret@unb.ca (S.A.M.); ghorbani@unb.ca (A.A.G.)
* Correspondence: amir.firouzi@unb.ca

Abstract

The widespread integration of Internet-connected devices into industrial environments has enhanced connectivity and automation but has also increased the exposure of industrial cyber–physical systems to security threats. Detecting anomalies is essential for ensuring operational continuity and safeguarding critical assets, yet the dynamic, real-time nature of such data poses challenges for developing effective defenses. This paper introduces DataSense, a comprehensive dataset designed to advance security research in industrial networked environments. DataSense contains synchronized sensor and network stream data, capturing interactions among diverse industrial sensors, commonly used connected devices, and network equipment, enabling vulnerability studies across heterogeneous industrial setups. The dataset was generated through the controlled execution of 50 realistic attacks spanning seven major categories: reconnaissance, denial of service, distributed denial of service, web exploitation, man-in-the-middle, brute force, and malware. This process produced a balanced mix of benign and malicious traffic that reflects real-world conditions. To enhance its utility, we introduce an original feature selection approach that identifies features most relevant to improving detection rates while minimizing resource usage. Comprehensive experiments with a broad spectrum of machine learning and deep learning models validate the dataset's applicability, making DataSense a valuable resource for developing robust systems for detecting anomalies and preventing intrusions in real time within industrial environments.

Keywords: Internet of Things (IoT); Industrial Internet of Things (IIoT); dataset; security; IIoT security; intrusion detection system; anomaly detection; feature selection; feature engineering; machine learning; deep learning

Academic Editors: Gaoya Dong, Haoqiang Liu and Meijun Qu

Received: 3 September 2025
Revised: 2 October 2025
Accepted: 17 October 2025
Published: 19 October 2025

Citation: Firouzi, A.; Dadkhah, S.; Maret, S.A.; Ghorbani, A.A. DataSense: A Real-Time Sensor-Based Benchmark Dataset for Attack Analysis in IIoT with Multi-Objective Feature Selection. *Electronics* **2025**, *14*, 4095. https://doi.org/10.3390/electronics14204095

1. Introduction

The Internet of Things (IoT) has established itself as a transformative enabler in the modern interconnected world, enabling seamless communication among devices and driving advancements across numerous domains [1–4]. From smart homes to intelligent transportation systems, IoT enhances operational efficiency, reduces costs, and fosters new opportunities for data-driven innovation [5]. By facilitating the continuous exchange of vast amounts of data, IoT provides the foundation for smarter, more automated systems, reshaping industries and everyday life [2,4].

Building on these capabilities, the Industrial Internet of Things (IIoT) adapts and expands IoT technologies for industrial applications, serving as a cornerstone of modern industrial transformation. IIoT integrates advanced sensors, ultra-fast communication

infrastructures, and real-time data analytics to optimize processes, enhance safety, and enable data-driven decision-making across sectors such as manufacturing, energy, healthcare, transportation, and agriculture [6]. This interconnected industrial ecosystem generates vast amounts of heterogeneous data, which include both routine operational metrics and anomalous events. Such information is essential for ensuring system reliability, resilience, and overall performance [3,4,7].

In manufacturing, IIoT has transformed production lines through predictive maintenance systems that minimize equipment downtime and enable just-in-time production models, greatly improving efficiency and cost-effectiveness [2]. In the energy sector, IIoT supports the real-time monitoring of power grids, optimizes renewable energy integration, and enhances distribution through advanced load management and fault detection [1,5,8]. In healthcare, it enables continuous patient monitoring via smart medical devices, remote diagnostics, and connected healthcare systems, leading to improved patient outcomes and more efficient hospital operations [4,8]. The transportation sector benefits from intelligent fleet management, predictive vehicle maintenance, and real-time traffic optimization, which collectively enhance safety, reduce delays, and increase logistics efficiency [2]. In agriculture, precision farming technologies, including soil and crop monitoring, automated irrigation, and livestock tracking, make use of IIoT to increase yields and enhance resource utilization [1,4].

Despite these advances, the complexity and real-time nature of IIoT data introduce significant challenges for security and anomaly detection [5,8–10]. Traditional intrusion detection systems (IDSs), designed for conventional IT networks, are ill suited to address the heterogeneity and operational constraints of IIoT environments [10]. These systems encompass a wide spectrum of devices, ranging from resource-constrained sensors to high-capacity industrial machinery. Each operates with different capabilities, limitations, and communication protocols, some standardized and others proprietary [5,10,11]. The massive volume of real-time data generated via these devices demands anomaly detection systems capable of immediate processing and response [2,5,8]. Further complicating matters, IIoT devices often operate within interdependent, context-specific workflows, making it challenging for conventional intrusion detection systems to differentiate normal operational variations from actual security threats [12]. The diversity in device configurations, data representations, and communication structures further hinders the creation of universally applicable security solutions [2,4,5]. Moreover, stringent latency and reliability requirements render traditional offline or batch-processing approaches inadequate for timely threat detection [7,9]. Finally, the scarcity of comprehensive datasets that accurately represent the diversity and operational dynamics of real-world IIoT environments limits the training, validation, and benchmarking of modern IDS solutions [6,13]. Addressing these limitations requires novel methodologies and resources tailored to IIoT's unique demands, enabling resilient, real-time protection against evolving cyber threats.

This paper presents a novel IIoT dataset specifically designed to address the challenges of anomaly detection and security in industrial environments. A realistic testbed was developed, incorporating a diverse range of IIoT sensors for various industrial applications, alongside commonly used IoT devices and supporting network infrastructure. The testbed was carefully engineered to replicate real-world industrial conditions, capturing authentic interconnection patterns and device interactions that reflect the complexity and heterogeneity of modern IIoT ecosystems. To ensure comprehensive coverage, a series of controlled cyberattacks was executed across seven categories: reconnaissance (Recon), man-in-the-middle (MitM), denial of service (DoS), distributed denial of service (DDoS), web exploits, brute force, and malware. This process generated a rich blend of benign

and malicious traffic, producing a realistic dataset that is highly relevant to both academic research and industrial security development.

A novel feature selection approach is also proposed, focusing on identifying the features that contribute most substantially to improving detection accuracy while minimizing computational and resource overhead. Extensive experiments utilizing a diverse set of machine learning and deep learning methodologies demonstrate the dataset's adaptability and suitability for developing precise, efficient, and robust anomaly and intrusion detection systems in IIoT environments.

By releasing this dataset to the public, we aim to provide researchers and security practitioners with a comprehensive resource for designing, testing, and validating effective countermeasures. In doing so, we help close the gap between theoretical research and practical deployment while strengthening the security and resilience of industrial systems against emerging threats.

The key contributions of this work can be summarized as follows:

- Designed and implemented a realistic IIoT testbed incorporating diverse industrial sensors, IoT devices, and network infrastructure to replicate real-world industrial environments.
- Collected synchronized time-series sensor data and network traffic streams from all devices, producing a comprehensive dataset for in-depth security analysis.
- Executed and recorded 50 distinct attack types targeting network infrastructure, sensors, and devices, spanning multiple categories to capture diverse and realistic threat scenarios.
- Proposed and empirically validated an innovative multi-objective approach to feature selection for anomaly detection in IIoT environments, improving detection accuracy while reducing computational overhead by isolating the most relevant features from both sensor and network data streams.
- Developed a comprehensive real-time analysis benchmark integrating machine learning and deep learning methods for accurate detection and classification of cyberattacks in streaming data environments.
- Performed meticulous experiments to profile resource utilization of the proposed approaches under different operational scenarios, demonstrating the effectiveness of the feature selection method and comparing resource usage across multiple detection techniques.

The remainder of this paper is organized as follows. Section 2 reviews existing IoT and IIoT security datasets and related intrusion detection research. Section 3 presents the proposed DataSense dataset, detailing the experimental environment, testbed design, data collection process, and the generation of both benign and attack scenarios. Section 4 introduces the proposed hierarchical multi-stage information-driven feature selection framework, describing its methodology, components, and evaluation results. Section 5 reports the experimental setup and evaluation, including the performance of various machine learning and deep learning models on the dataset, as well as the effectiveness and resource efficiency of the proposed feature selection method. Finally, Section 7 concludes the paper and outlines potential directions for future work.

2. Related Work

There have been numerous efforts in generating datasets tailored to IIoT environments, each designed with unique purposes and characteristics to address specific research needs. These datasets play a vital role in advancing the understanding of IIoT systems, particularly in areas such as anomaly detection, security, and performance optimization. In this section, we review several prominent datasets utilized in the IIoT domain, highlighting their features, applications, and limitations in the context of industrial settings.

2.1. Intrusion and Anomaly Detection Datasets

Several notable datasets have been developed to address the scarcity of realistic data for intrusion detection in IoT and IIoT environments. X-IIoTID [14] presents an intrusion dataset that is both connectivity and device-agnostic, designed to capture the heterogeneity and interoperability of IIoT systems. It incorporates a wide range of connectivity protocols, up-to-date device activities, multiple attack categories, and multi-perspective features, including network traffic, host resource usage, and system logs. The CIC-IDS2017 dataset [15] provides realistic traffic patterns and various types of attacks, making it suitable for evaluating intrusion detection systems. CICIoT2023 [16] offers a large-scale IoT attack dataset generated from a real-world network comprising 105 IoT devices. The dataset covers seven attack categories, namely reconnaissance, web-based, brute force, spoofing, denial of service, distributed denial of service, and Mirai, with a total of 33 distinct attack scenarios. This dataset serves as a valuable resource for the development of advanced security analytics tailored to realistic IoT environments. Edge-IIoTset [17] is a comprehensive IoT/IIoT cybersecurity dataset generated from a diverse, purpose-built testbed with multiple devices, sensors, protocols, and edge/cloud configurations, encompassing fourteen realistic attack types across five major threat categories, with extensive feature extraction to support both centralized and federated machine learning–based intrusion detection research.

The NSL-KDD dataset [18], a refined successor to the KDD Cup 99 dataset, removes redundant records to provide a more reliable benchmark for evaluating anomaly detection algorithms. UNSW-NB15 [19] contains modern attack types and diverse network traffic, supporting intrusion detection studies. ToN-IoT [20] integrates IoT and IIoT device traffic with both benign and malicious data, ideal for hybrid network security research. MQTTset [21] is centered on the MQTT protocol, a broadly adopted messaging standard in IoT settings, and combines legitimate traffic with cyberattacks to facilitate the training of detection models for MQTT-based environments. Similarly, MQTT-IoT-IDS [22] presents an intrusion detection dataset for MQTT networks, incorporating seven types of MQTT attacks, including memory corruption, and evaluating both centralized and federated learning approaches on large-scale attack traces.

The DARPA dataset [23], one of the earliest benchmarks, captures simulated network intrusions in a controlled environment. CICAPT-IIoT [24] provides IIoT-specific network traffic data, including several real-world attack scenarios. WUSTL-IIoT [25] includes IIoT sensor and network data for performance and security analysis. Lastly, DAPT2020 [26] combines industrial device logs with network traffic to simulate IIoT threats.

These datasets provide diverse features, data types, and attack scenarios, offering valuable resources for intrusion and anomaly detection research in IIoT environments.

2.2. Malware Detection Datasets

The IoT-23 dataset [27] comprises an extensive set of labeled IoT network traffic, encompassing both benign and malicious activities. It focuses on capturing realistic IoT traffic patterns and a variety of attack scenarios, such as malware infections and command-and-control communications, making it a valuable resource for intrusion and anomaly detection research in IoT and IIoT environments. N-BaIoT [28] targets botnet detection in IoT systems by capturing behavior-based anomalies in IoT networks from nine IoT devices infected with Mirai and BASHLITE malware, demonstrating the capability of deep autoencoders to detect anomalous traffic in near-real time. Similarly, Bot-IoT [29] presents a large-scale and realistic IoT botnet dataset generated in a controlled testbed, combining legitimate and malicious traffic to address prior limitations in network-capture

completeness, labeling accuracy, and attack diversity for network forensic and intrusion detection research.

2.3. Industrial Systems

Many IIoT datasets target specific industrial applications, typically falling into four categories: water treatment systems, predictive maintenance and fault detection, gas systems, and environmental monitoring. Each category addresses domain-specific challenges and supports focused research and development.

2.3.1. Water Treatment Systems

Water treatment system datasets are crucial for addressing challenges in monitoring, controlling, and securing industrial water processes. The SWaT (Secure Water Treatment) dataset [30] is derived from a scaled-down water treatment testbed and includes both normal operation and attack scenarios, making it widely used for anomaly detection research. Similarly, the WADI (Water Distribution) dataset [31], derived from a controlled water distribution testbed, provides sensor and actuator data under normal and attack conditions, supporting fault detection and security studies in water distribution networks. The C-MAPSS (Commercial Modular Aero-Propulsion System Simulation) dataset [32], though primarily focused on engine simulation, is often adapted for fault detection research in water systems due to its relevance in predictive maintenance applications. Additionally, the SMD (Server Machine Dataset) [33], while not exclusive to water systems, offers valuable time-series data for anomaly detection when applied to water treatment contexts. These datasets collectively provide robust resources for research on security, fault detection, and predictive maintenance in water treatment environments.

2.3.2. Predictive Maintenance and Fault Detection

Datasets for predictive maintenance and fault detection are essential for improving the reliability and efficiency of industrial systems by facilitating the early identification of potential failures. The SECOM dataset [34] contains sensor data from a semiconductor manufacturing process and is widely used for fault detection and quality control studies. The APS (Air Pressure System) dataset [35] focuses on predicting failures in air pressure systems of heavy vehicles, offering valuable insights for maintenance and operational optimization. The MIMII (Malfunctioning Industrial Machine Investigation and Inspection) dataset [36] includes acoustic and operational data from industrial machines under both normal and malfunctioning conditions, making it a critical resource for developing fault detection and predictive maintenance models. These datasets provide comprehensive benchmarks for advancing research in industrial system reliability and fault diagnosis.

2.3.3. Gas Systems

Gas-system datasets are essential for monitoring and ensuring the safety and efficiency of industrial gas operations. The Gas Sensor Array Dataset [37] provides data collected from an array of chemical sensors exposed to various gas mixtures under controlled conditions. This dataset is widely used for developing models to detect gas leaks, identify gas types, and monitor air quality, making it a valuable resource for research in industrial safety and environmental monitoring.

2.3.4. Environmental Monitoring

Environmental monitoring datasets are designed to support research on weather conditions, energy consumption, and other factors impacting industrial and environmental efficiency. The Intel Berkeley Lab Dataset [38] contains sensor data collected for temperature, humidity, and light monitoring in an indoor environment, enabling studies on environ-

mental control and system optimization. The PLAID (Plug Load Appliance Identification Dataset) [39] provides detailed energy consumption data from various household and industrial appliances, supporting research on energy efficiency and appliance identification. These datasets contribute significantly to the development of solutions for sustainable environmental and energy management.

2.4. Time Series Analysis

Time series analysis datasets are critical for understanding and modeling sequential data across various applications, including industrial systems and health monitoring. The PSM (Predictive System Maintenance) dataset [40] provides sensor time-series data from industrial machines, supporting research in fault detection and predictive maintenance. The Rare Event Dataset [41] focuses on identifying anomalies in time-series data with infrequent but critical events, enabling robust anomaly detection model development. The Statlog (Shuttle) dataset [42] offers time-series data from a space shuttle simulation, commonly used for classifying anomalies in highly dynamic systems. The ECG5000 dataset [43], designed for health applications, contains time-series data of electrocardiogram signals, supporting research on detecting heart conditions and abnormalities. These datasets provide valuable resources for advancing time-series analysis in diverse domains.

2.5. Research Gap

Despite extensive research on IIoT anomaly detection and the availability of several public datasets, important gaps remain. Existing benchmarks often face limitations such as narrow device diversity, limited attack coverage, and the absence of synchronized network and sensor data, which reduces their applicability to real-world industrial settings. In addition, many datasets do not adequately assess resource efficiency, including memory footprint, inference time, and model size. These factors are critical for real-time deployment on resource-constrained IIoT edge devices. Finally, while prior studies have explored feature selection and ML/DL models, limited attention has been given to systematically comparing traditional, deep, and hybrid learning approaches under a unified experimental setup. To the best of our knowledge, no existing dataset simultaneously addresses these limitations. This gap motivates the development of the DataSense dataset, which provides heterogeneous device coverage, diverse attack scenarios, synchronized data streams, and comprehensive evaluation benchmarks tailored for IIoT security research.

3. The Proposed DataSense Dataset

This section presents a detailed overview of the proposed DataSense dataset, detailing the steps and resources involved in its generation. It covers the design and structure of the testbed, the types of IIoT and IoT devices included, and the processes for capturing data. Additionally, the section highlights the dataset's characteristics, such as its diversity, realism, and applicability for anomaly detection and security research in industrial environments.

3.1. IoT and IIoT Lab

Developing a realistic Industrial Internet of Things (IIoT) testbed poses substantial challenges due to the inherent complexity, heterogeneity, and resource-intensive requirements of industrial settings. IIoT ecosystems encompass a wide array of devices, from low-power, resource-limited sensors to high-performance industrial machines, operating over diverse communication protocols. Faithfully replicating such a heterogeneous environment requires significant financial and technical resources, including specialized hardware, software, and domain-specific expertise. As a result, many existing studies depend on simulations or a small subset of devices. Although these approaches are often more scalable, they are unable to capture the nuanced behaviors and operational intricacies

of real-world IIoT deployments. To address these limitations, the Canadian Institute for Cybersecurity (CIC) has established a dedicated IoT and IIoT research laboratory, enabling the creation of realistic testbeds that incorporate diverse devices within a sophisticated network infrastructure.

Figure 1 illustrates the IoT and IIoT Lab at the CIC, showcasing the advanced setup designed for research and experimentation. The lab features a diverse range of devices and a sophisticated network infrastructure, enabling the creation of realistic testbeds that replicate real-world industrial environments.

Figure 1. Canadian Institute for Cybersecurity (CIC) IoT and IIoT Lab.

3.2. IIoT Testbed

The overall architecture of the proposed IIoT testbed, illustrated in Figure 2, comprises five primary components: the IIoT and IoT Layer, Network Infrastructure, the Edge Layer, the Cloud Layer, and the Attacker Layer. These interconnected components are designed to emulate a realistic and heterogeneous IIoT environment for comprehensive security research and system evaluation.

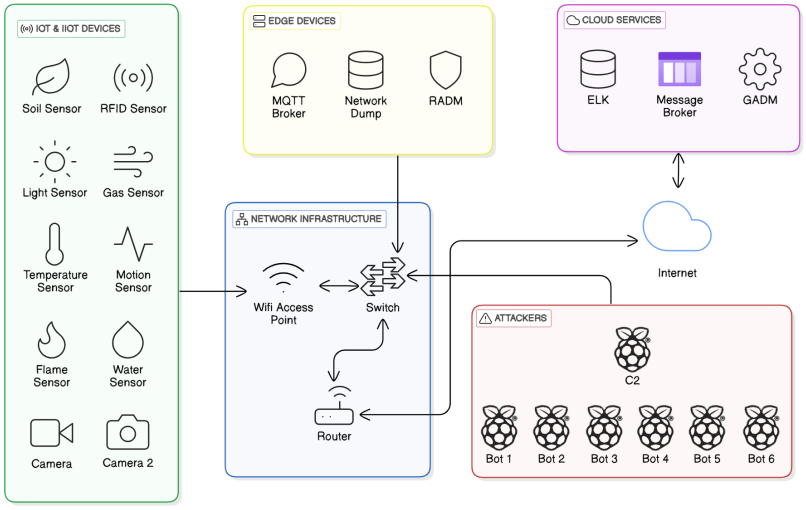

Figure 2. DataSense testbed overall architecture.

3.2.1. IIoT and IoT Layer

This layer consists of a wide range of industrial sensors integrated with Arduino boards, each configured with one or more sensors tailored to specific application domains. The variety of sensors reflects the heterogeneous nature of real-world industrial environments. In addition, several commercial IoT devices, such as surveillance cameras, security cameras, and smart plugs, are connected to the network via Wi-Fi. All IIoT and IoT devices

transmit real-time data through a dual-band Wi-Fi access point to an MQTT broker. The broker aggregates the data and forwards it to the storage and analysis systems, ensuring the seamless integration of different device types within the testbed. A comprehensive overview of all testbed devices is provided in Table 1, while the specific IIoT sensors are detailed in Table 2.

Table 1. List of all devices in the testbed.

Device Name	Category	Role	MAC Address	IP
TP-Link Router	Network	Router	28:87:BA:BD:C6:6C	192.168.1.1
Netgear Switch	Network	Switch	E0:46:EE:21:56:18	192.168.1.200
IIoT AP	Network	AP	30:DE:4B:E2:13:4E	192.168.1.205
MQTT Broker	Raspberry Pie	MQTT Broker	DC:A6:32:DC:28:46	192.168.1.193
Edge Device	Raspberry Pie	Edge Device	DC:A6:32:DC:27:D4	192.168.1.195
IIoT Laptop	Laptop	Capturer	E4:B9:7A:21:B2:F0	192.168.1.210
Weather Sensor	Sensor	Sensor	08:B6:1F:82:12:30	192.168.1.10
Water Sensor	Sensor	Sensor	08:B6:1F:84:66:78	192.168.1.11
Soil Sensor	Sensor	Sensor	F0:08:D1:CE:CF:0C	192.168.1.12
Steam Sensor	Sensor	Sensor	08:B6:1F:81:D2:CC	192.168.1.13
Gas Sensor	Sensor	Sensor	08:B6:1F:83:25:98	192.168.1.14
Sound Sensor	Sensor	Sensor	F0:08:D1:CE:CF:C8	192.168.1.15
Vibration Sensor	Sensor	Sensor	08:B6:1F:82:27:D0	192.168.1.16
Ultrasonic Sensor	Sensor	Sensor	08:B6:1F:82:EE:C4	192.168.1.17
Light Sensor	Sensor	Sensor	8C:AA:B5:8A:A9:B4	192.168.1.18
Accelerometer Sensor	Sensor	Sensor	08:B6:1F:82:EE:44	192.168.1.19
Proximity Collision	Sensor	Sensor	08:B6:1F:82:EF:30	192.168.1.20
Motion Sensor	Sensor	Sensor	08:B6:1F:82:1C:3C	192.168.1.21
RFID Sensor	Sensor	Sensor	08:B6:1F:82:2B:1C	192.168.1.22
Flame Sensor	Sensor	Sensor	08:B6:1F:82:EE:CC	192.168.1.23
Yi Camera	Camera	Camera	7C:94:9F:84:71:7E	192.168.1.50
Blurams Camera	Camera	Camera	14:C9:CF:45:3E:BA	192.168.1.52
Dekco Camera	Camera	Camera	44:29:1E:5C:DE:12	192.168.1.53
Liftmaster Camera	Camera	Camera	20:50:E7:F0:0A:04	192.168.1.54
Geeni Camera	Camera	Camera	DC:29:19:95:3A:79	192.168.1.55
Wisenet Camera	Camera	Camera	00:09:18:6D:73:B9	192.168.1.57
Plug All Cameras	Smart Plug	Smart Plug	C4:DD:57:15:5C:2C	192.168.1.80
Plug Laptop	Smart Plug	Smart Plug	C4:DD:57:0D:F2:76	192.168.1.81
Plug Mqtt	Smart Plug	Smart Plug	D4:A6:51:1F:F6:7C	192.168.1.82
Plug RFID	Smart Plug	Smart Plug	D4:A6:51:1D:C0:ED	192.168.1.83
Plug Edge	Smart Plug	Smart Plug	D4:A6:51:22:03:99	192.168.1.84
Plug Motion	Smart Plug	Smart Plug	D4:A6:51:1D:74:3A	192.168.1.85
Plug Flame	Smart Plug	Smart Plug	D4:A6:51:20:91:F7	192.168.1.86
Plug Proximity	Smart Plug	Smart Plug	D4:A6:51:20:0E:3F	192.168.1.87
Plug Vibration	Smart Plug	Smart Plug	D4:A6:51:79:68:75	192.168.1.88
Plug Cameras1 Yi	Smart Plug	Smart Plug	50:02:91:10:AC:D8	192.168.1.90
Plug Cameras2 Geeni	Smart Plug	Smart Plug	50:02:91:10:09:8F	192.168.1.91
Plug Cameras3 Dekco	Smart Plug	Smart Plug	50:02:91:11:05:8C	192.168.1.92
Plug All Sensors	Smart Plug	Smart Plug	D4:A6:51:82:98:A8	192.168.1.93
Attacker0	Raspberry Pie	Attacker (C2)	E4:5F:01:55:90:C1	192.168.1.100
Attacker1	Raspberry Pie	Attacker (Bot)	DC:A6:32:C9:E6:F3	192.168.1.101
Attacker2	Raspberry Pie	Attacker (Bot)	DC:A6:32:C9:E5:A3	192.168.1.102
Attacker3	Raspberry Pie	Attacker (Bot)	DC:A6:32:C9:E4:C5	192.168.1.103
Attacker4	Raspberry Pie	Attacker (Bot)	DC:A6:32:C9:E5:01	192.168.1.104
Attacker5	Raspberry Pie	Attacker (Bot)	DC:A6:32:C9:E4:AA	192.168.1.105

Table 2. List of all sensors in the testbed.

Application	IIoT Sensors	Sensor Types
Sound	Big Sound Detector	KY-037
	Small Sound Detector	KY-038
Weather	Temperature & Humidity	DHT11
	Linear temperature	LM35
	Analog temperature	KY-013
	Digital temperature	DS18B20
	Atmospheric pressure	BMP-180
Soil Moisture	Soil Moisture	YL-69
Motion	PIR Motion Sensor	HC-SR501
Vibration	Ceramic Vibration Sensor	SW-420
Water	Water Level Sensor	YL-83
Steam	Steam Sensor	KS0203
RFID	RFID Sensor	RFID-RC522
Accelerometer Gyroscope	Triaxial Digital Acceleration Tilt Sensor	ADXL345
Proximity	ALS Infrared LED Optical Proximity	APDS-9930
Collision	Collision (Crash Sensor)	KY-031
Ultrasonic	Ultrasonic Sensor	HC-SR04
Flame	Flame Detector	KY-026
Light Gesture	Light & Gesture Detection Sensor	APDS-9960
Gas	Analog gas detector	MQ-2
	Analog Alcohol detector	MQ-3

3.2.2. Network Infrastructure

The network backbone is established through a managed switch, which interconnects all key components and supports local communication. A TP-Link router connects the testbed to the internet, while a dual-band Wi-Fi access point enables wireless connectivity for IIoT and IoT devices. To facilitate detailed traffic monitoring, port mirroring is configured on the managed switch. All traffic, including data from the access point, router, and attacker nodes, is mirrored to a monitoring port that is connected to a Kali Linux laptop running Wireshark and TShark. This setup ensures comprehensive packet-level visibility for traffic analysis and network forensics. The complete list of network infrastructure devices and other devices' hardware and operating system descriptions is provided in Table 3.

Table 3. Testbed hardware and operating system specifications.

Device	Hardware Model	OS	Description
IIoT Sensors	Arduino MKR WiFi 1010 Arduino Ethernet	firmware	Arduino Board
Access Point	TP-Link Omada AC1750	firmware	Gigabit Wireless AP
Switch	Netgear GS316EP Switch	firmware	16-Port PoE Gigabit Eth Switch
Router	TP-Link AX1800 Router	firmware	WiFi Smart Router
Edge Devices	Raspberry Pi B 4 8GB Raspberry Pi B 4 8GB	Kali	MQTT Broker Web Server, Network Dump
Cloud	CloudPowerEdge R530 Server	Centos 9	Elasticsearch Message Broker
Attackers	1xRaspberry Pi B 4 4GB 5xRaspberry Pi B 4 2GB	Kali Raspbian	Attacker C2 Attacker Bots

3.2.3. Edge Layer

This layer is implemented using Raspberry Pi units, which serve as edge computing nodes directly connected to the managed switch. Each unit hosts an MQTT broker responsible for receiving and aggregating sensor data. In addition, a network dump module based

on TShark captures real-time traffic from all connected devices. A Rapid Anomaly Detection Module (RADM) is deployed at the edge to perform real-time analysis for the early detection of potential attacks, thereby enhancing responsiveness and reducing detection latency. Furthermore, a dedicated web server is hosted on one of the edge devices to expose RESTful APIs in order to monitor network traffic and share detection metrics and extracted rules. This facilitates the seamless integration of monitoring and analytical functionalities within the edge environment.

3.2.4. Cloud Layer

Positioned outside the local IIoT network, the cloud layer provides high-performance computing resources for advanced data processing. It is hosted on a Dell PowerEdge R530 server, which accommodates a robust Elasticsearch cluster used for scalable storage and in-depth analysis of the aggregated sensor data. A Kafka message broker operates within this layer to handle preprocessing, filtering, and the grouping of real-time data streams. In addition, the Global Anomaly Detection Module (GADM) is deployed in the cloud to execute computationally intensive tasks such as feature extraction, attack classification, and the detection of complex multi-stage attack patterns.

3.2.5. Attacker Layer

To simulate real-world threat scenarios, the testbed includes a collection of attacker devices implemented using Raspberry Pi units running both Kali Linux and Raspbian OS. These devices emulate various cyber threats, including reconnaissance, denial-of-service (DoS), distributed denial-of-service (DDoS), web-based attacks, brute-force intrusions, and malware propagation. Configured as a coordinated attacker army, these nodes represent both command-and-control (C2C) centers and bots, enabling robust experimentation with diverse attack strategies and defense mechanisms.

This comprehensive architecture provides a flexible and scalable platform for realistic IIoT experimentation, facilitating the development, deployment, and evaluation of advanced security solutions in dynamic and heterogeneous industrial environments.

3.3. Data Collection

The data collection process in the proposed testbed was designed to capture both network-level and application-level (log) data under benign and attack conditions, thereby enabling the development of a comprehensive and realistic dataset. IIoT and IoT devices such as Arduino-integrated industrial sensors, surveillance cameras, smart plugs, and other consumer devices (see Tables 1 and 2) generated continuous telemetry and operational traffic. This traffic was transmitted through a dual-band Wi-Fi access point to a managed switch, as detailed in Table 3. The MQTT broker, implemented using Eclipse Mosquitto [44] and hosted on a Raspberry Pi edge device, served as the central aggregator for IIoT sensor data. To support log-level data collection, the Filebeat service [45] was deployed on the MQTT broker, where it continuously harvested MQTT logs and forwarded them to an Elasticsearch cluster hosted on a Dell PowerEdge R530 server in the cloud. This configuration ensured structured indexing and the scalable storage of application-level logs for further analysis.

Simultaneously, network-level data was captured by enabling port mirroring on the managed switch, which directed all incoming and outgoing traffic from the access point, router, and attacker devices to a monitoring port connected to a Kali Linux laptop. TShark [46] was used for automated, continuous packet capture, while Wireshark [47] supported detailed post-capture traffic analysis. To ensure precise temporal correlation between network traffic and telemetry data, all devices in the testbed were synchronized to a common timezone, allowing unique and consistent timestamps. This synchronization

enabled the accurate mapping of captured packet data (PCAPs) to corresponding log entries, facilitating integrated analysis of system behavior. In benign scenarios, IIoT and IoT devices operated in normal modes, transmitting routine sensor readings and service traffic. In attack scenarios, Raspberry Pi-based attacker devices running Kali Linux and Raspbian OS simulated a wide range of cyber threats, including reconnaissance, DoS, DDoS, brute force, malware, and web-based attacks. These nodes acted as both command-and-control servers and distributed bots, generating diverse threat behaviors. All captured log and network data were archived in parallel, ensuring a synchronized and high-fidelity representation of both benign and malicious activity within the testbed.

Figure 3 presents a comparative analysis of the number of packets generated across different attack scenarios in the dataset, highlighting the variability in network traffic intensity for each type of attack. As illustrated, distributed denial of service (DDoS) attacks generate the highest packet volumes, followed by denial of service (DoS) attacks, while reconnaissance, web-based, and brute-force attacks produce significantly fewer packets. It is also worth noting that not all DoS or DDoS attacks result in high traffic; for instance, Slowloris, MQTT Publish Flood, and MQTT Connect Flood typically generate relatively low packet counts. This distinction is important, as high packet volumes (e.g., high PPS floods) can overwhelm network devices and exhaust resources, leading to disruption or denial of service, whereas low packet volumes are often used in low-rate or stealthy attacks designed to mimic legitimate traffic, bypass intrusion detection systems, and exploit vulnerabilities without triggering alarms. By including both high- and low-packet-count attack scenarios, the dataset enables researchers to study a wide spectrum of attack behaviors, from overt floods to subtle evasion techniques.

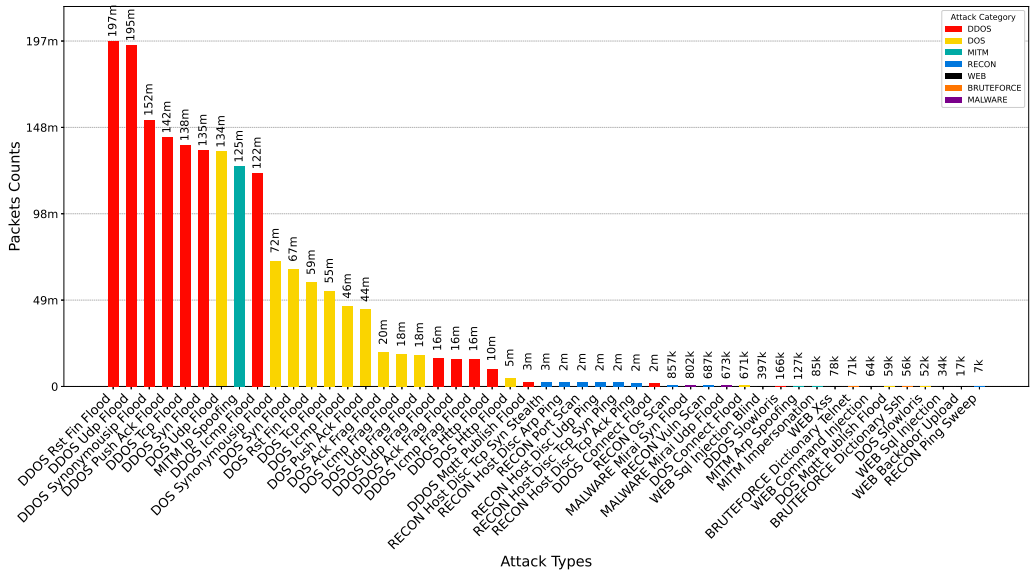

Figure 3. Number of packets for each attack scenario.

Figure 4 illustrates the distribution of sensor log entries recorded for each attack scenario, providing insights into the impact of different attacks on device functionality. As shown, the majority of log data is associated with reconnaissance attacks, indicating that such scenarios typically do not disrupt the normal operation of networked devices. This contrasts with denial of service (DoS) and distributed denial of service (DDoS) attacks, with which interference, such as jamming or resource exhaustion, can significantly reduce the number of sensor logs generated by targeted devices. This observation highlights the disruptive nature of DoS and DDoS attacks in comparison to the more passive behavior of reconnaissance activities.

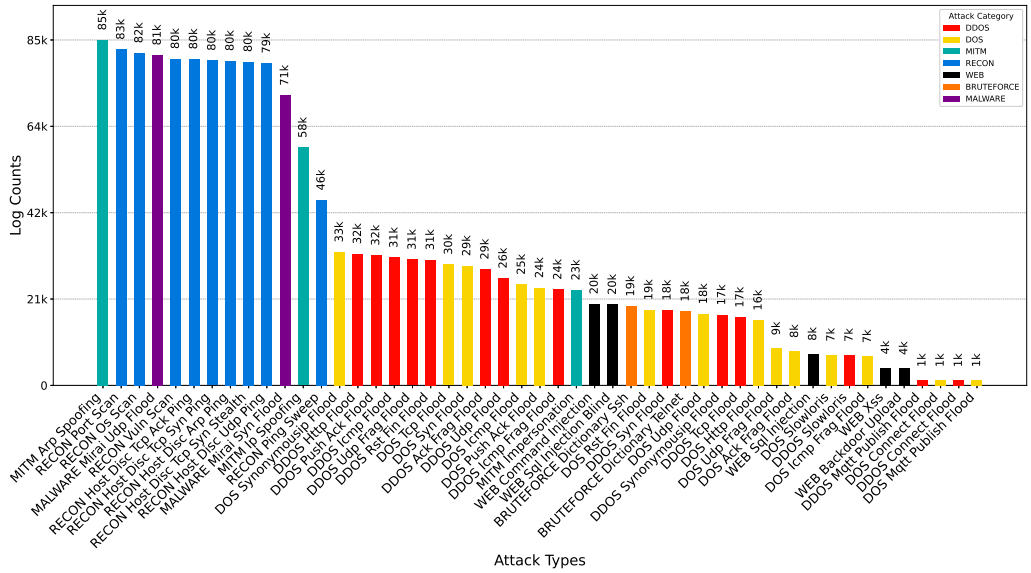

Figure 4. Number of logs for each attack scenario.

Figures 5 and 6 provide a high-level overview of the distribution of network packets and sensor log data across different attack categories. Figure 5 demonstrates that the DDoS and DoS categories collectively account for the largest volume of network packets, reflecting the inherently high-traffic nature of these attack types. In contrast, Figure 6 shows that the reconnaissance category generates the majority of sensor log entries. This discrepancy underscores the operational impact of each attack category: while DoS and DDoS attacks flood the network with excessive traffic, often impairing device communication, reconnaissance attacks tend to be non-intrusive and allow for continued device functionality, resulting in sustained log data generation.

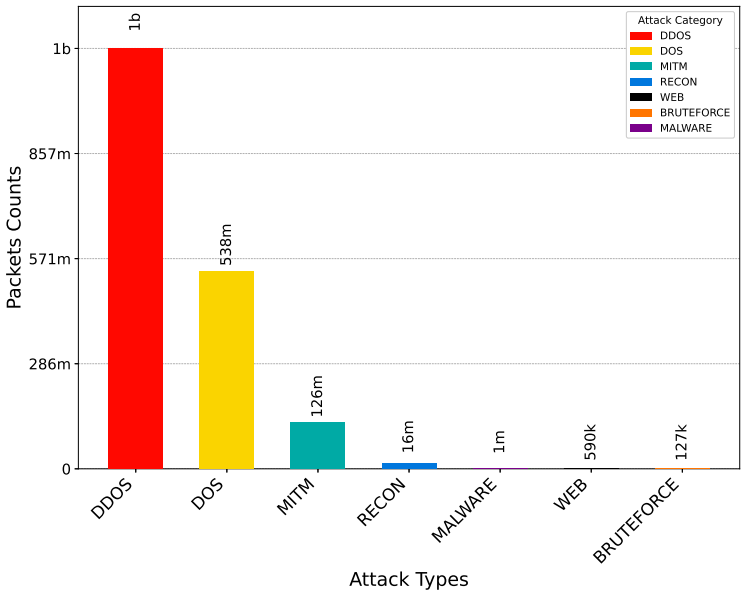

Figure 5. Number of packets for each category.

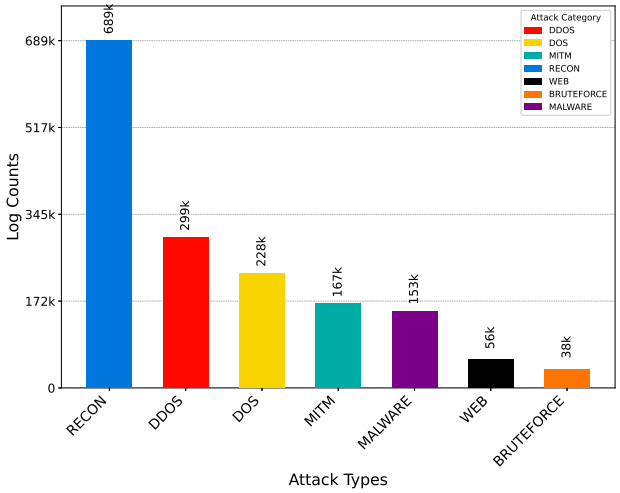

Figure 6. Number of logs for each category.

3.4. Benign Data Generation

The data collection process in the proposed IIoT testbed was designed to capture both network-level and application-level data under benign and attack scenarios. For benign data, IIoT and IoT devices such as industrial sensors, smart cameras, smart plugs, and other consumer electronics operated under normal conditions without interference or simulated threats. These devices continuously generated routine telemetry and service traffic over the testbed infrastructure, which included a dual-band Wi-Fi access point, a managed switch, and a centralized MQTT broker hosted on a Raspberry Pi.

To capture this data, MQTT logs were collected using Filebeat and stored in an Elasticsearch cluster, while network traffic was monitored through port mirroring and captured using TShark on a dedicated monitoring system. All components were time-synchronized to ensure accurate correlation between network and log data.

A total of 12 h of benign data was recorded, encompassing typical activity patterns during and after work hours to reflect realistic device behavior. For evaluation purposes, a one-hour subset was randomly selected, with 5 min allocated for device profiling. This subset was segmented into 10 s non-overlapping slices, excluding the profiling data from the evaluation set to preserve its integrity. The total number of network packets and logs collected for benign data is shown in Table 4.

Table 4. Summary of attack and benign data types, categories, data volumes, and tools used in the DataSense IIoT testbed.

Category	Benign/Attack Type	# Packets	# Logs	Tools
Benign	Benign Data	259,212	72,554	tshark [46]; filebeat [45]
DDoS	Ack Fragmentation Flood	15,895,816	28,560	hping3 [48]
	Connect Flood	1,937,806	1241	mqtt-connect-flood
	HTTP Flood	10,128,404	32,359	golang-httpflood [49]
	ICMP Flood	121,646,915	31,422	hping3 [48]
	ICMP Fragmentation Flood	15,715,035	23,594	hping3 [48]
	MQTT Publish Flood	2,732,308	1249	mqtt-benchmark [50]
	PSHACK Flood	142,102,006	31,994	hping3 [48]
	RSTFIN Flood	196,936,383	30,866	hping3 [48]
	Slowloris	166,168	7444	slowloris [51]
	TCP SYN Flood	134,658,233	18,493	hping3 [48]
	Synonymous IP Flood	152,046,341	17,315	hping3 [48]
	TCP Flood	137,626,631	16,744	hping3 [48]
	UDP Flood	194,623,658	26,354	hping3 [48]; udp-flood [52]
	UDP Fragmentation Flood	15,973,062	30,941	udp-flood [52]
	Total	1,142,218,766	298,576	

Table 4. *Cont.*

Category	Benign/Attack Type	# Packets	# Logs	Tools
DoS	Ack Fragmentation Flood	19,550,570	8414	udp-flood [52]
	Connect Flood	671,129	1248	mqtt-connect-flood
	HTTP Flood	4,899,179	16,021	golang-httpflood [49]
	ICMP Flood	45,789,955	24,824	hping3 [48]
	ICMP Fragmentation Flood	18,340,110	7127	hping3 [48]
	MQTT Publish Flood	58,739	1240	mqtt-benchmark [50]
	PSHACK Flood	43,897,855	24,015	hping3 [48]
	RSTFIN Flood	59,372,231	18,573	hping3 [48]
	Slowloris	51,603	7492	slowloris [51]
	TCP SYN Flood	66,872,061	29,405	hping3 [48]
	Synonymous IP Flood	71,659,415	32,791	hping3 [48]
	TCP Flood	54,515,535	29,738	hping3 [48]
	UDP Flood	133,945,490	17,594	hping3 [48]; udp-flood [52]
	UDP Fragmentation Flood	17,897,278	9071	udp-flood [52]
	Total	537,520,150	227,553	
Recon	Host Discovery ARP Ping	2,463,445	79,831	nmap [53]
	Host Discovery TCP ACK Ping	2,107,543	80,257	nmap [53]
	Host Discovery TCP SYN Ping	2,318,477	79,901	nmap [53]
	Host Discovery TCP SYN Stealth	2,600,465	79,580	nmap [53]
	Host Discovery UDP Ping	2,329,836	79,206	nmap [53]
	OS Scan	857,076	81,762	nmap [53]
	Ping Sweep	7474	45,556	nmap; fping [54]
	Port Scan	2,418,517	82,613	nmap [53]
	Vulnerability Scan	687,382	80,335	nmap; vulscan [55]
	Total	15,790,215	689,041	
Web	Backdoor Upload	16,606	4215	Remot3d [56]
	Command Injection	64,155	20,028	payloadbox [57]
	SQL Injection	33,727	7723	sqlmap [58]; payloadbox [59]
	Blind SQL Injection	397,486	19,907	sqlmap [58]; payloadbox [59]
	Cross Site Scripting	77,984	4369	payloadbox [60]; XSStrike [61]
	Total	589,958	56,242	
Bruteforce	SSH Bruteforce	55,793	19,470	thc-hydra [62]; SecLists [63]
	Telnet Bruteforce	71,025	18,205	thc-hydra [62]; SecLists [63]
	Total	126,818	37,675	
MITM	ARP Spoofing	126,759	84,927	ettercap [64]
	Impersonation	84,967	23,431	mqtt-benchmark [50]
	IP Spoofing	125,449,545	58,473	hping3 [48]
	Total	125,661,271	166,831	
Mirai	Syn Flood	801,628	71,308	Mirai Source Code [65]
	UDP Flood	672,801	81,226	Mirai Source Code [65]
	Total	1,474,429	152,534	
Total		**1,823,381,607**	**1,628,452**	

3.5. Attack Data Generation

To simulate realistic threat scenarios within Industrial Internet of Things (IIoT) environments, a total of 49 distinct attack types, spanning seven different categories, were executed on various components of the testbed, as well as the entire network infrastructure. During these attack simulations, both network traffic and application-level data (specifically sensor data) were comprehensively captured across the testbed. The collected data were systematically organized, with packet capture (PCAP) files segmented according to the targeted components of each attack. This structuring facilitates easy access for researchers, enabling them to isolate and analyze the network and sensor data corresponding to specific attack scenarios, along with the concurrent activity of other devices during the same time frame. The total number of network packets and logs collected for each attack category and attack scenario is shown in Table 4. Detailed descriptions of each executed attack type are provided in the subsequent subsections.

3.5.1. Execution of DoS and DDoS Attacks

Denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks are executed to overwhelm and disrupt the availability of Industrial IoT (IIoT) services. In DoS scenarios, a single Raspberry Pi generates high volumes of malicious traffic targeting specific IoT devices. In contrast, DDoS attacks employ multiple Raspberry Pis operating in a coordinated master–client architecture via SSH-based communication, thereby amplifying the scale and impact of the disruption.

The executed attack types are as follows:

- **ACK Fragmentation:** Uses a limited number of maximum-sized packets to impair network performance. These fragmented packets are often processed via network components such as routers and firewalls, as they typically do not perform packet reassembly.
- **Slowloris:** An application-layer DoS attack that keeps many connections to the target web server open by sending partial HTTP requests, thereby exhausting the server's resources.
- **TCP/UDP/ICMP/HTTP Flood:** Involves overwhelming the target device with excessive volumes of different protocol packets to exhaust its processing capacity.
- **RST-FIN Flood:** Continuously sends TCP packets with the RST and FIN flags to degrade network performance by forcing connection terminations.
- **PSH-ACK Flood:** Targets server performance by sending floods of TCP packets with PSH and ACK flags, disrupting normal communication.
- **UDP Fragmentation:** A variation of UDP flooding that utilizes fragmented packets to consume more bandwidth with fewer packets.
- **ICMP Fragmentation:** Employs fragmented IP packets containing segments of ICMP messages to bypass network defenses and consume resources.
- **TCP SYN Flood:** A TCP-based attack that generates a high volume of SYN requests without completing the three-way handshake, causing the server to accumulate half-open connections.
- **Synonymous IP Flood:** This attack sends spoofed TCP-SYN packets in which the source and destination IP addresses are set to the target's own address, causing the server to expend resources processing self-directed, invalid traffic.

These attack types collectively aim to overwhelm networked systems, degrade their performance, or render them entirely unresponsive, thereby threatening the availability and reliability of IIoT services.

3.5.2. Execution of Reconnaissance Attacks

Reconnaissance attacks are designed to collect detailed information about the IoT network topology, device configurations, and potential vulnerabilities. These attacks are often used as a preparatory phase for more disruptive threats, such as DoS, DDoS, or exploitation-based attacks. By gathering intelligence on active devices and services, attackers can refine their strategies and select high-impact targets with greater precision.

The reconnaissance methods employed in this context include the following:

- **Ping Sweep:** In this technique, ICMP Echo Request packets are sent to a range of IP addresses to determine active hosts, with devices responding via ICMP Echo Reply packets identified as online and reachable.
- **OS Scan:** Also referred to as operating system fingerprinting, this technique seeks to identify the specific operating system and its release version on a target device by analyzing its network responses and the behavior of open ports and services.

- **Vulnerability Scan:** This automated process probes devices and systems for known security weaknesses. It identifies exploitable flaws that could be leveraged in subsequent attacks, thereby supporting risk assessment and the prioritization of targets.
- **Port Scan:** This attack is employed to determine whether network ports on a target device are open, closed, or filtered. The attacker transmits a sequence of packets to multiple ports and analyzes the responses to identify available services and potential attack vectors.
- **Host Discovery:** This foundational step in many attacks involves identifying all active hosts within a network. Various techniques are employed to enumerate IP addresses of connected devices, providing a comprehensive view of the network environment.

Overall, reconnaissance attacks do not directly disrupt device functionality but enable the attacker to map the network and identify potential vulnerabilities. This low-impact nature allows them to remain stealthy while collecting critical intelligence for future exploitation.

3.5.3. Execution of Web-Based Attacks

Web-based attacks compromise IoT devices by exploiting vulnerabilities in their web applications or underlying communication protocols. These attacks seek to undermine system confidentiality, integrity, or availability by manipulating application inputs, injecting malicious code, or exploiting insecure web functionalities. The attacks executed in this study demonstrate various techniques commonly used to exploit such weaknesses.

The web-based attack types executed include the following:

- **SQL Injection:** This attack exploits web application input fields to inject malicious SQL statements, aiming to gain unauthorized access to the underlying database, retrieve sensitive information, or execute arbitrary queries.
- **Command Injection:** Similar in concept to SQL injection, this attack targets system-level commands by injecting malicious input into web forms or parameters, aiming to execute unauthorized commands on the host operating system.
- **Backdoor Malware:** Involves the installation of malicious software on the target system to establish a covert entry point. This enables persistent unauthorized access for executing malicious operations or exfiltrating data.
- **Uploading Attack:** Exploits vulnerabilities in file upload mechanisms of web applications to upload harmful files (e.g., scripts or executables). Once uploaded, these files can be executed to compromise the host system or escalate privileges.
- **Cross-Site Scripting (XSS):** This attack enables adversaries to inject malicious client-side scripts into web pages viewed by other users. Such scripts can be used to steal session cookies, redirect user traffic, or alter web content to deceive users and harvest sensitive information.
- **Browser Hijacking:** These attacks alter web browser configurations, including the homepage, search engine, or bookmarks, in order to redirect users to malicious sites or inject unwanted advertisements. The primary objective is often financial gain or data theft.

These attacks exploit the trust and accessibility of web interfaces in IoT systems, making them particularly dangerous for both data confidentiality and system integrity. Effective input validation, access control, and regular patching are essential countermeasures to mitigate these threats.

3.5.4. Execution of Spoofing and Man-in-the-Middle (MitM) Attacks

Spoofing and man-in-the-middle (MitM) attacks are conducted to deceive communication protocols and impersonate legitimate devices within an IoT network. These attacks

are primarily aimed at gaining unauthorized access, intercepting sensitive information, altering data in transit, or distributing malware. By masquerading as trusted entities, attackers can bypass security controls and compromise the confidentiality and integrity of network communications.

The spoofing and impersonation attacks executed include the following:

- **ARP Spoofing:** This attack compromises Address Resolution Protocol (ARP) tables by transmitting forged ARP messages that bind the attacker's MAC address to the IP address of a legitimate device. Consequently, network traffic intended for the legitimate device is rerouted to the attacker, enabling eavesdropping, data tampering, or service disruption.
- **IP Spoofing:** In this attack, the attacker forges the source IP address of packets to make them appear as if they originate from a trusted device. This enables the attacker to bypass IP-based access controls, inject malicious data into ongoing sessions, or initiate further attacks without revealing their true identity.
- **Impersonation Attack:** Leveraging information obtained during reconnaissance, the attacker mimics a legitimate device within the network. By forging identification attributes or communication patterns, the attacker sends data on behalf of the impersonated device, potentially misleading other systems or users and facilitating unauthorized data access or manipulation.

These attacks compromise the reliability and trust of network communications, posing significant threats to secure data exchange in IIoT environments. Defensive strategies include protocol hardening, traffic monitoring, and the deployment of authentication mechanisms to detect and prevent identity spoofing and unauthorized communication.

3.5.5. Execution of Brute-Force Attacks

Brute-force attacks attempt to obtain unauthorized access by systematically trying multiple login credentials. In this study, two dictionary-based brute-force attack variants were executed:

- **Telnet Brute-Force:** Attempts to access devices via Telnet using a predefined list of common credentials.
- **SSH Brute-Force:** Targets SSH services by automating login attempts with a dictionary of username–password pairs.

These attacks exploit weak authentication, highlighting the need for strong credentials and secure access configurations.

3.5.6. Execution of Malware (Mirai) Attacks

The Mirai malware is a prominent and highly disruptive threat targeting Internet of Things (IoT) and Industrial Internet of Things (IIoT) networks. Initially discovered in 2016, Mirai infects IoT devices by exploiting weak authentication mechanisms and leveraging default or hardcoded credentials, particularly over the Telnet protocol. Once compromised, these devices become part of a botnet that can be remotely controlled to launch large-scale distributed denial-of-service (DDoS) attacks or other malicious activities. The impact of Mirai on IIoT infrastructures is particularly critical due to the interconnected nature of industrial systems and their often inadequate security configurations, which can result in significant operational disruptions and safety risks.

Figure 7 illustrates the fundamental architecture and operational workflow of executing a malware attack using Mirai within our IIoT testbed environment. The execution is structured into two main phases: the Infection Phase and the Attack Execution Phase. In the diagram, black solid lines represent the standard network connections among devices

in the testbed. Orange dashed lines indicate the infection pathway of the Mirai malware as it propagates across devices. Dashed red lines denote the actual attacks initiated by infected devices under the control of the Mirai attacker. Finally, gray dashed lines depict the command and control communications exchanged between the Mirai CNC server and the infected devices.

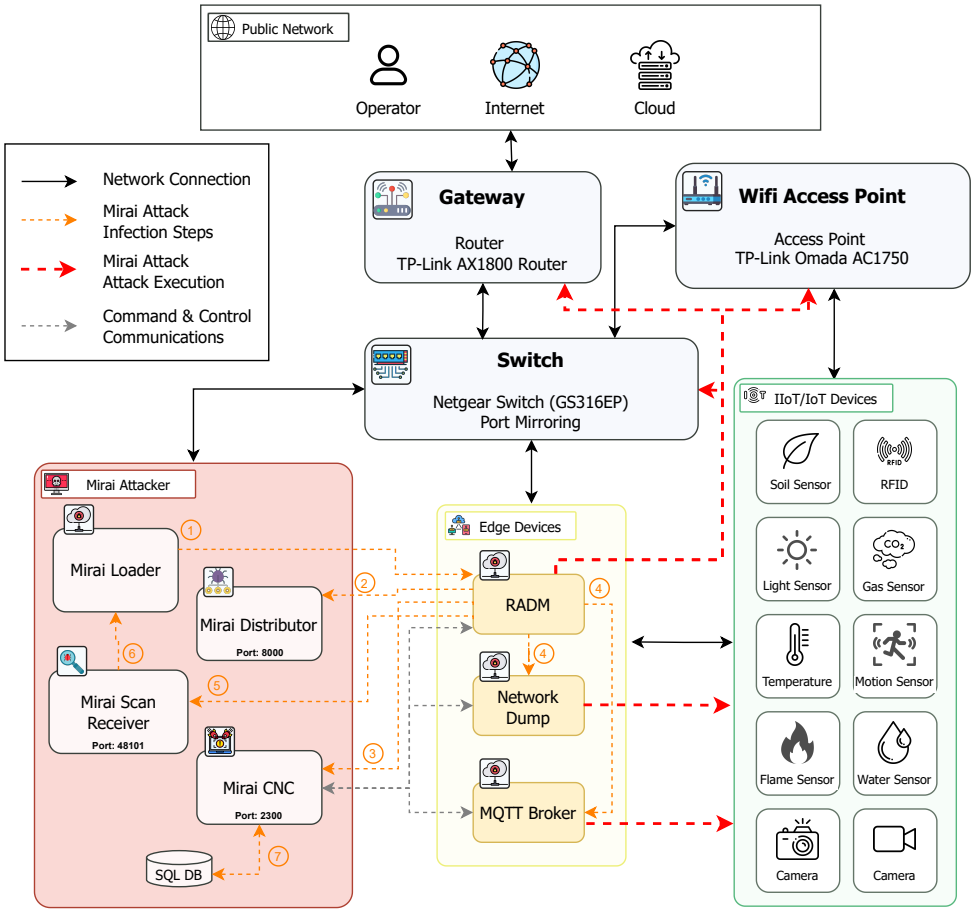

Figure 7. Attack framework for Mirai attack execution in the testbed.

In the Infection Phase, the attacker initially compromises one device, which then acts as a propagation vector to infect other devices within the network. The process continues recursively, rapidly expanding the botnet.

In the Attack Execution Phase, once a sufficient number of devices have been compromised, the attacker leverages the botnet to launch various network-level attacks, including DDoS or protocol-specific disruptions targeting other IIoT assets.

3.5.7. Mirai Malware Infection Phase

During the Infection Phase, the attacker initiates the malware deployment by compromising a vulnerable device, which subsequently facilitates the spread of the malware to additional devices across the network. As depicted in Figure 7, the infection mechanism is orchestrated through four core modules within the Mirai architecture: the Mirai Loader, Mirai Distributor, Mirai Scan Receiver, and Mirai Command and Control (CNC) server. The following steps summarize the infection workflow.

- **Step 1: Credential Bruteforce via Telnet.** The Mirai Loader targets a potential victim by initiating a Telnet connection and attempts to gain access using a brute-force attack

on default or weak credentials. This exploits the inherent vulnerabilities of the Telnet protocol commonly enabled on IoT devices.

- **Step 2: Malware Deployment.** Upon successful access, the Mirai Distributor transfers the Mirai executable to the compromised device using an HTTP connection. The executable is then loaded and executed on the device.
- **Step 3: Establishing CNC Communication.** The infected device now operates under the control of the Mirai malware. It connects to the Mirai CNC server, which issues commands to coordinate further malware propagation and manage the infected botnet. All communication with the CNC server is conducted over Telnet in encoded binary format.
- **Step 4: Network Scanning for New Victims.** The infected device initiates a scanning process targeting random IP addresses within the network, seeking additional vulnerable devices. It attempts to authenticate to these devices using the same brute-force credential attack over Telnet.
- **Step 5: Reporting to Scan Receiver.** When a new vulnerable device is discovered and successfully accessed, the infected device sends the target's IP address, port, and credentials to the Mirai Scan Receiver.
- **Step 6: Infection Propagation.** The Mirai Scan Receiver relays the newly discovered victim information to the Mirai Loader. The infection cycle then restarts from Step 1, targeting the new device.
- **Step 7: CNC Administration** A MySQL database is employed to support administrative operations of the Mirai CNC. It includes tables for user authentication to the CNC terminal, command execution logs, and IP whitelisting. This enables the botnet administrator to control access, review command history, and define IPs that are either excluded from or targeted for attacks.

This recursive and automated mechanism allows Mirai to quickly compromise a large number of devices, establishing a powerful and resilient botnet capable of executing coordinated attacks across an IIoT environment.

3.5.8. Mirai Malware Attack Execution Phase

Once the Mirai malware has successfully propagated through the network and infected a sufficient number of devices, the attacker can initiate coordinated distributed denial-of-service (DDoS) attacks. In this study, two common types of DDoS attacks were executed using the compromised devices: TCP SYN Flood and UDP Flood. Upon receiving the attack command from the Mirai CNC server, the infected devices began transmitting large volumes of SYN or UDP packets toward selected targets. The targets included various components of the IIoT infrastructure, such as networking equipment (e.g., access points and routers), edge computing nodes (e.g., MQTT brokers), as well as IIoT sensors and surveillance cameras. These attacks simulate realistic threat scenarios that can significantly disrupt industrial network operations.

4. From Data Generation to Hierarchical Information-Driven Feature Selection

This section presents the end-to-end workflow for generating, preparing, and engineering data to support anomaly detection in IIoT environments, as illustrated in Figure 8. The workflow proceeds through four major phases: data generation, data preparation, feature extraction and selection, and model evaluation using the constructed dataset.

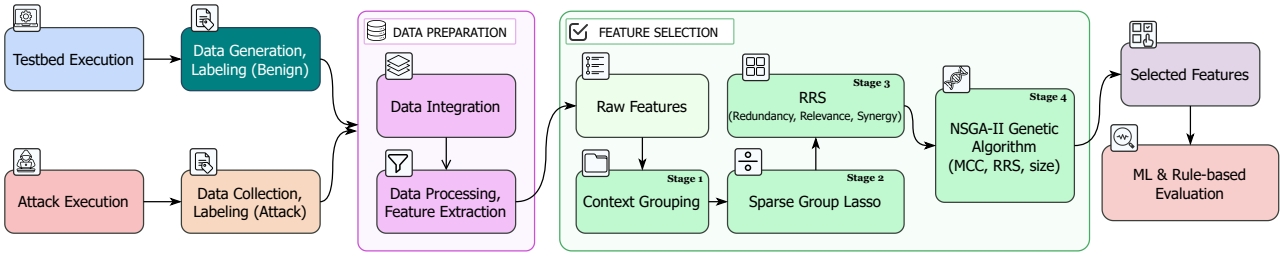

Figure 8. Proposed data generation workflow and feature engineering.

In the Data Generation phase (Sections 3.4 and 3.5), benign and malicious traffic is systematically collected under diverse operational and adversarial scenarios within the testbed. The Data Preparation phase then integrates benign and attack samples into a coherent dataset, applying preprocessing steps to remove redundancies and extract informative attributes from both network traffic and sensor readings. Next, the Feature Selection phase applies a novel information-driven strategy designed to retain only the most relevant and discriminative features across network- and sensor-level dimensions. Finally, in the Evaluation phase, the selected features are used to benchmark anomaly detection performance through a diverse set of machine learning and deep learning models, validating both the effectiveness and the generalizability of the dataset.

4.1. Proposed Hierarchical Multi-Stage Information-Driven Feature Selection Framework

Modern IIoT environments produce vast amounts of heterogeneous data, ranging from packet-level statistics and protocol indicators to sensor dynamics and temporal patterns. While this richness offers valuable insights, it also introduces challenges such as overfitting, a high computational cost, and reduced interpretability. To address these challenges, we propose a hierarchical multi-stage feature selection framework (Figure 8), beginning with a context-aware grouping strategy that reduces dimensionality while preserving the semantic structure of the feature space.

The primary motivations behind our feature-selection methodology are as follows:

- Dimensionality reduction: fewer input features lower training time, reduce memory usage, which is crucial for resource-constrained IIoT devices, and help mitigate the curse of dimensionality that negatively affects distance-based methods.
- Noise filtering: grouping features based on domain knowledge removes irrelevant or spurious correlations that can obscure meaningful patterns in the data.
- Interpretability: aggregating features into semantically meaningful categories (e.g., “Header Flags” or “Port Diversity”) improves transparency, enabling security practitioners to better understand and trust detection outcomes.

4.2. Overview of Hierarchical Information-Driven Feature Selection

This section integrates four complementary stages: context grouping, sparse group-lasso pruning, RRS scoring, and a multi-objective RRS-guided GA. Together, these stages produce a compact yet highly informative feature set for IIoT anomaly detection. The overall workflow is shown in Figure 8, and the main steps are summarized in Algorithm 1.

Stage 1: Context Grouping. Raw features ($p \approx 80$) are partitioned into $K = 10$ semantic blocks (e.g. “Packet Size”, “Header Flags”) using rule-based prefixes. The output is an index vector, $\texttt{gid}[j] \in \{1, \ldots, K\}$.

Stage 2: Sparse Group–Lasso. Given groups $\mathcal{G}$, SGL [66,67] solves:

$$\min_{\beta} \; \mathcal{L}(y, X\beta) + \lambda_1 \|\beta\|_1 + \lambda_2 \sum_{k=1}^{K} \|\beta_{G_k}\|_2,$$

with $\lambda_2 \ll \lambda_1$. Blocks survive; within each surviving block, we retain the top $k = 5$ coefficients, yielding $p' \approx 30$–50 columns.

Stage 3: RRS Tables. For the pruned matrix $X' \in \mathbb{R}^{n \times p'}$, we pre-compute *relevance* $R(j) = I(X_j; Y)$ [68], *redundancy* $D(i,j) = I(X_i; X_j)$ [68], and the *synergy proxy* $S(i,j) = \max\{0, I(X_i, X_j; Y) - R(i) - R(j)\}$. A per-column utility vector $u(j) = R(j) - \frac{1}{2}\sum_i D(i,j) + \sum_i S(i,j)$ guides the GA.

Stage 4: NSGA-II Optimization. [69] Each individual bit-string $z \in \{0,1\}^{p'}$ is scored through the objective vector

$$\big(-\mathrm{MCC}(z),\ -u^\top z,\ \|z\|_0\big),$$

which NSGA-II minimizes via two-point crossover, RRS-biased mutation, and crowding-distance selection. The algorithm returns a *Pareto set* of non-dominated masks that span accuracy, information value, and subset size.

Algorithm 1: Context-aware multi-stage feature selection.

Input: Raw matrix $\mathbf{X}$, labels $\mathbf{y}$
Output: Pareto-optimal subset list $\mathcal{F}^*$

```
// Stage 1: context grouping
gid ← GroupByPrefix(X);
// Stage 2: sparse group-lasso
β ← SGL(X, y, gid, λ1, λ2);
M ← indices of k largest |β| per block;
X' ← X_{:,M};
// Stage 3: RRS tables
(R, D, S) ← PrecomputeRRS(X', y);
u(j) ← R(j) − ½ Σ_i D(i,j) + Σ_i S(i,j);
// Stage 4: NSGA-II
Initial pop P ← random masks ∪ three RRS seed masks;
for g = 1 to G do                                   // 50 generations
    Evaluate each z ∈ P with (−MCC, −u^⊤ z, ‖z‖_0);
    P ← NSGA2NextGen(P);
F* ← first Pareto front of P;
return F*;
```

4.3. Input: Raw Feature Matrix

The feature-selection process begins with a raw feature matrix generated from the data-collection and preprocessing pipeline. This matrix comprises over 80 features (shown in Table 5) per instance, categorized as follows.

- Network-centric metrics: including byte counts, TCP/IP flag combinations, packet inter-arrival times, directional flow counts, and sets of unique IP/MAC/port values.
- Sensor and log metrics: covering sensor message intervals, value statistics (mean, max, min, standard deviation), and categorical distributions (e.g., type counts).

Formally, the dataset X is represented as follows:

$$X \in \mathbb{R}^{n \times p}, \quad \text{where } p \approx 80 \tag{1}$$

where n is the number of samples, and p is the number of features.

Table 5. List of all features in the dataset.

#	Feature	Description	Group
1	Message Interval	Time interval of messages in a time window	Log Data Rate
2	Messages Count	Total number of log messages in a time window	
3	Data Range Mean	Mean of value ranges across log entries	Log Data Stats
4	Data Range Maximum	Maximum value range across log entries	
5	Data Range Minimum	Minimum value range across log entries	
6	Data Range Std. Dev.	Std. deviation of value range across log entries	
7	Data Types Count	Number of distinct data types in log entries	
8	Data Types List	List of distinct data types in log entries	
9	Packets All Count	Total number of packets in a time window	Packet Traffic Rate
10	Packets Dst Count	Number of inbound Packets in a time window	
11	Packets Src Count	Number of outbound Packets in a time window	
12	Packet Interval	Time interval of packets in a time window	
13	Ports All Count	Number of all ports in a time window	Network Multiplexing
14	Ports Dst Count	Number of destination ports in a time window	
15	Ports Src Count	Number of source ports in a time window	
16	Ports All	List of all ports in a time window	
17	Ports Dst	List of destination ports in a time window	
18	Ports Src	List of source ports in a time window	
19	Protocols All Count	Number of unique protocols in a time window	
20	Protocols Dst Count	Number of destination protocols in a time window	
21	Protocols Src Count	Number of source protocols in a time window	
22	Protocols All	List of all protocols in a time window	
23	Protocols Dst	List of destination protocols in a time window	
24	Protocols Src	List of source protocols in a time window	
25	IPs All Count	Number of total IP addresses in a time window	Address Diversity
26	IPs Dst Count	Number of destination IP addresses in a time window	
27	IPs Src Count	Number of source IP addresses in a time window	
28	IPs All	List of all IP addresses in a time window	
29	IPs Dst	List of destination IP addresses in a time window	
30	IPs Src	List of source IP addresses in a time window	
31	MACs All Count	Number of all MAC addresses in a time window	
32	MACs Dst Count	Number of destination MAC addresses in a time window	
33	MACs Src Count	Number of source MAC addresses in a time window	
34	MACs All	List of all MAC addresses in a time window	
35	MACs Dst	List of destination MAC addresses in a time window	
36	MACs Src	List of source MAC addresses in a time window	
37	Fragmentation Score	Overall fragmentation score for a time window	Fragmentation
38	Fragmented Packets	Number of fragmented packets in a time window	
39	TCP ACK Flag Count	Number of TCP ACK flags in a time window	Header Flags
40	TCP FIN Flag Count	Number of TCP FIN flags in a time window	
41	TCP PSH Flag Count	Number of TCP PSH flags in a time window	
42	TCP RST Flag Count	Number of TCP RST flags in a time window	
43	TCP SYN Flag Count	Number of TCP SYN flags in a time window	
44	TCP URG Flag Count	Number of TCP URG flags in a time window	
45	TCP Flags Mean	Mean of TCP flag values in a time window	
46	TCP Flags Maximum	Maximum of TCP flag values in a time window	
47	TCP Flags Minimum	Minimum of TCP flag values in a time window	
48	TCP Flags Std. Dev.	Std. deviation of TCP flag values in a time window	
49	IP Flags Mean	Mean of IP flag values in a time window	
50	IP Flags Maximum	Maximum of IP flag values in a time window	
51	IP Flags Minimum	Minimum of IP flag values in a time window	
52	IP Flags Std. Dev.	Std. deviation of IP flag values in a time window	

Table 5. *Cont.*

#	Feature	Description	Group
53	Time Delta Mean	Mean inter-packet time delta in a time window	Timing Control
54	Time Delta Maximum	Maximum inter-packet time delta in a time window	
55	Time Delta Minimum	Minimum inter-packet time delta in a time window	
56	Time Delta Std. Dev.	Std. deviation of time deltas in a time window	
57	TTL Mean	Mean TTL value in a time window	
58	TTL Maximum	Maximum TTL value in a time window	
59	TTL Minimum	Minimum TTL value in a time window	
60	TTL Std. Dev.	Std. deviation of TTL values in a time window	
61	Window Size Mean	Mean TCP window size in a time window	
62	Window Size Maximum	Maximum TCP window size in a time window	
63	Window Size Minimum	Minimum TCP window size in a time window	
64	Window Size Std. Dev.	Std. deviation of window size in a time window	
65	Packet Size Mean	Mean packet size in a time window	Size Length
66	Packet Size Maximum	Maximum packet size in a time window	
67	Packet Size Minimum	Minimum packet size in a time window	
68	Packet Size Std. Dev.	Std. deviation of packet size in a time window	
69	Header Length Mean	Mean IP header length in a time window	
70	Header Length Maximum	Maximum IP header length in a time window	
71	Header Length Minimum	Minimum IP header length in a time window	
72	Header Length Std. Dev.	Std. deviation of header length in a time window	
73	IP Length Mean	Mean IP packet length in a time window	
74	IP Length Maximum	Maximum IP packet length in a time window	
75	IP Length Minimum	Minimum IP packet length in a time window	
76	IP Length Std. Dev.	Std. deviation of IP packet length in a time window	
77	MSS Mean	Mean maximum segment size in a time window	
78	MSS Maximum	Maximum segment size in a time window	
79	MSS Minimum	Minimum segment size in a time window	
80	MSS Std. Dev.	Std. deviation of segment size in a time window	
81	Payload Length Mean	Mean of payload lengths in a time window	
82	Payload Length Maximum	Maximum payload length in a time window	
83	Payload Length Minimum	Minimum payload length in a time window	
84	Payload Length Std. Dev.	Std. deviation of payload length in a time window	

4.4. Context Grouping

To impose semantic structure on the raw feature matrix, we employ a context-grouping mechanism that assigns each raw feature to one of nine high-level context categories (see Table 5, Column Group). This categorization is implemented using a deterministic, rule-based mapping mechanism combined with expert knowledge to classify raw features into semantically meaningful groups containing similar characteristics. This transformation compresses the input space into coherent feature blocks, serving as the foundation for subsequent selection and ranking stages in our anomaly detection pipeline.

Context grouping offers three key benefits: it enables domain-aware compression by capturing correlations within feature blocks (e.g., avg/max/std of packet size), preserves essential signals while allowing redundant features to be pruned, and supports block-level operations. This allows algorithms like Sparse Group Lasso and Genetic Algorithms to operate on group IDs, rather than individual features, drastically reducing the search space.

Formally, context groups are defined as a partition of the feature index set $\{1, 2, \ldots, p\}$, where each group G_k contains semantically related features. Let the following apply:

$$\begin{gathered} \mathcal{G} = \{G_1, G_2, \ldots, G_K\}, \quad \bigcup_{k=1}^{K} G_k = \{1, \ldots, p\}, \\ G_i \cap G_j = \emptyset \text{ for } i \neq j, \quad \phi : \{1, \ldots, p\} \to \{1, \ldots, K\} \end{gathered} \tag{2}$$

Here, $\mathcal{G}$ denotes the set of all groups, and ϕ is a deterministic mapping function based on domain knowledge and rule-based heuristics that assigns each feature to its

respective group. This grouping facilitates efficient group-level operations in downstream processing stages.

4.5. Sparse Group Lasso for Hierarchical Feature Selection

To further refine the feature space after context-aware grouping, we employ Sparse Group Lasso (SGL) as a structured regularization method that selects both informative groups and individual features within those groups [67]. SGL extends traditional Lasso and Group Lasso by introducing a dual-level sparsity mechanism, encouraging sparsity at both the group level and within groups. This property makes it particularly effective for high-dimensional IIoT data, where only a few groups and a limited number of features within each are truly informative for anomaly detection.

Let the input dataset be represented by the matrix $X \in \mathbb{R}^{n \times p}$ and the corresponding label vector by $y \in \mathbb{R}^n$, with $\mathcal{G} = \{G_1, G_2, \ldots, G_K\}$ denoting the predefined, non-overlapping feature groups defined in the previous section. The SGL objective function seeks the optimal coefficient vector $\beta \in \mathbb{R}^p$ by minimizing the following composite loss:

$$\hat{\beta} \;=\; \arg\min_{\beta} \Big\{ \underbrace{\mathcal{L}(y, X\beta)}_{\text{data fit}} + \lambda_1 \, \|\beta\|_1 + \; \lambda_2 \sum_{k=1}^{K} \|\beta_{G_k}\|_2 \Big\}, \tag{3}$$

where $\mathcal{L}(\cdot)$ is the logistic loss (for classification) or squared loss (for regression), β_{G_k} is the subvector of β corresponding to group G_k, λ_1 regulates individual feature sparsity via the ℓ_1-norm, and λ_2 imposes group-level sparsity through the ℓ_2-norm over groups. This formulation effectively balances empirical loss minimization, selective feature activation, and structured group reduction. The ℓ_1-norm encourages only a few individual features to be non-zero, while the ℓ_2-norm encourages entire groups to be pruned when non-informative. In our experiments, we set $\lambda_2 \ll \lambda_1$ so that no semantic block is discarded unless its best coefficient is negligible, while still encouraging each block to keep only its top–k ($k \leq 5$) statistics (e.g., *avg*, *max*).

We solve this optimization using an alternating proximal gradient descent method, which iteratively updates β until convergence. The resulting sparse solution identifies both the most predictive groups and the most relevant features within them, providing a compact and interpretable feature set for downstream anomaly detection in IIoT environments. In our experiments. This step preserves all K context blocks yet trims the feature count by more than 40%, producing a compact, interpretable set that feeds directly into the RRS + NSGA-II stage described next.

4.6. Relevance–Redundancy–Synergy (RRS) Scoring

We adopt the relevance–redundancy–synergy (RRS) [68] criterion to quantify the informational value of feature subsets for anomaly detection. RRS evaluates three core dimensions: **relevance**, measured as the mutual information between each feature and the target label Y, quantifies individual predictive power; in other words, it measures how much each feature tells us about the target variable; **redundancy**, defined as the pairwise mutual information between features, captures overlapping or duplicative information; and **synergy**, which reflects the additional predictive value obtained when features are considered jointly, beyond their individual contributions. This balanced scoring enables the selection of compact, informative, and complementary feature sets that improve detection accuracy and robustness. Simultaneously, uninformative or redundant features are excluded, reducing noise, resource usage, and computational overhead.

Let $\mathcal{S} \subseteq \{1, \ldots, p\}$ denote a subset of feature indices, and let X_j be the j-th feature column of the matrix $\mathbf{X}$. We estimate Shannon mutual information via histogramming (network and sensor data) or k-nearest neighbors (text logs) and denote it as $I(\cdot;\cdot)$.

The individual components of the RRS criterion are defined as follows [68,70]:

$$\text{Relevance:} \quad R(\mathcal{S}) = \sum_{j \in \mathcal{S}} I(X_j; Y), \tag{4}$$

$$\text{Redundancy:} \quad D(\mathcal{S}) = \frac{1}{|\mathcal{S}|(|\mathcal{S}|-1)} \sum_{\substack{i,j \in \mathcal{S} \\ i<j}} I(X_i; X_j), \tag{5}$$

$$\text{Synergy proxy:} \quad S(\mathcal{S}) = \frac{1}{2} \sum_{\substack{i,j \in \mathcal{S} \\ i<j}} \max\Big\{0,\, I(X_i, X_j; Y) - I(X_i; Y) - I(X_j; Y)\Big\}. \tag{6}$$

In Equation (6) [71,72], only the positive component of synergy is retained, while negative values, which indicate that joint contributions offer no additional information, are treated as zero.

4.6.1. Composite Score

We define the overall RRS score as a weighted linear combination of relevance, redundancy, and synergy:

$$\text{RRS}(\mathcal{S}) = \alpha\, R(\mathcal{S}) - \beta\, D(\mathcal{S}) + \gamma\, S(\mathcal{S}), \tag{7}$$

where the weights $\alpha, \beta, \gamma > 0$ are hyperparameters tuned using a validation split. This follows the general relevance–redundancy trade-off formulations used in prior work [68,70], but it extends them by explicitly incorporating a synergy component inspired by interaction information and recent advances in information decomposition [71,72]. To the best of our knowledge, this exact formulation is novel.

In our experiments, we use $\alpha = 1$, $\beta = 0.5$, and $\gamma = 1$. The formulation ensures that high relevance and synergy improve the score, while redundancy penalizes it.

4.6.2. Integration in Genetic Optimization

During the NSGA-II optimization process (Section 4.7), each individual in the population encodes a subset, $\mathcal{S}$, as a binary string. The second objective function, f_2, is defined as $\text{RRS}(\mathcal{S})$ from Equation (7), while the other two objectives are predictive accuracy (measured by MCC) and subset size. Since $R(\cdot)$, $D(\cdot)$, and $S(\cdot)$ are precomputed for the training split and cached in lookup tables, the fitness evaluation of individuals remains computationally efficient, even for large population sizes.

4.7. Multi-Objective Genetic Algorithm Guided by RRS

To effectively explore the exponentially large space of feature subsets, we employ the **NSGA-II** evolutionary algorithm [69], incorporating the Relevance–Redundancy–Synergy (RRS) score as an explicit optimization objective. Each individual in the population is encoded as a bit-string, $\mathbf{z} \in \{0,1\}^p$, where $z_j = 1$ indicates that feature X_j is selected. This encoding corresponds to a feature subset, $\mathcal{S}(\mathbf{z}) = \{\, j \mid z_j = 1 \,\}$.

4.7.1. Objective Vector

Each individual is evaluated using the following objective vector:

$$\mathbf{f}(\mathbf{z}) = \big(-\,\text{MCC}\big(\mathcal{S}(\mathbf{z})\big),\ -\,\text{RRS}\big(\mathcal{S}(\mathbf{z})\big),\ |\mathcal{S}(\mathbf{z})|\big), \tag{8}$$

where:

- MCC is the Matthews correlation coefficient achieved via a lightweight classifier (logistic regression) trained on $\mathcal{S}(\mathbf{z})$ using a 50% stratified subsample;
- RRS is the pre-computed relevance–redundancy–synergy score from Equation (7);
- $|\mathcal{S}|$ penalizes larger subsets to promote parsimony.

All three objectives are minimized. The negative signs convert higher-is-better scores (MCC and RRS) into a minimization framework suitable for NSGA-II.

4.7.2. NSGA-II Mechanics

Starting with an initial population, $\{\mathbf{z}_i^{(0)}\}_{i=1}^{P}$, NSGA-II applies evolutionary operators across generations: (i) two-point crossover; (ii) RRS-biased bit-flip mutation with a 70% probability of activating a feature ($0 \to 1$) and 30% for deactivation ($1 \to 0$); and (iii) elitist selection based on non-dominated sorting and crowding distance. Key algorithm parameters include pop_size $= 120$, n_gen $= 30$, crossover rate $c_x = 0.7$, mutation rate $\mu = 0.2$, and a hard upper bound $|\mathcal{S}| \leq 60$. The initial population is seeded with three individuals comprising the top-20, top-25, and top-30 features ranked by RRS utility, guiding the search toward informative regions.

4.7.3. Outcome

After 50 generations, the algorithm returns the first Pareto front, $\mathcal{F}^*$, consisting of non-dominated solutions, $\{z_1^*, z_2^*, \dots\}$. As Equation (8) jointly optimizes classification accuracy, information contribution, and feature parsimony, each solution on $\mathcal{F}^*$ represents a best-achievable trade-off. Empirically, these subsets consist of 10–25 features, delivering MCC performance within 1–2% of the full feature model while reducing inference cost by approximately a factor of 3.

4.8. Experimental Results of the Proposed Feature Selection Method

In this subsection, we present the results obtained from applying the proposed Hierarchical Information-Driven Feature Selection method. We highlight the features selected through this approach and discuss their significance. Feature selection was performed for two target variables: binary classification (distinguishing between benign and attack instances) and multi-label classification (including benign, reconnaissance, DoS, DDoS, and other attack types). We analyze the results for both scenarios and identify the optimal feature subset that offers the best performance for anomaly detection in IIoT environments.

4.8.1. Feature Selection Results for Binary Classification Target Variable

The results of the proposed Hierarchical Information-Driven Feature Selection method for the binary classification target variable (benign vs. attack) are presented in Figure 9. This figure illustrates the frequency with which each feature was selected via the RRS-guided genetic algorithm, where a higher frequency indicates greater importance of the corresponding feature. As detailed in the previous section, feature selection is performed in two stages: initially, the Sparse Group Lasso is applied to prune less relevant groups of features (Stage 1), followed by the application of RRS scoring to further evaluate the remaining features (Stage 2). The genetic algorithm uses these scores to identify the subset of features that provides the highest informational value. For the binary classification task, the feature selection module identified a total of 12 features out of more than 80 initial candidates, distributed across different context groups, as summarized in Table 6.

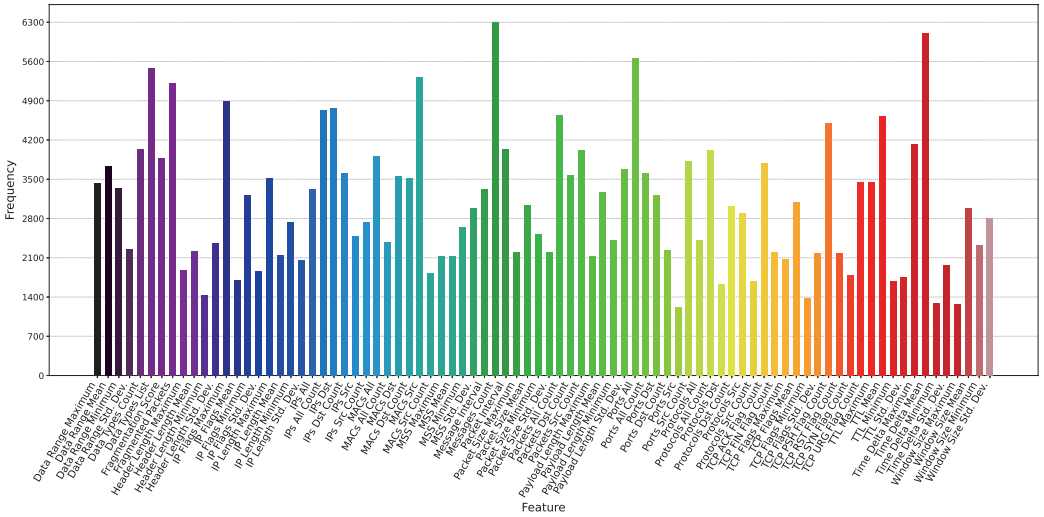

Figure 9. Features selected through RRS-guided genetic algorithm for binary classification.

Table 6. Selected features for analysis.

#	Feature Name	Context Group
1	Messages Count	Log Data Rate
2	Data Types List	Log Data Stats
3	Fragmented Packets	Fragmentation
4	IP Flags Maximum	Header Flags
5	TCP PSH Flag Count	
6	IPs All Count	Address Diversity
7	IPs Dst	
8	MACs Src	
9	Packets All Count	Packet Traffic Rate
10	Ports All	Network Multiplexing
11	Time Delta Mean	Timing Control
12	TTL Mean	

4.8.2. Feature Selection Results for Multi-Class Classification Target Variable

This subsection presents the results of the proposed hierarchical information-driven feature selection method for the multi-class classification target variable, which includes benign, reconnaissance, DoS, DDoS, and other attack types. Figure 10 displays the frequency with which each feature was selected via the RRS-guided genetic algorithm during different generations; a higher frequency indicates greater importance. For the multi-class classification task, the feature selection module identified 17 features out of more than 80 initial candidates, spanning different context groups, as summarized in Table 7. The experiments for binary and multi-class classification were conducted separately to ensure that all useful features relevant to the detection of various attack types are identified and included in the anomaly detection system.

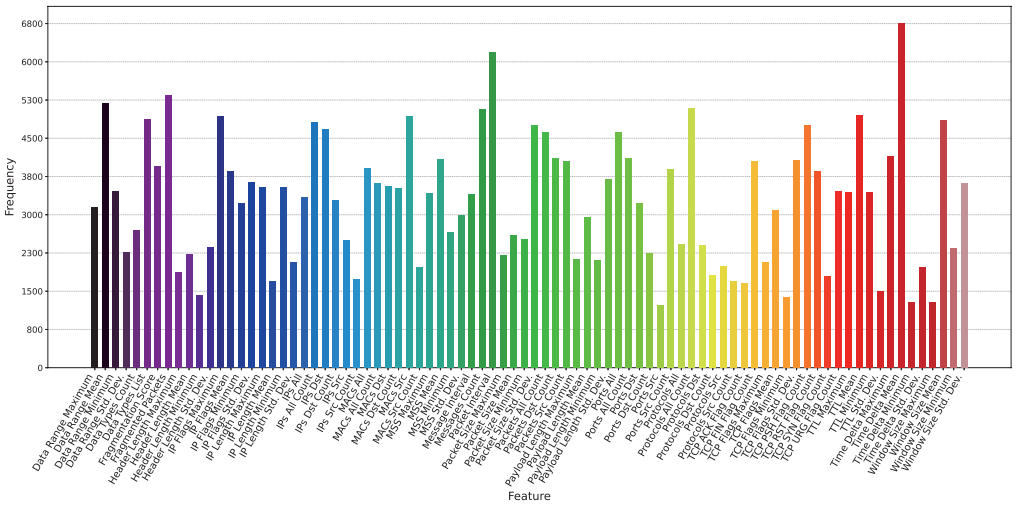

Figure 10. Features selected via RRS-guided genetic algorithm for multi-class classification.

Table 7. Extended set of selected features.

#	Feature Name	Context Group
1	Messages Count	Log Data Rate
2	Data Range Mean	Log Data Stats
3	Data Types List	
4	Fragmented Packets	Fragmentation
5	Packet Interval	Packet Traffic Rate
6	Packets All Count	
7	IPs Dst	Address Diversity
8	IPs All Count	
9	MACs Src	
10	Packet Size Std. Dev.	Size Length
11	Ports All	Network Multiplexing
12	Protocols All Count	
13	Time Delta Mean	Timing Control
14	TTL Mean	
15	Window Size Mean	
16	IP Flags Maximum	Header Flags
17	TCP PSH Flag Count	

4.8.3. Discussion

We employed the extended set of 17 features, as presented in Table 7, for the comparative experiments described in the following section, enabling a balanced and well-rounded assessment of the feature selection approach. This feature set includes all selected features relevant to both binary and multi-class classification tasks, as identified through our hierarchical selection process. By using this unified subset, we aim to assess the effectiveness of a summarized yet informative feature set on both detection accuracy and computational efficiency. This approach enables us to demonstrate the applicability of the proposed method in real-world IIoT scenarios, where sustaining high detection performance alongside minimal resource consumption is essential.

5. Experiments and Evaluations

We present in this section the results of evaluating the proposed dataset using a combination of machine learning and deep learning approaches. The objective is to demonstrate the applicability and versatility of the dataset for various anomaly detection mechanisms. By employing a diverse set of evaluation methods, we aim to show that the dataset effectively supports different detection paradigms, thereby facilitating comprehensive research and development of anomaly detection systems in Industrial Internet of Things (IIoT) environments.

5.1. Performance Assessment of Machine Learning and Deep Learning Methods

The DataSense evaluation pipeline designed for anomaly detection in IIoT environments is illustrated in Figure 11. The process begins with the ingestion of two primary data sources: raw network traffic data, captured in separate time-stamped PCAP files under various conditions (e.g., benign, reconnaissance, DoS), and IIoT sensor logs stored in Elasticsearch, containing time-stamped telemetry from industrial devices. These data streams are temporally aligned to extract sensor data corresponding to each attack scenario. The aligned network and sensor data are then aggregated into a unified multivariate time-series dataset, from which a comprehensive set of features is extracted. The resulting integrated dataset is partitioned into three detection-specific datasets: a Binary Dataset (normal vs. attack), a Multiclass-8 Dataset (seven attack types and normal), and a Multiclass-50 Dataset (fine-grained attack labels). Each dataset is stratified and then divided into training and testing sets, followed by data preprocessing steps such as vectorization of list-type values and normalization of numerical features. A multi-objective feature selection method is applied to improve both efficiency and detection accuracy. Subsequently, machine learning and deep learning algorithms are employed to assess detection performance. Finally, results from all experimental configurations are aggregated to produce a comprehensive evaluation summary.

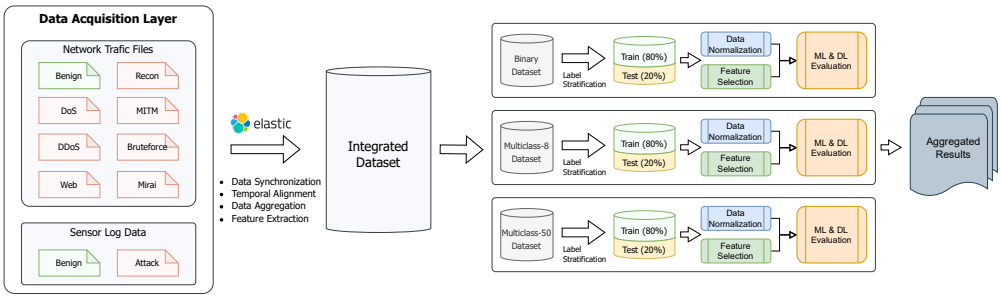

Figure 11. Datasense evaluation pipeline for anomaly detection in IIoT environments.

The experiments for evaluating both traditional machine learning and deep learning models were conducted on a Dell PowerEdge R530 server platform. The system is built on an Intel Xeon E5-2650 processor, featuring 24 physical cores clocked at a base speed of 2.20 GHz, and supported with 125 GB of RAM. This hardware configuration provided a sufficient computational capacity to train and evaluate a wide range of models under consistent conditions, ensuring reliable measurement of execution time and memory utilization across all experimental scenarios.

To evaluate the performance of detection mechanisms on the proposed dataset, we employed a set of widely used classification metrics. These metrics were selected to provide a comprehensive and objective assessment of both machine learning and deep learning models in the context of anomaly detection. The chosen metrics are accuracy, precision, recall, and F1-score. Each metric is formally defined below.

- **Accuracy**: the ratio of correctly classified samples to the total number of samples.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

where FP denotes the number of false positives, TP the number of true positives, FN the number of false negatives, and TN the number of true negatives.

- **Precision**: the proportion of true positive predictions relative to all predicted positive instances.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{10}$$

- **Recall (sensitivity)** represents the proportion of actual positives correctly identified via the model.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{11}$$

- **F1-score**: the F1-score is the harmonic mean of precision and recall, serving as a unified measure that captures the trade-off between the two.

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{12}$$

To comprehensively evaluate the proposed dataset under diverse anomaly detection scenarios, we employed 22 widely used machine learning (ML) and deep learning (DL) algorithms, spanning both traditional and contemporary approaches. These models are well established in cybersecurity and anomaly detection research due to their proven effectiveness and adaptability. The ML models include k-nearest neighbors (KNN), support vector machine (SVM), decision tree (DT), random forest (RF), logistic regression (LR), naive Bayes (NB), extreme gradient boosting (XGBoost), and two hybrid ensembles: HybridML_V1 (RF, SVM, and KNN as base learners with logistic regression as the meta-classifier) and HybridML_V2 (RF, KNN, and MLP as base learners with logistic regression as the meta-classifier). The DL models comprise long short-term memory (LSTM), convolutional neural network (CNN), bidirectional LSTM (BiLSTM), gated recurrent unit (GRU), CNN–LSTM and BiCNN–LSTM hybrids, bidirectional GRU (BiGRU), transformer, deep transformer, one-dimensional residual network (ResNet1D), Deep ResNet1D, autoencoder, and recurrent neural network (RNN).

Figure 12 presents the performance of these models across different classification scenarios: binary classification (benign vs. attack), 8-class classification (benign, DoS, DDoS, Recon, etc.), and fine-grained 50-class classification (e.g., Benign, DoS-Connect Flood, Recon-OS Scan, etc.). Table 8 further details the performance of each model across these scenarios with respect to various evaluation metrics, providing a comprehensive comparison of their effectiveness in handling different levels of classification granularity.

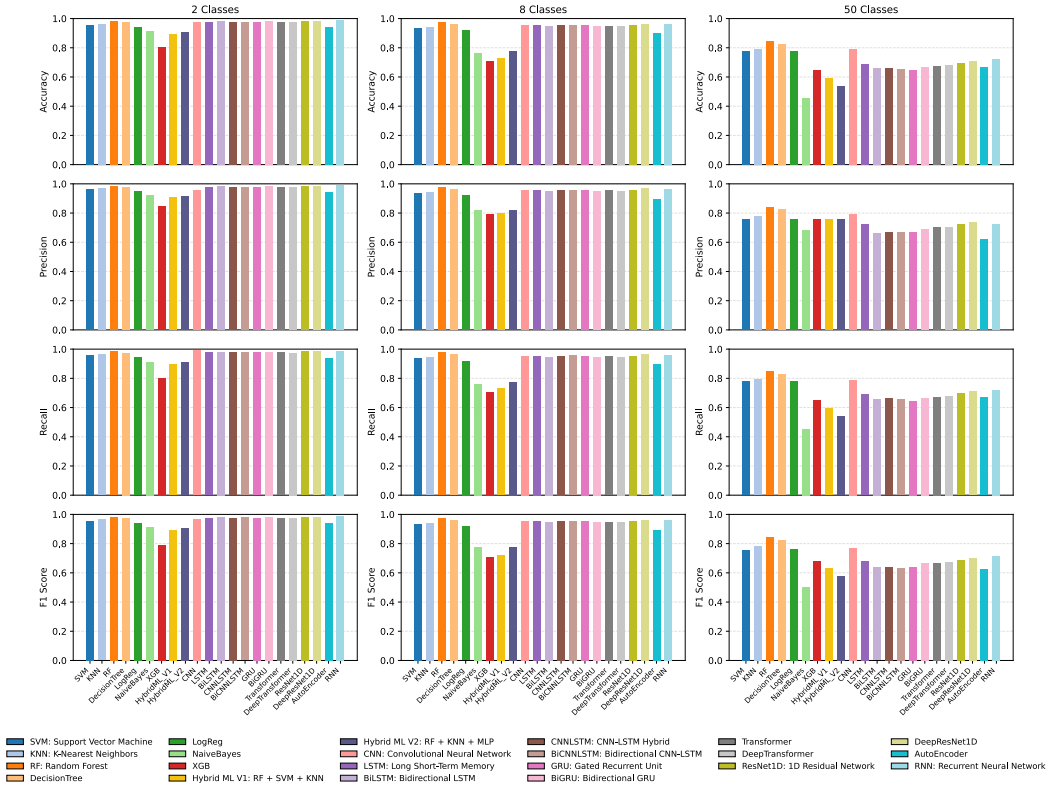

Figure 12. Results for anomaly detection on proposed dataset using different ML and DL techniques.

Discussion

As shown in Figure 12, model performance varied notably across the binary, 8-class, and 50-class classification tasks. In the two-class classification scenario, the best-performing models in terms of accuracy, F1-score, recall, and precision were LSTM, ResNet1D, BiCNN-LSTM, CNN-LSTM, RNN, and DeepTransformer, each achieving metrics in the range of 0.98 to 0.99, reflecting high reliability in binary detection. In the eight-class scenario, although overall metric values slightly declined due to increased complexity, models such as random forest, DeepResNet1D, CNN-LSTM, LSTM, BiCNN-LSTM, RNN, and transformer-based models demonstrated strong performance, with F1-scores mostly exceeding 0.94, indicating their robustness in handling moderate class diversity. Among these, random forest stood out with an F1-score of 0.97, the highest among all models. For the 50-class fine-grained scenario, performance dropped further, as was expected due to increased complexity. Nonetheless, random forest and decision tree remained among the most effective traditional models with F1-scores of 0.8455 and 0.8311, respectively. Among deep learning models, CNN performed best with an F1-score of 0.7766, while others, such as RNN, DeepTransformer, and CNN-LSTM, achieved moderate results with F1-scores ranging between 0.67 and 0.73, highlighting their capacity to generalize across a large number of classes despite the increased challenge.

5.2. Performance Assessment of Feature Selection Method: Detection Performance Perspective

To assess the efficacy of the proposed feature selection mechanism, a series of comparative experiments were conducted. Initially, the classification models, which included both machine learning and deep learning algorithms, were trained and evaluated using the complete set of features. Subsequently, the same models were executed using only the subset of features selected by the proposed method. Performance was then evaluated based on key detection metrics and computational efficiency, specifically training and evaluation

time and memory usage. The impact of feature selection on detection performance across various machine learning and deep learning models is shown in Figures 13–15.

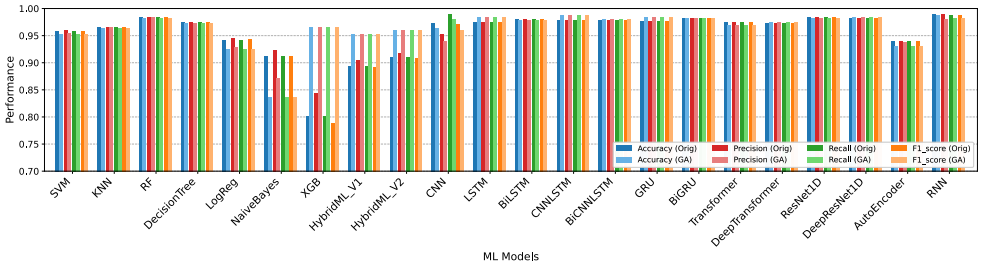

Figure 13. Impact of feature selection on binary classification performance.

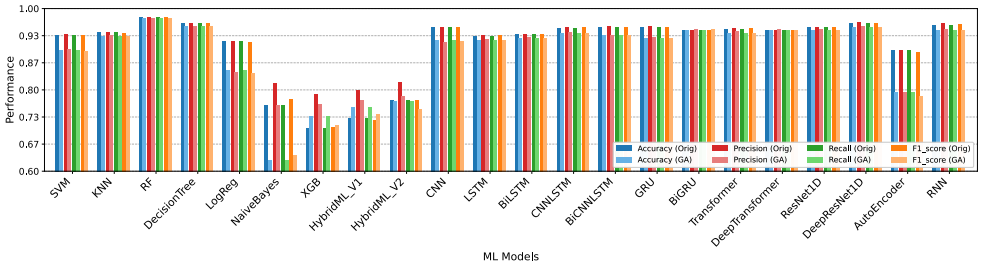

Figure 14. Impact of feature selection on eight-class classification performance.

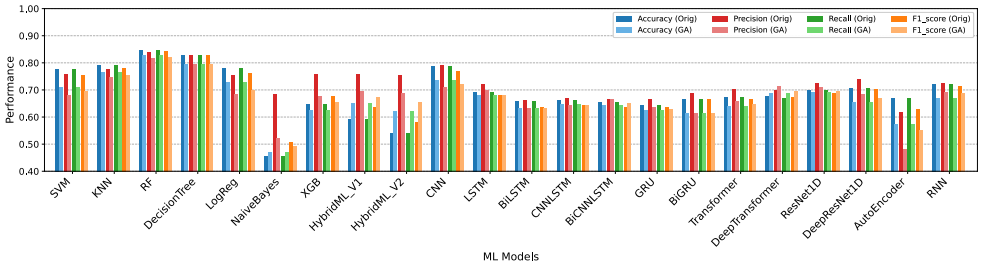

Figure 15. Impact of feature selection on 50-class classification performance.

5.3. Comparison with Established Feature Selection Methods

To further validate the effectiveness of the proposed feature selection approach, we compared its performance with principal component analysis (PCA), a widely adopted dimensionality reduction and feature selection method in anomaly detection research. Unlike PCA, which projects features into a lower-dimensional space based on variance maximization, our method directly identifies the most discriminative features correlated with the target variable, thereby preserving interpretability while enhancing detection performance.

The comparison results demonstrate that the proposed approach provides competitive or superior detection accuracy across multiple anomaly detection scenarios. The detection accuracy results are summarized in Table 8, highlight the efficiency of our method in balancing accuracy and computational costs, making it a practical choice for IIoT anomaly detection tasks.

Table 8. Classification performance of the evaluated algorithms over 2-, 8-, and 50-class scenarios.

Algorithm	Metric	2 Classes			8 Classes			50 Classes		
		Orig	PCA	FS	Orig	PCA	FS	Orig	PCA	FS
SVM	Accuracy	0.9576	0.9448	0.9519	0.9349	0.8989	0.8982	0.7770	0.7105	0.7099
	Precision	0.9603	0.9493	0.9544	0.9377	0.9026	0.8988	0.7599	0.6949	0.6815
	Recall	0.9576	0.9448	0.9519	0.9349	0.8989	0.8982	0.7770	0.7105	0.7099
	F1-score	0.9577	0.9450	0.9520	0.9335	0.8956	0.8948	0.7550	0.6845	0.6947
KNN	Accuracy	0.9649	0.9636	0.9638	0.9408	0.9333	0.9323	0.7911	0.7826	0.7639
	Precision	0.9663	0.9648	0.9651	0.9420	0.9344	0.9330	0.7763	0.7684	0.7463
	Recall	0.9649	0.9636	0.9638	0.9408	0.9333	0.9323	0.7911	0.7826	0.7639
	F1-score	0.9650	0.9637	0.9639	0.9402	0.9325	0.9315	0.7811	0.7730	0.7529
Random Forest	Accuracy	0.9843	0.9701	0.9831	0.9781	0.9435	0.9749	0.8482	0.8079	0.8269
	Precision	0.9845	0.9707	0.9833	0.9783	0.9441	0.9752	0.8409	0.7965	0.8183
	Recall	0.9843	0.9701	0.9831	0.9781	0.9435	0.9749	0.8482	0.8079	0.8269
	F1-score	0.9843	0.9701	0.9831	0.9780	0.9429	0.9748	0.8439	0.8007	0.8221
Decision Tree	Accuracy	0.9743	0.9510	0.9727	0.9628	0.9084	0.9572	0.8280	0.7640	0.7949
	Precision	0.9744	0.9510	0.9728	0.9630	0.9086	0.9572	0.8279	0.7621	0.7946
	Recall	0.9743	0.9510	0.9727	0.9628	0.9084	0.9572	0.8280	0.7640	0.7949
	F1-score	0.9743	0.9510	0.9727	0.9627	0.9083	0.9571	0.8277	0.7624	0.7944
Logistic Reg.	Accuracy	0.9424	0.9173	0.9246	0.9187	0.8084	0.8478	0.7792	0.7287	0.7299
	Precision	0.9455	0.9245	0.9289	0.9205	0.8066	0.8439	0.7539	0.6942	0.6840
	Recall	0.9424	0.9173	0.9246	0.9187	0.8084	0.8478	0.7792	0.7287	0.7299
	F1-score	0.9426	0.9176	0.9249	0.9172	0.8007	0.8409	0.7628	0.6968	0.6979
Naïve Bayes	Accuracy	0.9127	0.7423	0.8369	0.7614	0.6356	0.6257	0.4539	0.4693	0.4703
	Precision	0.9234	0.7665	0.8708	0.8163	0.7433	0.7618	0.6830	0.6268	0.5227
	Recall	0.9127	0.7423	0.8369	0.7614	0.6356	0.6257	0.4539	0.4693	0.4703
	F1-score	0.9130	0.7422	0.8365	0.7762	0.6496	0.6389	0.5059	0.5100	0.4916
XGBoost	Accuracy	0.8011	0.9576	0.9649	0.7065	0.7433	0.7353	0.6468	0.6090	0.6266
	Precision	0.8431	0.9586	0.9662	0.7886	0.7438	0.7637	0.7567	0.6854	0.6784
	Recall	0.8011	0.9577	0.9649	0.7065	0.7433	0.7353	0.6468	0.6190	0.6266
	F1-score	0.7886	0.9576	0.9650	0.7071	0.7426	0.7125	0.6778	0.6413	0.6535
HybridML1	Accuracy	0.8943	0.9502	0.9524	0.7293	0.7444	0.7565	0.5938	0.6075	0.6514
	Precision	0.9052	0.9508	0.9526	0.7984	0.7448	0.7742	0.7572	0.6643	0.6956
	Recall	0.8943	0.9502	0.9524	0.7293	0.7444	0.7565	0.5938	0.6275	0.6514
	F1-score	0.8922	0.9503	0.9523	0.7244	0.7439	0.7399	0.6353	0.6493	0.6729
HybridML2	Accuracy	0.9098	0.9579	0.9592	0.7749	0.7467	0.7729	0.5388	0.6097	0.6225
	Precision	0.9169	0.9585	0.9592	0.8193	0.7472	0.7846	0.7558	0.6780	0.6868
	Recall	0.9098	0.9569	0.9592	0.7749	0.7467	0.7729	0.5388	0.6197	0.6225
	F1-score	0.9085	0.9580	0.9592	0.7741	0.7462	0.7526	0.5798	0.6322	0.6554
CNN	Accuracy	0.9736	0.9570	0.9642	0.9533	0.9025	0.9209	0.7881	0.7295	0.7351
	Precision	0.9522	0.9251	0.9406	0.9543	0.8996	0.9176	0.7918	0.6986	0.7088
	Recall	0.9894	0.9813	0.9800	0.9533	0.9025	0.9209	0.7881	0.7295	0.7351
	F1-score	0.9704	0.9524	0.9599	0.9530	0.8989	0.9183	0.7675	0.7005	0.7228
LSTM	Accuracy	0.9757	0.9816	0.9845	0.9328	0.9251	0.9209	0.6901	0.6785	0.6798
	Precision	0.9757	0.9810	0.9845	0.9334	0.9267	0.9253	0.7199	0.7011	0.6985
	Recall	0.9757	0.9826	0.9845	0.9328	0.9251	0.9209	0.6901	0.6785	0.6798
	F1-score	0.9757	0.9816	0.9845	0.9338	0.9255	0.9222	0.6806	0.6710	0.6808
BiLSTM	Accuracy	0.9803	0.9795	0.9780	0.9363	0.9351	0.9260	0.6576	0.6453	0.6312
	Precision	0.9803	0.9793	0.9780	0.9377	0.9379	0.9293	0.6605	0.6611	0.6328
	Recall	0.9803	0.9785	0.9780	0.9363	0.9351	0.9260	0.6576	0.6453	0.6312
	F1-score	0.9803	0.9786	0.9780	0.9367	0.9360	0.9265	0.6365	0.6541	0.6326
CNN-LSTM	Accuracy	0.9803	0.9795	0.9780	0.9363	0.9351	0.9260	0.6576	0.6453	0.6312
	Precision	0.9803	0.9793	0.9780	0.9377	0.9379	0.9293	0.6605	0.6611	0.6328
	Recall	0.9803	0.9785	0.9780	0.9363	0.9351	0.9260	0.6576	0.6453	0.6312
	F1-score	0.9803	0.9786	0.9780	0.9367	0.9360	0.9265	0.6365	0.6541	0.6326

Table 8. *Cont.*

Algorithm	Metric	2 Classes			8 Classes			50 Classes		
		Orig	PCA	FS	Orig	PCA	FS	Orig	PCA	FS
BiCNN-LSTM	Accuracy	0.9788	0.9808	0.9811	0.9545	0.9472	0.9340	0.6532	0.6434	0.6454
	Precision	0.9788	0.9801	0.9811	0.9563	0.9508	0.9342	0.6674	0.6567	0.6641
	Recall	0.9788	0.9807	0.9811	0.9545	0.9472	0.9340	0.6532	0.6434	0.6454
	F1-score	0.9788	0.9806	0.9811	0.9551	0.9483	0.9337	0.6358	0.6360	0.6518
GRU	Accuracy	0.9765	0.9874	0.9847	0.9530	0.9522	0.9274	0.6448	0.6356	0.6253
	Precision	0.9766	0.9874	0.9847	0.9566	0.9562	0.9283	0.6654	0.6505	0.6352
	Recall	0.9765	0.9874	0.9847	0.9530	0.9522	0.9274	0.6448	0.6356	0.6253
	F1-score	0.9765	0.9874	0.9847	0.9542	0.9535	0.9275	0.6370	0.6427	0.6301
BiGRU	Accuracy	0.9815	0.9839	0.9828	0.9467	0.9458	0.9474	0.6649	0.6379	0.6138
	Precision	0.9815	0.9840	0.9829	0.9456	0.9401	0.9501	0.6881	0.6667	0.6144
	Recall	0.9815	0.9839	0.9828	0.9467	0.9458	0.9474	0.6649	0.6479	0.6138
	F1-score	0.9815	0.9839	0.9828	0.9457	0.9473	0.9478	0.6643	0.6434	0.6141
Transformer	Accuracy	0.9753	0.9692	0.9700	0.9499	0.9319	0.9383	0.6727	0.6571	0.6419
	Precision	0.9754	0.9682	0.9701	0.9518	0.9368	0.9431	0.7010	0.6756	0.6573
	Recall	0.9753	0.9686	0.9700	0.9499	0.9319	0.9383	0.6727	0.6371	0.6419
	F1-score	0.9753	0.9684	0.9700	0.9503	0.9337	0.9397	0.6643	0.6470	0.6482
DeepTransformer	Accuracy	0.9736	0.9701	0.9746	0.9470	0.9409	0.9470	0.6779	0.6824	0.6888
	Precision	0.9737	0.9711	0.9746	0.9476	0.9466	0.9480	0.6989	0.7016	0.7153
	Recall	0.9736	0.9721	0.9746	0.9470	0.9409	0.9470	0.6709	0.6814	0.6888
	F1-score	0.9736	0.9716	0.9746	0.9467	0.9421	0.9473	0.6717	0.6926	0.6943
ResNet1D	Accuracy	0.9839	0.9807	0.9822	0.9539	0.9576	0.9455	0.6983	0.6952	0.6918
	Precision	0.9840	0.9807	0.9824	0.9549	0.9591	0.9492	0.7236	0.7264	0.7111
	Recall	0.9839	0.9807	0.9822	0.9539	0.9576	0.9455	0.6983	0.6952	0.6918
	F1-score	0.9839	0.9807	0.9822	0.9541	0.9577	0.9465	0.6884	0.6948	0.6964
DeepResNet1D	Accuracy	0.9828	0.9851	0.9839	0.9643	0.9453	0.9537	0.7079	0.6426	0.6549
	Precision	0.9829	0.9851	0.9840	0.9655	0.9459	0.9569	0.7386	0.6850	0.6836
	Recall	0.9828	0.9851	0.9839	0.9643	0.9453	0.9537	0.7079	0.6526	0.6549
	F1-score	0.9828	0.9851	0.9839	0.9646	0.9454	0.9546	0.7035	0.6622	0.6689
AutoEncoder	Accuracy	0.9389	0.9387	0.9305	0.8972	0.8651	0.7938	0.6698	0.6437	0.5742
	Precision	0.9404	0.9428	0.9387	0.8964	0.8634	0.7943	0.6189	0.5831	0.4810
	Recall	0.9389	0.9387	0.9305	0.8972	0.8651	0.7938	0.6698	0.6437	0.5742
	F1-score	0.9390	0.9389	0.9308	0.8922	0.8585	0.7844	0.6288	0.6034	0.5512
RNN	Accuracy	0.9897	0.9807	0.9878	0.9595	0.9461	0.9473	0.7201	0.6684	0.6697
	Precision	0.9895	0.9787	0.9798	0.9625	0.9467	0.9501	0.7256	0.6705	0.6923
	Recall	0.9870	0.9774	0.9830	0.9595	0.9461	0.9473	0.7201	0.6624	0.6697
	F1-score	0.9882	0.9780	0.9814	0.9605	0.9460	0.9475	0.7156	0.6641	0.6892

Orig—original feature set, PCA—principal component analysis, FS—proposed feature selection.

Discussion

The impact of the proposed feature selection method on detection performance was evaluated using four key metrics: accuracy, precision, recall, and F1-score. In the two-class classification scenario, several models exhibited notable improvements with the selected feature set. Most prominently, XGBoost showed a substantial increase in F1-score by +0.176, followed by improvements in DeepTransformer, BiCNN-LSTM, transformer, and RNN. In these models, other detection metrics also improved consistently, highlighting the effectiveness of the selected feature subset. Additionally, models such as CNN-LSTM, GRU, and BiGRU demonstrated marginal yet stable improvements, underscoring the benefit of reducing feature redundancy while retaining critical information.

In the eight-class classification task, while some models experienced moderate performance degradation, key models such as KNN, random forest, decision tree, and ResNet1D maintained nearly identical detection performance, showing only marginal decreases in F1-score. This suggests that these models are relatively robust to the reduced feature set.

In contrast, models like naive Bayes, logistic regression, and a few deep learning models showed more sensitivity, indicating a greater dependence on a larger feature space for effective multi-class classification.

For the 50-class classification scenario, detection performance generally declined across most models when using the reduced feature set. This was expected, as fine-grained classification requires more comprehensive feature representation to preserve class separability. However, some models, such as naive Bayes, still exhibited slight gains, indicating that targeted feature selection can provide benefits even in high-complexity tasks.

Overall, these findings, summarized in Table 8 and illustrated in Figures 13–15, demonstrate that the proposed feature selection method is particularly effective in binary and moderate multi-class settings, where it enhances or preserves detection performance across a variety of models. As further shown in Figures 16–18, reducing feature dimensionality not only maintains or improves detection metrics but also significantly lowers resource usage in terms of both memory and computation, making it a practical and efficient approach for deployment in resource-constrained IIoT environments.

5.4. Performance Assessment of Feature Selection Method: Resource Efficiency Perspective

This section evaluates the impact of the proposed feature selection method on resource utilization across binary, 8-class, and 50-class classification tasks. Specifically, we analyze and compare the computational efficiency of various machine learning and deep learning models when using the full feature set versus the reduced sets obtained through our feature selection method and PCA. Resource profiling considered CPU execution time and memory consumption during both training and evaluation phases. The results are presented in Figures 16–18, which correspond to the binary, 8-class, and 50-class tasks, respectively. Each figure shows three stacked bar plots per model, comparing training and evaluation times for the original feature set and the reduced sets.

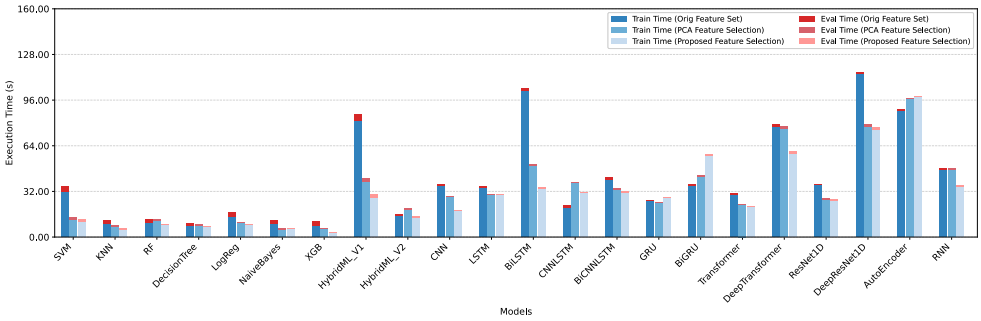

Figure 16. Impact of feature selection on training and evaluation time in binary classification.

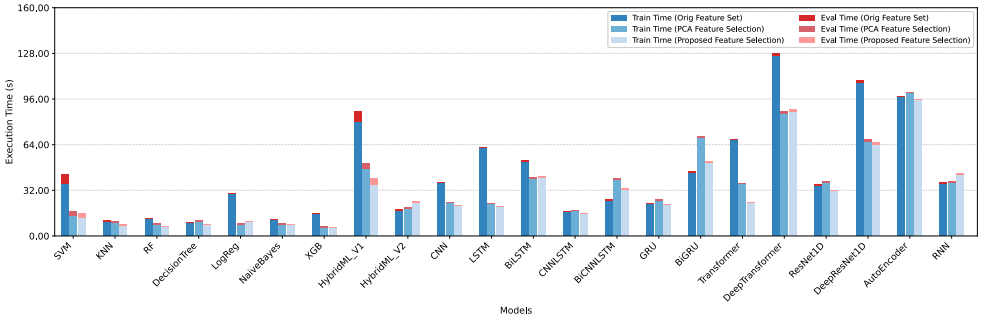

Figure 17. Impact of feature selection on training and evaluation time in eight-class classification.

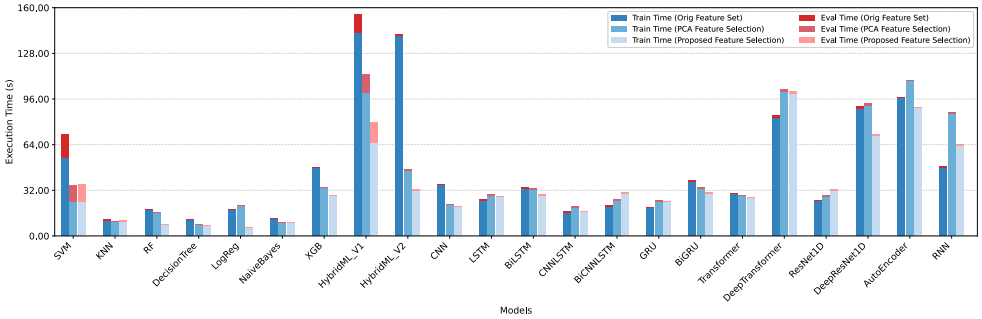

Figure 18. Impact of feature selection on training and evaluation time in 50-class classification.

Figures 19–21 correspond to the binary, 8-class, and 50-class classification tasks, respectively. Each figure presents three stacked bar plots (green) per model, showing memory usage during training and evaluation with the original feature set, PCA-selected features, and the proposed feature selection method. In addition, three stacked bar plots (purple) per model report the corresponding model sizes under the same feature configurations. The charts employ dual y-axes: the left axis indicates memory consumption (MB), while the right axis represents model size (MB).

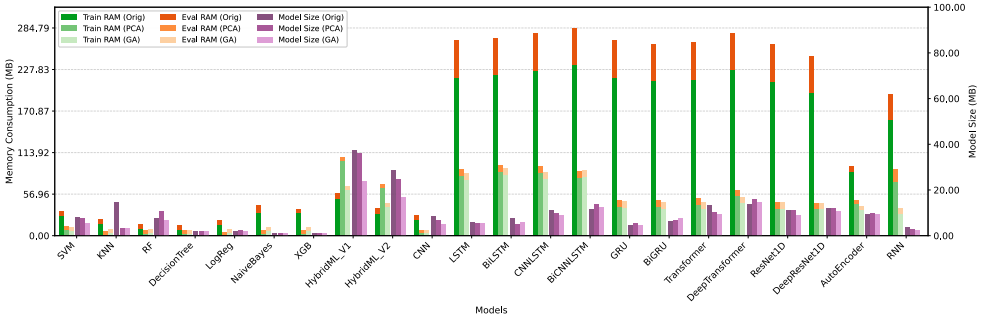

Figure 19. Impact of feature selection on resource usage in binary classification.

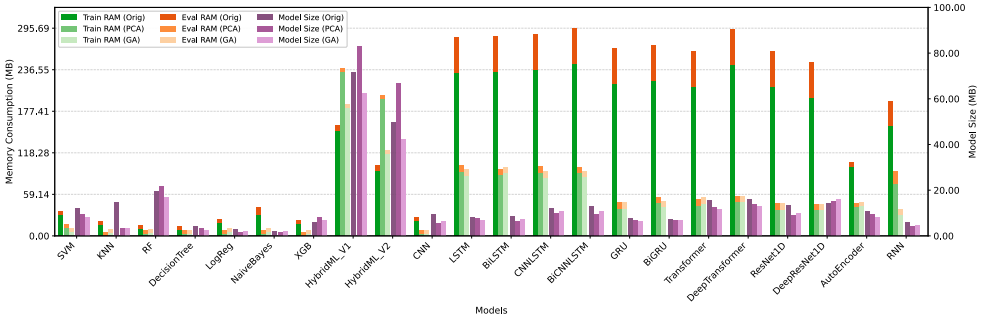

Figure 20. Impact of feature selection on resource usage in eight-class classification.

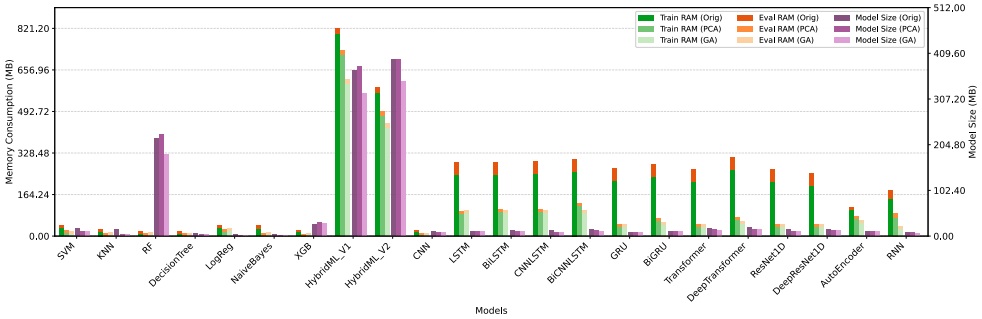

Figure 21. Impact of feature selection on resource usage in 50-class classification.

Discussion

The effectiveness of the proposed feature selection approach in enhancing resource efficiency is demonstrated across binary, 8-class, and 50-class classification scenarios (Figures 16–21, Table 8). The evaluation focused on execution time, memory usage, and model size during training and evaluation.

In the binary classification setting, several models achieved substantial reductions, with *BiLSTM* and *SVM* reducing training time by about 65%, and models such as *ResNet1D*, *naive Bayes*, and *GRU* lowering memory usage by over 80%. Similar patterns were observed in the eight-class task, where classical ML models (*SVM*, *RF*, *KNN*) and DL models (*CNN*, *LSTM*, *transformer*) consistently reduced execution time by 25–40% and memory consumption by more than 70%. In the more demanding 50-class scenario, the proposed method proved especially effective: traditional ML models such as *RF* and *SVM* reduced training and evaluation time by over 40%, while DL models including *BiLSTM*, *CNN*, and *deep transformer* achieved average memory savings exceeding 74%.

Beyond runtime efficiency, the proposed approach also reduced model sizes, outperforming PCA and yielding up to 25% size reduction compared to the original feature set. This improvement is particularly critical for real-time Industrial IoT applications, where memory and storage constraints directly affect system scalability and responsiveness.

Overall, across all models and scenarios, the proposed method achieved average reductions of 26% in training time, 35% in evaluation time, 73% in training memory usage, 44% in evaluation memory usage, and significant model size compression. These results confirm that the method not only maintains competitive detection accuracy but also enables scalable, resource-efficient, and real-time deployment in IIoT environments.

5.5. Aggregated Confusion Matrix Analysis Across Models

To assess classification performance across multiple models, we computed normalized confusion matrices for each of the 22 evaluated classifiers in the 8-class scenario. Each matrix was row-normalized so that entries represent proportions, rather than raw counts, allowing fair comparison across classes despite possible imbalance in sample sizes.

From these matrices, two key quantities were derived: (i) the average true-classified score (diagonal entries), reflecting how often samples from a class were correctly recognized, and (ii) the average misclassification rate (off-diagonal entries), indicating how frequently that class was confused with others. These values were then aggregated across all models to provide a global view of class-wise difficulty. The mean and standard deviation of misclassification rates summarize both the central tendency and variability of errors among models.

We constructed an aggregate confusion matrix by averaging the normalized matrices across all models. This matrix highlights the most common misclassification patterns, such as DoS samples mislabeled as DDoS. To improve interpretability, each off-diagonal cell also lists the top models that most frequently produced that error, while each diagonal cell identifies the models that most successfully detected the class. In the visualization, the bold value in each cell represents the average normalized score, and the subscripted model names indicate the leading contributors, either to correct detection (diagonals) or to misclassification (off-diagonals).

The aggregate results (Figure 22) show that Web (0.94), Malware (0.93), Recon (0.93), and Benign (0.93) traffic are reliably detected, whereas Bruteforce (0.58) and MITM (0.75) remain problematic. Notably, Bruteforce samples are frequently confused with MITM (0.11) and Web (0.09) traffic, suggesting shared traffic signatures. Similarly, DoS is often mislabeled as DDoS (0.11), reflecting the well-known similarity between flooding attacks at different scales.

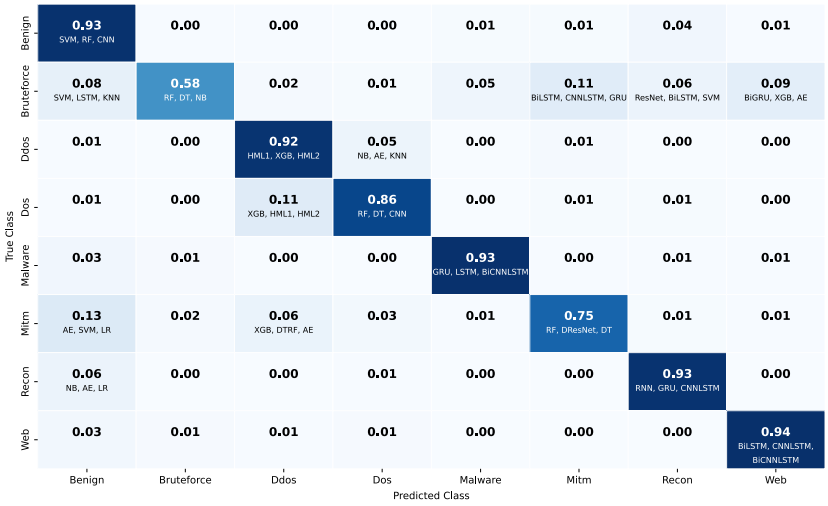

Figure 22. Aggregated confusion matrix showing misclassifications accross different models in eight-class scenario.

Overall, this analysis highlights both the intrinsically challenging classes (brute-force, MITM, DoS vs. DDoS) and the stable classes (web, malware, recon, benign). The subscripted model names in the visualization further point to which model families are most responsible for these confusions, enabling targeted improvements in feature engineering and algorithm selection.

5.6. Quantitative Comparison and Ranking of Benchmark Datasets

A comparative analysis of existing IIoT security datasets (Table 9) reveals major gaps in attack coverage, device diversity, and evaluation features. Most benchmarks provide only network-layer traffic and cover a limited range of attacks. In contrast, our dataset spans seven attack groups with 50 realistic scenarios, offering broader and more representative coverage for anomaly detection. Unlike prior datasets that rely on synthetic traffic, coarse temporal granularity, or small-scale setups, DataSense provides high-resolution, time-synchronized network and sensor data collected from a realistic IIoT testbed, ensuring both authenticity and reproducibility. It is also the first dataset to include CPU and memory utilization profiling, offering insights into the operational costs of attacks and detection methods. Finally, DataSense contributes a heterogeneous mix of IoT and IIoT devices and introduces a novel feature selection framework, validated through extensive ML/DL evaluations. Collectively, these features overcome the limitations of previous datasets and establish DataSense as a comprehensive benchmark for IIoT anomaly detection.

Table 9. Comparison of IoT/IIoT datasets based on key evaluation criteria.

Dataset	Network + Sensor Data	Attack Diversity	# Attacks Type	# Attacks Cat.	Device Diversity	No. Device	ML/DL Eval.	Feature Sel.	Resource Util.	Rank
WUSTL-IIoT	-	-	4	4	-	8	-	-	-	30.5
IoT-23	-	-	10	2	-	4	✓	-	-	38.1
MQTT-IoT-IDS	-	-	8	4	✓	15	✓	-	-	46.6
N-BaIoT	-	-	10	2	-	9	✓	✓	-	51.8
MQTTset	-	-	5	5	-	10	✓	✓	-	53.7
BoT-IoT	-	-	10	4	-	7	✓	✓	-	53.9
ToN_IoT	✓	-	9	6	-	10	✓	-	-	62.4
X-IIoTID	✓	✓	18	9	✓	15	✓	-	-	70.0
CICIoT2023	-	✓	33	7	✓	105	✓	✓	-	73.0
Edge-IIoTset	✓	-	14	4	-	10	✓	✓	-	74.4
DataSense	✓	✓	49	7	✓	40	✓	✓	✓	**98.5**

Legend: # Attacks Cat. = Attack Categories; No. Devices = Number of Devices; ML/DL Eval. = Machine Learning/Deep Learning Evaluation; Feature Sel. = Feature Selection; Resource Util. = Resource Utilization. ✓in *Attack Diversity* correspond to datasets with >15 attack types; ✓in *Device Diversity* correspond to datasets with >15 devices.

5.6.1. Ranking Approach

To further quantify these differences, we applied a *weighted scoring scheme* across seven criteria: (i) Network + Sensor Data, (ii) Number of Attack Categories, (iii) Number of Attack Types, (iv) Device Coverage, (v) ML/DL Evaluation, (vi) Feature Selection, and (vii) Resource Utilization. Binary attributes were scored as 1 (yes) or 0 (No), while quantitative attributes (attack categories, attack types, and device coverage) were log-normalized to the range of $[0, 1]$. This approach prevents datasets with unusually large values from dominating the comparison while still preserving their superiority.

Each criterion was assigned a weight to reflect its importance for IIoT security research: 15%, 15%, 20%, 20%, 10%, 10%, and 10%, respectively. The final dataset score was computed as the weighted sum:

$$\text{Score}(D) = \sum_{i=1}^{7} w_i \cdot x_i(D), \tag{13}$$

where w_i is the weight of criterion i, and $x_i(D)$ is the normalized score of dataset D for that criterion. The results were then scaled to a 0–100 range.

5.6.2. Results and Discussion

The weighted ranking outcomes are shown in Table 9. The proposed DataSense dataset achieves the highest score (98.5/100) , confirming its comprehensive coverage across all evaluation dimensions. Among the existing datasets, Edge-IIoTset (74.4), CICIoT2023 (73.0), and X-IIoTID (70.0) rank highest due to their inclusion of network data, broader attack coverage, and moderate device diversity. ToN_IoT (62.4) follows, while mid-tier benchmarks such as BoT-IoT (53.9), MQTTset (53.7), and N-BaIoT (51.8) remain limited in scope. Smaller-scale datasets, including IoT-23 (38.1), MQTT-IoT-IDS (46.6), and WUSTL-IIoT (30.5), show the narrowest applicability.

Compared with recent benchmarks such as CICIoT2023, X-IIoTID, and Edge-IIoTset, the proposed DataSense dataset provides several unique advantages that advance the state of the art. While CICIoT2023 includes a diverse collection of IoT devices, it does not incorporate industrial IIoT devices or sensor-level data, which are essential for capturing the physical processes of industrial environments. Similarly, none of the existing datasets provide synchronized network and sensor data that can be used to generate realistic streaming workloads, whereas DataSense offers fully timestamped benign and attack traces that enable reproducible real-time simulations. By providing raw synchronized sensor and network data, DataSense also allows researchers to extract additional features tailored to their studies and to extend the set of features already included in the dataset. Edge-IIoTset, although valuable, is limited in scale and device diversity, making it less representative of actual industrial deployments. Moreover, current benchmarks do not provide resource utilization measurements, and only a few offer guidance on relevant features for anomaly detection. In contrast, DataSense integrates resource profiling (CPU, memory, and model size) alongside a novel feature selection framework, thereby offering both the data and the analytical tools necessary for evaluating and designing efficient IIoT anomaly detection systems. The complete list of attacks in all compared datasets are provided in Table 10.

Table 10. Attack coverage comparison across IoT/IIoT datasets.

	Attack	Edge-IIoTset	X-IIoTID	WUSTL-IIoT	IoT-23	BoT-IoT	ToN_IoT	MQTTset	N-BaIoT	MQTT-IoT-IDS	DataSense
Recon	Host Discovery-TCP Ack Ping	-	-	-	-	-	-	-	-	-	✓
	Host Discovery-TCP Syn Stealth	-	-	-	-	-	-	-	-	-	✓
	Host Discovery-ARP Ping	-	-	-	-	-	-	-	-	-	✓
	Host Discovery-UDP Ping	-	-	-	-	-	-	-	-	-	✓
	Host Discovery-TCP Syn Ping	-	✓	-	-	-	-	-	-	-	✓
	Port Scan	✓	✓	✓	-	✓	✓	-	✓	✓	✓
	Vulnerability Scan	✓	✓	-	-	-	-	-	✓	✓	✓
	Ping Sweep	-	-	-	-	-	-	-	-	-	✓
	OS Scan	✓	✓	-	-	✓	-	-	-	✓	✓
DoS	TCP Syn Flood	✓	-	✓	-	-	-	✓	✓	✓	✓
	SynonymousIP Flood	-	-	-	-	-	-	-	-	-	✓
	Slowloris	-	-	-	-	-	-	-	-	-	✓
	UDP Fragmentation Flood	-	-	-	-	-	-	-	-	-	✓
	RST Fin Flood	-	-	-	-	-	-	-	-	-	✓
	ICMP Flood	✓	-	-	-	-	-	-	-	-	✓
	UDP Flood	✓	-	-	-	✓	-	-	✓	-	✓
	HTTP Flood	✓	-	-	-	✓	-	-	✓	-	✓
	Push Ack Flood	-	-	-	-	-	-	-	-	-	✓
	Ack Fragmentation Flood	-	-	-	-	-	-	-	-	-	✓
	ICMP Fragmentation Flood	-	-	-	-	-	-	-	-	-	✓
	TCP Flood	-	-	-	-	✓	✓	-	✓	-	✓
	MQTT Connect Flood	-	-	-	-	-	-	-	-	-	✓
	MQTT Publish Flood	-	-	-	-	-	-	✓	-	-	✓
DDoS	UDP Flood	✓	-	-	-	✓	-	-	✓	-	✓
	HTTP Flood	✓	-	-	-	✓	-	-	✓	-	✓
	Slowloris	-	-	-	-	-	-	-	-	-	✓
	Push Ack Flood	-	-	-	-	-	-	-	-	-	✓
	TCP Flood	-	-	-	-	✓	✓	-	✓	-	✓
	Synonymousip Flood	-	-	-	-	-	-	-	-	-	✓
	UDP Fragmentation Flood	-	-	-	-	-	-	-	-	-	✓
	Ack Fragmentation Flood	-	-	-	-	-	-	-	-	-	✓
	TCP Syn Flood	✓	-	-	-	-	-	-	✓	✓	✓
	MQTT Publish Flood	-	-	-	-	-	-	-	-	-	✓
	MQTT Connect Flood	-	-	-	-	-	-	-	-	-	✓
	ICMP Fragmentation Flood	-	-	-	-	-	-	-	-	-	✓
	ICMP Flood	✓	-	-	-	-	-	-	-	-	✓
	RST Fin Flood	-	-	-	-	-	-	-	-	-	✓
MitM	IP Spoofing	-	-	-	-	-	-	-	-	-	✓
	ARP Spoofing	✓	✓	-	-	-	✓	-	-	✓	✓
	Impersonation	-	-	-	-	-	-	-	-	-	✓
BruteForce	Dictionary Telnet	-	-	-	-	-	-	✓	-	-	✓
	Dictionary SSH	-	✓	-	-	-	✓	-	-	✓	✓
Web	XSS	✓	-	-	-	-	✓	-	-	-	✓
	Command Injection	-	-	✓	-	-	✓	-	-	-	✓
	SQL Injection	✓	-	-	-	-	-	-	-	-	✓
	SQL Injection Blind	-	-	-	-	-	-	-	-	-	✓
	Backdoor Upload	✓	✓	✓	-	-	✓	-	-	-	✓
Malware	Mirai Syn Flood	✓	✓	-	✓	-	-	-	✓	-	✓
	Mirai UDP Flood	-	-	-	✓	-	-	-	✓	-	✓

6. Limitations and Future Directions

While the DataSense dataset provides a diverse and comprehensive benchmark for IIoT anomaly detection, certain limitations should be acknowledged. First, although the dataset covers a broad range of devices and attack scenarios, its scalability to very large-scale industrial deployments may require further validation. As part of our future work, we plan to design a hybrid lightweight detection system tailored for large-scale IIoT setups, ensuring that the dataset can be effectively leveraged in such environments. Second, while efforts were made to ensure representativeness, the dataset may not fully capture all operational nuances of large industrial systems, where workload patterns, traffic volumes, and device diversity can be more complex. Finally, the rapidly evolving nature of cyber threats means that new attack vectors may emerge that are not yet represented in the dataset. These limitations highlight opportunities for future extensions of DataSense, including expanding the range of industrial devices, incorporating more recent and sophisticated attack campaigns, and continuously updating the dataset to reflect the evolving IIoT threat landscape.

7. Conclusions

The increasing connectivity of Industrial Internet of Things (IIoT) environments offers significant operational benefits but simultaneously exposes critical infrastructure to an expanding range of cyber threats. Developing realistic and representative datasets is, therefore, essential to advancing the research and development of robust anomaly detection and intrusion prevention systems in these domains. In this work, we presented DataSense, a comprehensive IIoT dataset generated from a meticulously designed and implemented testbed comprising 40 heterogeneous devices, including a diverse range of industrial sensors, IoT devices, and network equipment. This diversity enhances the realism of the dataset and improves its relevance for industrial use cases. The testbed enabled the execution of 50 distinct and realistic attack scenarios spanning several categories, including reconnaissance, denial-of-service (DoS), distributed denial-of-service (DDoS), man-in-the-middle (MitM), web exploitation, brute-force intrusions, and malware infections, thereby closely simulating operational threat conditions. These attacks provide essential data for developing and evaluating effective countermeasures to secure both IoT and IIoT environments. We also developed an integrated data pipeline that captures and processes both network traffic and sensor-logged data, supporting real-time stream processing for prompt threat analysis. In addition, we introduced a new feature-selection approach that pinpoints the features most strongly correlated with the target variables, thereby enhancing the precision of attack detection and classification. We further compared its effectiveness against established methods such as PCA, and validated performance across a wide spectrum of machine learning, deep learning, and hybrid ML/DL techniques. The DataSense dataset is publicly accessible through the CIC Dataset portal (https://www.unb.ca/cic/datasets/iiot-dataset-2025.html, accessed on 26 September 2025), offering researchers and industry specialists a robust resource for developing and evaluating advanced IIoT security mechanisms.

Author Contributions: Conceptualization, A.F., S.D. and A.A.G.; methodology, A.F., S.D. and A.A.G.; software, A.F. and S.A.M.; validation, A.F., S.D., S.A.M. and A.A.G.; formal analysis, A.F. and A.A.G.; investigation, A.F. and S.A.M.; resources, A.F., S.D. and A.A.G.; data curation, A.F. and S.A.M.; writing—original draft preparation, A.F.; writing—review and editing, A.F., S.D., A.A.G. and S.A.M.; visualization, A.F.; supervision, S.D. and A.A.G.; project administration, S.D. and A.A.G.; funding acquisition, A.A.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The DataSense dataset is publicly accessible through the CIC Dataset portal (https://www.unb.ca/cic/datasets/iiot-dataset-2025.html, accessed on 18 October 2025).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chataut, R.; Phoummalayvane, A.; Akl, R. Unleashing the Power of IoT: A Comprehensive Review of IoT Applications and Future Prospects in Healthcare, Agriculture, Smart Homes, Smart Cities, and Industry 4.0. *Sensors* **2023**, *23*, 7194. [CrossRef] [PubMed]
2. Alotaibi, B. A Survey on Industrial Internet of Things Security: Requirements, Attacks, AI-Based Solutions, and Edge Computing Opportunities. *Sensors* **2023**, *23*, 7470. [CrossRef]
3. Nuaimi, M.; Fourati, L.C.; Hamed, B.B. Intelligent approaches toward intrusion detection systems for Industrial Internet of Things: A systematic comprehensive review. *J. Netw. Comput. Appl.* **2023**, *215*, 103637. [CrossRef]
4. Mengistu, T.M.; Kim, T.; Lin, J.W. A Survey on Heterogeneity Taxonomy, Security and Privacy Preservation in the Integration of IoT, Wireless Sensor Networks and Federated Learning. *Sensors* **2024**, *24*, 968. [CrossRef]
5. Anjum, N.; Latif, Z.; Chen, H. Security and privacy of industrial big data: Motivation, opportunities, and challenges. *J. Netw. Comput. Appl.* **2025**, *237*, 104130. [CrossRef]
6. Yang, X.; Tong, F.; Jiang, F.; Cheng, G. A Lightweight and Dynamic Open-Set Intrusion Detection for Industrial Internet of Things. *IEEE Trans. Inf. Forensics Secur.* **2025**, *20*, 2930–2943. [CrossRef]
7. Savaglio, C.; Mazzei, P.; Fortino, G. Edge Intelligence for Industrial IoT: Opportunities and Limitations. *Procedia Comput. Sci.* **2024**, *232*, 397–405. [CrossRef]
8. Andriulo, F.C.; Fiore, M.; Mongiello, M.; Traversa, E.; Zizzo, V. Edge Computing and Cloud Computing for Internet of Things: A Review. *Informatics* **2024**, *11*, 71. [CrossRef]
9. Holdbrook, R.; Odeyomi, O.; Yi, S.; Roy, K. Network-Based Intrusion Detection for Industrial and Robotics Systems: A Comprehensive Survey. *Electronics* **2024**, *13*, 4440. [CrossRef]
10. Kong, X.; Song, Z.; Ye, X.; Jiao, J.; Qi, H.; Liu, X. GRID: Graph-Based Robust Intrusion Detection Solution for Industrial IoT Networks. *IEEE Internet Things J.* **2025**, *12*, 26646–26659. [CrossRef]
11. Zhang, S.; Xu, Y.; Xie, X. Universal Adversarial Perturbations Against Machine-Learning-Based Intrusion Detection Systems in Industrial Internet of Things. *IEEE Internet Things J.* **2025**, *12*, 1867–1889. [CrossRef]
12. Ahmed, M.R.; Zhang, M. Context-Aware Intrusion Detection in Industrial Control Systems. In Proceedings of the 2024 Workshop on Re-Design Industrial Control Systems with Security, New York, NY, USA, 6 May 2024; RICSS: London, UK, 2024 ; pp. 1–7. [CrossRef]
13. Sheng, C.; Zhou, W.; Han, Q.L.; Ma, W.; Zhu, X.; Wen, S.; Xiang, Y. Network Traffic Fingerprinting for IIoT Device Identification: A Survey. *IEEE Trans. Ind. Inform.* **2025**, *21*, 3541–3554. [CrossRef]
14. Al-Hawawreh, M.; Sitnikova, E.; Aboutorab, N. X-IIoTID: A Connectivity-Agnostic and Device-Agnostic Intrusion Data Set for Industrial Internet of Things. *IEEE Internet Things J.* **2022**, *9*, 3962–3977. [CrossRef]
15. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In Proceedings of the International Conference on Information Systems Security and Privacy, Madeira, Portugal, 22–24 January 2018.
16. Neto, E.C.P.; Dadkhah, S.; Ferreira, R.; Zohourian, A.; Lu, R.; Ghorbani, A.A. CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment. *Sensors* **2023**, *23*, 5941. [CrossRef]
17. Ferrag, M.A.; Friha, O.; Hamouda, D.; Maglaras, L.; Janicke, H. Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning. *IEEE Access* **2022**, *10*, 40281–40306. [CrossRef]
18. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A Detailed Analysis of the KDD-CUP 99 Data Set,. In Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009. Available online: https://ieeexplore.ieee.org/document/5356528 (accessed on 27 August 2024).
19. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, Australia, 10–12 November 2015; pp. 1–6. [CrossRef]
20. Alsaedi, A.; Moustafa, N.; Tari, Z.; Mahmood, A.; Anwar, A. TON_IoT Telemetry Dataset: A New Generation Dataset of IoT and IIoT for Data-Driven Intrusion Detection Systems. *IEEE Access* **2020**, *8*, 165130–165150. [CrossRef]
21. Vaccari, I.; Chiola, G.; Aiello, M.; Mongelli, M.; Cambiaso, E. MQTTset, a New Dataset for Machine Learning Techniques on MQTT. *Sensors* **2020**, *20*, 6578. [CrossRef]

22. Omotosho, A.; Qendah, Y.; Hammer, C. IDS-MA: Intrusion Detection System for IoT MQTT Attacks Using Centralized and Federated Learning. In Proceedings of the 2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC), Torino, Italy, 26–30 June 2023; pp. 678–688. [CrossRef]
23. Lippmann, R.; Haines, J.W.; Fried, D.J.; Korba, J.; Das, K. The 1999 DARPA Off-Line Intrusion Detection Evaluation. MIT Lincoln Laboratory, 2000. Available online: https://www.ll.mit.edu/r-d/datasets/1999-darpa-intrusion-detection-evaluation-dataset (accessed on 27 August 2024).
24. Ghiasvand, E.; Ray, S.; Iqbal, S.; Dadkhah, S.; Ghorbani, A.A. CICAPT-IIOT: A provenance-based apt attack dataset for IIOT environment. *arXiv*, **2024**, arXiv:2407.11278. 2024.
25. Zolanvari, M.; Teixeira, M.A.; Gupta, L.; Jain, R. *WUSTL-IIOT-2021 Dataset for IIoT Cybersecurity Research*; Washington University: St. Louis, MO, USA, 2021.
26. Myneni, S.; Chowdhary, A.; Sabur, A.; Sengupta, S.; Agrawal, G.; Huang, D.; Kang, M. *DAPT 2020—Constructing a Benchmark Dataset for Advanced Persistent Threats*; Springer International Publishing: Cham, Switzerland, 2020; pp. 138–163. [CrossRef]
27. Garcia, S.; Parmisano, A.; Erquiaga, M.J. IoT-23: A Labeled Dataset with Malicious and Benign IoT Network Traffic, 2020, [Dataset]. Available online: https://doi.org/10.5281/zenodo.4743746 (accessed on 17 November 2024)
28. Meidan, Y.; Bohadana, M.; Mathov, Y.; Mirsky, Y.; Shabtai, A.; Breitenbacher, D.; Elovici, Y. N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders. *IEEE Pervasive Comput.* **2018**, *17*, 12–22. [CrossRef]
29. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B. Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset. *Future Gener. Comput. Syst.* **2019**, *100*, 779–796. [CrossRef]
30. Mathur, A.P.; Tippenhauer, N.O. SWaT: A water treatment testbed for research and training on ICS security. In Proceedings of the 2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater), Vienna, Austria, 11 April 2016; pp. 31–36. [CrossRef]
31. Ahmed, C.; Palleti, V.; Mathur, A. WADI: A water distribution testbed for research in the design of secure cyber physical systems. In Proceedings of the 3rd international workshop on cyber-physical systems for smart water networks, Pittsburgh, PA, USA, 21 April 2017; pp. 25–28. [CrossRef]
32. C-MAPSS Aircraft Engine Simulator Data, 2008. Available online: https://data.nasa.gov/dataset/c-mapss-aircraft-engine-simulator-data (accessed on 27 August 2024).
33. Su, Y.; Zhao, Y.; Niu, C.; Liu, R.; Sun, W.; Pei, D. Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, New York, NY, USA, 4–8 August 2019; pp. 2828–2837. [CrossRef]
34. McCann, M., Johnston, A. SECOM [Dataset]. UCI Machine Learning Repository, 2008. Available online: https://doi.org/10.24432/C54305 (accessed on 27 August 2024).
35. Scania CV AB, APS Failure at Scania Trucks [Dataset], UCI Machine Learning Repository, 2016. Available online: https://archive.ics.uci.edu/ml/datasets/APS+Failure+at+Scania+Trucks (accessed on 27 August 2024).
36. Purohit, H.; Tanabe, R.; Ichige, K.; Endo, T.; Nikaido, Y.; Suefusa, K.; Kawaguchi, Y. MIMII Dataset: Sound Dataset for Malfunctioning Industrial Machine Investigation and Inspection. *arXiv* **2019**, arXiv:1909.09347. [CrossRef]
37. Gas Sensor Array Drift Dataset. Available online: https://archive.ics.uci.edu/dataset/224/gas+sensor+array+drift+dataset (accessed on 27 August 2024).
38. Intel Lab Data. 2004. Available online: https://kaggle.com/datasets/caesarlupum/iot-sensordata (accessed on 27 August 2024)).
39. Gao, J.; Giri, S.; Kara, E.C.; Bergés, M. PLAID: A public dataset of high-resoultion electrical appliance measurements for load identification research: Demo abstract. In Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings, New York, NY, USA, 5–6 November 2014; pp. 198–199. [CrossRef]
40. Abdulaal, A.; Liu, Z.; Lancewicki, T. Practical Approach to Asynchronous Multivariate Time Series Anomaly Detection and Localization. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, New York, NY, USA, 14 August 2021; pp. 2485–2494. [CrossRef]
41. Ranjan, C.; Reddy, M.; Mustonen, M.; Paynabar, K.; Pourak, K. Dataset: Rare event classification in Multivariate Time Series. *arXiv* **2019**, arXiv:1809.10717. [CrossRef]
42. Statlog (Shuttle) Dataset, UCI Machine Learning Repository, 1999. Available online: https://archive.ics.uci.edu/ml/datasets/Statlog+(Shuttle) (accessed on 27 August 2024).
43. Chen, Y. Keogh, E. ECG5000 Dataset [Dataset], UCR Time Series Classification Archive, 2015. Available online: https://www.cs.ucr.edu/~eamonn/time_series_data_2018/ (accessed on 27 August 2024).
44. Eclipse Mosquitto. Available online: https://mosquitto.org/ (accessed on 26 August 2024).
45. Filebeat. Available online: https://www.elastic.co/beats/filebeat (accessed on 26 August 2024).
46. TShark. Available online: https://www.wireshark.org/docs/man-pages/tshark.html (accessed on 26 August 2024).
47. Wireshark Foundation. *Wireshark: Network Protocol Analyzer*; Wireshark Foundation: Davis, CA, USA, 2024. Available online: https://www.wireshark.org (accessed on 26 August 2024).

48. Linux, K. hping3: TCP/IP Packet Assembler and Analyzer. 2019. Available online: https://www.kali.org/tools/hping3 (accessed on 19 June 2023).
49. Leeon123. Golang HTTP Flood Tool. 2020. Available online: https://github.com/Leeon123/golang-httpflood (accessed on 19 June 2023).
50. Krylovsk. MQTT Benchmark: MQTT Broker Benchmarking Tool. 2023. Available online: https://github.com/krylovsk/mqtt-benchmark (accessed on 19 June 2023).
51. Yaltirakli, G. Slowloris: Low Bandwidth DoS Tool. 2015. Available online: https://github.com/gkbrk/slowloris (accessed on 19 June 2023).
52. EPC-MSU. UDP Flood Attack Script. 2023. Available online: https://github.com/EPC-MSU/udp-flood (accessed on 19 June 2023).
53. Lyon, G. Nmap: Network Mapper. 2014. Available online: http://nmap.org/ (accessed on 22 June 2023).
54. Linux, K. fping: Parallel Ping for Network Scanning. 2023. Available online: https://fping.org/ (accessed on 19 June 2023).
55. Security, S. Vulscan: Vulnerability Scanning NSE Script for Nmap. 2023. Available online: https://github.com/scipag/vulscan (accessed on 19 June 2023).
56. Makiraid. Remot3d: PHP Remote Shell Backdoor. 2023. Available online: https://github.com/makiraid/Remot3d (accessed on 19 June 2023).
57. Payloadbox. Command Injection Payload List. 2023. Available online: https://github.com/payloadbox/command-injection-payload-list (accessed on 19 June 2023).
58. Project, S. sqlmap: Automatic SQL Injection and Database Takeover Tool. 2023. Available online: https://github.com/sqlmapproject/sqlmap (accessed on 19 June 2023).
59. Payloadbox. SQL Injection Payload List. 2023. Available online: https://github.com/payloadbox/sql-injection-payload-list (accessed on 19 June 2023).
60. Payloadbox. XSS Payload List. 2023. Available online: https://github.com/payloadbox/xss-payload-list (accessed on 19 June 2023).
61. S0md3v. XSStrike: Advanced XSS Detection Suite. 2023. Available online: https://github.com/s0md3v/XSStrike (accessed on 19 June 2023).
62. Miessler, D. SecLists: Password Collections for Security Testing. 2023. Available online: https://github.com/danielmiessler/SecLists/blob/master/Passwords (accessed on 19 June 2023).
63. Hauser, V. THC-Hydra: Network Logon Cracker. 2023. Available online: https://github.com/vanhauser-thc/thc-hydra (accessed on 19 June 2023).
64. Ornaghi, A.; Valleri, M. Ettercap: Comprehensive Suite for MITM Attacks. 2005. Available online: https://www.ettercap-project.org/ (accessed on 19 June 2023).
65. Gamblin, J. Mirai BotNet: Leaked Mirai Source Code for Research/IOC Development Purposes. Available online: https://github.com/jgamblin/Mirai-Source-Code (accessed on 1 June 2023).
66. Meier, L.; van de Geer, S.; Bühlmann, P. The Group Lasso for Logistic Regression. *J. R. Stat. Soc. Series B* **2008**, *70*, 53–71. [CrossRef]
67. Simon, N.; Friedman, J.; Hastie, T.; Tibshirani, R. A Sparse-Group Lasso. *J. Comput. Graph. Stat.* **2013**, *22*, 231–245. [CrossRef]
68. Brown, G.; Pocock, A.; Zhao, M.J.; Luján, M. Conditional likelihood maximisation: A unifying framework for information theoretic feature selection. *J. Mach. Learn. Res.* **2012**, *13*, 27–66.
69. Hamdani, T.M.; Won, J.-M.; Alimi, A.M.; Karray, F. Multi-objective Feature Selection with NSGA-II. In Proceedings of the EvoWorkshops (EvoCOP, EvoBIO, EvoWorkshops), Valencia, Spain, 11–13 April 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 240–250.
70. Ding, C.; Peng, H. Minimum Redundancy Feature Selection from Microarray Gene Expression Data. *J. Bioinform. Comput. Biol.* **2005**, *3*, 185–205. [CrossRef]
71. McGill, W.J. Multivariate Information Transmission. *Psychometrika* **1954**, *19*, 97–116. [CrossRef]
72. Wollstadt, P.; Nili, H.; Lizier, J.T.; Vicente, R. A Rigorous Information-Theoretic Definition of Redundancy and Synergy in Feature Selection. *J. Mach. Learn. Res.* **2023**, *24*, 1–69.